POPULAR AND DEMOCRATIC REPUBLIC OF ALGERIA
Ministry of Higher Education and Scientific Research
Mohamed Khider University – BISKRA
Faculty of Exact Sciences, Natural Sciences and life
Computer Science department

Order N° : SIOD10/M2/2021

Dissertation

Presented to obtain the academic master's degree in

Computer science

Option : **SIOD**

Product rating based on online consumer reviews

By :
BENDJEDIDI OUSSAMA

Defended on ../../…., in front of the jury composed of:

Nom Prénom	grade	Président
Meadi nadjib	MCB	Rapporteur
Nom Prénom	grade	Examinateur

Academic year 2020-2021

Dedication

“

Every challenging work needs efforts as well as guidance of elders especially those who were very close to our heart.
My humble effort i dedicate it to my sweet and loving

Mother

Whose affection, love, encouragement and prays day and night make me able to get such success and honer,

Along with all hard working and respected

Teachers

”

- Oussama

Acknowledgement

First of all, I thank Allah the Almighty for giving me the courage and patience to complete this work.

I would like to especially thank my supervisor **Dr.MEADI Nadjib**, for his competent help, for his patience and encouragement. His critical eye was invaluable to me in structuring the work and in improving the quality of the different sections. I would like to thank everyone who contributed directly or indirectly to the achievement of this work. I also thank my friends and family who encouraged me throughout the time of my research. This thesis is heartily dedicated to my father who took the lead to heaven before the completion of this work.

May Allah bless you with prosperity, success and achieve every goal in life .

Abstract

Today E-commerce has become an important part day to day life. Right now many companies use product rating and consumer review to their websites so it can help the sellers to get feedback about their product and helps the customers to ensure the product of its good quality but product rating alone does not give the complete picture and the number of reviews is in the hundreds or thousands.
This work aims to analyse Arabic online reviews to rate a product using sentiment analysis and deep learning specifically the BERT model.

Keywords : E-commerce, Online reviews, Product rating , Arabic, Sentiment analysis, Deep learning, BERT.

Résumé

Aujourd'hui, le E-commerce est devenu une partie importante de la vie quotidienne. À l'heure actuelle, de nombreuses entreprises utilisent l'évaluation des produits et les avis des consommateurs sur leurs sites Web afin que cela puisse aider les vendeurs à obtenir des commentaires sur leur produit et aider les clients à garantir la bonne qualité du produit, mais l'évaluation du produit à elle seule ne donne pas une image complète et le nombre des avis se comptent par centaines ou par milliers.
Ce travail vise à analyser les critiques en ligne arabes pour évaluer un produit à l'aide de l'analyse des sentiments et de l'apprentissage en profondeur, en particulier du modèle BERT.

Mots clés : E-commerce, Avis en ligne, Évaluation du produit, arabe, Analyse des sentiments, L'apprentissage en profondeur, BERT.

ملخص

أصبحت التجارة الإلكترونية اليوم جزءًا مهمًا من الحياة اليومية. في الوقت الحالي ، تستخدم العديد من الشركات تصنيف المنتج ومراجعة المستهلكين لمواقعهم على الويب حتى يتمكنوا من مساعدة البائعين في الحصول على تقيم حول منتجاتهم كما يساعد العملاء على ضمان جودة المنتج ولكن تصنيف المنتج وحده لا يعطي الصورة الكاملة و عدد المراجعات غالبا ما تكون بالمئات أو بالآلاف.

يهدف هذا العمل إلى تحليل التقييمات العربية عبر الإنترنت لتقييم منتج باستخدام تحليل المشاعر والتعلم العميق على وجه التحديد نموذج BERT.

كلمات مفتاحية : التجارة الإلكترونية ، مراجعات عبر الإنترنت ، تصنيف المنتج ، العربية ، تحليل المشاعر ، التعلم عميق ، BERT

Contents

List of Figures

List of Tables

List of abbreviations and acronyms

NLP *Natural Language Processing*

SA *Sentiment Analysis*

WA *Word Embeddings*

BOW *Bag Of Words*

TF *Term Frequency*

IDF *Inverse Document Frequency*

CBOW *Common Bag Of Words*

GloVe *Global Vectors for Word Representation*

ELMo *Embeddings from Language Model*

CNN *Convolutional Neural Network*

biLM *Bidirectional LSTMs*

LSTM *Long Short-Term Memory*

GPT-3 *Generative Pre-trained Transformer 3*

RNN *Recurrent Neural Network*

BERT *Bidirectional Encoder Representations from Transformers*

MLM *Masked-Language Modeling*

NLI *Next Sentence Prediction*

ALBERT *A Lite BERT For Self-Supervised Learning of Language Representations*

RoBERTa *A Robustly Optimized BERT Pretraining Approach*

WOM *Word-of-mouth*

GPU *Graphics Processing Unit*

General Introduction

Context

People are increasingly turning to the internet to purchase goods and services. On the one hand, this may be explained by the proliferation of commercial websites. On the other hand, it can be explained by customers acceptance of online transactions (Stein et al. 2004). Customer feedback is extremely important in everyday life. When we have to make a decision, we take into account the views of others. Many online users nowadays express their thoughts on a variety of items via blogs, review sites, and social networking sites. Businesses and corporations are constantly interested in hearing from consumers or individuals about their products, support and service. As a result, it's critical to develop ways for automatically classifying them.

One of these ways is sentiment analysis, also known as Opinion mining, which is the process of extracting and analyzing evaluations, views, feelings, and opinions from text, large data, and voice using a variety of approaches.

Problematic

Machine learning-based sentiment analysis uses a database of sentiment-based terms that includes both positive and negative phrases. The words in the user comments section are compared to the words in the database, and a score is assigned. The system determines if the product is good, poor, or worst by comparing it to terms in the database.
Opinion mining has sparked a lot of academic interest in the previous decade, although it's largely been done in English. Opinion mining in Arabic is behind, due to difficulties

processing the morphologically complex Arabic natural language, as well as a lack of tools and resources for extracting Arabic feelings from text

Goals

We aim to tackle the problem of product rating based on Arabic online consumer reviews based on Deep learning.
text classification is one of the fundamental problems of sentiment analysis and we will use the BERT model to solve this problem. We will explore this new model and evaluate our work with previous works with other languages.

thesis organization

This thesis is organized into four chapters:

The first chapter **Deep learning** , gives a brief look at deep learning and its architecture that is used in natural language processing.

The second chapter **Sentiment Analysis and Word embeddings** . The first part of this chapter discusses Sentiment analysis, its levels and techniques. The second part of this chapter provides an overview of word embeddings and their models especially the BERT model. And The final part defines what is online reviews and product rating .

The third chapter **System Design** This chapter describes Our system in detail with all its phases.

The fourth chapter **Implementation and results** defines the tool used in this project, and how to implement the system. Evaluation and results are also presented in this chapter.

lastly **Conclusion and perspectives** we summarize and evaluate our ideas and outcomes, offering suggestions for improvements and new perspectives.

Chapter 1

Deep learning

1.1 introduction

Machine learning and deep learning have opened up a whole new field of research for us to investigate. Deep learning has been studied by researchers in recent years as a result of more available technology and GPU advancements. Deep learning approaches attempt to mimic the way the human brain functions, which is made up of hidden layers made up of multiple neurons. Deep learning can provide good results in a variety of applications. This chapter provides a detailed explanation for deep learning and particularly the architecture used in NLP.

1.2 Definition

Deep Learning is a type of machine learning that excels at working with unstructured data. Current machine learning techniques are outperformed by deep learning approaches. It allows computer models to learn characteristics from data at several levels in a step-by-step manner. Deep learning grew in popularity as the amount of data accessible grew, as did the progress of technology (Mathew et al. 2021). Also known as deep neural learning or deep neural network.

1.3 Artificial Neural Networks

ANN is inspired by the human brain (Figure 1.1), which may be utilized for machine learning and artificial intelligence. Various computer-based problems can be solved via these networks. Various computer-based problems can be solved via these networks. The artificial neural network (ANN) is based on the structure of the biological brain to some extent. It is made up of an abstracted model of linked neurons whose unique layout and connectivity may be utilized to address computer-based application issues in disciplines like statistics, technology, and economics(Mijwil et al. 2019).

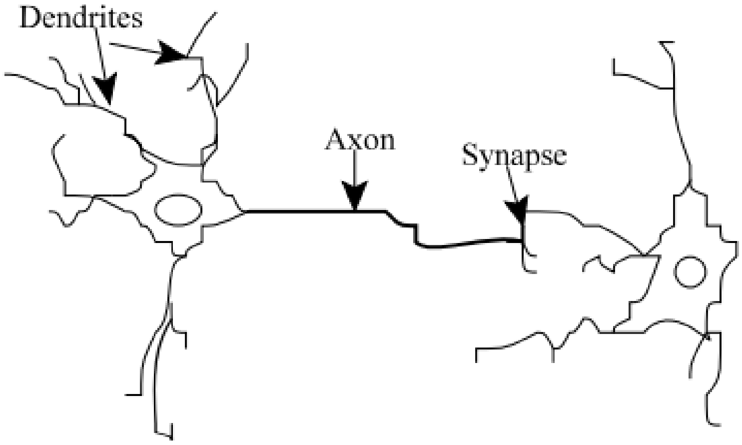


Figure 1.1: Natural neurons (Gershenson 2003).

When modeling artificial neurons (Figure 1.2), the complexity of real neurons is heavily abstracted. Inputs (such as synapses) are multiplied by weights (the intensity of the various signals), and the result is determined by a mathematical function, which defines the neuron's activity(Mathew et al. 2021).

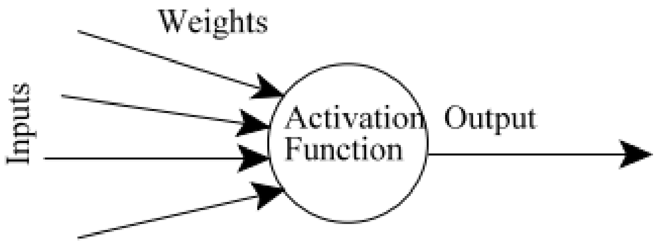


Figure 1.2: Artificial neuron (Gershenson 2003).

1.4 Applications of Deep Learning

Self-driving cars, Natural Language Processing, Google's Virtual Assistant, Visual Recognition, Fraud detection, healthcare, detecting developmental delays in children, adding sound to silent movies, automatic machine translation, text to image translation, image to image synthesis, automatic image recognition, and image colorization are just some of the applications for deep learning networks.

1.5 Deep learning architectures

To begin, we must state that deep learning architecture is made up of many topologies of deep/neural networks. In general, neural networks consist of many layers that process and integrate data: an input layer (raw data), hidden layers (which process and integrate input data), and an output layer (it produces the outcome: result, estimation, forecast, etc.). There are different sorts of neural networks when it comes to deep learning. These networks serve as the foundation for deep learning architectures. Here the most frequent deep learning architectures :

- Cnn : Convolutional Neural Networks
- RNN : Recurrent Neural Networks
- LSTM : Long Short-Term Memory
- GAN : Generative Adversarial Networks
- RBFN : Radial Basis Function Networks

1.6 Recurrent Deep Learning

As one of the latest and effective Natural language processing methods. Here are some definitions of Recurrent Neural Networks (RNN) and Long Short-Term Memory (LSTM)

1.6.1 Recurrent Neural Networks

Humans don't start thinking all over again every second. You comprehend each word as you read this paragraph depending on your comprehension of prior words. You don't toss everything out and start again from the beginning. Your thoughts are persistent.
This is something that traditional neural networks can't achieve, and it appears to be a fundamental flaw.
Consider the following scenario: you wish to categorize the type of event that occurs at each moment in a movie. It's unclear how a typical neural network might utilize prior events in the movie to guide subsequent ones. This problem is addressed by recurrent

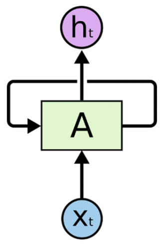


Figure 1.3: Recurrent Neural Networks have loops (F. Liu et al. 2019).

neural networks. It's a network with loops in it. allowing information to persist.
A piece of a neural network, in the (Figure 1.3) above, looks at some input and outputs a value. Information can be transmitted from one network step to the next via a loop. Recurrent neural networks appear mysterious because of these loops. However, if you think about it, they aren't all that different from a traditional neural network. A recurrent neural network is made up of several copies of the same network, each of which sends a message to its successor. Consider what happens if the loop is unrolled:(Figure 1.4)

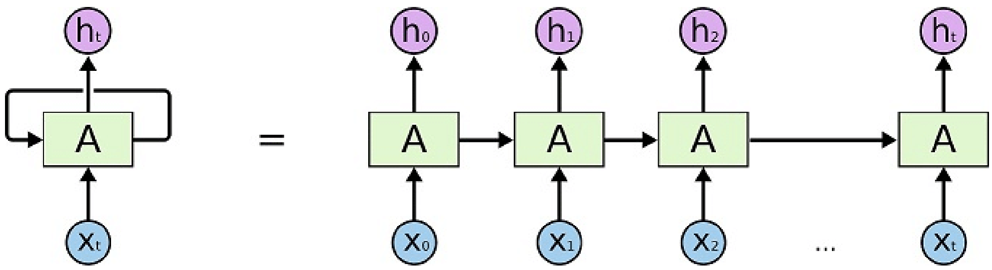


Figure 1.4: An unrolled recurrent neural network (F. Liu et al. 2019).

They are very remarkable. The usage of "LSTMs," a particularly specific type of recurrent neural network that performs far better than the regular version for many tasks, is critical to these accomplishments. Almost all of the most fascinating recurrent neural network achievements are achieved with them. We will explore them next.

The Problem of Long-Term Dependencies

One of the appeals of RNNs is the possibility of connecting earlier data to the current task, such as using prior video frames to inform comprehension of the current frame. RNNs would be immensely handy if they could accomplish this. Can they, however, do it? It is debatable.
Sometimes all we need to do is glance at recent data to complete the work at hand.

Consider a language model that tries to anticipate the next word based on the ones that came before it. We don't need much more information to guess the last word in ”the clouds are in the sky” — it's very evident that the following word is going to be sky. However, there are times when we require further background. Consider predicting the text's final word: ”I grew up in France... “I speak a fluent French speaker.” According to recent information, the following term is most likely the name of a language, but we need the context of France from further back to narrow down the language. It's entirely possible for the space between relevant data and the moment at which it's required to grow significantly (Staudemeyer et al. 2019).
Unfortunately, as the gap widens, RNNs lose their ability to learn to make the connection.

1.6.2 Long Short-Term Memory

Long Short Term Memory Networks (LSTMs) are a kind of RNN that can learn long-term dependencies. Hochreiter Schmidhuber were the ones who presented them 1997 (Hochreiter et al. 1997) LSTMs are specifically developed to prevent the problem of long-term dependency. They don't have to exercise hard to remember knowledge for long periods of time; it's like part of the routine to them.
All recurrent neural networks are made up of a series of repeated neural network modules. This repeating module in ordinary RNNs will have a relatively basic structure, such as a single tanh layer.

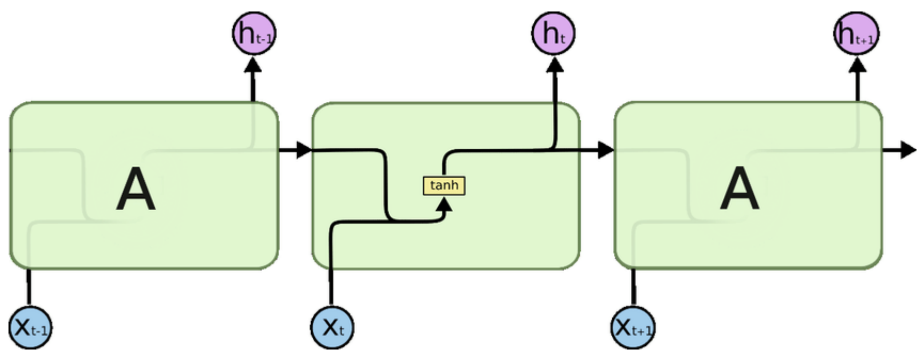


Figure 1.5: The repeating module in a standard RNN contains a single layer (F. Liu et al. 2019).

LSTMs have a chain-like structure as well, but the repeating module is different. Instead of a single neural network layer, there are four, each of which interacts in a special way.

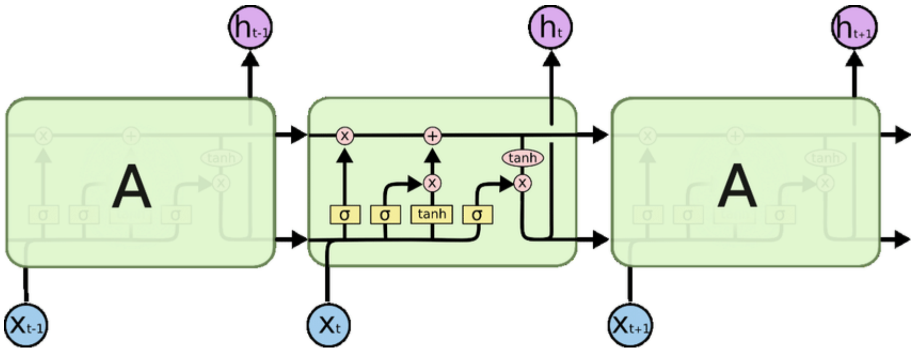


Figure 1.6: The repeating module in an LSTM contains four interacting layers (F. Liu et al. 2019).

In (Figure 1.6). Each line transports a full vector from one node's output to the inputs of others. The pink circles indicate pointwise operations, such as vector addition, while the yellow boxes denote learned neural network layers. Concatenation occurs when lines merge, while forking occurs when a line's content is replicated and the copies are sent to various locations.

The cell state, the horizontal line going across the top of the figure, is the key to LSTMs. The condition of the cell is similar to that of a conveyor belt. With only a few tiny linear interactions, it flows straight down the whole chain. It's incredibly easy for data to just travel over it unchanged.

The LSTM could delete or add information to the cell state, which is carefully controlled by structures called gates.

Gates are a way to selectively allow information to pass through. A sigmoid neural net layer plus a pointwise multiplication operation The sigmoid layer produces numbers between zero and one, indicating how much of each component should be allowed through. A value of zero indicates that "nothing should be let through," whereas a value of one indicates that "everything should be let through!". Three of these gates are present in an LSTM to protect and control the cell state.

1.7 Conclusion

We attempted to cover some principles and algorithms in the modern Deep Learning approach used in the field of sentiment analysis in this chapter. Because these are the approaches utilized in this work, we have concentrated our explanations on RNN and LSTM techniques.

The coming chapter will contain everything we need to know from sentiment analysis and word embeddings address our problem that we will solve .

Chapter 2

Sentiment Analysis and Word embeddings

2.1 Introduction

Sharing thoughts and experiences on the quality and usefulness of goods and services has long been beneficial to consumers. Ecommerce and online platform ratings and reviews have become increasingly available and important as the internet has grown in popularity. Millions of people create and rely on user-generated ratings and reviews to help them make purchase decisions. Businesses have recognized their ability to influence customer purchasing behavior and have built systems to collect, organize, and display online consumer ratings and reviews, which have evolved into major advertising tools using sentiment analysis and natural language processing
However, in terms of the Arabic language, this field is still in its infancy.
This chapter introduces the sentiment analysis, its techniques and applications as well as word embedding and its models after that We will discuss our product rating for online reviews with the Arabic language in detail and its challenges

2.2 Sentiment analysis

Sentiment analysis is one of the popular tasks in natural language processing. Here are some basic explanations on sentiment analysis.

2.2.1 Definition

Sentiment Analysis (SA) is a subject of research in the field of text mining that is still ongoing. SA is the computer handling of text's views, feelings, and subjectivity, also known as opinion mining or contextual mining. It is used in Natural Language Processing (NLP), computational linguistics, and text analysis to detect, systematically extract, and quantify subjective data. (Luhach et al. 2019).Sentiment is defined by three terms in sentiment analysis. These are the following: the object about which an opinion is expressed, the features of that item, and the opinion holder who expresses his opinion about the object. Sentiment analysis addresses a number of issues, including object recognition, feature extraction, and finding the opinion's direction.(v. k. v. 2019).

2.2.2 Sentiment Analysis Levels

Sentiment Analysis performs the classification task in 3 steps :

Document level

This is the most basic classification system. The entire document of opinionated text is regarded as a fundamental element of data. It is believed that the document only has an opinion on a single item (film, book or hotel). This method is ineffective when a document has several perspectives about the same object, like in forums and blogs. The entire document is categorized as favorable or bad. Irrelevant sentences need to be eliminated before processing(Kolkur et al. 2015).

Sentence level

Sentence level sentiment analysis is the most fine-grained analysis of the document. In this, polarity is calculated for each sentence as each sentence is considered as a separate unit and each sentence can have a different opinion. Sentence level sentiment analysis has two tasks:

- Subjectivity Classification: Depending on the context, a sentence might be subjective or objective. The facts are contained in the objective sentence. It expresses no opinion or judgment on the item or thing, whereas a subjective phrase expresses opinions. Objective sentences should be taken away since they have no bearing on the polarity of the evaluation. (Kolkur et al. 2015) .
- Sentiment Classification: A Sentence can be classified as positive, negative or neutral depending upon the opinion words present in it.

Feature level or Aspect level

Sentiment Analysis requires feature engineering, which is a very fundamental and important activity. The first step in Feature Level sentiment analysis is to recognize the text as a product feature. Battery life, for example, is quite lengthy. Battery is a product

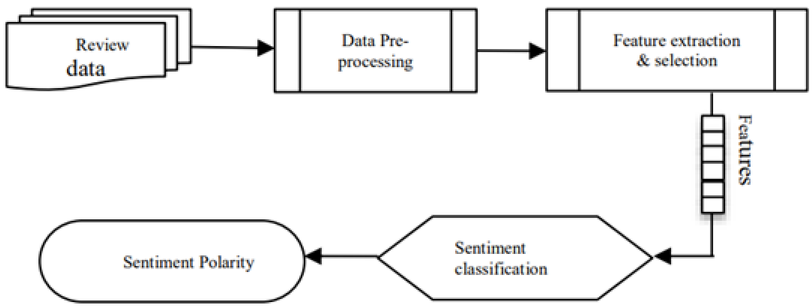


Figure 2.1: Sentiment Analysis Process (v. k. v. 2019).

feature (noun) in this review, while 'very long lasting' is an opinion term (adjective). The basic steps for feature-based sentiment analysis (Figure 2.1) are (Tribhuvan et al. 2014) :

1. Preparing Review Database
2. Part-of-Speech Tagging
3. Feature Extraction
4. Opinion Word Extraction
5. Opinion Word Polarity Identification
6. Opinion Sentence Polarity Identification

2.2.3 Sentiment Analysis Techniques

sentiment analysis contains three different techniques (Figure 2.2)

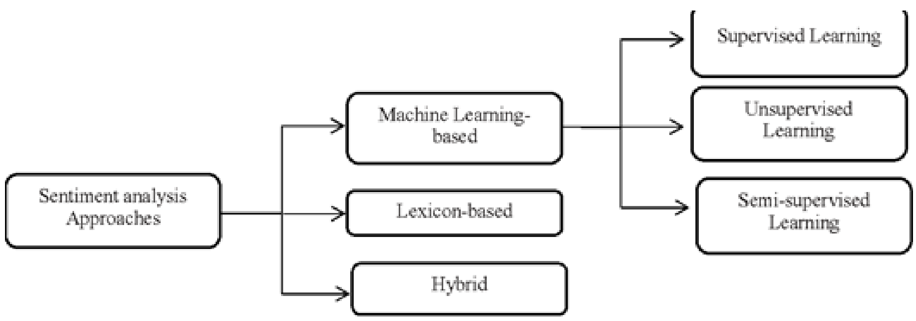


Figure 2.2: Sentiment Analysis Techniques (Madhoushi et al. 2015).

Lexical analysis

The usage of a dictionary made up of pre-tagged lexicons controls this method. The Tokenizer converts the supplied text into tokens. Every new token discovered is then compared to the dictionary's vocabulary. If there is a positive match, the score is added to the input text's overall pool of scores. Otherwise, the score is decremented or the word is tagged as negative. Though this technique appears to be amateur in nature, its variants have proved to be worthy (Thakkar et al. 2015), (Figure 2.3) shows the working of a lexical technique.

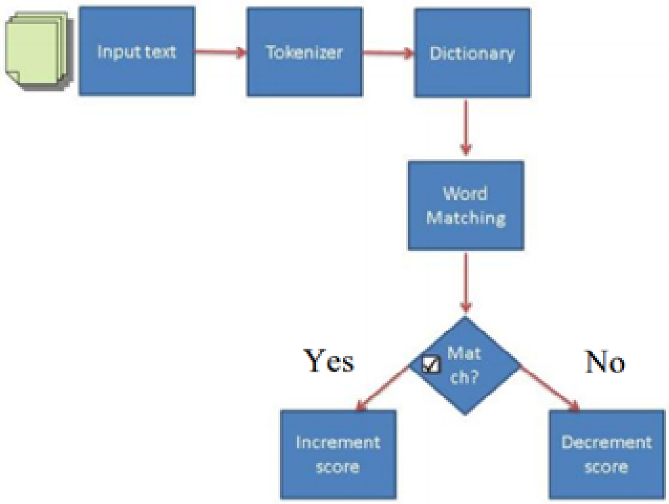


Figure 2.3: Working of a Lexical Technique (Thakkar et al. 2015).

Machine learning based analysis

Machine learning is one of the most popular approaches that researchers are interested in because of its versatility and accuracy. The supervised learning variations of this approach are primarily used in sentiment analysis. Data gathering, pre-processing, training data, classification, and displaying results are the three steps. A set of labeled corpora is included in the training data. A sequence of feature vectors from the preceding data is provided to the Classifier. The training data set is used to develop a model, which is then applied to new/unseen text for categorization (Thakkar et al. 2015). The machine learning technique faces challenges in: designing a classifier, availability of training data, correct interpretation of an unforeseen phrase. It overcomes the limitation of the lexical approach of performance degradation and it works well even when the dictionary size grows exponentially (Thakkar et al. 2015).

Hybrid analysis

s method combines the previous two methods, and there are three ways to go about it. The first step is to build the corpus using linguistic tools and then categorize the texts using a supervised learning method. The second way is to use machine learning to establish the body of opinion necessary for the lexicon-based approach. The third way is the combination of the two previous approaches and the joint of their results (Poirier et al. 2009).

2.2.4 Applications of sentiment analysis

Sentiment analysis has various applications going from identifying customer opinion towards products and services, to voters' reaction to political adverts. Other application areas in which sentiment analysis can be very useful are:

- Business Intelligence :
  Sentiment analysis plays a very important role in many business intelligence applications such as credit rating or company reputation. It is useful to classify each opinion according to the aspect of the business or transaction describes: e.g., product quality, ordering, or integrity. Sentiment analysis helps to evaluate the limitations of particular products and then exploit this information to improve the products or services. It also helps enterprises to understand their customers as well as to plan future products and services (Funk et al. 2008).

- Recommendation system:
  Sentiment Analysis helps in knowing individual's review of a product or issue. On based of that customers' take decision and further predict what impact that topic has on other domains. Sentiment Analysis helps in knowing customer's review on a product or service. On based of customers' decision, predict what effect that topic has on other domains and predict what impact demonetization has on economic

and social fronts (Aggarwal et al. 2018).

- Summarization of Reviews :
  Sentiment analysis can be used to extract opinions about a specific entity. It will offer a general assessment of a topic. A customer may not be able to make an educated judgment about a product by reading all evaluations, and the manufacturer may not be able to keep the pace of consumer opinion. (Aggarwal et al. 2018).

- Government Intelligence :
  In the field of government intelligence, sentiment analysis allows for the extraction of public opinion on government policies and actions in order to predict probable public reaction to policy execution.(Asghar et al. 2014).

2.3 Word embeddings

Before we can use words in sentiment analysis, we need to convert them into numbers that have meaning and understandable by the machine. Word embeddings address these issues. so what is word embeddings?

2.3.1 Definition

Generally speaking, the objective of word embedding is to map the words in unlabeled text data to a low-dimensional, continuously valued space in order to capture the inherent semantic and grammatical information. (Li et al. 2017). Word embeddings, to be more exact, are unsupervised word representation vectors whose relative similarity correlates with semantic similarity. They're also known as distributed representations or distributional semantic models in computational linguistics.(Goldberg 2017).

2.3.2 Word embeddings models

Word embeddings models can be divided into three main categories:

Count-based methods

The count-based technique is the first type of word embedding. The count-based technique evaluates a target word based on the nature of words that appear alongside it in a variety of situations. This is determined using a co-occurrence estimate method. The meaning of a word is conceived in this case by the words that appear with it in a variety of situations.

- Bag of words :

  The bag-of-words method is a frequent feature of sentence and document extraction procedures (BOW). We look at the histogram of the words inside the text in this method, using each word count as a feature (Goldberg 2017). It involves two things :

  - A vocabulary of known words.
  - A measure of the presence of known words.

  Let's understand this with an example. Suppose we wanted to vectorize the following:
  the cat sat
  the cat sat in the hat
  the cat with the hat
  Step 1: Determine the Vocabulary of known words
  We first define our vocabulary, which is the set of all words found in our document set. The only words that
  are found in the 3 documents above are:
  the, cat, sat, in, the, hat, and with.
  Step 2: measure of the presence of known words
  To vectorize our documents, all we have to do is count how many times each word appears:

  Now we have length-6 vectors for each document!
  the cat sat: [1, 1, 1, 0, 0, 0]
  the cat sat in the hat: [2, 1, 1, 1, 1, 0]
  the cat with the hat: [2, 1, 0, 0, 1, 1]
  Notice that we lose contextual information, e.g where in the document the word

Document	the	cat	sat	in	hat	with
the cat sat	1	1	1	0	0	0
the cat sat in the hat	2	1	1	1	1	0
the cat with the hat	2	1	0	0	1	1

Figure 2.4: BOW representation (Dandale 2021).

appeared when we use BOW. It's like a literal bag-of-words: it only tells you what words occur in the document, not where they occurred.

- TF-IDF :

  (TF) Term Frequency and (IDF) InverseDocument Frequency are two separate terms combined into one. IDF is used to determine the significance of a term in a text document collection, whereas TF is used to determine how many times a word appears in a document. (Aizawa 2003)

Variants of term frequency (tf) weight

weighting scheme	tf weight
binary	$0, 1$
raw count	$f_{t,d}$
term frequency	$f_{t,d} \Big/ \sum_{t' \in d} f_{t',d}$
log normalization	$\log(1 + f_{t,d})$
double normalization 0.5	$0.5 + 0.5 \cdot \frac{f_{t,d}}{\max_{\{t' \in d\}} f_{t',d}}$
double normalization K	$K + (1 - K) \frac{f_{t,d}}{\max_{\{t' \in d\}} f_{t',d}}$

Figure 2.5: TF various equations (Gorakala 2017).

where ($f_{t,d}$) is the raw count of a term in a document, i.e., the number of times that term t occurs in document d.

Variants of inverse document frequency (idf) weight

weighting scheme	idf weight ($n_t = \|\{d \in D : t \in d\}\|$)
unary	1
inverse document frequency	$\log \frac{N}{n_t} = -\log \frac{n_t}{N}$
inverse document frequency smooth	$\log \left(\frac{N}{1 + n_t} \right) + 1$
inverse document frequency max	$\log \left(\frac{\max_{\{t' \in d\}} n_{t'}}{1 + n_t} \right)$
probabilistic inverse document frequency	$\log \frac{N - n_t}{n_t}$

Figure 2.6: IDF various equations (Gorakala 2017).

Then tf–idf is calculated as :

$$TF - IDF(t, d) = TF(t, d) * IDF(t, d) \tag{2.1}$$

- Co-occurrence Matrix :

  Co-occurrence matrix M is a matrix representation of words' contexts. Each row of the matrix represents the context distribution of the target word t. Each column describes the distribution of context word c. Each term represents the co-occurrence relation between t and c. The context of t is a symmetrical window W centered on t. Consider the sentence "There is a student reading in the classroom". Let the target word be "student" and the size of the window be 2; then, the context is "is a (student) reading in", and the corresponding co-occurrence vector is as (Figure 2.7)

There	is	a	student	reading	in	the	classroom
0	1	1	0	1	1	0	0

Figure 2.7: The Unweighted Cooccurrence Vector of "Student" (Gao et al. 2019).

If we assume that words closer to "student" should have greater weights, the cooccurrence vector can be rewritten as (Figure 2.8)

There	is	a	student	reading	in	the	classroom
0	1	2	0	2	1	0	0

Figure 2.8: The Weighted Cooccurrence Vector of “Student” (Gao et al. 2019).

Predictive methods

- Word2vec :

  Word2vec is a combination of models used to represent distributed representations of words in a corpus C. Word2Vec (W2V) is an algorithm that accepts text corpus as an input and outputs a vector representation for each word, as shown in (Figure 2.9) below:

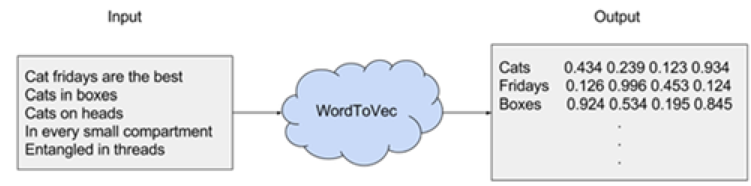


Figure 2.9: Representations of words with Word2vec (Tulasiram 2021).

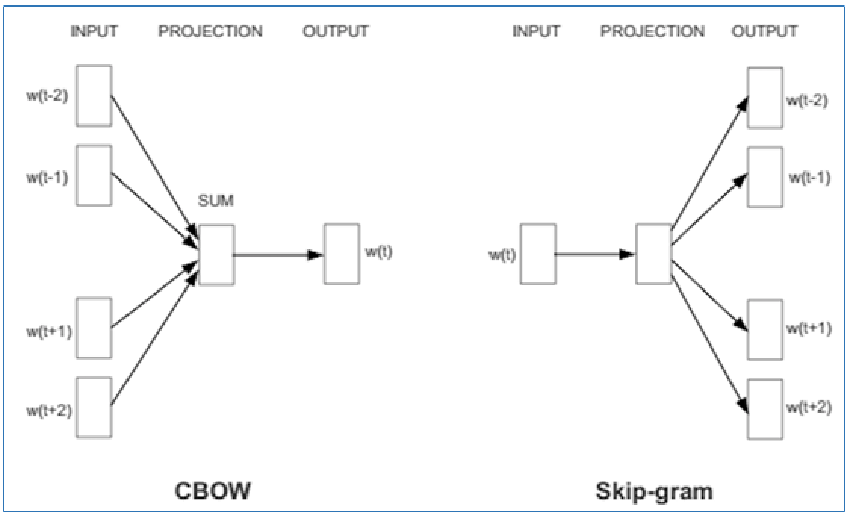


Figure 2.10: CBOW and Skip-gram architectures (Popov 2018).

The technique Word2Vec is used to create such an embedding. There are two ways to acquire it:

– Common Bag Of Words (CBOW) Model

– Skip-Gram Model

Both are architectures to learn the underlying word representations for each word by using neural networks. In the CBOW model, the distributed representations of context (or surrounding words) are combined to predict the word in the middle. While in the Skip-gram model, the distributed representation of the input word is used to predict the context.

- GloVe :

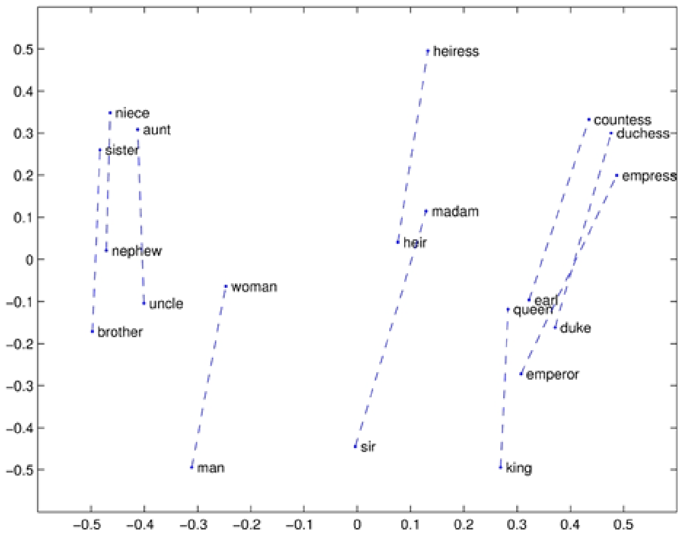


Figure 2.11: Global Vectors for Word Representation (*GloVe: Global Vectors for Word Representation* 2021).

Stands for global vectors for word representation, it is another well-known model that learns vectors or words from their co-occurrence information GloVe uses a weighted least squares objective J that minimizes the difference between the dot product of the vectors of two words and the logarithm of their number of co-occurrences:

$$J = \sum_{(i,j=1)}^{V} f(X_{ij})(w_i^T w \enspace _j + b_i + b \enspace _j - log \; X_{ij})^2 \tag{2.2}$$

where w_i and b_i are the word vector and bias respectively of word $_i$, w_i and b_j are the context word vector and bias respectively of word k, X_{ij} is the number of times word i occurs in the context of word $_j$, and f is a weighting function that assigns lower weights to rare and frequent co-occurrences (Pennington et al. 2014).

- Fastext :

  Embedding of rarely used words can sometimes be poorly estimated. Therefore several methods have been proposed to remedy this issue, including the FastText method. FastText uses the subword information explicitly so embedding for rare words can still be represented well. It is still based on the skip-gram model, where each word is represented as a bag of character n-grams or subword units. A vector representation is associated with each of character n-grams, and the average of these vectors gives the final representation of the word. This model improves the performance on syntactic tasks significantly but not much in semantic questions (Wang et al. 2019).

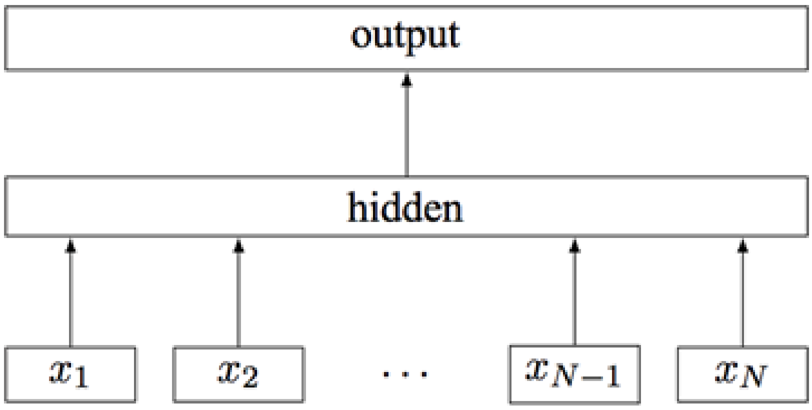


Figure 2.12: Model architecture of Fastext for sentence with n-gram features (Herwanto et al. 2019).

Deep Contextualized Models

To represent complex characteristics of words and word usage across different linguistic contexts effectively, a new model for deep contextualized word representation was introduced First, an Embeddings from Language Models (ELMo) representation is generated with a function that takes an entire sentence as the input. the ELMo representation as it is effective in incorporating the sentence information. By following ELMo, a series of pre-trained neural network models for language tasks are proposed such as BERT Their effectiveness is proved in lots of language tasks (Wang et al. 2019) .

ELMo

Typical word embeddings models or representations, such as word2vec, GloVe, or FastText, are fast to train and have been pre-trained for a number of different languages. They do not capture the context, though, so each word is always given the same vector, regardless of its context or meaning. This is especially problematic for polysemous words. ELMo (Embeddings from Language Models) embedding is one of the state-of-the-art pre-trained transfer learning models, that remedies the problem and introduces a contextual component(Ulčar et al. 2019).

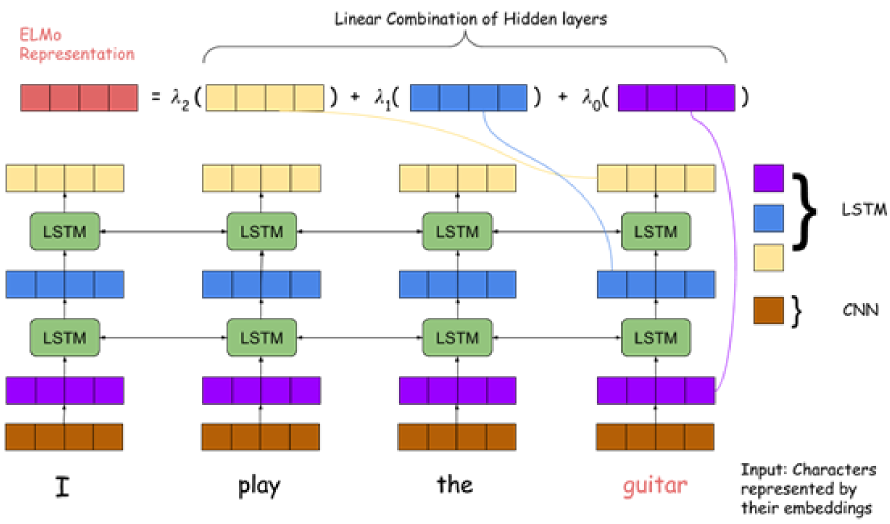


Figure 2.13: ELMo architecture (Ghosh 2020).

ELMo model's architecture consists of three neural network layers. The output of the model after each layer gives one set of embeddings, altogether three sets. The first layer is a CNN layer, which operates on a character level. It is context-independent, so each word always gets the same embedding, regardless of its context. It is followed by two biLM layers. A biLM layer consists of two concatenated LSTMs. In the first LSTM, we try to predict the word, based on the given past words, where each word is represented by the embeddings from the CNN layer. In the second LSTM, we try to predict the preceding word, based on the given following words. It is equivalent to the first LSTM, just reading the text in reverse. In NLP tasks, any set of these embeddings may be used; however, a weighted average is usually used. The weights of the average are learned during the training of the model for the specific task. Additionally, an entire ELMo model can be fine-tuned on a specific end task. Although ELMo is trained on character level and is able to handle out-of-vocabulary words, a vocabulary file containing the most common tokens is used for efficiency during training and embedding generation. The original ELMo model was trained on a one billion word large English corpus, with a given vocabulary file of about 800,000 words. Later, ELMo models for other languages were trained as well but limited to larger languages with many resources, like German and Japanese(Ulčar et al. 2019).

- Bidirectional Language Model

  Given a sequence of n tokens,$(x_1, x_2, ..., x_n)$, a forward language model computes the probability of the sequence by modeling the probability of token t given the history$(x_1, x_2, ..., x_{k-1})$:

  - In the forward pass, the history contains words before the target token :

$$p(x_1, ..., x_n) = \prod_{i=1}^{n} p(x_i|x_1, ..., x_{i-1}) \tag{2.3}$$

  - In the backward pass, the history contains words after the target token :

$$p(x_1, ..., x_n) = \prod_{i=1}^{n} p(x_i|x_{i+1}, ..., x_n) \tag{2.4}$$

  A biLM combines both a forward and backward LM, and train the model to maximize the probability of likelihood

- ELMo Representation

  each token x , an L-layer biLM computes a set of 2L + 1 representations

$$R_i = (h_{i,l}|l = 0, ..., L) \tag{2.5}$$

$$h_{il} = [h_i^{\rightarrow}{}_l = h_i^{\leftarrow}{}_l] \tag{2.6}$$

  Outputs from both LSTM are concatenated together to form Hi,l

GPT-3

Generative Pre-trained Transformer 3 (GPT-3) is an open-source artificial intelligence created by OpenAI in February 2020. The GPT architecture implements a deep neural network, specifically a transformer model, lace of the previous recurrence- and convolution-based architectures(Vaswani et al. 2017) . Attention mechanisms allow the model to selectively focus on segments of input text. This model allows for greatly increased parallelization and outperforms previous benchmarks for RNN/CNN/LSTM-based models.(Radford et al. 2018). GPT-3's full version has a capacity of 175 billion machine learning parameters, which makes the Integration a challenge. At present, it has been used by some users who use open AI API, if the model needs to be integrated into mainstream applications, then developing the necessary infrastructure to obtain data and models will be a difficult task.

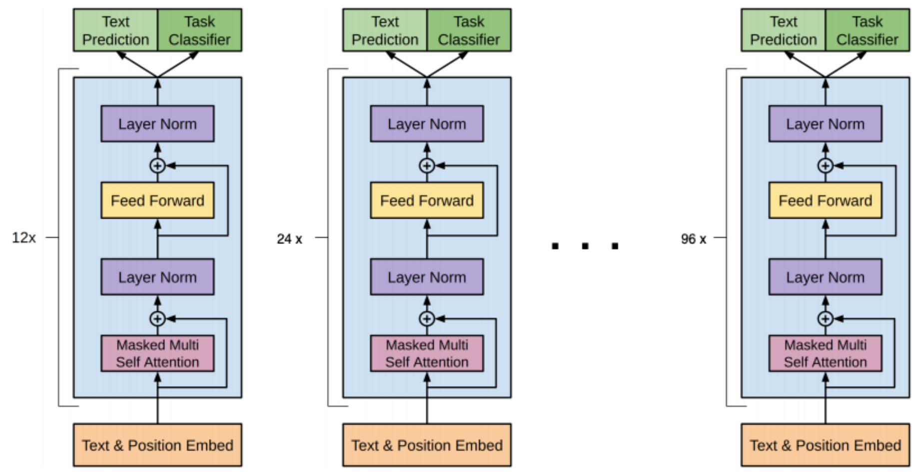


Figure 2.14: GPT-3 architecture (dzlab 2021).

2.3.3 BERT

Researchers at Google AI Language open-sourced presented a new Natural Language Processing (NLP) approach called BERT towards the end of 2018, a huge breakthrough that caught the Deep Learning world by storm due to its amazing performance. Since BERT is likely to stay around for quite some time, We'll learn about it in this section.

Definition

stands for Bidirectional Encoder Representations from Transformers. It is designed to pretrain deep bidirectional representations from an unlabeled text by jointly conditioning on both left and right context in all layers. As a result, the pre-trained BERT model can be finetuned with just one additional output layer to create state-of-the-art models for a wide range of tasks, such as question answering and language inference, without substantial task-specific architecture modifications (Devlin et al. 2018).

Architecture

BERT is a multi-layer Transformer encoder (Vaswani et al. 2017) with bi-directionality. A Transformer's architecture is purely based on self-attention, abandoning the RNN and CNN architectures commonly used for sequence modelling. The goal of self-attention is

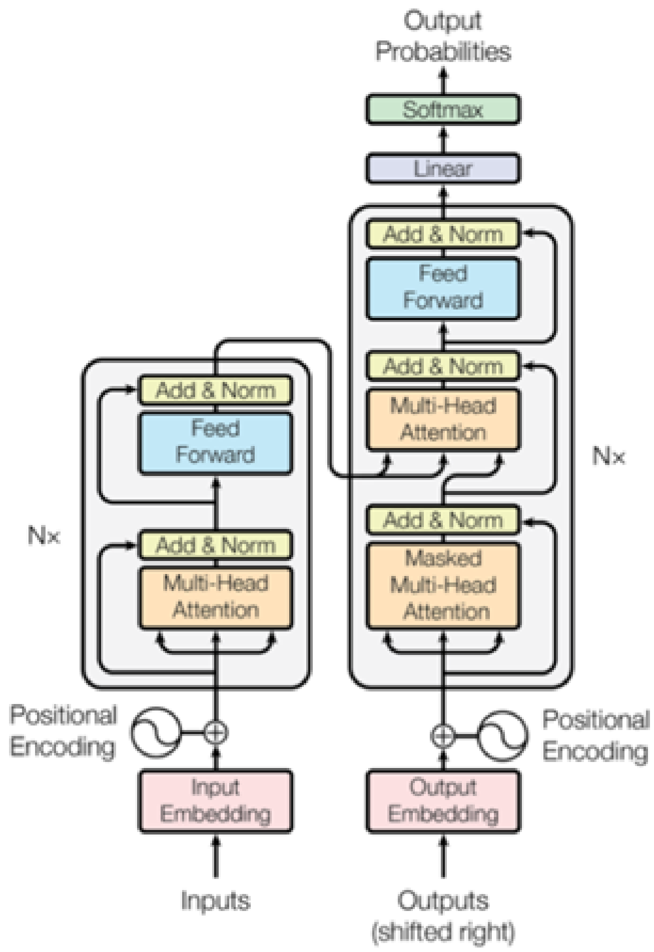


Figure 2.15: Transformer model architecture (Vaswani et al. 2018).

to capture the representation of each sequence by relating different positions of the sequence. Using this foundation, BERT breaks the familiar left-to-right indoctrination that is inherent in earlier text analysis and representation models. The original Transformer architecture, as depicted in (Shaw et al. 2018) , is shown below (Figure 2.15). Note that BERT uses only the encoder portion (left side) of this Transformer.

Input Representation

BERT represents a given input token using a combination of embeddings that indicate the corresponding token, segment, and position. Specifically, WordPiece embeddings (Wu et al. 2016) with a token vocabulary of 30,000 are used. . The first token of every sequence is always a special classification token ([CLS]). The final hidden state corresponding to this token is used as the aggregate sequence representation for classification tasks. Sentence pairs are packed together into a single sequence. We differentiate the sentences in two ways. First, we separate them with a special token ([SEP]). Second, we add a learned embedding to every token indicating whether it belongs to sentence A or sentence B. We denote input embedding as E, the final hidden vector of the special [CLS] token as C R H, and the final hidden vector for the i th input token as $Ti \in RH$. For a given token, its input representation is constructed by summing the corresponding token, segment, and position embeddings (Devlin et al. 2018) . A visualization of this construction can be seen in (Figure 2.16)

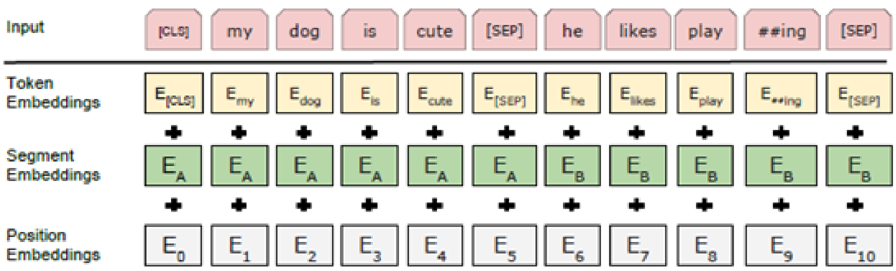


Figure 2.16: BERT input representation (Paul et al. 2020).

Pre-training BERT

Pre-training BERT uses two unsupervised tasks. This step is presented in the left part of (Figure 2.17) .

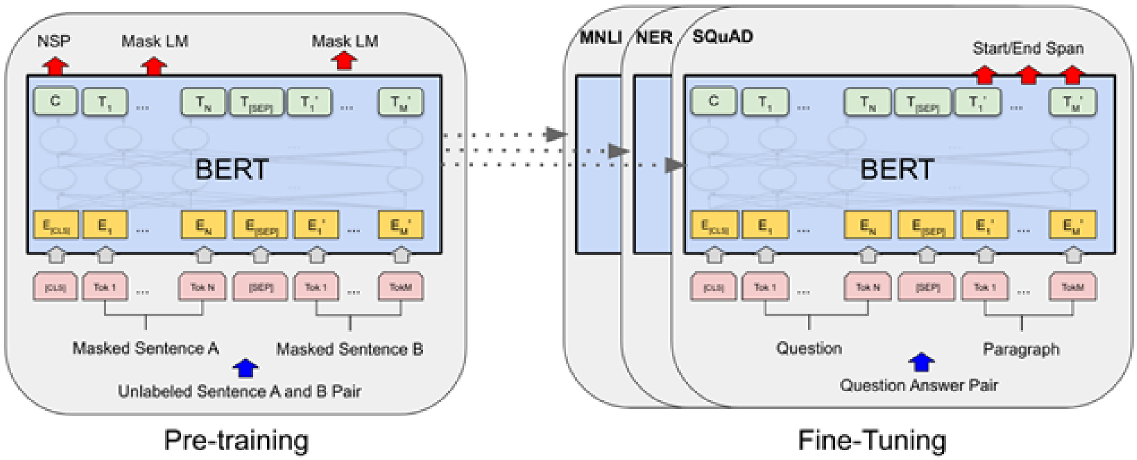


Figure 2.17: pre-training and fine-tuning procedures for BERT (Devlin et al. 2018).

- Task 1: Masked LM :
  - Intuitively, it is reasonable to believe that a deep bidirectional model is strictly more powerful than either a left-to-right model or the shallow concatenation of a left-to-right and a right-to-left model. Unfortunately, standard conditional language models can only be trained left-to-right or right-to-left, since bidirectional conditioning would allow each word to indirectly "see itself", and the model could trivially predict the target word in a multi-layered context
  - In order to train a deep bidirectional representation, we simply mask some percentage of the input tokens at random and then predict those masked tokens. We refer to this procedure as a "masked LM" (MLM). In this case, the final hidden vectors corresponding to the mask tokens are fed into an output softmax over the vocabulary, as in a standard LM. In all of our experiments, we mask 15% of all WordPiece tokens in each sequence at random. In contrast to denoising auto-encoders (Vincent et al. 2008), we only predict the masked words rather than reconstructing the entire input.
  - Although this allows us to obtain a bidirectional pre-trained model, a downside is that we are creating a mismatch between pre-training and fine-tuning, since the [MASK] token does not appear during fine-tuning. To mitigate this, we do not always replace "masked" words with the actual [MASK] token. The training data generator chooses 15% of the token positions at random for prediction. If the i-th token is chosen, we replace the i-th token with (1)

the [MASK] token 80% of the time (2) a random token 10% of the time (3) the unchanged i-th token 10% of the time. Then, Ti will be used to predict the original token with cross-entropy loss (Devlin et al. 2018).

- Task 2: Next Sentence Prediction (NSP) :
  Many important downstream tasks such as Question Answering (QA) and Natural Language Inference (NLI) are based on understanding the relationship between two sentences, which is not directly captured by language modeling. In order to train a model that understands sentence relationships, we pre-train for a binarized next sentence prediction task that can be trivially generated from any monolingual corpus. Specifically, when choosing the sentences A and B for each pretraining example, 50% of the time B is the actual next sentence that follows A (labeled as IsNext), and 50% of the time it is a random sentence from the corpus (labeled as NotNext). As we show in (Figure 2.17). C is used for next sentence prediction (NSP), only sentence embeddings are transferred to downstream tasks, where BERT transfers all parameters to initialize end-task model parameters (Devlin et al. 2018).

Fine-tuning BERT

Fine-tuning is straightforward since the self attention mechanism in the Transformer allows BERT to model many downstream tasks— whether they involve single text or text pairs—by swapping out the appropriate inputs and outputs. For applications involving text pairs, a common pattern is to independently encode text pairs before applying bidirectional cross attention. BERT instead uses the self-attention mechanism to unify these two stages, as encoding a concatenated text pair with self-attention effectively includes bidirectional cross attention between two sentences. For each task, we simply plug in the task-specific inputs and outputs into BERT and finetune all the parameters end-to-end. At the input, sentence A and sentence B from pre-training are analogous to (1) sentence pairs in paraphrasing, (2) hypothesis-premise pairs in entailment, (3) question-passage pairs in question answering, and (4) a degenerate text-pair in text classification or sequence tagging. At the output, the token representations are fed into an output layer for token level tasks, such as sequence tagging or question answering, and the [CLS] representation is fed into an output layer for classification, such as entailment or sentiment analysis (Devlin et al. 2018).

BERT Models

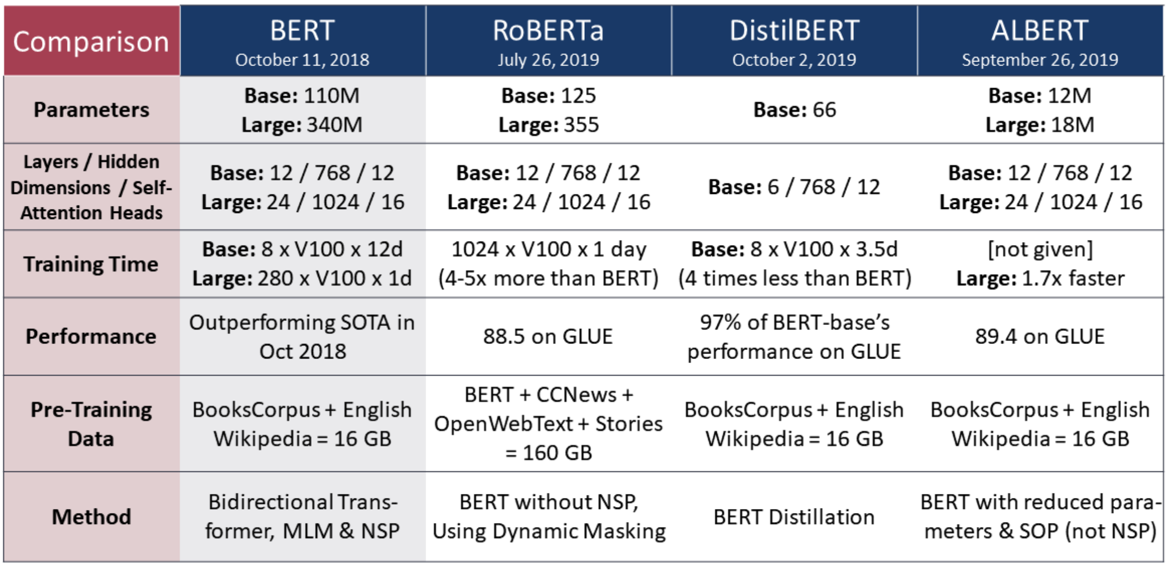
As of 2019, Google has been leveraging BERT to better understand user searches. The original English-language BERT has two models:

1. the BERTBASE: 12 Encoders with 12 bidirectional self-attention heads
2. the BERTLARGE: 24 Encoders with 16 bidirectional self-attention heads.

after that, numerous models appeared including :

- ALBERT:
  Stands for "A LITE BERT FOR SELF-SUPERVISED LEARNING OF LANGUAGE REPRESENTATIONS". It presents two parameter-reduction techniques to lower memory consumption and increase the training speed of BERT (Lan et al. 2020):
  - Splitting the embedding matrix into two smaller matrices.
  - Using repeating layers split among groups.
- RoBERTa:
  A Robustly Optimized BERT Pretraining Approach by (Y. Liu et al. 2019) . It builds on BERT and modifies key hyperparameters, removing the next-sentence pretraining objective and training with much larger mini-batches and learning rates.
- DistilBERT:
  A distilled version of BERT. It is a small, fast, cheap and light Transformer model trained by distilling BERT base. It has 40% less parameters than bert-base-uncased, runs 60% faster while preserving over 95% of BERT's performances as measured on the GLUE language understanding benchmark (Sanh et al. 2020).

In the table below (Figure 2.18) summarize the model differences between BERT, RoBERTa, DistilBERT and ALBERT



Comparison	BERT October 11, 2018	RoBERTa July 26, 2019	DistilBERT October 2, 2019	ALBERT September 26, 2019
Parameters	**Base:** 110M **Large:** 340M	**Base:** 125 **Large:** 355	**Base:** 66	**Base:** 12M **Large:** 18M
Layers / Hidden Dimensions / Self-Attention Heads	**Base:** 12 / 768 / 12 **Large:** 24 / 1024 / 16	**Base:** 12 / 768 / 12 **Large:** 24 / 1024 / 16	**Base:** 6 / 768 / 12	**Base:** 12 / 768 / 12 **Large:** 24 / 1024 / 16
Training Time	**Base:** 8 x V100 x 12d **Large:** 280 x V100 x 1d	1024 x V100 x 1 day (4-5x more than BERT)	**Base:** 8 x V100 x 3.5d (4 times less than BERT)	[not given] **Large:** 1.7x faster
Performance	Outperforming SOTA in Oct 2018	88.5 on GLUE	97% of BERT-base's performance on GLUE	89.4 on GLUE
Pre-Training Data	BooksCorpus + English Wikipedia = 16 GB	BERT + CCNews + OpenWebText + Stories = 160 GB	BooksCorpus + English Wikipedia = 16 GB	BooksCorpus + English Wikipedia = 16 GB
Method	Bidirectional Transformer, MLM & NSP	BERT without NSP, Using Dynamic Masking	BERT Distillation	BERT with reduced parameters & SOP (not NSP)

Figure 2.18: Comparison between BERT, RoBERTa, DistilBERT and ALBERT (*Economic Uncertainty Identification* 2021).

Advantages of Fine-Tuning in BERT

We have already discussed the fine-tuning approach. A simple fine-tuning procedure is typically adding one fully connected layer on top of BERT and training for a few epochs.

- Quicker Development:

  The fine-tuned model takes significantly less time to train. It's as if we've already thoroughly trained the bottom layers of our network and all we have to do now is fine-tune them while using their output as features in our assignment.

- Less Data:

  We can fine-tune our task using pre-trained weights on a considerably smaller dataset than we could with a model constructed from start. In order to train our network to a decent accuracy, an NLP model constructed from scratch often requires a prohibitively big dataset. As a result, we should spend a significant amount of time and effort on data collection. BERT can be fine-tuned by doing so.

- Better Results:

  For a range of tasks including as classification, linguistic inference, semantic similarity, question answering, and so on, this easy fine-tuning technique was demonstrated to yield state-of-the-art results with little task-specific tweaks. Rather than building specialized or complex architectures that have been proved to perform well on a certain job, fine-tuning BERT has been proved to be a better (or, in some cases, equivalent) proxy.

Limitations and challenges of BERT

- Sequence length :
  One of BERT's flaws is its inability to handle long text sequences. BERT supports up to 512 tokens by default (Devlin et al. 2018) . There are several options for dealing with it :
  - Ignore text after 512 tokens.
  - Split tokens to 2 or more inputs and predict it separately.
- Tokenization problem :
  At every sub-word unit, BERT chooses the biggest in-vocab substring from the left for the tokenized output. While this works quite well for words with suffixed roots (or stems), prefixed terms are prone to bad tokenization (Nayak et al. 2020).

  Taking "deconstructed, deactivated and unequal" as examples, even though the vocabulary had the prefixes de and un as well as the words constructed, activated and equal, the tokenizer chose the substrings deco, dea and une
  if the BERT tokenizer correctly separates the prefixes while the model is being trained, it can help the model to learn better representations for the prefix as well as the subword units since the attention mechanism would understand the influence of the different categories of prefixes

2.4 Online reviews for products rating

Product reviews are posted online by the hundreds and thousands for popular products. Handling such a large volume of continuously generated online content is a challenging task for buyers, sellers and researchers.
Our purpose is to rate a product based on online consumer reviews in the Arabic language

using BERT. Before we start we need to know what is Online reviews, products rating, Arabic language and its challenges in sentiment analysis.

2.4.1 Online reviews

Consumers want tools to assist them in deciding whether or not to acquire a product or service when presented with a large number of options. Customer recommendations are one of the most frequent tools: consumer evaluations, suggestions, or references may influence decisions ranging from which hotel to stay in to which smartphone app to download (Fagerstrøm et al. 2010). Word-of-mouth (WOM), or what other customers say about a product or service, is the most common kind of customer recommendation. Such suggestions are believed to have a greater influence on purchase decisions than suggestions from companies or advertisements since they are seen as more trustworthy (Arndt 1967). As a result, customer WOM has been recognized as the most powerful element in predicting the long-term success of experience goods. Twitter comments have recently been identified as one of the most important factors of brand image (Robson et al. 2013). Indeed, WOM can make or break a consumer's decision to buy.

- Importance and challenges

  Consumers and marketers may both benefit from online reviews wealth of information: consumers may use them to aid decision-making, while marketers may utilize them as a source of useful feedback. Extracting information from customer evaluations, on the other hand, is not without its difficulties; in fact, both marketers and consumers struggle to make sense of the huge quantity of data accessible in online evaluations. For example, online reviews can be based on ratings (i.e. a five-star rating), on a rating plus a comment, or on comments alone. Furthermore, even a five-star rating is not clearly defined (for instance, how do consumers define the threshold between stars?). Finally, a user may give a product a high star rating but still make critical comments about it (or vice versa). In general, Internet evaluations and answers are unstructured and unsystematic. Despite the difficulties in reading user reviews, written customer feedback is one of the most valuable sources of information for both consumers and marketers.

  Consumer comments had a larger influence on purchase decisions and perceived

trustworthiness than star ratings, according to (Tsang et al. 2009).
Consumers read and utilized information supplied in written reviews, even if they looked at star ratings when making decisions (Mayzlin et al. 2003). The task of interpreting these evaluations is tough and time-consuming, and these obstacles are compounded by the sheer amount of online customer reviews: many items receive thousands of ratings and reviews. Indeed, one of the most difficult tasks for consumers and marketers when using internet evaluations is synthesizing the data into a usable message (Robson et al. 2013).

2.4.2 Products rating

Ratings are the consumer evaluations and judgments containing stars, digits or any quantitative number that you often see in the form of the five-star rating system. It can also be on a scale of 10 or 100. So, no matter the scale, whenever there's a number involved, the system is called a rating system.

- Advantages :
  - Rating scales add a measure of data precision
  - They create standardization, allowing you to compare different people, topic, or products easily
  - They work as a general system; therefore, appraisals and assessments can be created for almost anything
  - They provide an opportunity for things to be graded fairly
  - Equality can be reached in a more successful manner than other systems that are more subjective
- Disadvantages :
  - The disadvantage of rating systems is that information might be misunderstood.
  - There's also the risk of skewed statistics; the great majority of users will give it a good rating because they prefer it to competing options and wish to explain

their choice. Only a few handfuls will have performed an objective appraisal of the available options.

– The ranking is interpreted differently by various people. As with any grading system, some people are more charitable than others.

2.4.3 Relation between Online reviews and products rating

Ratings and reviews are basically evaluations and judgments given by customers, clients or consumers to a specific business, professional or a service. Ratings and reviews are given based on the consumer's experience and level of satisfaction with the specific service that they have availed.We see the product rating, but it's not always satisfactory because if only one consumer gives the product a five-star rating, it stays that way, which may differ from other individuals who haven't given the product a five-star rating but have given negative evaluations. As a result, the product's rating is a little hazy and may or may not be accurate.

2.4.4 Some product rating based on online reviews works

Using online consumer reviews to extract product ratings using BERT has gotten a lot of attention in recent years in a variety of languages, but not in Arabic. In this subsection we review the work done in this field :

- Aidan Browne used in (Browne 2020) the Yelp Open Dataset to predict the star rating of a review of a business (*Yelp Review Sentiment Dataset* 2021), The datasets were split training 600,000 reviews and test 50,000 reviews. When created (D1) Dataset 1 contained 699,985 reviews and (D2) Dataset 2 contained 700,004 reviews. As the difference is not significant it was decided to proceed using both datasets. three models were used ALBERT-base , ELECTRA-base and ELECTRA-base to get the accuracy Score see (Figure 2.19).
- In the work of (*nlptown/bert-base-multilingual-uncased-sentiment · Hugging Face* 2021) used bert-base-multilingual-uncased model finetuned for sentiment analysis on product reviews in six languages: English, Dutch, German, French, Spanish and Italian. It predicts the sentiment of the review as a number of stars (between 1 and

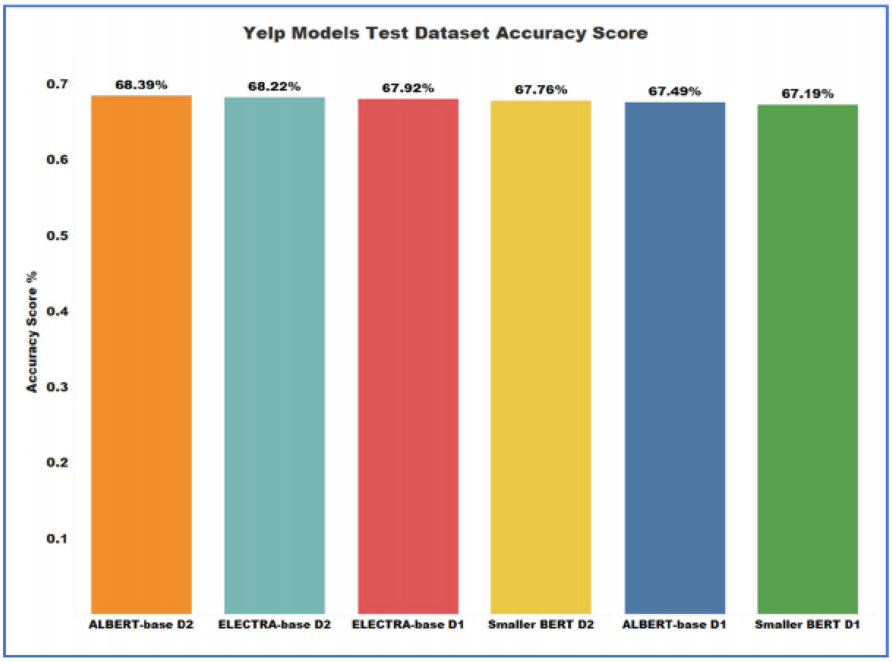


Figure 2.19: Final Test Dataset Accuracy Score (Browne 2020).

5).

The finetuned model obtained the following accuracy on 5,000 held-out product reviews in each of the languages (Table 2.1):

– Accuracy (exact) is the exact match on the number of stars.

– Accuracy (off-by-1) is the percentage of reviews where the number of stars the model predicts differs by a maximum of 1 from the number given by the human reviewer.

Language	Accuracy (exact)	Accuracy (off-by-1)
English	67%	95%
Dutch	57%	93%
German	61%	94%
French	59%	94%
Italian	59%	95%
Spanish	58%	95%

Table 2.1: Nlptown Accuracy Score for multiple languages .

Here is the number of product reviews we used for finetuning the model:

Language	Number of reviews
English	150k
Dutch	80k
German	137k
French	140k
Italian	72k
Spanish	50k

Table 2.2: MORPHOLOGICAL CHARACTERISTICS.

2.4.5 Arabic reviews analysis

With over 300 million speakers in twenty-two Arab nations, Arabic is one of the world's main languages. Arabic, along with Chinese, Russian, English, French, and Spanish, was designated as one of the United Nations' official languages in 1974. As a Semitic language, Arabic possesses many unique linguistic characteristics such as writing from the right to the left, the dual number of the nouns which is not found in English, the two genders, feminine and masculine, beside the root, the most salient feature of Semitic languages (Al-Huri 2016).

In comparison to English, Arabic is a morphologically rich language with few resources and a less researched syntax. Due to these constraints, Arabic Natural Language Processing (NLP) tasks such as Sentiment Analysis are limited.

Due to its nature and qualities, the Arabic language is difficult and complex. The following paragraphs demonstrate Arabic's intricacy.

- Word meaning :

  The term "word" refers to a single, isolated item that exists between two spaces and has a specific meaning. It is typical in Arabic for a single word to have many meanings depending on the situation (Al-Osaimi 2017). For example (Table 2.3)

Sentences	Phrases in English	Word Meaning
سهل منبسط	”flat plain”	Flat floor
بن سهل سعد	”Sahel bin Saad”	Name
المرام سهل	”easy to get”	Easy

Table 2.3: MEANINGS OF THE WORD SAHEL AS A NOUN.

- Variations in lexical category :

  A word can be a noun, verb, or particle in Arabic linguistics. Prepositions and conjunctions, for example, are classified as particles since they are neither nouns or verbs.(Table 2.4)

Word Type	Example in English	English Translation
Noun	كتاب	Book
Verb	يكتب	Write
Particle	على	On

Table 2.4: WORD TYPES IN THE ARABIC LANGUAGE.

Furthermore, depending on the context, a word might belong to many lexical groups. (Table 2.5) demonstrates how the term halq might be employed in various contexts.

Phrases	Phrases in English	Word Category	Word Meaning
حلق الانسان	”human throat”	Noun	Throat
حلق رأسه	”shaving his head”	Verb	Shaving
حلق الطائر	”flying bird”	Verb	Fly

Table 2.5: LEXICAL CATEGORIES FOR THE WORD HALQ.

- Morphological characteristics :

  A word, as defined above, is a single, isolated entity with a specific meaning. A word in Arabic can be a noun, verb, or particle, and the same word can be classified differently depending on the context.

  The smallest language unit with meaning is the morpheme. A morpheme cannot be broken down into smaller pieces. Morphemes should offer the word they are a part of a meaning (Al-Osaimi 2017).

A root is a single morpheme that gives a word its essential meaning. In Arabic, the root is the word's initial form, before any alteration takes place. A single root can be used to create a variety of words (Al-Osaimi 2017).

A stem is a morpheme that does not have an affix. The stem conveys a distinct concept or meaning. The root is sometimes known as the stem or word base in English, although the stem (or base) in Arabic is distinct from the root. The morphological properties of Arabic are shown in (Table 2.6).

Morphological characteristics	Definition
Word	a single and isolated item between two spaces
Morpheme	smallest linguistic unit that has a meaning
Stem	The basic form of word
Root	The original form of word

Table 2.6: MORPHOLOGICAL CHARACTERISTICS.

The stemming technique is commonly used in text mining to transform a term to its root form. The stemming process's principal goal is to eliminate all potential affixes, lowering the complexity of a word and the number of features and tokens in corpora (Al-Sughaiyer et al. 2004). Because the Arabic root is context-dependent, a stem can lead to many definitions in Arabic (Mesleh 2008). (Table 2.7) shows terms that share the same root but have diverse meanings.

Sentences	English translate	Root	Meaning
يخرج من المنزل	"leaves home"	خرج	Goes out
يتخرج من الجامعة	"graduates from college"	خرج	Graduate

Table 2.7: DIFFERENT WORDS WITH THE SAME ROOT.

- Vowelization or diacritization :

  n is the process of putting diacritical mark vowels above or under letters in Arabic words (fateha ,kasrah ,dammah). Nunation is the process of putting a set of diacritically marked vowels at the end of a word to create the sound of the letter

N. The kasheeda or tatweel is the symbol used to stretch some Arabic characters (Al-Sughaiyer et al. 2004) .

2.5 Conclusion

Throughout this chapter, we have provided details about Sentiment analysis and word embeddings. We focused on the BERT model and the Arabic online reviews rating problem that we are going to address.

In the next chapter, we will introduce the design of the system, dataset and the proposed architectures will be mentioned as well.

Chapter 3

System Design

3.1 Introduction

In this chapter, We present our conceptual vision for rating based on product reviews from Arabic texts using BERT. Then we are going to introduce and explain the general architecture and show how to implement the different phases of this system.

3.2 General System Architecture

Generally. Our system will be following certain steps, as we represent the system in (Figure 3.1).

The first step is to get the reviews and their ratings from the data set then prepossess it to get the final dataset then fine tune a pretrained BERT model for our classification task finally, we get our model to classify and rate the consumers reviews.

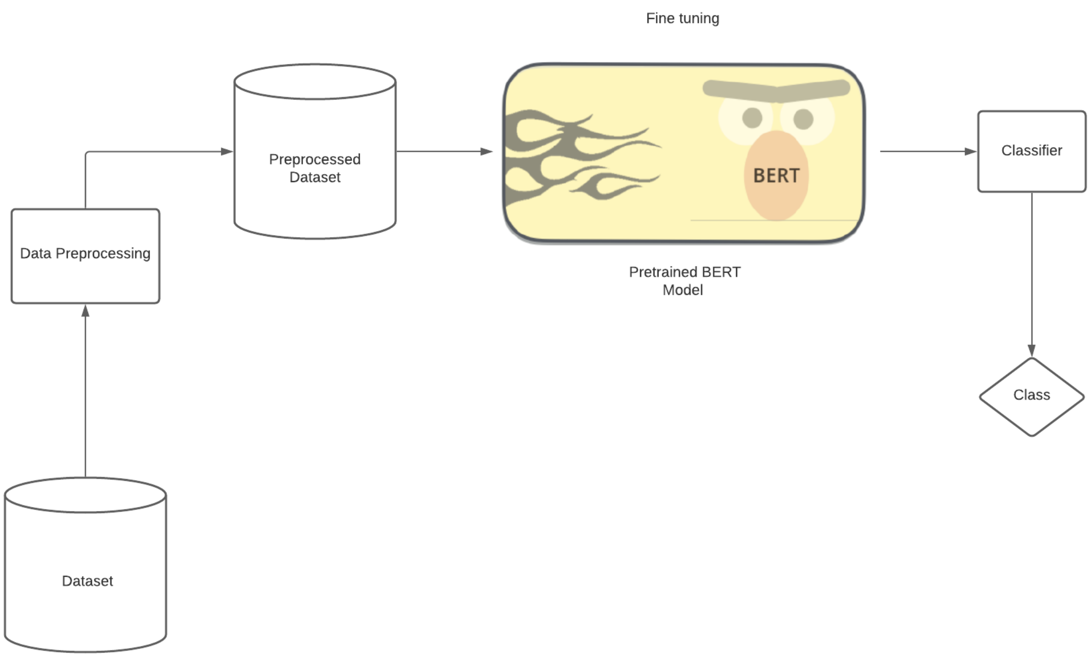


Figure 3.1: The General system architecture.

3.3 Detailed System Architecture

3.3.1 Dataset description

Due to the lack of Arabic product reviews and their rating (between 1 and 5) Dataset We used Amazon Fine Food Reviews Dataset from Kaggle (*Amazon Fine Food Reviews* 2021) then we translate it to Arabic.

This dataset consists of reviews of fine foods from amazon. The data span a period of more than 10 years, including all 500,000 reviews up to October 2012. Reviews include product and user information, ratings, and a plain text review. It also includes reviews from all other Amazon categories.

Dataset structure

Data includes:

- Reviews from Oct 1999 - Oct 2012
- 568,454 reviews
- 256,059 users
- 74,258 products
- 260 users with > 50 reviews

3.3.2 Preprocessing

As we can see in (Figure 3.2) the Dataset contains multiple columns and we only need the Score and Text columns. For the equivalent of the classification we took 10000 reviews divided into 2000 reviews for each rating (between 1 and 5) and save it into a new CSV file.

	Id	ProductId	UserId	ProfileName	HelpfulnessNumerator	HelpfulnessDenominator	Score	Time	Summary	Text
0	1	B001E4KFG0	A3SGXH7AUHU8GW	delmartian	1	1	5	1303862400	Good Quality Dog Food	I have bought several of the Vitality canned d...
1	2	B00813GRG4	A1D87F6ZCVE5NK	dll pa	0	0	1	1346976000	Not as Advertised	Product arrived labeled as Jumbo Salted Peanut...
2	3	B000LQOCH0	ABXLMWJIXXAIN	Natalia Corres "Natalia Corres"	1	1	4	1219017600	"Delight" says it all	This is a confection that has been around a fe...
3	4	B000UA0QIQ	A395BORC6FGVXV	Karl	3	3	2	1307923200	Cough Medicine	If you are looking for the secret ingredient i...
4	5	B006K2ZZ7K	A1UQRSCLF8GW1T	Michael D. Bigham "M. Wassir"	0	0	5	1350777600	Great taffy	Great taffy at a great price. There was a wid...

Figure 3.2: Sample of Amazon Fine Food Reviews dataset.

Then we translate the Reviews from English to Arabic (Figure 3.3):

```
Review in english :
Candy was delivered very fast and was purchased at a reasonable price.  I was home bound and unable to get to a store so this was perfect for me.
translated Review to arabic :
تم تسليم الحلوى بسرعة كبيرة وتم شراؤها بسعر معقول. كنت متجها إلى المنزل وغير قادر على الوصول إلى متجر ، لذا كان هذا مثاليًا بالنسبة لي.
Review Rating : 5 Stars
```

Figure 3.3: Translated Review to Arabic .

Tokenization

To feed our text to BERT, it must be split into tokens, and then these tokens must be mapped to their index in the tokenizer vocabulary (Figure 3.4)

```
Original:  هذا طعام كلب صحي للغاية. جيد للهضم. جيد أيضًا للجراء الصغيرة. كلبي يأكل الكمية المطلوبة لها في كل رضعة.
Tokenized:  ['هذا', 'طعام', 'كلب', 'صحي', 'للغاية', '.', 'جيد', 'لله', '##ضم', '.', 'جيد', '[UNK]', 'للج', '##راء', 'الصغيرة', '.', 'كلب', '##ي', 'يأكل', 'الكمية', 'المطلوبة', 'لها', 'في', 'كل', 'رض', '##عة', '.']
Token IDs:  [391, 7741, 394, 1217, 418, 11989, 183, 461, 4593, 9095, 461, 4094, 20191, 7154, 306, 19846, 103, 732, 35974, 450, 10963, 777, 18050, 35372, 13664, 306, 10592, 8, 1980, 21416, 325,
```

Figure 3.4: Example of tokenazing and mapping a text.

3.3.3 Fine tuning BERT model

In this step, we will use our preprocessed Dataset to fine tune the AraBERTv2 Model which the second version of AraBERT (Antoun et al. 2020) model for the classification task

AraBERT :

AraBERT is an Arabic pretrained language model based on Google's BERT architecture. AraBERT uses the same BERT-Base config. According to (Antoun et al. 2021) The pretraining data used for AraBERTv2 :

- OSCAR unshuffled and filtered.
- Arabic Wikipedia dump from 2020/09/01
- The 1.5B words Arabic Corpus
- The OSIAN Corpus
- Assafir news articles.

To use AraBERTv2 for Classification task we feed the model with reviews(single sentence) with max length of 512 . (Figure 3.5) explains this task

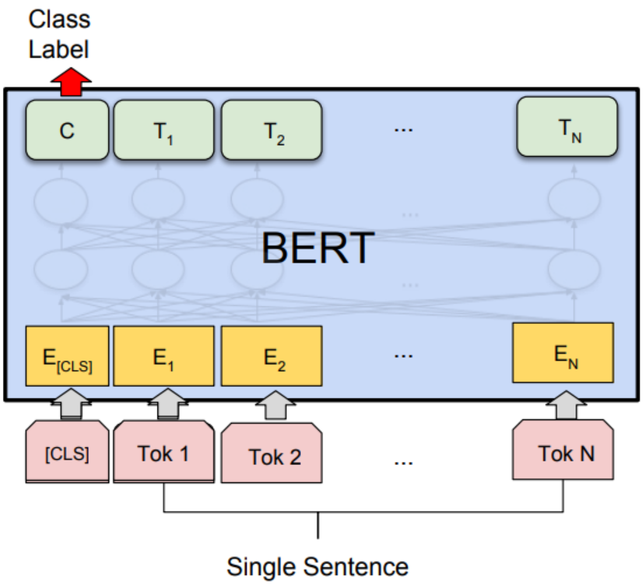


Figure 3.5: BERT text classification (Taunk 2020).

then we save the model to use in Classification and rating consumers reviews.

3.4 Conclusion

In this chapter, we have presented our proposed method to arrive at our model. we have detailed the method used and the three essential phases(dataset description,preprocessing and fine tuning). And in the next chapter, we will describe the implementation of our system and its results.

Chapter 4

Implementation and results

4.1 Introduction

This section contains the core of our work; we will demonstrate how to carry out these steps and give the most important result. The working environment, programming language, and tools we utilized to develop the system will all be discussed. We developed the scripts using the Python programming language and the Google Colab environment to build and evaluate our system.

4.2 Implementation frameworks and tools

4.2.1 Development environment

Python

Figure 4.1: Python language logo.

To validate our application we will adopt Python which is a powerful versatile programming language. Python is a high-level programming language that is interpreted, interactive, object-oriented, and general-purpose. Guido van Rossum designed it between 1985 and 1990 (*Welcome to Python.org* 2021) .

Google colab

Colaboratory, or ”Colab” for short, allows you to write and execute Python in your browser, with (*Google Colaboratory* 2021):

- Zero configuration required
- Free access to GPUs
- Easy sharing

Figure 4.2: Google colab logo.

4.2.2 development tools

TensorFlow

Figure 4.3: Tensorflow logo.

TensorFlow is an open-source machine learning platform that runs from start to finish. It features a large, flexible ecosystem of tools, libraries, and community resources that allow academics to advance the state-of-the-art in machine learning and developers to quickly construct and deploy ML applications (*TensorFlow* 2021).

Pytorch

PyTorch is a tensor library intended for usage with GPUs and CPUs in Deep Learning applications. It is a Python-based open-source machine learning package created mostly by the Facebook AI Research team. It's one of the most well-known Machine Learning libraries (*PyTorch* 2021).

Figure 4.4: Pytorch logo.

Transformers

Transformers (formerly pytorch-transformers and pytorch-pretrained-bert) offers general-purpose architectures (BERT, GPT-2, RoBERTa, XLM, DistilBert, XLNet...) for Natural Language Understanding (NLU) and Natural Language Generation (NLG) with over 32+ pretrained models in 100+ languages and deep interoperability between Jax, PyTorch, and TensorFlow (*Transformers* 2021).

Figure 4.5: Huggingface transformers logo.

Pandas

pandas is a data manipulation and analysis software package for the Python programming language. It includes data structures and procedures for manipulating numerical tables and time series, in particular (*pandas - Python Data Analysis Library* 2021). We will use it to manipulate the dataset.

Figure 4.6: Pandas logo.

Numpy

NumPy is perhaps the most important Python package for scientific computing. NumPy arrays make it easier to do complex mathematical and other operations on enormous amounts of data (*NumPy* 2021).

Figure 4.7: Numpy logo.

Deep_translator

A flexible free and unlimited tool to translate between different languages in a simple way using multiple translators. Supports multiple translators such as (Baccouri 2021):

- Support for google translate
- Support for the microsoft translator (version >= 1.3.5)
- Support for Pons translator
- Support for the Linguee translator
- Support for the Mymemory translator
- Support for the Yandex translator (version >= 1.2.1)
- Support for the QCRI translator (version >= 1.2.4)
- Support for the DeepL translator (version >= 1.2.5)
- Support for the Papago translator (version >= 1.4.4)

Figure 4.8: Deep_translator logo.

Matplotlib

matplotlib pyplot is a set of functions that allow matplotlib to behave similarly to MATLAB. Each pyplot function modifies a figure in some way, such as creating a figure, a plotting area in a figure, charting certain lines in a plotting area, decorating the plot with labels, and so on (*Matplotlib: Python plotting — Matplotlib 3.4.2 documentation* 2021).

Figure 4.9: Matplotlib logo.

4.3 Loading and preprocessing the dataset

As mentioned in the preceding chapter, we load (**noauthor_amazon_nodat**) dataset than to ensures the balance we take 2000 reviews from each rating to collect 10000 reviews overall.
after that, we translate the dataset to Arabic with the help of deep_translator as shown in (Figure 4.10)

```
[ ] # Get the lists of sentences and their labels.
    sentences = df["text"]
    labels = df["rating"]
    df["rating"].value_counts()
    print("Review in english :")
    print(df1["Text"][19])
    print("translated Review to arabic :")
    print(sentences[19])

    print("Review Rating : 5 Stars ")

    Review in english :
    Candy was delivered very fast and was purchased at a reasonable price.  I was home bound and unable to get to a store so this was perfect for me.
    translated Review to arabic :
    تم تسليم الحلوى بسرعة كبيرة وتم شراؤها بسعر معقول. كنت متجهًا إلى المنزل وغير قادر على الوصول إلى متجر ، لذا كان هذا مثاليًا بالنسبة لي.
    Review Rating : 5 Stars
```

Figure 4.10: Example of a translated review from English to Arabic.

4.3.1 Tokenization

we'll transform our dataset into the format that BERT can be trained on.

To feed our text to AraBERTv2, we must first break it down into tokens, then map each token to its corresponding index in the tokenizer vocabulary. For example see (Figure 4.11)

```
print(' Original: ', sentences[9])
print('Tokenized: ', tokenizer.tokenize(sentences[9]))

print('Token IDs: ', tokenizer.convert_tokens_to_ids(tokenizer.tokenize(sentences[10])))

 Original:  هذا طعام كلب صحي للغاية. جيد للهضم. جيد أيضًا للجراء الصغيرة. كلبي يأكل الكمية المطلوبة لها في كل رضعة.
Tokenized:  ['هذا', 'طعام', 'كلب', 'صحي', 'للغاية', '.', 'جيد', 'لله', '##ضم', '.', 'جيد', '[UNK]', '##عة', 'رض', 'كل', 'في', 'لها', 'المطلوبة', 'الكمية', 'يأكل', '##ي', 'كلب', '.', 'الصغيرة', '##راء', 'للج', '
Token IDs:  [391, 7741, 394, 1217, 418, 11989, 183, 461, 4593, 9095, 461, 4094, 20191, 7154, 306, 19846, 103, 732, 35974, 450, 10963, 777, 18050, 35372, 13664, 306, 10592, 8, 1980, 21416,
```

Figure 4.11: Example of using the tokenizer on a single sentence.

4.3.2 Dataset spliting

90% of our dataset will be used for training, and 10% will be used for validation (Figure 4.12).

```
[ ] from torch.utils.data import TensorDataset, random_split

    dataset = TensorDataset(input_ids, attention_masks, labels)

    train_size = int(0.9 * len(dataset))
    val_size = len(dataset) - train_size

    train_dataset, val_dataset = random_split(dataset, [train_size, val_size])
    print('{:>5,} training samples'.format(train_size))
    print('{:>5,} validation samples'.format(val_size))

    9,000 training samples
    1,000 validation samples
```

Figure 4.12: Training and validation Split.

After preprocessing our dataset. We jump into the next step witch is training and creating our model.

4.4 Fine tuning Classification Model

In this step we used :

- The GPU: Tesla T4 for the training.
- Learning Rate
- Batch size of 32
- Number of training epochs = 4. (The BERT authors recommend between 2 and 4).

4.4.1 Text Classification with BERT

Load BertForSequenceClassification, the pretrained BERT model with a single linear classification layer on top. And tell pytorch to run this model on the GPU (Figure 4.13).

```
[ ]  from transformers import BertForSequenceClassification, AdamW, BertConfig

     model = BertForSequenceClassification.from_pretrained(
         "aubmindlab/bert-base-arabertv02",
         num_labels = len(set(labels)),

         output_attentions = False,
         output_hidden_states = False,
     )

     model.cuda()
```

Figure 4.13: Training and validation Split.

4.4.2 helper functions

We define 3 functions to help measure and evaluate our training (Figure 4.14) .
Define a helper function for calculating accuracy.

Define a helper function for computing accuracy (off-by-1). The proportion of reviews where the model predicts a number of stars that differs by no more than one from the number supplied by the human reviewer is called accuracy (off-by-1).
Helper function for formatting elapsed times as ‘hh:mm:ss‘

```python
import numpy as np

# Function to calculate the accuracy of our predictions vs labels
def flat_accuracy(preds, labels):
    pred_flat = np.argmax(preds, axis=1).flatten()
    labels_flat = labels.flatten()
    return np.sum(pred_flat == labels_flat) / len(labels_flat)

def off_by_one_accuracy(preds, labels):
    pred_flat = np.argmax(preds, axis=1).flatten()

    labels_flat = labels.flatten()
    i1=np.sum(pred_flat == labels_flat)
    i2 =np.sum(pred_flat == labels_flat+1)
    i3=np.sum(pred_flat == labels_flat-1)
    i=(i1+i2+i3 )/ len(labels_flat)
    return i / len(labels_flat)
import time
import datetime

def format_time(elapsed):
    '''
    Takes a time in seconds and returns a string hh:mm:ss
    '''
    # Round to the nearest second.
    elapsed_rounded = int(round((elapsed)))

    # Format as hh:mm:ss
    return str(datetime.timedelta(seconds=elapsed_rounded))
```

Figure 4.14: Helper functions code.

4.4.3 Training the model

after shuffling our data we train it with a batch size of 32 through 4 epochs. Then we start the training. We'll keep track of a variety of metrics, including training and validation loss, validation accuracy, and timings. (Figure 4.15) shows the output of epoch 1/4 in the fine tuning phase.

```
======== Epoch 1 / 4 ========
Training...
  Batch    40  of    282.    Elapsed: 0:00:16.
  Batch    80  of    282.    Elapsed: 0:00:32.
  Batch   120  of    282.    Elapsed: 0:00:47.
  Batch   160  of    282.    Elapsed: 0:01:03.
  Batch   200  of    282.    Elapsed: 0:01:19.
  Batch   240  of    282.    Elapsed: 0:01:34.
  Batch   280  of    282.    Elapsed: 0:01:50.

  Average training loss: 2.36
  Training epcoh took: 0:01:50

Running Validation...
  Accuracy: 0.67
  off by one Accuracy: 0.89
  Validation Loss: 1.03
  Validation took: 0:00:04
```

Figure 4.15: Output of epoch 1/4 .

4.4.4 Saving the fine tuned model

After several training tests, we save the best model with the best results to google drive, so we can use it for the classification task (Figure 4.16).

```
import os

output_dir = '/content/drive/MyDrive/model_save/'

if not os.path.exists(output_dir):
    os.makedirs(output_dir)

print("Saving model to %s" % output_dir)

model_to_save = model.module if hasattr(model, 'module') else model
model_to_save.save_pretrained(output_dir)
tokenizer.save_pretrained(output_dir)
```

```
Saving model to /content/drive/MyDrive/model_save/
('/content/drive/MyDrive/model_save/tokenizer_config.json',
 '/content/drive/MyDrive/model_save/special_tokens_map.json',
 '/content/drive/MyDrive/model_save/vocab.txt',
 '/content/drive/MyDrive/model_save/added_tokens.json',
 '/content/drive/MyDrive/model_save/tokenizer.json')
```

Figure 4.16: Saving the fine tuned model.

4.5 Results Comparison

4.5.1 Model evaluation

Finally, we evaluate the performance of our model as shown in (Table 4.1) and we see that we have reached 69% accuracy and 96% (off by one) Accuracy with a Total training time of 0:07:37 (h:mm:ss) .

epoch	Training Loss	Validation Loss	Validation Accuracy	(off by one) Accuracy	Training Time	Validation Time
1	2.36	1.03	0.67	0.89	0:01:50	0:00:04
2	0.88	0.91	0.68	0.96	0:01:50	0:00:04
3	0.71	0.90	0.69	0.97	0:01:50	0:00:04
4	0.62	0.92	0.69	0.96	0:01:50	0:00:04

Table 4.1: Summary of the training process.

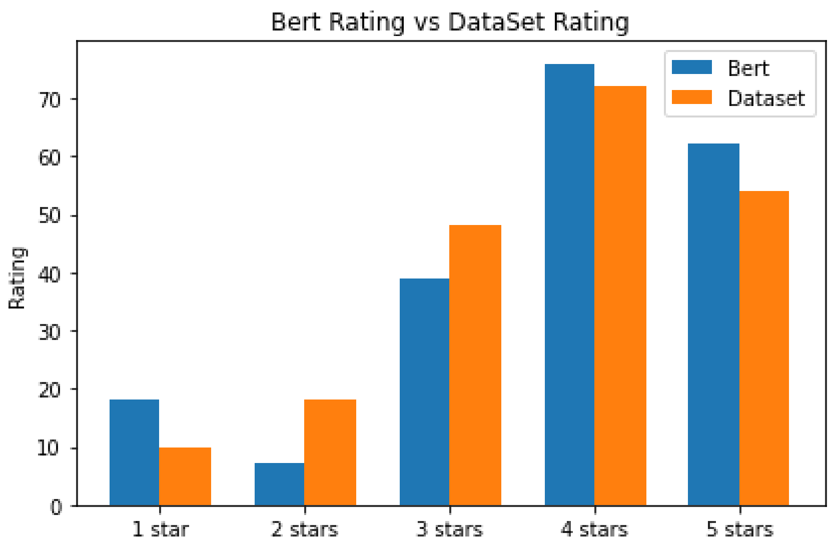
To get a better look at our model we use another Dataset(*Hotel Reviews - dataset by datafiniti | data.world* 2021). this dataset got a list of 1,000 hotels and their online reviews.
we took for example "Metro Points Hotel-Washington North" hotel reviews (202 reviews)and compare them with our model result (Figure 4.17).

4.5.2 Comparisons our work and the previous works

Despite the lack of a dataset in Arabic and the colab GPU usage limit. We mange to get a better result using AraBERTv2 (Antoun et al. 2020) than :

- In the work of (Browne 2020) : their best results with ALBERT-base were 68%
- The work of (*nlptown/bert-base-multilingual-uncased-sentiment · Hugging Face* 2021) : their best results with were the English language with exact accuracy 67% and an (off-by-1) accuracy of 95%

we conclude from this result that there are great prospects for development and to get better for the Arabic language in this field of NLP.




```
Bert Rating average  : 3.7 stars
DataSet Rating average  : 3.8 stars
```

Figure 4.17: Dataset vs model rating chart .

4.6 Conclusion

The implementation and experiments were described and explained in this chapter, the results were shown as figures, tables were commented on, and comparisons of the acquired related works were made.

Conclusion and perspectives

Conclusion

With the most up-to-date advancements in sentiment analysis. allowing for more development in text classification, particularly in product rating. In this work, we have explored the potential of the Arabic BERT model in rating products based on online reviews in the Arabic language. The proposed deep learning model showed a promising results, which encourages us to improve our research more and more.
This study allowed us to put our neural network expertise into practice and enhance it, and most importantly, it allowed us to take the first step toward deep learning, one of the most significant disciplines of artificial intelligence.

Perspectives

We have suggestions for future perspectives and views that can help us enhance our work such as :

- train our model with Arabic reviews dataset from different dialects.
- fine tune our model on a bigger dataset.
- train our model for multitasks like summarizing and rating a product reviews.

Bibliographie

Aggarwal, Deepti G. and CSE Department, JSS Academy of Technical Education, Noida, Uttar Pradesh, India (May 2018). "Sentiment Analysis An insight into Techniques, Application and Challenges." en. In: *International Journal of Computer Sciences and Engineering* 6.5, pp. 697–703.

Aizawa, Akiko (2003). "An information-theoretic perspective of tf–idf measures." In: *Information Processing & Management* 39.1, pp. 45–65.

Antoun, Wissam, Fady Baly, and Hazem Hajj (2020). "Arabert: Transformer-based model for arabic language understanding." In: *arXiv preprint arXiv:2003.00104*.

— (Mar. 2021). "AraBERT: Transformer-based Model for Arabic Language Understanding." In: *arXiv:2003.00104 [cs]*. arXiv: 2003.00104.

Arndt, Johan (1967). "Role of Product-Related Conversations in the Diffusion of a New Product." In: *Journal of Marketing Research* 4.3. Publisher: American Marketing Association, pp. 291–295.

Asghar, M. Z. et al. (2014). "A Review of Feature Extraction in Sentiment Analysis." en. In: *undefined*.

Browne, Aidan (2020). "Fined-Grained Sentiment Analysis of Yelp Reviews Using Deep Learning Models." en. MA thesis. Dublin, National College of Ireland.

Devlin, Jacob et al. (Oct. 2018). *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*.

Fagerstrøm, Asle and Gheorghita Ghinea (Mar. 2010). "Web 2.0's Marketing Impact on Low-Involvement Consumers." In: *Journal of Interactive Advertising* 10, pp. 67–71.

Funk, Adam et al. (Jan. 2008). "Opinion analysis for business intelligence applications." In: p. 3.

Gao, Lizheng et al. (Nov. 2019). "Word Embedding With Zipf's Context." In: *IEEE Access* PP, pp. 1–1.

Gershenson, C. (2003). "Artificial Neural Networks for Beginners." In: *ArXiv* cs.NE/0308031.

Goldberg, Yoav (Apr. 2017). "Neural Network Methods for Natural Language Processing." In: *Synthesis Lectures on Human Language Technologies* 10, pp. 1–309.

Herwanto, Guntur et al. (Dec. 2019). "Hate Speech and Abusive Language Classification using fastText." In: pp. 69–72.

Hochreiter, Sepp and Jürgen Schmidhuber (Nov. 1997). "Long Short-Term Memory." In: *Neural Computation* 9.8, pp. 1735–1780.

Al-Huri, Ibrahim (Aug. 2016). *Arabic Language: Historic and Sociolinguistic Characteristics.*

Kolkur, Seema, Gayatri Dantal, and Reena Mahe (2015). "Study of different levels for sentiment analysis." In: *International Journal of Current Engineering and Technology* 5.2, pp. 768–770.

Lan, Zhenzhong et al. (Feb. 2020). "ALBERT: A Lite BERT for Self-supervised Learning of Language Representations." In: *arXiv:1909.11942 [cs]*. arXiv: 1909.11942.

Li, Yang and Tao Yang (May 2017). "Word Embedding for Understanding Natural Language: A Survey." In: vol. 26.

Liu, Feng, Zhigang Chen, and Jie Wang (Feb. 2019). "Video image target monitoring based on RNN-LSTM." In: *Multimedia Tools and Applications* 78.

Liu, Yinhan et al. (July 2019). "RoBERTa: A Robustly Optimized BERT Pretraining Approach." In: *arXiv:1907.11692 [cs]*. arXiv: 1907.11692.

Luhach, Ashish Kumar et al. (Nov. 2019). *First International Conference on Sustainable Technologies for Computational Intelligence: Proceedings of ICTSCI 2019*. en. Google-Books-ID: 2ZO7DwAAQBAJ. Springer Nature.

Madhoushi, Z., A. Hamdan, and S. Zainudin (2015). "Sentiment analysis techniques in recent works." In: *2015 Science and Information Conference (SAI)*, pp. 288–291.

Mathew, Amitha, Amudha Arul, and S. Sivakumari (Jan. 2021). "Deep Learning Techniques: An Overview." In: pp. 599–608.

Mayzlin, Dina and Judith Chevalier (Sept. 2003). "The Effect of Word of Mouth on Sales: Online Book Reviews." In: *Journal of Marketing Research* 43.

Mesleh, Abdelwadood (Aug. 2008). "Support Vector Machines Based Arabic Language Text Classification System: Feature Selection Comparative Study." In: *Advances in*

Computer and Information Sciences and Engineering. Journal Abbreviation: Advances in Computer and Information Sciences and Engineering, pp. 11–16.

Mijwil, Maad, Adam Esen, and Aysar Alsaadi (Apr. 2019). "Overview of Neural Networks." In: 1, p. 2.

Nayak, Anmol et al. (Nov. 2020). "Domain adaptation challenges of BERT in tokenization and sub-word representations of Out-of-Vocabulary words." In: *Proceedings of the First Workshop on Insights from Negative Results in NLP*. Online: Association for Computational Linguistics, pp. 1–5.

Al-Osaimi, Salha (Feb. 2017). "Sentiment Analysis Challenges of Informal Arabic Language." In: *International Journal of Advanced Computer Science and Applications* 8.

Paul, Sayanta and Sriparna Saha (Nov. 2020). "CyberBERT: BERT for cyberbullying identification." In: *Multimedia Systems*, pp. 1–8.

Pennington, Jeffrey, Richard Socher, and Christopher Manning (Jan. 2014). "Glove: Global Vectors for Word Representation." In: vol. 14, pp. 1532–1543.

Poirier, Damien et al. (Nov. 2009). "Approches Statistique et Linguistique Pour la Classification de Textes d'Opinion Portant sur les Films." In:

Popov, Alexander (Mar. 2018). "Neural Network Models for Word Sense Disambiguation: An Overview." In: *Cybernetics and Information Technologies* 18, pp. 139–151.

Radford, Alec et al. (2018). "Improving language understanding by generative pre-training." In:

Robson, Karen et al. (July 2013). "Making Sense of Online Consumer Reviews: A Methodology." In: *International Journal of Market Research* 55, pp. 521–537.

Sanh, Victor et al. (Feb. 2020). "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter." In: *arXiv:1910.01108 [cs]*. arXiv: 1910.01108.

Shaw, Peter, Jakob Uszkoreit, and Ashish Vaswani (Apr. 2018). "Self-Attention with Relative Position Representations." In: *arXiv:1803.02155 [cs]*. arXiv: 1803.02155.

Staudemeyer, Ralf C. and Eric Rothstein Morris (Sept. 2019). "Understanding LSTM – a tutorial into Long Short-Term Memory Recurrent Neural Networks." In: *arXiv:1909.09586 [cs]*. arXiv: 1909.09586.

Stein, Jeremy, Harrison Hong, and Jeffrey Kubik (Feb. 2004). "Social Interaction and Stock-Market Participation." In: *Journal of Finance* 59, pp. 137–163.

Al-Sughaiyer, Imad A. and Ibrahim A. Al-Kharashi (2004). “Arabic morphological analysis techniques: A comprehensive survey.” In: *J. Assoc. Inf. Sci. Technol.*

Thakkar, Harsh and Dhiren Patel (Dec. 2015). “Approaches for Sentiment Analysis on Twitter: A State-of-Art study.” In: *arXiv:1512.01043 [cs]*. arXiv: 1512.01043.

Tribhuvan, Padmapani, S Bhirud, and Amrapali Tribhuvan (Feb. 2014). “A Peer Review of Feature Based Opinion Mining and Summarization.” In: *International Journal of Computer Science and Information Technologies* Vol. 5, pp. 247–250.

Tsang, Alex and Gerard Prendergast (Nov. 2009). “Is a “star” worth a thousand words?: The interplay between product-review texts and rating valences.” In: *European Journal of Marketing* 43, pp. 1269–1280.

Ulčar, Matej and Marko Robnik-Sikonja (Nov. 2019). *High Quality ELMo Embeddings for Seven Less-Resourced Languages*.

Vaswani, Ashish et al. (Dec. 2017). “Attention Is All You Need.” In: *arXiv:1706.03762 [cs]*. arXiv: 1706.03762.

Vaswani, Ashish et al. (Mar. 2018). “Tensor2Tensor for Neural Machine Translation.” In:

, vinay kumar vinay (Sept. 2019). “Sentiment Analysis Techniques for Social Media Data: A Review.” In:

Vincent, Pascal et al. (Jan. 2008). “Extracting and composing robust features with denoising autoencoders.” In: pp. 1096–1103.

Wang, Bin et al. (2019). “Evaluating Word Embedding Models: Methods and Experimental Results.” In: *APSIPA Transactions on Signal and Information Processing* 8. arXiv: 1901.09785, e19.

Wu, Yonghui et al. (Oct. 2016). “Google’s Neural Machine Translation System: Bridging the Gap between Human and Machine Translation.” In: *arXiv:1609.08144 [cs]*. arXiv: 1609.08144.

Webographie

Amazon Fine Food Reviews (2021). en. URL: https://kaggle.com/snap/amazon-fine-food-reviews (visited on 06/16/2021).

Baccouri, Nidhal (2021). *deep-translator: A flexible python tool to translate between different languages in a simple way.* URL: https://github.com/nidhaloff/deep_translator (visited on 06/16/2021).

Dandale, Priyanka (May 2021). *Build a system to identify fake news articles!* en. URL: https://medium.com/nerd-for-tech/build-a-system-to-identify-fake-news-articles-6604968043cb (visited on 06/23/2021).

dzlab (2021). *GPT-3 An Overview.* en-us. URL: https://dzlab.github.io/ml/2020/07/25/gpt3-overview/ (visited on 06/24/2021).

Economic Uncertainty Identification (2021). en-us. URL: https://humboldt-wi.github.io/blog/research/information_systems_1920/uncertainty_identification_transformers// (visited on 06/24/2021).

Ghosh, Arunabh (May 2020). *Deep Contextualized Word Representations — A new approach to word embeddings.* en. URL: https://towardsdatascience.com/deep-contextualized-word-representations-a-new-approach-to-word-embeddings-66e0f520654d (visited on 06/24/2021).

GloVe: Global Vectors for Word Representation (2021). URL: https://nlp.stanford.edu/projects/glove/ (visited on 06/24/2021).

Google Colaboratory (2021). en. URL: https://colab.research.google.com/notebooks/intro.ipynb#scrollTo=5fCEDCU_qrC0 (visited on 06/16/2021).

Gorakala, suresh kumar (Nov. 2017). *information retrieval document search using vector space model in R | R-bloggers.* en-US. URL: https://www.r-bloggers.com/2017/11/information-retrieval-document-search-using-vector-space-model-in-r/ (visited on 06/23/2021).

Hotel Reviews - dataset by datafiniti / data.world (2021). URL: https://data.world/datafiniti/hotel-reviews (visited on 06/18/2021).

Matplotlib: Python plotting — Matplotlib 3.4.2 documentation (2021). URL: https://matplotlib.org/ (visited on 06/16/2021).

nlptown/bert-base-multilingual-uncased-sentiment · Hugging Face (2021). URL: https://huggingface.co/nlptown/bert-base-multilingual-uncased-sentiment (visited on 06/15/2021).

NumPy (2021). URL: https://numpy.org/ (visited on 06/16/2021).

pandas - Python Data Analysis Library (2021). URL: https://pandas.pydata.org/ (visited on 06/16/2021).

PyTorch (2021). en. URL: https://www.pytorch.org (visited on 06/16/2021).

Taunk, Dhaval (June 2020). *How to fine-tune BERT on text classification task?* en. URL: https://medium.com/analytics-vidhya/how-to-fine-tune-bert-on-text-classification-task-723f82786f61 (visited on 06/24/2021).

TensorFlow (2021). en. URL: https://www.tensorflow.org/ (visited on 06/16/2021).

Transformers (2021). en. URL: index.html (visited on 06/16/2021).

Tulasiram (Mar. 2021). *My Internship Experience on Standard NLP Work Flow, word-embedding, sentence embedding, SOTA models*. en. URL: https://tulasiram574.medium.com/my-internship-experience-on-standard-nlp-work-flow-word-embedding-sentence-embedding-sota-models-f7254d068d4a (visited on 06/24/2021).

Welcome to Python.org (2021). en. URL: https://www.python.org/ (visited on 06/16/2021).

Yelp Review Sentiment Dataset (2021). en. URL: https://kaggle.com/ilhamfp31/yelp-review-dataset (visited on 06/15/2021).