Imam Muhammad Ibn Saud Islamic University
College of Computer and Information Sciences
Department of Computer Science

Sentiment Analysis Using Stacked Gated Recurrent Unit for Arabic Tweets

تحليل الآراء باستخدام وحدة التكرار المبوبة للتغريدات باللغة العربية

Submitted to the Department of Computer Science in partial fulfillment of the requirements for the degree of Master of Science in Computer Science

By

Name: Asma Ali Nasser Al Wazrah

ID: 438021678

Supervised by

Dr. Sarah Omar Alhumoud

Submission date

Nov 2020

Al-Imam Muhammad Ibn Saud Islamic University
College of Computer and Information Sciences
Department of Computer Science

Sentiment Analysis Using Stacked Gated Recurrent Unit for Arabic Tweets

Supervisor:

Approved by	**Signature**	**Date of Approval**
Dr. Sarah Alhumoud		

Thesis Discussion Committee:

Approved by	**Signature**	**Date of Approval**

Declaration

I, Asma Al Wazrah, hereby declare that all information in this document has been obtained and presented following academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name: Asma Al Wazrah

Signature:

Acknowledgment

First of all, I thank God for giving support and providing me with patience and guidance, which I used in making this work. I would like to express great gratitude to my supervisor Dr. Sarah Omar Alhumoud for providing her valuable time to express her professional thoughts. I would like to thank her for her help, continuous encouragement, fertile discussion, and valuable suggestions and comments throughout the research and the thesis work. Thank you for pushing me to be a better researcher. I owe more than thanks to my family members especially, my mother, for everything that I have reached. Very thanks to my lovely sister, Nada, for her support, love, and caring of my children during the busy days. Many thanks to my husband who supports me a lot through this journey. I would like to thank the love of my life, my children Ziyad and Jannah for their patience with my preoccupation. I love you very much and I dedicate this thesis to you.

Abstract

Arabic content generated on the internet in websites and social media platforms increased vastly in the last decade. In social media specifically, people express their opinions openly and freely offering a rich source for analyzing trends and opinions. Those opinions could be mined for valuable indicators to enhance products or services leading to an attempt to improve them as an Arabic Natural language processing (NLP). Recently, using deep learning as a powerful machine learning tool for analyzing these opinions became popular because of its accuracy in predicting unstructured data. Unlike English, Arabic has several specifics that complicate processing and analyzing it by traditional methods. I propose a neural-based model, Stacked Gated Recurrent Unit with word embedding for mining Arabic opinions effectively and accurately using a deep learning method instead of traditional machine learning methods. The gulf dialect tweets are first preprocessed and then each word is transformed into a vector using AraVec to train the tweets dataset. Then, using the Gated Recurrent Unit (GRU), Stacked Gated Recurrent Unit (SGRU), Stacked Bidirectional Gated Recurrent Unit model (SBi-GRU), to classify the resulted vectors. After discovering the performance of these models, I compare them with machine learning model such as Support Vector Machine (SVM). Moreover, I compare the mentioned models with recent pre-trained Arabic Bidirectional Encoder Representations from Transformers (AraBERT). Also, an Ensemble of different architectures of all mentioned models to find the best model architecture for Arabic NLP. To the best of our knowledge, until now, no studies have applied unidirectional nor bidirectional SGRU for Arabic sentiment classification. Also, no ensemble models have been implemented from mentioned architectures for the Arabic language. The results show that the 6-layer SGRU stacking and 5-layer SBi-GRU stacking score the highest results in terms of accuracy. The ensemble method outperforms all models' results alone with an accuracy exceeding 90%. Hence, I conclude that the ensemble model contains GRU and the transformers in this sentiment analysis task outperforms the singular models of SVM, GRU and transformers separately.

Abstract (ملخص عربي)

ازداد المحتوى العربي الموجود في شبكة الإنترنت، في المواقع الإلكترونية وفي منصات التواصل الاجتماعي بشكل كبير خلال العقد الماضي. في وسائل التواصل الاجتماعي تحديداً، يعبر الناس عن آراءهم بكل وضوح وحرية فيعرضون مصدراً غنياً للتوجهات والآراء. يمكن استخراج هذه الآراء وتحليلها للحصول على مؤشرات قيمة لتعزيز المنتجات أو الخدمات، مما يؤدي إلى محاولة تحسينها عن طريق معالجة اللغة الطبيعية العربية. في الآونة الأخيرة، أصبح استخدام التعلم العميق كأداة قوية للتعلم الآلي لتحليل هذه الآراء شائعًا، بسبب دقته في التنبؤ بالبيانات غير المهيكلة. وعلى عكس اللغة الإنجليزية، تحتوي اللغة العربية على العديد من الخصائص التي تعقّد معالجتها وتحليلها بالطرق التقليدية. أَقترح أنموذجًا للتعلم العميق، باستخدام وحدة التكرار المبوبة المرصوصة مع تضمين الكلمة (word embedding) لتعدين الآراء العربية بفعالية وبدقة باستخدام طريقة التعلم العميق بدلاً من أساليب التعلم الآلي التقليدية. تتم معالجة التغريدات باللهجة الخليجية أولاً، ثم يتم تحويل كل كلمة إلى متجه باستخدام (AraVec) لتدريب مجموعة البيانات. ثم يتم استخدام نموذج وحدة التكرار المبوبة المرصوصة (SGRU)، نموذج وحدة التكرار المبوبة المرصوصة ثنائية الاتجاه (SBi-GRU) لتصنيف المتجهات الناتجة. بعد استخراج نتائج هذه النماذج، نقوم بمقارنتها مع نموذج تعلم الآلة مثل آلة المتجهات الداعمة (SVM). علاوة على ذلك، نقارن النماذج المذكورة مع تمثيلات ترميز عربية ثنائية الاتجاه من المحولات (AraBERT) مدربة مسبقاً. بالإضافة إلى ذلك، يتم تجميع الهياكل المختلفة المذكورة أعلاه للعثور على أفضل بنية نموذجية لمعالجة اللغة العربية. لم تقم أي دراسات سابقة بتطبيق (SGRU) أحادي الاتجاه أو ثنائي الاتجاه لتصنيف المشاعر العربية. أيضاً، لم يتم تنفيذ أي نماذج لخوارزميات مجمعة من النماذج المذكورة للغة العربية. تظهر النتائج أن تكديس (SGRU) المكون من 6 طبقات وتكديس (SBi-GRU) المكون من 5 طبقات يؤدي إلى أعلى النتائج من حيث الدقة. تظهر النتائج تفوق خوارزمية التجميع على جميع النماذج المنفردة بدقة تتجاوز 90٪. نستنتج أن نموذج التجميع الذي يحتوي على (GRU) و المحولات في مهمة تحليل المشاعر هذه تتفوق على النماذج الفردية لـ (SVM)، (GRU) والمحولات بشكل منفصل.

Keywords

Arabic language, AraBERT, Bidirectional GRU, Deep learning, Ensemble Method, Gated Recurrent Unit, Opinion mining, Recurrent neural network, Sentiment analysis, Sentiment classification. Stacked model, Transformers,

List of Abbreviation

AraBERT: Arabic Bidirectional Encoder Representations from Transformers.

ASTD: Arabic Sentiment Tweets Dataset.

BERT: Bidirectional Encoder Representations from Transformers.

BOW: Bag of Words.

CRF: Conditional random field.

GloVe: Global Vector.

GRU: Gated Recurrent Unit.

KNNs: k-nearest neighbors.

LABR: Large-scale Arabic Book Reviews Dataset.

LSTM: Long Short-Term Memory.

MD-ArSenTD: Multi-dialect Arabic sentiment Twitter dataset.

ME: maximum entropy.

MSA: Modern Standard Arabic.

NB: Naive Bayes.

NLP: Natural Language Processing.

RNN: Recurrent Neural Network.

SBi-GRU: Stacked Bi-directional Gated Recurrent Unit.

SG: Skip-gram.

SGRU: Stacked Gated Recurrent Unit.

SVM: Support Vector Machines.

Table of Contents:

List of Figures:

List of Tables:

CHAPTER ONE: INTRODUCTION

1.1 Introduction

The Internet's strength and reach continue to expand globally. According to the International Data Corporation, the amount of digital data generated worldwide exceeded 33 zettabytes in 2018, with a projected growth of 175 zettabytes by 2025 [1]. The number of Internet users worldwide has risen to four billion users; in particular, the number of Middle East users has increased to 164 million users [2]. In recent years, an increasing number of people have used social media to share their views or leave a positive or negative review of a specific service. As of January 2018, three billion people are on social media, and 130 million of them are Arabs [2].

Due to the influence of social media content on governing, development, diplomacy, and business, sentiment analysis is required in social media monitoring, as it allows an overview of the wider public opinion on topics appearing in a variety of posts, from politics-related posts to customer reviews. The ability to discern the sentiment and attitude behind a post on any subject facilitates strategizing and planning, thus providing better services.

Sentiment analysis is the computational assessment of people's attitudes, feelings, and emotions towards structures, persons, cases, news, or subjects. Understanding open opinions, thoughts, and questions expressed is a matter of urgency at this point, which is the reason for excitement regarding sentiment analysis. A considerable amount of research has been conducted to improve the precision of sentiment analysis, from basic linear approaches to more complex deep neural network models [3].

Machine learning techniques have been commonly used in sentiment analysis. However, these techniques have limited ability to process raw data, and feature representation greatly affects the performance of the machine learning model. For this reason, deep learning is used for feature representation at multiple levels. deep learning automatically discovers discriminative and explanatory text representations from data using nonlinear neural networks, each of which transforms the representation at one level into a representation at a higher and more abstract level [4]. deep learning has shown to be highly productive in sentiment analysis and is considered a modern multilingual model for sentiment analysis. The analysis of Arabic sentiments, however, still needs improvement. Due to its complex structure and different dialects, as well as a lack of resources, Arabic language processing faces many challenges. While current deep learning approaches have enhanced the accuracy of Arabic sentiment analysis, these approaches can still be improved [3].

A promising new research field in Arabic sentiment analysis is the application of Recurrent Neural Network (RNN) in textual models to demonstrate the learning process and measure understanding of the text at the level of semantic analysis [5].

1.2 Motivation and Contribution

The social media texts are unstructured, full of spelling mistakes, and have many peculiarities and conventions. The analysis of these texts becomes hard when conducting sentiment analysis on Arabic social media text. This is due to the limitation in the existing natural language processing tools and resources available for the Arabic language which is developed to deal with Modern Standard Arabic (MSA) only. The main challenging aspects of sentiment analysis and opinion mining exist with the use of colloquial words, merging words, repeated letters, and spelling errors, as shown in Table 1.1. A specific challenge is

encountered when attempting to deal with Arabic formal (news) and informal (sports and politics) tweets. Arabic Gulf tweets are generally characterized to be written in a highly informal Arabic language that is used in colloquial speaking. This language is subject to differences in dialects of Gulf regions and difficult to model and analyze. With the challenge of analyzing Gulf dialect tweets with a lack of tools and resources, the enhancements attempts are made on the Arabic language in general without focusing on the dialect, which motivates us to enhance the analyzing the Gulf dialect tweets with deep model structure and effective pre-processing methods.

Table 1. 1 Different Challenges of Colloquial Gulf

Challenge	**English**	**Arabic**	**Colloquial Gulf**
Abridgement	everything	كل شيء	كلش
Expression	not	ليس	مهو, مهب, مو, مب,ماهو
Repeated letters	Good job	عمل جيد	كفوووووو
Writing mistakes	this	هذا	هاذا
Cutting words	in	في	ف
Antonym	great	رائع	يهبل, يروع, يجنن

The objective of this thesis is to find techniques and suitable models to automatically determine the sentiment of tweets posted in specific domains (news, sports, and politics). It specifically aims to develop a classifier that can be used to automatically classify gulf dialect comments into positive, negative, or neutral.

The core contribution of the thesis is Proposing SGRU, and SBi-GRU and compare them with SVM and AraBERT transformer model, to investigate their performance for analyzing Arabic, with the use of effective preprocessing techniques. Additionally, the contributions are summarized as follows:

- Present a systematic review to identify recent RNN models for sentiment analysis.
- Highlight available annotated Arabic sentiment analysis datasets.
- Categorize related research studies by sentiment analysis type (emoji analysis, emotion detection, sentiment classification, and hate speech detection). Also, based on the level (sentence, document, and aspect-based).
- Highlight related research limitations and drawbacks, to find research and alternative approaches.
- Review recent studies of transformers, a novel neural network architecture that based on a self-attention mechanism.
- Implement an ensemble models from different models (SGRU, SBi-GRU, AraBERT) to generate the best-suited model for the Arabic language in the field of sentiment analysis.

Based on the proposed model, increasing the depth of the network provides an alternate solution that requires fewer neurons and trains faster. Ultimately, adding depth is a type of representational optimization.

1.3 Choosing RNN Approach for Sentiment Analysis

A promising new research field in Arabic sentiment analysis is the application of RNNs in textual models to demonstrate the learning process and measure understanding of the text at the level of semantic analysis [5].

In this study, I choose RNN approach for Arabic sentiment analysis due to the following reasons:

First, RNNs vs. machine learning models: As machine learning models rely on feature representation, the goodness of the data representation is directly proportional to the performance gains [6]. Since there is no correct way to represent these features, it is domain-specific, costly, and requires much preprocessing, which is especially true for the Arabic language with its multiple dialects. Moreover, upon analyzing the literature on Arabic sentiment analysis, I found that RNNs for Arabic sentiment analysis always outperformed machine learning models for Arabic sentiment analysis except in only one study where an SVM outperformed an RNN because of the use of rich handcrafted feature sets (morphological, N-grams, syntactic, and semantic features) [7]. Furthermore, RNNs do not require domain expertise or hardcore feature extraction, as they learn high-level features from data in an incremental manner. The main reason is due to the rich sequence-to-sequence model of RNNs.

Second, RNNs vs. other deep learning models: A related study [8] indicated that RNNs are one of the best and most important methods for text-based sentiment analysis, while CNNs have shown good results for image processing tasks such as image sentiment analysis. Based on a comparison between an RNN and a CNN for NLP [9], the CNN outperformed the RNN in specific tasks (answer selection and question relation matching), while the RNN showed promising results in the remaining NLP tasks (sentiment classification, relation classification, textual entailment, path query answering and part-of-speech tagging), especially in the sentiment classification tasks because RNNs are trained to recognize temporal patterns, while CNNs learn to recognize spatial patterns. In textual analysis, CNNs can identify special patterns of n-gram words regardless of the positions and orders of these words [9], which leads to ignoring the semantic dependency and losing information. Hence, the integration depth of CNN is not sufficient to process the language complexity. On the other hand, RNNs are inherently temporally deep, since their hidden state is a function of all previous hidden states, which means it is more suitable to capture sequential data. To enhance an RNN to capture spatial depth, I could add multiple hidden layers, which means that I could use an RNN alone to achieve spatiotemporal depth integration [10]. In general, RNNs are crucial in sentiment analysis, whether used alone or with other architectures, because they are specifically designed for text classification tasks. Despite the importance and effectiveness of RNNs in sentiment analysis for various languages, this aspect has not been studied in depth for Arabic. Therefore, in this study, RNNs are the focal point of the review to analyze and compare different studies on effective Arabic sentiment analysis.

1.4 Research Questions

RNN approach is selected due to its ability in using previous sequential states to compute the current input, which is suitable for the natural language context. In addition, when using SGRU to classify English texts it gives promising results with effective performance in

classification tasks, along with reducing the training time because it needs few iterations to update the hidden states. I investigate implementing it with gulf dialect text in terms of increasing the performance and lowering the training time cost. Moreover, after discovering the performance of the mentioned models, I compare them with machine learning model such as SVM. Moreover, the mentioned models are compared with recent pre-trained AraBERT. Moreover, an Ensemble of different models (SGRU, SBi-GRU, AraBERT) is built to find the best model architecture for Arabic NLP. Table 1.2 indicates research questions and their motivations, these questions are asked in the context of the Arabic text, in addition to the related chapters.

Table 1. 2 Research Questions with the Motivation

Definition	**Motivation**	**Related Chapter**
RQ1: What are the neural network approaches for sentiment analysis?	To present systematic background information for techniques that are used in the field of sentiment analysis.	Ch2
RQ2: What are the recurrent neural network approaches for sentiment analysis?	To explore the state-of-the-art techniques and implementations of recurrent neural networks in the field of sentiment analysis. In our study, I focus on recurrent LSTM and GRU as types of RNN.	Ch3
RQ3: What are the studies on Arabic sentiment analysis using recurrent neural networks?	To track all the latest models, techniques, and lexical resources and explore the weakness and research gaps for Arabic sentiment analysis using RNNs.	Ch3
RQ4: What are the stacked RNN models in the field of sentiment analysis?	To explore the effectiveness of stacked RNN models above others for sentiment classification.	Ch3
RQ5: How does adding layers to the GRU affect the accuracy?	To investigate the performance of multilayer GRU.	Ch5
RQ 6: How does deep learning models perform compared to machine learning models?	To compare the performance between the proposed models and a machine learning model like SVM.	Ch5
RQ 7: How do transformers create an impact on the overall accuracy of both SGRU and SBi-RU?	To investigate the performance and compare transformers with machine learning models and the proposed models.	Ch5
RQ 8: What are the performance differences between the ensemble method and singular methods?	To investigate the performance of the ensemble model in the process of handling Arabic language complexity compared to singular methods.	Ch5

1.5 Thesis Timeline

Error! Reference source not found. shows the estimated time by months for each task. The project is implemented roughly in 16 months. The timeline is shown in Table 1.3.

Table 1. 3 Thesis Timeline

	2019								2020												2021
	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec	Jan
Write Introduction																					
Write background																					
Write related works																					
Review SLR																					
Implement models																					
Models' evaluation																					
Write Results & Discussions																					
Conclusion																					

1.6 Outline

This section presented the thesis outline as follows:

Chapter 1 has introduced the thesis; it discussed the motivation and proposed the objectives and contribution with analyzing Arabic gulf dialect tweets. This chapter also presented the 8 research questions that this thesis aims to address and answer.

In Chapter 2, a background of the sentiment analysis and deep learning methods to understand recent trends in this field will be discussed.

In Chapter 3, RQ1, RQ2, RQ3, and RQ4 will be answered, respectively. I present a systematic literature review method and the strategies that were used to extract the targeted studies for both English and Arabic, besides, to discuss the related studies that use RNN models for Arabic sentiment analysis and stacked RNN models in general, in addition to present the findings from related studies under consideration.

In Chapters 4, I describe the methodology and the architecture of the proposed GRU models.

In Chapter 5, the models' evaluation based on the evaluation metrics and with the comparison with different GRU architectures with different embedding is presented to answer RQ5, RQ6, RQ7, and RQ8.

Finally, Chapter 6 provides a conclusion, the thesis challenges and future scope of our study.

1.7 Conclusion

This chapter has introduced the thesis, it discussed the importance of data and how this thesis will make an advantage out of analyzing it. This chapter also highlighted the motivation and proposed the objectives and contribution with analyzing Arabic gulf dialect tweets. This chapter also presented the 8 research questions that this thesis aims to address and answer. Moreover, the chapter presented the timeline and the outline of the thesis.

CHAPTER TWO: BACKGROUND

2.1 Introduction

In this chapter, I will present recent trends in sentiment analysis using neural networks. In the next sections, the principles of sentiment analysis and its importance will be explained (Section 2.2), the techniques that have previously been used to obtain a sentiment analysis system (Section 2.3), and the classification levels (Section 2.4). In Section 2.5, I explore word embedding to extract features and represent text to be classified. In Section 2.6, a discussion on neural networks and deep learning will be presented. Subsequently, I review related techniques that have been used for sentiment analysis, including deep learning, in Section 2.7. In addition, in Section 2.8, the complexity of the Arabic language will be discussed; This section discusses the diversity of Arabic forms, orthography, and morphology and how these challenges affect language processing. Lastly, Section 2.9 gives the conclusion of this chapter. This chapter satisfies RQ1.

2.2 Sentiment Analysis

Sentiment analysis or opinion mining is a computational method for the classification of thoughts, emotions, and attitudes regarding goods, services, or subjects expressed using text, sound, or image. Textual analysis is achieved by defining and categorizing these opinions using NLP techniques to derive subjective knowledge [11].

A large amount of unstructured data has been generated daily since the 2000s with the rise of the World Wide Web and the Internet. This amount has started to grow more rapidly with the advent of social media and smartphones, which creates a big data. Since then, sentiment analysis has become an important natural language processing tool to grasp trends and classify opinions. It is widely used to mine reviews, posts, and social media to evaluate services and improve the quality of decision making in different areas such as politics, commerce, tourism, education, and health [12]. sentiment analysis involves collecting reliable data from various resources, identifying the opinion embedded in each phrase, preparing labeled data for analysis, and extracting features by classifying these phrases [13], [14].

2.3 Techniques

There are two types of techniques for sentiment analysis systems [15], [16]: Machine learning and lexicon-based techniques; furthermore, there are hybrid methods that use both machine learning and lexicon-based techniques [17], [18].

2.3.1 Machine Learning Techniques

These techniques can be supervised, unsupervised, or semi-supervised. They are classified into “shallow learning” approaches, such as Naive Bayes (NB), maximum entropy (ME), SVMs, and deep learning approaches using neural networks for feature learning and sentiment classification.

SVMs [15] are machine learning models used for classification and regression. I focus on SVM because of its performance superiority compared to other machine learning models especially for Arabic sentiment analysis [17], [18]. An SVM is a large-margin non-probabilistic linear classifier. The principle of SVMs is to determine a hyperplane that separates training data points into two classes and keeps the margin, which is the distance

between the hyperplane and the nearest data point, as large as possible, considering the correct side of each data point. Thus, an SVM makes decisions based on the nearest data points, which are called support vectors and are chosen as the only successful elements in the training collection. Pang et al. [19] were the first to use machine learning for emotion classification. In their research, SVMs, NB, and ME were used to classify film review unigrams into two groups (positive and negative). It was demonstrated that machine learning models perform better than simple counting methods but do not perform as well in sentiment classification as traditional topic-based categorization, with an accuracy of approximately 83%.

2.3.2 Lexicon-Based Techniques

These techniques can be divided into "dictionary-based" and "corpus-based" approaches. The system does not require training to classify the data. Instead, it has predetermined sentiment values to compare the features in the text [20]. A sentiment lexicon dictionary contains lists of words that express feelings and opinions; it uses scores for the opinion words to count the most dominant lexicons and determine whether a sentence is positive or negative [21]. By contrast, a corpus relies on syntactic or co-occurrence patterns and has a large collection of texts for other opinion words. The technique starts with a list of seed opinion adjectives and a set of linguistic constraints such as and, but, either-or, and neither-nor to explore additional adjective opinion words and their orientations [21].

2.3.3 Hybrid Techniques

The hybrid approach combines both lexicon-based and machine learning-based approaches such that the trained model considers the lexicon-based results in its features, as in [22]. They compared the performances of the SVM and k-nearest neighbors (KNNs) machine learning algorithms to the performance of the hybrid approach. Their experiment confirmed that the use of the hybrid approach yields better accuracy.

2.4 Classification Levels

Sentiment analysis classifications can be applied at the document, sentence, and aspect granularity levels [12]. Sentiment analysis is studied at the three levels mentioned previously (document, sentence, and aspect), but these levels are not the only ones. A variety of researchers dealt with the problem using other levels such as word-level, clause-level, phrase level, and concept-level. In the next subsections, I focus on the main three granularity levels for sentiment analysis.

2.4.1 Document-level

In document-level sentiment classification [19], the entire document (which is about a single product or entity) is classified as either positive or negative, thus, this kind of classification is not applicable for a document that contains several products or entities.

2.4.2 Sentence Level

In sentence-level sentiment classification [23], the first step of the classification is called subjectivity classification, which classifies each sentence as subjective (that express opinions or subjective views) or objective (that express factual information from sentences). In the second step, the subjective sentences are classified as positive or negative orientation, while objective sentences are classified as neutral orientation (no opinion). The problem

with this kind of classification is that the objective sentence may include hidden opinions behind (example: I bought drinks yesterday and I found them opened).

2.4.3 Aspect-based Level

Aspect-level sentiment classification is first known as feature-based summary [24], the product features are identified and extracted from the source data, where entity and aspect/feature extraction are performed along with aspect sentiment classification. At the aspect-level, two core tasks are performed: Opinion target extraction, and aspect sentiment detection. In the first task, the target is often the aspect or topic to be extracted from a sentence, both entities and their aspects are extracted. Entities appoint to product names, services, events, and aspects, which can be expressed implicitly or explicitly, generally identify the attributes and components of entities. In the second task, the sentiment of the extracted aspects is determined within a given sentence.

2.5 Word Embedding

In sentiment analysis, words can be numerically represented before classification. Word embedding is a technique for language modeling and feature learning to enhance NLP [12] in which features are contextually learned, and words are converted into real-valued vectors with lower dimensionality. The advantage of this technique is that words with similar meanings are represented as similar vectors. However, most existing word-embedding techniques only capture the syntactic context and ignore the sentiment information of text. There are several available techniques for word embedding; some of the most common are briefly represented in the following. The first is manual feature extraction, such as the bag-of-words, and the second method is using Word to Vector. Those techniques will be represented in the next subsections.

2.5.1 The Bag-of-Words

The bag-of-words (BOW) model [25]–[27] is a representation of text that describes the occurrence of words within a document, which considering each word count as a feature regardless the word's position in the text. For sentiment analysis task, BOW uses a huge lexicon which has duplications of word and repetition. This lexicon is built manually which requires creating a 'positive' and 'negative' words list by recognizing the sentiment polarities based on personal observation. However, computing the total score of sentiments reviews is a challenging and time-consuming. In addition, BOW neglects text grammatically and ordering of words. Moreover, BOW cannot capture relationships between words or words that have the same meanings, and bigram and trigram approaches are required to handle this issue.

2.5.2 Word to Vector

Word to Vector (word2vec) [28] is unsupervised learning algorithm for obtaining vector representations for words. It consists of continuous BOW and skip-gram (SG) models. The continuous BOW model predicts the target word from the surrounding words within a window of a specified length. In contrast, the SG model is used to predict the surrounding words from the target word.

2.5.3 Global Vector

Global Vector (GloVe) [29] is another unsupervised learning algorithm , which is trained

on the nonzero entries of a global word-word co-occurrence matrix using Latent Semantic Analysis (LSA). Thus, unlike word2vec, which is a predictive model, GloVe does not use neural networks. The loss function is the difference between the product of word embeddings and the log of the probability of co-occurrence. Moreover, GloVe creates a global co-occurrence matrix by estimating the probability a given word will co-occur with other words.

2.5.4 Embedding Using Transformer

Transformers are a deep neural network architecture specially designed for the NLP tasks introduced in the paper "Attention Is All You Need" [30]. This architecture was proposed as an improvement of the traditional sequential models using recurrent network architecture which was used to capture the temporal information and relationship between the elements of a sequence. The architecture of the model is explained in Section 3.6. Bi-directional Encoder Representation from Transformers (BERT) [31] is the first multilingual model architecture that makes use of Transformers [30]. It is pre-trained on Wikipedia text from 104 languages and comes with hundreds of millions of parameters. Unlike the other embedding model such as word2vec and GloVe which are context-free models, BERT generate a representation of each word that is based on the other words in the sentence.

2.5.5 Arabic Word Embedding Models

There are some Arabic word-embedding models for NLP tasks [32]–[34] as follows: AraVec [32] is an example of Arabic word embedding for NLP tasks. Six different word-embedding models were constructed for the Arabic language using three different resources[1]: Wikipedia, Twitter, and Common Crawl webpage crawl data. Two models for each resource and the SG model were provided. These models were evaluated using qualitative and quantitative measures on several tasks that involved capturing word similarity. The proposed approach presents significant results. The method can effectively identify the similarity between words and can enhance the process of other NLP tasks. However, character-level embedding has not been performed.

Alayba et al. [33] presented an Arabic word-embedding model using a 1.5-billion-word corpus. Different word2vec models were constructed using the Abu El-Khair Corpus [35] to choose the most suitable one for the study. A continuous BOW model with 200 dimensions was chosen for an automatic Arabic lexicon. It was used with different machine learning methods and convolutional neural networks (CNNs) and was compared with different feature selection methods for sentiment classification. In addition, a health services dataset was used to test the generated model. Compared with their previous study [36] on a health services dataset, this approach increased the sentiment classification accuracy from 85% to 92% for the main dataset and from 87% to 95% for the sub dataset.

Altowayan and Tao [34] used a continuous BOW model for word representation learning using a large Arabic corpus[2] that includes completed texts of the Qur'an, MSA from news articles, the Arabic edition of international networks, and Dialectal Arabic from consumer reviews, with a total of 159,175 vocabulary items. To test the model, Twitter and book

[1] https://github.com/bakrianoo/aravec

[2] https://github.com/iamaziz/ar-embeddings

review datasets for sentiment classification and news article datasets for subjectivity classification were used with six different machine learning classification models. A comparison of the proposed subjectivity classification model with handcrafted models [37], [38] demonstrated that the proposed model outperforms the handcrafted models on the same dataset.

Fasttext[3] [39] is another word embedding and text classification method, particularly in the case of rare words. It ignores the morphology of the word and uses a bag of character-level n-gram to represent the words. It uses subword-level information (which is between word and character) to get word vectors for out-of-the-vocabulary words, that helps to capture the meaning of shorter words and allows the embeddings to understand suffixes and prefixes to generate better word embeddings.

AraBERT [171] is an Arabic pre-trained language model based on Google's BERT architecture. Two versions of AraBERT (AraBERTv0.1 and AraBERTv1) are available. The difference between these versions is that the v1 uses pre-segmented text where prefixes and suffixes were splatted using the Farasa Segmenter. The model has trained on ~70M sentences or ~23GB of Arabic text with ~3B words. AraBERT achieved state-of-the-art performance compared with other contextualized embedding and it will be presented in Section 3.6.

2.6 Artificial Neural Network

Neural networks are an interconnected group of nodes that simulate the function of the human brain [40]. Neural network consists of an input layer and an output layer and may include hidden layers of nonlinear processing units that link neurons. Moreover, neural network learns features depending on the real-valued activations and suitable weights that make the neuron exhibit desired behaviors [40].

Neural networks may be feedforward or recurrent/recursive [41]. Feedforward neural networks use a straightforward data processing scheme from the input layer through a hidden layer to the output layer. In the hidden layers, there are no cycles, and thus, the output of any layer does not affect that same layer. These networks use a backpropagation algorithm to train and compute a gradient of the cost function using the most recent input to update the network parameters and thereby reduce errors during training. In contrast, recurrent/recursive neural networks contain a loop. Therefore, they can process data from prior connections/values, as well as input from the most recent layer to predict the output of the current layer. It uses temporal backpropagation, which is a regular backpropagation but calculates the gradient of a cost function using all the inputs, not just the most recent inputs.

Deep learning is a machine learning technique that uses neural network with multiple deep layers for data processing, and thus, it learns complex features from simpler features as it proceeds from lower to higher layers using real-number activations for each neuron and weights for each link [40], [41].

[3] https://fasttext.cc

2.7 Sentiment Analysis Using Deep Learning

In recent years, deep learning has achieved satisfactory results in natural language processing, speech recognition, and computer vision tasks [42]. RNNs process the sequence of the inputs and help in processing the information contained in it (Rohith et al. 2018). In sentiment analysis, several types of models using deep neural networks have been employed [44], such as CNNs [45] and RNNs [46], including bidirectional recurrent neural networks (Bi-RNN) [47], LSTM [48], GRUs[49], recursive neural network [50], and hybrid methods. In the next chapter, I will present several RNN architectures. Figure 2.1 shows the classification techniques that have been used for the task of sentiment analysis.

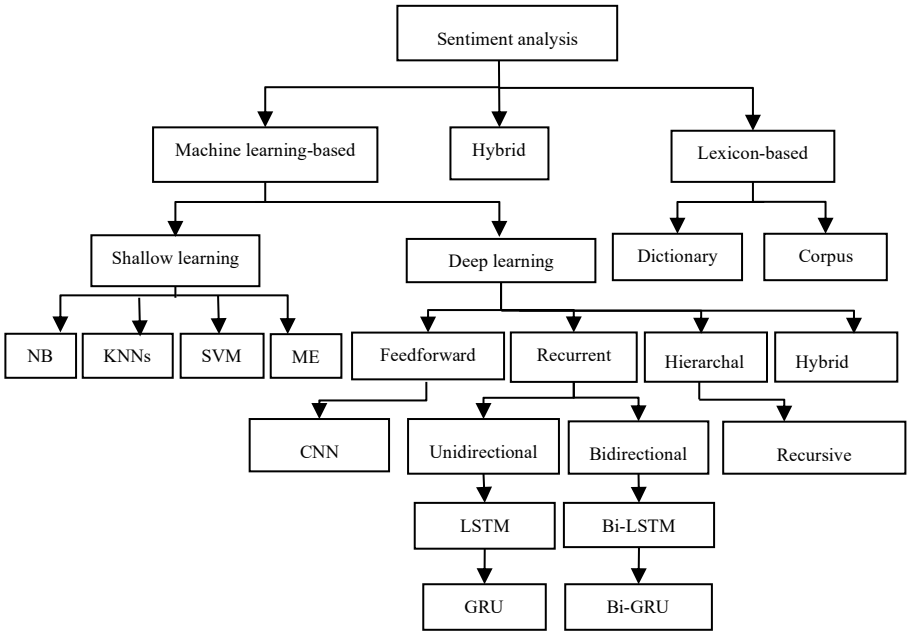

Figure 2. 1 Sentiment Classification Techniques

One of the challenges of SA using ML methods is the lack of correctly labeled data owing to difficulties related to subjective interpretation and high labor intensity. These difficulties hinder the size of the training data, affect performance, and can lower the classification accuracy [51]. To address these issues, sentiment analysis has utilized the deep learning technique owing to the automatic learning capability of the latter. This technique allows algorithms to understand sentence structure and semantics and to generate new feature representations; in contrast, traditional methods choose the most frequent word in a given input.

Given the importance of sentiment analysis using deep learning, numerous related studies involving English texts have been conducted. However, few studies have been published focusing on Arabic sentiment analysis.

2.8 Sentiment Analysis in Arabic

In Alsayat and Elmitwally paper [52], the different components present in the Arabic language are explained. Arabic sentiment analysis involves the following components:

phonetics, syntax, lexicology, morphology, semantics. The importance of the application of the levels that determine the sentiment in tweets is also a challenge that is explained in Abdulla et al. [53] paper. This part of the sentiment, once labeled across the different tweets, is then used as a reference dataset that can be then analyzed to give the sentiment of incoming tweets in real-time. The developed models were trained and validated on the tweets and the data generated in-house to see the validity of the model generation scheme and performance.

In the next subsection, I discuss the challenges of Arabic sentiment analysis in detail.

2.8.1 Arabic Sentiment Analysis Challenges

Arabic sentiment analysis challenges can be categorized into three groups: language forms, language orthography, and language morphology [54]–[57]. These groups are explained in greater detail in the following:

1- **Arabic forms:** Arabic has three primary forms: Classical Arabic, MSA, and dialectical Arabic. The Qur'an, the holy book of Islam, is in classical Arabic. MSA is similar to classical Arabic with less sophistication and more modern words; it is used in formal written and spoken media such as the news, education, and literature. Dialectical or colloquial Arabic is used mostly in daily life and has regional variations. Although dialectical Arabic is used mainly as a spoken language, currently, it is used in written social communication on social media and short messages and includes more than 30 dialects [55]. Dialects are generally classified into six basic groups [55]: Egyptian (including Egypt and Sudan), Levantine (including Jordan, Syria, Palestine, and Lebanon), Gulf (including Saudi Arabia, Oman, Kuwait, United Arab Emirates, Qatar, and Bahrain), North African (including Morocco, Tunisia, Algeria, Libya, and Mauritania), Iraqi, and Yemeni. For example, the expression **كيف حالك** (how are you) in MSA has different forms in each dialect (Egyptian: **إزيك**, Levantine: **كيفك**, Gulf: **شلونك**, North African: **شنوه احوالك**, Iraqi: **اشونك**, Yemeni: **كيفوك**).

2- **Arabic orthography:** Arabic text is written from right to left and is characterized by the absence of upper or lower cases. Its alphabet contains 28 letters: 25 consonants and only 3 vowels (**أ, و, ي**). In addition, short vowels (ً ٌ ٍ َ ُ ِ ْ) are used as diacritical marks and are put either over or under letters to indicate the exact articulation and to explain the meaning of the text. Moreover, based on the presence or absence of such diacritics, the meaning of words can be different. For example, the word (**علم**) may mean **عِلْم** (knowledge), **علم** (flag), or **عَلَّمَ** (teach), wherein the letters are identical while the diacritics are different. In the form of written Arabic used on social media, most words are written without diacritics, which makes it even more challenging to analyze [57].

3- **Arabic morphology:** The Arabic language has a highly complicated morphology, in which a word may carry significant information. A term in Arabic has several morphological characteristics as a spatially delimited token: agglutinative, inflectional, and derivational morphology [54], [56], [57]. The morphological features can be described as follows:

   - Agglutinative morphology: Compared to nonagglutinative languages, the number of possible words produced by different morphemes of a single root is high. A word

in Arabic can consist of a stem plus one or more affixes and clitics. A stem is a combination of a root and derived morphemes that can be attached to one or more affixes. On the other hand, the clitics include proclitics, which occur at the beginning of a word, such as the letters (**و**, **ف**) meaning (and, then), and enclitics that occur at the ending of a word, which is complement pronouns. For example, the Arabic expression (**ليكتبونها**) meaning (for them to write it) contains several attached affixes and clitics as follows: (**ل** proclitic), (**ي** prefix), (**كتب** stem), (**ون** suffix), and (**ها** enclitic). The complex nature of Arabic morphology leads to analytical ambiguity. For example, the word (**وجدنا** I found) can be analyzed as (**وجد** found + **نا** we) and as (**و** and + **جد** grandfather + **نا** we).

- Derivational morphology: Derivation is a method of deriving a new word from an existing word, involving the alteration of a part-of-speech, a change in meaning, or both. For example, from the root (**ك ت ب**) that constructs the verb (**كَتَبَ**) meaning (wrote), one can derive five or more derivations such as (**كَاتِب** writer), (**مَكْتُوب** letter), (**كِتَابَة** writing), (**كِتَاب** book), and (**مَكْتَب** office); thus, one root can generate different words with different meanings, which increases ambiguity when analyzing the language.
- Inflectional morphology: Inflection is the variation or change in the grammatical form that words undergo to mark distinctions of the case, gender, number, and person. For example, the variation in gender can be noticed in the word (student), which can be written as (**طالب**) for males and (**طالبة**) for females. Another word (white) can be written as (**أبيض**) for males and (**بيضاء**) for females. This group of inflected word shapes is called a lexeme category. To characterize a lexeme, a lemma, which is an exact shape, is conservatively chosen. The diversity of inflected Arabic words leads to a major challenge in NLP.

The diversity of dialects of the Arabic language, along with the richness in scripts, orthography, morphology, phonology, and semantics, poses research challenges that require appropriate systems and solutions to handle the ambiguity through the application of tokenization, spelling checks, stemming, lemmatization, pattern matching, and part-of-speech tagging. Extraction techniques are the backbone affecting model performance. In other words, the better the feature extraction technique is, the better the sentiment analysis. Feature extraction in different languages impacts the sentiment analysis results and can be carried out in multiple ways. Feature extraction is mainly concerned with the task of finding and yielding different statistical and transfer functions for important words in a dataset, which is a well-defined procedure in English compared to other languages [58]. In addition, the main difference of this approach is the seed corpus, which is prepared by delimiting a sequence of characters by noncharacter symbols, by counting the number of occurrences that follow a negated adverbial phrase, by counting the number of unconstructed sequences, [58]. In a review of sentiment analysis carried out in the domain of the Arabic language, an in-depth analysis was performed on the approaches related to sentiment analysis [59]. To create a corpus for machine learning approaches, a hybrid-based method is proposed, which tries to combine corpus-based and lexicon-based approaches, that focuses on the domain adaptation of sentiments and addresses poor language resources by using annotated corpora and lexicon resources.

2.9 Conclusion

Given the basic aspects of our study, the importance of sentiment analysis using deep learning, and the challenges of Arabic language processing, numerous related studies involving English texts have been conducted. This chapter gives a general insight into the recent techniques in Arabic sentiment analysis along with knowing the diversity of challenges while processing the Arabic language. The chapter provides the reason for choosing RNNs as an effective method in Arabic sentiment analysis. However, few studies have been published focusing on RNN Arabic sentiment analysis. In the next chapter, the shortage of related studies for Arabic sentiment analysis using RNNs will be discussed.

CHAPTER THREE: LITERATURE REVIEW

3.1 Introduction

In this chapter, the literature review will be introduced. This chapter aims to review all related studies that support the usability of the stacking method, also, to present all Arabic sentiment analysis RNN studies. This review leads to identifying the weakness of using RNN for Arabic and highlights the research gaps from these studies. To confine all related studies, I conduct a systematic review following guidelines described in Kitchenham [60] and Heckman [61].

The chapter is divided into four parts: in Section 3.1, I conduct the process of the systematic literature review method for RNN with English and Arabic text, stacked RNN, and Arabic gulf dataset. In Section 3.2, the latest trends for sentiment analysis using recurrent neural network will be considered. In Section 3.3, the related studies that use RNN models for Arabic sentiment analysis will be presented. In Section 3.4, the related works that use RNNs, LSTMs, GRUs, and stacked RNNs for Arabic sentiment analysis will be discussed. In Section 3.5, the transformers and their related studies will be presented. In Section 3.6, I discuss the findings from the related studies from Section 3.3 and Section 3.4.

3.2 Systematic Literature Review Methodology

A systematic review methodology is presented following guidelines in Kitchenham [60] and Heckman [61]. This method ensures that I enumerate all related studies and take them into consideration. The search terms, search strategy, and databases will be presented for the three review parts; First: RNN for sentiment analysis. Second: RNN for Arabic sentiment analysis. Third: stacked RNN for sentiment analysis. Fourth: Gulf dialect datasets.

In the first systematic review, an observation of studies that used RNN was considered. Then, those that used non-English for sentiment classification were excluded to discover the number of studies in English and compare it with the number of studies in Arabic. I focus on English, as it is the dominant language for studies, given the abundance of datasets and available supportive tools and resources. This part will be discussed in Section 3.2.1. In the second part, a deep step has been taken by considering studies on RNN Arabic sentiment analysis. The obtained studies from this review will be discussed in Subsection 3.2.2. In the third part, Stacked RNN studies for sentiment analysis was considered, as it is the focus of our research scope. The obtained studies from this review will be discussed in Subsection 3.2.3. In the fourth part, I tackle the shortage of datasets in the gulf region in Subsection 3.2.4. Those four parts are having different keywords and elimination strategies as depicted in the following subsections.

3.2.1 RNN for Sentiment Analysis

The search sentence "x y" has been used after many searches and observations to find the correct keywords, where x is "recurrent" or "neural," and y is "sentiment" or "opinion." That is, I focus on all studies that use RNNs to evaluate sentiment by combining x and y, resulting in two possible search sentences. The targeted publisher databases are: Springer, IEEE, ACM, Science Direct, and other databases from Google Scholar, aclweb, NIPS, AAAI, and Semantic Scholar. The studies under consideration were published from 2013 through 2018. The year 2013 was selected because no studies using RNNs for sentiment analysis prior 2013 were found. The year 2018 was selected to gauge the English studies

on RNNs in general. The initial search resulted in 38,926 papers. The search aimed to cover all related studies with names, abstracts, keywords, and complete texts. These papers were checked for final selection criteria according to the following: papers related to artificial intelligence were included, whereas irrelevant papers (such as those involving non-English sentiment analysis) were excluded. In addition, those concerned with nontarget areas, such as image analysis, video analysis, and gender identification, were excluded as well. After applying the selection criteria, there were 193 papers. Table 3.1 shows the total number of papers for each database using the mentioned keywords for RNNs in English.

Table 3. 1 Number of Studies for Initial Search for Each Keyword for All Databases

Databases Keyword	Springer	IEEE	Science Direct	ACM	Other
recurrent + neural + sentiment	691	859	325	524	11,600
recurrent + neural + opinion	2,238	953	1,870	466	19,400
Total (initial search)	2,929	1,812	2,195	990	31,000
Total (after selection)	37	52	12	32	60

3.2.2 RNN with Arabic Text for Sentiment Analysis

In this part of the search, the focus was on studies concerned with RNNs for Arabic sentiment analysis, and a combination of three search terms was used. The first two terms were the same x and y as in the previous subsection, and the additional term "Arabic" is added to the two search sentences. The same targeted databases in the previous subsection have been used in this search. Initially, the keyword search resulted in 2,636 papers from 2013 through 2019. The search targeted titles, abstracts, keywords, and full texts to cover all relevant studies. The papers were reviewed for final selection according to the criteria used in the previous subsection. Moreover, papers on non-Arabic sentiment analysis were excluded. After selection, there were 30 papers. By comparing these studies with those from the previous subsection, I conclude that using an RNN for sentiment classification for Arabic text is still in its infancy. Table 3.2 shows the total number of papers found in each database using the mentioned keywords for Arabic text. In Table 3.2, the total initial search refers to the total studies found in each journal, while the total after selection means the last studies that were selected as related studies from these journals.

Table 3. 2 Number of Arabic Studies for Initial Search for Each Keyword for All Databases

Databases Keyword	Springer	IEEE	Science Direct	ACM	Other
Arabic+ recurrent + neural + sentiment	81	103	58	26	1,450
Arabic + recurrent + neural + opinion	138	74	81	26	1,720
Total (initial search)	219	177	139	52	3,170
Total (after selection)	8	6	6	0	10

3.2.3 Stacked RNN Models

This part of the literature focuses on studies for stacked RNN applied for sentiment analysis for all languages, a combination of three search terms have been used. The first two terms are equal to x and y of those for the English and Arabic studies mentioned in the previous subsections. An additional term "Stack" or "deep" is combined with the two search

sentences. The targeted publisher databases are: Springer, IEEE, ACM, Science Direct, and other databases from Google Scholar, aclweb, NIPS, AAAI, and Semantic Scholar

The time frame where the studies are conducted is from 2013 until 2019. Initially on searching the sentences 2,636 papers obtained between 2013 and 2019. The search targeted titles, abstracts, keywords, and full text to cover all relevant studies. Those papers are reviewed for final selection according to the following criteria: include papers in the Artificial Intelligence domain, excluding non-prime papers, in addition to excluding non-target areas such as image analysis, video analysis, and gender identification. Although the term" deep" has an extended meaning, the goal is to capture any related paper that uses many layers of RNNs. Hence, I ended up with 19 papers, three of them using the Arabic language. Table 3.3 shows the final resulting papers compared with the initial search for each database.

Table 3. 3 Number of Stacked RNN Studies for Initial Search for Each Keyword for All Databases

Databases Keyword	**Springer**	**IEEE**	**Science Direct**	**ACM**	**Other**
stack+ recurrent + neural + sentiment	262	359	155	202	1,950
stack + recurrent + neural + opinion	341	392	227	182	2,490
deep+ recurrent + neural + sentiment	944	1,149	459	717	13,100
deep + recurrent + neural + opinion	1,867	1,050	1,341	596	17,200
Total (initial search)	3,414	2,950	2,182	1,697	34,740
Total (after selection)	8	5	0	1	5

About 19 studies between 2013 and 2019 are identified. These papers will be discussed in Section 3.5. Since the trend towards the use of RNNs for Arabic sentiment analysis in recent research is increasing, the stacking method will be an enhanced analysis of the kind of RNN used compared to the machine learning method. Although different machine learning algorithms [62] have been employed for sentiment analysis in the past, the sequence and latency of the meaning of the phrases of a particular text are lost especially for long texts, capturing sequence and latency is important to obtain the actual sentiment. This information is retained in RNNs, which improves the prediction accuracy. In the next subsection, I present how sentiment analysis is performed in various studies and the different techniques that have been employed.

3.2.4 Gulf Dialect Datasets

One of the main obstacles in Arabic sentiment analysis is the scarcity of high-quality resources especially for the gulf region, this includes datasets, corpora, and lexicons. To tackle the shortage of datasets in the gulf region, a systematic review was implemented to give an insight into recent gulf datasets and their size.

A search sentence "x y" was used, where x is: 'dataset sentiment', and y is: 'Saudi' or 'gulf' or 'khaliji'. That is, the focus will be on all studies that generate a gulf dataset for sentiment analysis by combining "x" terms with each "y" term resulting in two possible search sentences. The targeted databases: Springer, IEEE, ACM, Science Direct, and other databases such as aclweb, NIPS, AAAI, and Semantic Scholar.

Although I didn't define a time frame, I noticed that all datasets are in the year 2014 and

above. Initially, on searching the sentences, a total of 6,012 papers was obtained. The search targeted titles, abstracts, keywords, and full text to cover all relevant studies. Note that I did not search multi-dialectal datasets due to the shortage of gulf tweets or reviews (less than 400) in these datasets that are already small.

Hence, I ended up with 19 papers. Table 3.4 shows the final resulting papers compared with the initial search for each database. Table 3.5 presents the 19 datasets founded with the size and class.

Table 3. 4 Number of Studies that used gulf dataset for Initial Search for Each Keyword for All Databases

Database **Keyword**	**ACM**	**IEEE**	**Elsevier**	**Springer**	**Other**
dataset+saudi+sentiment	77	200	226	283	4,250
dataset+gulf+sentiment	41	68	155	195	4,390
dataset+khaliji+sentiment	0	5	1	3	18
Total (initial search)	118	273	382	481	4,758
Total (after selection)	2	5	3	1	8

Table 3. 5 Gulf Dataset with Size and Classes

Papers	**Type**	**Size**	**Class**
Alhumoud et al. [18] [22]	Tweets	2,690	Positive, negative
Al-Biqami Saudi Dataset [63]	Tweets	11,070	Positive, negative, neutral
Al-Rubaiee et al (Gulf) [64]	Tweets	1,331	Positive, negative, neutral
Al-Harbi and Emam [65]	Tweets	5,500	Positive, negative, neutral
Assiri et al. [66]		4,700	Positive, negative, neutral
Al-Subaihin and Al-Khalifa [67]	Restaurant reviews	NA	Positive, negative, neutral
Ben Salamah and Elkhlifi [68]	Tweets	4,213	Adjectives classes (positive, negative)
Al-Thubaity et al. [69]	Tweets	5,400	The sentiment (positive, negative, neutral, objective, spam, and not sure) Emotions (anger, fear, disgust, sadness, happiness, surprise, no emotion, and not sure
Aldayel and Azmi [70]	Words	1,500	Positive, negative
Alahmary et al. [71]	Tweets	32,063	Positive, negative
Qamar et al [72]	Tweets	1,331	Positive, negative, neutral
Baly et al. [73]	Tweets	1,200	Positive, negative, neutral
Adouane and Johansson [74]	Restaurant review	4072	Positive, negative, neutral, and mixed
Al-Obaidi and Samawi [75]	reviews	18,282	Positive, negative, neutral
Al Suwaidi et al. [76]	Tweets	1,000	Positive, negative, neutral
Alqarafi et al. [77]	Tweets	4,000	Positive, negative
Al-Twairesh et al. [78]	Tweets	17,573	positive, negative, neutral, and mixed

Al-Thubaity et al. [79]	Tweets	1,500	Positive, negative, neutral
Azmi and Alzanin [80]	Texts	815	strongly positive, positive, negative, and strongly negative

The results show the lack of a large gulf dataset and resources for sentiment analysis purposes. For the Arabic language, there are not many available datasets not only for sentiment analysis but also for other NLP tasks, especially for Saudi Arabia that has the highest annual growth rate of social media users anywhere in the world [2], That shows the need for a larger dataset.

3.3 Recurrent Neural Network Approaches for Sentiment Analysis

This section answers RQ2. To consider the latest trends in RNNs for sentiment analysis, the concept of RNNs was explained. The RNNs as well as their architecture and function will be presented; moreover, the different RNN architectures, such as Bi-RNN, LSTM, GRU, and the stacked architecture will be introduced, which are crucial to understanding related studies that will be reviewed in the next subsection.

RNN architectures vary depending on the task. For example, in machine translation, several inputs are used to generate several outputs: the translated sentences. In sentiment analysis task, several inputs are used to create a single output: the prediction, as shown in Figure 3.1 [81].

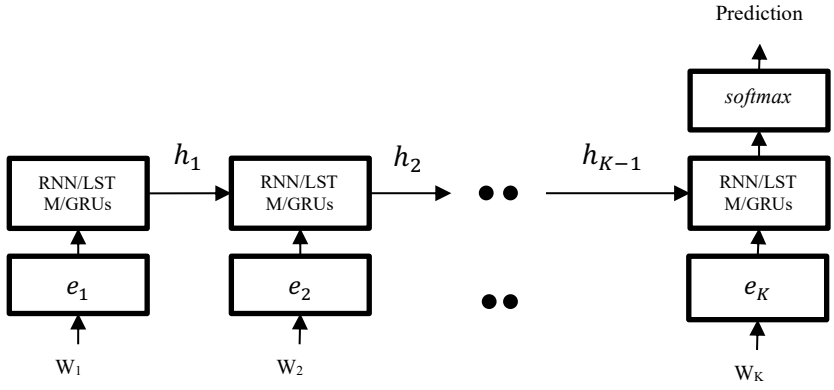

Figure 3. 1 Many-to-one Architecture

3.3.1 Vanilla RNN

RNN shares feature learned and perform the same task for every element of a sequence; it uses previous computations to compute the current output. The inputs are a hidden state vector for the previous timestamp $(t-1)$, and an input state vector at a certain time (t). The Equation (3.1) generates the hidden state vector (h). W, U, and b are parameter metrics, where (Wh) is the weight matrix used to condition the input (Xt). (Uh) is the weight matrix used to condition the previous hidden state $(ht\text{-}1)$. The result of the activation function as seen in Equation (3.1) is passed to the next timestamp and $softmax$ activation, Table 3.6 to generate the output as seen in equation (3.2). Figure 3.2 shows the architecture of an RNN for three timestamps, and Table 3.6 defines the symbols appearing in Equation (3.1) and Equation (3.2).

$$h_t = tanh(W_h X_t + U_h h_{t-1} + b_h) \quad (3.1)$$

$$y_t = softmax(h_t) \quad (3.2)$$

There are many problems with basic RNN [82]: First, it uses an earlier sequence to make a prediction, this can lower the accuracy of results. This problem can be solved using Bi-RNN [83]. Second, exploding gradient problem, which is a problem that may occur during a training process that generates unstable weights and directions. This problem can be solved using gradient clipping [84]. Third, the vanishing gradient problem. In this problem, basic RNN cannot capture long-term dependency; this problem can be solved using LSTM and GRU using a unique additive gradient structure that includes direct access to the forget gate's activations, enabling the network to encourage desired behavior from the error gradient using frequent gates update on every time step of the learning process.

Table 3. 6 Symbols and their Definitions for RNN Equations

Symbol	Definition
$\boldsymbol{h_t}$	The hidden state vector for the current time step.
$\boldsymbol{h_{t-1}}$	The hidden state vector for the previous time step.
$\boldsymbol{X_t}$	The input state vector for the current time step.
$\boldsymbol{y_t}$	The output for the current time step.
$\boldsymbol{W_h, U_h, b_h}$	Parameter metrics, while Wh and Uh refer to the weights assigned to the hidden state vector. b refers to bias. They are initialized with random numbers and learned as the network trains.
$\boldsymbol{tanh}$	Hyperbolic tangent function. The output range is [-1, 1].
$\boldsymbol{softmax}$	A function used in the final layer of neural network that turns numbers into probabilities that sum to one.

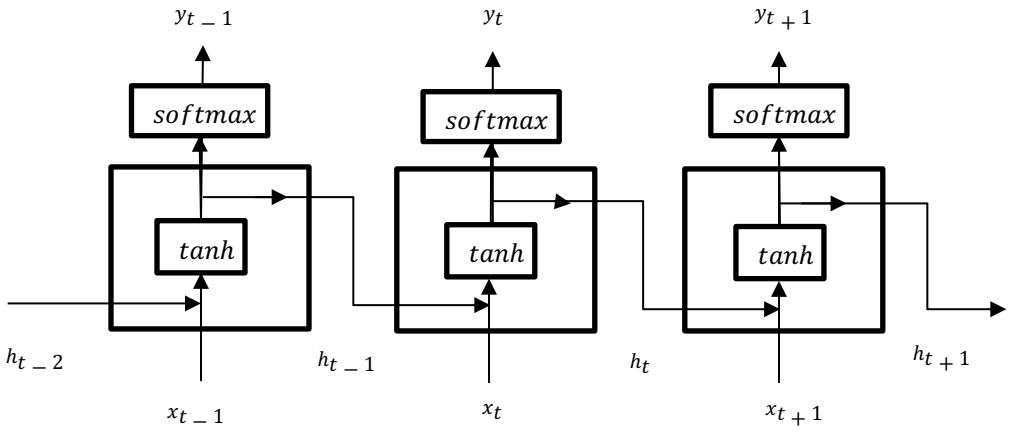

Figure 3. 2 RNN Architecture

3.3.2 Bi-RNN

Schuster and Paliwal propose first Bidirectional RNN [83]. The idea behind Bi-RNN is that the output at each timestamp depends on the forward elements along with the backward elements in the sequence. Bi-RNN consists of two RNNs, the first RNN goes from left to

right direction while the other RNN doing the reverse, this approach can increase the amount of input information available to the network instead of using an earlier sequence that generates lower information and hence generate more accurate outputs in each timestamp. Figure 3.3 presents the architecture of Bi-RNN [85]. The symbol x is the input state vector, (h) is the hidden state vector, and (y) is the output.

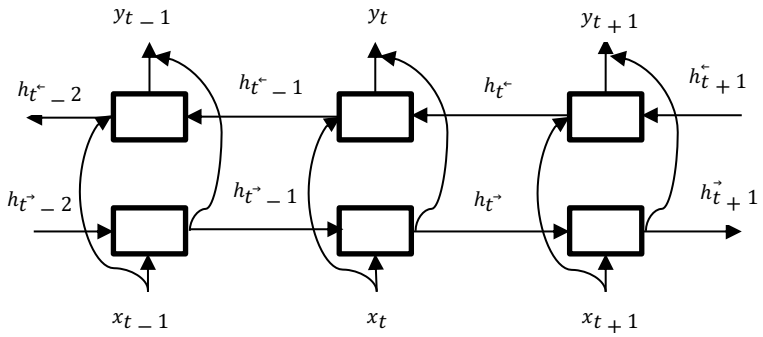

Figure 3. 3 Bi-RNN Architecture

3.3.3 Stacked RNN

Stacked RNNs have been proposed by Schmidhuber [86], El Hihi, and Bengio [87] as a way of constructing deep RNNs. An empirical investigation by Hermans and Schrauwen [10] showed that multiple layers in the stack can operate at different time scales, ensuring the integration of depth not only in time but also in space; RNNs are inherently deep in the time since their hidden state is a function of all previous hidden states. Depth in space can be obtained by stacking multiple recurrent hidden layers on top of each other, they can better use parameters by distributing them over the space through multiple layers. When applied to natural language sentences, such hierarchies might better model the multi-scale language effects that are emblematic of natural languages.

Pascanu et al. [88] indicate that deep learning is built around a hypothesis that a deep, hierarchical model can be exponentially more efficient at representing some functions than a shallow one. In addition, they explore other ways of constructing deep RNNs that are orthogonal to the concept of stacking layers on top of each other. Figure 3.4 shows the architecture of the stacked RNNs.

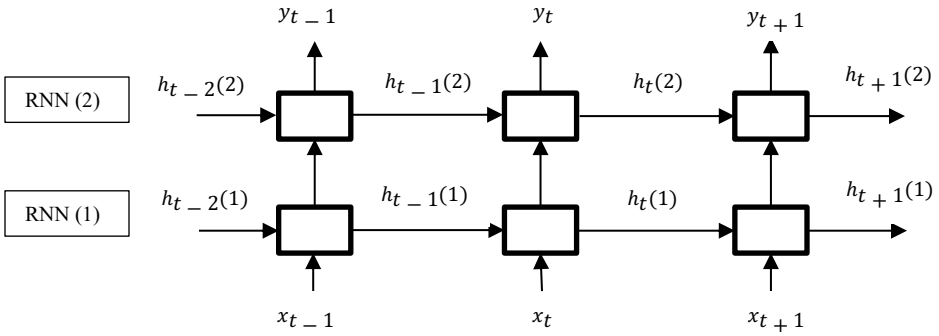

Figure 3. 4 Stacked (deep) RNN Architecture

3.3.4 LSTM

LSTM [89], [90] is an RNN that uses a complex activation function to capture long-term dependencies confidently and resolve the vanishing gradient problem using three gates. Along with the input state, it has a hidden and a cell state. In addition to calculating the activation function and output probability at each timestamp, LSTM calculates the forget, input, and output gates using the sigmoid function σ, whose values range from 0 to 1. In Equation (3.3), the forget gate f is used to determine what information be discarded from the cell state C. If the result is approximately zero, the information for the cell state will be discarded; otherwise, it will be retained.

$$f_t = \sigma\left(W_f\, h_{t\text{-}1} + U_f\, X_t + b_f\right) \tag{3.3}$$

The next step is to decide what new information will be added to the cell state using input gate (i) as seen in Equation (3.4) if there is no update, the value is zero.

$$i_t = \sigma\left(W_i\, h_{t\text{-}1} + U_i\, X_t + b_i\right) \tag{3.4}$$

In Equation (3.6), a $tanh$ layer creates a vector of new candidate value $\left(\tilde{C}\right)$. Subsequently, a cell state (C) as seen in Equation (3.6) is updated using Equations (3.3), (3.4), and (3.5).

$$\tilde{\mathrm{C}}_t = \tanh\left(W_c\, h_{t\text{-}1} + U_c\, X_t + b_c\right) \tag{3.5}$$

$$C_t = C_{t\text{-}1} * f_t + i_t * \tilde{\mathrm{C}}_t \tag{3.6}$$

Finally, the new hidden state vector h is generated. In Equation (7), the output gate O is used to determine which parts of the cell state will be output using the sigmoid function. Subsequently, using the *tanh* function of the new cell state C, the hidden state is obtained in Equation (3.8). Figure 3.5 illustrates the LSTM architecture [91]. Table 3.7 defines the symbols appearing in Equations (3.3), (3.4), (3.5), (3.6), (3.7), and (3.8) for more clarity.

$$O_t = \sigma\left(W_o\, h_{t\text{-}1} + U_o\, X_t + b_o\right) \tag{3.7}$$

$$h_t = O_t * \tanh\left(C_t\right) \tag{3.8}$$

Table 3. 7 Symbols and their Definitions for LSTM Equations

Symbol	Definition
h_t	The hidden state vector for the current time step.
h_{t-1}	The hidden state vector for the previous time step.
X_t	The input state vector for the current time step.
f_t	The forget gate for the current time step.
i_t	The input gate for the current time step.
O_t	The output gate for the current time step.
$\tilde{C}_t$	The candidate value for the current time step.
C	The cell state vector for the current time step.
C_{t-1}	The cell state vector for the previous time step.
$W_f, W_i, W_c, W_o, U_f, U_i, U_c, U_o, b_f, b_i, b_c, b_o$.	Parameter metrics, while (W) and (U) refers to the weight (e.g. W_o is a weight that is assigned to the output gate o). The (b) refers to the bias (e.g. b_o is a bias that is assigned to output gate). They are initialized with random numbers and learned as the network trains.
σ	The sigmoid function takes a real-valued number and returns a value in the range [0, 1].
$tanh$	Hyperbolic tangent function. which takes a real-valued number. The output range is [-1, 1].

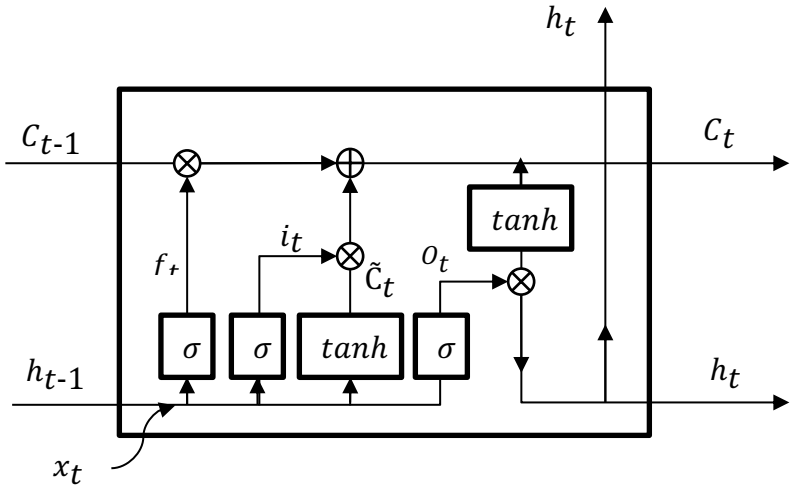

Figure 3. 5 LSTM Architecture

3.3.5 GRU

A GRU is a variation on LSTM. Choe et al. [92] proposed an RNN encoder-decoder and introduced the GRU model, which uses two gates (instead of three gates in LSTM) and fewer parameters, and thus it is a simpler model. The two gates are the reset gate r_t, which indicates the relevance of the previous cell state for computing the next candidate, as shown in Equation (3.11), and the update gate u_t, which is a combination of the forget and the input gates, as seen in Equation (3.10). Initially, the cell state equals the hidden state, that is, a *tanh* layer generates a vector of new candidate values $\tilde{C}$ in Equation (3.9) using the reset gate r_t. Subsequently, in Equation (3.12), the hidden state *h* is updated using Equations (3.9) and (3.10).

$$\tilde{C}_t = \tanh(W_c . [r_t * h_{t-1}] + U_c X_t + b_c) \quad (3.9)$$

$$u_t = \sigma(W_u h_{t-1} + U_u X_t + b_u) \quad (3.10)$$

$$r_t = \sigma(W_r h_{t-1} + U_r X_t + b_r) \quad (3.11)$$

$$h_t = u_t * \tilde{C}_t + (1 - u_t) * h_{t-1} \quad (3.12)$$

As $C_t = C_{t-1}$ and the update gate u_t is always zero, GRU does not suffer from vanishing gradients problem, thus allowing the RNN to learn long dependencies. Table 3.8 defines the symbols appearing in Equations (3.9), (3.10), (3.11), and (3.12). GRU architecture is shown in Figure 3.6 [89].

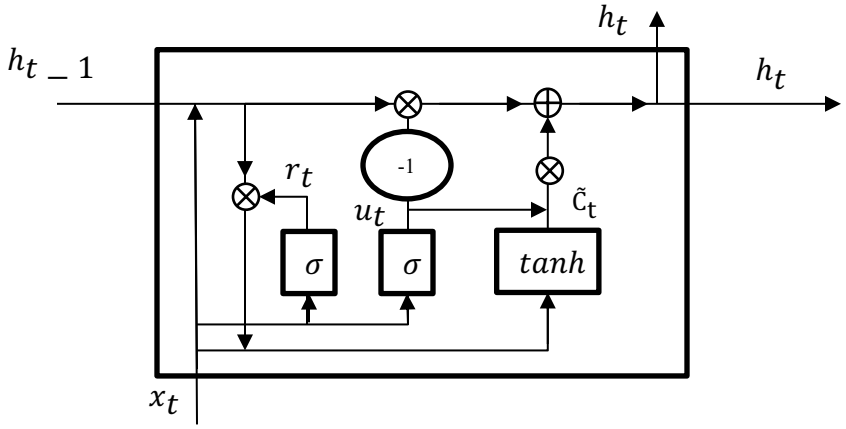

Figure 3. 6 GRU Architecture

Table 3. 8 Symbols and their Definitions for GRU Equations

Symbol	Definition
C_t	The cell state vector for the current time step.
C_{t-1}	The cell state vector for the previous time step.
X_t	The input state vector for the current time step.
r_t	The reset gate for the current time step.
u_t	The update gate for the current time step.
$\tilde{C}_t$	The candidate value for the current time step.
$W_u, W_r, W_c,$ $, U_u, U_r, U_c,$ $b_u, b_r, b_c.$	Parameter metrics, while (W) and (U) refers to the weight (e.g. U_u is a weight that is assigned to the update gate o to condition the current input state). While (b) refers to the bias (e.g. b_u is a bias that is assigned to update gate). They are initialized with random numbers and learned as the network trains.
σ	The sigmoid function takes a real-valued number and returns a value in the range [0, 1].
$tanh$	Hyperbolic tangent function. which takes a real-valued number. The output range is [-1, 1].

According to Chung et al. [93], there are similarities and differences between LSTM and GRUs. The most prominent common feature is their ability to retain existing components and add the updated component instead of replacing the entire activation function as in conventional recurrent units. Accordingly, long-term dependencies may be captured, and shortcut paths may be created so that the error can be easily backpropagated without rapidly vanishing, thus eliminating the vanishing gradient problem.

A difference between LSTM and GRUs is that the former controls the memory content and cell state exposure to other units using the output gate, whereas the latter exposes the entire cell state to other units without control. Another difference, as mentioned earlier, is that the LSTM unit has separate input and forget gates, whereas the GRU merges these operations through the reset gate. The LSTM unit does not control the information flow from the previous timestamp, but it controls the amount of the new content being added to the memory cell from the forget gate. By contrast, a GRU controls the information flow from the previous activation when computing the new candidate activation but does not control the amount of the candidate activation being added, as the control is performed through the update gate [93].

3.4 Arabic Sentiment Analysis Using Recurrent Neural Networks

This subsection addresses RQ3 by discussing related studies that use RNN models for Arabic sentiment analysis. There are only 30 such studies. The studies under consideration are those that introduce a new Arabic dataset for sentiment analysis and use an RNN to test the accuracy of these corpora. Moreover, as Arabic datasets are scarce and limited, the various datasets used in all studies under consideration have been highlighted in Table 3.9. The studies are divided according to the classification level (sentence or aspect) and the type of analysis (emotion detection, emoji analysis, hate speech detection, or sentiment classification). I discuss the six papers related to aspect-based sentiment analysis in Subsection 3.4.1. In addition, the 24 papers related to sentence-level will be discussed in Subsection 3.4.2; five papers related to emotion detection (Subsubsection 3.4.2.1), three papers on emoji analysis (Subsubsection 3.4.2.2), one paper related to hate speech detection (Subsubsection 3.4.2.3), and 15 papers focused on only sentiment classification (Subsubection 3.4.2.4).

Table 3. 9 Related Works and Used Arabic Datasets for sentiment analysis Using RNN

Datasets	size	Classes	Arabic form	Papers
Large-scale Arabic Book Reviews Dataset (LABR) [94].	63,257 book reviews.	3-class	MSA/Dialectal	Abbes et al. [95]. Baniata and Park [96].
SemEval-2016 Task 5 [97].	3,000 hotel reviews	4-class	MSA/Dialectal.	Tamchyna and Veselovska [98]. Ruder et al. [99]. Wang and Lu [100]. Ponti et al. [101]. Al-Smadi et al. [7]. Al-Smadi et al. [102].
Arabic Sentiment Tweets Dataset (ASTD) [103] .	10,006 tweets.	4-class	Egyptian dialect.	Al-Azani and El-Alfy [104]. Al-Azani and El-Alfy [105]. Baccouche et al. [106]. Alayba et al. [107]. Heikal et al. [3].
ArTwitter [53].	2,000 tweets.	2-class	MSA/Jordanian dialect.	Al-Azani and El-Alfy [104]. Alayba et al. [107]. Al-Azani and El-Alfy [105].
QCRI [37].	2,300 tweets.	(objective, subjective), 2-class	MSA/Egyptian, Levantine, and Gulf dialects.	Al-Azani and El-Alfy [105].
Syria [108].	2,000 tweets.	2-class, 3-class	Syrian dialect.	Al-Azani and El-Alfy [105].
Semeval-2017 Task 4 [109].	1,656 (2-class) + 3,355 tweets.	2-class, 3-class 5-class	Dialectal.	Al-Azani and El-Alfy [105]. Gonzalez et al. [110]. Samy et al. [111].
SemEval-2018 Task 1 [112].	11,781 tweets.	7-class,12-emotion classes	Dialectal.	Abdullah and Shaikh [113]. Abdullah et al. [114]. Samy et al. [111]. Alhuzali et al. [115]. Abdou et al. [116].
DINA [117].	3,000 tweets.	6-emotion classes	Dialectal.	Alhuzali et al. [115].
AraSenTi [78].	17,573 tweets.	2-class	Saudi dialect.	Alwehaibi and Roy [118].
Multi-dialect Arabic sentiment Twitter datase (MD-ArSenTD) [73].	14,400 tweets.	3-class	Dialectal.	Baly et al. [73].
BRAD 2.0 [119].	692,586 book reviews.	3-class	MSA/Dialectal.	Elnager et al. [120].
Arabic Health Services Dataset [36].	2,026 tweets.	2-class	Dialectal.	Alayba et al. [107].
Collected tweets.	12,897 tweets. 6,600 tweets.	2-class, 2-hate class	Dialectal. Dialectal.	El-Kilany et al. [121]. Albadi et al. [122].
ArSenL	157,969 synsets	3-class	MSA	Badaro et al. [123].
Semeval-2016 Task 7	1,366 tweets	4-class	MSA/Dialectal	Kiritchenko et al.[124].
Arabic Gold Standard Twitter Data for Sentiment Analysis	4,191 tweets	3-class	MSA/Dialectal	Refaee and Rieser [125]
ArSAS	19,762 tweets	3-class	MSA/Dialectal	Elmadany et al. [126]

3.4.1 Aspect-based Level Sentiment Analysis Using RNN

In this subsection, I focus on studies that classify text at the aspect level. These works are categorized based on RNN variations (LSTM, and GRU).

Al-Smadi et al. [7] proposed an aspect-based sentiment analysis of Arabic hotel reviews using two implementations: deep RNN and SVM, along with word embedding, lexical, syntactic, morphological, and semantic features. The F1 score was employed to evaluate aspect opinion target expression extraction (T1) and aspect category identification (T2) and the accuracy score to evaluate sentiment polarity identification (T3). Different models were used for performance comparison: a baseline model [127], INSIGHT-1 [128], and UFAL [98]. The results demonstrated that the SVM outperformed the RNN approach in all tasks because the feature sets used to train the SVM were large, particularly in T3, with an accuracy of 95.4%, compared with RNN with an accuracy of 87%. However, the deep RNN outperformed the SVM in terms of the execution time, particularly in T2, with a speed rate of 5.3× compared with T1, with a speed rate of 4.4×, and T3, with a speed rate of 2.1×.

Tamchyna and Veselovska [98] employed a multilingual LSTM model for aspect-based sentiment analysis using the categories of consumer electronics, restaurants, and hotel reviews in seven languages: Arabic, Dutch, English, French, Russian, Spanish, and Turkish. The main goal was to discover the linguistic patterns in the data automatically without feature extraction or language-specific tools. The model was compared with five different versions of the same model: the official baseline model (provided by the task organizer) using an SVM; a baseline model using logistic regression without any type of feature except BOW, a model submitted for official evaluation, where the networks were not fully optimized by the submission deadline; an optimized model, where the results were obtained after the deadline; and the best model, where the best score for each language and domain was reported. Regarding the Arabic language aspect-based sentiment analysis, the optimized model exhibited the best results in terms of the F1 score (52.59%) compared with all the other models, particularly the best model, which achieved an F1 score of 52.11%.

Al-Smadi et al. [102] improved the deep learning approach in [7] for aspect-based sentiment analysis of Arabic hotel reviews using LSTM: first, with a character-level Bi-LSTM along with a conditional random field (CRF) classifier for aspect opinion target expression extraction, and second, with an aspect-based LSTM for aspect sentiment polarity classification in which the aspect opinion target expressions are considered attention expressions to support sentiment polarity identification. For sentiment polarity identification, the authors compared the proposed model with the baseline model [127], INSIGHT-1 [128], and IIT-TUDA [129]. The authors used the F1 score to evaluate opinion target expression extraction and the accuracy to evaluate sentiment polarity identification. The results demonstrated that this approach outperformed the baseline on both tasks, with an improvement of 39% for aspect- opinion target expression extraction and 6% for aspect sentiment polarity classification. For opinion target expression extraction, the proposed model yielded promising results, particularly when Fasttext was used for character embedding, with an F1 score of 69.98%, compared with the same model when word2vec was used, which achieved an F1 score of 66.32%. Moreover, for sentiment polarity identification, the proposed model outperformed other models, with an accuracy of 82.6%, except INSIGHT-1, which used a CNN and had an accuracy of 82.7%.

Ruder et al. [99] presented a hierarchical bidirectional LSTM that can consider inter

sentence relations such as background and inter sentence relations (e.g., aspect-based sentiment analysis using bidirectional sentence-level LSTM and bidirectional review-level LSTM). The evaluation was performed using seven different models: the best model for each domain and language [130], XRCE [131], IIT-TUDA [129], a sentence-level CNN (INSIGHT-1) [128], a sentence-level LSTM, which is the first layer of the proposed model, and the proposed model with randomly initialized word embeddings (H-LSTM) and with pre-trained embeddings (HP-LSTM). Regarding the Arabic language analysis, the proposed model with pre-trained word embeddings outperformed the other models, including the sentence-level models, with an accuracy of 82.9% compared with the best model, which achieved an accuracy of 82.7%.

Wang and Lu [100] proposed a segmentation attention-based Bi-LSTM model for aspect-based sentiment analysis that extracts sentiment information and learns latent opinions by capturing the structural dependencies between the given target and the sentiment expressions using a CRF layer. The proposed model was tested on two groups of datasets. One group was from online reviews (laptop and restaurant) and social comments on Twitter, and SemEval 2014 Task 4 was used to analyze each component of the model. In the other group, the SemEval 2016 Task 5 dataset [97] was used to examine the model's language sensitivity. The evaluation was performed by comparing the extracted opinions with the manually annotated opinions using several models from Ruder et al. [99], LSTM with standard attention *Softmax* (A-LSTM), and the proposed model with segmentation attention layer with and without penalty terms (SA-LSTM and SA-LSTM-P, respectively). Regarding the hotel reviews in Arabic, the experiments demonstrated that the proposed model with penalty achieved the best results (accuracy of 86.9%) compared with other models, particularly with the proposed model without penalty, which achieved an accuracy of 86.7%.

Ponti et al. [101] examined the amount of information that representations retain about the polarity of sentences for each language. Moreover, they presented a model that decodes sentiment from unsupervised sentence representations learned by different architectures (sensitive to words, sensitive to order, or neither), such as additive SG, FastSent, sequential denoising autoencoder, and distributed BOW models, and they compared these models with bidirectional LSTM. Regarding the Arabic language sentiment analysis, the distributed BOW model yielded the best results compared with other unsupervised representations, with a weighted F1 score of 76.76%, whereas the sequential denoising autoencoder model achieved an F1 score of 72.13%. However, Bi-LSTM outperformed all the unsupervised strategies, with an F1 score of 86.56%. Bi-LSTM used the SG model for sentence representation; hence, an RNN model can be chosen as a ceiling, especially for the Arabic language. Table 3.10 shows the results for aspect-based sentiment analysis for Arabic text using RNNs.

Table 3. 10 Aspect Based Sentiment Analysis Using RNN for Arabic Text

Papers	Classifier	Document/Text Representation	Dataset	Accuracy
Al-Smadi et al. [7].	RNN.	Word2vec.	SemEval-2016 Task 5 : Arabic Hotel reviews [97].	87%.
Tamchyna and Veselovska [98].	LSTM.	Word2vec: continuous BOW.	SemEval-2016 Task 5 including Arabic Hotel reviews [97].	F1 score: 52.59%.
Al-Smadi et al. [102].	LSTM.	Word2vec and Fasttext for character level embedding.	SemEval-2016 Task 5 : Arabic Hotel reviews [97].	LSTM: 82.6%
Ruder et al. [99].	Hierarchical Bi-LSTMs.	Trained embedding by Leipzig Corpora Collection.	SemEval-2016 Task 5 including Arabic Hotel reviews [97].	LSTM: 80.5%. H-LSTM: 82.8%. HP-LSTM: 82.9%.
Wang and Lu [100].	Bi-LSTM + CRF	Trained embedding by Leipzig Corpora Collection.	SemEval-2016 Task 5 including Arabic Hotel reviews [97].	A-LSTM: 86.5%. SA-LSTM: 86.7%. SA-LSTM-P: 86.9%.
Ponti et al [101].	Bi-LSTM.	Additive SG, Paragraph Vector distributed BOW, FastSent, sequential denoising autoencoder.	SemEval-2016 Task 5 including Arabic Hotel reviews [97].	Weighted F1 score: 86.56%.

It should be noted that no study used GRUs, Bi-GRUs, or hybrid methods for aspect-based classification. In addition, multilingual models, as shown in [98], were not effective in improving the Arabic language sentiment analysis specifically. Moreover, all the studies depended on a unified dataset, demonstrating the need for well-annotated datasets for aspect terms, categories, and sentiment polarity.

3.4.2 Sentence Level Affect Analysis Using RNN

In this part, the studies will be divided according to the type of analysis that is involved, namely, emotion detection, emoji analysis, hate speech detection, and sentiment classification, and they are all different types of affect or emotion detection.

3.4.2.1 Emotion Detection

This subsubection will introduce studies concerned with the detection and classification of emotions such as anger, fear, joy, and sadness using RNN.

Samy et al. [111] proposed using GRU and context-aware GRU architectures to investigate the role of social influence on shaping others' opinions and emotions in the same environment and the effect on the determination of the sentiment polarity. These models extracted contextual information (topics) and used both topic and sentence information to detect multilabel emotions. The context-aware GRU architecture was compared with a support vector classifier with a linear kernel, L1 regularization from [132], and a context-free GRU architecture. The results demonstrated that the context-aware GRU architecture

achieved an accuracy of 53.2%, a macro-average F1 of 64.8%, and a micro-average F1 of 49.5%, outperforming the simple GRU architecture, which achieved an accuracy of 52.4%, a macro-average F1 of 64.2%, and a micro-average F1 of 49.6%.

Abdullah and Shaikh [113] presented TeamUNCC's system, which used an LSTM network to detect emotion intensity or sentiment in English, Arabic, and translated Arabic tweets from SemEval 2018. The system attempted to complete all five subtasks and determine the intensity and sentiment of the tweets from SemEval 2018. The tweets were preprocessed and fed into word2vec, doc2vec, and other feature vectors for feature extraction. Subsequently, these vectors were fed into the deep neural network layers for prediction. The Spearman correlation scores demonstrated that the proposed system yielded promising results, with an emotion regression score of 59.7%, an emotion classification score of 51.7%, a sentiment regression score of 77.3%, and a sentiment classification score of 74.8%, compared with the baseline model by the SemEval Task 1 organizers, which is based on SVM-Unigrams and achieved an emotion regression of score 45.5%, an emotion classification score of 31.5%, a sentiment regression score of 57.1%, and a sentiment classification score of 47.1%.

Abdullah et al. [114] presented a new version of TeamUNCC's system, called SEDAT, to explore the emotional intensity and sentiment for Arabic tweets. This model consisted of two submodels: the first used a collection of Arabic tweets with five dimensions and translated tweets with 4,903 dimensions using a set of features to produce vectors, and the other used only Arabic tweets with 300 dimensions, which are trained using SG-Twitter from AraVec to produce vectors. The first submodel passed generated vectors to a fully connected neural network, whereas in the second submodel, the vectors were fed into a CNN-LSTM model. The output layer consisted of one sigmoid neuron that produced a real-valued number between 0 and 1. The Spearman correlation scores demonstrated that SEDAT outperformed TeamUNCC's system [113] with an emotion regression score of 66.1%, an emotion classification score of 56.9%, a sentiment regression score of 81.7%, and a sentiment classification score of 78.6%; moreover, SEDAT is only 1 to 2 points behind the state of the art models, that is, AffecThor for emotion (58.7%) and EiTAKA for the sentiment (80.9%) [112].

Abdou et al. [116] presented AffecThor, which consists of three different models. These models used learned and manually crafted representations, such as character embedding, word embedding, inference, and average lexicon representations. The proposed models used a feedforward neural network or gradient boosted trees for regression and two ensemble regressors: simple averaging cross-validation and a nonlinearity (sigmoid) layer on top of the different submodels, such as CNN and Bi-LSTM. The use of simple averaging provides the best results for all the SemEval 2018 models in the Arabic language, particularly in emotion classification, with an emotion classification score of 58.7%.

Alhuzali et al. [115] described and comprehensively confirmed a technique for the mechanical reproduction of labeled emotion information; an improved dataset was also used to extract emotions from modern and dialectical Arabic text, which focused on Robert Plutchik's eight-core emotions. Using a mixed supervision method that exploits the seeds of first-person feelings, it was also demonstrated that promising results can be obtained through an RNN with deep gates. Alhuzali et al. extended the manually annotated dataset DINA [117] to LAMA. The proposed dataset was based on emotion existence and intensity

in one stage for English and Arabic. The proposed approach is based on emotion phrase seeds from Robert Plutchik's eight basic emotional types: anger, anticipation, disgust, fear, joy, sadness, surprise, and trust. In addition, the authors proposed a hybrid supervised method that automatically determines emotion intensity and sentiments for English and Arabic, and they compared it with supervised and distant supervised methods. The proposed method was applied to baseline models, such as a multinomial NB classifier, a passive-aggressive classifier, a perceptron classifier, and SVM classifier, an SVM classifier trained with stochastic gradient descent, and the proposed GRU classifier. In addition, the models were validated using SemEval-2018 [112]. The results demonstrated that the GRU model yielded the best results, particularly in terms of emotion detection, on all datasets, particularly the DINA and LAMA-DINA datasets, with an F1 score of 98%, compared with SVM with stochastic gradient descent, which achieved an F1 score of 92%. Moreover, hybrid supervision of the LAMA-DINA and LAMA-DIST datasets using the GRU model yielded the best results in terms of emotion classification, with an average F1 score of 70% compared with SVM with stochastic gradient descent, which achieved an F1 score of 60%. Table 3.11 shows the emotion detection models for Arabic text.

Table 3. 11 Sentence/Document/Word level sentiment analysis Based on Emotion Detection for Arabic Text

Papers	Classifier	Sentiment Classification Level	Document/ Text Representation	Dataset	Accuracy
Samy et al. [111].	GRU, context aware GRU.	Sentence level.	AraVec [32].	From SemEval-2017 [109] and SemEval-2018 [112].	GRU: 53.2%. C-GRU: 52.4%.
Abdullah and Shaikh [113].	Dense and LSTM networks.	Sentence and document level.	Word2vec, doc2vec with different feature vectors.	From SemEval-2018 (Task 1: Affect in Tweets)[112]: row Arabic and translated Arabic into English.	Emotion detection: 44.6%. The Spearman correlation scores: Emotion classification: 51.7 %. Sentiment classification: 74.8%.
Abdullah et al. [114].	CNN-LSTM	Sentence and document level.	AraVec [32], doc2vec, and a set of semantic features.	From SemEval-2018 (Task 1: Affect in Tweets) [112]: row Arabic and translated Arabic into English.	The Spearman correlation scores: Emotion classification: 56.9%. Sentiment classification: 78.6%.
Abdou et al. [116].	CNN, Bi-LSTM	Sentence and word level.	Word2vec SG embeddings.	From SemEval-2018 (Task 1: Affect in Tweets) [112].	The Spearman correlation scores: Emotion classification: 58.7%. Sentiment classification: 75.2%.
Alhuzali et al. [115].	GRU.	Sentence level.	Word2vec.	LAMA, DINA [117], , DIST and SemEval-2018 [112].	LAMA-DINA+LAMA-DIST: F1 score: 70%.

As noticed in Table 3.11, there is a lack of research on emotion detection and analysis using neural nets. Moreover, it is noted that there is a dominant dataset that is used for emotion analysis studies, namely, SemEval-2018 (Task 1: affect in Tweets).

3.4.2.2 Emoji Analysis

This subsubsection introduces studies on emoji analysis using RNNs. Emoji are ideograms and smileys that are used frequently on social media to express ideas and emotions. They differ from emoticons, which use letters, numbers, and symbols to create icons.

Al-Azani and El-Alfy [105] used deep RNN models, namely, LSTM, Bi-LSTM, GRU, and Bi-GRU, with different modes (summation, multiplication, concatenation, and the average

of outputs) for emoji-based tweets to detect sentiment polarity and compared these models with deep neural networks and baseline machine learning classifiers, such as stochastic gradient descent, Gaussian NB, SVM, k-nearest neighbors, and decision tree classifiers. These models used a set of 843 Arabic microblogs with emojis from different resources, such as the ASTD [103], ArTwitter [53], QCRI [37], Syria [108], and SemEval-2017 Task 4 Subtask A [109]; in addition, data were collected from Twitter and YouTube and were manually annotated. They used the emoji sentiment ranking lexicon to identify emojis in the dataset. The performance was evaluated in terms of the precision, recall, F1 score, accuracy, Matthews correlation coefficient, and geometric mean. The results demonstrated that the LSTM and GRU models significantly outperformed the other models. Specifically, the bidirectional GRU performed best, with an accuracy of 78.71% and an F1 score of 78.76%, compared with Bi-LSTM, which had an accuracy of 77.99% and an F1 score of 78.10%.

Baccouche et al. [106] proposed an automatic labeling technique for health-related tweets in three different languages: English, French, and Arabic. In addition, they applied sentiment analysis models such as a CNN model and an LSTM model to classify tweets into two and three classes. The dataset contained both a domain-specific health-related dataset and a nonspecific domain dataset from Amazon, IMDB, Yelp, and ASTD. The automatic labeling technique preprocessed the health-related tweets to detect the annotations using domain knowledge, NLP, and sentiment lexicon dictionaries. The proposed annotation models yielded the best results using an RNN, particularly for English, with an accuracy of 98% and an F1 score of 97%, compared with the CNN, which had an accuracy of 97% and an F1 score of 96%. Moreover, the proposed model improved by 1.1% when a nonspecific domain dataset was added to the LSTM model.

Heikal et al. [3] presented a model that combined CNN and Bi-LSTM to predict the sentiment of Arabic tweets using the ASTD dataset. This model followed the same method as in a previous study for English [133], using various hyperparameters to improve the accuracy and handling emoticons by mapping them to the related emojis. Although the proposed model did not use any feature engineering to extract special features, it achieved the best results, with an accuracy of 65.05% and an F1 score of 64.46% compared with the state-of-the-art model from Baly et al. [134], which achieved an accuracy of 58.5% and an F1 score of 53.6%. Table 3.12 summarizes the studies that focus on emoji analysis for Arabic text.

Table 3. 12 Emoji Analysis for Arabic Text

Papers	Classifier	Document/Text Representation	Dataset	Accuracy
Al-Azani and El-Alfy [105].	LSTM, Bi-LSTM, GRU, Bi-GRU.	Emoji sentiment ranking lexicon.	Combined datasets from ASTD [103], ArTwitter [53], QCRI [37], Syria [108], Semeval-2017 Task4 Subtask#A [109] and other resources	Bi-GRU: 78.71%. Bi-LSTM: 77.99%.
Baccouche et al. [106].	LSTM.	Word2vec. (Wikipedia).	Health-related dataset (authors didn't mention the final number), and non-health-related dataset from Amazon, IMDB, Yelp, and ASTD dataset [103].	83%.
Heikal et al. [3].	CNN-Bi-LSTM.	AraVec [32].	ASTD dataset [103].	CNN-LSTM: 65.05%.

Only three studies performed emoji analysis; the best accuracy was 83%, which highlights the need for more research in this direction. As emojis are used paradoxically, a robust model that detects sarcasm and the actual sentiment is required. In addition, there is a scarcity of datasets containing emojis. Accordingly, more datasets need to be created along with emoji detection algorithms.

3.4.2.3 Hate Speech Detection

Hate speech is a type of public speech that encourages hate or violence towards a person or a community based on a trait. Only one study [122] focused on hate speech detection in Arabic. This is a fairly new topic, and the study introduced the first dataset that can be used to address this issue.

Albadi et al. [122] provided the first dataset for detecting religious hate speech in Arabic tweets. It consists of 6,000 labeled tweets [135]. Moreover, they created the first three Arabic lexicons consisting of common terms in religious discussions, with scores that describe the polarity and strength of these terms. Moreover, they developed three different approaches to capture religious hate speech: a lexicon-based approach using three Arabic lexicons, an n-gram-based approach using logistic regression and SVM, and a deep learning -based approach using GRU with AraVec embedding[32]. The GRU model achieved the best results compared with all the other models, with an accuracy of 79%, and F1 score of 77%, and an area under the receiver operating characteristic curve of 84%. The second-best model (SVM) had an accuracy of 75%, and F1 score of 72%, and an area under the receiver operating characteristic curve of 81%.

Since detecting hate speech is an important feature of controlling spam and arguments in public forums. Hence, more studies must be conducted to analyze and detect hate speech. Additionally, the existing method has a low accuracy of only 79%, which cannot be considered significant. Therefore, higher accuracies must be targeted to effectively improve the detection of hate speech.

3.4.2.4 Sentiment Classification

In this subsubsection, I divide the related studies into two parts. First, studies that used LSTM, GRU, or both. Second, hybrid models, where an additional neural network was used as a layer in the model along with an RNN such as CNN. They are presented in the following.

LSTM/GRU Models. Abbes et al. [95] proposed two models based on deep learning sentiment analysis: a deep neural network and an RNN for Arabic social media. The proposed models followed several steps. The first stage was gathering and collecting data from the LABR [94]. The second stage was the preprocessing and building of a lexicon model. The third phase was the feature extraction step, consisting of deriving the lexicon-based relevant features from the stored data using word embedding. The final steps involved applying deep learning models, that is, deep neural network and RNN (LSTM), and classifying sentence polarity. The experimental results demonstrated that RNN outperformed deep neural network, with an accuracy of 71%, precision of 68.3%, recall of 77%, and F1 score of 72.4%, whereas deep neural network had an accuracy of 64.4%, the precision of 61.1%, recall of 75.3%, and F1 score of 67.5%. Although the comparison made using 300 epochs for deep neural network and only 30 epochs for RNN was unfair, the highest result was at epoch 200 for deep neural network and epoch 12 for deep neural network.

Alwehaibi and Roy [118] presented LSTM to classify Arabic texts with different pre-trained word embedding techniques, namely, AraVec, ArabicNews, and Arabic Fasttext to investigate the effect of these techniques on the accuracy of the model. Initially, the model preprocessed the AraSenTi tweet datasets. Subsequently, the tweets were processed by a pre-trained word embedding technique to generate vectors for each word. Then, the embedding was fed to the LSTM layer with a 128-dimensional hidden state to classify each tweet into positive, negative, or neutral. Arabic Fasttext achieved the best accuracy (93.5 %) compared with AraVec (88%) and ArabicNews (91%). While the dataset is divided equally for three classes, the low F1 score (43% for ArabicNews, 40% for AraVec, and 41% for Arabic Fasttext) is an indication of both poor precision and poor recall.

Baly et al. [73] provided the first MD-ArSenTD. It contains annotated tweets (for both sentiment and dialect) that were collected from 12 Arab countries from the Gulf, Levant, and North Africa. In addition, the authors analyzed tweets from Egypt and the United Arab Emirates to investigate the regional variation of data characteristics and their effect on sentiment by using feature engineering for SVM and generic and dialect-specific embeddings for LSTM. The results demonstrated the superiority of LSTM over SVM, particularly when lemma embedding was used for Egyptian tweets, with an accuracy of 70.0% and weighted F1 score of 69.1%, compared with UAE tweets, with an accuracy of 63.7% and weighted F1 of 64.8% for 3-class classification.

Elnager et al. [120] presented book reviews in the Arabic dataset BRAD 2.0, which is an extension of BRAD 1.0 with more than 200,000 additional records and a total of 692586 annotated reviews, and it combines both MSA and dialect Arabic with three classes: positive, negative and neutral. In the experiment, only two classes have been used. The authors applied NB, decision tree, random forest, XGBoost, SVM, CNN, and LSTM to verify and validate the proposed dataset. Machine learning classifiers generally performed better using unigram and bigram, especially without TF-IDF, while SVM achieved the best

result with TF-IDF with an accuracy of 90.86%. Using deep learning models, the results show that LSTM outperforms CNN, especially when using GloVe, with an accuracy of 90.05% compared with CNN with an accuracy of 90.02%. However, an imbalanced classified dataset leads to a bias for a positive class; 64% of data have been classified as positive, 15% as negative, and 21% as neutral.

El-Kilany et al. [121] presented a model that recognized the sentiment targets from Arabic tweets using two layers: a character and a word2vec embedding layer, and a bidirectional LSTM with a CRF classification layer. The Bi-LSTM used vectors produced by the first layer to generate the features for CRF, which used these contextual features to predict a tag for the current word based on the previous and consequent word tags in the sentence. The results demonstrated that the proposed model achieved higher recall (71.4%) compared with the same model that used only character-embedding without word2vec, which achieved 61.5%. This implies that it discovered more target entities from the tweets and conserved almost the same extraction precision (approximately 73%).

Alahmary et al. [71] proposed deep learning models: LSTM, Bi-LSTM, and SVM for sentiment analysis using a collected dataset of 32063 tweets classified as positive and negative, and used word2vec continuous BOW to generate vectors from the dataset. The results show that the deep learning techniques outperform the SVM algorithm, especially for Bi-LSTM with an accuracy of 94% compared with LSTM and SVM with an accuracy of 92% and 86.4% respectively.

Jerbi et al. [136] explored and compared various classification models based on LSTM for the Tunisian dialect which are characterized by frequent use of code-switching, which is an alternation of at least two linguistic codes in a single conversation. Different RNN models have been explored such as LSTM, Bi-LSTM, deep LSTM, and deep Bi-LSTM. The experimental evaluation showed that the accuracy of 2-LSTM reached 90% which outperformed the latest best-proposed models on TSAC datasets in Mdhaffer et al. [136] with an accuracy of 78%. Table 3.13 indicates the sentence/character-level sentiment analysis using LSTM/GRU for Arabic text.

Table 3. 13 Sentence/Character Level sentiment analysis Using LSTM/GRU for Arabic Text

Papers	Classifier	Sentiment Classification Level	Document/Text Representation	Dataset	Accuracy
Abbes et al. [95].	LSTM.	Sentence level.	TF-IDF, BOW.	LABR [94].	64.4%.
Alwehaibi and Roy [118].	LSTM.	Sentence level.	AraVec [32], ArabicNews [137], and Arabic Fasttext [138].	AraSenTi dataset [78].	Arabic Fasttext: 93.5 %. AraVec: 88%. Arabic news: 91%.
Baly et al. [73].	LSTM.	Sentence level.	Word2vec, lemma embedding, and stem embedding: SG.	MD-ArSenTD.	Egyptian tweets: 70.0% UAE tweets: 63.7%.
Elnager et al. [120].	LSTM.	Sentence level.	GloVe [29], TF-IDF.	BRAD 2.0 [119].	90.05%.
El-Kilany et al. [121].	Bi-LSTM + CRF.	Sentence and character level.	Word2vec and character embedding.	3,000 tweets.	F1 score: 72.6%.
Alahmary et al. [71]	LSTM, Bi-LSTM, and SVM	Sentence level.	Word2vec continuous BOW	32,063 tweets	Bi-LSTM:94% LSTM: 92% SVM 86.4%
Jerbi et al. [136]	Stacked LSTM+ Stacked Bi-LSTM	Sentence level	Embedding layer	13,655 reviews from the TSAC [139]	2-LSTM: 90%

Hybrid Models. Al-Azani and El-Alfy [104] compared various CNN and LSTM approaches for sentiment analysis of Arabic microblogs using six models: LSTM, CNN, CNN with LSTM, three-stacked LSTM layers, and two LSTMs combined with summation, multiplication, and concatenation. These models were evaluated for Arabic sentiment analysis using four evaluation measures: precision, recall, accuracy, and F1. Two benchmark Arabic tweet datasets were used: ASTD [103] and Arabic sentiment analysis ArTwitter [53]. Word2vec was used as input to the investigated models, with static and non-static word initialization for continuous BOW and SG word embedding. The experiments demonstrated that using word2vec vectors updated during learning achieved the best results in nearly all cases. In addition, LSTM outperformed CNN. Moreover, non-static models with the combined LSTM architectures performed better than other models, particularly when two LSTMs were combined with concatenation along with ArTwitter dataset and with SG word embedding. The results demonstrated that precision reached 87.36%, recall 87.27%, accuracy 87.27%, and F1 score 87.28% compared with the same architecture for the ArTwitter dataset and with continuous BOW word embedding, which had a precision of 86.46%, recall of 86.45%, the accuracy of 86.45%, and F1 score of 86.45%.

Alayba et al. [107] presented a model that combined a CNN and LSTM for sentiment classification at the character, character 5-gram, and word levels to expand the number of features, and they investigated its accuracy using different datasets such as main and sub-Arabic Health Services Dataset [36], ArTwitter [53], and ASTD [103]. Moreover, the proposed model was compared with those by Alayba et al. [33], Abdulla et al. [53], and Dahou et al. [140]. Although the model by Dahou et al. achieved the best results for the ASTD dataset, with an accuracy of 79.07%, the proposed model achieved a promising accuracy, particularly with at the character 5-gram level and with the sub-Arabic health dataset, with an accuracy of 95.68%, compared with the word-level model on the main Arabic health dataset, with an accuracy of 94.24%. The high accuracies are due to the unbalanced datasets for main and sub–Arabic Health Services Dataset. Although the character, character 5-gram, and word level reached results of 74.19%, 77.62%, and 76.41% respectively for ASTD dataset, the proposed model from [104] showed the best results with LSTM-MUL and an accuracy of 81.63% with continuous BOW, which shows the importance of word-embedding techniques as compared to the row-embedding techniques.

Gonzalez et al. [110] proposed the ELiRF-UPV system for enhancing Task 4 of SemEval-2017. It involved five different subtasks: (A) message polarity classification, (B) and (C) tweet classification according to a two and five-point scale, respectively, and (D) and (E) tweet quantification according to a two and five-point scale, respectively. The proposed model used a CNN and an RNN and the combination of general and specific word embeddings with polarity lexicons using both English and Arabic. The system used three CRNNs (including a CNN, max pooling, and Bi-LSTM) to extract spatial relations among the words of a sentence with an input of three different embeddings: out-domain embeddings, in-domain embeddings, and sequences of word polarities. The outputs of these networks were used as an input to a fully connected multilayer perceptron. Different measurements were made to evaluate each task. Regarding subtask A, the proposed model using Arabic achieved an accuracy of 50.8% compared with English, where accuracy of 59.9% was obtained.

Baniata and Park [96] proposed a combined CNN and Bi-LSTM in two different models using Arabic. The first model consisted of three convolutional layers with filter sizes 3, 4, and 5 followed by the average pooling Bi-LSTM layer. The third layer was a merged layer that was connected to a fully connected layer including a multi perception layer and a sigmoid classifier. The second model was an inversed model starting with the Bi-LSTM layer, which was connected to three convolutional layers with filter sizes 3, 4, and 5 followed by a fully connected layer, which was a sigmoid classifier. The results demonstrated that the CNN-Bi-LSTM achieved better sentence feature representation, with an accuracy of 86.43% compared with Bi-LSTM-CNN, which had an accuracy of 66.26%.

Abu Kwaik et al. [141] investigated deep models for dialectal Arabic sentiment analysis by combining LSTM with CNN, and compare them with a different combination of LSTM and Bi-LSTM, and the Kaggle winner model[4] as a baseline model. The author used Arabic sentiment datasets: LABR, ASTD, and Shami-Senti [142] with different sizes and different dialects. The model achieves an accuracy of 93.5% for binary classification and 76.4% for three-way classification especially with Shami-Senti, focusing on ASTD, the accuracy reached 68.62% for three-way and 85.58% for binary classification, the proposed model

[4] https://www.kaggle.com/monsterspy/conv-lstm-sentiment-analysis-keras-acc-0-96.

has beaten the proposed model from [104] being the best model for ASTD for binary classification.

Elshakankery and Ahmed [143] proposed a hybrid approach named HILATSA that combines both lexicon-based and machine learning approaches to identify the tweets' sentiments polarities. in addition, essential lexicons have been built such as words lexicon, idioms lexicon, emoticon lexicon, and special intensified words lexicon along with intensification tools and the negation tools. Three models: Logistic Regression, RNN, and SVM are used to evaluate the proposed approach with six different datasets: ArTwitter, ASTD, mini ASTD, Syrian Tweets Corpus, ArSAS [126], and Arabic Gold Standard Twitter Data for Sentiment Analysis [125]. Focusing on ArTwitter, the results showed improvement before and after the lexicon update, RNN achieved the highest accuracy before and after the learning phase with an accuracy of 68.45% before the learning phase and an accuracy of 85% after the learning phase.

Barhoumi et al. [144] compared several specific embeddings (word, token, token\clitics, lemma, light stem, and stem) in the Arabic sentiment analysis framework, and evaluated them with two neural architectures: CNN ad CNN-Bi-LSTM using unbalanced LABR dataset for polarity classification. Results show that CNN outperforms CNN-BiLSTM. The best system CNN+lemma w2v achieves 91.5% of accuracy compared with CNN-BiLSTM+lemma w2v with an accuracy of 91%, showed that lemma is the best lexical unit for sentiment analysis.

Rehman et al. [145] proposed LSTM with very deep CNN for sentiment analysis with IMDB movie review and Amazon movie reviews dataset, taking the advantages of the CNN and LSTM model in extracting local features and long-distance dependencies. The results showed that the proposed Hybrid CNN-LSTM Model outperformed traditional deep learning and machine learning techniques and achieved 91% in accuracy as compared to traditional machine learning and deep learning models, especially for LSTM and CNN that gained an accuracy between 85% and 90%.

Table 3.14 indicates the sentence/character-level sentiment analysis using Hybrid networks for Arabic text.

Table 3. 14 Sentence/Character Level sentiment analysis Using Hybrid Networks for Arabic Text

Papers	Classifier	Sentiment Classification Level	Document /Text Representation	Dataset	Results
Al-Azani and El-Alfy. [104].	CNN, LSTM, CNN-LSTM, Stacked-LSTM, Combined-LSTM-SUM, Combined-LSTM-MUL, Combined-LSTM-CONC.	Sentence level.	Word2vec.	Two datasets of Arabic tweets: ASTD [103] and ArTwitter [53].	LSTM-CONC with ArTwitter and SG: 87.27%. LSTM-MUL with ASTD and continuous BOW: 81.63%.
Alayba et al. [107].	CNN-LSTM.	Character level, character 5-gram level, and word level.	Row embedding is based on the level.	Arabic Health Services Dataset [36], ArTwitter [53], and ASTD [103].	Main-AHS: 94.24%. Sub-AHS: 95.68% ArTwitter: 88.10% ASTD: 77.62%.
Gonzalez et al. [110].	CNN-Bi-LSTM.	Sentence level.	Word2vec.	SemEval-2017 Task 4 [109].	(task A): Arabic: 50.8%.
Baniata and Park [96].	CNN-Bi-LSTM, Bi-LSTM-CNN.	Sentence level.	polyglot [146].	LABR [94].	CNN-Bi-LSTM: 86.43%. Bi-LSTM-CNN: 66.26%.
Abu Kwaik et al. [141]	stacked Bi-LSTM-CNN	Sentence level.	AraVec.	LABR, ASTD, and Shami-Senti.	Shami-Senti: 3-class: 76.4% 2-class: 93.5%
Elshakankery and Ahmed [143]	LR, RNN, SVM.	Sentence level.	Built feature vectors based on words' weights and the tweet structure.	ArTwitter, ASTD, MASTD, ArSAS and Arabic Gold Standard Twitter Data for Sentiment Analysis.	Before the learning phase: RNN: 68.45% After the learning phase: RNN: 85%
Barhoumi et al. [144]	CNN, CNN-Bi-LSTM	Character level, word level, sentence level	Word2vec, Char-n gram	LABR	CNN:91.5% CNN-BiLSTM: 91%
Rehman et al. [145]	Stacked CNN and LSTM	Sentence level	Word2vec	IMDB and Amazon movie reviews	CNN-LSTM: 91% LSTM: 85%-90% CNN: 85%-90%

As noticed, no research has been conducted on Arabic sentiment analysis using GRU, Bi-LSTM, or Bi-GRU as a stacked model for comparison with stacked models using LSTM. Moreover, using CNN as the first layer could enhance accuracy. Although there is a lack of character-level analysis, there are promising results. Moreover, Arabic pre-trained word embedding strongly affects the analysis, which requires a huge corpus for training the embedding model at the word and character level. Moreover, using a word embedding concurrently with character embedding can enhance the accuracy as shown in [121].

3.5 Stacked Recurrent Neural Networks for Sentiment Analysis

The related works that use Stacked RNNs, LSTMs, or GRUs for sentiment analysis especially for sentence-level analysis will be presented. The tree-structured representation models were discarded because the focus of the thesis is about the notion of the stack, and the tree-structure is more relevant to the recursive network. There are only 19 papers in this field, three of those papers are using the Arabic language proposed by Al-Azani and El-Alfy. [104], Jerbi et al. [136] and Abu Kwaik et al. [141].

Irsoy and Cardie [47] proposed the first deep RNN for opinion mining, specifically detecting direct subjective expressions and expressive subjective expressions. The authors used two performance metrics: Binary and Proportional overlap with three performance evaluations: precision, recall, and F1 score. Focusing on Bi-RNN, the 3-layer deep Bi-RNN for direct subjective expressions and expressive subjective expressions outperform conventional CRFs with results of with F1 score of 71.72% for binary overlap compared with CRF with F1 score of 64.45% (for direct subjective expressions), and with results of with F1 score of 67.18% for binary overlap compared with CRF with F1 score of 58.85% (for expressive subjective expressions).

Zhou et al. [147] proposed a stacked Bi-LSTM to analyse Chinese microblogs. The authors first pre-processing comments constructed from Weibo and apply word representation using word2vec models: continuous BOW and SG. Next, they used the Stacked Bi-LSTM model to conduct the feature extraction of sequential word vectors. Finally, they apply a binary *Softmax* classifier to predict the sentiment polarity. The proposed model was compared with baseline models: SVM, LR, CNN, stacked CNN, LSTM, and Bi-LSTM. The results showed that Stacked Bi-LSTM gives promising results with an accuracy of 90.3% with continuous BOW and an accuracy of 89.5% with SG. Moreover, as observed by authors, increasing the number of layers up to 4 layers leads to an increase the prediction accuracy and decreases prediction loss, that is because an increasing number of layers leads the model to extract more features.

Xiao and Liang [148] proposed Bi-LSTM with word embedding for Chinese sentiment analysis. The author used corpus consists of 13000 reviews, tagged as positive and negative data. the author used the CRF-based model, LSTM, 2-layer LSTM, and 2-layer Bi-LSTM to evaluate the model. Experimental results show that the model achieves 91.46% accuracy compared with using 2-layer Bi-LSTM with a result of 90.28% for the sentiment analysis task. In this study, increasing the layers didn't lead to better results due to changing the number of hidden units for each model. The comparison would be fair with fixed hidden units during increasing layers.

Wen and Xu [149] proposed a sentiment classification model that combines a residual Bi-LSTM and a wide word embedding network architecture instead of one embedding vectors,

since one pre-trained embedding vectors in the residual model may restrict the feature space. First, they train more than one word-embedding vectors: Glove, Word2Vec, and Crawl, to obtain a larger feature space; Second, they use Bi-LSTM models and employ residual learning to ease the deep training of networks. the proposed model compared with Bi-LSTM, Deep Bi-LSTM, and Residual Bi-LSTM using two datasets: Crowdflower Twitter Airline Sentiment and Jigsaw Toxic Comment Classification datasets for binary classification. The results show that increasing the number of embeddings leading to better results especially with the Crowdflower Twitter Airline Sentiment dataset, with an accuracy of 92.12% for three embeddings and an accuracy of 90.82% with two embeddings. However, the datasets are unbalanced since the positive samples are much higher than negative samples for both datasets. Moreover, the testing sample is 10% thus increase the accuracy percentage. The increasing of layers leads to higher accuracy regardless of increasing the embeddings (87.36% for Bi-LSTM and 88.23% for deep Bi-LSTM).

Gao and Chen [150] implemented a sentiment ordinal classification system that is incrementally learned with five sub-models based on Bi-LSTM: 1- A (three-class) polarity classification model. 2- A negative model to discriminate negative ordinal classes. 3- A neutral model for neutral/non-neutral classification. 4- A positive model to discriminate positive ordinal classes. 5- An (seven-class) ordinal classification model. The weighted average and stacking techniques have been used on the proposed systems. The proposed methods are applied to the Sem Eval 2018 Task 1 Affects in Tweets Subtask V-of (ordinal classification task) with five-word embedding models based on two algorithms: GloVe and SG, in addition, to apply ensemble methods to combine their outputs to boost overall performance. The proposed method with weighted average and pretrained DeepMoji has ranked the 4th in the SemEval-2018 with a Pearson correlation coefficient of 80.6% compared with the stacking technique with the Pearson correlation coefficient of 77%. Although the weighted average technique gained the highest correlation, the stacking technique achieves the highest accuracy of 46.1 % compared with the weighted average technique, with an accuracy of 41.9%

Sakenovich and Zharmagambetov [151] proposed a sentiment analysis model of the news articles based on stacking LSTMs in Kazakh and Russian languages. Different stacking LSTMs have been proposed to discover the best feature representation. The dataset consists of around 30,000 news articles in the Russian language + 10000 news articles in the Kazakh language with a balanced split, labeled as positive, negative, neutral. The authors used word2vec, GloVe for feature selection. The results showed that the best stacking representation was by stacking 2 LSTMs + neural network Layer, especially for the Russian dataset, with an accuracy of 86.3 % compared with the Kazakh language with an accuracy of 69.8 %. Although the non-availability of the morphological lemmatization tool causes the expected result. The authors concluded that by using stacked LSTMs, "good results can be achieved even without knowing linguistic features of particular language".

Nguyen et al.[152] implemented a deep Bi-LSTM for learning sentence level presentation using character-level as input. The first layer operates at character-level input and the last layer makes predictions at the Tweet-level whether it is of positive or negative sentiment. The proposed model by authors achieves 85.86% accuracy on Stanford Twitter Sentiment corpus and 84.82% accuracy on the subtasks B of SemEval-2016 Task 4 corpus. Moreover, deep Bi-LSTM outperforms Bi-LSTM especially with character-level analysis (82.87% for Bi-LSTM and 83.08% for deep Bi-LSTM).

Ma et al. [153] proposed stacked Bi-GRU with label inference, to address target extracting for target-based sentiment analysis, since the character-level features and context features play important roles in target extraction, and represent each word by concatenating word embedding and character-level representations which are learned via character-level Bi-GRU. Experimental evaluation on two open-domain datasets with 30,000 Spanish tweets and 10,000 English, all are labelled for named entities from Mitchell et al. [154] showed that the proposed model outperformed CRF-based approaches with an F1 score of 56.58% for English language and 66.05% for the Spanish language, compared with Discrete model with an F1 score of 43.8% for English language and 57% for the Spanish language. In addition, focusing on SBi-GRU, the model outperforms the baseline Bi-LSTM (47.72% for Bi-GRU and 48.06% for SBi-GRU). Moreover, using character-level features with SBi-GRU boost the performance (55.61%), character-level features are important to target extracting because character-level features include morphological characteristics and grammatical features.

Chen et al. [155] presented a study that attempted to develop a community sentiment analysis process based on deep learning and different experimental designs using sentiment dictionaries and model parameter setting (including activation function and network layer selection) and use these settings for building multiple types of sentiment analysis models and exploring better learning mechanisms through evaluation indicators. The authors used the Military life PTT board of Taiwan's largest online forum as a source of the evaluation. The results showed that using LSTM and Bi-LSTM led to enhance accuracy, especially when using 2 layers and when using activation function Tanh. The accuracy reached 90.11% for 2-LSTM and 92.68% for 2-Bi-LSTM.

Pal et al. [156] presented a different type of LSTM architectures and stacking to discover their effectiveness in the field of sentiment analysis for movie reviews. The authors hereby showed that a layered deep LSTM with bidirectional connections has better performance in terms of accuracy compared to the simpler versions of LSTMs, the 3-LSTM model reached an accuracy of 81.32% compared with 3-Bi-LSTM with an accuracy of 83.83% with loss decrement for both models.

Hong and Fang [157] reviewed and implemented several methods such as Average of Word Vectors, paragraph vectors, LSTM, and deep recursive- neural networks, and evaluate these algorithms on several sentiment-labelled datasets: movie review and SST. focusing on the SST dataset. A deep recursive neural network achieved a test accuracy of 84.7% compared with 3-LSTM with a test accuracy of 84.3% for binary classification which shows similar performance.

Xie [158] described CNN and Bi-GRU with different stacking methods for DSE and ESE tasks. Different layers of CNN and GRU have been used to discover the best representative model. One or more CNN layer extract local contextual features from embedded vectors of each token. The features are then fed into one or more Bi-GRU layers that mine semantic information and extract global contextual features for each token. Finally, an output layer predicted the labels for each token. The results showed that the proposed model outperforms traditional methods like CRF and state of the art 3-RNN model in some metrics. Although 3- layers RNNs achieved better recall at DSE task, the F1 score for 2-layers CNN + 2-layers GRU outperformed 3-RNN with a result of 72.12% for binary classification compared with 3-RNN with a result of 71.75%.

Wu et al. [159] proposed phrase-level valence-arousal ratings for the Dimensional sentiment analysis for Chinese Phrases task using a densely connected LSTM network and word features such as embedding, POS, and word cluster to identify dimensional sentiment on valence and arousal for words and phrases jointly. The evaluation results showed the effectiveness of the proposed architecture to predict valence and arousal, especially for phrase level, with Pearson correlation coefficient of 96.1% for Valence and 91.1% for Arousal rating.

Another paper from Wu et al. [160] proposed a fine-grained sentiment information model with a multi-task learning strategy for irony detection to achieve the best result in Semeval-2018 task 3, especially for subtask A that aimed to detect the ironic tweets, and subtask B that aimed to detect the ironic types using several types of features such as sentiment features, sentence embedding feature and POS tags feature to improve the model performance. The model achieved an F-score of 70.54% (ranked 2/43) in the subtask A and 49.47% (ranked 3/29) in subtask B.

Anil et al. [161] presented a Memory-based Collaborative Filter Recommendation Systems that gives recommendations to users based on item-item similarities using the information provided by reviews, which is built using deep learning models such as stacked LSTM, stacked GRU, and stacked LSTM-GRU. The experimental evaluation indicated that hybrid LSTM-GRU showed the maximum validation accuracy and the least loss and training time. The accuracy reached 47.91% for stacked GRU, 45.15% for stacked LSTM, and 48.38% for stacked LSTM-GRU.

Feizollah et al. [162] presented a sentiment analysis model based on tweets of two halal products: halal tourism and halal cosmetics in English and Malay languages. A twitter data was extracted filtered, then, an experiment was conducted to calculate and analyse the tweets' sentiment using different stacked models of RNN, CNN, and LSTM. The results showed that that the Word2vec feature extraction method combined with a stack of the CNN+LSTM algorithms achieved the highest accuracy of 93.78%, compared with stacked Bi-RNN + Bi-LSTM with an accuracy of 92.58%

Table 3.15 shows the stacked (deep) RNN models for sentiment analysis.

Table 3. 15 Stacked RNN Models for sentiment analysis

Papers	Classifier	Sentiment Classification Level	Document/ Text Representation	Dataset	Results
Irsoy and Cardie [47]	stacked RNN	Sentence level	Word2vec	MPQA 1.2 [163]	F1(direct subjective expressions): 71.72% F1(expressive subjective expressions): 67.18%
Zhou et al. [147]	stacked Bi-LSTM	Sentence level	Word2vec	3,000 comments	Continuous BOW: 90.3% SG: 89.5%
Xiao and Liang [148]	Bi-LSTM	Sentence level	Word2vec	13,000 reviews from Weibo	Bi-LSTM 91.46% 2-layer Bi-LSTM: 90.28%
Wen and Xu [149]	Residual Bi-LSTM with more than one embedding.	Sentence level	Glove, Word2Vec, and Crawl	Crowdflower Twitter Airline Sentiment: 11,541 tweets Jigsaw Toxic Comment Classification: 95,851 comments	(Crowdflower Twitter Airline Sentiment dataset): Bi-LSTM: 87.36% Deep Bi-LSTM:88.23% Residual Bi-LSTM: Residual Bi-LSTM (two embeddings): 90.82% Residual Bi-LSTM (three embeddings):92.12 %
Gao and Chen [150]	Bi-LSTM	Sentence level	GloVe and SG	From SemEval-2018 (Task 1: Affect in Tweets) [112].	Pearson correlation coefficient: Stacking: 77% weighted average: 80.6%
Sakenovich and Zharmagambetov [151]	Stacked LSTMs	Sentence level	word2vec, GloVe	30,000 news articles in Russian language + 10,000 news articles in Kazakh language	1 LSTM + neural network: 82.8 % 2 LSTMs + neural network: 86.3%
Nguyen et al.[152]	Stacked Bi-LSTMs	Word and Sentence level	Row character embedding	SemEval-2016 Task 4 + Stanford Twitter Sentiment	Stanford Twitter Sentiment corpus: 85.86% SemEval-2016 Task

				corpus.	4: 84.82%
Ma et al. [153]	Stacked Bi-GRU+CNN	Word and Sentence level	Row character embedding	English and Spanish datasets from Mitchell et al. [154]	English: Bi-GRU(F1): 47.72% SBi-GRU(F1): 48.06% SBi-GRU+ char(F1): 55.61%
Chen et al. [155]	Stacked LSTM, Stacked Bi-LSTM	Sentence level	Word2vec (continuous BOW)	17,819 articles	2-LSTM: 90.11% 2-Bi-LSTM: 92.68%
Pal et al. [156]	Stacked LSTM, Stacked Bi-LSTM	Sentence level	Embedding layer	Movie reviews [164]	3-LSTM:81.32% 3-Bi-LSTM: 83.83%.
Hong and Fang [157]	3-LSTM compared with recursive neural networks	Sentence level	GloVe	Movie reviews [164] and SST [50]	Recursive neural networks: 84.7% 3-LSTM: 84.3%
Xie [158]	Stacked CNN+ Stacked Bi-GRU	Sentence level	Embedding layer	MPQA 1.2 [163]	2-layers CNN + 2-layers GRU: 72.12% 3-RNN: 71.75%
Wu et al. [159]	Stacked LSTM.	Word level+ phrase level	Word2vec	SogouCA News dump[165], wiki dump[166] and other 500 collected sentences	Pearson correlation coefficient: Valence: 96.1% Arousal: 91.1%
Wu et al. [160]	Stacked Bi-LSTM	Sentence level	Two pre-trained embeddings from Godin et al. [167] and Barbieri et al. [168]	SemEval-2018 Task3 [169]	F1 score: Subtask A: 70.54% Subtask B: 49.47%
Anil et al. [161]	Stacked LSTM, stacked GRU, stacked LSTM-GRU	Sentence level	GloVe	568,454 reviews of fine foods	stacked GRU: 47.91% stacked LSTM: 45.15% stacked LSTM-GRU: 48.38%
Feizollah et al. [162]	Different stacked models of RNN, LSTM, and CNN	Sentence level	Word2vec	83,647 tweets related to halal tourism and halal cosmetic topics	CNN+LSTM: 93.78% Bi-RNN + Bi-LSTM: 92.58%

As shown in Table 3.15, increasing RNN, LSTM or GRU layers lead to enhance performance. Some models use CNN in the model architecture, and other models use character level analysis to utilize the best stacking representation. Stacking RNN, LSTM, or GRU in the same model lead to enhance the performance in some models [161], [162]. Increasing the layers up to 2 layers is the most efficient way to obtain better results with less training time.

3.6 Transformers

Transformers are a deep neural network architecture specially designed for the NLP tasks introduced in the paper “Attention Is All You Need” [30]. This architecture was proposed as an improvement of the traditional sequential models using recurrent network architecture which was used to capture the temporal information and relationship between the elements of a sequence. In the paper, self-attention was proposed as the main concept to reduce dependency over the recurrent networks which have computation limitations during implementation like the parallelization of the code along with the algorithm's training limitations. The self-attention mechanism was proposed to know the importance of the relationship between the elements of a sequence for a sentence. For example, if I take the sentence “The animal didn't cross the street because it was too tired” using the self-attention approach I can calculate which word has the highest probability to be considered as 'it', either street or animal. Finding these relations is simple for a person but a machine, it is challenging to determine without any proper computation. To find this, in self-attention I use key vector, query vector, and reply vector over which I apply this equation.

3.6.1 Transformers Architecture

Figure 3.7 is the representation of the Transformers architecture, the left and right half for encoder and decoder, respectively [30].

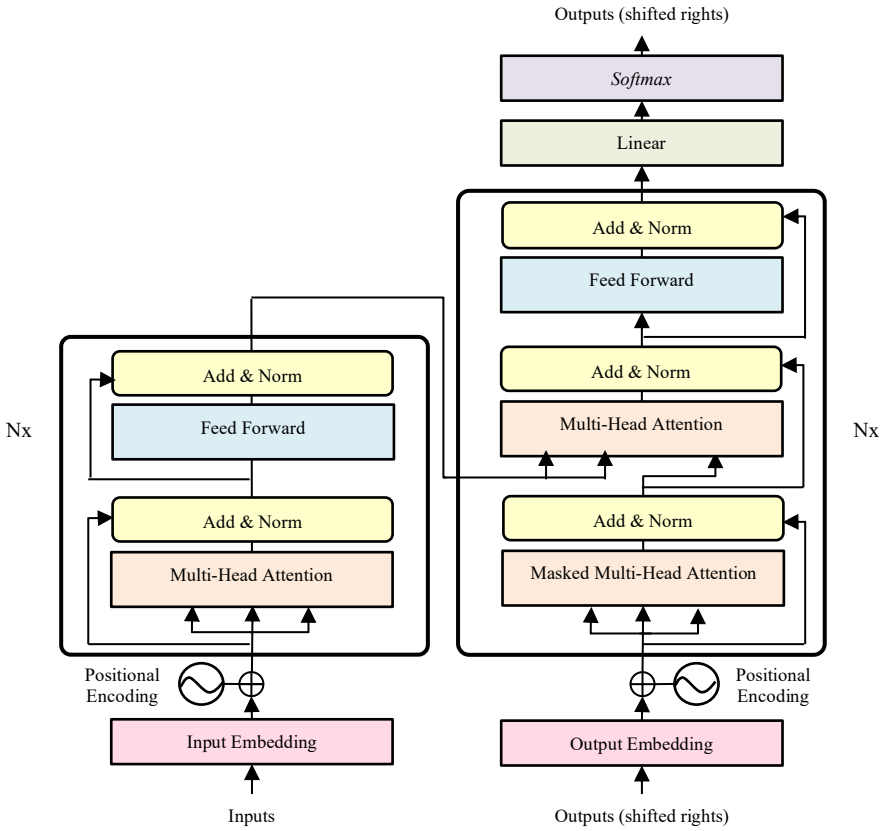

Figure 3. 7 Transformer Model Representation

Figure 3.8 presents the encoder and decoder process.

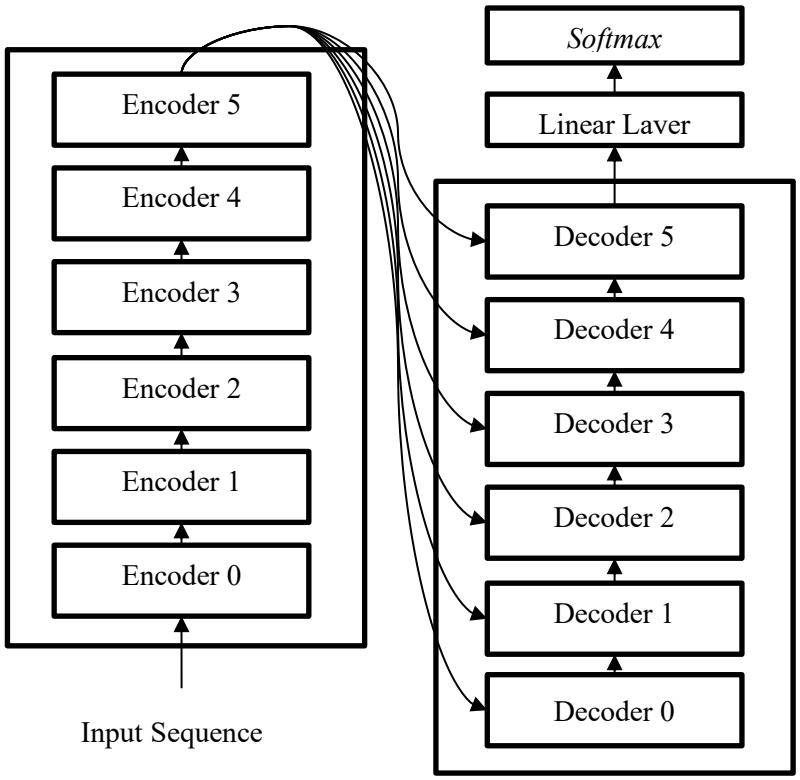

Figure 3. 8 Encoder and Decoder Process

Figure 3.7 and Figure 3.8 shown has two sections, Encoder, and Decoder.

Encoder: The encoder consists of 6 layers stack (N=6), where N is the number of layers. There are two sub-layers in each row. The first is a multifunctional self-attention system, and the second is a basic feed-forward network that is completely linked. A residual connection was used, accompanied by layer normalization, on each sub-layer [30]. In other terms, each sub-layer's performance is Layer Norm (x + Sub-layer(x)), where Sub-layer(x) is the sub-layer itself. Both sub-layers in the model and the embedding layers produce outputs of dimensions 512 such that these residual relations can be facilitated [170].

Decoder: The decoder often includes an N = 6 layers buffer (N=6), where N is the number of layers. The decoder adds a third sub-Layer, under which the priority is given to the performance of the encoder row, under comparison to the two sub-layers of each encoder sheet. A residual connection was used in the sub-layer, accompanied by layer normalization, as the encoder. I often adjust the sub-attention in the decoder stack to avoid additional positions from happening. Combined with the fact that the performance embedding is compensated by one position, this masking ensures that position I predictions only depend on the established results in positions below i [30], [170].

Attention: An attention method may be defined as mapping a query and a collection of key-value pairs into an output that includes all vectors such as the input, request, values, and output. The results are determined as the weighted sum of the values, where the weight assigned to each value is measured with the corresponding key in the query consistency feature [170]. After the transformer layer was proposed, many different models were tried

based on this architecture. Here the focus is only on BERT [31] and AraBERT [171]. The AraBERT model has the same architecture as the BERT model; the only difference is that the Arabic data has been used to train the model. The BERT model is an encoder model which has the same architecture as an encoder of a transformer.

The difference between pre-trained representations is confined between two concepts: context-free or contextual. The context-free models generate a single "word embedding" representation for each word in the vocabulary (e.g. word2vec or GloVe) so the same words with different meanings can have the same representation. On the other hand, the contextual models such as BERT generate a representation of each word that is based on the other words in the sentence [172].

In the next subsections, four contextual embedding models that focus on the Arabic language will be discovered. These models were built between 2018-2020 and achieved state-of-the-art performance on Arabic NLP specifically.

3.6.2 BERT

Bi-directional Encoder Representation from Transformers (BERT) [31] is the first multilingual model architecture that makes use of Transformers [30]. It is pre-trained on Wikipedia text from 104 languages and comes with hundreds of millions of parameters. It contains an encoder with 12 Transformer blocks, a hidden size of 768, and 12 self-attention heads. BERT involves two self-supervised learning tasks:

Masked language models: Before feeding sequence to BERT, 15% of the words are replaced with a [MASK] token, BERT attempts to predict the masked words based on the other words in the sequence.

Next sentence prediction: In the training phase, BERT receives pairs of sentences as input and learns to predict if the second sentence in the pair is the subsequent sentence in the original document.

When training the BERT model, masked language models and next sentence prediction are trained together, to minimize the combined loss function of the two strategies and overcome the challenge of defining a prediction goal.

3.6.3 hULMonA

The first Universal Language Model in Arabic (hULMonA) [173] is an Arabic language model that is based on Universal Language Model Fine-tuning (ULMfit) architecture [174]. It is the first Arabic specific universal language model, that can be fine-tuned for almost any Arabic text classification task. hULMonA consists of three main stages: First, train the state-of-the-art language model Average-Stochastic Gradient Descent Weight-Dropped LSTM (AWD-LSTM) [174] on all Arabic Wikipedia to capture the various properties of the Arabic language. Second: fine-tuning the pre-trained general-domain language model on the target task data to adapt to the new textual properties. Third: augment the fine-tuned hULMonA with two fully connected layers with *ReLU* and *Softmax* activations respectively for downstream task classification.

3.6.4 ArabicBERT

ArabicBERT [175] consists of four models of different sizes trained using masked language models with whole word masking [31]. Those models were pre-trained on ~8.2

Billion words from the unshuffled Arabic version of the OSCAR, Arabic Wikipedia, and other Arabic resources which sum up to ~95GB of text.

3.6.5 AraBERT

AraBERT [171] is an Arabic pre-trained language model based on Google's BERT architecture. Two versions of AraBERT (AraBERTv0.1 and AraBERTv1) are available. The difference between these versions is that the v1 uses pre-segmented text where prefixes and suffixes were splatted using the Farasa segmenter. The model has trained on ~70M sentences or ~23GB of Arabic text with ~3B words. The training corpora are a collection of publicly available large scale raw Arabic text (Arabic Wikidumps, The 1.5B words Arabic Corpus [35], The OSIAN Corpus [176], Assafir news articles, and 4 other manually crawled news websites (Al-Akhbar, Annahar, AL-Ahram, AL-Wafd) from the Wayback Machine[5]. Table 3.16 presents the 4 recent models in Arabic.

Table 3. 16 Recent contextual models in Arabic

Model	BERT	hULMonA	Arabic-BERT Base	AraBERT V0.1	AraBERT V1
ArsenTD-Lev dataset	51.0%	51.1%-52.4%	55.2%	58.9%	59.4%

The results show that a language model that is bidirectionally trained can have a deeper sense of language context and flow than single-direction language models. In addition, as shown in Table 3.16, AraBERT achieved state-of-the-art performance, it proves that pre-trained language models on a single language only surpass the performance of a multilingual model.

3.7 Findings

This part answers RQ4 by presenting the findings from the related studies. Based on our findings, Arabic sentiment analysis is a challenging research area with diverse and complicated tasks. The main research directions focus on subjectivity categorization, sentiment categorization, lexicon creating, feature mining, feature sentiment classification, and attitude spam recognition [57]. Using RNNs for sentiment analysis has yielded accurate results, as these networks use previous sequential states to compute the current input, which is suitable for the natural language context [5], [46], [82], [177], and [178].

3.7.1 The Lack of RNN Arabic Sentiment Analysis Studies

I observed the lack of studies that use RNNs for Arabic sentiment analysis compared to other languages, such as English. As of 2018, the number of RNN English-language sentiment analysis studies reached 193, while it was only 24 studies for Arabic sentiment analysis. Moreover, a wider look at multilingual studies shows that in [110], [113], [114], sentiments were analysed in both Arabic and English. However, in [98], sentiments were analysed using seven languages: Arabic, Dutch, English, French, Russian, Spanish, and Turkish, whereas [106] analysed data in three different languages: English, French, and Arabic. This demonstrates that using Arabic in this task is promising and still in its infancy, particularly for sentence-level tasks, where there are only six studies on Arabic aspect-based analysis and five studies on emotion analysis. Samy et al. [111] used GRU and C-GRU, but

[5] http://web.archive.org/

in [115], a hybrid model was developed using supervised algorithms for automatically determining the emotional intensity and feelings for English and Arabic text. GRU was demonstrated to yield the best results, particularly in emotion detection. It was also found that the use of supervised algorithms with GRU yields an average F1 score of 70% on emotion detection compared with 60% by SVM with stochastic gradient descent. There was only one study on hate speech detection [122], and there have not been studies on sarcasm or cyberbullying detection. This demonstrates the need for further research in that direction. Moreover, constructing deeper models may enhance the classification accuracy. For example, Al-Azani and El-Alfy [104] used different architectures of LSTM networks, and the deeper architectures performed better than the simple LSTM. The denser layers being used in the case of deep learning models such as LSTM or GRUs try to obtain more enhanced features and a better relation that is not possible with the usual machine learning and state-of-the-art sentiment analysis algorithms. The time dependency factor of RNNs, as discussed in the previous subsections, helps in determining the meaning of a user's text block or phrase, which makes these algorithms suitable for sentiment analysis.

3.7.2 The Dataset Effect on Accuracy

I observed that 55% of the datasets under study have fewer than 5,000 labeled data samples, 75% of which are unspecified dialects. This observation can be problematic in two ways: first, the small size of the datasets leads to less trustworthy results prone to overfitting, as shown in [107], with accuracies greater than 90%. Second, the undetermined dialects of most of them can result in a waste of effort that can accumulate from labeling each dataset with its dialect. These two observations are limitations to the research prosperity, and it is recommended that they be taken into consideration in future studies. Nevertheless, creating a unified open dataset for each dialect may help achieve fair comparisons. Moreover, creating a well-annotated dataset for different classification domains is required to enhance aspect-based analysis.

A limited number of training and validation datasets have been either generated or organized from the Internet for both aspect-based and sentence-level sentiment analysis. More Arabic datasets must be created and analyzed. All of the existing research on aspect-level sentiment analysis achieved an accuracy of less than 90%; thus, tuning better models is a research opportunity. Hence, retraining the existing models that have been indicated to be superior, and creating new architectures requires development and research. In sentence-level sentiment analysis, there is a lack of studies dealing with the analysis of emojis; only 3 such studies exist. Since emojis are represented syntactically as numbers and symbols, which are discarded in the preprocessing phase to simplify the analysis, emoji analysis holds a high potential of trend and sentiment indication, given the challenge of analyzing the exact sentiment.

3.7.3 RNN Challenges

RNNs have connections between nodes to form a directed graph along a sequence. It is a sequence of neural network blocks linked to each other like a chain. Each message is passed to a successor. This property allows RNNs to exhibit temporal behavior and capture sequential data, which makes it a more natural approach when dealing with textual data such as that of the Arabic language since the text is naturally sequential. RNNs cannot be used to process exceptionally long arrangements on the off chance that it utilizes *tanh* as its initiation function. It is entirely flexible on the off chance that I use *ReLU* as the activation

function.

In the case of GRU, tuning the hyperparameters is a difficult task that is also reflected while trying to change slightly in the already proven architectures. Even though many changes are made, the improvement in accuracy is not significant enough. In the case of LSTM, the computation time is greater than that of the corresponding GRU structure. Hence, a tradeoff needs to be set up while choosing the best algorithm; this tradeoff depends on the problem statement and the kind of data.

3.7.4 Arabic Transformers trend

One of the biggest milestones in the evolution of NLP recently is the release of Google's BERT, which is described as the beginning of a new era in NLP. Unlike previous efforts that looked at the text's sequence either from the right or from the left, BERT is applying the bidirectional training of Transformer, a popular attention model, to language modelling. This type of training can have a deeper sense of language context and flow than single-direction language models. Given the models that are based on BERT [31], [171], [175], AraBERT achieved state-of-the-art performance for the Arabic language. This could be a contribution to enhance the classification of the Arabic sentiment analysis based on the ensemble method using the most recent study.

3.7.5 Future Directions

Only one study (Albadi et al. 2018) explored hate speech detection in Arabic in addition to a unified dataset for this approach. Hate speech detection is one of the important directions of sentiment analysis, as social media makes it easy for anonymized opinions and speeches to be shared, and there is an increasing likelihood of showing hate and aggression without accountability or liability [179]. The existing method has a low accuracy of only 79%, which suffers from a lack of interpretability and hence cannot be considered significant. Therefore, higher accuracies must be targeted to effectively improving the detection of hate speech. For sentiment classification levels, approximately 13 papers considered only sentence-level classification, and 6 papers considered aspect-based analysis. The models from [118], [120] have higher efficiency, while other models yield lower accuracy because of imbalanced datasets and due to focus on achieving higher accuracy, regardless of the precision and recall, which shows that the proposed models are overfitting. The discussed models must be tested on other datasets to check their performance consistency across all the datasets. On the other hand, the hybrid techniques use a mixture of both sentence-level and character-level analysis. In general, the accuracy of hybrid models is higher than that of the component models since they provide different perspectives for analysis, as shown in Table 3.14. Hence, further hybrid models are a research opportunity and are recommended for Arabic sentiment analysis studies. Many research options can be used while implementing these kinds of methodologies. Some of the evident scenarios for success are dealing with the tradeoff between the bias and the variance caused by a mixture of different types of methods. They not only try to fit the model to achieve a better prediction but also try to minimize the overfitting of the model in the training data, which makes the algorithm or the methodology poorly generalizable.

Also, developing pre-trained word and character embeddings enhance the accuracy of sentiment classification, covering the different types of feature engineering, such as part-of-speech, named entity recognition, and handling negation, for different dialects. Based on

the study by [118], character embedding has shown promising results, especially when combined with word embedding, as shown in [102], [116], and [121].

For the stacking architecture in Subsection 3.4.3, increasing RNN, LSTM or GRU layers lead to enhance performance. Some models use CNN in the model architecture, and other models use character level analysis to utilize the best stacking representation. Stacking RNN, LSTM, or GRU in the same model lead to enhance the performance in some models [161], [162]. Increasing the layers up to 2 layers is the most efficient way to obtain better results with less training time.

Table 3.17 presents the reviewed algorithms, their advantages, and their drawbacks in addition to assessments concluded from 30 studies (Section 3.4) under consideration.

Table 3. 17 Types of algorithms that have been reviewed in related works

Model type	**Advantages**	**Drawbacks**	**Assessment**
RNN	It can utilize the same transport function with the same parameters in each phase	Difficulty in training data, high gradients at several phases, and not suitable for an image or tabular datasets.	It is a type of neural networks; it is particularly helpful in translations and sentiment analysis.
SVM	It is most efficient in high-dimensional areas or when the number of dimensions is better than the number of samples, and it is relatively memory-efficient.	It is not suitable for large data sets and is not effective when the dataset contains more noise so that the target groups overlap.	It is one of the most accurate algorithms in sentiment analysis, but it cannot be used to analyse large datasets.
LSTM	One of the most successful RNN algorithms, it overcomes repeated network training problems and is capable of learning long-term dependencies.	It takes long time to train and requires more memory than GRU to train. LSTM is sensitive to different random weight initializations.	The repeating module for LSTM is more complicated. Instead of a single neural network layer, there are four layers interacting in a special manner.
CRF	It is perfect for various segmentation and sequence tagging tasks such as sentiment analysis.	It is computationally complex in the training phase, and retraining is difficult when new training data samples are available.	It is a supervised machine learning algorithm; therefore, it requires a sufficiently large training sample.
GRU	GRU is superior to LSTM, as it can be trained in less time and more effectively. Moreover, it is simple to modify and does not require memory modules.	It cannot extract local context features.	It is an enhanced LSTM algorithm that has been improved in terms of network architecture and effectiveness, but it does overcome the inherent defect of LSTM in capturing local text features.

3.8 Conclusion

This chapter has introduced a systematic review on RNN, Arabic sentiment analysis, and related studies. In addition, an overview of the term RNN, LSTM, and GRU were presented. Then, the related studies that obtained from the systematic review were discussed in Section 3.1. After that, the notion of Transformers was introduced. In addition, a literature review findings and gaps were presented. In the next chapter, the methodology of the proposed model will be presented in detail.

CHAPTER FOUR: METHODOLOGY

4.1 Introduction

In this chapter, I present the models to be implemented and applied. The models to be implemented are GRU, SGRU, and SBi-GRU that contains two or more GRU stacked layers to learn high-level abstractions of sequential features. The performance comparison of different embeddings such as AraVec, Fasttext, and ArabicNews for accuracy and training time will be applied. A comparison of the impact of an increasing number of layers for GRU and Bi-GRU will be applied. Moreover, a comparison between the proposed models and baseline machine learning models such as SVM will be presented. In addition, a comparison between the proposed models and AraBERT, a pre-trained Arabic transformer model based on BERT will be presented. Finally, an ensemble model based on SGRU, SBi-GRU, and AraBERT will be implemented to find the most suitable architecture for analyzing Gulf tweets. The implemented models will be compared in terms of accuracy, precision, recall, F1 score, and loss.

The models under focused are listed in Table 4.1.

Table 4. 1 Models to be Implemented and Applied

Implemented Models	**Baseline models To Be Applied**
SGRU SBi-GRU Ensemble models (based on SGRU, SBi-GRU, AraBERT)	GRU Machine learning model (SVM) AraBERT

4.2 General Model Architecture

The general model architecture is shown in Figure 4.1. The Arabic gulf tweets are first preprocessed to eliminate insignificant characters, smooth noisy data, and normalize inconsistencies. Then, apply AraVec for language modeling and feature selection for the training data. Afterward, different models use the resulted vectors and learn further representations from these vectors such as SGRU and SBi-GRU. Next, a single output produced to conduct sentiment prediction. Hence, after training our model, the testing data is used to investigate the effectiveness of our model in terms of accuracy, precision, recall, and F1 score.

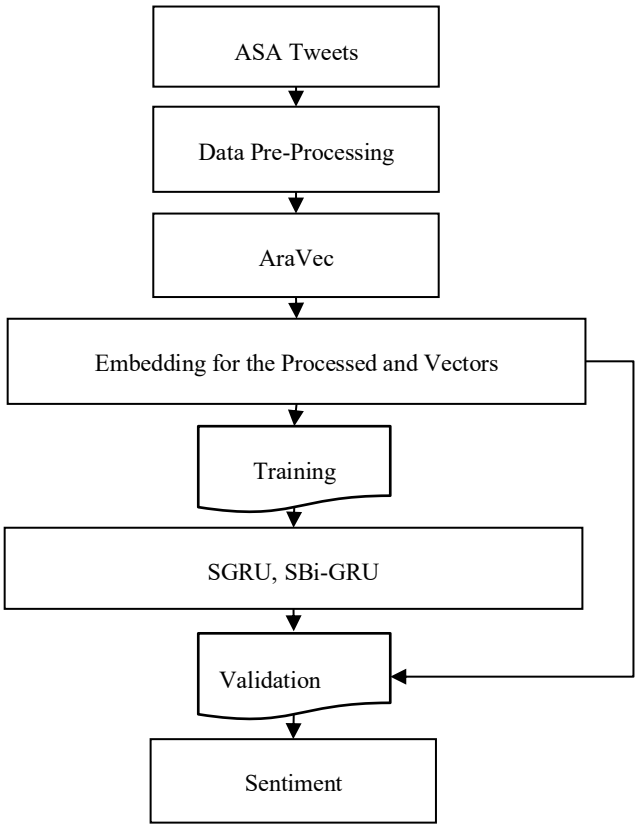

Figure 4. 1 General Model Architecture

4.3 Dataset

I use ASA, the largest annotated gulf dataset provided by Arabic sentiment analysis Research Group [180] at Imam Muhammad Ibn Saud Islamic University. The Arabic Sentiment Analysis (ASA) dataset is freely available through GitHub[6].

The dataset consists of 56,674 tweets labeled into three classes: positive, negative, and neutral with an extremely balanced count, which makes it the largest gulf dataset as shown in Table 4.2.

Table 4. 2 The size of ASA Dataset

	Positive	**Negative**	**Neutral**
No. of tweets	17,217	20,731	18,726
Percentage	30.37%	36.57%	33.04%

From **Error! Reference source not found.**, the number of negative tweets is more as compared to positive and neutral tweets. Also, the positive class has the least number of tweets among the three classes in the dataset. When the length of the tweets in terms of letters against the frequency of their occurrence was plotted in a histogram, the following results have been obtained as shown in Figure 4.2.

[6] https://github.com/imamu-asa/ASA

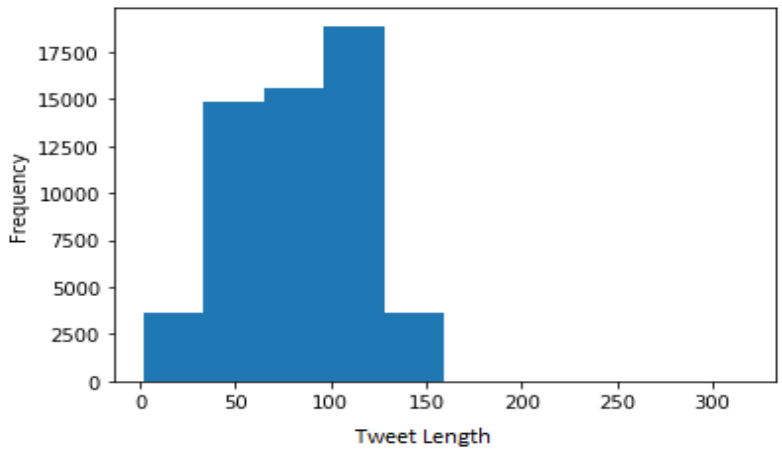

Figure 4. 2 Length of the Tweets Against the Frequency of the Occurrence

It can be observed from the histogram that most of the tweets have a length of around 110-120 characters which is well within the limits of the 140 characters limit of Twitter (this dataset contains tweets that have been made before Twitter increased the character limit to 280 characters). In addition, I observe that around 6% of tweets have a length greater than the 140-character limit which is because HTML encodings are not decoded properly in the dataset. This issue will be handled during the data preprocessing step.

4.4 Configuration

All model's implementation is implemented on a laptop with Intel® Core (TM) i5-8250U CPU@ 1.60 GHz, 8 GB memory, Nvidia GeForce 940 2 GB. The intelligent assessment model is built using Anaconda platform with *Keras 2.3.1* on the deep learning environment *TensorFlow 2.2.0*. The GPU used is from Google Colaboratory (Google Colab)[7]; it is an executable document that allows us to write and execute Python in the browser, with free access to GPUs. Google Colab offers free NVIDIA® Tesla® K80 GPU and connects to virtual machines that have maximum lifetimes of 12 hours, which are suitable for our running time. Table 4.3 presents the model hyperparameters and configurations for the deep learning models.

Moreover, a free Google Colab TPU was used to run the transformers with *TensorFlow 1.14.1*

[7] https://colab.research.google.com/

Table 4. 3 Model Hyperparameters and their Configuration

Hyperparameters	SGRU and SBIGRU Configuration	AraBERT Configuration	Definition
Batch size	64	32	The number of samples that will be passed through to the network at one time.
Dropout	0.2 in each layer	-	Refers to ignoring units during the training phase of certain set of units randomly.
Nodes	100 in each layer	-	Number of units (neurons) in one-layer neural networks.
Training split	0.8	0.9	The training data rate from the dataset.
Validation split	0.1	-	The validation data rate from the dataset.
Testing split	0.1	0.1	The testing data rate from the dataset.
Epoch	5	6	It is one complete presentation of the data set to be passed forward and backward through the neural network only once.
Optimizer	Adam Optimizer	-	Algorithm used to change the attributes of the neural network such as weights and learning rate in order to reduce the losses.
Loss function	Categorical cross-entropy	-	Calculates how poorly the model is performing by comparing what the model is predicting with the actual value it is supposed to output
Learning rate	Change during training using Adam optimizer	0.00002 (constant)	A hyperparameter that controls how much to change the model in response to the estimated error each time the model weights are updated.
Vector size	200 for AraVec, 300 for Fasttext and ArabicNews	-	The size of the vector space in which words will be embedded.

The libraries used in the coding are listed in Table 4.4.

Table 4. 4 Python Libraries' functions and their use

Library	Function	Use
nltk.tokenize	WordPunctTokenizer	Extract the tokens from a string of words or sentences in the form of Alphabetic and non-Alphabetic character.
bs4	BeautifulSoup	Getting data out of HTML, XML, and other markup languages.
Re	re	Check if a particular string matches a given regular expression.
sklearn.model_selection	train_test_split	Split data arrays into two subsets: training data and testing data.
Sklearn.feature_extraction.text	CountVectorizer	tokenize a collection of text documents and build a vocabulary of known words.
keras.utils	to_categorical	Convert an array of labeled data (3 classes) to a one-hot vector.
keras.preprocessing.text.Tokenizer	fit_on_texts texts_to_sequences	**fit_on_texts** creates the vocabulary index based on word frequency. **texts_to_sequences:** Transforms each text in texts to a sequence of integers.
keras.preprocessing.sequence	pad_sequences	Used to put all sequences in a list at the same length
keras.models	Sequential	A straightforward layer, it is limited to single-input, single-output stacks of layers.
keras.layers	Dense Flatten Embedding	**Dense:** deeply connected neural network layer **Flatten:** Converting the data into a 1-dimensional array for inputting it to the next layer. **Embedding:** The first hidden layer of a network, it is initialized with random weights and will learn an embedding for all of the words in the training dataset
keras_metrics	keras_metrics	Provides metrics for the evaluation of Keras classification models.
keras.callbacks	TensorBoard	A tool for providing the measurements and visualizations needed during the machine learning workflow.
Tensorflow	tensorflow	Open-source library for various machine learning tasks.
Shutil	shutil	Copy or move files to other directories.
Pandas	pandas	Used for data manipulation and analysis.
gensim.models	KeyedVectors	Used for mapping between entities and vectors.
bidi.algorithm	get_display	solve the problem of Arabic left-to-right form.
Wordcloud	WordCloud	Create word clouds and tag clouds for the data.
arabic_reshaper	reshape	solve the problem of Arabic isolated character shape.

4.5 Preprocessing

The first stage is the preprocessing phase to clean the tweet and prepare it for classification. The data is cleaned using certain regular expression substitution, the beautiful soup library is used to properly decode the HTML encodings that were mentioned previously. The *WordPunctTokenizer* from the *NLTK* library has been used to tokenize the words while preprocessing.

The cleaning process can be generalized as follow:

1. Decoding HTML encoding (using beautiful soup library).
2. Remove duplicated tweets.
3. Cleaning unrelated contents such as URLs, special characters, emojis, and usernames.
4. Removing any digit, non-Arabic words, and duplicated characters.
5. Normalizing some letters such as (آ ، إ ، أ) to (ا), ة to ه, and ؤ to و.
6. Tokenizing tweets into separate words.

In addition to the cleaning process, an experiment will be done on three additional cleaning process:

1. Removing hashtags.
2. Removing stop words.
3. Removing words based on some conditions.

I will check which one of these three cleaning processes is suitable for dataset cleaning to be used with the previous cleaning process. In the next three subsections, the additional preprocessing techniques will be defined to investigate which technique will enhance the results.

4.5.1 Removing Hashtags

Removing hashtags shows a good result with other preprocessing techniques especially in 2-class classification. I will check if the results are enhanced with 3-classes classification.

4.5.2 Removing Stop Words

To remove stop words, I can rely on lists of stop words available as open source[8], also around 689 stop words have been collected manually, most are gulf dialect words (وشو, مب, ما, بس...). However, those lists are limited and do not contain a lot of the Arabic stop words. To alleviate that, some conditions were used to discard specific words based on word frequency.

Instead of collecting specific words as stop words which is time-consuming and not always effective, and effective way to discard specific words based on some conditions will be presented as follows: First, by discovering the most frequent words in the dataset, as shown in Table 4.5.

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

Table 4. 5 most frequent words in the dataset

word	negative	positive	neutral	total
الهلال	8,293	10,265	4,688	23,246
في	6,550	4,139	8,151	18,840
من	6,624	3,599	3,781	14,004
على	3,017	2,069	2,282	7,368
الاهلي	1,606	3,250	2,106	6,962

Next, applying some functions to extract the list to be used as stop words; by using label generation (negative, positive, neutral, or null) on each word using auto-generated stop words algorithm:

Each column will be saved in separate lists:

1- A list of words in the whole dataset where the total number of words N= 86,024 words, $W = \{w_0, w_1, \ldots, w_{N-1}\}$.

2- A list of positive frequency $POS = \{p_0, p_1, \ldots, p_{N-1}\}$ where *POS* is a positive count, and it represents the number of occurrences of each word in the positive corpus.

3- A list of negative frequency $NEG = \{n_0, n_1, \ldots, n_{N-1}\}$ where *NEG* is a negative count, and it represents the number of occurrences of each word in the negative corpus.

4- A list of neutral frequency $NEU = \{e_0, e_1, \ldots, e_{N-1}\}$ where *NEU* is neutral count, and it represents the number of occurrences of each word in the neutral corpus.

For example, in Table 4.5, when I check the word W_0= "الهلال", this word has appeared 10,265 times in the positive tweets *POS[0]*=10,265, it has appeared 8,293 times in the negative tweets *NEG[0]*=8,293, and it has appeared 4,688 times in the neutral tweets *NEU[0]*=4,688, all in the same row 0. Figure 4.3 illustrates the algorithm that is used to generate stop word list *L* automatically.

```
Input: lists W, POS, NEG, NEU of N values each
Output: list L of all null words
T=5;
V ← {}; L ← {};
for i=0 to N-1 do
    if NEG[i] - POS[i] >= T and NEG[i] - NEU[i] >= T then
        V[i] = negative;
    else if POS[i] - NEG[i] >= T and POS[i] - NEU[i] >= T then
        V[i] = positive;
    else if NEU[i] - NEG[i] >= T and NEU[i] - POS[i] >= T then
        V[i] = neutral;
    else if NEG[i] == 0 and POS[i] == 0 then
        V[i] = neutral;
    else if NEG[i] == 0 and NEU[i] == 0 then
        V[i] = positive;
    else if POS[i] == 0 and NEU[i] == 0 then
        V[i] = negative;
    else
        V[i] = null;
    end if
V ← INSERT(V[i])
end for
for i=0 to N-1 do
    if V[i]== null then
        L ← INSERT(W[i])
    end if
end for
return L;
```

Figure 4. 3 Auto-Generated Stop Words Algorithm

From Figure 4.3, I have 4 predefined lists *W, POS, NEG*, and *NEU* with *N* values for each list. *V* and *L* are empty lists; *V* will store the values (positive, negative, neutral and null), and *L* will store the words that will be used as stop words. The *T* is threshold, I choose 5 because I found it was the most suitable value by trial and error, it can be tuned to find the most suitable value based on the dataset. The first **For** loop will run 86,024 times until it specifies the class of each word, these classifications will be sorted in the *V* list, note that the *V* list will have a total of 86,024 values as well. The second **For** loop will run 86,024 times to check all values of *V* list; for example, if the value of position 0 is null (V_0= null), then the word from *W* list in the same position 0 (W_0= "الهلال") will be stored in a new list *L*. The *L* list will be returned as an output.

Figure 4.4 shows the word cloud of generated stop words. Almost all words are non-sentimental which can be discarded with no issues.

Figure 4. 4 Generated stop words

4.6 Word Representation

The different stacked GRUs are used along with AraVec to learn rich semantic information and rich contextual information effectively. AraVec provides six different word embedding models, where each text domain (Tweets, WWW and Wikipedia) has two different models; one built using continuous BOW technique and the other using the SG technique.

For the AraVec embedding, I use only Twitter domain that included two different models: continuous BOW and SG, which were pre-trained using word2vec, on 204,448 terms collected from 66,900,000 tweets.

The continuous BOW and SG have combined for better results [32]. For each model, I have a 100-dimension vector representation of the word, and by combining these representations, each word will have a 200-dimension vector representation.

In addition, I compare the performance of AraVec with two different embeddings: Fasttext and ArabicNews. The Fasttext model has trained on top of Wikipedia Arabic articles with a dimension of 300 and a vocabulary size of 610,977, while ArabicNews from Altowayan and Tao [34] has trained with a dimension of 300 and a vocabulary size of 159,175. The ArabicNews has built on top of Qur'an, MSA from news articles, the Arabic edition of international networks, and Dialectal Arabic from consumer reviews, but the most dominant source was the Arabic news from various websites.

After applying the embedding, the average tweet length has been discovered, which is represented as K, as a reference to the maximum size of network inputs.

To make sure our model can get equal-sized inputs, the length of tweets must be normalized. If the length of a tweet is less than K, I replenish it by zero-padding until its length reaches K. Afterwards, I replace K words in the tweet with D dimensional vector values. If a word has no corresponding vector in the word embedding model, I replace this word with a zero vector with the D dimension. After I replace all words in the tweet with their embedding values, I can get a series of $K \times D$ tweet vectors. The final step is to combine N tweet vectors into the $N \times K \times D$ vector.

The dataset has been split into training, validation, and test set by randomly picking 80% of the data for the training set, 10% for the validation set, and another 10% for the test set using the *train_test_split* function of the *scikit_learn* module.

4.7 Modeling

Pseudocode:

Step 1: Import the Dataset [Tweets in Arabic with the Polarity]
Step 2: Pre-process the Data
i. Tokenize the Arabic words
ii. Clean the text with the removal of the special letters, symbols and numbers
iii. Remove one of three techniques (Stop words, hashtag, words based on conditions)
iv. Normalize the Arabic Text
v. Remove repeated letters
Step 3: Convert the tokens into Word2Vec (AraVec, Arabic news or Fasttext)
Step 4: Convert the Result of step 3 to Embedding to pass to the Algorithms
Step 5: Split the data into the training, validation and testing
Step 6: Input to the Model
i. Input Layer with 28 units
ii. Continuous Bag of Word Embeddings and Skipped N-Grams Embeddings Combination
iii. Layer 1 GRU with 100 units (increase layer with the same unit)
iv. Dense Layer with 3 units
Step 7: Output in 3 classes

Figure 4. 5 Pseudo-Code for SGRU

Pseudocode:

Step 1: Import the Dataset [Tweets in Arabic with the Polarity]
Step 2: Pre-process the Data
i. Tokenize the Arabic words
ii. Clean the text with the removal of the special letters, symbols and numbers
iii. Remove one of three techniques (Stop words, hashtag, words based on conditions)
iv. Normalize the Arabic Text
v. Remove repeated letters
Step 3: Convert the tokens into Word2Vec (AraVec, Arabic news or Fasttext)
Step 4: Convert the Result of step 3 to Embedding to pass to the Algorithms
Step 5: Split the data into the training, validation, and testing
Step 6: Input to the Model
i. Input Layer with 28 units
ii. Continuous Bag of Word Embeddings and Skipped N-Grams Embeddings Combination
iii. Forward Layer GRU with 100 units (increase layer with the same unit)
iv. Backward Layer GRU with 100 units (increase layer with the same unit)
v. Bidirectional Layer by concatenating Forward and Backward GRU
vi. Flatten Layer
vii. Dense Layer with 64 units
viii. Dense Layer with 3 units
Step 7: Output in 3 classes

Figure 4. 6 Pseudo-Code of SBi-GRU

Given a batch of N tweets and each tweet with K words, let $X = \{X_1, X_2, \ldots, X_N\}$ be a set of tweets in a batch and $X_n = \{w_1, w_2, \ldots, w_K\}$ be a set of words in any tweet X_n. It is worth mentioning that any word X_K is a D-dimensional embedding word vector. The goal of this model is to predict sentiment y' for each tweet. In our case, the input is a 3-dimensional matrix with a size of $N \times K \times D$. Each model represents the past and future context and combines both parts' features as outputs of the model.

For any tweet, hits sentiment y_n can only be negative, positive, or neutral. In addition, each feature of tweets either negative, positive, or neutral, appear as discrete values. Therefore, one hot is needed to encode these discrete features. Three features of a tweet will be mapped into 3 bits one hot code. Specifically, [1,0,0] represents negative and [0,0,1] represents positive, and [0,1,0] represents neutral. After encoding, a sentiment label is obtained y_n ($y_n \in$ {[1,0,0], [0,1,0], [0,0,1]}) for any tweet X_n.

The next step is to fit a tokenizer on the tweets and then convert the tweets to sequences using the predefined functions in the *Keras* library. I also pad the tweets so that all of them are of equal length as a *Keras* neural network needs all the inputs to be uniform.

Afterward, I use the pre-trained word embedding model for getting the word embedding representation of the Arabic tweets. The continuous BOW and SG models have combined as presented in AraVec 3.0 [32] to create an embedding index which is then used to map

the words to their word vectors.to generate the embedding matrix.

4.7.1 SGRU Architecture

From Section 3.7, I concluded that an increasing number of layers leads to higher accuracy. However, all studies stop increasing layers when they reach layer 4. Moreover, the number of hidden layers should be chosen carefully to avoid overfitting, a fixed hidden unit has been used during increasing layers (100 unit) for a fair comparison. A dense GRU model will be implemented for classification which is a combination of Figure 3.4 and Figure 3.6. Specifically, the input of the first layer in SGRU is the original data, and the formulas are the same as the GRU unit in Subection 3.3.5. The input of each GRU unit in the upper layers is the output of the hidden layer of the lower layer GRU unit.

For time sequence *T*, the input sequence $\{X_1, X_2, \ldots, X_T\}$ enters into first hidden layers $\{h_1^1, h_2^1, \ldots, h_T^1\}$ to obtain complete information from all past time steps. After that, the upper hidden layers take the outputs from lower hidden layers at each time step as their inputs to extract further features. Specifically, the upper layers of hidden layers are $\{h_1^2, h_2^2, \ldots, h_T^2\}$. Figure 4.7 indicates SGRU architecture.

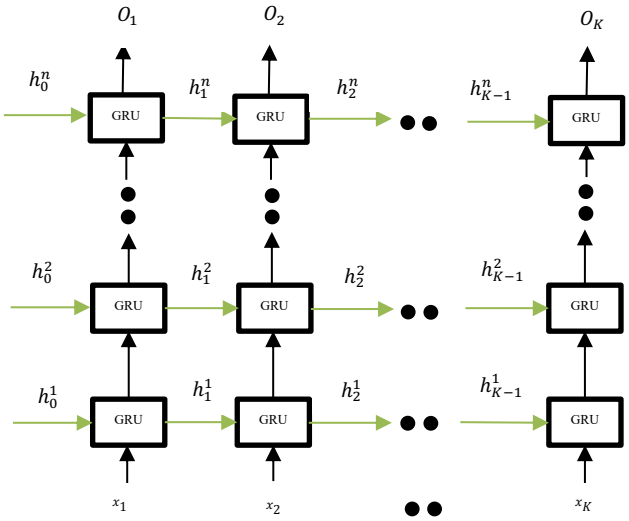

Figure 4. 7 SGRU Architecture

For each layer, a hidden state $h^i t$, as shown in Equation (4.4) is given by the following Equations (4.1), (4.2), and (4.3) for obtaining update gate, reset gate, and candidates value respectively Note that in Equation (4.1), (4.2), and (4.3). I insert the embedding vector $e^{i-1}t$ to the first layer. Starting from the second layer and above, I use the hidden state from recent time step in the past layer $h^{i-1}t$ instead of $e^i t$ in (4.1), (4.2), and (4.3).

$$u^i{}_t = \sigma\left(W^i{}_u\, h^i{}_{t\text{-}1} + U^i{}_u\, e^i{}_t + b^i{}_u\right) \tag{4.1}$$

$$r^i{}_t = \sigma\left(W^i{}_r\, h^i{}_{t\text{-}1} + U^i{}_r\, e^i{}_t + b^i{}_r\right) \tag{4.2}$$

$$\tilde{C}^i{}_t = \tanh\left(W^i{}_c \cdot \left[r^i{}_t * h^i{}_{t\text{-}1}\right] + U^i{}_c\, e^i{}_t + b^i{}_c\right) \tag{4.3}$$

$$h^i{}_t = u^i{}_t * \tilde{C}^i{}_t + (1 - u^i{}_t) * h^i{}_{t-1} \quad (4.4)$$

The output for the last layer as shown in Equation (4.5) [147]:

$$O_t = W^o h^n{}_t + b^o \quad (4.5)$$

4.7.2 Stacked Bi-GRU

I implement GRU instead of using LSTM in Zhou et al. paper [147]. This architecture was implemented in 2019 and from our knowledge, no GRU model has been implemented for this kind of architecture. For time sequence T, the input sequence $\{e_1, e_2, \ldots, e_T\}$ enters into hidden layers in the forward direction $\{h_1^a, h_2^a, \ldots, h_T^a\}$ to obtain complete information from all past time steps and hidden layers in the reverse direction $\{h_1^c, h_2^c, \ldots, h_T^c\}$ to get complete information from all future time steps. After that, the upper hidden layers take the outputs from lower hidden layers at each time step as their inputs to extract further features. Specifically, the upper layers of forwarding hidden layers are $\{h_1^b, h_2^b, \ldots, h_T^b\}$ and the upper layers of backward hidden layers are $\{h_1^d, h_2^d, \ldots, h_T^d\}$. At last, output layers integrate two upper layers' hidden vector together as their output. Figure 4.8 shows SBi-GRU architecture.

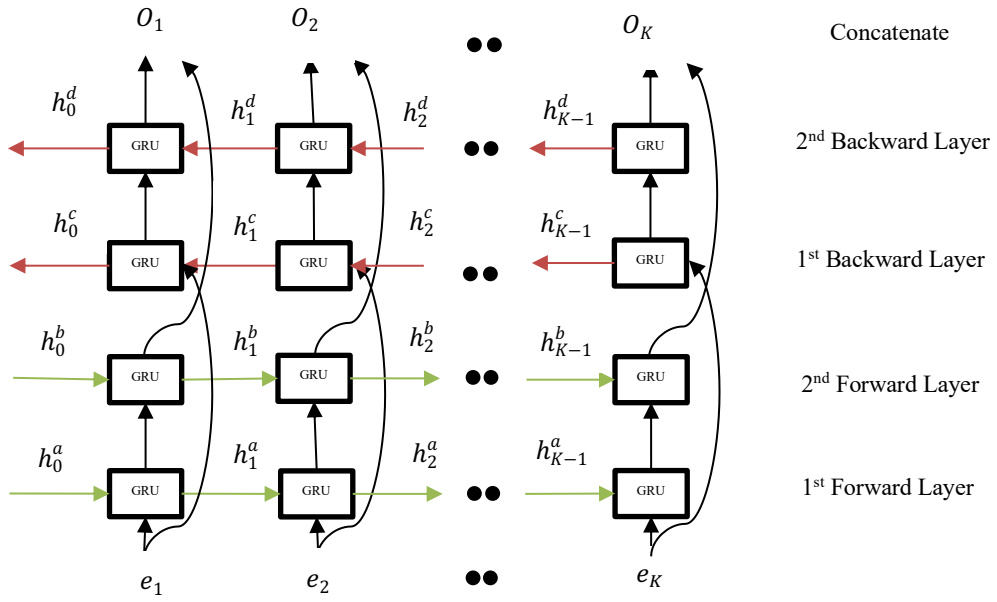

Figure 4. 8 SBi-GRU Architecture

For the first forward layer, hidden state $h^a t$ as shown in Equation (4.9) is given by the following Equations (4.6), (4.7), and (4.8) for obtaining update gate, reset gate, and candidates value respectively:

$$u^a t = \sigma(W^a u\, h^a t\text{-}1 + U^a u\, e^a t + b^a u) \quad (4.6)$$

$$r^a t = \sigma (W^a r \, h^a t\text{-}1 + U^a r \, e^a t + b^a r) \quad (4.7)$$

$$\tilde{C}^a t = \tanh(W^a c . [r^a t * h^a t\text{-}1] + U^a c \, e^a t + b^a c) \quad (4.8)$$

$$h^a t = u^a t * \tilde{C}^a t + (1 - u^a t) * h^a t\text{-}1 \quad (4.9)$$

The second forward layer:

For the second forward layer, the hidden state $h^b{}_t$, as shown in Equation (4.13) is given by the following Equations (4.10), (4.11), and (4.12) for obtaining update gate, reset gate, and candidates value respectively, noted that in Equations (4.10), (4.11), and (4.12) the hidden state from the first forward layer in the same time step has been used:

$$u^b{}_t = \sigma (W^b{}_u \, h^b{}_{t\text{-}1} + U^b{}_u \, h^a{}_t + b^b{}_u) \quad (4.10)$$

$$r^b{}_t = \sigma (W^b{}_r \, h^b{}_{t\text{-}1} + U^b{}_r \, h^a{}_t + b^b{}_r) \quad (4.11)$$

$$\tilde{C}^b{}_t = \tanh(W^b{}_c . [r^b{}_t * h^b{}_{t\text{-}1}] + U^b{}_c \, h^a{}_t + b^b{}_c) \quad (4.12)$$

$$h^b{}_t = u^b{}_t * \tilde{C}^b{}_t + (1 - u^b{}_t) * h^b{}_{t\text{-}1} \quad (4.13)$$

For the first backward layer, hidden state $h^c{}_t$ as shown in Equation (4.17) is given by the following Equations (4.14), (4.15), and (4.16) for obtaining update gate, reset gate, and candidates value respectively:

$$u^c{}_t = \sigma (W^c{}_u \, h^c{}_{t+1} + U^c{}_u \, e^c{}_t + b^c{}_u) \quad (4.14)$$

$$r^c{}_t = \sigma (W^c{}_r \, h^c{}_{t+1} + U^c{}_r \, e^c{}_t + b^c{}_r) \quad (4.15)$$

$$\tilde{C}^c{}_t = \tanh(W^c{}_c . [r^c{}_t * h^c{}_{t+1}] + U^c{}_c \, e^c{}_t + b^c{}_c) \quad (4.16)$$

$$h^c{}_t = u^c{}_t * \tilde{C}^c{}_t + (1 - u^c{}_t) * h^c{}_{t\text{-}1} \quad (4.17)$$

For the second backward layer, the hidden state $h^d{}_t$, as shown in Equation (4.21) is given by the following Equations (4.18), (4.19), and (4.20) for obtaining update gate, reset gate, and candidates value respectively, noted that in Equation (4.18), (4.19), and (4.20) the hidden state from the first backward layer in the same time step has been used:

$$u^d{}_t = \sigma (W^d{}_u \, h^d{}_{t+1} + U^d{}_u \, h^c{}_t + b^d{}_u) \quad (4.18)$$

$$r^d{}_t = \sigma (W^d{}_r \, h^d{}_{t+1} + U^d{}_r \, h^c{}_t + b^d{}_r) \quad (4.19)$$

$$\tilde{C}^d{}_t = \tanh(W^d{}_c . [r^d{}_t * h^d{}_{t+1}] + U^d{}_c \, h^c{}_t + b^d{}_c) \quad (4.20)$$

$$h^d{}_t = u^d{}_t * \tilde{C}^d{}_t + (1 - u^d{}_t) * h^d{}_{t\text{-}1} \quad (4.21)$$

The output of the combining second forward and backward layer as shown in Equation (4.22):

$$O_t = U^o h^b{}_t + W^o h^d{}_t + b^o \tag{4.22}$$

4.8 Sentiment Prediction

The $softmax$ classifier takes the output at the last step K. The O_K will be used as input for the prediction. Given N tweets with K words, I predict the sentiment y for each tweet. Real annotations of tweets are represented by Y ($Y = Y_1, Y_2, \ldots Y_N$) The predicted values y' can be calculated as shown in Equation (4.23) and Equation (4.24):

$$p(y|X) = softmax\ (W^s O_K + b^s) \tag{4.23}$$

$$y' = arg_y max\ p(y|X) \tag{4.24}$$

Where *p* denotes the probability of real-valued nodes for each tweet *X* to generate a vector of probabilities. The *argmax* function is applied to the vector of probabilities. To specify the tweet class, the Equation (4.24) gives the maximum value from these probabilities in Equation (4.23).

Then the cross-entropy is used to train the loss function. First the loss of each labeled tweet is derived, and the final loss is averaged over all the labeled tweets N by the following Equation (4.25):

$$Loss = -\frac{1}{N}\sum_{n=1}^{N} Y_n\ .logp(yn|Xn) \tag{4.25}$$

where the subscript n indicates the n[th] input tweet. Given the true distribution *Yn* and the estimated distribution *yn/Xn,*

Then, Adam optimizer is used to adaptively adjust the learning rate and optimize the parameters of the model. At each hidden layer, a dropout of 20% is defined to avoid overfitting.

4.9 AraBERT

Language-specific BERT models have recently proved to be effective in language comprehension with the proliferation of transformer-based models because they are pre-trained in an incredibly broad corpus. Such frameworks set high benchmarks for certain NLP activities and produced state-of-the-art performance. I have expressly qualified BERT for Arabic to accomplish the same accomplishment BERT has accomplished in English. AraBERT's performance is compared with Google's multilingual BERT and other cutting-edge approaches. For AraBERT, we use the parameter configuration in Table 4.3. In addition, we set the maximum sequence length to 512.

Figure 4.9 shows the steps for implementing the Sentiment Analysis using AraBERT to compare its performance with proposed SGRU and SBi-GRU models [62].

Pseudocode:

Step 1: Initializing the Transformers
Step 2: Model Tokenizer using Auto tokenizer (*"aubmindlab / bert base arabertv01"*)
Step 3: Loading Auto Model from Transformer (initializing the AraBERT model)
Step 4: Configuring the AraBERT learned weight (Inputting the weights folder)
Step 5: Training the AraBERT model (varying Epochs and Batch Size)
Step 6: Predicting the *Validation Sample*
Step 7: Model Deployment

Figure 4. 9 Pseudo-Code for AraBERT model

4.10 Ensemble Model (AraBERT+SGRU+SBIGRU)

Ensemble modeling is a technique of weighing individual opinions and combining them to arrive at a final decision [181]. These techniques have been successful in improving the accuracy of machine learning models by training several individual classifiers and combining them to improve the overall predictive power of the model. They utilize several classifiers by combining them in some manner to obtain the result either by performing weighted average or majority voting of the individual classifiers to improve the overall accuracy of the model when compared to the accuracy obtained by using a single classifier. Here, I use three models (AraBERT, SGRU, and SBi-GRU) to obtain the voting of the final classification. The following are the steps used for the ensemble model for the trained AraBERT, trained SGRU model, and the trained SBi-GRU model with the Arabic Tweets as shown in Figure 4.1. Figure 4.10 shows the ensemble model architecture.

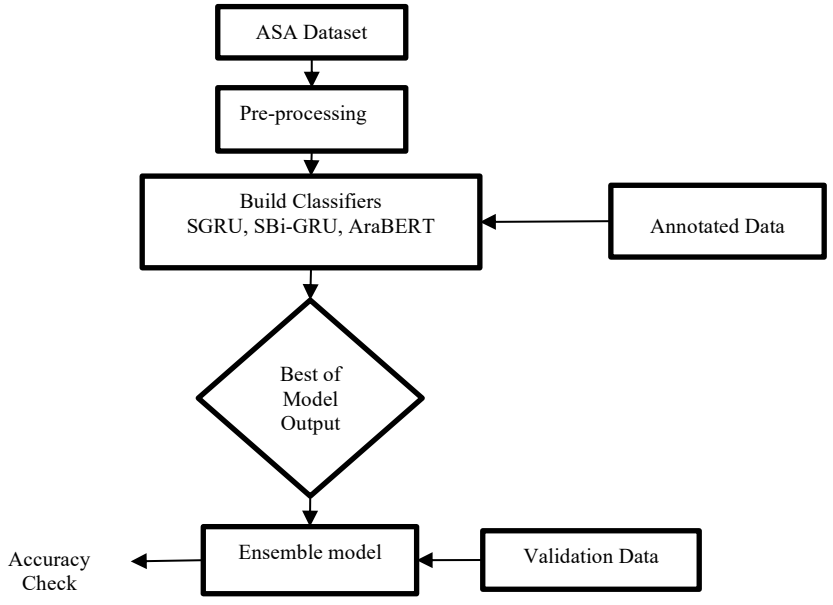

Figure 4. 10 Ensemble Model Architecture

Figure 4.11 presents the pseudo-Code for the ensemble model [62].

Pseudocode:

Step 1: Data Acquisition

Step 2: Training the Model for the AraBERT, SGRU and SBIGRU model with the Arabic Tweets

Step 3: Checking the best of the results for the validation tweet

Step 4: Model

Figure 4. 11 Pseudo-Code for Ensemble Model

4.11 Conclusion

This chapter presented the models to be implemented and the models to be applied as baselines for comparison. The architecture and workflow in addition to the configuration for SGRU and SBi-GRU were discussed. In the next chapter, the results for training these models will be presented along with the discussion based on these results.

CHAPTER FIVE: RESULTS AND DISCUSSION

5.1 Introduction

This chapter presents the results of experiments and answers RQ5, RQ6, RQ7, and RQ8. Also, it presents a comparison between the performance under different techniques and settings such as: embeddings, pre-processing techniques, hashtag and stop words removal, stacking GRUs and Bi-GRUs, and the ensemble model performance. In specific, I present the evaluation metrics, the performance of the proposed and applied models, and finally a discussion on the results for each model. In Section 5.2, the evaluation metrics that will be used to evaluate each model will be presented. In Section 5.3 I compare different word embeddings (Subsection 5.3.1), different preprocessing techniques (Subsection 5.3.2), different GRU structures, and present the results based on the evaluation metrics to Answer RQ5 RQ6 (Subsection 5.3.3). Also, the outcome from these experiments will be discussed and compared with the ensemble method and machine learning SVM to answer RQ7 and RQ8 (Subsection 5.3.4). In Section 5.4 I present the conclusion for Chapter 5.

5.2 Evaluation Metrics

In this section, I present the evaluation metrics that will be used in the experiment. It is important to evaluate and quantify the performances of different methods in order to track the changes and select the best classifier of the given data set. To evaluate the performance of the different algorithms and the proposed approach, four standard evaluation metrics are calculated [182]. They are listed in the following: Accuracy, precision, recall, and F1 score.

1. Accuracy:

To evaluate the accuracy of the model, and it is computed as the percentage of correctly classified tweets to the total tweets. Therefore, it can be represented mathematically as seen in Equation (5.1):

$$Accuracy = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{FP} + \mathrm{TN} + \mathrm{FN}} \tag{5.1}$$

2. Precision:

Precision evaluates the strictness of the classifier output. Precision is the percentage of tweets classified as positive correctly to the total number of samples classified as positive. The precision can be calculated as shown in Equation (5.2).

$$Precision = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \tag{5.2}$$

3. Recall:

A recall is used to measure the integrity of the classifier's output. Recall measures the percentage of actual tweets that were correctly classified. The recall can be calculated as seen in Equation (5.3).

$$Recall = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \tag{5.3}$$

4. F1 Score:

F1 indicates Formula one score, it is a formula to gather the scores of precision and recall, and it is defined as the harmonic mean of the model's precision and recall as shown in Equation (5.4).

$$F1 = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \tag{5.4}$$

Where:

True Positive (TP): the number of tweets that were correctly classified as positive.

True Negative (TN): the number of tweets that were correctly classified as negative.

False Positive (FP): the number of tweets that were incorrectly classified as positive.

False Negative (FN): the number of tweets that were incorrectly classified as negative.

Moreover, I focus on the training time. Training time refers to how much computation time is required for the model to learn and is measured in minutes. There are several points that could impact learning time such as the model structure, the number of the classes in the dataset, and weather the data is preprocessed or not; The complex model structure may affect the learning time to be longer. Also, the training time for 3-class dataset is relatively longer that 2-class dataset with the same size. Moreover, unsuitable preprocessing techniques may affect the learning time and the accuracy, respectively.

I also focus on the loss that is mentioned in Section 4.8. Loss calculates how poorly the model is performing by comparing what the model is predicting with the actual value it is supposed to output. If the loss is high, its effect will propagate through the network while it's training, and the node weights will be changed frequently. If it's small, then the weights won't change as frequent since the network already performed well. Hence the objective is to minimize the loss function, the lower the loss the better the model.

5.3 Results and Discussion

In this section, I present the experiments' results and compare the models' efficiency using the evaluation metrics mentioned previously. To gauge the effectiveness of the proposed systems and its features, I apply four phases of comparison to obtain as best results as possible: compare different embeddings, pre-processing techniques, GRU and Bi-GRU architectures while increasing the number of layers, and finally, compare these models with the ensemble model and machine learning SVM. I gradually choose the best method in each phase to be implemented in the remaining phases. Moreover, this section also answers the remaining RQs under consideration (RQ5, RQ6, RQ7, RQ8).

Subsection 5.3.1 compares the baseline GRU model with different embeddings (AraVec, Fasttext, and ArabicNews), the best embedding will be selected and implement in the next experiments. In Subection 5.3.2, a comparison between different preprocessing techniques is applied to specify the most suitable techniques for processing dataset. In Subsection 5.3.3, the best preprocessing technique will be used when implementing SGRU and SBi-GRU

models. I compare the stacking of GRU and SGRU and their performance during layers' increment and I compare them with the SVM model to answer RQ5 and RQ6. In Subsection 5.3.4, the performance comparison of the ensemble method compared with SVM, the best SGRU model, and the best SBi-GRU model, and AraBERT will be presented, in addition to answer RQ7 and RQ8.

5.3.1 Comparison Between Different Embeddings

This subsection introduces a comparison between different embeddings: AraVec [32], ArabicNews [34], and Fasttext [39]. I apply one-layer GRU with 100 units, and I set the maximum sentence length to 29 words. The following testing results were obtained using different embeddings as shown in Table 5.1.

Table 5. 1 GRU Results Using Different Embeddings

Metrics / Embedding	Accuracy	Precision	Recall	F1 Score	Loss	Training Time
AraVec	79.08%	83.60%	69.95%	76.12%	56.26%	4min 24s
Fasttext	76.35%	75.47%	76.58%	75.97%	96.96%	5min 42s
ArabicNews	74.93%	76.81%	70.23%	73.35%	87.11%	4min 8s

Figure 5.1 presents the accuracy of different embeddings in 5 epochs. AraVec performs better than all other embeddings. Even though the vocabulary size of Fasttext is bigger than AraVec size, I notice that AraVec outperformed Fasttext in term of accuracy and training time and that was due to the nature of data with which AraVec is trained. As I mentioned in Section 4.5, I use only Twitter domain from AraVec, while for Fasttext, the model has built on top of Wikipedia. That explains the dramatic decrement of Fasttext embedding during training phase, in the first epoch, the accuracy of the Fasttext embedding outperform the AraVec because of the larger vocabulary size of Fasttext. After the second epoch, the loss has increased, and the representation wasn't as effective as AraVec.

For the ArabicNews, the loss in the first epoch caused because the vocabulary size is relatively small compared to other embeddings, and the most dominant source was the Arabic news from several resources. The accuracy increasement in the next epochs is because our dataset contains news-related tweets. However, the overall performance is lower than other embeddings.

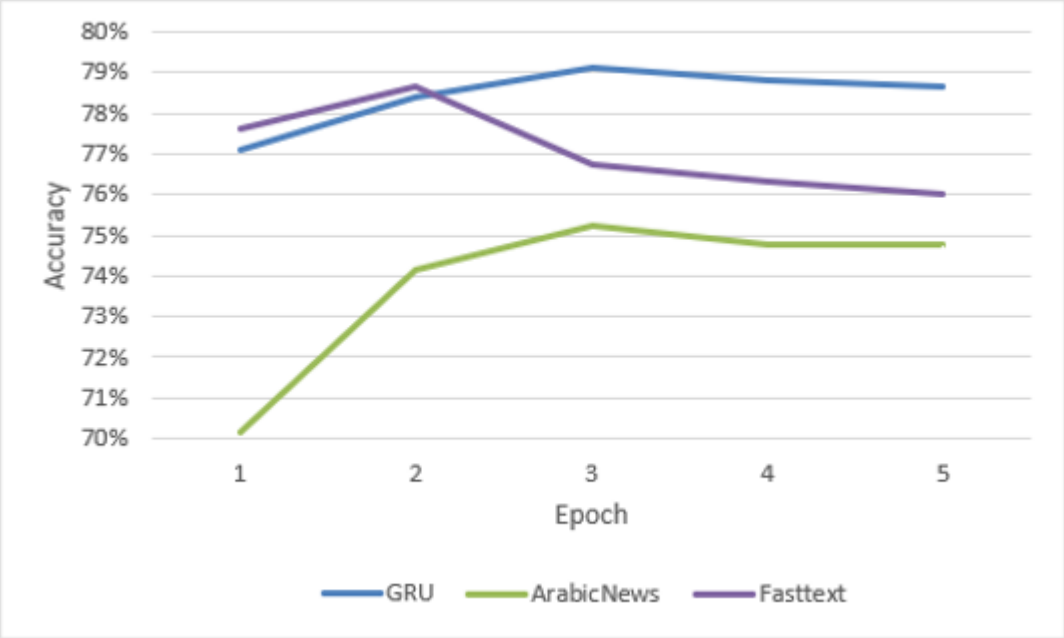

Figure 5. 1 Comparison between Different Embeddings in Terms of Accuracy

In the next experiments, I use AraVec as it gives the best results compared to other embeddings as shown in Table 5.1.

5.3.2 Comparing Different Preprocessing Techniques

In this subsection, I compare several preprocessing techniques to discover the most effective technique for a 3-class dataset, positive, negative, and neutral. As I mentioned in the previous subsection, I use GRU with AraVec which was the best embedding for feature representation. The preprocessing techniques that I compare are; Hashtag removal, stop words removal, and conditional words removal those are explained in Section 4.5. For the hashtag's removal, I remove all hashtags and symbols associated with them. For the stop word removal, I have collected 689 stop words to clean the dataset. For the auto-generated stop words, I apply the algorithm that is mentioned in Figure 4.3. Those different techniques will be applied separately along with the basic pre-processing steps by removing the following: URLs, special characters, emojis, usernames, digit, non-Arabic words, duplicated characters, normalizing, and tokenizing. The results are obtained as shown in Table 5.2.

Table 5. 2 GRU Results Using Different Pre-processing Techniques

Metrics / Preprocessing Technique	Accuracy	Precision	Recall	F1 Score	loss	Training Time
Basic preprocessing	79.08%	83.60%	69.95%	76.12%	56.26%	4min 24s
Basic preprocessing + Hashtag removal	77.07%	81.41%	71.58%	76.15%	59.83%	6min 19s
Basic preprocessing + Stop words removal	78.53%	84.11%	75.32%	79.46%	57.43%	5min 42s
Basic preprocessing + auto-generated stop words removal	81.39%	87.44%	76.51%	81.60%	48.74%	4min 8s

As shown in Table 5.2, not all preprocessing techniques yielded good results. The accuracy of hashtag removal has decreased by 2% compared with the accuracy of using basic preprocessing only, since hashtags have useful information that I could not ignore. Some hashtags are related to the news which is considered as neutral classification. Also, some hashtags have negative impacts, which affects the data negatively. In addition, people use hashtags to complete their sentences.

The accuracy of the stop words removal was not enhanced compared with the accuracy of using basic preprocessing only. In the case of stop words, there are no unified stop words to be used in the pre-processing phase, because of the diversity of dialects, data sets' domains, and the words. Some researchers may think that some words are not important and can be discarded from the dataset, while other researchers may think that these words should be kept, the negation words are an example. Some researchers considered negation words as stop words while other researchers considered these words as important words and thus should be retained. Moreover, considering domain-specific stop words in addition to generic words such as club names for sports domain and city names for news domain to be discarded is time-consuming and not always effective especially for the gulf tweets that have many dialects based on the regions. Hence, using stop words in the pre-processing phase does not always lead to better results.

In the case of conditional words, the accuracy has increased along with decrement in the loss as shown in Figure 5.2. The results show the effectiveness of ignoring non-sentimental words with an enhancement of 8% in the loss decrement. The benefit of this method is that the stop words will be auto generated, which means it can be used on any datasets with no need to collect stop words manually from scratch. The method discards the words that appear almost equally in the three classes. Those words could be considered as noisy words and hence, removing them strengthen the sentiment analysis by focusing on sentimental words only to boost the performance.

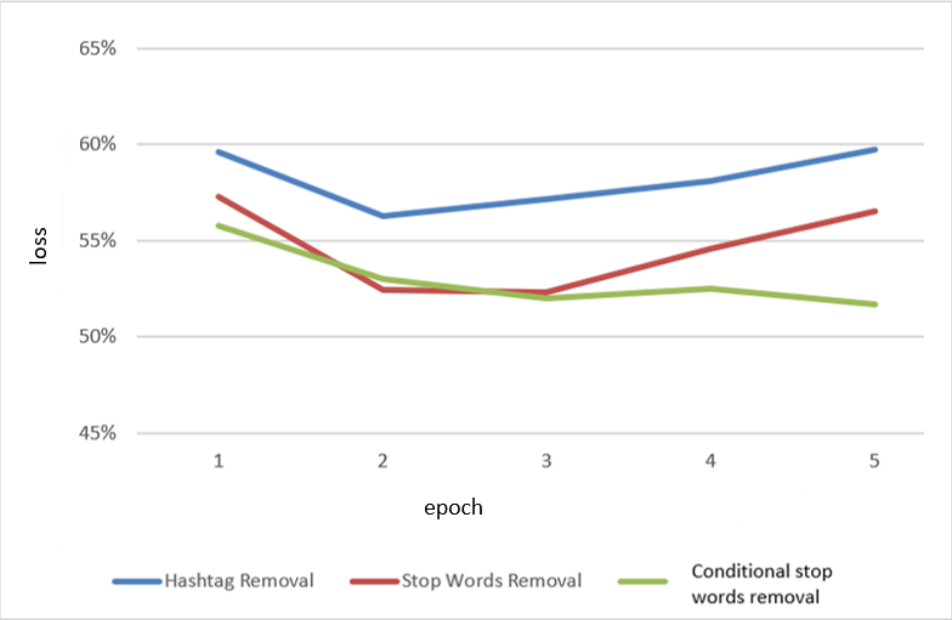

Figure 5. 2 The Loss of Different Pre-processing Techniques

5.3.3 Comparison Between Stacking Layers of GRU and Bi-GRU

In this subsection, I attempt to stack GRUs and Bi-GRUs and measure the performance during the layers' increasement. I make this comparison to find out the best number of layers that the model work with based on the best accuracy yielded. For the first comparison, I increase the number of GRU layers and check the improvement during layers' incensement., the AraVec embedding, basic pre-processing, and conditional stop words removal have been selected in the implementation since they perform the best accuracy as mentioned in the previous subsections. Table 5.3 shows the results of increasing GRU layers. Here

Table 5. 3 Results with Different GRU Architectures

Metrics / Model	Accuracy	Precision	Recall	F1 Score	loss	Training Time
GRU	81.39%	87.44%	76.51%	81.60%	48.74%	4min 8s
SGRU-2 layers	81.54%	89.17%	74.50%	81.15%	47.89%	7min 28s
SGRU-3 layers	81.54%	89.12%	74.10%	80.90%	48.28%	10min 51s
SGRU-4 layers	81.79%	87.69%	78.16%	**82.64%**	**47.37%**	14min 7s
SGRU-5 layers	81.96%	**89.46%**	76.10%	82.22%	48.43%	17min 35s
SGRU-6 layers	**82.08%**	86.32%	**78.70%**	82.33%	47.73%	21min 8s
SGRU-7 layers	81.34%	88.24%	74.95%	81.04%	51.63%	24min 32s

In the comparison of finding the best of the GRU architecture, I found that the SGRU with 6 layers performed the highest accuracy with 82.08%.

For the second comparison, multiple Bi-GRUs are stacked and the performance during layers' increment was tracked. Table 5.4 shows the results of increasing Bi-GRU layers, altering its architecture.

Table 5. 4 Results with Different Bi-GRU Architectures

Metrics / Model	Accuracy	Precision	Recall	F1 Score	loss	Training Time
BIGRU	80.78%	**87.80%**	**81.53%**	**84.55%**	57.04%	12min 6s
SBIGRU-2 layers	81.23%	87.43%	81.26%	84.23%	52.38%	17min 28s
SBIGRU-3 layers	81.27%	87.46%	80.92%	84.06%	56.68%	22min 58s
SBIGRU-4 layers	80.13%	87.24%	80.98%	83.99%	56.83%	28min 29s
SBIGRU-5 layers	**81.59%**	86.97%	81.26%	84.02%	52.71%	33min 58s
SBIGRU-6 layers	81.05%	86.96%	81.04%	83.89%	52.61%	39min 24s
SBIGRU-7 layers	80.89%	86.75%	81.18%	83.87%	**51.82%**	45min 4s

When the Bi-GRU model was used and compared with the Chinese model [147], the results were on par with those of a Bi-LSTM model. The Bi-LSTM had a better F1 score as compared to its Bi-GRU counterpart (84.23% for SBi-GRU 2 layer, 84.49% for SBi-LSTM 2 layers). In general, SBi-LSTM and SBi-GRU are relatively the same.

On comparing an SGRU model with a similar SBi-GRU model, I can see that the accuracy has increasing with the layers' increasing. I found that SBi-GRU outperformed the SGRU model in terms of the recall and F1 score while giving similar performance when I take other metrics into account. In the case of the SBi-GRU, for the accuracy, the best model that has performed to achieve the highest accuracy is the architecture with the 5 layers. The accuracy achieved is 81.59% which is equivalent to that of the SGRU that achieved 82.08%. If other metrics are considered like the Precision, the best result is by SBi-GRU with 87.80% in comparison to 5-layers SGRU having 89.46%. For the Recall, SBi-GRU achieved 81.53% in comparison to 6-layer SGRU with an accuracy of 78.7%. The best FI Score is achieved by SBi-GRU with 84.55% in comparison to SGRU with 6 layers that achieved 82.33%. On the other hand, SGRU outperforms SBi-GRU in terms of loss. Figure 5.3 presents the performance between SGRU and SBi-GRU in terms of the F1 Score.

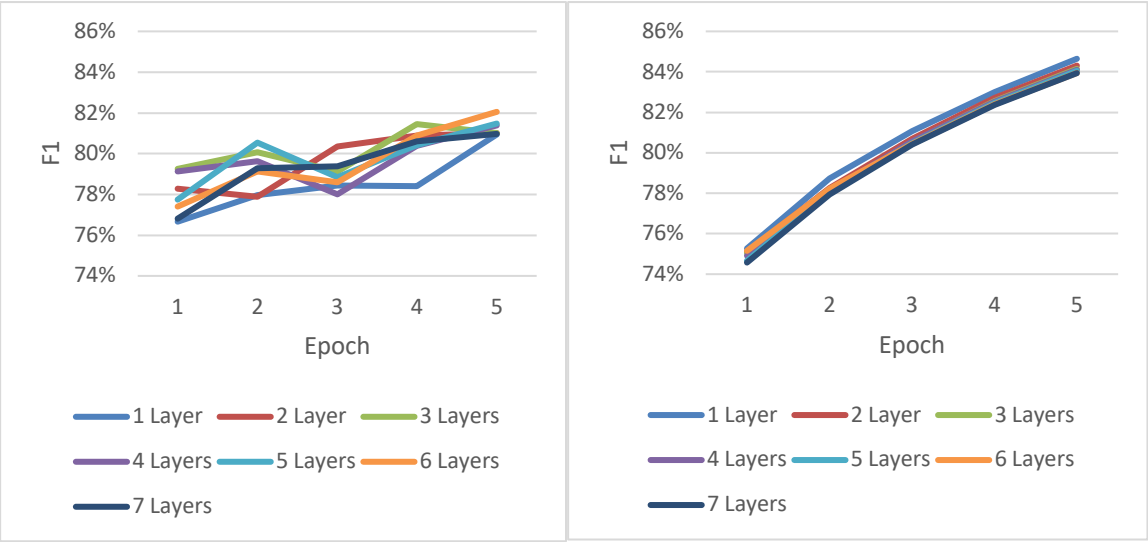

Figure 5. 3 F1-Score for SGRU (left) and SBi-GRU (right)

From Figure 5.3, the performance of SBi-GRU increased smoothly, the F1-score reaches 85% as compared to SGRU counterpart (82%) in the same epoch, with a possibility for

better results when the number of the epoch is increased. This result is due to the dense architecture of the SBi-GRU model. The benefit of utilizing the bi-directional keeps the performance increased more stable during iterations.

Based on the results, RQ5 and RQ6 have been answered as follows:

RQ5: How does adding layers to the GRU affect the accuracy?

The addition of layers gets additional extracted features from the model. But in the previous scenario, I did observe this pattern for a certain increase in the layers and then the performance either became stable or decease.

In the comparison between SBi- GRU, and SGRU, the pattern is evident. For the SGRU, the accuracy improved until layer 6, and then accuracy decrease. If some other hyperparameters would have been tuned, it could have resulted better. But to get the hyperparameters would have made the system highly unstable. Hence, for the neural network, some of the parameters are taken to tune and the rest are kept the same to check the model performance and then compare. The accuracy that has got is the optimum for that set of the hyper parameters for the SGRU and SBi-GRU. The terms of computation time, the time is taken increases with the increase in the additional layers.

RQ6: How does deep learning models perform compared to machine learning models?

The stacking layer performed well with GRU and Bi-GRU more than the SVM model. Table 5.4 shows that the performance of GRUs in this sentiment analysis task is comparable with machine learning model SVM. This is because it is a sequential task with less dependency on long-term sequences since the length of tweets is restricted to 140 characters only. Also, the dataset which was used for our study despite being one of the largest corpora of Arabic tweets with sentiment labeling with around 56k tweets, it is still not sufficient for GRU to result in state-of-the-art performance with stacking layers only, it could have been resulted in better in some other. Since the setting was focused on only the hidden neurons and layers. The accuracy for the GRU models varies around 79-82% for 3-class classification, this can be improved further with some more complex architectures and with better hyper-parameter tuning. The experiments were done in the study to show that the dataset affects the results and thus a better dataset is needed to get better results. The models can be improved by obtaining clean data for the dataset and further refining the dataset that I already have at hand. Since the character limit for twitter has been increased to 280 characters since 2017 while ASA dataset was collected between 2014 and 2016, getting recent tweets might help as models can learn more about the language as some more contextual information can be contained within the text. I could experiment with some better GRU architectures to see how it impacts our task of sentiment classification. In previous studies, only four GRU layers were stacked but increasing the number of layers in our study gives better results. The results obtained assumes that whatever pre-processing has been done is sufficient and the dataset does not require any further modification, but I can further improve the pre-processing with certain language modeling techniques that are specific to the source language. Taking some other machine learning algorithms to compare through the results, SVM gave an accuracy of 77%. To take the best of the results since there are 4 best models selected, I used a voting ensemble to take the best results from the model fitted in the training datasets.

5.3.4 Ensemble Model Compared with Other Models

In this subsection, I present the result of applying the ensemble model. As I mentioned in the previous subsection, increasing layers led to better accuracy, but to get the highest benefit of stacking layers, I apply the ensemble method by combining three pretrained models (6-layer SGRU, 5-Layer SBi-GRU, and AraBERT). Those models were pretrained using ASA dataset, and the result of this model will be compared with other approaches. To figure out which model performed better; I must fix the performance measure. Therefore, accuracy will be considered to give more priority to the correct predictions. Figure 5.4 shows the best models of SGRU and Bi-GRU along with benchmark models such as SVM, AraBERT, in addition to the accuracy of the ensemble method.

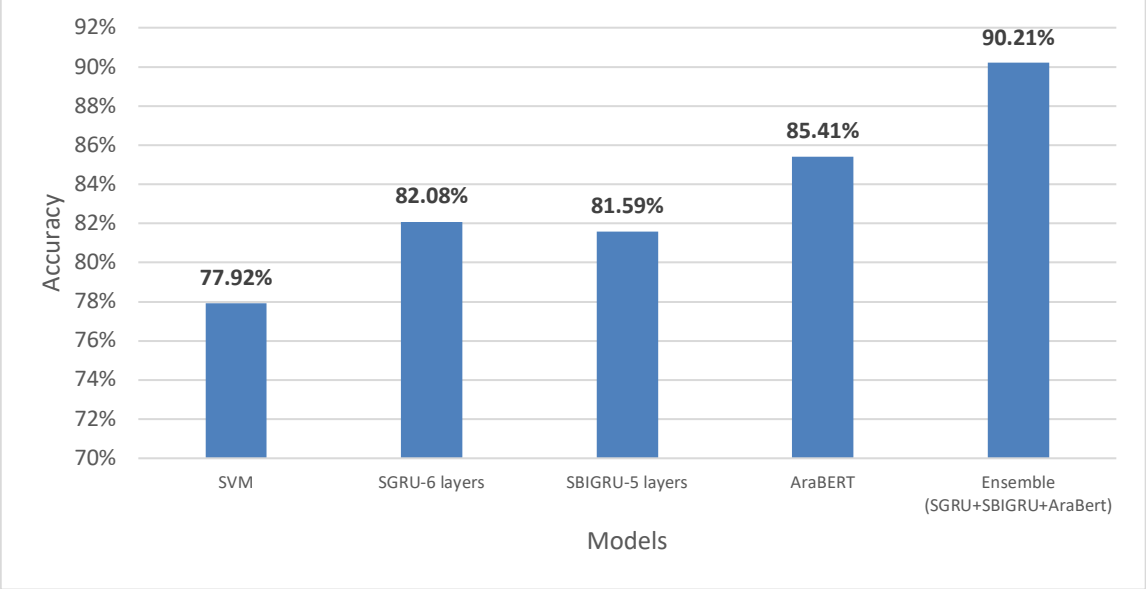

Figure 5. 4 Best Results from All Models

As shown in Figure 5.4, the Ensemble model outperforms SGRU, SBi-GRU, and AraBERT with an accuracy of 90.21%. Other models did not cross the 86% mark in the accuracy measure. Based on the results given in Figure 5.4, I answer RQ7 and RQ8 as follows

RQ7: How do transformers create an impact on the overall accuracy of both SGRU and SBi-GRU?

I used the AraBERT, a Bidirectional Transformer Encoder stacked based on the BERT model. This model is commonly regarded as the basis for most advanced findings in several languages in various NLP tasks. The BERT-based configuration, with 12 encoders blocks, 768 hidden sizes, 12 focus heads, a maximum sequence length of 512, and a total of 110M parameters, is used. This helped sentiment analyzer to provide better accuracy and performance in comparison to the models developed using deep learning techniques like GRU. Ensemble methods are successful as they use the combined strength of different classifiers. Here, AraBERT is successful because it uses the excellent representational power of transformers, using the SGRU and SBi-GRU with the transformers encourage extra diversity in the ensembles.

RQ 8: What are the performance differences between the ensemble method and singular methods?

Transformers-based AraBERT model outperformed every other model in terms of accuracy when compared with the single predicting model. In terms of the ensemble of best of the models, this newly proposed model achieved 90% accuracy. The same BERT-Base configuration is used for AraBERT. The model AraBERTv0.1 and AraBERTv1 are two models, and AraBERTv1 uses pre-segmented text with its Farasa segmenter prefixes and suffixes split. The model is trained in Arabic with words of ~70 M or ~23 GB in size count. Methods assembled help reduce these factors like unwanted errors. The ensemble created is more accurate than its individual components. Within the classification context, the individual components generate different decision boundaries with independent errors are produced by each classifier, and combining these errors usually reduces the total error. Because every sentiment classification method has its advantages and disadvantages, the overall accuracy of many different sentiment classifiers, with majority vote, give higher than any individual sentiment classifier.

5.4 Conclusion

This chapter evaluated the application of different model approaches in Arabic sentiment analysis and how ensemble models increase the accuracy of classifying the tweets by utilizing specialized learners as opposed to singular models. Different combination models were tested to determine which models give the best accuracy. The Ensemble model gives the best result compared to other models, with around 5% greater accuracy than the best singular base classifier.

CHAPTER SIX: CONCLUSION AND FUTURE SCOPE

6.1 Introduction

Using RNNs for sentiment analysis yielded accurate results due to its ability in using previous sequential states to compute the current input, which is suitable for the natural language context. Throughout the thesis, I presented a systematic review of Arabic sentiment analysis using RNN, in addition to answering eight research questions. These questions covered the neural network approaches for sentiment analysis, the RNN approaches for sentiment analysis and the related studies that use RNN for Arabic sentiment analysis, along with the research gaps. Also, the proposed SGRU model for Arabic sentiment classification using gulf dialect tweets was presented. A simple GRU, SVM model have been used as benchmarks to compare their performance with our proposed model. An ensemble model from three different models (SGRU, SBi-GRU, AraBERT) was implemented and compared its performance with the stacked models' best results.

6.2 Future Scope

The future directions include the following: develop Arabic word-embedding models that deal with negation without ignoring word structure. In addition, explore other types of models to identify the most suitable model for analyzing emojis and to enhance document and aspect classification. Moreover, emojis have a high potential for sentiment indication, given the challenge of analyzing the exact sentiment, which can be presented as a future direction. Furthermore, several algorithms can be combined for efficient sentiment analysis in large datasets, since the accuracy of hybrid models is higher than that of the singular models. A hybrid model needs to be constructed for Arabic text analyses considering grammatical composition and semantic accuracy to learn further representation in each layer. Therefore, various CNN and RNN algorithms need to be developed and trained with a variable number of parallel torsion layers, which can be considered for future studies.

6.3 Challenges

In this part I list the challenges that I faced during preparing the thesis. The section has been divided into two parts: Writing challenges, and implementation challenges. In the writing challenge's part, I discuss the challenges that I faced during the writing phase. In the implementation challenge's part, I discuss the challenges that I faced when I implemented the model.

1. Writing Challenges:

The idea was about SGRU, as I start preparing the proposal in late 2018, however, during this time, several papers were published using this method. This shifted our focus to expand ideas and exploring topics such as I try to add ideas such ass adding recent papers along with the transformers to strength the thesis. This also implies expanding the literature with state-of-the-art studies to address the new territory I focus on. In addition, at the end of 2019, I realized that I have to add the recent studies for 2019, that means I need to update the literature review with these studies. Moreover, writing the research gaps was challenging and took a long time to write.

2. Implementation Challenges:

Experiments have been done for the preprocessing phase to find the suitable steps. For example, collecting a stop word list for preprocessing was time-consuming and the collected words weren't effective for increasing the accuracy. Moreover, I try to discard the

first 1000 frequent words and the accuracy has decreased. This challenge opens a chance of finding a new way of discarding noisy words using auto-generated stop words.

In addition, I faced a problem with unifying the *Keras* and *TensorFlow* versions for both SGRU and SBi-GRU.]. The best setting for our models was (*Keras 2.2.4, Tensorflow1.13.1*) for SGRU, and (*Keras 2.4.3*, *TensorFlow 2.3.0*) for SBi-GRU. I discovered that *Tensor Board* for plotting both models require *TensorFlow* version 2 or more, I could not use the previous version of *TensorFlow*. The versions (*Keras 2.3.1*, *TensorFlow 2.2.0)* work for both models and with Tensor Board with no issues.

Moreover, when implementing the SBi-GRU model using Zhou et al. paper [147], I found the code as Java. In addition, I did not use the same configuration such as the number of epochs, the batch size, and the dropout rate.

In addition, I spent a long time to find the best configuration. Regarding the epoch number, the accuracy will increase if I increase the number of epochs, however the time given by Google Colab and the number of models implemented enforce us to keep the epoch =5. Also, starting with 265 units at the first layer was good but the accuracy decreased immediately when I add the second layer. I decided to keep the number of unit unified (100 units) with small number of dropout rates to utilize as units as possible.

6.4 Conclusion

This chapter reviewed a summary of what has been done in the thesis. Moreover, the future work was discussed by listing the possible enhancements in the sentiment analysis field.

I conclude that the ensemble model contains GRU and the transformers in this sentiment analysis task outperforms the singular models of SVM, GRU and transformers separately. This is because it is a sequential task on long-term sequences. since the length of tweets is restricted to 140 characters only. The ensemble approach can improve the overall accuracy of individual approach in twitter sentiment classification in this domain. The proposed model can be developed and enhanced using recent techniques such as the attention mechanism. In addition, the ensemble method is effective for gaining the best results with the power of transformers.

REFERENCES

[1] D. Reinsel, J. Gantz, and J. Rydning, "The Digitization of the World From Edge to Core," An IDC White Paper sponsored by seagate #US44413318, Nov. 2018. [Online]. Available: https://www.seagate.com/www-content/our-story/trends/files/idc-seagate-dataage-whitepaper.pdf.

[2] S. Kemp, "Digital In 2018: Essential Insights into Internet, Social Media, Mobile, and Ecommerce Use Around The World," We Are Social&Hootsuite, 2018.

[3] M. Heikal, M. Torki, and N. El-Makky, "Sentiment Analysis of Arabic Tweets using Deep Learning," *Procedia Comput. Sci.*, vol. 142, pp. 114–122, 2018, doi: 10.1016/j.procs.2018.10.466.

[4] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015, doi: 10.1038/nature14539.

[5] A. Souri, Z. El Maazouzi, M. Al Achhab, and B. E. El Mohajir, "Arabic Text Generation Using Recurrent Neural Networks," in *Big Data, Cloud and Applications*, Cham, 2018, pp. 523–533.

[6] M. M. Najafabadi, F. Villanustre, T. M. Khoshgoftaar, N. Seliya, R. Wald, and E. Muharemagic, "Deep learning applications and challenges in big data analytics," *J. Big Data*, vol. 2, no. 1, Dec. 2015, doi: 10.1186/s40537-014-0007-7.

[7] M. Al-Smadi, O. Qawasmeh, M. Al-Ayyoub, Y. Jararweh, and B. Gupta, "Deep Recurrent neural network vs. support vector machine for aspect-based sentiment analysis of Arabic hotels' reviews," *J. Comput. Sci.*, vol. 27, pp. 386–393, Jul. 2018, doi: 10.1016/j.jocs.2017.11.006.

[8] A. Yadav and D. K. Vishwakarma, "Sentiment analysis using deep learning architectures: a review," *Artif. Intell. Rev.*, Dec. 2019, doi: 10.1007/s10462-019-09794-5.

[9] W. Yin, K. Kann, M. Yu, and H. Schütze, "Comparative Study of CNN and RNN for Natural Language Processing," 2017.

[10] M. Hermans and B. Schrauwen, "Training and Analyzing Deep Recurrent Neural Networks," in *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 1*, USA, 2013, pp. 190–198, [Online]. Available: http://dl.acm.org/citation.cfm?id=2999611.2999633.

[11] W. Medhat, A. Hassan, and H. Korashy, "Sentiment analysis algorithms and applications: A survey," *Ain Shams Eng. J.*, vol. 5, no. 4, pp. 1093–1113, Dec. 2014, doi: 10.1016/j.asej.2014.04.011.

[12] L. Zhang, S. Wang, and B. Liu, "Deep learning for sentiment analysis: A survey," *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.*, vol. 8, no. 4, p. e1253, Jul. 2018, doi: 10.1002/widm.1253.

[13] F. Hemmatian and M. K. Sohrabi, "A survey on classification techniques for opinion mining and sentiment analysis," *Artif. Intell. Rev.*, Dec. 2017, doi: 10.1007/s10462-017-9599-6.

[14] S. Rani and P. Kumar, "A journey of Indian languages over sentiment analysis: a systematic review," *Artif. Intell. Rev.*, vol. 52, no. 2, pp. 1415–1462, Aug. 2019, doi: 10.1007/s10462-018-9670-y.

[15] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, Sep. 1995, doi: 10.1007/BF00994018.

[16] Z. Madhoushi, A. R. Hamdan, and S. Zainudin, "Sentiment analysis techniques in recent works," in *2015 Science and Information Conference (SAI)*, Jul. 2015, pp. 288–291, doi: 10.1109/SAI.2015.7237157.

[17] S. Alhumoud, T. Albuhairi, and M. Altuwaijri, "Arabic Sentiment Analysis using WEKA a Hybrid Learning Approach:," in *Proceedings of the 7th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management*, Lisbon, Portugal, 2015, pp. 402–408, doi: 10.5220/0005616004020408.

[18] S. Alhumoud, T. Albuhairi, and W. Alohaideb, "Hybrid sentiment analyser for Arabic tweets using R," in *2015 7th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K)*, Nov. 2015, vol. 01, pp. 417–424.

[19] B. Pang, L. Lee, and S. Vaithyanathan, "Thumbs up?: sentiment classification using machine learning techniques," in *Proceedings of the ACL-02 conference on Empirical methods in natural language processing -*

EMNLP '02, Not Known, 2002, vol. 10, pp. 79–86, doi: 10.3115/1118693.1118704.

[20] M. S. M. Vohra and J. Teraiya, "A COMPARATIVE STUDY OF SENTIMENT ANALYSIS TECHNIQUES 1," vol. 02, no. 02, pp. 313–317, 2013.

[21] B. Liu and L. Zhang, "A Survey of Opinion Mining and Sentiment Analysis," in *Mining Text Data*, C. C. Aggarwal and C. Zhai, Eds. Boston, MA: Springer US, 2012, pp. 415–463.

[22] S. Alhumoud, T. Albuhairi, and M. Altuwaijri, "Arabic sentiment analysis using WEKA a hybrid learning approach," in *2015 7th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K)*, Nov. 2015, vol. 01, pp. 402–408.

[23] J. M. Wiebe, R. F. Bruce, and T. P. O'Hara, "Development and Use of a Gold-Standard Data Set for Subjectivity Classifications," in *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, College Park, Maryland, USA, Jun. 1999, pp. 246–253, doi: 10.3115/1034678.1034721.

[24] M. Hu and B. Liu, "Mining and summarizing customer reviews," in *Proceedings of the 2004 ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '04*, Seattle, WA, USA, 2004, p. 168, doi: 10.1145/1014052.1014073.

[25] Y. Zhang, R. Jin, and Z.-H. Zhou, "Understanding bag-of-words model: a statistical framework," *Int. J. Mach. Learn. Cybern.*, vol. 1, no. 1–4, pp. 43–52, Dec. 2010, doi: 10.1007/s13042-010-0001-0.

[26] L. Augustyniak *et al.*, "Simpler is better? Lexicon-based ensemble sentiment classification beats supervised methods," in *2014 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2014)*, China, Aug. 2014, pp. 924–929, doi: 10.1109/ASONAM.2014.6921696.

[27] G. Paltoglou and M. Thelwall, "More than Bag-of-Words: Sentence-based Document Representation for Sentiment Analysis," in *Proceedings of the International Conference Recent Advances in Natural Language Processing RANLP 2013*, Hissar, Bulgaria, Sep. 2013, pp. 546–552, [Online]. Available: https://www.aclweb.org/anthology/R13-1072.

[28] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient Estimation of Word Representations in Vector Space," in *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*, 2013.

[29] J. Pennington, R. Socher, and C. Manning, "Glove: Global Vectors for Word Representation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar, 2014, pp. 1532–1543, doi: 10.3115/v1/D14-1162.

[30] A. Vaswani *et al.*, "Attention is All you Need," in *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. Curran Associates, Inc., 2017, pp. 5998–6008.

[31] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," 2019.

[32] A. B. Soliman, K. Eissa, and S. R. El-Beltagy, "AraVec: A set of Arabic Word Embedding Models for use in Arabic NLP," *Procedia Comput. Sci.*, vol. 117, pp. 256–265, 2017, doi: 10.1016/j.procs.2017.10.117.

[33] A. M. Alayba, V. Palade, M. England, and R. Iqbal, "Improving Sentiment Analysis in Arabic Using Word Representation," in *2018 IEEE 2nd International Workshop on Arabic and Derived Script Analysis and Recognition (ASAR)*, London, Mar. 2018, pp. 13–18, doi: 10.1109/ASAR.2018.8480191.

[34] A. A. Altowayan and L. Tao, "Word embeddings for Arabic sentiment analysis," in *2016 IEEE International Conference on Big Data (Big Data)*, Washington DC,USA, Dec. 2016, pp. 3820–3825, doi: 10.1109/BigData.2016.7841054.

[35] I. A. El-khair, "1.5 billion words Arabic Corpus," *ArXiv161104033 Cs*, Nov. 2016, Accessed: Feb. 08, 2019. [Online]. Available: http://arxiv.org/abs/1611.04033.

[36] A. M. Alayba, V. Palade, M. England, and R. Iqbal, "Arabic Language Sentiment Analysis on Health Services," *2017 1st Int. Workshop Arab. Scr. Anal. Recognit. ASAR*, pp. 114–118, Apr. 2017, doi: 10.1109/ASAR.2017.8067771.

[37] A. Mourad and K. Darwish, “Subjectivity and Sentiment Analysis of Modern Standard Arabic and Arabic Microblogs,” in *Proceedings of the 4th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, Atlanta, Georgia, 2013, pp. 55–64, [Online]. Available: http://aclweb.org/anthology/W13-1608.

[38] C. Banea, R. Mihalcea, and J. Wiebe, “Multilingual Subjectivity: Are More Languages Better?,” in *Coling 2010 - 23rd International Conference on Computational Linguistics, Proceedings of the Conference*, 2010, vol. 2, pp. 28–36.

[39] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, “Enriching Word Vectors with Subword Information,” *Trans. Assoc. Comput. Linguist.*, vol. 5, pp. 135–146, 2017, doi: 10.1162/tacl_a_00051.

[40] J. Schmidhuber, “Deep learning in neural networks: An overview,” *Neural Netw.*, vol. 61, pp. 85–117, Jan. 2015, doi: 10.1016/j.neunet.2014.09.003.

[41] A. Arabiyat, “Automatic Arabic Text Diacritization Using Recurrent Neural Networks,” The University of Jordan, 2017.

[42] T. Young, D. Hazarika, S. Poria, and E. Cambria, “Recent Trends in Deep Learning Based Natural Language Processing [Review Article],” *IEEE Comput. Intell. Mag.*, vol. 13, no. 3, pp. 55–75, Aug. 2018, doi: 10.1109/MCI.2018.2840738.

[43] V. Rohith and M. D, “SENTIMENT ANALYSIS ON TWITTER: A SURVEY,” *Int. J. Pure Appl. Math.*, vol. 118, no. 22, pp. 365–375, 2018.

[44] Q. T. Ain *et al.*, “Sentiment Analysis Using Deep Learning Techniques: A Review,” 2017, vol. 8, pp. 424–433.

[45] Y. Kim, “Convolutional Neural Networks for Sentence Classification,” in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar, 2014, pp. 1746–1751, doi: 10.3115/v1/D14-1181.

[46] T. Kobayashi, K. Hirose, and S. Nakamura, Eds., “Recurrent neural network based language model,” Makuhari, Chiba, Japan, Sep. 2010, pp. 1045--1048, [Online]. Available: http://www.isca-speech.org/archive/interspeech_2010/i10_1045.html.

[47] O. Irsoy and C. Cardie, “Opinion Mining with Deep Recurrent Neural Networks,” in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar, 2014, pp. 720–728, doi: 10.3115/v1/D14-1080.

[48] K. S. Tai, R. Socher, and C. D. Manning, “Improved Semantic Representations From Tree-Structured Long Short-Term Memory Networks,” in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Beijing, China, 2015, pp. 1556–1566, doi: 10.3115/v1/P15-1150.

[49] D. Tang, B. Qin, and T. Liu, “Document Modeling with Gated Recurrent Neural Network for Sentiment Classification,” in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, Lisbon, Portugal, 2015, pp. 1422–1432, doi: 10.18653/v1/D15-1167.

[50] R. Socher *et al.*, “Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank,” in *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, Seattle, Washington, USA, 2013, pp. 1631–1642, Accessed: Jan. 15, 2019. [Online]. Available: http://aclweb.org/anthology/D13-1170.

[51] B. Pang and L. Lee, “Opinion Mining and Sentiment Analysis,” *Found. Trends® Inf. Retr.*, vol. 2, no. 1–2, pp. 1–135, 2008, doi: 10.1561/1500000011.

[52] A. Alsayat and N. Elmitwally, “A comprehensive study for Arabic Sentiment Analysis (Challenges and Applications),” *Egypt. Inform. J.*, vol. 21, no. 1, pp. 7–12, Mar. 2020, doi: 10.1016/j.eij.2019.06.001.

[53] N. A. Abdulla, N. A. Ahmed, M. A. Shehab, and M. Al-Ayyoub, “Arabic sentiment analysis: Lexicon-based and corpus-based,” in *2013 IEEE Jordan Conference on Applied Electrical Engineering and Computing Technologies (AEECT)*, Amman, Jordan, Dec. 2013, pp. 1–6, doi: 10.1109/AEECT.2013.6716448.

[54] I. M. Saleh, “Automatic extraction of lemma-based bilingual dictionaries for morphologically rich languages,” 2009.

[55] N. Y. Habash, “Introduction to Arabic Natural Language Processing,” *Synth. Lect. Hum. Lang. Technol.*, vol. 3, no. 1, pp. 1–187, Jan. 2010, doi: 10.2200/S00277ED1V01Y201008HLT010.

[56] I. Turki Khemakhem, S. Jamoussi, and A. Ben Hamadou, “Arabic morpho-syntactic feature disambiguation in a translation context,” in *Proceedings of the 4th Workshop on Syntax and Structure in Statistical Translation*, Beijing, China, Aug. 2010, pp. 61–65, [Online]. Available: https://www.aclweb.org/anthology/W10-3808.

[57] N. Boudad, R. Faizi, R. Oulad Haj Thami, and R. Chiheb, “Sentiment analysis in Arabic: A review of the literature,” *Ain Shams Eng. J.*, vol. 9, no. 4, pp. 2479–2490, Dec. 2018, doi: 10.1016/j.asej.2017.04.007.

[58] K. Dashtipour *et al.*, “Multilingual Sentiment Analysis: State of the Art and Independent Comparison of Techniques,” *Cogn. Comput.*, vol. 8, no. 4, pp. 757–771, Aug. 2016, doi: 10.1007/s12559-016-9415-7.

[59] O. Oueslati, E. Cambria, M. Ben HajHmida, and H. Ounelli, “A review of sentiment analysis research in Arabic language,” *Future Gener. Comput. Syst.*, 2020.

[60] B. Kitchenham, “Procedures for Performing Systematic Reviews,” *Keele UK Keele Univ*, vol. 33, 2004.

[61] S. Heckman and L. Williams, “A Systematic Literature Review of Actionable Alert Identification Techniques for Automated Static Code Analysis,” *Inf Softw Technol*, vol. 53, no. 4, pp. 363–387, Apr. 2011, doi: 10.1016/j.infsof.2010.12.007.

[62] N. C. D. Adhikari *et al.*, “Sentiment Classifier and Analysis for Epidemic Prediction,” in *Computer Science & Information Technology (CS & IT)*, Jul. 2018, pp. 31–48, doi: 10.5121/csit.2018.81004.

[63] T. Khalil, A. Halaby, M. Hammad, and S. R. El-Beltagy, “Which Configuration Works Best? An Experimental Study on Supervised Arabic Twitter Sentiment Analysis,” in *2015 First International Conference on Arabic Computational Linguistics (ACLing)*, Cairo, Egypt, Apr. 2015, pp. 86–93, doi: 10.1109/ACLing.2015.19.

[64] H. Al-Rubaiee, R. Qiu, and D. Li, “Identifying Mubasher software products through sentiment analysis of Arabic tweets,” in *2016 International Conference on Industrial Informatics and Computer Systems (CIICS)*, Mar. 2016, pp. 1–6, doi: 10.1109/ICCSII.2016.7462396.

[65] W. Al-Harbi and A. Emam, “Effect of Saudi dialect preprocessing on Arabic sentiment analysis,” 2015, pp. 91-99.

[66] A. Assiri, A. Emam, and Hmood Al-Dossari, “Saudi Twitter Corpus For Sentiment Analysis,” *Zenodo*, Jan. 2016, doi: 10.5281/zenodo.1338816.

[67] A. S. Al-Subaihin and H. S. Al-Khalifa, “A System for Sentiment Analysis of Colloquial Arabic Using Human Computation,” *Sci. World J.*, vol. 2014, pp. 1–8, 2014, doi: 10.1155/2014/631394.

[68] J. Ben Salamah and A. Elkhlifi, “Microblogging Opinion Mining Approach for Kuwaiti Dialect,” Dubai, UAE, 2014.

[69] A. Al-Thubaity, M. Alharbi, S. Alqahtani, and A. Aljandal, “A Saudi Dialect Twitter Corpus for Sentiment and Emotion Analysis,” in *2018 21st Saudi Computer Society National Computer Conference (NCC)*, Apr. 2018, pp. 1–6, doi: 10.1109/NCG.2018.8592998.

[70] H. K. Aldayel and A. M. Azmi, “Arabic tweets sentiment analysis – a hybrid scheme,” *J. Inf. Sci.*, vol. 42, no. 6, pp. 782–797, Dec. 2016, doi: 10.1177/0165551515610513.

[71] R. M. Alahmary, H. Z. Al-Dossari, and A. Z. Emam, “Sentiment Analysis of Saudi Dialect Using Deep Learning Techniques,” in *2019 International Conference on Electronics, Information, and Communication (ICEIC)*, Jan. 2019, pp. 1–6, doi: 10.23919/ELINFOCOM.2019.8706408.

[72] A. Mustafa, S. A., and S. Sohail, “Sentiment Classification of Twitter Data Belonging to Saudi Arabian Telecommunication Companies,” *Int. J. Adv. Comput. Sci. Appl.*, vol. 8, no. 1, 2017, doi: 10.14569/IJACSA.2017.080150.

[73] R. Baly *et al.*, “Comparative Evaluation of Sentiment Analysis Methods Across Arabic Dialects,” *Procedia Comput. Sci.*, vol. 117, pp. 266–273, 2017, doi: 10.1016/j.procs.2017.10.118.

[74] W. Adouane and R. Johansson, “Gulf Arabic Linguistic Resource Building for Sentiment Analysis,” in *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC’16)*, Portorož, Slovenia, May 2016, pp. 2710–2715, [Online]. Available: https://www.aclweb.org/anthology/L16-

1430.

[75] A. Al-Obaidi and V. Samawi, “Opinion Mining: Analysis of Comments Written in Arabic Colloquial,” San Francisco, USA, 2016.

[76] hanadi Al Suwaidi, T. R. Soomro, and K. Shaalan, “Sentiment analysis for emiriti dialects in twitter,” 2016, pp. 707–710.

[77] A. Alqarafi, A. Adeel, A. Hawalah, K. Swingler, and A. Hussain, “A Semi-supervised Corpus Annotation for Saudi Sentiment Analysis Using Twitter,” in *Advances in Brain Inspired Cognitive Systems*, Cham, 2018, pp. 589–596.

[78] N. Al-Twairesh, H. Al-Khalifa, A. Al-Salman, and Y. Al-Ohali, “AraSenTi-Tweet: A Corpus for Arabic Sentiment Analysis of Saudi Tweets,” *Procedia Comput. Sci.*, vol. 117, pp. 63–72, 2017, doi: 10.1016/j.procs.2017.10.094.

[79] A. Al-Thubaity, Q. Alqahtani, and A. Aljandal, “Sentiment lexicon for sentiment analysis of Saudi dialect tweets,” *Procedia Comput. Sci.*, vol. 142, pp. 301–307, 2018, doi: 10.1016/j.procs.2018.10.494.

[80] A. M. Azmi and S. M. Alzanin, “Aara’– a system for mining the polarity of Saudi public opinion through e-newspaper comments,” *J. Inf. Sci.*, vol. 40, no. 3, pp. 398–410, Jun. 2014, doi: 10.1177/0165551514524675.

[81] A. Karpathy, “The Unreasonable Effectiveness of Recurrent Neural Networks,” *Andrej Karpathy blog*, May 21, 2015. http://karpathy.github.io/2015/05/21/rnn-effectiveness/.

[82] Y. Bengio, P. Simard, and P. Frasconi, “Learning long-term dependencies with gradient descent is difficult,” *IEEE Trans. Neural Netw.*, vol. 5, no. 2, pp. 157–166, Mar. 1994, doi: 10.1109/72.279181.

[83] M. Schuster and K. K. Paliwal, “Bidirectional recurrent neural networks,” *IEEE Trans. Signal Process.*, vol. 45, no. 11, pp. 2673–2681, Nov. 1997, doi: 10.1109/78.650093.

[84] R. Pascanu, T. Mikolov, and Y. Bengio, “On the Difficulty of Training Recurrent Neural Networks,” in *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28*, Atlanta, GA, USA, 2013, p. III-1310-III–1318.

[85] P. Wang, Y. Qian, F. K. Soong, L. He, and H. Zhao, “A Unified Tagging Solution: Bidirectional LSTM Recurrent Neural Network with Word Embedding.,” *CoRR*, vol. abs/1511.00215, 2015.

[86] J. Schmidhuber, “Learning Complex, Extended Sequences Using the Principle of History Compression,” *Neural Comput*, vol. 4, no. 2, pp. 234–242, Mar. 1992, doi: 10.1162/neco.1992.4.2.234.

[87] S. El Hihi and Y. Bengio, “Hierarchical Recurrent Neural Networks for Long-term Dependencies,” in *Proceedings of the 8th International Conference on Neural Information Processing Systems*, Cambridge, MA, USA, 1995, pp. 493–499, [Online]. Available: http://dl.acm.org/citation.cfm?id=2998828.2998898.

[88] R. Pascanu, C. Gulcehre, K. Cho, and Y. Bengio, “How to Construct Deep Recurrent Neural Networks,” Apr. 2014, Accessed: Nov. 09, 2019. [Online].

[89] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, doi: 10.1162/neco.1997.9.8.1735.

[90] A. Graves, A. Mohamed, and G. Hinton, “Speech recognition with deep recurrent neural networks,” in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, Vancouver, BC, Canada, May 2013, pp. 6645–6649, doi: 10.1109/ICASSP.2013.6638947.

[91] C. Olah, “Understanding LSTM Networks,” *http://colah.github.io/posts/2015-08-Understanding-LSTMs/*. http://colah.github.io/posts/2015-08-Understanding-LSTMs/.

[92] K. Cho *et al.*, “Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation,” in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar, 2014, pp. 1724–1734, doi: 10.3115/v1/D14-1179.

[93] J. Chung, Ç. Gülçehre, K. Cho, and Y. Bengio, “Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling,” in *NIPS 2014 Workshop on Deep Learning, December 2014*, 2014, vol. abs/1412.3555.

[94] M. Aly and A. Atiya, “LABR: A Large Scale Arabic Book Reviews Dataset,” *Unpublished*, 2013, doi:

10.13140/2.1.3960.5761.

[95] M. Abbes, Z. Kechaou, and A. M. Alimi, “Enhanced Deep Learning Models for Sentiment Analysis in Arab Social Media,” in *Neural Information Processing*, vol. 10638, D. Liu, S. Xie, Y. Li, D. Zhao, and E.-S. M. El-Alfy, Eds. Cham: Springer International Publishing, 2017, pp. 667–676.

[96] L. H. Baniata and S.-B. Park, “Sentence Representation Network for Arabic Sentiment Analysis,” in *In Proceedings of The 43rd Annual Meeting and Winter Conference (제43회 정기총회 및 동계학술발표회)*, Gangwon-do , South Korea, Dec. 2016, pp. 470–472.

[97] M. Pontiki, D. Galanis, H. Papageorgiou, S. Manandhar, and I. Androutsopoulos, “SemEval-2016 Task 5 data and tools.” http://alt.qcri.org/semeval2016/task5/index.php?id=data-and-tools.

[98] A. Tamchyna and K. Veselovská, “UFAL at SemEval-2016 Task 5: Recurrent Neural Networks for Sentence Classification,” in *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, San Diego, California, 2016, pp. 367–371, doi: 10.18653/v1/S16-1059.

[99] S. Ruder, P. Ghaffari, and J. G. Breslin, “A Hierarchical Model of Reviews for Aspect-based Sentiment Analysis,” in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, Austin, Texas, Nov. 2016, pp. 999–1005, doi: 10.18653/v1/D16-1103.

[100] B. Wang and W. Lu, “Learning Latent Opinions for Aspect-level Sentiment Classification,” 2018, pp. 5537–5544.

[101] E. M. Ponti, I. Vulić, and A. Korhonen, “Decoding Sentiment from Distributed Representations of Sentences,” in *Proceedings of the 6th Joint Conference on Lexical and Computational Semantics (*SEM 2017)*, Vancouver, Canada, Aug. 2017, pp. 22–32, doi: 10.18653/v1/S17-1003.

[102] M. Al-Smadi, B. Talafha, M. Al-Ayyoub, and Y. Jararweh, “Using long short-term memory deep neural networks for aspect-based sentiment analysis of Arabic reviews,” *Int. J. Mach. Learn. Cybern.*, vol. 10, no. 8, pp. 2163–2175, Aug. 2019, doi: 10.1007/s13042-018-0799-4.

[103] M. Nabil, M. Aly, and A. Atiya, “ASTD: Arabic Sentiment Tweets Dataset,” in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, Lisbon, Portugal, 2015, pp. 2515–2519, doi: 10.18653/v1/D15-1299.

[104] S. Al-Azani and E.-S. M. El-Alfy, “Hybrid Deep Learning for Sentiment Polarity Determination of Arabic Microblogs,” in *Neural Information Processing*, vol. 10635, D. Liu, S. Xie, Y. Li, D. Zhao, and E.-S. M. El-Alfy, Eds. Cham: Springer International Publishing, 2017, pp. 491–500.

[105] S. Al-Azani and E.-S. El-Alfy, “Emojis-Based Sentiment Classification of Arabic Microblogs Using Deep Recurrent Neural Networks,” in *2018 International Conference on Computing Sciences and Engineering (ICCSE)*, Kuwait City, Mar. 2018, pp. 1–6, doi: 10.1109/ICCSE1.2018.8374211.

[106] A. Baccouche, B. Garcia-Zapirain, and A. Elmaghraby, “Annotation Technique for Health-Related Tweets Sentiment Analysis,” in *2018 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)*, Louisville, KY, USA, Dec. 2018, pp. 382–387, doi: 10.1109/ISSPIT.2018.8642685.

[107] A. M. Alayba, V. Palade, M. England, and R. Iqbal, “A Combined CNN and LSTM Model for Arabic Sentiment Analysis,” in *Machine Learning and Knowledge Extraction*, vol. 11015, A. Holzinger, P. Kieseberg, A. M. Tjoa, and E. Weippl, Eds. Cham: Springer International Publishing, 2018, pp. 179–191.

[108] S. M. Mohammad, M. Salameh, and S. Kiritchenko, “How Translation Alters Sentiment,” *J. Artif. Intell. Res.*, vol. 55, pp. 95–130, Jan. 2016, doi: 10.1613/jair.4787.

[109] S. Rosenthal, N. Farra, and P. Nakov, “SemEval-2017 task 4: Sentiment analysis in Twitter,” in *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, 2017, pp. 502–518.

[110] J.-Á. González, F. Pla, and L.-F. Hurtado, “ELiRF-UPV at SemEval-2017 Task 4: Sentiment Analysis using Deep Learning,” in *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, Vancouver, Canada, 2017, pp. 723–727, doi: 10.18653/v1/S17-2121.

[111] A. E. Samy, S. R. El-Beltagy, and E. Hassanien, “A Context Integrated Model for Multi-label Emotion Detection,” *Procedia Comput. Sci.*, vol. 142, pp. 61–71, 2018, doi: 10.1016/j.procs.2018.10.461.

[112] S. Mohammad, F. Bravo-Marquez, M. Salameh, and S. Kiritchenko, "SemEval-2018 Task 1: Affect in Tweets," in *Proceedings of The 12th International Workshop on Semantic Evaluation*, New Orleans, Louisiana, 2018, pp. 1–17, doi: 10.18653/v1/S18-1001.

[113] M. Abdullah and S. Shaikh, "TeamUNCC at SemEval-2018 Task 1: Emotion Detection in English and Arabic Tweets using Deep Learning," in *Proceedings of The 12th International Workshop on Semantic Evaluation*, New Orleans, Louisiana, 2018, pp. 350–357, doi: 10.18653/v1/S18-1053.

[114] M. Abdullah, M. Hadzikadicy, and S. Shaikhz, "SEDAT: Sentiment and Emotion Detection in Arabic Text Using CNN-LSTM Deep Learning," in *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, Dec. 2018, pp. 835–840, doi: 10.1109/ICMLA.2018.00134.

[115] H. Alhuzali, M. Abdul-Mageed, and L. H. Ungar, "Enabling Deep Learning of Emotion With First-Person Seed Expressions," 2018.

[116] M. Abdou, A. Kulmizev, and J. Ginés i Ametllé, "AffecThor at SemEval-2018 Task 1: A cross-linguistic approach to sentiment intensity quantification in tweets," in *Proceedings of The 12th International Workshop on Semantic Evaluation*, New Orleans, Louisiana, Jun. 2018, pp. 210–217, doi: 10.18653/v1/S18-1032.

[117] M. Abdul-Mageed, H. Alhuzali, D. Abu Elhija, and M. Diab, "Dina: A multidialect dataset for arabic emotion analysis," in *The 2nd Workshop on Arabic Corpora and Processing Tools 2016 Theme: Social Media*, May 2016, p. 29.

[118] A. Alwehaibi and K. Roy, "Comparison of Pre-Trained Word Vectors for Arabic Text Classification Using Deep Learning Approach," in *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, Dec. 2018, pp. 1471–1474, doi: 10.1109/ICMLA.2018.00239.

[119] A. Elnagar, "BRAD: Books Reviews in Arabic Dataset," *https://github.com/elnagara/BRAD-Arabic-Dataset*. https://github.com/elnagara/BRAD-Arabic-Dataset.

[120] A. Elnagar, L. Lulu, and O. Einea, "An Annotated Huge Dataset for Standard and Colloquial Arabic Reviews for Subjective Sentiment Analysis," *Procedia Comput. Sci.*, vol. 142, pp. 182–189, 2018, doi: 10.1016/j.procs.2018.10.474.

[121] A. El-Kilany, A. Azzam, and S. R. El-Beltagy, "Using Deep Neural Networks for Extracting Sentiment Targets in Arabic Tweets," in *Intelligent Natural Language Processing: Trends and Applications*, vol. 740, K. Shaalan, A. E. Hassanien, and F. Tolba, Eds. Cham: Springer International Publishing, 2018, pp. 3–15.

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

[123] G. Badaro, R. Baly, H. Hajj, N. Habash, and W. El-Hajj, "A Large Scale Arabic Sentiment Lexicon for Arabic Opinion Mining," in *Proceedings of the EMNLP 2014 Workshop on Arabic Natural Language Processing (ANLP)*, Doha, Qatar, 2014, pp. 165–173, doi: 10.3115/v1/W14-3623.

[124] S. Kiritchenko, S. Mohammad, and M. Salameh, "SemEval-2016 Task 7: Determining Sentiment Intensity of English and Arabic Phrases," in *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, San Diego, California, 2016, pp. 42–51, doi: 10.18653/v1/S16-1004.

[125] E. Refaee and V. Rieser, "An Arabic Twitter Corpus for Subjectivity and Sentiment Analysis," in *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, Reykjavik, Iceland, May 2014, pp. 2268–2273, [Online]. Available: http://www.lrec-conf.org/proceedings/lrec2014/pdf/317_Paper.pdf.

[126] A. A. Elmadany, H. Mubarak, and W. Magdy, "An Arabic Speech-Act and Sentiment Corpus of Tweets," 2018.

[127] M. AL-Smadi, O. Qwasmeh, B. Talafha, M. Al-Ayyoub, Y. Jararweh, and E. Benkhelifa, "An enhanced framework for aspect-based sentiment analysis of Hotels' reviews: Arabic reviews case study," in *2016 11th International Conference for Internet Technology and Secured Transactions (ICITST)*, Barcelona, Spain, Dec. 2016, pp. 98–103, doi: 10.1109/ICITST.2016.7856675.

[128] S. Ruder, P. Ghaffari, and J. G. Breslin, "INSIGHT-1 at SemEval-2016 Task 5: Deep Learning for Multilingual Aspect-based Sentiment Analysis," in *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, San Diego, California, 2016, pp. 330–336, doi: 10.18653/v1/S16-1053.

[129] A. Kumar, S. Kohail, A. Kumar, A. Ekbal, and C. Biemann, "IIT-TUDA at SemEval-2016 Task 5: Beyond Sentiment Lexicon: Combining Domain Dependency and Distributional Semantics Features for Aspect Based Sentiment Analysis," in *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, San Diego, California, 2016, pp. 1129–1135, doi: 10.18653/v1/S16-1174.

[130] M. Pontiki *et al.*, "SemEval-2016 task 5 : aspect based sentiment analysis," in *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, San Diego, California, 2016, pp. 19–30, [Online]. Available: http://www.aclweb.org/anthology/S16-1002.

[131] C. Brun, J. Perez, and C. Roux, "XRCE at SemEval-2016 Task 5: Feedbacked Ensemble Modeling on Syntactico-Semantic Knowledge for Aspect Based Sentiment Analysis," in *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, San Diego, California, 2016, pp. 277–281, doi: 10.18653/v1/S16-1044.

[132] G. Badaro *et al.*, "EMA at SemEval-2018 Task 1: Emotion Mining for Arabic," in *Proceedings of The 12th International Workshop on Semantic Evaluation*, New Orleans, Louisiana, 2018, pp. 236–244, doi: 10.18653/v1/S18-1036.

[133] M. Cliche, "BB_twtr at SemEval-2017 Task 4: Twitter Sentiment Analysis with CNNs and LSTMs," in *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, Vancouver, Canada, 2017, pp. 573–580, doi: 10.18653/v1/S17-2094.

[134] R. Baly *et al.*, "A Characterization Study of Arabic Twitter Data with a Benchmarking for State-of-the-Art Opinion Mining Models," in *Proceedings of the Third Arabic Natural Language Processing Workshop*, Valencia, Spain, 2017, pp. 110–118, doi: 10.18653/v1/W17-1314.

[135] N. Albadi, "Religious Hate Speech Detection for Arabic Tweets," *https://github.com/nuhaalbadi/Arabic_hatespeech*. https://github.com/nuhaalbadi/Arabic_hatespeech.

[136] M. A. Jerbi, H. Achour, and E. Souissi, "Sentiment Analysis of Code-Switched Tunisian Dialect: Exploring RNN-Based Techniques," in *Arabic Language Processing: From Theory to Practice*, vol. 1108, K. Smaïli, Ed. Cham: Springer International Publishing, 2019, pp. 122–131.

[137] "Sentiment Analysis for Arabic Text (tweets, reviews, and standard Arabic) using word2vec," *https://github.com/iamaziz/ar-embeddings*. https://github.com/iamaziz/ar-embeddings.

[138] "fasttext: Word vectors for 157 languages," *https://fasttext.cc/docs/en/crawl-vectors.html*. https://fasttext.cc/docs/en/crawl-vectors.html.

[139] M. Mataoui, O. Zelmati, and M. Boumechache, "A Proposed Lexicon-Based Sentiment Analysis Approach for the Vernacular Algerian Arabic," *Res. Comput. Sci.*, vol. 110, pp. 55–70, 2016.

[140] A. Dahou, S. Xiong, J. Zhou, M. H. Haddoud, and P. Duan, "Word Embeddings and Convolutional Neural Network for Arabic Sentiment Classification," in *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, Osaka, Japan, Dec. 2016, pp. 2418–2427, [Online]. Available: http://www.aclweb.org/anthology/C16-1228.

[141] K. Abu Kwaik, M. Saad, S. Chatzikyriakidis, and S. Dobnik, "LSTM-CNN Deep Learning Model for Sentiment Analysis of Dialectal Arabic," in *Arabic Language Processing: From Theory to Practice*, vol. 1108, K. Smaïli, Ed. Cham: Springer International Publishing, 2019, pp. 108–121.

[142] K. Abu Kwaik, M. Saad, S. Chatzikyriakidis, and S. Dobnik, "Shami: A Corpus of Levantine Arabic Dialects," Miyazaki, Japan, May 2018, [Online]. Available: https://www.aclweb.org/anthology/L18-1576.

[143] K. Elshakankery and M. F. Ahmed, "HILATSA: A hybrid Incremental learning approach for Arabic tweets sentiment analysis," *Egypt. Inform. J.*, vol. 20, no. 3, pp. 163–171, Nov. 2019, doi: 10.1016/j.eij.2019.03.002.

[144] A. Barhoumi, N. Camelin, C. Aloulou, Y. Estève, and L. Hadrich Belguith, "An Empirical Evaluation of Arabic-Specific Embeddings for Sentiment Analysis," in *Arabic Language Processing: From Theory to Practice*, vol. 1108, K. Smaïli, Ed. Cham: Springer International Publishing, 2019, pp. 34–48.

[145] A. U. Rehman, A. K. Malik, B. Raza, and W. Ali, "A Hybrid CNN-LSTM Model for Improving Accuracy of Movie Reviews Sentiment Analysis," *Multimed. Tools Appl.*, vol. 78, no. 18, pp. 26597–26613, Sep. 2019, doi: 10.1007/s11042-019-07788-7.

[146] R. Al-Rfou, B. Perozzi, and S. Skiena, “Polyglot: Distributed Word Representations for Multilingual NLP,” in *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, Sofia, Bulgaria, 2013, pp. 183–192, [Online]. Available: http://aclweb.org/anthology/W13-3520.

[147] J. Zhou, Y. Lu, H.-N. Dai, H. Wang, and H. Xiao, “Sentiment Analysis of Chinese Microblog Based on Stacked Bidirectional LSTM,” *IEEE Access*, vol. 7, pp. 38856–38866, 2019, doi: 10.1109/ACCESS.2019.2905048.

[148] Z. Xiao and P. Liang, “Chinese Sentiment Analysis Using Bidirectional LSTM with Word Embedding,” in *Cloud Computing and Security*, Cham, 2016, pp. 601–610.

[149] Y. Wen, A. Xu, W. Liu, and L. Chen, “A Wide Residual Network for Sentiment Classification,” in *Proceedings of the 2018 2nd International Conference on Deep Learning Technologies - ICDLT ’18*, Chongqing, China, 2018, pp. 7–11, doi: 10.1145/3234804.3234807.

[150] Z.-Y. Gao and C.-P. Chen, “AI Deep Learning with Multiple Labels for Sentiment Classification of Tweets,” in *2019 IEEE International Symposium on Circuits and Systems (ISCAS)*, Sapporo, Japan, May 2019, pp. 1–5, doi: 10.1109/ISCAS.2019.8702139.

[151] N. S. Sakenovich and A. S. Zharmagambetov, “On One Approach of Solving Sentiment Analysis Task for Kazakh and Russian Languages Using Deep Learning,” in *Computational Collective Intelligence*, vol. 9876, N. T. Nguyen, L. Iliadis, Y. Manolopoulos, and B. Trawiński, Eds. Cham: Springer International Publishing, 2016, pp. 537–545.

[152] N. K. Nguyen, A.-C. Le, and H. T. Pham, “Deep Bi-directional Long Short-Term Memory Neural Networks for Sentiment Analysis of Social Data,” in *Integrated Uncertainty in Knowledge Modelling and Decision Making*, Cham, 2016, pp. 255–268.

[153] D. Ma, S. Li, and H. Wang, “Target Extraction via Feature-Enriched Neural Networks Model,” in *Natural Language Processing and Chinese Computing*, vol. 11108, M. Zhang, V. Ng, D. Zhao, S. Li, and H. Zan, Eds. Cham: Springer International Publishing, 2018, pp. 353–364.

[154] M. Mitchell, J. Aguilar, T. Wilson, and B. Van Durme, “Open domain targeted sentiment,” Seattle, Washington, USA, Oct. 2013, pp. 1643–1654, [Online]. Available: http://www.aclweb.org/anthology/D13-1171.

[155] L.-C. Chen, C.-M. Lee, and M.-Y. Chen, “Exploration of social media for sentiment analysis using deep learning,” *Soft Comput.*, Oct. 2019, doi: 10.1007/s00500-019-04402-8.

[156] S. Pal, S. Ghosh, and A. Nag, “Sentiment Analysis in the Light of LSTM Recurrent Neural Networks:,” *Int. J. Synth. Emot.*, vol. 9, no. 1, pp. 33–39, Jan. 2018, doi: 10.4018/IJSE.2018010103.

[157] J. Hong and M. Fang, “Sentiment Analysis with Deeply Learned Distributed Representations of Variable Length Texts,” 2015.

[158] X. Xie, “Opinion Expression Detection via Deep Bidirectional C-GRUs,” in *2017 28th International Workshop on Database and Expert Systems Applications (DEXA)*, Lyon, France, Aug. 2017, pp. 118–122, doi: 10.1109/DEXA.2017.40.

[159] C. Wu, F. Wu, Y. Huang, S. Wu, and Z. Yuan, “THU NGN at IJCNLP-2017 Task 2: Dimensional Sentiment Analysis for Chinese Phrases with Deep LSTM,” 2017.

[160] C. Wu, F. Wu, S. Wu, J. Liu, Z. Yuan, and Y. Huang, “THU_NGN at SemEval-2018 Task 3: Tweet Irony Detection with Densely connected LSTM and Multi-task Learning,” in *Proceedings of The 12th International Workshop on Semantic Evaluation*, New Orleans, Louisiana, Jun. 2018, pp. 51–56, doi: 10.18653/v1/S18-1006.

[161] D. Anil, A. Vembar, S. Hiriyannaiah, S. Gm, and K. Srinivasa, “Performance Analysis of Deep Learning Architectures for Recommendation Systems,” in *2018 IEEE 25th International Conference on High Performance Computing Workshops (HiPCW)*, Bengaluru, India, Dec. 2018, pp. 129–136, doi: 10.1109/HiPCW.2018.8634192.

[162] A. Feizollah, S. Ainin, N. B. Anuar, N. A. B. Abdullah, and M. Hazim, “Halal Products on Twitter: Data Extraction and Sentiment Analysis Using Stack of Deep Learning Algorithms,” *IEEE Access*, vol. 7, pp. 83354–83362, 2019, doi: 10.1109/ACCESS.2019.2923275.

[163] J. Wiebe, T. Wilson, and C. Cardie, “Annotating Expressions of Opinions and Emotions in Language,” *Lang. Resour. Eval.*, vol. 39, no. 2–3, pp. 165–210, May 2005, doi: 10.1007/s10579-005-7880-9.

[164] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, “Learning Word Vectors for Sentiment Analysis,” in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, Portland, Oregon, USA, Jun. 2011, pp. 142–150.

[165] “SogouCA News.” https://www.sogou.com/labs/resource/ca.php.

[166] “wiki dump.” http://download.wikipedia.com/zhwiki/latest/zhwikilatest-pages-articles.xml.bz2.

[167] F. Godin, B. Vandersmissen, W. De Neve, and R. Van de Walle, “Multimedia Lab @ ACL WNUT NER Shared Task: Named Entity Recognition for Twitter Microposts using Distributed Word Representations,” in *Proceedings of the Workshop on Noisy User-generated Text*, Beijing, China, Jul. 2015, pp. 146–153, doi: 10.18653/v1/W15-4322.

[168] F. Barbieri, G. Kruszewski, F. Ronzano, and H. Saggion, “How Cosmopolitan Are Emojis?: Exploring Emojis Usage and Meaning over Different Languages with Distributional Semantics,” in *Proceedings of the 2016 ACM on Multimedia Conference - MM ’16*, Amsterdam, The Netherlands, 2016, pp. 531–535, doi: 10.1145/2964284.2967278.

[169] C. Van Hee, E. Lefever, and V. Hoste, “SemEval-2018 Task 3: Irony Detection in English Tweets,” in *Proceedings of The 12th International Workshop on Semantic Evaluation*, New Orleans, Louisiana, Jun. 2018, pp. 39–50, doi: 10.18653/v1/S18-1005.

[170] I. Sutskever, O. Vinyals, and Q. V. Le, “Sequence to Sequence Learning with Neural Networks,” in *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2014, pp. 3104–3112.

[171] W. Antoun, F. Baly, and H. Hajj, “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*, Marseille, France, May 2020, pp. 9–15, [Online]. Available: https://www.aclweb.org/anthology/2020.osact-1.2.

[172] X. Qiu, T. Sun, Y. Xu, Y. Shao, N. Dai, and X. Huang, “Pre-trained models for natural language processing: A survey,” *Sci. China Technol. Sci.*, vol. 63, no. 10, pp. 1872–1897, Oct. 2020, doi: 10.1007/s11431-020-1647-3.

[173] O. ElJundi, W. Antoun, N. El Droubi, H. Hajj, W. El-Hajj, and K. Shaban, “hULMonA: The Universal Language Model in Arabic,” in *Proceedings of the Fourth Arabic Natural Language Processing Workshop*, Florence, Italy, 2019, pp. 68–77, doi: 10.18653/v1/W19-4608.

[174] J. Howard and S. Ruder, “Universal Language Model Fine-tuning for Text Classification,” in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Melbourne, Australia, 2018, pp. 328–339, doi: 10.18653/v1/P18-1031.

[175] A. Safaya, M. Abdullatif, and D. Yuret, *KUISAIL at SemEval-2020 Task 12: BERT-CNN for Offensive Speech Identification in Social Media*. 2020.

[176] I. Zeroual, D. Goldhahn, T. Eckart, and A. Lakhouaja, “OSIAN: Open Source International Arabic News Corpus - Preparation and Integration into the CLARIN-infrastructure,” in *Proceedings of the Fourth Arabic Natural Language Processing Workshop*, Florence, Italy, Aug. 2019, pp. 175–182, doi: 10.18653/v1/W19-4619.

[177] J. Elman, “Finding structure in time,” *Cogn. Sci.*, vol. 14, no. 2, pp. 179–211, Jun. 1990, doi: 10.1016/0364-0213(90)90002-E.

[178] V. K. Ayyadevara, “Recurrent Neural Network,” in *Pro Machine Learning Algorithms : A Hands-On Approach to Implementing Algorithms in Python and R*, V. K. Ayyadevara, Ed. Berkeley, CA: Apress, 2018, pp. 217–257.

[179] S. MacAvaney, H.-R. Yao, E. Yang, K. Russell, N. Goharian, and O. Frieder, “Hate speech detection: Challenges and solutions,” *PLOS ONE*, vol. 14, no. 8, p. e0221152, Aug. 2019, doi: 10.1371/journal.pone.0221152.

[180] “ASA.” https://units.imamu.edu.sa/rcentres/en/asa/Pages/default.aspx.

[181] R. Polikar, “Ensemble based systems in decision making,” *IEEE Circuits Syst. Mag.*, vol. 6, no. 3, pp. 21–45, 2006, doi: 10.1109/MCAS.2006.1688199.

[182] D. M. W. Powers, “Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness & Correlation,” *J. Mach. Learn. Technol.*, vol. 2, no. 1, pp. 37–63, 2011.

APPENDIX

Code:

The code can be accessed through the link:

https://github.com/AsmaAlWazrah/ASA-Stacked-GRU

Publication:

Alhumoud, S., AlWazrah, A. (2020). Arabic Sentiment Analysis Using Recurrent Neural Networks A Review. Manuscript accepted for publication in Artificial Intelligence Review- Q1 Journal. (impact factor: 5.747 DOI: https://doi.org/10.1007/s10462-021-09989-9

AlWazrah, A., Alhumoud, S. (2021). Sentiment Analysis Using Stacked Gated Recurrent Unit for Arabic Tweets. Submitted to IEEE Access – Q1 Journal. (impact factor: 3.745).