REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Mohamed Khider – BISKRA
Faculté des Sciences Exactes, des Sciences de la Nature et de la Vie
Département d'informatique

N° d'ordre :SIOD 15/M2/2021

Mémoire

Présenté pour obtenir le diplôme de master académique en

Informatique

Parcours : **Systèmes d'information, Optimisation et Décision (SIOD)**

Topic Modeling For Arabic Short Texts

Par :

HERIZ OUMAIMA

Soutenu le 22/06/2022 devant le jury composé de :

TAREK ZERNADJI	MCB	Président
OKBA TIBERMACINE	Professeur	Rapporteur
MOUSSAOUI MANEL	MAA	Examinateur

Année universitaire 2021-2022

Acknowledgements

I would like to express my sincere gratitude to my supervisor "DR.Tibermacine Okba" for the continuous support of this work,for his patience, motivation and immense knowledge. His guidance helped me in all the time of research. Besides my supervisor, I would like to thank the members of the discussion committee for their kind reading of this note And all those who have helped me do this work And thanks to all the professors of the computer science department.

Dedication

First and foremost, I would like to praise and thank God, the almighty, who has granted countless blessing, knowledge and opportunity to me, so that I have been finally able to accomplish the thesis **ELHAMDULILLAH** .

To my great parents, who never stop giving of themselves in countless ways.

To my dearest grandmother who leads me through the valley of darkness with light of hope and support .

To my beloved brothers and sister,particularly my little brother **"LOKMAN"** .

To my lovely friend **"FERIEL"** who has supported me throughout the process.

Abstract

Topic modelling is a probabilistic generative model that has been applied on text mining and information retrieval. Recent topic models achieve good performances when dealing with English long texts. However, the subject is still challenging with short text especially with non-English short texts due to many factors like sparsity. In this work, We conduct a study on the application of recent topic models dedicated to English long-text on short Arabic texts, providing a comparaison between these models and their performances. We explain the main differences between recent algorithms, providing intuitions on how they operate under the hood, and explaining preprocessing requirements for each algorithm. In addition, we provide their tuning and comparatively evaluate their performance on clustering short-text from many short Arabic datasets.

ملخص:

نمذجة الموضوع هي نموذج توليدي احتمالي تم تطبيقه على التنقيب عن النص واسترجاع المعلومات. تحقق نماذج الموضوعات الحديثة أداءً جيدًا عند التعامل مع النصوص الإنجليزية الطويلة. ومع ذلك ، لا يزال الموضوع يمثل تحديًا مع النص القصير خاصةً مع النصوص القصيرة غير الإنجليزية بسبب العديد من العوامل مثل التباين. في هذا العمل ، نجري دراسة حول تطبيق نماذج الموضوعات الحديثة المخصصة للنص الإنجليزي الطويل على النصوص العربية القصيرة ، مما يوفر مقارنة بين هذه النماذج وأدائها. نفسر الاختلافات الرئيسية بين الخوارزميات الحديثة ، ونقدم الحدس حول كيفية عملها تحت الغطاء ، وشرح متطلبات المعالجة المسبقة لكل خوارزمية. بالإضافة إلى ذلك ، نقدم ضبطها ونقيم أدائها نسبيًا على تجميع نص قصير من العديد من مجموعات البيانات العربية القصيرة.

Résumé

Topic Modeling est un modèle génératif probabiliste qui a été appliqué à l'exploration de texte et à la recherche d'informations. Les topic models récents obtiennent de bonnes performances lorsqu'il s'agit de textes longs en anglais. Cependant, le sujet reste difficile avec des textes courts, en particulier avec des textes courts non anglais en raison de nombreux facteurs tels que la rareté. Dans ce travail, nous menons une étude sur l'application des topic modeling récentes dédiées aux textes longs anglais sur des textes courts arabes, en proposant une comparaison entre ces modèles et leurs performances. Nous expliquons les principales différences entre les algorithmes récents, en fournissant des intuitions sur la façon dont ils fonctionnent sous le capot et en expliquant les exigences de prétraitement pour chaque algorithme. De plus, nous fournissons leur réglage et évaluons comparativement leurs performances sur le regroupement de texte court à partir de nombreux ensembles de données arabes courts.

Contents

List of Figures

General Introduction

Social media have influenced drastically on modern societies, where billions of people are involved actively in these platforms or applications. Large amounts of textual data is generated daily in form of comments, reviews and short text messages over these platform. The analysis of these texts is crucial to provide better services, information research, advertising and security [21].

Consequently, there is a need for more efficient methods and tools that can aid in detecting and analyzing content in online social networks (OSNs).Furthermore, there is a need to extract more useful and hidden information from numerous online sources that are stored as text and written in natural language within the social network landscape (e.g., Twitter, LinkedIn, and Facebook). It is convenient to employ a natural approach, similar to a human–human interaction, where users can specify their preferences over an extended dialogue [22]. Natural language processing (NLP) is a field that combines the power of computational linguistics, computer science, and artificial intelligence to enable machines to understand, analyze, and generate the meaning of natural human speech. The first actual example of the use of NLP techniques was in the 1950s in a translation from Russian to English that contained numerous literal transaction misunderstandings [21]. Essentially, keyword extraction is the most fundamental task in several fields, such as information retrieval, text mining, and NLP applications, namely, topic detection and tracking [22].In this dissertation, we focused on the topic modeling (TM) task, which was described by Miriam (2012) as a method to find groups of words (topics) in a corpus of text. In general, the procedure of exploring data to collect valuable information is stated as text mining. Text mining includes data mining algorithms, NLP, machine learning, and statistical operations to derive useful content from unstructured formats such as social media textual data.

We will see the first chapter some of ANLP challenges and to present relation between machine learning and the field of Arabic natural language processing (ANLP). In the second chapter,will talk about Topic Modeling and we will conduct a comprehensive review of various short text topic modeling techniques under the three categories of methods. In the third chapter,We will see exactly how the work is pass with arabic datasets and all the work steps. Finally , we will show and discuss our implementation and the final results.

Chapter 1

Arabic Natural Language Processing

1.1 Introduction

By definition, Natural language processing (NLP) is a field of computer science that seeks to create concepts, find methods, and construct software that is able to comprehend, study, and produce natural human languages to enable human-machine interaction through writing and/or speech. That is, NLP helps computers identify the ways in which humans use language. Arabic Natural Language Processing (ANLP) is the application of NLP techniques and methods on Arabic Language which is spoken in middle east countries and north Africa. Recently, ANLP has received more attention, and several applications have been developed including text categorization, web page spam detection, and sentiment analysis that supports Arabic language. In This chapter, we discuss the characteristics and complexity of the Arabic language in addition to the importance and needs of ANLP.

1.2 Arabic language

Arabic is an Afro-Asiatic language that developed in the Middle East. More than 250 million individuals speak the Arabic language across the world [3]. The Arabic language was widely disseminated after the emergence of Islam, although it existed centuries before the religion. As a universal religion, Islam has delivered the Arabic language to its followers, estimated to be 1.5 billion people [3]. Historically speaking, Arabic is rooted in Classical Arabic (CA), which has been used as the Arab peoples' native language since 600 AD. It is associated with Islam and the Quran. However, over the centuries, the language has evolved and been simplified to create what is known as Modern Standard Arabic (MSA). The terminology and the linguistic features of MSA differ from those of CA, but the structure of words and sentences have remained. In addition to CA and MSA, each region has a dialect of Arabic spoken in the community (between friends and family) [15].

1.3 Arabic language characteristics

The Arabic language consists of grammar, spelling, punctuation marks, slang as informal language, idioms, and pronunciation. Many characteristics make the Arabic language distinctive [17]:

- Reading and writing in Arabic moves from right to left.
- The language consists of 28 characters.
- In Arabic, upper and lower cases are not distinguished, like Chinese, Japanese, and Korean.
- Numbers are divided into plural, dual, and singular, with two genders feminine and masculine.
- The language comprises several words formed from roots, and several root words are composed of three letters.
- Verbs in the past tense are identified by suffixes, and verbs in the present or future tenses are designated by prefixes; for example, " ذهبت" means "she went," but " تذهب" means "she goes."

1.4 Arabic language complexity

Arabic can be considered more complex than English. Arabic writing does not possess vowels; rather, diacritics are placed above or below letters. Modern writers have abandoned these diacritics; readers are expected to understand the lost diacritics based on their knowledge of the language . This characteristic induces both structural and lexical ambiguity in Arabic texts because various diacritics may lead to different meanings .

Another complexity consists of dots, which are frequently used in Arabic. The structure of many letters is similar or even identical, so letters are differentiated by the number of dots and their locations, such that the letters all have the same structure but with different dot locations and numbers.

In addition, some letters possess diverse forms that rely on their location in the word. Of the 28 letters in the language, 22 take four different shapes each (at the beginning/middle/end of the word, and at a non-linked letter).

Moreover, nouns and adjectives in Arabic can be masculine or feminine .

Furthermore, vocabulary of the Arabic language can have different meanings. For example, the word darkness has 52 synonyms, the word rain has 34, the word moon has 16, the word light has 21, the word short has 164, the word long has 91, and there are 50 synonyms for the word cloud. Arabic also uses specific inflections; usually, a term may be stated as a mix of prefix (es) (which can be articles, prepositions, or conjunctions), lemma, and suffix (es) (which are objects or personal/possessive anaphora) .Figure 1 is an example of Arabic inflection [2].

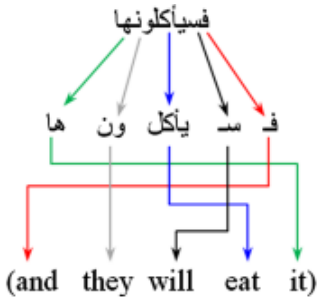

Figure 1.1: Example of inflection [2].

1.5 Arabic natural language processing

1.5.1 Natural language processing

NLP is an area of investigation that aims to enable computers to analyze and use original text or human speech as it is spoken to perform valuable applications. NLP researchers seek to identify how humans manipulate language, which helps them develop techniques and tools that allow computer systems to handle natural languages and perform necessary tasks. Several disciplines have formed the basics of NLP, such as artificial intelligence, computer and information sciences, linguistics, electronic and electrical engineering, robotics, mathematics, and psychology. NLP has many applications in different domains, such as summarizing natural language texts, machine translation, and language information retrieval, among others. Many NLP applications are now available in many languages such as Chinese, Arabic, and English ...ANLP is an advanced application of AI and machine learning used to understand Arabic language with all the complexity [16].

1.5.2 Arabic natural language processing application

ANLP has become an exciting research domain. It involves the development of techniques and tools using the Arabic language. Numerous existing systems have been created for different applications such as machine translation, information retrieval and extraction, localization, and multilingual information retrieval systems . These applications encounter numerous intricate problems related to the structure and nature of the Arabic language [17].

1.5.3 Necessity and needs in ANLP

Most developed ANLP systems are dedicated to allowing non-Arabic speakers in the Western world to understand Arabic texts. For example, sentiment analysis, machine translation, and Arabic named entity recognition are the most commonly used ANLP tools .

ANLP is highly demanded in several sectors. For example, the task of correctly identifying Arabic names is challenging for non-Arabic government institutions. ANLP tools that scan

and recognize names, places, and dates are becoming necessary and beneficial because they save time spent waiting for language experts to carry out the task.

On the other hand, Google Translate and Babylon are always looking for translating English sentences into Arabic (or vice versa) by giving the meaning of each word without considering the meaning of the whole sentence.

ANLP applications are useful in accomplishing the task of sentiment analysis; they can be used worldwide to identify the sense of words in addition to morphological structure. They are also used at the sentence level to recognize the sense of whole phrases [3].

1.6 Machine Learning based ANLP process and evolution

The development of the machine learning applications often includes many phases,as shown in Figure 2 and presented in detail in the following subsections.

1.6.1 Corpora

The first step in Machine learning-based ANLP studies is data collection. These data are text samples that are suitable for the concerned subject area. Few freely available collected and classified Arabic corpora exist, such as Al-Nahar, Al-Jazeera, Al Ahram, Al-hayat, and Al-Dostor newspapers , Hadith corpus , Akhbar-Alkhaleej corpus , Arabic NEWSWIRE , Quranic Arabic Corpus , corpus Watan-2004 , KACST Arabic corpus, BBC Corpus , CCN Corpus and Open Source Arabic Corpora (OSAC) , NADA corpus. These corpora have been used in Arabic processing studies, especially in text categorization. For named entity recognition studies, benchmark data exist such as ACE 2003 and ANERcorp . On the other hand, for the other applications like spam detection, sentiment analysis, readability, and web document classification, benchmark corpora are still missing [17].

1.6.2 Preprocessing

Text preprocessing is a core natural language processing task. It applies operations to create another form from the inputted text. In ANLP, MADAMIRA and RapidMiner offer natural language processing operations. MADAMIRA provides the study of the structure of words and part of words (root words, prefixes, and suffixes) in the Arabic language . It includes the critical characteristics of the well-known Arabic processing systems MADA and AMIRA . RapidMiner consists of many preprocessing operations including stemming, cleaning, and visualization. It is implemented using Java and can be operated with any operating system . Data Cleaning, normalization, tokenization, and stemming are common text preprocessing operations in most ANLP applications .

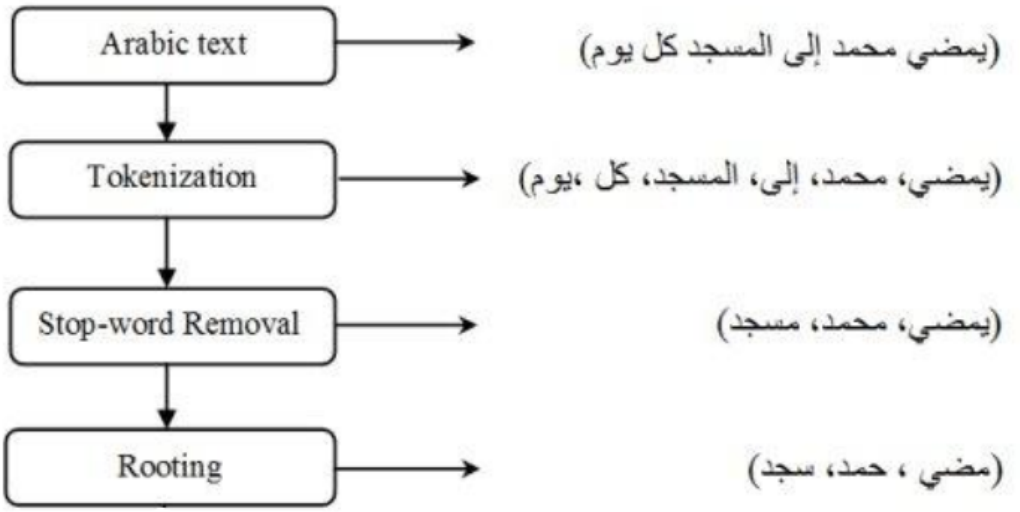

Figure 1.2: Summary of the preprocessing operations [17].

1.6.2.1 Data Cleaning:

According to Oueslati , Arabic text preprocessing is an essential step in any ANLP application. Preprocessing Arabic text on social media, which is usually informal (not standard), is more complicat edowing to several reasons such as the presence of dialect text, common spelling mistakes, extra characters, diacritical marks, and elongations.Consequently, to preprocess such Arabic text, we must perform additional processing such as stripping the elongations, diacritical marks, and extra characters. Additionally, we must convert the text into its normal Arabic form. To this end, this approach provides two preprocessing modules:

a- Text cleaning: Most Arabic text on social media contains noise such as elongations,diacritical marks, extra symbols, and/or mixed language. The first and the most important step in preprocessing is cleaning Arabic text by removing such noises.Most of the previous algorithms clean the noises by expecting all possible noises and then searching each noise in the text and cleaning it,which makes the cleanliness of the text depend on the degree of anticipation of noises. This method of cleaning may not provide accurate results because it is not easy to expect all noises, in addition to the possibility of introduction of new noises. Our cleaning algorithm works differently than the previous algorithms—it does not look at the noises but only selects the required text [2].

```
S= a set of Arabic Characters
Select Text
Cleaning Text=" "
While not last of the Text do
Read the Text character by character
        If the character not in S then
                New character = " "
        Else
                New character = old character
        End if
        Cleaning Text = Cleaning Text & new character
Loop
```

Figure 1.3: Cleaning algorithm [18].

As shown in Figure 3, the cleaning algorithm reads the text, character by character, and then checks whether the character belongs to the Arabic alphabet or the required character; if yes, the algorithm selects it,otherwise, replaces it with a space. This method of cleaning eliminates all noises, such as additional characters, samples, non-Arabic characters, URLs, and any shapes, and provides clean Arabic text without affecting its meaning or content.Moreover, it transforms every letter into its standard form, e.g. "alif" has several forms "أ" and "ا" . Given this, the module provides the possibility of choosing additional characters if required.According to Jianqiang and Xiaolin , removing stop words,numbers, and URLs is appropriate for reducing noise and does not affect the performance of sentiment analysis, which we think is more sensitive to these components than the remaining processing algorithms such as information extracting and mining algorithms.

b- Text normalization: After cleaning the tweet text, the second step is to change the text into its normal form. Some Arabic words on social media are written in non-standard ways, e.g., some words contain repeated characters such a instead "مبروووووك" of" مبروك ", which means congratulations,emotions such as "هههههههههه" which indicate laughing, and include common spelling mistakes and/or dialects.Most of the previous works did not provide individual algorithms for normalization; they used a single algorithm for both cleaning and normalization. In our approach, we considered normalization to be a different preprocessing task because we used a different approach while cleaning the noises. Our proposed normalization algorithm replaces a non-normal word by a normal one by removing the repeated characters and using a set of common non-normal words [23].

1.6.2.2 Tokenization

According to Faragly and Shalan , tokenization in Arabic is problematic owing to the complexity of the structure of the Arabic language,e.g., an Arabic name has no capitalization and may have a middle token such as " بن", some Arabic words may have up to three different tokens. Based on the cleaning and normalization results provided by the algorithms in figure 4 , the tokenization algorithm can be implemented easily and in a straightforward manner.The tokenization algorithm uses the spaces between words and punctuation, such as stop marks, commas, and semicolons to fragment text into words. Then, it stores each word in an individual database field [17].

1.6.2.3 Stemming and root generation:

There are several Arabic stemming algorithms. According to Sawalha , each stemmer is based on different text corpora, and most of them are designed to work for MSA; therefore, the application of any one of them in our approach may be difficult.This subsection proposes a special stemmer/root generating algorithm to handle the text words that result from the above stage (the tokenization stage). Certain words that result from applying the previous algorithm may contain dedicated words or concatenated letters or maybe written in an informal manner, e.g., in some words, "و" which means and, is connected to the word without using the space character.

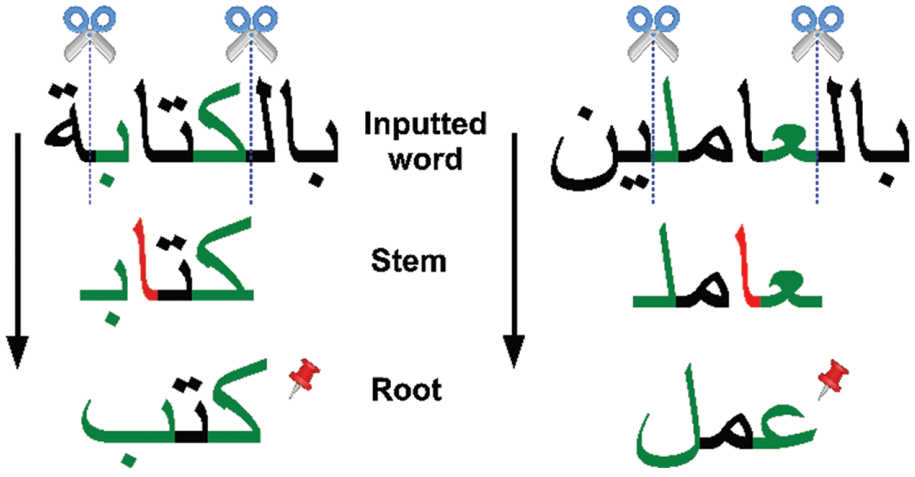

Figure 1.4: Stemming and root word example [17].

1.6.2.4 Stemming vs Lemmatization

Stemming and Lemmatization are text normalization techniques within the field of Natural language Processing that are used to prepare text, words, and documents for further processing. In this blog, you may study stemming and lemmatization in an exceedingly practical approach covering the background, applications of stemming and lemmatization, and the way to stem and lemmatize words, sentences and documents using the Python nltk package which is the natural language package provided by Python.

Stemming is the process of producing morphological variants of a root/base word. Stemming programs are commonly referred to as stemming algorithms or stemmers. For example: " كتب " would be the stem for " يكتب " ," مكتبة "
in contrast to stemming, **lemmatization**looks beyond word reduction and considers a language's full vocabulary to apply a morphological analysis to words. The lemma of " يكون " is " كان " 'be' and the lemma of " نساء" is "مرأة".

1.6.3 Supervised machine learning

Supervised learning approaches are related to a labeled training data. Several types of supervised classifiers exist.the most frequently used classifiers in ANLP applications [13]:

• SUPPORT VECTOR MACHINES (SVM): Are well-known supervised machine learning techniques .SVM have been effectively employed in many problems related to pattern recognition in fields such as bioinformatics and biometrics. Regarding text processing, SVMs have achieved the best results in text categorization and are used extensively in NLP-related problems in different languages such as Arabic for methods like readability prediction and sentiment analysis

• NAIVE BAYES CLASSIFICATION (NB):The most straightforward and the second-most-used classifier. The Naive Bayes classifier is a prevalent technique for text categorization that assigns documents to associated categories such as spam or legitimate and positive, negative, or neutral .

• DECISION TREES: Have been used in many NLP-related classification problems . In addition to the Naive Bayes classifier, Decision Tree classifiers provide excellent results for spam detection . The decision tree classifier is a favored machine learning technique because the model is easy to understand. Abdallah showed that their hybrid approach with the rule-based approach using decision tree performed better than the existing classifiers for Arabic Named Entity Recognition applications.

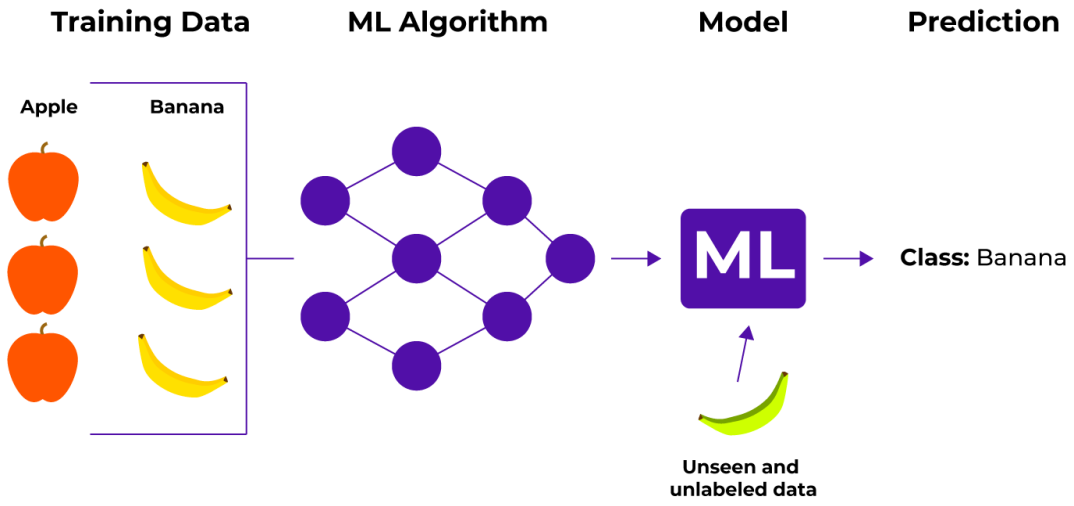

Figure 1.5: supervised learning [13].

1.6.4 Unsupervised Machine Learning

As the name suggests, unsupervised learning is a machine learning technique in which models are not supervised using training dataset. Instead, models itself find the hidden patterns and insights from the given data. It can be compared to learning which takes place in the human brain while learning new things [23].

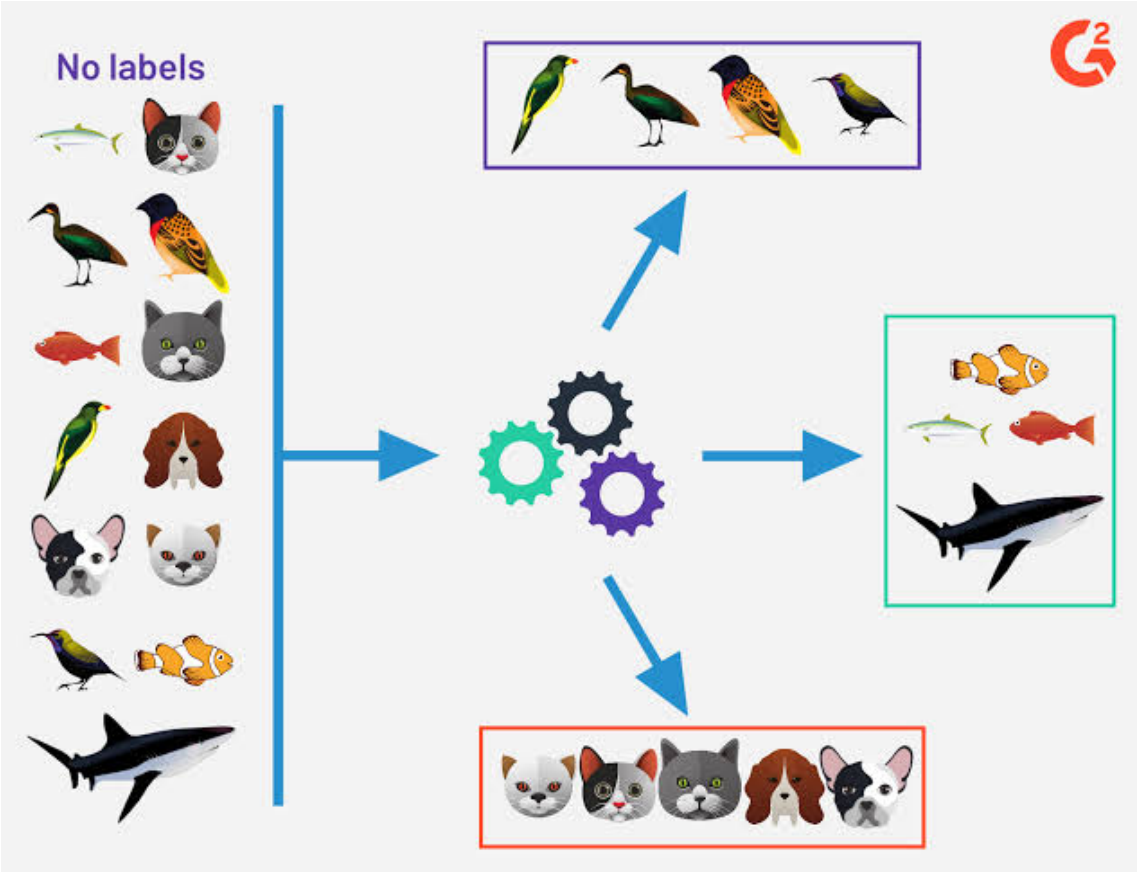

Figure 1.6: Working of unsupervised learning [23].

Here, we have taken an unlabeled input data, which means it is not categorized and corresponding outputs are also not given. Now, this unlabeled input data is fed to the machine learning model in order to train it. Firstly, it will interpret the raw data to find the hidden patterns from the data and then will apply suitable algorithms such as k-means clustering, Decision tree, etc.

Once it applies the suitable algorithm, the algorithm divides the data objects into groups according to the similarities and difference between the objects.

There are a lot of algorithms but the most popular are:

- K-means clustering [34]
- KNN (k-nearest neighbors) [23]
- Hierarchal clustering [29]
- Anomaly detection [23]
- Networks[23]
- Principle Component Analysis [23]

- Independent Component Analysis [23]
- Apriori algorithm [23]
- Topic modeling [23]
- Singular value decomposition [23]

1.7 Conclusion

As we have seen, NLP steps for Arabic are almost the same for English , but with more complexity specially at tokenization, POS tags ,..etc. Also it's more challenging to find tools that support Arabic.

For the next chapter we would find out what is topic modeling and how to apply.

Chapter 2

Topic Modeling

2.1 Introduction

In Natural Language Processing (NLP), topic modeling has become an important research sub-area. It is a text mining technique that identify topics in a set of documents and identify hidden patterns that are present in a text corpus. Topic model serves best for the purpose of organizing huge volume of textual data, retrieval of information from unstructured or semi-structured documents, extraction of feature, and document clustering. It is also a powerful tool and has wide applications in many fields such as historical science, software engineering, linguistic science and Marketing to name few. In this chapter, we present topic modeling concepts and their categories and algorithms.

2.2 Topic Modeling

One of the most extensively application of NLP in the industry for more than a decade is topic modeling. In a nutshell, topic modeling intends to discover latent topics or the global structure of a large, typically unlabeled, set of documents. Especially when dealing with an enormous number of documents, as manually assigning them topics is costly, topic modeling can help gain great insights with less time and effort [8]. Topic modeling is developed based on the assumption that a topic can be interpreted as a probability distribution of a set of words or tokens. Likewise, a document does not fall into a unique topic but rather a statistical mixture of different topics. In other words, topic modeling can be considered as a soft clustering method. It is different from topic clustering in that the hard clustering technique is where a document can only belong to one cluster . Noticeably, both tasks are also different from topic classification, which is a supervised learning one with labeled datasets (in particular, the topic of a document is provided in the first place) [38]. The result from topic modeling is a matrix of documents by topics. The matrix's cell indicates the probability that a particular document belongs to a specific topic. A topic here is not explicitly defined but rather a collection of themed words. Therefore, in the end, topics need to be defined by human intellectuals. The results can be considered as a dimensionality reduction product because the

matrix is a kind of structured data (numerical) of the original unstructured data (texts). More specifically, a unique word needs one dimension to be represented. Therefore, the entire corpus needs millions of dimensions. On the other hand, topic modeling's matrix has significantly fewer dimensions as they represent a defined number of topics rather than numerous single words[38].The advantage of topic modeling is the potential of spotting remarkable hidden topics in the corpus. The downside is that the model performance cannot be precisely evaluated since there are no real targets to categorize documents. Nevertheless, topic modeling is an efficient method as it does not require much computing power. Consequently, it is used widely in NLP, especially in the first stage of analyzing corpora [38].

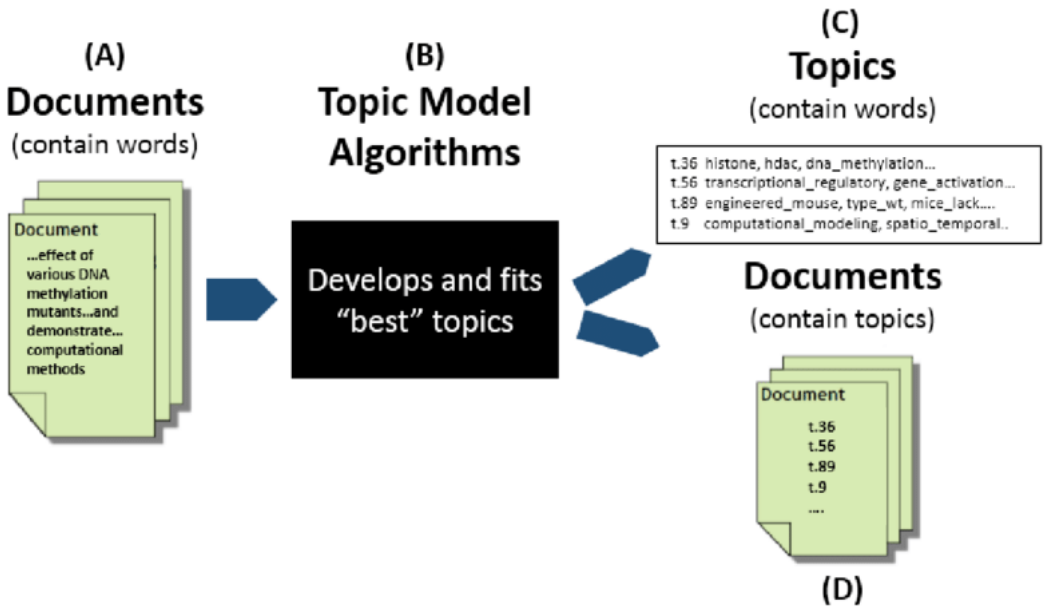

Figure 2.1: Basic Topic Modeling Process [38].

2.3 Topic modeling Application

Topic modeling is actively used in various industries, primarily for automating laborious tasks. For instance, instead of having technicians collect and track customers conversations and reviews of the company or its products over social media, these comments and trends can be automatically detected and classified by applying topic modeling .In particular,in the hospitality industry,by monitoring online reviews using topic modeling, accommodation owners can understand key drivers that affect customer satisfaction to improve them .Similarly,a different study explores patients' thoughts of physicians to help doctors acquire a deeper understanding to enhance healthcare quality.Moreover,topic modeling can assist in routing customer support tickets based on subjects, keywords, urgency, and in delivering them to the proper departments.Thus, the technicians benefit from the ease of manual workloads, which allow them to focus on more critical tasks and complicated problems.Hence, they can provide better services and products and help improve business results [15]. Extensively, topic modeling can be ap-

plied in other industries as well. For instance, chemistry-related topic modeling is developed to categorize large molecules into "chemical topics" and illustrate their similarities. Adopting the same approach, in biology, the technique groups different cell types into diverse "chromatin topics" successfully. Furthermore,topic modeling is employed in new technologies like blockchains to improve the analysis of its technological trend.Finally, topic modeling's outputs can also be used as the inputs for topic classification [7]. In summary,topic modeling's powerful impact on analyzing corpus makes it an attractive tool to computer engineers and business owners.In the last decades,this research section has witnessed a revolutionary transformation thanks to the novel approach: using probabilistic and ANN/DL instead of linear algebra based methods to train topic models [7].

2.4 Topic modeling for short text

In recent years, short texts are increasingly prevalent due to the explosive growth of online social media. For example, about 500 million tweets are published per day on Twitter1 , one of the most popular online social networking services. Probabilistic topic models are broadly used to uncover the hidden topics of tweets, since the low-dimensional semantic representation is crucial for many applications, such as product recommendation , hashtag recommendation , user interest tracking , sentiment analysis. However, the scarcity of context and the noisy words restrict LDA and its variations in topic modeling over short texts [6].

2.5 Difficulties of STTM [1]

Topic modeling is notoriously more challenging to do when the text is shorter and the main reasons why TM difficulty on short are [38]:

1. No common definition of what short-form text is:
    - Social media (Tweets, Reddit posts, and comments, Facebook posts, YouTube video comments...all of which vary greatly in their length limits)
    - Instant messages
    - Short message exchanges
    - Public forum comments
    - News headlines
2. Lack of context:each short text lacks enough word co occurrence information.
3. Lack of context:each short text lacks enough word co occurrence information.
4. Need for extensive data pre-processing
5. Due to a few words in each text, most texts are probably generated by only one topic

[1] STTM: Short Text Topic Modeling

2.6 Classification of STTM

The short text topic modeling algorithms basically fall into the following three major categories:

- Dirichlet multinomial mixture,
- Global word co-occurrences,
- Self-aggregation.

2.6.1 Dirichlet multinomial mixture

Dirichlet multinomial mixture (DMM) is a generative topic model with the assumption that each document covers only a single topic. Actually, this assumption can indirectly enrich word co-occurrences at the document level, making the model more effective for short texts [49].

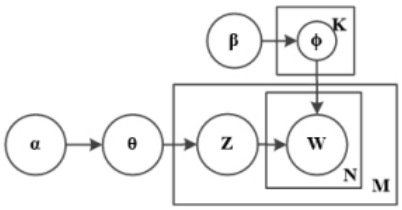

Figure 2.2: The DMM model generation process [49].

- M: The total documents in the corpus.
- N: The number of words in the document.
- w: The Word in a document.
- z: The latent topic assigned to a word.
- β and α:Dirichlet parameters.
- θ:The topic distribution.
- ϕ:The word distribution for topic k

2.6.2 Global word co-occurrences

Global word co-occurrences (GW) is model for distributed word representation, The model is an unsupervised learning algorithm for obtaining vector representations for words. This is achieved by mapping words into a meaningful space where the distance between words is related to semantic similarity.Training is performed on aggregated global word-word co-occurrence statistics from a corpus, and the resulting representations showcase interesting linear substructures of the word vector space. It is developed as an open-source project at Stanford and was launched in 2014. As log-bilinear regression model for unsupervised learning

of word representations, it combines the features of two model families, namely the global matrix factorization and local context window methods [33].

Some models try to use the rich global word co-occurrence patterns for inferring latent topics,Due to the adequacy of global word co-occurrences, the sparsity of short texts is mitigated for these models.According to the utilizing strategies of global word co-occurrences [33]
.

2.6.3 Self-aggregation

The success of topic modeling on social media through heuristic aggregation of tweets has shed light on how to develop a generalized topic modeling solution for short texts .Motivated by this, a natural strategy would be to resort to automatic clustering algorithms for aggregating short texts into long pseudo documents, before a standard topic model is applied. A Self-Aggregation (SA) topic model is proposed to aggregate short texts for data sparsity. However, its number of parameters increase with the data size, being prone to overfitting and computationally expensive [43].

2.7 STTM techniques

Figure 4.15 depicts the previous classification of Short Text Topic Models. In the next subsections, we present the algorithms of each category.

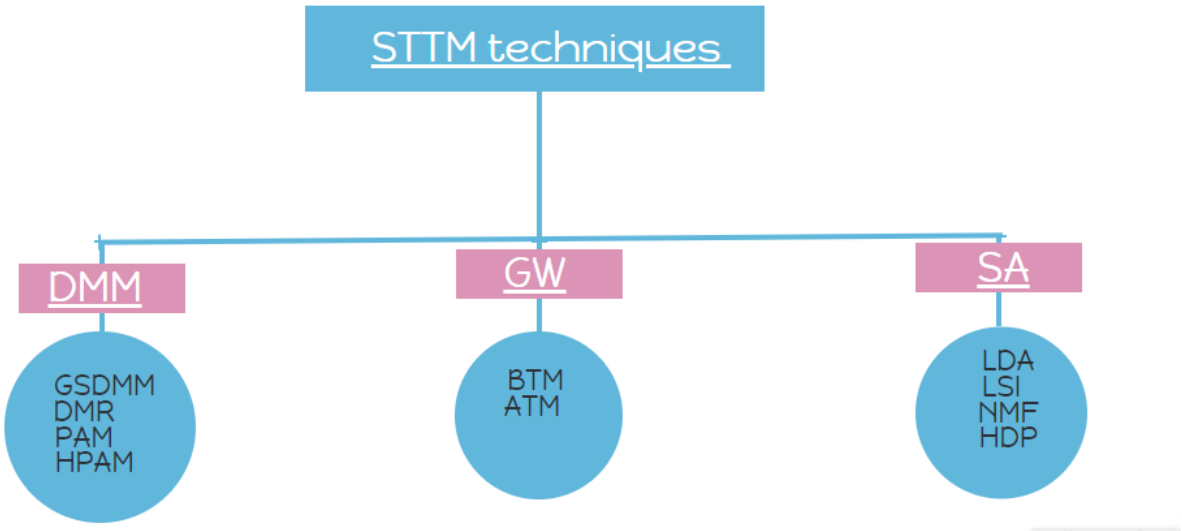

Figure 2.3: STTM techniques.

2.7.1 Dirichlet multinomial mixture algortithms

2.7.1.1 Gibbs Sampling Dirichlet multinomial mixture

GSDMM is a short text clustering model,GSDMM is essentially a modified Latent Dirichlet Allocation which assumes that a document (tweet or text for instance) encompasses 1 topic

this differs from LDA which assumes that a document can have multiple topics. The basic principle of GSDMM is described using an analogy called "Movie Group Approach". Imagine a group of students (documents) who all have a list of favorite movies (words). The students are randomly assigned to K tables. At the instruction of a professor the students must shuffle tables with 2 goals in mind: 1) Find a table with more students 2) Pick a table where your film interests align with those at the table. Rinse and repeat until you reach a plateau where the number of clusters does not change [36].

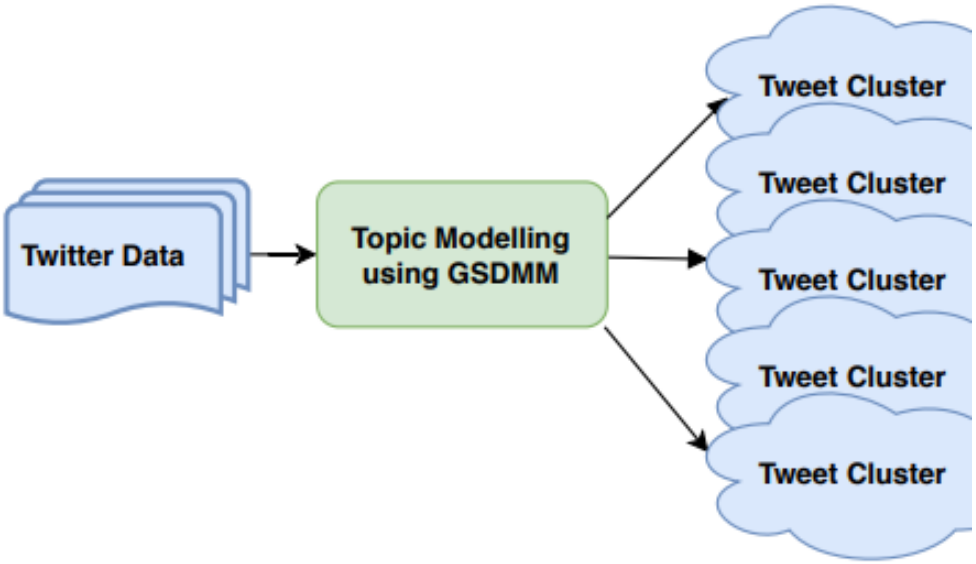

Figure 2.4: GSDMM model [44]

2.7.1.2 Pachinko allocation model(PAM)

This model is structured as a directed acyclic graph (DAG) where leaf nodes are words in the vocabulary of the corpus, and interior nodes are topics which have Dirichlet distributions over their child nodes. Some implementations of PA include many layers of interior nodes/topics, which allow for more granular examination of the distributions of words and topics across other topics. Tomotopy library has an implementation of Pachinko Allocation supporting 2 layers of topic nodes. The tomotopy implementation requires 2 parameters, k1 and k2 which correspond to the number of nodes at the first and second levels of the DAG. [25].

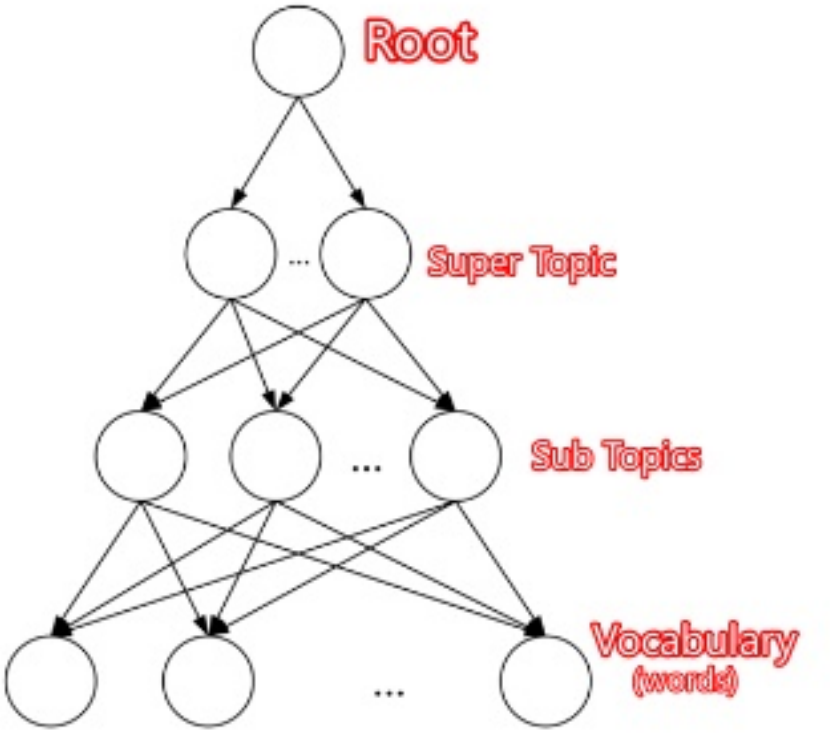

Figure 2.5: PAM architecture [25].

2.7.1.3 Hierarchical Pachinko allocation model

Hierarchical Pachinko allocation (HPAM) is a special type of PAM model that creates a hierarchy of topics, but since PAM uses a DAG instead of a tree, child topics can have multiple parent topics. HPAM does this by giving each node at every level have a Dirichlet distribution over the vocabulary itself rather than only the lowest level of topics having a distribution over the vocabulary. The Dirichlet distributions at interior nodes are critical to HPAM because their parameters represents the hierarchy through the parameters of these Dirichlet distributions

Tomotopy has an implementation of HPAM, which takes the same paramters as PAM, k1k1 and k2k2 which correspond to the number of nodes at the first and second levels of the DAG [25].

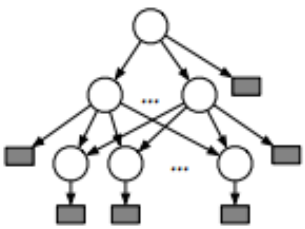

Figure 2.6: HPAM architecture [25].

2.7.1.4 The Dirichlet-multinomial regression

DMR topic model is a powerful method for rapidly developing topic models that can take into account arbitrary features. It can emulate many previously published models, achieving similar or improved performance with little additional statistical modeling or programming work by the user [28]. One interesting side effect of using the DMR model is efficiency. Adding additional complexity to a topic model generally results in a larger number of variables to sample and a more complicated sampling distribution [28].

2.7.2 Global word co-occurrences algortithms

2.7.2.1 Biterm topic model(BTM)

The key idea of BTM is to learn topics over short texts based on the aggregated biterms in the whole corpus to tackle the sparsity problem in single document. Specifically, we consider that the whole corpus as a mixture of topics, where each biterm is drawn from a specific topic independently. The probability that a biterm drawn from a specifictopic is further captured by the chances that both words in the biterm are drawn from the topic [46].

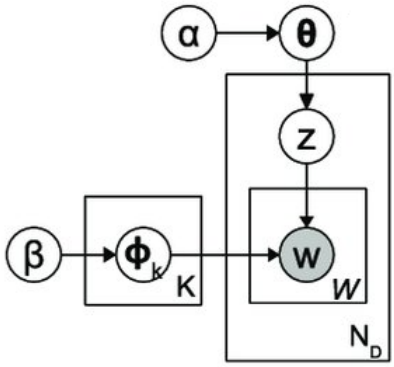

Figure 2.7: BTM model [46].

- M: The total documents in the corpus.
- N: The number of words in the document.
- w: The Word in a document.
- z: The latent topic assigned to a word.
- β and α:Dirichlet parameters.
- θ:The topic distribution.
- ϕ:The word distribution for topic k

2.7.2.2 Advesarial-neural Topic Model

They propose (ATM) as shown in Figure below . The proposed ATM contains three main components: (1) the document sampling module shown at the top of Figure below, which defines the representation mapping function and samples a real document dr ∈ R V from an input text corpus; (2) the generator G takes a topic distribution θ sampled from a dirichlet prior as input and generates the corresponding fake document df ; (3) the discriminator D takes df and dr as input and discriminates the fake document from the real ones, whose output is subsequently used as a learning signal to update the parameters of G and D [45].

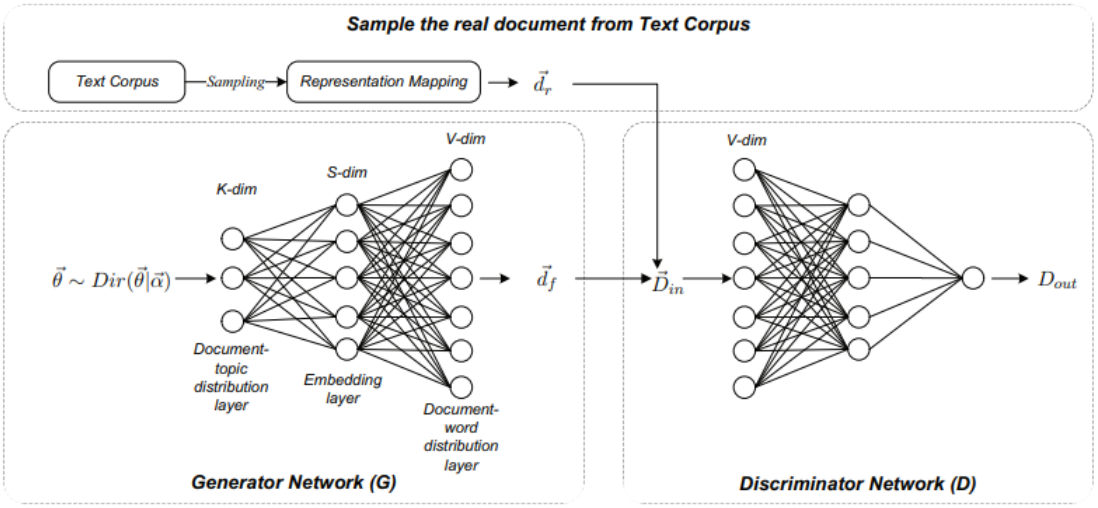

Figure 2.8: The framework of the Adversarial-neural Topic Model (ATM) [45].

2.7.3 Self-aggregation algortithms

2.7.3.1 Latent Dirichlet Allocation(LDA)

LDA is a generative probabilistic model of a corpus. The basic idea is that the documents are represented as random mixtures over latent topics, where a topic is characterized by a distribution over words. Latent Dirichlet allocation (LDA), first introduced by Blei, Ng and Jordan in 2003 [11], is one of the most popular methods in topic modeling. The words with highest probabilities in each topic usually give a good idea of what the topic is can word probabilities from LDA. It is one of the most popular topic modeling methods. Each document is made up of various words, and each topic also has various words belonging to it. The aim of LDA is to find topics a document belongs to, based on the words in it.

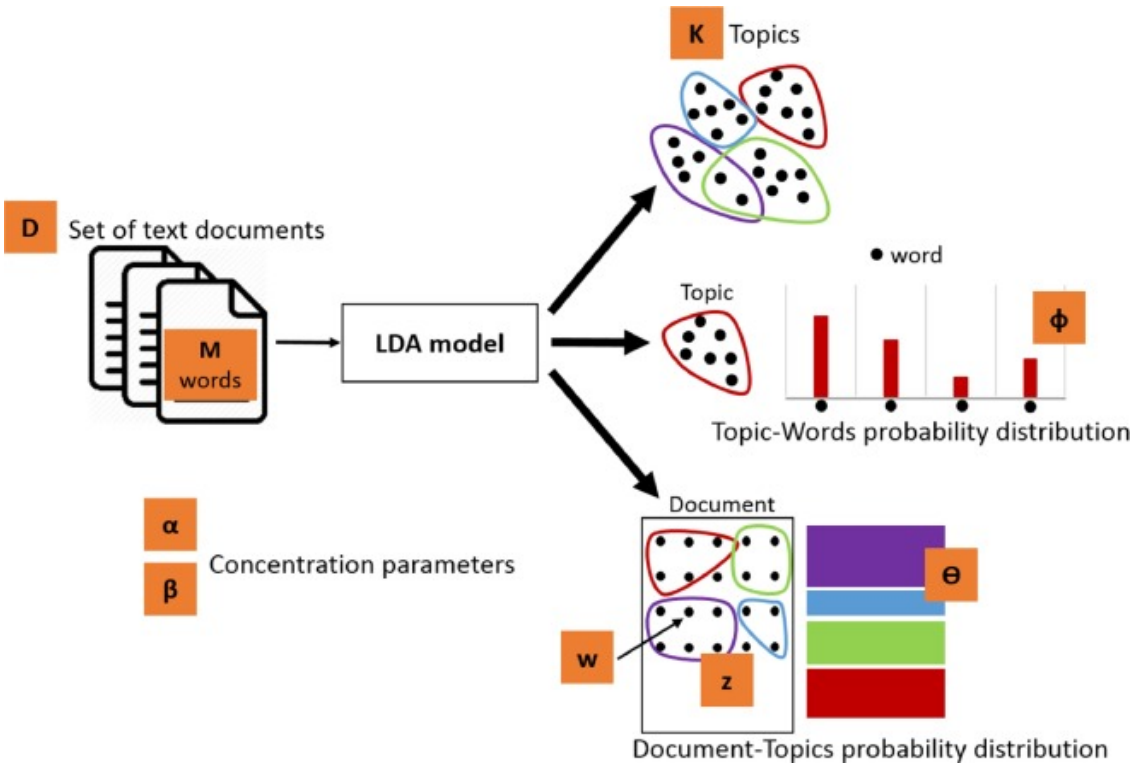

Figure 2.9: LDA model [11].

2.7.3.2 Non-negative Matrix Factorization

NMF is a statistical method to reduce the dimension of the input corpora. It uses factor analysis method to provide comparatively less weightage to the words with less coherence [24]. For a general case, consider we have an input matrix V of shape m x n. This method factorizes V into two matrices W and H, such that the dimension of W is m x k and that of H is n x k. For our situation, V represent the term document matrix, each row of matrix H is a word embedding and each column of the matrix W represent the weightage of each word get in each sentences (semantic relation of words with each sentence). You can find a practical application with example below [24].

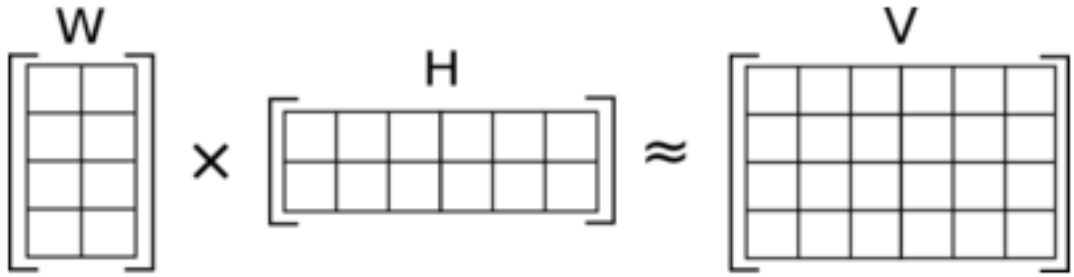

Figure 2.10: NMF example [24].

2.7.3.3 Latent Semantic Indexing(LSI)

LSI is one of the foundational techniques in topic modeling. The core idea is to take a matrix of what we have[documents] and [terms] and decompose it into a separate [document][topic] matrix and a [topic][term] matrix[35].

Figure 2.11: LSI matrix [35].

The first step is generating our document-term matrix. Given m documents and n words in our vocabulary, we can construct an m $\times$ n matrix A in which each row represents a document and each column represents a word. In the simplest version of LSA, each entry can simply be a raw count of the number of times the j-th word appeared in the i-th document. In practice, however, raw counts do not work particularly well because they do not account for the significance of each word in the document. For example, the word "nuclear" probably informs us more about the topic(s) of a given document than the word "test."[35] [10].

2.7.3.4 Hierarchical Dirichlet process

HDP is a powerful mixed-membership model for the unsupervised analysis of grouped data. Applied to document collections, the HDP provides a nonparametric topic model where documents are viewed as groups of observed words, mixture components (called topics) are distributions over terms, and each document exhibits the topics with different proportions. Given a collection of documents[31]. HDP is a revamped version of LDA, which can infer the number of topics through the learning phase, on the contrary of choosing the number of topics initially as the above algorithms. More advanced than the LDA formula, HDP has an appended preceding level of Dirichlet processes for producing a non-parametric prior for the number of topics hence, the hierarchical word in its name. The method is beneficial considering that it might be tricky to foresee how many topics are in the unknown, enormous corpus. A variant of this is online variational inference for the HDP. It overcomes some drawbacks of the previous

one, as this enhanced algorithm can easily handle huge datasets and streaming texts like web APIs . Because HDP is relatively new and has not been widely used yet [31].

2.8 Conclusion

In this chapter, we presented an overview about Topic Modeling and its algorithms with more focus on short texts. At the best of our knowledge the aforementioned techniques are applied on English text and just a few of them were applied to Arabic short Texts [19, 26]. In the next chapter we will present our application of all the previous techniques on Arabic short texts.

Chapter 3

Topic Models for Arabic Short Text

3.1 Introduction

As mentioned in the previous chapters, topic modeling has been proven to be successful in summarizing long texts and extracting important concepts and topics from them. However, the need to analyze short texts became significantly relevant as the popularity of microblogs, such as Twitter, grew. The challenge with inferring topics from short text is that it often suffers from noisy data, so it can be difficult to detect topics in a smaller corpus. In this chapter, we present our work on the application of a set topic modeling methods on Arabic short text. To the best of our knowledge, These methods are applied only in the context of the English language as we have presented in the previous chapter. we apply these methods a on a collection of Arabic short texts that we have compiled from different sources. Different coherence metrics are used to compare between their performances.

3.2 Related Studies

The origin of topic modeling approach goes back to the 1990s when Latent Semantic Analysis(LSA),also known as Latent Semantic Indexing (LSI).Afterward, another approach called Probabilistic Latent Semantic Analysis (PLSA), based on the likelihood principles, that defines a generative data model and also minimizes word perplexity, was proposed to discover the latent topic in document collection which was a significant contribution to enhance topic modeling methods .

In 2003, LDA was presented which outperforms PLSA and LSA by its ability to reduce the dimensionality and also can be applied in complex methods.Many variations of LDA have been suggested to take advantage of auxiliary data related to the corpus to enhance the topics quality. For example, in ([41],[50]), with the idea that a word has adifferent sentiment in different contexts.

DMM has been proved to handle the problem of a severe sparsity of short texts. The authors in [27] investigate the performance of LDA and DMM on short and long texts.They used Gibbs sampling for DMM (GSDMM)proposed in [48].They conclude that LDA outperforms DMM

on long texts and GSDMM more performance on short texts. The work in topic modeling on Arabic content is almost limited to the works that have been listed in [37] a survey paper in which the authors investigate the recent significant published articles related to topic detection and topic modeling on Arabic content.
the hidden topics of the Quran have been uncovered using LDA. Each surah has been treated as a document. The authors confirm that the main themes for each surah have been revealed and the two major topics of the Quran with different themes in the Madani and Makki surahs have been recognized.
In the domain of Arabic, content in social media referred to as short texts [5] The framework consists of five stages: "data collection, preprocessing, classification, clustering, and summarization." They confirm that LDA has a poor performance in their case because of the shortness of the tweets.

3.3 Methodology

Figure 4.15 depicts the followed steps to conduct our comparative analysis. Each step in the process is described in details in the next subsections.

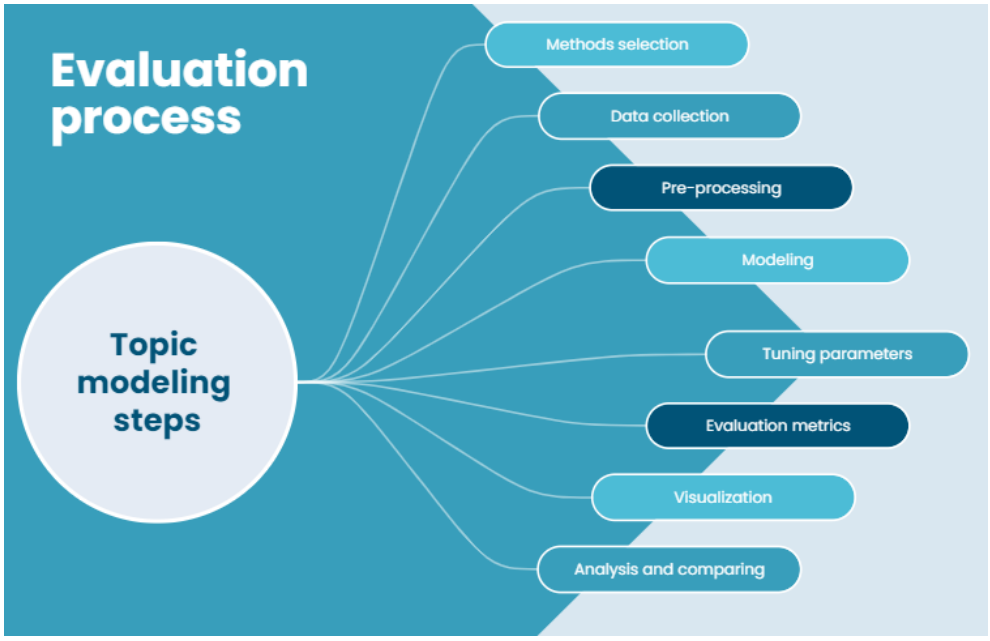

Figure 3.1: Topic Modeling Steps

3.3.1 Methods selection

As we have mentioned in the previous chapter , we divide the short text topic modeling algorithms basically into the three major categories; Dirichlet multinomial mixture (DMM), Global word co-occurrences (GW), and Self-aggregation(SA). For our analysis, we select all the described methods as competitive methods.

3.3.2 Dataset Collection

A dataset in machine learning is quite simply a collection of data pieces that can be treated by a computer as a single unit for analytic and prediction purposes.High-quality datasets are the key to good performance in natural language processing (NLP) projects. for this study, we employ three datasets [1] ,each one containsa large curated base for training.

3.3.3 Pre-processing

Data-preprocessing can be defined as the process of "cleaning and transformation" of the data, data which can be in any format may be structured, unstructured or semi-structured and has been extracted or originated from different sources like historical data, stream-data, application-data etc [47] . there is steps need to be carried out for Data-preprocessing:

3.3.3.1 Importing Relevant Libraries

The very first step for any data pre-processing using python is importing all the relevant libraries according to your problem statements. The most commonly used or we can say basic libraries are pandas for importing and exporting the datasets, numpy for mathematical calculations and sklearn which is one of the most powerful library used for data pre-processing [47].

3.3.3.2 Loading the Datasets

The next step after importing the libraries is loading the datasets. Using python we can read different data format files like image files, coma separated values (csv files), Excel files, Text files, HTML, Hierarchical Data format, audio files, video files and many more [47].

3.3.3.3 Tokenization

Tokenization is a step which splits longer strings of text into smaller pieces, or tokens. Larger chunks of text can be tokenized into sentences, sentences can be tokenized into words, etc. Further processing is generally performed after a piece of text has been appropriately tokenized. Tokenization is also referred to as text segmentation or lexical analysis. Sometimes segmentation is used to refer to the breakdown of a large chunk of text into pieces larger than words (e.g. paragraphs or sentences), while tokenization is reserved for the breakdown process which results exclusively in words[47].

3.3.3.4 Handling the Null-values

The most crucial step in data pre-processing is to handle the missing values or null values in python we say NaN values. Simply we can handle the null values by: Dropping the entire row containing the null values, which sometimes tends to result unpredictable result and is not the best way to handle missing values[12].

[1] https://metatext.io/datasets-list/arabic-language

3.3.3.5 Removal of stop words

Stop words are available in abundance in any human language. By removing these words, we remove the low-level information from our text in order to give more focus to the important information. In order words, we can say that the removal of such words does not show any negative consequences on the model we train for our task [12].

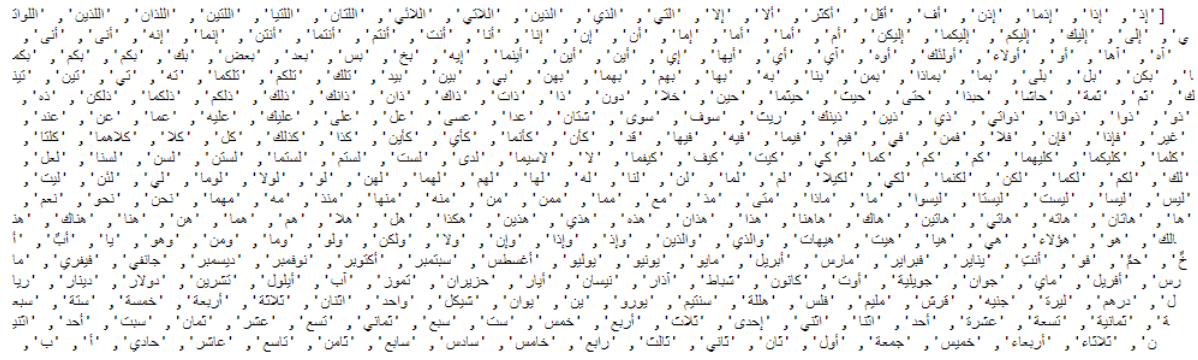

Figure 3.2: Arabic stop words

3.3.3.6 Removal of punctuation

An important NLP preprocessing step is punctuation marks removal, this marks used to divide text into sentences, paragraphs and phrases affects the results of any text processing approach, especially what depends on the occurrence frequencies of words and phrases,it typically do not add extra meaning to the text [12].

3.3.3.7 Bag of words

Bag of words is a Natural Language Processing technique of text modelling. In technical terms, we can say that it is a method of feature extraction with text data. This approach is a simple and flexible way of extracting features from documents [30] .

A bag of words is a representation of text that describes the occurrence of words within a document. We just keep track of word counts and disregard the grammatical details and the word order. It is called a “bag” of words because any information about the order or structure of words in the document is discarded. The model is only concerned with whether known words occur in the document, not where in the document.

One of the biggest problems with text is that it is messy and unstructured, and machine learning algorithms prefer structured, well defined fixed-length inputs and by using the Bag-of-Words technique we can convert variable-length texts into a fixed-length vector [12] . Let’s see this example:

- Sentence :” welcome great learning now start learning”

Word	Frequency
welcome	1
great	1
learning	2
now	1
start	1
good	0
practice	0

Figure 3.3: Bag of Word example

The output is this vector: Sentence [1,1,2,1,1,0,0]

3.3.3.8 Term frequency-inverse document frequency

TF-IDF is a statistical measure that evaluates how relevant a word is to a document in a collection of documents. It has many uses, most importantly in automated text analysis, and is very useful for scoring words in machine learning algorithms for Natural Language Processing (NLP) [12].

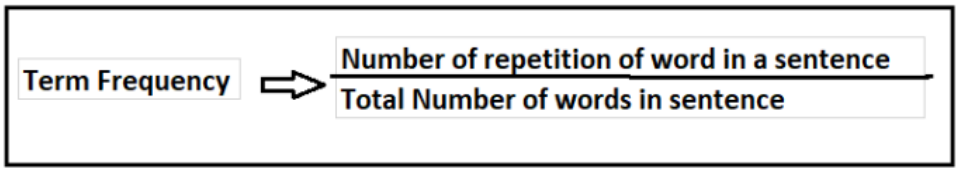

Figure 3.4: TF formula

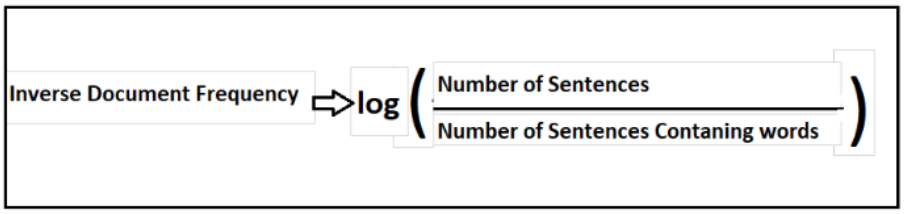

Figure 3.5: IDF formula

TF-IDF was invented for document search and information retrieval. It works by increasing proportionally to the number of times a word appears in a document, but is offset by the number of documents that contain the word. So, words that are common in every document,

such as this, what, and if, rank low even though they may appear many times, since they don't mean much to that document in particular [4].

Let's check an example:

Sentences
eat veg
eat nonveg
eat food

Words	Frequency
eat	3
veg	1
nonveg	1
food	1

		Term Frequency	
Sentences **Words**	**eat veg**	**eat nonveg**	**eat food**
eat	1/2	1/2	1/2
veg	1/2	0/2	0/2
nonveg	0/2	1/2	0/2
food	0/2	0/2	1/2

Figure 3.6: Example of calculating Tf

Words	Inverse Document Frequency
eat	log(3/3) = 0
veg	log(3/1) = 0.477
nonveg	log(3/1) = 0.477
food	log(3/1) = 0.477

Figure 3.7: Example of calculating IDF

	Feature1	Feature2	Feature3	Feature4	O/P
Sentences	eat	veg	nonveg	food	
eat veg	1/2*0	1/2*0.477	0/2*0.477	0/2*0.477	
eat nonveg	1/2*0	0/2*0.477	1/2*0.477	0/2*0.477	
eat food	1/2*0	0/2*0.477	0/2*0.477	1/2*0.477	

Figure 3.8: Calculate TF-IDF

3.3.3.9 Removal of emojis

The emojis are removed since they can't be analyzed.moreover, not all emojis convey the message. Many of them are meaningless. we should remove it as they are not providing any helpful information [12].

Figure 3.9: Emojis

3.3.3.10 Lemmatization

Lemmatization is a linguistic term that means grouping together words with the same root or lemma but with different inflections or derivatives of meaning so they can be analyzed as one item. The aim is to take away inflectional suffixes and prefixes to bring out the word's dictionary form. In lemmatization, the transformation uses a dictionary to map different variants of a word back to its root format [30].

3.3.4 Modeling

In this part, we have applied the techniques on the dataset, we present the full details in the next chapter.

3.3.5 Tuning parameters

Algorithms	Parameter	Best_velue
LDA	Corpus Num_topics id2word	Processed_docs 10 Dictionary
LSI	Corpus Num_topics id2word	Processed_docs 10 Dictionary
HDP	Corpus id2word	Processed_docs Dictionary
DMR	TermWeight (TW) K(the number of topics) Corpus	The default value is TermWeight.ONE 10 Processed_docs D
NMF	Corpus Num_topics id2word	Processed_docs 5 Dictionary
ATM	Corpus Num_topics id2word	Processed_docs 4 Dictionary
HPAM	**k1** **K2**	The number of super topics the number of sub topics
PAM	**k1** **K2**	The number of super topics the number of sub topics
GSDMM	Nbr clusters Alpha(controlling a documents affinity for a larger cluster) Beta(controlling a documents affinity for a more similar cluster) Iteration	K=20 Alpha=0.1 Beta=0.1 n_iters=1

Figure 3.10: Parameter Selection

3.3.6 Evaluation Metrics

The evaluation of machine learning techniques often relies on accuracy scores computed comparing predicted results against a ground truth. In the case of unsupervised techniques like topic modeling, the ground truth is not always available. For this reason, in the literature, we can find metrics which enable to evaluate a topic model independently from a ground truth, among which, coherence measures are the most popular ones for topic modeling [40].

3.3.6.1 Coherence metrics

The coherence metrics rely on the joint probability P(wi, wj) of two words wi and wj that is computed by counting the number of documents in which those words occur together divided by the total number of documents in the corpus. The documents are fragmented using sliding windows of a given length, and the probability is given by the number of fragments including both wi and wj divided by the total number of fragments. This probability can be expressed through the Pointwise Mutual Information (PMI) [14] ,defined as:

$$PMI(w_i, w_j) = log \frac{P(w_i, w_j) + \epsilon}{P(w_i) \cdot P(w_j)}$$

Figure 3.11: Coherence formula

A small value is chosen for 'ϵ', in order to avoid computing the logarithm of 0.Different metrics based on PMI have been introduced in the literature, differing in the strategies applied for token segmentation, probability estimation, confirmation measure, and aggregation.

• The UCI coherence averages the PMI computed between pairs of topics, according to:

$$C_{UCI} = \frac{2}{N \cdot (N-1)} \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} PMI(w_i, w_j)$$

Figure 3.12: UCI Coherence formula

• The UMASS coherence relies instead on differently computed joint probability:

$$C_{UMASS} = \frac{2}{N \cdot (N-1)} \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} log \frac{P(w_i, w_j) + \epsilon}{P(w_j)}$$

Figure 3.13: UMASS Coherence formula

• The Normalized Pointwise Mutual Information (NPMI) applies the PMI in a confirmation measure for defining the association between two words:

$$NPMI(w_i, w_j) = \frac{PMI(w_i, w_j)}{-log(P(w_i, w_j) + \epsilon)}$$

Figure 3.14: NMPI Coherence formula

3.3.7 Analysis and comparaison

After we calculate all the coherence, we have set the results in table for make the comparison more easy and visible as depicted in Figures (4.15, 3.17,

	Model	Score_by_CV	Score_by_uMass	Score_by_cuci	Score_by_cnpmi
7	HPAM	0.919085	-12.476921	-6.902184	0.072879
8	DMR	0.914398	-19.290410	-13.772321	-0.364590
5	PAM	0.877856	-15.735913	-12.624725	-0.252541
2	HDP	0.814996	-22.766987	-17.670157	-0.629957
4	ATM	0.736028	-22.795574	-13.223506	-0.472107
0	LDA	0.640115	-20.336094	-16.870939	-0.546090
6	GSDMM	0.563951	-19.160797	-15.752256	-0.490611
1	LSI	0.436042	-10.701486	-9.450320	-0.076327
3	NMF	0.435451	-14.301026	-11.624023	-0.208031

Figure 3.15: Results of Dataset 1

	Model	Score_by_CV	Score_by_uMass	Score_by_cuci	Score_by_cnpmi
8	DMR	0.648829	-0.328794	0.279165	0.015349
6	GSDMM	0.639117	-3.719139	-7.056772	-0.086978
3	NMF	0.622987	-2.834385	-9.760540	-0.226133
5	PAM	0.616775	-7.773416	-5.305575	-0.123359
7	HPAM	0.612837	-0.212452	-0.072143	-0.031144
2	HDP	0.604061	-19.964997	-17.684793	-0.619468
1	LSI	0.597249	-4.115930	-8.845057	-0.176683
0	LDA	0.396314	-6.283976	-9.474097	-0.244678
4	ATM	0.373339	-20.310427	-8.873530	-0.317331

Figure 3.16: Result of dataset 2

	Model	Score_by_CV	Score_by_uMass	Score_by_cuci	Score_by_cnpmi
7	HPAM	0.685053	-0.499438	0.612756	0.058021
8	DMR	0.669697	-0.310306	0.271987	0.012294
5	PAM	0.637953	-8.600523	-5.762147	-0.164395
6	GSDMM	0.635397	-3.678383	-7.530143	-0.123860
3	NMF	0.614532	-3.317835	-9.615589	-0.221437
2	HDP	0.594216	-19.794679	-17.734180	-0.621466
1	LSI	0.555494	-5.540306	-10.239105	-0.241033
4	ATM	0.452084	-21.278247	-9.598314	-0.344493
0	LDA	0.354132	-9.451742	-11.031946	-0.329501

Figure 3.17: Result of dataset 3

These results are visualized and analysed in the next chapter.

3.4 Conclusion

In this chapter, we have presented a general methodology by which we apply several STTM methods from different categories on a dataset of Arabic short texts. The process steps were explained in details. In the next chapter, we present the practical aspect of the study and the analysis of the obtained results.

Chapter 4

Implementation and Results

4.1 Introduction

The process of comparing several topic models for Arabic short texts was presented in the previous chapter. In this chapter, we presents the practical aspects of the project, where we give an overview about tools, techniques and languages used for the implementation of the application of the different topic models on the Arabic dataset. Moreover, we present the obtained results and a comparison between the algorithms [1].

4.2 Language and tools

The problem and the purpose in this project are always creating a comparison between results of few known algorithms in topic modelling and evaluating them using some techniques in NLP and hyperparameter tuning to obtain the best results , in this case, the first programming language used is Python and a bunch of followed packages each one had particular roles and benefits, meanwhile, the text editor is Jupyter Notebook because of many advantages and particular strong benefits :

- They're great for showcasing your work. You can see both the code and the results. The notebooks at Kaggle is a particularly great example of this.
- It's easy to use other people's work as a starting point. You can run cell by cell to better get an understanding of what the code does.
- Very easy to host server side, which is useful for security purposes. A lot of data is sensitive and should be protected, and one of the steps toward that is no data is stored on local machines. A server-side Jupyter Notebook setup gives you that for free.

[1] `https://github.com/oumaima19992022/PFEE`

Jupyter Notebook has installed with Anaconda program which helped us to work with more flexibility and well organised

4.2.1 Anaconda

Anaconda is a distribution of the Python and R programming languages for scientific computing (data science, machine learning applications, large-scale data processing, predictive analytics, etc.), that aims to simplify package management and deployment [1].

Figure 4.1: Anaconda Logo

4.3 PC Performance

PC Performance depend on what size of dataset it is that you are going to be work. In our experiment I used my PC with :

- CPU:Intel(R) Core(TM) i3-6006U CPU 2.00GHz 2.00 GHz.
- RAM:4,00 Go
- Hard disk: 500 GO HDD + 250 GO SSD

4.3.0.1 Anaconda Navigator

Anaconda Navigator is a desktop graphical user interface (GUI) included in Anaconda distribution that allows users to launch applications and manage conda packages, environments and channels without using command-line commands. Navigator can search for packages on Anaconda Cloud or in a local Anaconda Repository, install them in an environment, run the packages and update them. It is available for Windows, macOS and Linux [1].

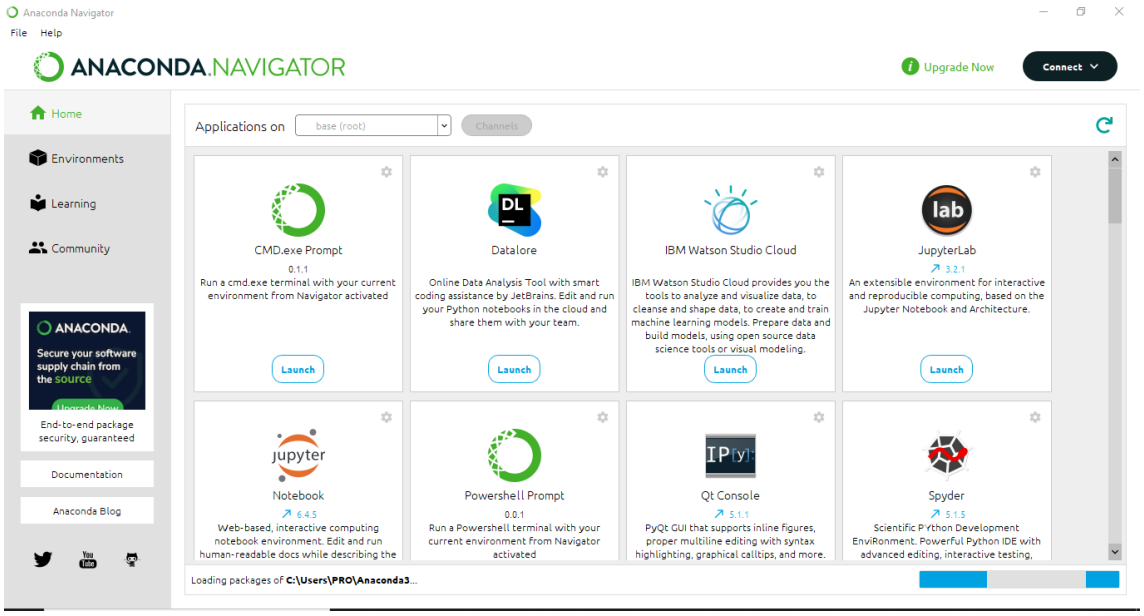

Figure 4.2: Anaconda Navigator

The following applications are available by default in Navigator:

- JupyterLab
- **Jupyter Notebook**
- QtConsole
- Spyder
- Glue
- Orange
- RStudio
- Visual Studio Code

4.3.1 Python

Python is a high-level programming language used for complex scenarios as well as a general-purpose language that is used across various domains. It is the favourite language among the developers because of its simplicity and less complex syntax. It is open-source and available for all the operating systems and is platform-independent and has an extensive library for Python programming code [42].

Figure 4.3: Python Logo [42].

4.4 Packages

In [2]:
```
import pandas as pd
import numpy as np
import warnings
warnings.filterwarnings("ignore")
from pprint import pprint
import pip
import setuptools
import tmplot as tmp
import gensim
from gensim.utils import simple_preprocess
from gensim.parsing.preprocessing import STOPWORDS
from nltk.stem import ISRIStemmer
from nltk.stem.porter import *
import nltk
from nltk.corpus import stopwords
from gensim.models import CoherenceModel
import gensim.corpora as corpora
from gensim.models import CoherenceModel
from sklearn.feature_extraction.text import CountVectorizer, TfidfTransformer;
from sklearn.decomposition import NMF;
from sklearn.preprocessing import normalize;
from bidi.algorithm import get_display
import arabic_reshaper
from wordcloud import WordCloud
from gensim.models.nmf import Nmf as GensimNmf
from gsdmm import MovieGroupProcess
from gensim.models import AuthorTopicModel
import tomotopy as tp
import string
from gensim import models
from statistics import mean
import bitermplus as btm
from gensim.models import LdaSeqModel
import qalsadi.lemmatizer
from gensim.models import HdpModel
import matplotlib.pyplot as plt
import pyLDAvis
import pyLDAvis.gensim_models
import seaborn as sns
import plotly.graph_objs as go
import plotly.offline as py
```

Figure 4.4: Python Packages

4.4.1 Pandas

Pandas is an open source, BSD-licensed library providing high-performance, easy-touse data structures and data analysis tools for the Python programming language.

Figure 4.5: Pandas Logo

4.4.2 NumPy

NumPy is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more.

Figure 4.6: NumPy Logo

4.4.3 Gensim

Gensim is a Python library for topic modelling, document indexing and similarity retrieval with large corpora. Target audience is the natural language processing (NLP) and information retrieval (IR) community [39].

Figure 4.7: Gensim Logo [39].

4.4.4 NLTK

NLTK is a leading platform for building Python programs to work with human language data. It provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries, and an active discussion forum [9].

4.4.5 Scikit-learn

Scikit-learn (also known as sklearn) is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support-vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy. Scikit-learn is a NumFOCUS fiscally sponsored project [32].

Figure 4.8: Sklearn Logo[32].

4.4.6 Tomotopy

Tomotopy is a Python extension of tomoto (Topic Modeling Tool) which is a Gibbs-sampling based topic model library written in C++. It utilizes a vectorization of modern CPUs for maximizing speed. The current version of tomoto supports several major topic models including.

Figure 4.9: Tomotopy Logo.

4.4.7 Matplotlib

Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. Matplotlib produces publication-quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python

scripts, the Python and IPython shell, web application servers, and various graphical user interface toolkits [20] .

Figure 4.10: Matplotlib logo [20].

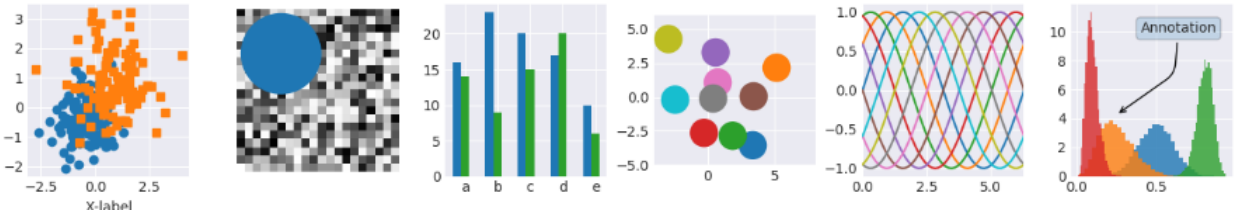

Figure 4.11: style sheet example

4.5 Experiment description

4.5.1 Load required packages

Depending on my choice of python notebook, I need to install and load the following packages to perform topic modeling

```
In [1]: # Installations

        !pip install pandas
        !pip install numpy
        !pip install gensim
        !pip oinstall nltk
        !pip install matplotlib.pyplot
        !pip install qalsadi
        !pip install pyLDAvis
        !pip install tomotopy
        !pip install git+https://github.com/rwalk/gsdmm.git
        !pip install tmplot
        !pip install arabic_reshaper
```

Figure 4.12: Installing Packages

4.5.2 Dataset

Topic modeling is an 'unsupervised' machine learning technique but all the Arabic short texts dataset has class column so we need to edit it .

	text	sentiment
0	أهنئ الدكتور أحمد جمال الدين، القيادي بحزب مصر...	POS
1	البرادعي يستقوى بامريكا مرةاخرى و يرسل عصام ال...	NEG
2	#ليلة_الاتحادية# الحرية_والعدالة	شاهد الأن: ...	OBJ
3	الوالده لو اقولها بخاطري حشيشة تضحك بس من اقول...	NEUTRAL
4	#انتخبوا_العرص #انتخبوا_البرص #مرسى_رئيسى #بن...	NEUTRAL
...	...	...
9688	والغاز مش مدعوم يا إنسان؟ وماذا عن الأسمدة وال...	NEG
9689	اغلاق كل الساحات والميادين الكبرى لمنع صلاة ال...	NEG
9690	#الشروق "الداخلية": 400 ألف مواطن تقدموا لأداء...	OBJ
9691	#بحبك هبحبك_لو صحتها من النوم علشان تقولها بحبك ;)	POS
9692	كل شى كتبته غلط كل شى حسبته غلط فى #الامتحانات	NEG

9693 rows × 2 columns

```
count_sentiment = data.sentiment.value_counts()
count_sentiment
```

```
OBJ        6469
NEG        1642
NEUTRAL     805
POS         777
Name: sentiment, dtype: int64
```

Figure 4.13: Dataset Printing

Figure 4.14 shows how the class column deleted:

```
In [6]: data.drop(['sentiment'], axis=1)
```

Out[6]:

	text
0	أهنئ الدكتور أحمد جمال الدين، القيادي بحزب مصر...
1	البرادعي يستقوى بامريكا مرةاخرى و يرسل عصام ال...
2	#الحرية_والعدالة	شاهد الآن: #ليلة_الاتحادية ...
3	الوالده لو اقولها بخاطري حشيشة تضحك بس من اقول...
4	#انتخبوا_العرص #انتخبوا_البرص #مرسى_رئيسى #ابن...
...	...
9688	والغار مش مدعوم يا إنسان؟ وماذا عن الأسمنت وال...
9689	اغلاق كل الساحات والميادين الكبرى لمنع صلاة ال...
9690	الشروق "الداخلية": 400 ألف مواطن تقدموا لأداء...
9691	#بحبك_لو صحتها من النوم علشان تقولها بحبك ;) هتح...
9692	كل شى كتبته غلط كل شى حسبته غلط فى #الاتحادات

9693 rows × 1 columns

Figure 4.14: New dataset shape

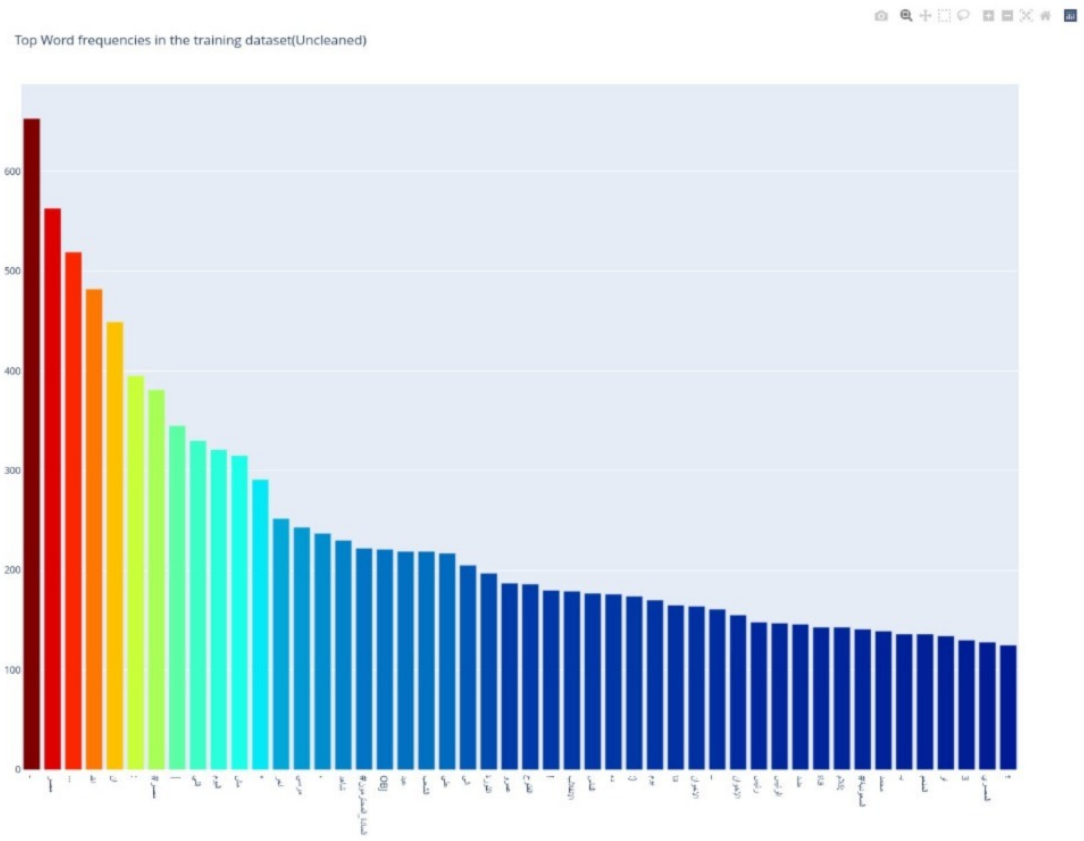

Figure 4.15: Top Word frequencies in the training dataset(Uncleaned)

4.5.3 Data Pre-processing

Our goal in the data pre-processing stage is to convert sentences into words, convert words to their root and removing words that are too common or too irrelevant to the purpose of our topic modeling project,We used the following techniques to reach our goal:

4.5.3.1 Removing punctuation

Figure 4.16 depicts the step of punctuation removing.

```
In [9]: print(string.punctuation)

        !"#$%&'()*+,-./:;<=>?@[\]^_`{|}~

In [10]: new_string = data['text'].str.translate(str.maketrans('', '', string.punctuation))
         print(new_string)

         أهنئ الدكتور أحمد جمال الدين، القيادي بحزب مصر...        0
         البرادعي يستقوى بامريكا مرةاخرى و يرسل عصام ال...        1
         الحريةوالعدالة  شاهد الآن لبلةالاتحادية أول في...        2
         الوالدة لو اقولها بخاطري حشيشة تضحك بس من اقول...        3
         ö  انتخبواالعرص انتخبواالبرص مرسرئيسى اينرئيسى           4
                                                   ...
         والغاز مش مدعوم يا إنسان؟ وماذا عن الأسمنت وال...     9688
         اغلاق كل الساحات والميادين الكبرى لمنع صلاة ال...     9689
         الشروق الداخلية 400 ألف مواطن تقدموا لأداء الح...     9690
          هتحبكلو صحتها من النوم علشان تقولها بحبك             9691
         كل شى كتبته غلط كل شى حسبته غلط فى الامتحانات         9692
         Name: text, Length: 9693, dtype: object
```

Figure 4.16: Removing punctuation

4.5.3.2 Removing stop words

Stop words are removed as depicted in Figure 4.17.

Removing stop words

```
: data['text'] = data['text'].apply(lambda x: ' '.join([word for word in x.split() if word not in (stopwords_list)]))
  data
```

Figure 4.17: Removing Stop words

4.5.3.3 Dropping missing values(Texts)

Missing values are then droped using the instruction depicted in Figure 4.18.

Dropping missing values(Texts) in the DataFrame if exist

```
data.isnull().sum()
```

```
text         0
sentiment    0
dtype: int64
```

Figure 4.18: Removing missing values

It can be seen that our dataset does not contain missing values, which leads to pass directly to the next step.

4.5.3.4 Lemmatization and Removing emojis function

In the Figure 4.19 we apply the Lemmatization function for grouping different inflected forms of words into the root form , we use Qalsadi library because it is Arabic Morphological Analyzer and lemmatizer for Python . The code inside the square for removing the existing emojis in our dataset (Figure 4.19).

```
def lemmatize_stemming(text):
    return qalsadi.lemmatizer.Lemmatizer().lemmatize(text)
def preprocess(text):
    result = []
    for token in gensim.utils.simple_preprocess(text):
        if token not in stopwords_list and len(token) > 3:
            emoji_pattern = re.compile("["
            u"\U0001F600-\U0001F64F"  # emoticons
            u"\U0001F300-\U0001F5FF"  # symbols & pictographs
            u"\U0001F680-\U0001F6FF"  # transport & map symbols
            u"\U0001F1E0-\U0001F1FF"  # flags (iOS)
                               "]+", flags=re.UNICODE)
            emoji_pattern.sub(r'', token)
            result.append(lemmatize_stemming(token))
    return result
```

Figure 4.19: Lemmatization function

4.5.3.5 TF-IDF

The output below (Fig. 4.20) represents only TF-IDF for 10 documents from the dataset.

TF-IDF:

```
In [31]: tfidf = gensim.models.TfidfModel(processed_docs)

In [32]: corpus_tfidf = tfidf[processed_docs]

In [33]: type(corpus_tfidf)
Out[33]: gensim.interfaces.TransformedCorpus

In [34]: for doc in corpus_tfidf:
             pprint(doc)
             break

[(0, 0.3168379818467526),
 (1, 0.27296119629219473),
 (2, 0.28390733143407193),
 (3, 0.20524186812088974),
 (4, 0.2345975140729241),
 (5, 0.24276796761263442),
 (6, 0.364539673009944),
 (7, 0.3610900770518482),
 (8, 0.2953256759817791),
 (9, 0.31532053440398466),
 (10, 0.372137730341686)]
```

Figure 4.20: Calculate Tf-Idf

4.5.4 Modeling part

In this section, we are use the topic modeling algorithms and then calculate the corresponding measure "coherence".

4.5.4.1 Latent Dirichlet Allocation Model

The figure (4.21) represents documents with the most important words with their high importance in the document itself, using LDA algorithm in Gensim library.

1- Latent Dirichlet Allocation(LDA)

```
lda_model = gensim.models.LdaMulticore(bow_corpus, num_topics=10, id2word=dictionary, passes=2, workers=2)

for idx, topic in lda_model.print_topics(-1):
    print('Topic: {} \nWords: {}'.format(idx, topic))
```

```
Topic: 0
Words: 0.012*"ناس"*0.004 + "ثورة"*0.004 + "محترم"*0.004 + "سادة"*0.004 + "الاخوان"*0.005 + "يوسف"*0.005 + "بلد"*0.006 + "مرس"*0.006 + "إخوان"*0.008 + "رئيس"
Topic: 1
Words: 0.021*"السيسي"*0.003 + "سلم"*0.004 + "مشروع"*0.004 + "مرس"*0.004 + "عمر"*0.004 + "حب"*0.004 + "شاهد"*0.005 + "حكومة"*0.005 + "كلام"*0.005 + "الله"
Topic: 2
Words: 0.007*"شاهد"*0.004 + "عرب"*0.004 + "هلال"*0.004 + "عمل"*0.004 + "سعودي"*0.004 + "ليل"*0.005 + "الله"*0.005 + "نصر"*0.005 + "جامع"*0.006 + "يوم"
Topic: 3
Words: 0.009*"حر"*0.003 + "شعب"*0.003 + "حياة"*0.004 + "دول"*0.004 + "ناس"*0.004 + "جديد"*0.004 + "رئيس"*0.005 + "يوم"*0.008 + "الله"*0.008 + "ثورة"
Topic: 4
Words: 0.008*"طريق"*0.004 + "مصر"*0.004 + "نفس"*0.004 + "إخوان"*0.004 + "رئيس"*0.004 + "سلط"*0.004 + "مرس"*0.005 + "لي"*0.006 + "يوم"*0.006 + "الله"
Topic: 5
Words: 0.008*"أحيا"*0.004 + "عمر"*0.004 + "يوم"*0.004 + "لي"*0.004 + "السيسي"*0.005 + "سياسة"*0.005 + "دول"*0.005 + "دستور"*0.006 + "نفس"*0.006 + "كلام"
```

Figure 4.21: LDA model

In the figures(4.22, 4.23,4.24,4.25) we calculated the coherence with four types (c-v, U-mass, C-uci,C-nmpi) and we got those results:

LDA cohernece with c_v

```
# Compute Coherence Score with tfidf and u_mass parameter
coherence_model_lda = CoherenceModel(model=lda_model, dictionary=dictionary, coherence='c_v', texts =processed_docs  )
coherence_lda_cv = coherence_model_lda.get_coherence()
print('\nCoherence Score with c_v: ', coherence_lda_cv)
```

```
Coherence Score with c_v:  0.437808650536455
```

Figure 4.22: LDA coherence by CV

LDA Cohernece with C_uci

```
coherence_model_lda = CoherenceModel(model=lda_model, texts=processed_docs, dictionary=dictionary, coherence='c_uci')
coherence_lda_uci = coherence_model_lda.get_coherence()
print('\nCoherence Score: ', coherence_lda_uci)
```

```
Coherence Score:  -2.7158336153871607
```

Figure 4.23: LDA coherence by C-uci

LDA Cohernece with c_npmi

```
#corpus_tfidf
# Compute Coherence Score with tfidf and u_mass parameter
coherence_model_lda = CoherenceModel(model=lda_model, dictionary=dictionary, coherence='c_npmi', texts =processed_docs  )
coherence_lda_npmi = coherence_model_lda.get_coherence()
print('\nCoherence Score with c_npmi: ', coherence_lda_npmi)
```

```
Coherence Score with c_npmi:  -0.10568224740790624
```

Figure 4.24: LDA coherence by C-nmpi

LDA Coherence with u_mass

```
In [38]: # Compute Coherence Score
         coherence_model_lda = CoherenceModel(model=lda_model, dictionary=dictionary, coherence='u_mass',corpus=processed_docs)
         coherence_lda_uMass = coherence_model_lda.get_coherence()
         print('\nCoherence Score with u_mass: ', coherence_lda_uMass)

         Coherence Score with u_mass:  -3.744871885576395
```

Figure 4.25: LDA coherence by U-mass

The last step is to find the optimal number of topics as we show in the figure (4.26). We need to build many LDA models with different values of the number of topics (k) and pick the one that gives the highest coherence value. Choosing a 'k' that marks the end of a rapid growth of topic coherence usually offers meaningful and interpretable topics. Picking an even higher value can sometimes provide more granular sub-topics. If you see the same keywords being repeated in multiple topics, it's probably a sign that the 'k' is too large.

```
plt.figure(figsize = (10,5))
s = pd.Series(results.score.values, index=results.topic.values)
plt.xlabel("number of topics")
plt.ylabel("Score")
_ = s.plot()
```

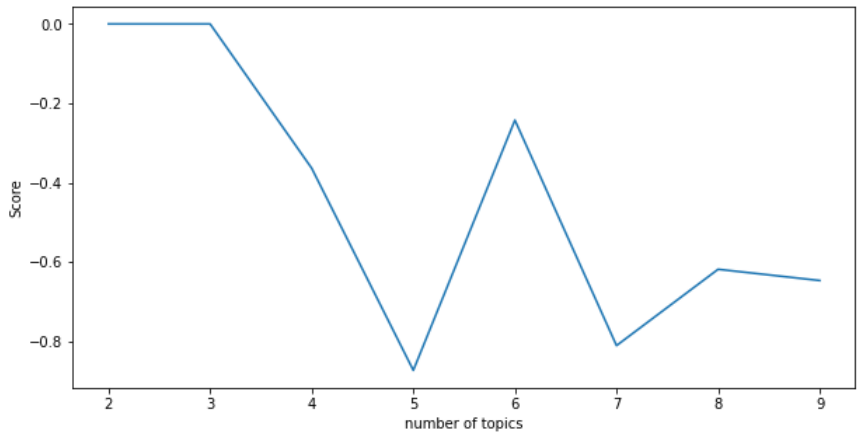

Figure 4.26: Topic score

This figure(4.27) presents pyLDAVis , the most commonly used and a nice way to visualise the information contained in a topic model.

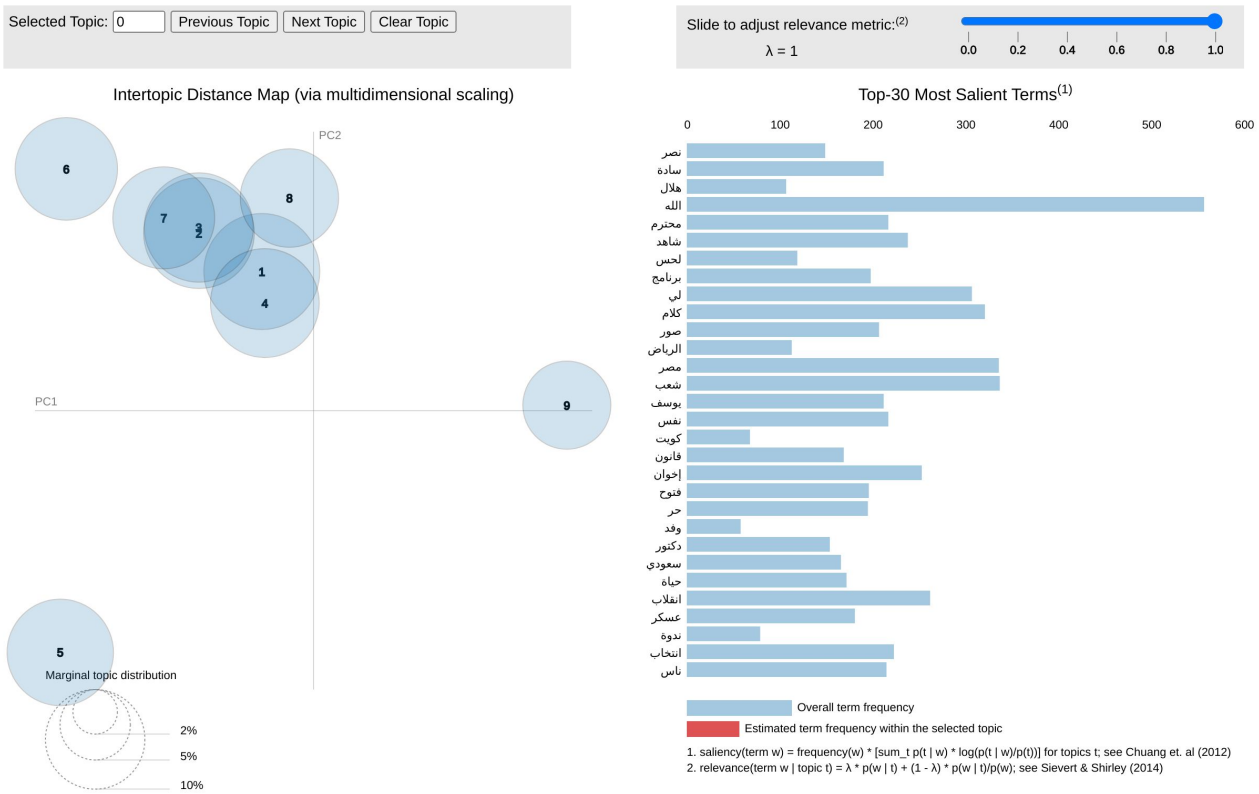

Figure 4.27: LDA visualisation

4.5.4.2 Biterm topic model (BTM)

The figure (4.28) represents how we applied BTM algorithm.

2- Biterm topic model (BTM)

```
X, vocabulary, vocab_dict = btm.get_words_freqs(data['text'])
docs_vec = btm.get_vectorized_docs(data['text'], vocabulary)
biterms = btm.get_biterms(docs_vec)
model = btm.BTM(
    X, vocabulary, seed=12321, T=8, M=20, alpha=50/8, beta=0.01)
model.fit(biterms, iterations=20)
```

```
100%|██████████| 20/20 [00:02<00:00,  8.42it/s]
```

Figure 4.28: BTM model

4.5.4.3 Latent Semantic Indexing (LSI)

The figure (4.29) represents the using of LSI algorithm in Gensim library.

```
lsi_model = gensim.models.lsimodel.LsiModel(
    corpus=bow_corpus, id2word=dictionary, num_topics=10
)
```

Figure 4.29: LSI model

The figure (4.30) represents documents with the most important words with their high importance in the document itself with LSI algorithm .

```
for indx, topics in lsi_model.print_topics(num_topics=5, num_words=10):
    print('Topic: {} \nWords: {}'.format(indx, topics))
```

```
Topic: 0
Words: 0.261*"يوسف"*0.144 + "عمر"*0.152 + "رئيس"*0.155 + "ثورة"*0.158 + "سادة"*0.160 + "محترم"*0.161 + "مصر"*0.165 + "الله"*0.192 + "شاهد"*0.217 + "كلام"
Topic: 1
Words: -0.351*"كوريث"*0.181- + "الله"*0.188- + "كلام"*0.189 + "رئويث"*0.190- + "صور"*0.229- + "سعودي"*0.241- + "الرياض"*0.252- + "متابع"*0.294- + "هلال"*0.320- + "نصر"
Topic: 2
Words: -0.363*"وطن"*0.179 + "شاهد"*0.213- + "حمزاوي"*0.245 + "عمر"*0.246 + "الله"*0.253 + "لحس"*0.255- + "يوسف"*0.269- + "كلام"*0.292- + "محترم"*0.361- + "سادة"
Topic: 3
Words: 0.545*"قناة"*0.065- + "اسم"*0.077- + "إخوان"*0.084- + "ثورة"*0.127 + "لحس"*0.235- + "يوسف"*0.272- + "سادة"*0.297- + "محترم"*0.297- + "شاهد"*0.515 + "كلام"
Topic: 4
Words: 0.681*"رسول"*0.094 + "دستور"*0.107- + "هلال"*0.107- + "متابع"*0.135- + "خير"*0.135 + "وطن"*0.204- + "ذكر"*0.238 + "عمر"*0.276- + "حمزاوي"*0.306- + "الله"
```

Figure 4.30: LSI Topics

4.5.4.4 Hierarchical Dirichlet Process (HDP)

The figure (4.31) represents documents with the most important words with their high importance in the document itself, using HDP algorithm in Gensim library.

4- Hierarchical Dirichlet Process (HDP)

```
hdp_model = HdpModel(bow_corpus,dictionary)
```

```
for idxh, topich in hdp_model.print_topics(num_topics=5, num_words=10):
    print('Topic: {} \nWords: {}'.format(idxh, topich))
```

```
Topic: 0
Words: 0.001*منعم*0.001 + النخاب*0.001 + مصري*0.001 + الي*0.001 + يوم*0.001 + صور*0.001 + القلاب*0.001 + ثورة*0.001 + اسم*0.001 + الله
Topic: 1
Words: 0.001*رابع*0.001 + يوسف*0.001 + القلاب*0.001 + صور*0.001 + كلام*0.001 + الاخوان*0.001 + شعب*0.001 + رئيس*0.001 + مرس*0.001 + الله
Topic: 2
Words: 0.001*ثورة*0.001 + زير*0.001 + أسبوع*0.001 + عالم*0.001 + الي*0.001 + القلاب*0.001 + عسكر*0.001 + سادة*0.001 + محترم*0.001 + مرس
Topic: 3
Words: 0.001*رئيس*0.001 + المتضررين*0.001 + الله*0.001 + حكومة*0.001 + عملاولا*0.001 + لازم*0.001 + إخوان*0.001 + ثورة*0.001 + مصر*0.001 + يوم
Topic: 4
Words: 0.001*الله*0.001 + قرى*0.001 + مبار*0.001 + ناس*0.001 + شاهد*0.001 + ثورة*0.001 + شعب*0.001 + سياسة*0.001 + عسكر*0.001 + مطالب
```

Figure 4.31: HDP model

The figures(4.32,4.33,4.34,4.35,4.36,4.37) represents documents with the most important words with their high importance in the document itself, using NMF,PAM,HPAM,DMR,ATMand GSDMM algorithms .

4.5.4.5 Non-Negative Matrix Factorization(NMF)

```
In [81]:
nmf = GensimNmf(
    corpus=bow_corpus,
    num_topics=5,
    id2word=dictionary

)
```

Figure 4.32: NMF model

4.5.4.6 Pachinko allocation (PAM)

6- Pachinko allocation (PAM)

```
:
k = 5
mdl = tp.PAModel(k1=5, k2=25)
for line in [processed_docs]:
    mdl.add_doc(line.strip().split())

for i in range(10,100,10):
    mdl.train(100)
    print('Iteration: {}\tLog-likelihood: {}'.format( i,mdl.ll_per_word))



for s in range(mdl.k):
    if not mdl.get_sub_topics(s): continue
    print('Top 10 words of topic #{}'.format(s))
    print(mdl.get_topic_words(k,top_n=10))
```

Figure 4.33: PAM model

4.5.4.7 Hierarchical Pachinko allocation (PAM)

7- Hierarchical Pachinko allocation (HPAM)

```
k = 5
mdlH = tp.HPAModel(k1=5, k2=25)

for line in [processed_docs]:
    mdlH.add_doc(line.strip().split())

for i in range(10,100,10):
    mdlH.train(100)
    print('Iteration: {}\tLog-likelihood: {}'.format( i,mdlH.ll_per_word))


for s in range(mdlH.k):
    if not mdlH.get_sub_topics(s): continue
    print('Top 10 words of topic #{}'.format(s))
    print(mdlH.get_topic_words(k,top_n=10))
```

Figure 4.34: HPAM model

4.5.4.8 Dirichlet Multinomial Regression(DMR)

8- Dirichlet Multinomial Regression(DMR)

```
]:
corpus = tp.utils.Corpus()

mdlD = tp.DMRModel(tw=tp.TermWeight.ONE,
    k=20,
    corpus=corpus)

for line in [processed_docs]:
    mdlD.add_doc(line.strip().split())

for i in range(10,100,10):
    mdlD.train(100)
    print('Iteration: {}\tLog-likelihood: {}'.format( i,mdlD.ll_per_word))
```

Figure 4.35: DMR model

4.5.4.9 Adversarial-neural Topic Model(ATM)

9- Adversarial-neural Topic Model(ATM)

```
]: model = AuthorTopicModel(
       corpus=bow_corpus, id2word=dictionary, num_topics=4
   )

]: for indx, topics in model.print_topics(num_topics=5, num_words=10):
       print('Topic: {} \nWords: {}'.format(indx, topics))
```

```
Topic: 0
Words: 0.000*"ماعن"*0.000 + "أضاء"*0.000 + "نأى"*0.000 + "نهب"*0.000 + "دا"*0.000 + "تناقل"*0.000 + "خلط"*0.000 + "اكسترانتشيع"*0.000 + "فرع"*0.000 + "إفريقي"
Topic: 1
Words: 0.000*"مكرم"*0.000 + "الرجاله"*0.000 + "إيهاب"*0.000 + "ألا"*0.000 + "تحدى"*0.000 + "حاسس"*0.000 + "نيتكنه"*0.000 + "لفظ"*0.000 + "اسطه"*0.000 + "فهمي"
Topic: 2
Words: 0.000*"جدا"*0.000 + "تعريف"*0.000 + "بابا"*0.000 + "طبعاً"*0.000 + "وثن"*0.000 + "استطعم"*0.000 + "بروفيلكم"*0.000 + "إشباع"*0.000 + "ديليسيبس"*0.000 + "التلطئين"
Topic: 3
Words: 0.000*"معبر"*0.000 + "اسوان"*0.000 + "اخطار"*0.000 + "الادوية"*0.000 + "تيأسك"*0.000 + "سقراط"*0.000 + "ماذاكرت"*0.000 + "تجاه"*0.000 + "ياموتسى"*0.000 + "ماتعرفش"
```

Figure 4.36: ATM model

4.5.4.10 Short for Gibbs Sampling Dirichlet Multinomial Mixture(GSDMM)

10- Short for Gibbs Sampling Dirichlet Multinomial Mixture(GSDMM)

In [110]:
```
documents = data['text'].values
mgp = MovieGroupProcess(K=20, alpha=0.1, beta=0.1, n_iters=1)

# Transform documents
tokens = [doc.split() for doc in documents]

id2word = gensim.corpora.Dictionary(tokens)

y = mgp.fit(tokens ,len(id2word))
```

```
In stage 0: transferred 8002 clusters with 20 clusters populated
```

In [111]:
```
doc_count = np.array(mgp.cluster_doc_count)
print('Number of documents per topic :', doc_count)
print('*'*20)
# Topics sorted by the number of document they are allocated to
top_index = doc_count.argsort()
print('Most important clusters (by number of docs inside):', top_index)
print('*'*20)
# Show the top 5 words in term frequency for each cluster
for i in range(20):
    print("Cluster",i,':',sorted(mgp.cluster_word_distribution[i].items(), key=lambda item: item[1])[:10])
```

Figure 4.37: GSDMM model

4.5.5 Comparison the performance by score

The figures (4.38,4.39,4.40,4.41) visualized a bar graph of the measures of c-v,u-mass,c-uci,c-nmpi coherences of each model.

4.5.5.1 Using CV coherence

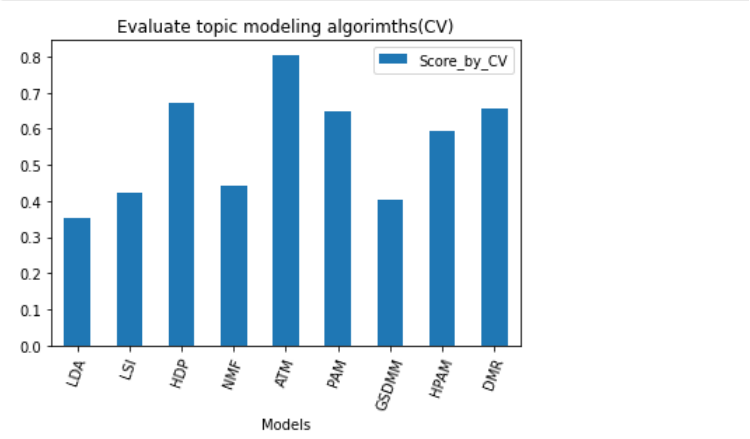

Figure 4.38: Evaluation by CV

4.5.5.2 Using U-Mass coherence

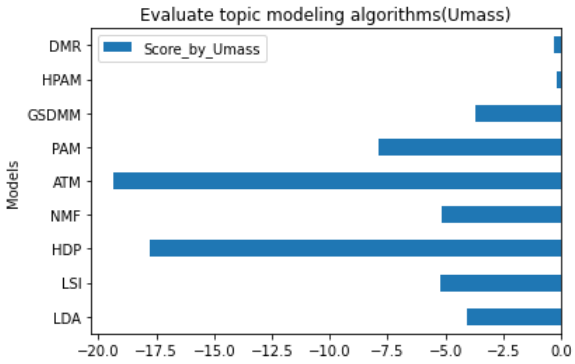

Figure 4.39: Evaluation by U-Mass

4.5.5.3 Using C-UCI coherence

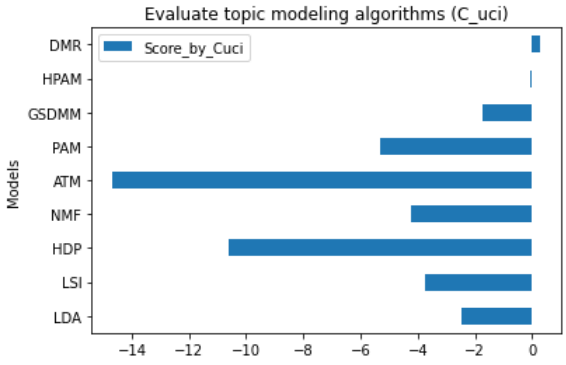

Figure 4.40: Evaluation by C-uci

4.5.5.4 Using C-NMPI coherence

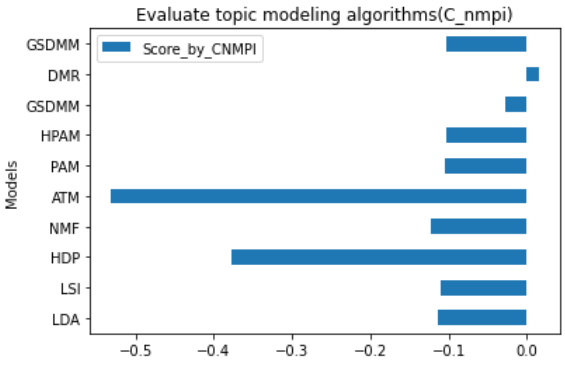

Figure 4.41: Evaluation by C-NMPI

4.6 Results

Out[109]:

	Model	Score_by_CV	Score_by_uMass	Score_by_cuci	Score_by_cnpmi
0	LDA	Low	Low	Low	Good
6	GSDMM	Low	Low	Low	Good
1	LSI	Good	Low	Low	Good
2	HDP	Good	Invalid	Low	Good
3	NMF	Good	Low	Low	Good
4	ATM	Good	Invalid	Low	Good
5	PAM	Good	Low	Low	Good
7	HPAM	Good	Good	Good	Good
8	DMR	Good	Good	Good	Good

Figure 4.42: Evaluation

The figure 4.42 shows the evaluation of the performance by coherence, It is knowing that cohernece score is a widely a used performance metric to evaluate topic modeling as well as perplexity and many other evaluations methods, but in this experiment cohernece was preferably used as a suitable choice with its differnent formulas or scores like C-V(typically 0 $<$ x $<$ 1), U-mass(typically -14 $<$ x $<$ 14),C-UCI(typically -14 $<$ x $<$ 14), C-NMPI(typically -1 $<$ x $<$ 0). As depicted in Figures 4.42, the coherence gives a realistic measure to identify the identified topics. We have tested 9 algorithms over the Arabic tweets dataset for topic modelling, more attention would be paid for the following 2 metrics :

1. Coherence cv (content vector), It creates content vectors of words using their co-occurrences and, after that, calculates the score using normalized pointwise mutual information (NPMI) and the cosine similarity.

2. Coherence UMass is recommended to be used by several recent approaches as Instead of using the CV score. Regarding the experiment results comparing all algorithms with different datasets, and looking at the 2 focused metrics, we decided to move on with the HPAM and DMR algorithms at the top , which yields the best CV score and Umass score is unstable.

BTM algorithm has only U-mass coherence but it is bad results and unstable.

4.7 Conclusion

In this chapter the variant types of algorithms used helped us to distinguish which algorithm can be more likely the best fit for our data-set to achieve the result of the topic modelling, however with help of Data reprocessing part.

As we know , any ML model is so sensitive to the training dataset , which we tried to be as much generic as we could, however, we couldn't generalize or expect the same exact performance on any different dataset.

General Conclusion

The Arabic short text topic modeling (STTM) techniques reviewed were divided into three broad categories: DMM, global word co-occurrences, and self-aggregation. We investigated the structure and properties preserved by various topic modeling algorithms, as well as the challenges faced by Arabic short text topic modeling techniques in general and by each approach category. Various STTM applications were presented. We used two famous open-source python libraries, GENSIM and TOMOTOPY, which are made up of surveyed short text topic modeling methodologies and evaluation tasks including clustering and topic coherence. Finally, we examined the outcomes of some publicly accessible real datasets used to evaluate the surveyed methodologies to these evaluation tasks.

Arabic STTM is an emerging field of machine learning and NLP, with numerous promising research directions:

(1) Visualization: Another difficult problem is determining how to display a document using topic models. Topic modeling provides useful information about the structure of each document. This structure, when combined with topic labels, can aid in identifying the most interesting parts of the document.

(2) Evaluation: Useful evaluation metrics for topic modeling algorithms (Nan values) have never been solved. Topic coherence is incapable of distinguishing between topics. One open area for topic modeling is the development of new evaluation metrics that correspond to how the methods are used.

(3) Model validation: According to the experimental results of this work, each method performs differently on different Arabic datasets. We cannot decide which topic modeling algorithms to use when dealing with a new corpus or task.

Bibliography

[1] Anaconda software distribution, 2020.

[2] Belal Abuata and Asma Al-Omari. A rule-based stemmer for arabic gulf dialect. *Journal of King Saud University - Computer and Information Sciences*, 27(2):104–112, 2015.

[3] Amani A. Al-Ajlan, Hend Suliman Al-Khalifa, and AbdulMalik S. Al-Salman. Towards the development of an automatic readability measurements for arabic language. *2008 Third International Conference on Digital Information Management*, pages 506–511, 2008.

[4] James F Allen. Natural language processing. In *Encyclopedia of computer science*, pages 1218–1222. 2003.

[5] Nasser Alsaedi, Pete Burnap, and Omer Rana. Sensing real-world events using arabic twitter posts. *Proceedings of the International AAAI Conference on Web and Social Media*, 10(1):515–518, Aug. 2021.

[6] David Andrzejewski, Anne Mulhern, Ben Liblit, and Xiaojin Zhu. Statistical debugging using latent topic models. In *European conference on machine learning*, pages 6–17. Springer, 2007.

[7] Sushil Bajracharya and Cristina Lopes. Mining search topics from a code search engine usage log. In *2009 6th IEEE International Working Conference on Mining Software Repositories*, pages 111–120, 2009.

[8] Bhagyashree Vyankatrao Barde and Anant Madhavrao Bainwad. An overview of topic modeling methods and tools. In *2017 International Conference on Intelligent Computing and Control Systems (ICICCS)*, pages 745–750. IEEE, 2017.

[9] Steven Bird, Ewan Klein, and Edward Loper. *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.", 2009.

[10] Halil Bisgin, Zhichao Liu, Hong Fang, Xiaowei Xu, and Weida Tong. Mining fda drug labels using an unsupervised learning technique-topic modeling. In *BMC bioinformatics*, volume 12, pages 1–8. BioMed Central, 2011.

[11] David Blei, Andrew Ng, and Michael Jordan. Latent dirichlet allocation. volume 3, pages 601–608, 01 2001.

[12] KR1442 Chowdhary. Natural language processing. *Fundamentals of artificial intelligence*, pages 603–649, 2020.

[13] Pádraig Cunningham, Matthieu Cord, and Sarah Jane Delany. Supervised learning. In *Machine learning techniques for multimedia*, pages 21–49. Springer, 2008.

[14] Ran Ding, Ramesh Nallapati, and Bing Xiang. Coherence-aware neural topic modeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 830–836, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

[15] Ali Farghaly and Khaled Shaalan. Arabic natural language processing: Challenges and solutions. *ACM Transactions on Asian Language Information Processing*, 8(4), dec 2009.

[16] Imane Guellil, Houda Saâdane, Faical Azouaou, Billel Gueni, and Damien Nouvel. Arabic natural language processing: An overview. *Journal of King Saud University-Computer and Information Sciences*, 33(5):497–507, 2021.

[17] Nizar Y. Habash. 2010.

[18] Mohamed Osman Hegazi, Yasser Al-Dossari, Abdullah Al-Yahy, Abdulaziz Al-Sumari, and Anwer Hilal. Preprocessing arabic text on social media. *Heliyon*, 7(2):e06191, 2021.

[19] Liangjie Hong and Brian D Davison. Empirical study of topic modeling in twitter. In *Proceedings of the first workshop on social media analytics*, pages 80–88, 2010.

[20] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007.

[21] Richard Hussey, Shirley Williams, and Richard Mitchell. Automatic keyphrase extraction: a comparison of methods. In *Proceedings of the International Conference on Information Processing and Knowledge Management*, pages 18–23, 2012.

[22] Ian T Jolliffe. *Principal component analysis for special types of data*. Springer, 2002.

[23] Alboukadel Kassambara. *Practical guide to cluster analysis in R: Unsupervised machine learning*, volume 1. Sthda, 2017.

[24] Da Kuang, Sangwoon Yun, and Haesun Park. Symnmf: nonnegative low-rank approximation of a similarity matrix for graph clustering. *Journal of Global Optimization*, 62(3):545–574, 2015.

[25] Lin Liu, Lin Tang, Wen Dong, Shaowen Yao, and Wei Zhou. An overview of topic modeling and its current applications in bioinformatics. *SpringerPlus*, 5(1):1–22, 2016.

[26] Tinghuai Ma, Raeed Al-Sabri, Lejun Zhang, Bockarie Marah, and Najla Al-Nabhan. The impact of weighting schemes and stemming process on topic modeling of arabic long and short texts. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*, 19(6):1–23, 2020.

[27] Jocelyn Mazarura and Alta de Waal. A comparison of the performance of latent dirichlet allocation and the dirichlet multinomial mixture model on short text. In *2016 Pattern Recognition Association of South Africa and Robotics and Mechatronics International Conference (PRASA-RobMech)*, pages 1–6, 2016.

[28] David M Mimno and Andrew McCallum. Topic models conditioned on arbitrary features with dirichlet-multinomial regression. In *UAI*, volume 24, pages 411–418. Citeseer, 2008.

[29] Fionn Murtagh and Pedro Contreras. Algorithms for hierarchical clustering: an overview. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2(1):86–97, 2012.

[30] Prakash M Nadkarni, Lucila Ohno-Machado, and Wendy W Chapman. Natural language processing: an introduction. *Journal of the American Medical Informatics Association*, 18(5):544–551, 2011.

[31] John Paisley, Chong Wang, David M Blei, and Michael I Jordan. Nested hierarchical dirichlet processes. *IEEE transactions on pattern analysis and machine intelligence*, 37(2):256–270, 2014.

[32] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[33] Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics.

[34] Duc Truong Pham, Stefan S Dimov, and Chi D Nguyen. Selection of k in k-means clustering. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, 219(1):103–119, 2005.

[35] Nektaria Potha and Efstathios Stamatatos. Intrinsic author verification using topic modeling. In *Proceedings of the 10th Hellenic Conference on Artificial Intelligence*, pages 1–7, 2018.

[36] Jipeng Qiang, Zhenyu Qian, Yun Li, Yunhao Yuan, and Xindong Wu. Short text topic modeling techniques, applications, and performance: a survey. *IEEE Transactions on Knowledge and Data Engineering*, 2020.

[37] Ahmed Rafea and Nada A. GabAllah. Topic detection approaches in identifying topics and events from arabic corpora. *Procedia Computer Science*, 142:270–277, 2018. Arabic Computational Linguistics.

[38] Daniel Ramage, David Hall, Ramesh Nallapati, and Christopher D Manning. Labeled lda: A supervised topic model for credit attribution in multi-labeled corpora. In *Proceedings of the 2009 conference on empirical methods in natural language processing*, pages 248–256, 2009.

[39] Radim Rehurek and Petr Sojka. Gensim–python framework for vector space modelling. *NLP Centre, Faculty of Informatics, Masaryk University, Brno, Czech Republic*, 3(2), 2011.

[40] Michael Röder, Andreas Both, and Alexander Hinneburg. Exploring the space of topic coherence measures. In *Proceedings of the eighth ACM international conference on Web search and data mining*, pages 399–408, 2015.

[41] Huan Rong, Tinghuai Ma, Jie Cao, Yuan Tian, Abdullah Al-Dhelaan, and Mznah Al-Rodhaan. Deep rolling: A novel emotion prediction model for a multi-participant communication context. *Information Sciences*, 488:158–180, 2019.

[42] Guido Van Rossum and Fred L Drake Jr. *Python reference manual*. Centrum voor Wiskunde en Informatica Amsterdam, 1995.

[43] Ike Vayansky and Sathish A.P. Kumar. A review of topic modeling methods. *Information Systems*, 94:101582, 2020.

[44] Bo Wang, Maria Liakata, Arkaitz Zubiaga, and Rob Procter. A hierarchical topic modelling approach for tweet clustering. In *International Conference on Social Informatics*, pages 378–390. Springer, 2017.

[45] Rui Wang, Deyu Zhou, and Yulan He. Atm: Adversarial-neural topic model. *Information Processing & Management*, 56(6):102098, 2019.

[46] Xiaohui Yan, Jiafeng Guo, Yanyan Lan, and Xueqi Cheng. A biterm topic model for short texts. In *Proceedings of the 22nd international conference on World Wide Web*, pages 1445–1456, 2013.

[47] J. Yi, T. Nasukawa, R. Bunescu, and W. Niblack. Sentiment analyzer: extracting sentiments about a given topic using natural language processing techniques. In *Third IEEE International Conference on Data Mining*, pages 427–434, 2003.

[48] Jianhua Yin and Jianyong Wang. A dirichlet multinomial mixture model-based approach for short text clustering. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, page 233–242, New York, NY, USA, 2014. Association for Computing Machinery.

[49] Jia Yuan Yu and Lirong Qiu. Ulw-dmm: An effective topic modeling method for microblog short text. *IEEE Access*, 7:884–893, 2019.

[50] Yueting Zhuang, Hanqi Wang, Jun Xiao, Fei Wu, Yi Yang, Weiming Lu, and Zhongfei Zhang. Bag-of-discriminative-words (bodw) representation via topic modeling. *IEEE Transactions on Knowledge and Data Engineering*, 29(5):977–990, 2017.