

**FINAL MASTER’S THESIS**

**Exploring Topic Modeling Approaches: Unraveling the 2020 US Presidential Election Discourse with BERTopic, Top2Vec, and LDA**

Name: Sherin Thomas

Candidate No: 260845

Supervisor’s Name: Prof David Weir

Submitted for the degree of Master's of Science

University of Sussex

August 2023

# Statement of Originality

This report is submitted as part requirement for the degree of MSc Data Science at the University of Sussex. This is to certify that to the best of my knowledge, this thesis and the work presented is the product of my own labor and has been generated by me as the result of my own original research except were indicated in the text.

Signature:

Sherin Thomas

# Acknowledgement

I want to sincerely thank my supervisor, Prof David Weir, for supervising this project and for sharing his knowledge of Natural Language Processing and topic modelling with me. His valuable guidance, support, and dedicated time throughout every stage of my project have been greatly appreciated.

# ABSTRACT

Social media is continuously evolving, and Twitter has played an important role in influencing the ideology and minds of the platform’s user. Many political parties' ideology, movements, decisions, history affect the party’s chance of getting votes and recognition from people. Understanding the underlying themes discussed by twitter users can help to understand the peoples’ stand for the given presidential candidate and help to improve their chances by adopting suitable measures. Topic modelling is the approach through which you can find the latent topic in the tweets which are tagged by the hashtag relevant to the party or candidate or tweets discussing relevant topics. There are many models for topic modelling which use different approaches like BERTopic which uses BERT for contextualized embedding, Top2Vec uses document embedding and Latent Dirichlet Allocation. Each model uses different approaches to give different topic labels which are human understandable and represent the documents. BERTopic, Top2Vec and LDA are adopted to find the underlying topics in the twitter dataset for 2020 US presidential election. Analyzing and comparing these models helps us understand how these models work and which has better performance among these. The analysis will help us understand contextual embedding topic model will give more coherent topics that were discussed in the tweets. After comparing and analyzing it was observed that BERTopic performed better than Top2Vec and LDA with highest coherence score of 0.75 and highest accuracy, F1, precision and recall value. It gives more coherent topics than the other model.

***Keywords –*** *BERTopic, Latent Dirichlet Allocation (LDA), Top2Vec, topic Model, UMAP, HDBSCAN, Coherence score*

Content

[INTRODUCTION 10](#_Toc145030863)

[1.1 Motivation 11](#_Toc145030864)

[1.2 Objective 11](#_Toc145030865)

[BACKGROUND 13](#_Toc145030866)

[PRELIMINARIES 15](#_Toc145030867)

[3.1 Text representation 15](#_Toc145030868)

[3.1.1 Bag-of-Words (BoW) 15](#_Toc145030869)

[3.1.2 Doc2Vec 15](#_Toc145030870)

[3.1.3 Word Embedding 15](#_Toc145030871)

[3.2 Topic Models 16](#_Toc145030872)

[3.2.1 Latent Dirichlet Allocation 16](#_Toc145030873)

[3.2.1.1 Generative Process for LDA 17](#_Toc145030874)

[3.2.1.2 Hierarchical structure 18](#_Toc145030875)

[3.2.1.3 Bayesian statistics in LDA 18](#_Toc145030876)

[3.2.2 Top2Vec 19](#_Toc145030877)

[3.2.2.1 Distributed Representation of Words, Documents and Topics 19](#_Toc145030878)

[3.2.2.2 Create Semantic embedding 20](#_Toc145030879)

[3.2.2.3 Find the number of topics 21](#_Toc145030880)

[3.2.2.2 Calculate Topic vectors 22](#_Toc145030881)

[3.2.3 BERTopic 23](#_Toc145030882)

[3.2.3.1 BERTopic algorithm 24](#_Toc145030883)

[3.2.3.2 Embedding 24](#_Toc145030884)

[3.2.3.3 Dimensionality Reduction 25](#_Toc145030885)

[3.2.3.4 Clustering 25](#_Toc145030886)

[3.2.3.5 Tokenizer 25](#_Toc145030887)

[3.2.3.6 Weighting Scheme 25](#_Toc145030888)

[3.3 Optuna 26](#_Toc145030889)

[3.4 Coherence score 26](#_Toc145030890)

[3.5 Perplexity 26](#_Toc145030891)

[METHODOLOGY 27](#_Toc145030892)

[4.1 Experimental Environment 27](#_Toc145030893)

[4.2 Study Overview 27](#_Toc145030894)

[4.3 Data Collection 28](#_Toc145030895)

[4.4 Data Preprocessing 28](#_Toc145030896)

[4.5 Topic Modelling 28](#_Toc145030897)

[4.5.1 BERTopic 28](#_Toc145030898)

[4.5.1.1 Visualization 29](#_Toc145030899)

[4.5.1.2 Coherence Score 29](#_Toc145030900)

[4.5.1.3 Hyperparameter Tuning 29](#_Toc145030901)

[4.5.1.4 Calculating Accuracy, Precision, Recall and F1 30](#_Toc145030902)

[4.5.2.1 Visualization 31](#_Toc145030903)

[4.5.2.2 Coherence and Perplexity Score 32](#_Toc145030904)

[4.5.2.2 Hyperparameter Tuning 32](#_Toc145030905)

[4.5.2.3 Calculating accuracy, precision, recall, F1 33](#_Toc145030906)

[4.5.3 Top2Vec 33](#_Toc145030907)

[4.5.3.1 Visualization 34](#_Toc145030908)

[4.5.3.2 Coherence Score 34](#_Toc145030909)

[4.5.3.3 Hyperparameter Tuning 34](#_Toc145030910)

[RESULTS & ANALYSIS 35](#_Toc145030911)

[5.1 BERTopic 35](#_Toc145030912)

[5.1.1 Intertopic Distance Map 35](#_Toc145030913)

[5.1.2 Hierarchical Clustering 36](#_Toc145030914)

[5.1.3 Barchart Representation and Word Cloud 37](#_Toc145030915)

[5.1.4 Coherence score and Performance metrics 38](#_Toc145030916)

[5.2.1 Topics 40](#_Toc145030917)

[5.2.2 Intertopic Distance Map 40](#_Toc145030918)

[5.2.3 Word Cloud 41](#_Toc145030919)

[5.2.4 Coherence score and Performance metrics 41](#_Toc145030920)

[5.3 Top2Vec 43](#_Toc145030921)

[5.3.1 Topics generated 43](#_Toc145030922)

[5.3.2 Word Cloud 43](#_Toc145030923)

[5.3.3 Coherence Score 44](#_Toc145030924)

[5.4 Comparison of BERTopic, LDA and Top2Vec 44](#_Toc145030925)

[5.5 Limitations 46](#_Toc145030926)

[CONCLUSION & FUTURE WORKS 48](#_Toc145030927)

[6.1 Conclusion 48](#_Toc145030928)

[6.2 Future Works 49](#_Toc145030929)

[REFERENCES 50](#_Toc145030930)

# Lists of Figures

[Fig 1. Generative process for LDA 17](#_Toc145022684)

[Fig 2: Graphical representation of LDA 18](#_Toc145022685)

[Fig 3: An example of a semantic space 20](#_Toc145022686)

[Fig 4: Topic vector identified in dense area of documents. 22](#_Toc145022687)

[Fig 5. Illustration of BERTopic architecture and its modularity throughout a variety of sub-models 23](#_Toc145022688)

[Fig 6. BERTopic’s default model algorithm 24](#_Toc145022689)

[Fig 7. Algorithm for BERTopic 24](#_Toc145022690)

[Fig 8. Study Overview 27](#_Toc145022691)

[Fig 9: Intertopic distance map for topics generated by BERTopic 35](#_Toc145022692)

[Fig 10: Hierarchical Clustering for topics obtained by BERTopic 36](#_Toc145022693)

[Fig 11: Barchart for representative words for topics obtained by BERTopic 37](#_Toc145022694)

[Fig 12: Word Cloud for topics obtained by BERTopic 38](#_Toc145022695)

[Fig 13: Optimal values for parameters for BERTopic after hyperparameter tuning 38](#_Toc145022696)

[Fig 14: Coherence score obtained for BERTopic 38](#_Toc145022697)

[Fig 15: Accuracy, Precision, Recall. F1 obtained for BERTopic 38](#_Toc145022698)

[Fig 16: Classification Report obtained for BERTopic 39](#_Toc145022699)

[Fig 17: Confusion metrics obtained for BERTopic 39](#_Toc145022700)

[Fig 18: Topics and representative words achieved by LDA 40](#_Toc145022701)

[Fig 19: Intertopic Distance map for topics obtained by LDA 41](#_Toc145022702)

[Fig 20: Word Cloud for the topics discovered by LDA 41](#_Toc145022703)

[Fig 21: Optimal values for hyperparameters of LDA 41](#_Toc145022704)

[Fig 22: Coherence score for LDA 42](#_Toc145022705)

[Fig 23: Accuracy, Precision, Recall. F1 obtained for LDA 42](#_Toc145022706)

[Fig 24: Classification Report obtained for LDA 42](#_Toc145022707)

[Fig 25: Confusion metrics obtained for LDA 43](#_Toc145022708)

[Fig 26: Word Cloud for representative word of top 3 topics in Top2Vec model 44](#_Toc145022709)

[Fig 27: Optimal values of hyperparameters of Top2Vec 44](#_Toc145022710)

[Fig 28 : Coherence score of Top2Vec with and without hyperparameter tuning 44](#_Toc145022711)

[Fig 29: Comparison of Coherence score for each of the topic models using Barchart 45](#_Toc145022712)

[Fig 30: Coherence score of BERTopic, LDA and top2vec 45](#_Toc145022713)

[Fig 31: Accuracy, recall, precision and F1 for BERTopic, LDA 45](#_Toc145022714)

[Fig 32:Barchart for comparison of BERTopic and LDA accuracy, recall, precision and F1 46](#_Toc145022715)

# List of Tables

[Table 1 List of hyperparameter values used in optuna for BERTopic 30](#_Toc145023066)

[Table 2 List of hyperparameter values used in optuna for LDA 32](#_Toc145023067)

[Table 3 List of hyperparameter values used in optuna for Top2Vec 34](#_Toc145023068)

**Chapter 1**

# INTRODUCTION

Elections are the fair formal process which ensures that the citizen of the country can express their opinion, ideology about the future of the country and elect the candidates to hold the important posts in the public office for the public welfare. Elections ensure democracy of the country. Elections let people express their opinion through voting and choosing the candidate they deemed suitable.

Social media plays a pivotal role in expressing the users’ opinion. Nowadays most people use social media platforms to express their heartfelt opinions. Be it about events, people, movements, people share their honest reviews on social media. This is also true about elections and campaigns. Users discuss the events and campaigns that can affect elections outcomes. Social media has become an important instrument to spread political messages, for political campaigns to connect with voters, to spread voters’ opinion.

The US presidential election 2020 is one of the cases where social media platforms have influenced the political opinion of the voters. People worldwide were enthralled by the US presidential election. The discussion around these elections has found a significant forum on social media, particularly Twitter, in the age of digital communication. Twitter has developed into a virtual battleground for ideas, opinions, and discussions relating to political campaigns, candidates, policy, and election results because of its real-time nature and large user base. Many are in the notion that social media has played a crucial role in shaping the political views and mindset of the voters. Many politicians even tried to harness social media platforms to actively campaign for their candidacy. Many users shared their views about each presidential candidate, the events related to election or that can impact the people's view about the candidate. US 2020 presidential election saw an intense public participation on social media platforms, particularly Twitter, as users, journalists, analysts, and even the candidates themselves communicated their immediate thoughts, opinions, and reactions. For example, many are in belief that social media also played a key role in Capitol riot 2021 following the defeat of Donald Trump in 2020 elections. There were a range of topics being discussed during the campaigning period. These topics were directly or indirectly related to candidates or their campaign and hence affected the electoral process. These topics held a political significance and helped shape the mindset of the social media users regarding the candidates and election. One of the most used social media platforms during the campaigning was Twitter. Twitter users tweeted about all the issues and events that they thought should be addressed and brought to attention before election so that these problems can be fixed or dealt with by the new President.

Uncovering hidden themes in the tweets allows us to explore a range of topics that captivated the minds of Twitter users. This will also allow us to understand what the voters deem to be obstacles to democracy. And many hashtags were used to depict different events, movements, influential people, and issues. For example, #blacklifematter was a movement to bring attention to the racism and try to eradicate the racism. Insights about popular sentiment, latest trends, and the dominant discourse can be gained from analyzing the large amount of Twitter data collected during a presidential election. Topic modelling is an effective method for sifting through this twitter sea for significant trends. We can use topic modelling to find hidden themes in text data, revealing the complex debates and varied points of view that take place online.

Natural Language processing has various innovative approaches to uncover the hidden topics and meaning within the textual data. Latent Dirichlet Allocation (LDA), Top2Vec, and BERTopic are powerful approaches for topic modelling, where LDA is a traditional topic modelling approach, BERTopic and Top2vec are the recent innovations that focus on improving the shortcoming of LDA. Top2Vec focuses on using word embeddings to capture semantic meaning for topic modelling. On the other hand, BERTopic uses contextualized embedding that captures intricate linguistic nuance which improves performance of topic model.

## 1.1 Motivation

The 2020 US Presidential election was historic moment for American democracy which represented the unprecedent involvement of social media. In the age of information where social media sites play an important role, Twitter has emerged as the platform where real-time discussions, debates, and expressions of political sentiment are freely expressed. So, understanding the multifaceted role that Twitter played in 2020 US presidential election motivated to discover the topics discussed in the tweets. Identifying the topics discussed by the people in twitter using topic modelling can help to get insights into public sentiment and opinions about the candidates, policies, and events associated with the election. Topic modeling helps us to identify the topics which contain misleading information hence it can aid in identifying tweets spreading false information and help in fact checking efforts. Topic modeling can help assess the effectiveness of campaign messaging and identify which campaign topics and messages resonated most with the public. Performing topic modelling in geopolitical twitter dataset can help political parties, candidates to tailor their strategies to address voters’ concerns. Topic modelling in the political dataset gives insight into various topics being discussed which gives idea into critical event in democratic process that driven to achieve topic modeling in this dataset.ginsights into a critical event in democratic processes.

## 1.2 Objective

The project objective is to answer the following questions:

1. **What are the underlying themes that are discussed for the US presidential election 2020 among the people on Twitter?**

We will employ topic modeling methodologies, including BERTopic, Latent Dirichlet Allocation (LDA), and Top2Vec, to delve into the topics that were prevalent in Twitter conversations during the 2020 US presidential election. Our objective is to reveal the latent themes and subjects discussed in tweets featuring relevant hashtags and discussions related to the election. This research will serve as a comprehensive exploration of the digital discourse surrounding this significant event.

1. **How does each topic modelling perform for the given monolingual twitter dataset?**

We will evaluate three different topic modelling techniques: BERTopic, Latent Dirichlet Allocation (LDA), and Top2Vec. Our assessment will be based primarily on two key metrics: coherence scores and the visual representations created by each of these models. These metrics are critical in determining how well each topic model is at uncovering coherent and interpretable themes from our dataset of interest.

1. **Can we achieve better topic modeling using contextual embedding topic models than traditional topic models?**

We will compare the performance of two topic modelling methods, BERTopic and LDA, in this dissertation research utilizing key classification measures such as accuracy, recall, precision, and the F1-score. We will manually categorize 300 tweets and use both models to evaluate their ability to reliably identify subjects. This assessment seeks to evaluate which model works best in terms of subject categorization, providing insights for text analysis applications.

**Chapter 2**

# BACKGROUND

This section we will see all the previous research and work done to perform topic modelling on the tweets and comparing and analyzing the performance of different topic modeling approaches for the different datasets. The three topic modelling approaches were carried out on twitter dataset. The majority of the background research will include the different approaches of topic modelling carried out in twitter dataset.

In terms of NLP, a topic model is the unsupervised tool that utilizes statistical algorithms to detect common theme and underlying semantic structures. Topic modeling was first implemented through probabilistic latent semantic analysis (PLSA) by Thomas Hofman et al 2001 based on the latent semantic indexing (LSI), a non-probabilistic method for automatic indexing and retrieval of semantic structures within texts, introduced by Deerwester et al. in 1990. Blei, Ng et al in 2002 generalized PLSI to introduce Latent Dirichlet allocation (LDA) which is the most widely utilized topic model. Many approaches were introduced as improvement of the existing topic models such as Top2Vec, BERTopic so on.

There has been some work done to identify the underlying latent topics in tweets on Twitter. Three topic modeling approaches that we will focus are LDA, Top2Vec and BERTopic. In 2015, David Alfred Ostrowoski to understand the topics or ideas propagated through twitter, he performed traditional topic modeling (LDA) in twitter messages [13]. For the purpose of Customer Relationship management, he performed the LDA on the filtered Twitter messages to understand the topics discussed by the customer. He evaluated the topics obtained by this approach to the pre-established topic categories that were of interest. First, he filtered the tweets that include “ford” and “focus”. On further examination of the tweet messages, he divided into four categories that is blather, news-driven, Interest towards product, desire towards purchase with percentage 45%, 35%, 15%, 5% for respective class. Conversational, sales and demand were the main categories they were interested in. He removed stop words, non-alphanumeric characters from 5k tweets for each interested category. Then LDA was applied to established collection considering the range of topic between 3 to 10. Clusters generated by LDA where average probabilities of words per topic above certain threshold were. They manually determined topics among each category of interest. They calculated precision and recall for each assigned topic and found topic 7 had the highest precision/recall in all three categories conversational, sales and demand I.e. (.520/4I2, .662/.204, .5I6/.124). Of the three categories, sales had highest precision/recall and conversational had lowest precision/recall. This was due to the fact that among sales there was commonality in the word collection and in conversational there were the highest number of cross-over topics as well as the highest number of unique words. For supporting generation of keyword/subtopic, topics were labelled manually. They extracted three topics in alignment with each category. Due to word diversity within their dataset the keywords that held the most significance within each assigned topic category had relatively lower probabilities compared to what is typically observed in other published works. Notably, in the sales category, words with high probabilities included "model years," which emerged as a prevalent cross-cutting theme. These terms were also identified in the other two assigned categories, albeit at a substantially lower probability level. Topics identified within the demand category had keywords like 'will' and 'going,' indicating an inclination towards immediate purchase intent. Additionally, the term 'afford' shed light on financial considerations. In the sales category, correlated words such as 'used' inferred the context of a sales event involving second-hand products. Within the conversation category, the term 'Sirius' denoted an expressed interest in product features.

LDA and Top2Vec was applied on the tweets related to covid19 to understand the topics discussed related to vaccine hesitancy by Ma, Treitler, Nelson in their research paper [14]. They trained the LDA model on the dataset which contained tweets that were related to vaccination. Grid Search hyperparameter tuning was performed on the smaller set of data to obtain a combination with the highest coherence score. The best performance was obtained for the number of topics, α and β with value 20, 0.01 and 0.31 respectively. Top2Vec was performed on the same dataset and a total of 3918 topics were obtained. To inspect the tweets the number of topics were reduced. And they identified 4 topics in LDA model relevant to vaccine hesitancy. And in top2Vec a total of 8 topics were obtained that were relevant to vaccine hesitancy.

Another research that was done for comparing and analyzing topic models was done by Abhinandan Udupa, Adarsh K N, Anvitha Aravinda, Neelam H Godihal in 2022 for GSDMM and BERTopic[15]. In their study they perform topic modelling on short text. They carried out this in three different dataset that are Trump Twitter Archive Dataset, ABC News Headlines Dataset, Stack Overflow Dataset. They evaluated each model based on the coherence score of each model. They observed the coherence scores that were obtained for different values of different parameters. It was observed that that N-gram with same maximum and minimum value gives high coherence score. It was also studied that the semantic similarity between the words that describe a topic decrease with an increase in words representing the topic. A more distinctive phrase is present with an increase in N-gram which decreases the coherence score.

**Chapter 3**

# PRELIMINARIES

## In this section, we will discuss three different approaches to topic modeling: BERTopic, LDA, and Top2Vec. The primary focus of this section is to explore the fundamental concepts and architecture of these topic models, each of which employs distinct methods to identify latent topics. The objective is to gain a deep understanding of how these three approaches work.

## 3.1 Text representation

Identifying and extracting meaningful topics from a collection of documents is carried out by text representation, it plays a crucial role in topic modelling. Topic modelling algorithms work with numerical representation text, so raw text is converted into the form that can be processed by the topic modelling algorithm, which is referred to as text representation. In this section we will discuss text representation that is used by BERTopic, Top2Vec and LDA.

### 3.1.1 Bag-of-Words (BoW)

BoW model is a simplified representation used in Natural Language Processing (NLP) and Information Retrieval (IR) according to A.A.A. Karim and R.A. Sameer [1]. BoW represents each document as bag of words present in documents. It creates a vocabulary with all distinct words across all the documents. In this document is represented using a fixed length vector which is equal to vocabulary size and each dimension in vector corresponds to the occurrence of the word in the respective document. Since it doesn’t consider order or grammar it doesn’t capture semantic and contextual information

### 3.1.2 Doc2Vec

Doc2Vec is vector representation, which was proposed by Li, Mikolov in 2014[2] is the extension of the Word2Vec. It generates a vector representation of documents. Word2Vec learns the learns dense continuous embeddings by taking into account the context of the words. In Doc2Vec, it uses word2vec to create word embedding and in addition it also has paragraph vector to represent document. This combined vector representation is assigned to each document. It is useful to capture the context and semantic structure of the document.

### 3.1.3 Word Embedding

Word embeddings capture semantic relationships and contextual information by representing words in continuous space as dense vector. They are commonly used in many NLP algorithms like BERT, transformer since it captures the meaning and the context of the word in the sentence or document.

## 3.2 Topic Models

### 3.2.1 Latent Dirichlet Allocation

LDA is generative probabilistic model for topic modelling which is extended from Hoffman’s PLSI model mentioned in section 2.1. PLSI model is probabilistic model, but it can’t perform probabilistic modelling at document level [3]. The fundamental concept of LDA is that it considers documents as random mixtures of the hidden topic in probabilistic manner.

LDA is a three-level hierarchical Bayesian statistical modeling which makes it a more complete generative model unlike PLSI which adopts the two-level Dirichlet-multinomial clustering model. LDA is a hierarchical structure with documents, topics and words as its component [4].

**Notation**

The terms used by Blei to discuss LDA are [3]:

* A word is the basic unit of data in discrete form which is defined as part of vocabulary indexed as {1*,…, }.* The word is represented as unit basis vector where only a single component is 1 and others are 0. So, word in vocabulary is represented as vectored such that =1 and =0 where
* A document is composed of sequence of N words denoted by w= ()where is the word in the sequence
* Collection of M documents is known as corpus which is denoted as D=() where represent document in the given corpus.

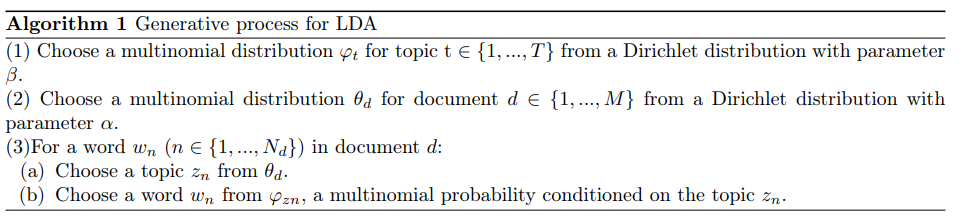
The fundamental idea on which LDA is based, it represents documents as random combination of hidden topic. Each topic is defined by a set of words, and documents are seen as composed of various latent topics, each contributing with a specific distribution of words. This approach allows us to uncover the underlying themes present in a collection of documents by modeling how topics are mixed together in each document. In LDA, it is considered words are generated by topics.

The LDA generative model allows a group of unobserved variables to explain a set of observations [4]. In a document a topic is sampled for each word in the documents. It associates word to topic. The multinomial distribution from which you sample topics is influenced by a prior belief about the distribution of topics in the corpus [5]. The likelihood of choosing a specific topic for a word is determined by both the words in the document and the prior distribution. Words that you believe are likely to belong to specific topics will have a higher chance of being assigned to those topics in the new documents. LDA allows each document to be represented as a probability distribution over topics. This distribution represents the likelihood that a particular topic contributes to the content of the document. It is assumed that the topic distribution in all documents shares a common Dirichlet prior. The primary objective of LDA is to learn a probabilistic model that accurately represents the observed documents while also capturing the hidden thematic structure. This allows LDA to assign high probabilities not only to the documents themselves but also to similar documents based on their topic distributions.

#### 3.2.1.1 Generative Process for LDA

The generative process assumed by LDA for each document in a corpus consisting of documents having words .

Algorithm for generative process is shown in Fig 1.



*Fig 1. Generative process for LDA*

In the generative process of LDA only words in the document are the observable variable and the rest are the latent variables. The variables used are:

*M* denotes the number of documents

*N* is number of words in a given document (document *i* has )

*α* is the parameter of the Dirichlet prior on the per-document topic distributions

*β* is the parameter of the Dirichlet prior on the per-topic word distribution

is the topic distribution for document i

is the word distribution for topic k

is the topic for the *j*-th word in document i

is the specific word.

The joint distribution of topic distribution for document , a set of N topics , and a set of N words the parameter, document-topic distribution topic-word distribution given by

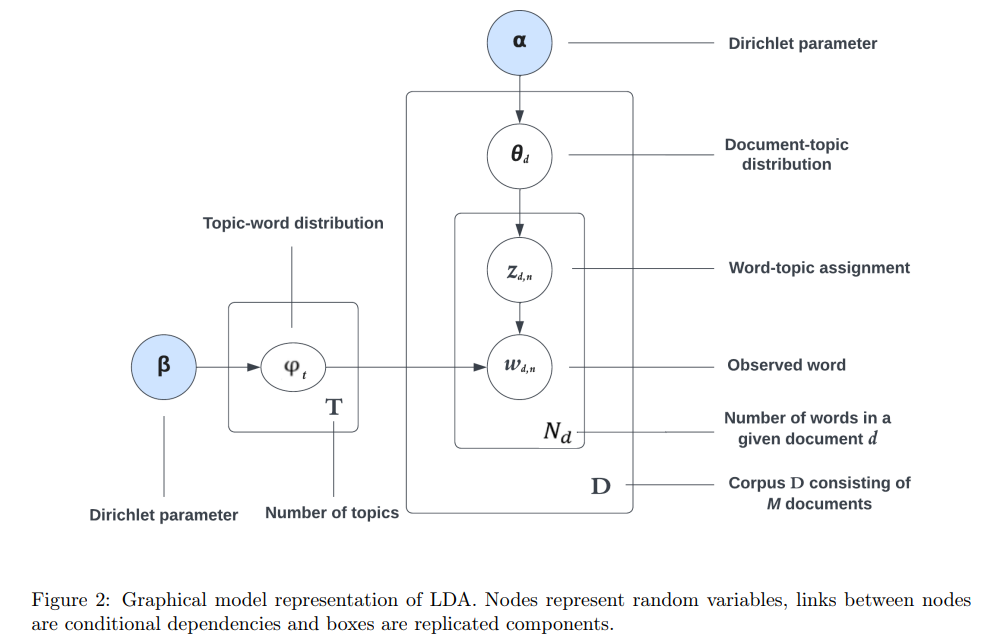
where is simply for the unique i such that . Integrating over θ and summing over , we obtain the marginal distribution of a document

The product of the marginal probabilities of single documents, we obtain the probability of a corpus:

#### 3.2.1.2 Hierarchical structure

LDA assumes documents are being generated involving topics and words by hierarchical process. LDA has three levels. These three levels are for corpus, document, word. The parameters and are corpus parameter assumed to be sampled only once during the generation of the entire collection of documents in corpus. The variables are document level parameters and represent the proportions of topics in a specific document d. These are sampled once for each individual document and determine how much of each topic is present in that document. The and are word level parameters. These variables are sampled once for each word and determine the topic assignment for the word as well as the actual word that is generated from the chosen topic. The graphical model of LDA is shown in Fig 2. This three-level structure ensures that multiple topics are assigned to a document.

This structure utilizes Bayes Theorem to make inference about model’s parameters [6]. Bayesian statistics utilize prior knowledge and probability distributions with information in observed data to update the prior knowledge of the parameters of the model. In this model Dirichlet distribution , and multinomial distribution is updated. The words within each document are used to update the topic proportions within that document and the topic-word distributions for each topic.



*Fig 2: Graphical representation of LDA*

#### 3.2.1.3 Bayesian statistics in LDA

The Bayesian statistics have three component prior distribution, likelihood function, posterior distribution. The prior distribution is prior knowledge of parameter which in case of LDA is Dirichlet priors. The likelihood function is a probability distribution that expresses the probability of observing the given data under a specific set of parameter values. The likelihood function in LDA is given latent variables it depicts how well the observed words fits generative process of the model. Third component posterior distribution is updated probability distribution of the parameter of the interest. It captures the probability of the latent variables after considering both the prior and the likelihood of observing the data.

### 3.2.2 Top2Vec

Top2Vec is a topic modelling algorithm which was introduced by Dimo Angelov in his paper “Top2Vec: Distributed Representation of Topics”. This topic modelling method generates the jointly embedded distributed representation of documents, topics, words to detect the latent topics in the document [7].

The major variation of Top2Vec than other traditional models like LDA, is that it utilizes the neural network generated joint document and word embedding in the semantic space. This joint embedding helps to capture semantic meaning of the document. The LDA model creates new document-word distribution by considering topic as distribution of words. So, many uninformative words have high probabilities in topics because they occur more in the documents. On the contrary, within a semantic embedding, a topic vector derived from top2vec signifies a predominant theme that is commonly present across various documents. LDA considers BoW so it doesn’t take into account the order of the words and semantic meaning whereas Top2Vec utilizes Doc2Vec vector to represent topics in semantic space. Topics generated by Top2Vec algorithm gives more information about corpus than LDA.

Top2Vec uses neural network to generate pre-trained doc2vec which is extension of word2vec to represent distributed representation of document and word2vec to represent distributed representation of word. These distributed representations create jointly embedded document and word vector which is known as a semantic embedding. The semantic embedding is represented in semantic space where each point in semantic space represents separate topic and the dense cluster of documents in space represent the documents with similar topic and the word vector close to document vector likely represent words similar to document thus document topic. After getting dense area of document, we calculate centroid to find the topic vector which is the representative of that topic. Since it takes joint document and word embedding stop word removal, lemmatization, stemming and prior knowledge that is required is LDA is not needed for top2Vec.

#### 3.2.2.1 Distributed Representation of Words, Documents and Topics

Word2Vec utilizes neural network for generating distributed representations (word embeddings) of words in a continuous vector space to capture syntactic and semantic word relationships. Word2Vec is based on the idea that similar words will occur in similar context, so it predicts the adjacent words for a given word based on the context by sliding window over each document and thus it calculates similarity between two words. Continuous Bag of Words (CBOW) model and the Skip-gram model are part of Word2Vec.

Doc2Vec extends Word2Vec by adding distributed paragraph vector. A paragraph vector predicts the neighboring words which should be present and the context of windows of words. So, it stores the topic of documents. Each context window is informed which information is missing by paragraph vector. This allows us to represent distributed representation of documents.

Doc2Vec utilizes pretrained word vectors to learn embedded document and word vectors. By learning from joint embedded vector, the document vector is close to word vector with similar meaning. This joint semantic embedding learns semantic association between documents and words. Each point in semantic space is different topic which is characterized by the nearest words. The dense area of documents vector which represents many documents which have same topic is used to calculate distributed topic vector. Topic vector is calculated by finding the centroid of each dense area of documents. Each dense region of document vectors is a different topic. The word vector close to document vector means the words represent the document. The fig for document and word vector in semantic space is shown in fig 3

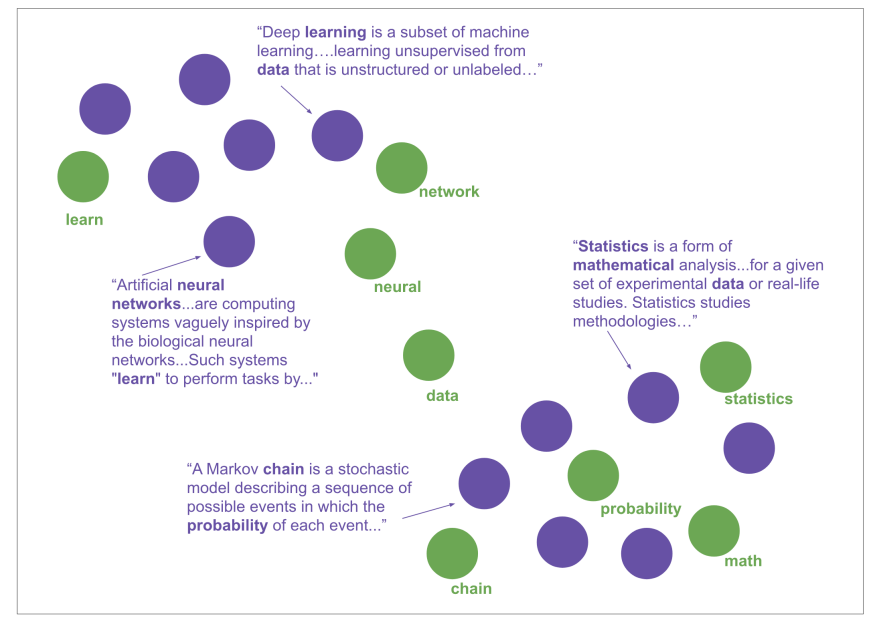


Fig 3: An example of a semantic space. *Documents are represented by purple points and green spots denote words. Words are placed near to documents they best represent, and similar documents are placed together [7].*

#### 3.2.2.2 Create Semantic embedding

For extracting topics, a jointly embedded document and word embedding in the semantic space is created using word2vec and doc2vec. Similar document vectors are placed closely in the semantic space. To learn word embedding top2vec use word2vec approach and document embedding using doc2vec. There are two versions of doc2vec to learn jointly embedded word and document vector: the Paragraph Vector with Distributed Memory (DM) and Distributed Bag of Words (DBOW). The DM model predicts target word in the context window from context words and document vector. DBOW predicts neighboring words in the context window in the document using document vector.

Top2Vec learns word vectors from word2vec. In word2vec, skip-gram model for each word in the vocabulary it learns an input word and word vector. The word2vec model consists of two matrices for input word and context word vector that are and where n is corpus vocabulary size and d is vector size for each word that is to be learned. Each row in the contains a word vector and each row consists of context word vector . For a given context window size k and for each 2k surrounding words w of context word , their context words are predicted from input vector using . This produces a probability distribution throughout the vocabulary, with each word being the context word . The learning process updates the context word vector and input word vectors such that the probability of the context vector given the surrounding word , is greatest in the probability distribution over the vocabulary so that semantically similar words will have context words placed closely.

The DBOW in doc2vec is similar to skip-gram of word2vec. This model consists of matrix where c is the number of documents in the corpus and d is the size of vector to be learned for each document and context word matrix . The document’s vector for each document d in the corpus using . This generates probability distribution over the corpus for each document being the document the word is from, is greatest in the probability distribution over the corpus of documents. This learning approach requires a document vector placed closely to the word vector of word present in document. Thus, a jointly embedded document and word vector in a semantic space is generated which is continuous representation of topics.

#### 3.2.2.3 Find the number of topics

In semantic space the document vector represents the document’s topic, and the word vectors close to document vectors describes the document’s topic. In semantic space dense area of document vector represent area of similar documents, so we can calculate the underlying topic of the similar documents by calculating the centroid of the vectors which represent topic vector. But these high dimensional semantic embeddings lead to sparse document vectors in semantic space and difficult to cluster. So, we reduce the dimensionality of these vectors using UMAP and after dimensionality reduction we will cluster the joint embeddings to find dense areas of documents in semantic space which will be performed using HDBSCAN.

**UMAP**

Uniform Manifold Approximation and Projection (UMAP) is dimensionality reduction algorithm proposed by McInnes, L, Healy, J in their paper "UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction" from 2018 [8]. It is a technique which is used for visualizing high-dimensional data in lower dimensions while preserving the underlying structure and relationships within the data. UMAP is particularly effective in capturing complex and non-linear patterns in the data. UMAP ensures that the local and global structure of data is intact.

UMAP constructs the fuzzy topological representation of data using graphical representation. Points close to each other are connected by edges and this distance between points can be calculated by different metric. UMAP preserves the distance between the points in high dimensional space while projecting them into low dimensional space by optimizing an objective function that balances the preservation of the global and local structure of data. This approach makes it suitable for a wide range of data types and patterns, including clusters, manifolds, and intricate relationships.

**HDBSCAN**

We need to find the dense area of the document vector in the semantic space to find topic vector. This can be achieved by HDBSCAN. Hierarchical Density-Based Spatial Clustering of Applications with Noise (HDBSCAN) is an extension of DBSCAN. It clusters the data points by calculating the density of the data points in the feature space. The algorithm's initial step involves creating a hierarchical clustering tree. This tree is subsequently utilized to establish clusters within it using a Minimum Spanning Tree (MST) based on the calculated densities and distances. Lastly, a process of cluster stability analysis is applied to determine the most suitable clustering solution [9].

A metric is used to calculate pairwise distance between points in the space, to construct a distance matrix. HDBSCAN can form clusters for varying densities, it also identifies and separate noise points from meaningful clusters. HDBSCAN labels each cluster in the semantic space, and which later can be used to calculate topic vector. HDBSCAN assigns label “-1” to the outlier documents that cannot be assigned to any clusters, thus dealing with noise.

#### 3.2.2.2 Calculate Topic vectors

A topic vector is calculated after labeling for each dense cluster that has been identified in semantic space. To calculate topic vector, we can calculate the centroid of all the document vectors present in the same dense cluster which is shown in Fig 4. To find the words representative of the topic we can find the word vectors occurring close to the topic vector in semantic vector. The distance between each word vector and topic indicates how semantically similar the word is to the topic.

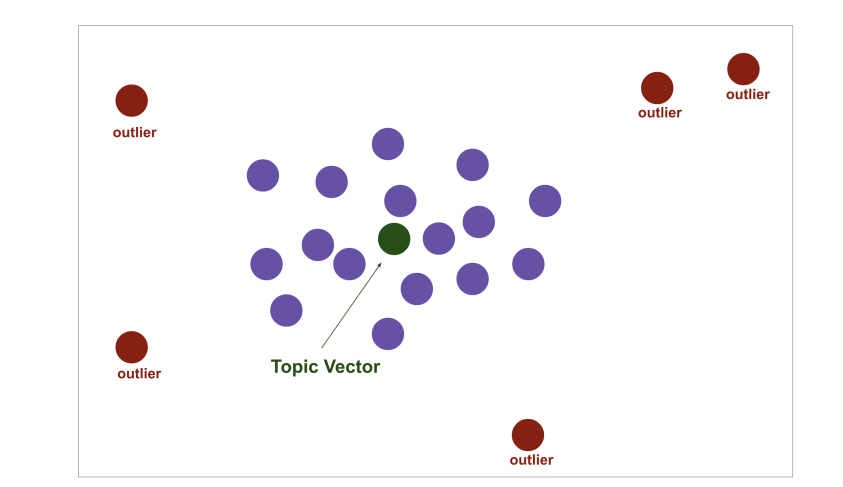


Fig 4: Topic vector identified in dense area of documents. *The topic vector is the centroid of the dense area of documents identified by HDBSCAN, which are purple points. The outliers identified by HDBSCAN are not used to calculate the centroid.[7]*

### 3.2.3 BERTopic

BERTopic is state-of-the-art topic modeling algorithm proposed by Maarten Grootendorst in his paper "BERTopic: Neural topic modeling with a class-based TF-IDF procedure" [10]. The key innovation of this model is that it utilizes transformers and c-TF-IDF to generate dense clusters to interpret topics and consider important words that are more accurately describe the topics.

BERTopic utilizes transformers to convert the documents into embeddings to capture semantic meaning. These embeddings are represented in high dimension vector space, the clustering will be inaccurate and difficult. To overcome this problem, we reduce the dimensionality of the embeddings using UMAP. These reduced dimensionality embeddings form dense areas of embeddings which are clustered using HDBSCAN. The topic representations are constructed by assigning topics to each cluster using c-TF-IDF.

BERTopic generates the topic representation using 5-6 steps. Each step we can choose from many sub-models which ensures modularity and allows you to build your own topic model. This is represented in Fig 5.

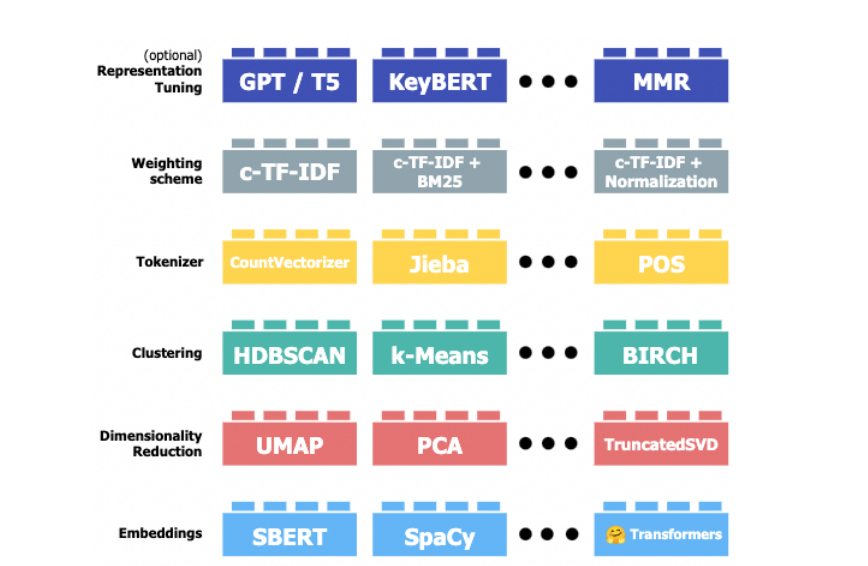


Fig 5. Illustration of BERTopic architecture and its modularity throughout a variety of sub-models. *Figure from [11].*

The default values for each step in the BERTopic architecture is shown in the Fig 6 below.

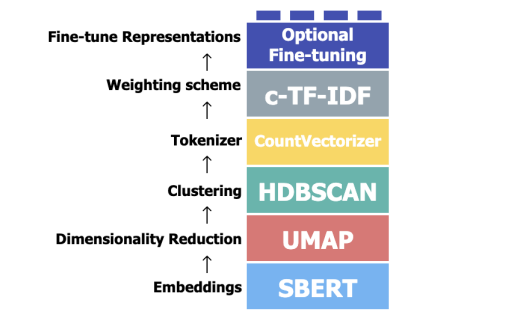


Fig 6. BERTopic’s default model algorithm [11]

#### 3.2.3.1 BERTopic algorithm

To summarize how BERTopic find the topic representation, the algorithm for BERTopic is given Fig 7

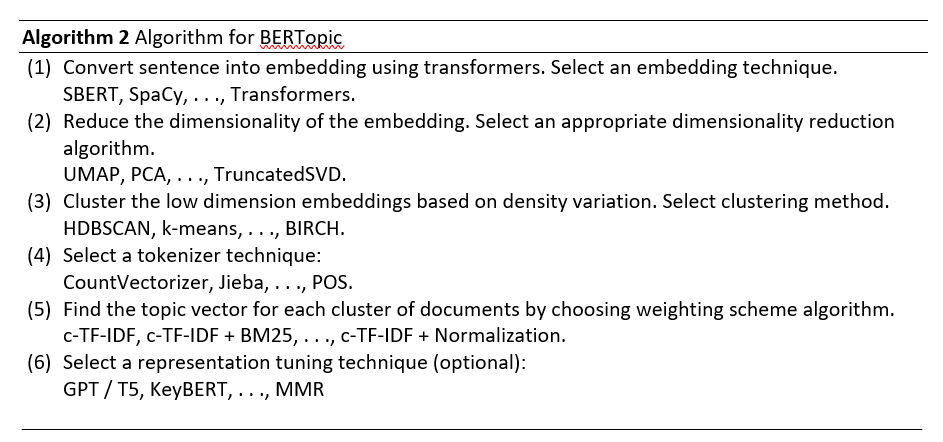


Fig 7. *Algorithm for BERTopic*

#### 3.2.3.2 Embedding

BERTopic leverages transformers to convert the sentence in the document to numerical representations through word embeddings. The default value for embedding model is *sentence-transformers* also known as SBERT(Sentence-BERT). We are going to use the SBERT in our project.

**Sentence-BERT**

SBERT is an extension of BERT that is utilized for generating semantically meaningful sentence-level embeddings. To perform sentence similarity tasks on large collection of sentences, BERT is unsuitable as it takes a lot of time (65 hours) and yields bad sentence embeddings. To address this problem SBERT uses Siamese network architecture to generate sentence embedding from input sentences.

Siamese network consists of two neural networks with same weights which generate semantically meaningful sentence embeddings and compare the embeddings using cosine similarity. SBERT adds pooling function to output that is obtained from BERT model to generate fixed size sentence embedding. It then performs objective function to compute similarity. The generated sentence embedding captures semantic meaning of sentence in robust manner. This approach was introduced by Reimers and Gurevych in 2019 [12].

SBERT in BERTopic converts sentences and paragraphs in the documents to dense vector representations. These vectors then can be clustered such that semantically similar documents are in same cluster and will have same topics.

#### 3.2.3.3 Dimensionality Reduction

Dimensionality Reduction enhances efficiency and accuracy in identifying topics in the large text dataset in BERTopic. The embeddings generated from SBERT are high dimensional vectors. Each sentence in documents is represented as vector in high dimensional space and dimensionality of vector space increases with increase in vocabulary size of dataset. As dimensionality of data increases the distance to the nearest neighboring data point tends to converge toward the distance to the farthest data point. So, we use UMAP to solve this issue. UMAP as discussed in the section ensures global and local structure of the embeddings is preserved while mapping in lower projected dimensions. It improves the performance of clustering techniques.

#### 3.2.3.4 Clustering

The lower dimension sentence embedding needs to be clustered such that semantically similar documents occur in the same cluster. This enables us to find the accurate topic representation for documents. There are various well-known clustering algorithms such as HDBSCAN, k-means and so on.

The default clustering that is used in BERTopic is HDBSCAN. As discussed in section, HDBSCAN identifies the cluster with varying densities and noises. The clusters created by these algorithms can be used to find the latent topic and theme in the collection of data.

#### 3.2.3.5 Tokenizer

Before performingthe weighting scheme, a technique should be adopted for modularity. A centroid based topic representation method may not be feasible on the clusters obtained as result of HDBSCAN, as the resulting clusters have varying levels of density and shape. Therefore, tokenizers convert documents into bag-of words representation on cluster level, by considering all documents in the cluster as single documents and then count frequency of each word in the cluster. In BERTopic we use Count Vectorizer as tokenizer.

#### 3.2.3.6 Weighting Scheme

The topic representations are constructed by analyzing the documents within each cluster, such that a single topic is assigned to each topic. So, we need to find the words that describe one topic by analyzing the importance of words to the document. So BERTopic utilizes TF-IDF to represent the importance of words to a topic such that it can describe the topics.

## 3.3 Optuna

Optuna is a software framework for hyperparameter optimization by using state-of-the-art algorithms. It utilizes algorithms which concentrate on the area where hyperparameters allow model to perform better so that it selects best hyperparameters combination and the algorithms that carries out statistical tests to eliminate search spaces that exhibit poor performance and are unlikely to contain the global minimum. It is light-weight and versatile framework with easy parallelization.

## 3.4 Coherence score

Coherence score is the score used to evaluate topic models. It measures how well the words in the topic relate to each other. It evaluates the performance of topic model in capturing the latent topics in a corpus of text. This helps us to understand how well the model extracts the topics and how interpretable the topic is. The most commonly used coherence score is c\_v. In this method, content vectors are generated by considering their co-occurrences. it computes a score using normalized pointwise mutual information (NPMI) and cosine similarity.

## 3.5 Perplexity

Perplexity measures how well the probability model predicts the unseen sample. In topic modeling the perplexity measures how well the topic model predicts new or unseen data. Perplexity is an evaluative metric used in topic modeling to gauge the level of "uncertainty" present in the predictive outcomes of a model.

**Chapter 4**

# METHODOLOGY

In this section we will discuss the steps and methods used to perform the experiment. In this explain approaches taken to perform the experiment and the data collection and preprocessing. We will also discuss hyperparameter tuning to improve the performance of the topic models.

## 4.1 Experimental Environment

The topic models were done in Visual Studio Code IDE. This was executed using Python 3.9, which is a popular and powerful programming language due to dynamic typing and its interpreted nature. The hardware environment used for performing the project is operated on Windows, powered by 1.70 GHz Dual Core 12th Gen Intel Core i7 processor and 16 GB RAM.

## 4.2 Study Overview

Figure shows the overview of the experiment. Firstly, I collected the relevant dataset for my topic which discussed the events, persons and opinions related to the US presidential election of 2020. Then I manually labelled the data according to a few topics of interest. I performed topic modelling using three approaches LDA, Top2Vec and BERTopic. I evaluated each model based on the coherence score of each model. And observed the topics I obtained from each model. I also performed hyperparameter tuning to find the value of the hyperparameter to get a better coherence score. Then I trained BERTopic and LDA and tested them to understand if BERTopic performed better than LDA. The step in Fig 8 shows the step-by-step approach to compare and analyze three topic models.

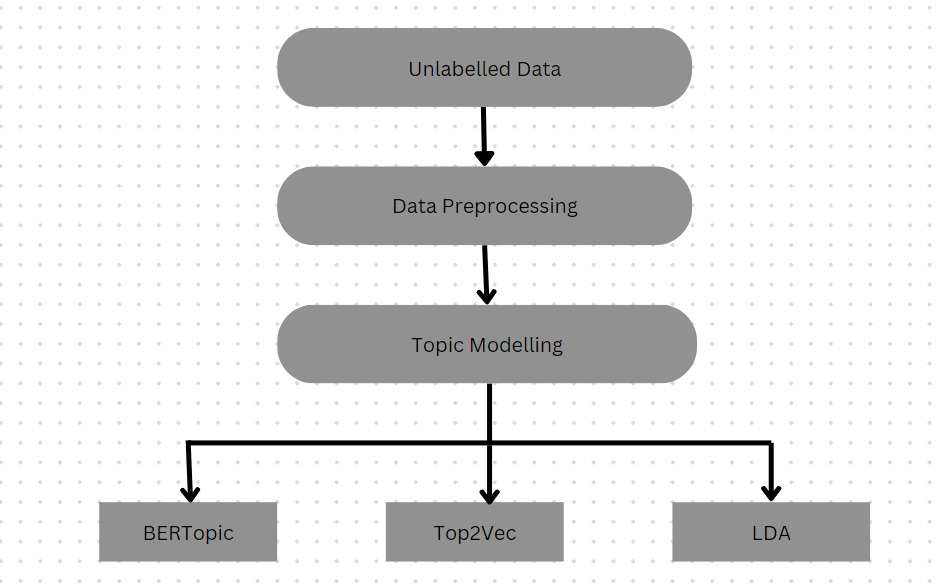


Fig 8. Study Overview

## 4.3 Data Collection

The data for these projects are the tweets which are somehow related to the U.S. presidential election 2020 and candidates participated in the election and hashtag related to election and events that have an impact on the election. Tweets can be text, image or video, but in this case, we are only considering text tweets. The data was collected from the open-source data-science platform Kaggle. Since the election was held on 3 November 2020 the dataset contains the tweets from 15.10.2020 and 04.11.2020. The dataset consists of columns related to the time the tweet was created, tweet, tweet id, likes and retweet count for the tweet, source through which tweet was posted, details about the user who posted the tweet, the location of user while posting the tweet. There were around 809,000 tweets, we discarded the tweets in the other language since we are trying to reveal topics in the monolingual data. We got around 612,000. We are only considering 5000 tweets for easy computation in the system. These 5000 records were all tweeted in English language.

## 4.4 Data Preprocessing

These tweets contain emojis, special characters, links, tags, multiple spaces, mentions. We need to remove this for further processing, so we will be performing the data cleaning to remove this and consider only relevant tweets and columns for processing. Firstly, we will be removing all the emojis in the tweets. Then we will be removing all punctuations, links, URLs, user mentions, extra space between characters that occur in the tweet using regular expression. Retweet data and duplicate data will be removed. We will be removing hashtags at the end of the tweet as it does add meaning to the statement but the hashtags that occur in the middle of the statement, we only remove the symbol # before the word so that the semantic meaning of the statement is not lost. The common structure of hashtags, which often involves combining multiple words together. As a result, studying them linguistically becomes challenging, as they fall under the category of unfamiliar terms. However, by categorizing the hashtag format into three scenarios, it becomes simpler to break down hashtags into individual words. All the special characters and the multiple spaces need to be removed.

## 4.5 Topic Modelling

All the three-topic model BERTopic, Top2Vec, LDA will be performed on the cleaned dataset, and we will inspect the topic obtained from topic models. We will also perform hyperparameter tuning to improve the performance of each topic model.

### 4.5.1 BERTopic

In this section we will discuss how to utilize BERTopic to identify the latent topics in the tweets in the dataset. The BERTopic topic modelling is implemented using bertopic library. Before passing text data after preprocessing, the data is converted into a list of text containing tweets in the dataset. First an instance for BERTopic model is created. In BERTopic, we will be using embedding paraphrase-MiniLM-L3-v2. This embedding model is a sentence transformer that transforms sentences and paragraphs into 384-dimensional dense representation. These embeddings capture the semantic meaning of each tweet into 384-dimensional vector space and can be utilized for clustering. The parameter “calculate\_probabilities” is assigned value True which indicates that model will calculate probabilities of every word in each topic. These probabilities can help to identify the distribution of words per topic. The “verbose” parameter is set to True this indicates that it helps to track the progress of the model. After creating an instance of the BERTopic with the following parameters, we will fit the model on the collection of collected tweets. After fitting the model, it returns topics and probabilities of assigned topic per document.

#### 4.5.1.1 Visualization

After obtaining the topics for each document and the words representing each topic, we can visualize the topics extracted using graphical representation in form of intertopic distance mapping. For this visualization BERTopic uses c-TF-IDF representation of topic embedded into 2D using UMAP and plotted using Plotly. Each circle on the distance map represents a topic, and the size of each circle corresponds to the frequency of that topic across all the tweets in the dataset. We will also visualize the topics generated by BERTopic using hierarchical clustering. Barchart is used to visualize the top 15 topics and the probabilities of each word within each topic. We also visualize frequency of each word in the collection of the document assigned the representative topics using wordcloud.

#### 4.5.1.2 Coherence Score

To calculate the coherence of the model, we create the dataframe with documents, id and the topics assigned to each tweet by the model and then cluster the documents based on the topics and concatenate all the text within each topic. Then we will extract the features (words) using inbuilt vectorizer and analyzer in BERTopic. BERTopic uses Count Vectorizer to vectorize the words in the documents. Dictionary and corpus are created using the words obtained by vectorizer. Then we will retrieve the words for each topic. Then we will build coherence model based on the dictionary, corpus and the words retrieved. We are using cv score to calculate the coherence score. CV scores calculate the score by creating content vectors of words using their co-occurrences and then uses normalized pointwise mutual information (NPMI) and the cosine similarity.

#### 4.5.1.3 Hyperparameter Tuning

To find the best value for each parameter for the model we will perform hyperparameter tuning using optuna. Optuna is an automatic hyperparameter optimization software framework [15]. We are using optuna because it helps to create search space dynamically for hyperparameters. In optuna study and trial are key concepts for optimizing hyperparameters. *Study* isthe optimization based on objective function. It manages the optimization process by searching for the best set of hyperparameters and holds the values for all optimization trials. *Trial* is a single execution of the objective function with specific set of hyperparameters. Each trial corresponds to training and evaluating the model using a particular set of hyperparameters. Optuna explores different sets of hyperparameters by running multiple trials. We will be finding the optimal value of min\_topic\_size, nr\_topics, top\_n\_words. The parameter top\_n\_words is the number of words per topic extracted by model. If this value is set too high, it can impact topic embeddings. The optimal value for this parameter is 10. The parameter min\_topic\_size is the minimum size of the topic and lower number of clusters is formed as the we increase this parameter. The parameter nr\_topics specifies the number of topics identified by the model. The range of possible values for each hyperparameter and the respective hyperparameter is shown in table 1

|  |  |  |
| --- | --- | --- |
| S No. | Parameters | Values |
| 1. | min\_topic\_size | [8,9,10] |
| 2. | nr\_topics | [90,91,92,93,94] |
| 3. | top\_n\_words | [10,11,12,13,14,15,16,17,18,19,20] |

##### Table 1 List of hyperparameter values used in optuna for BERTopic

We are passing parameter n\_trials = 3. This means that it will try a different set of values from the specified value in 3 runs.

#### 4.5.1.4 Calculating Accuracy, Precision, Recall and F1

To calculate the performance metrics for the topic model, we perform following steps:

1. Manually label the tweets in the dataset: We have manually labelled around 300 tweets in the dataset and assigned a topic to each tweet using a word for each topic.
2. Creating an instance of the model: Then these 300 tweets are converted into a list of texts and an instance of BERTopic model is created. We assigned embedding\_model= “paraphrase-MiniLM-L3-v2”, calculate\_probabilities= TRUE, verbose= TRUE, values for the parameters for BERTopic instance
3. Fitting the model: This model is then fitted into these labelled tweets.
4. Representing output: Representation for each topic is obtained as well as the topics assigned by the model to each document. We will also get 4 words representative of each topic and then assign a word to each tweet.
5. Assigning topic labels to the predicted output: We will append the output of BERTopic for each topic into the dataframe. For each document we will map the topics assigned by BERTopic to the representative words for that topic. This is done by map\_topic function. A user defined function check\_value\_in\_list() which takes the label and the predicted output as the parameter and checks if the labelled topic word is in the words present in for the topic representative words predicted by the model, we will assign the label as the predicted topic. If not present, we will assign the words with highest probabilities in the topics (i.e. it occurs in higher probability in the documents for each topic) as the predicted topic.

The performance of the model is measured by accuracy, precision, recall, F1 and confusion metrics. These metrics can be used to assess how well the model predicts the topics. Performance metrics gives insight into the accuracy, precision, recall and other measures which are measured from the values of True Positive, True Negative, False positive, False Negative. We can calculate the accuracy, precision, recall and F1 score for the tweets using accuracy\_score, precision\_score, recall \_score, f1\_score respectively from scikit learn library. For calculating each score, we are passing the predicted and label values. The parameter average= “weighted” specifies how the scores for each class should be combined to obtain an overall metric. The "weighted" average calculates the score for each class individually and then combines them, giving more weight to classes with more instances. The weight is determined by the proportion of instances in each class relative to the total number of instances.

To analyze the performance of the topic model for each topic predicted by the model we will use an in-built function classification report from scikit-learn library. We will pass the label and predicted value as the parameter. The classification report provides a more detailed and comprehensive report of all four scores for each class. It helps us to understand how good the model performs for the imbalanced dataset.

**4.5.2 Latent Dirichlet Allocation**

LDA focuses on finding the latent topics from the document using the Bag-of-Word representation. It considers the frequency of the words in the documents and ignores the order of the words. So, we need to generate BoW representations for the tweets in the dataset. To convert text into BoW representations we use "en\_core\_web\_lg" model from the spaCy library. SpaCy is a popular NLP library. This model is a pre-trained English language model that provides word vectors. We need to tokenize the tweets into individual tokens by splitting on whitespace, punctuation, and special character. It is done by in-built tokenizer present in spacy, it tokenizes the text based on the vocabulary present in spacy library. After performing tokenization, the tokens are converted into lowercase and then stop words are removed. We then rejoin the tokens and lemmatize the rejoined statement. Then we tokenize the statement obtained after lemmatization to remove all the links, hashtag, punctuations and generate the tokens for every tweet in the dataset. So, we obtained pre-processed and filtered word tokens that can be converted to BoW representation.

We are defining the function create\_dictionary to create the dictionary out of the tokenized words. We are using Gensim library, an open-source library for topic modelling, document indexing and similarity retrieval with large corpora. To map words to unique integer IDs in a corpus we are generating Gensim “Dictionary” object id2word\_lda. It processes a list of words in this case the list of tokenized words and map those tokens to unique integer IDs.

After creating dictionary id2word\_lda, we will create corpus\_l, a BoW representation of the tweets in the dataset. To convert it into BoW representation it applies Gensim doc2bow method on the tokenized text. Doc2bow represents words in documents as a list of (word\_id, word\_frequency) pairs by converting a list of tokens (words) into a format that counts the frequency of each word in the document. On each text document it applies doc2bow method using id2word\_lda dictionary. Thus, BoW representation for the entire corpus is generated.

After the corpus and dictionary is generated, an instance of LdaMulticore is created from Gensim library. It trains the LDA topic model using multiple cores for faster processing. We pass parameters corpus= corpus\_l, num\_topics=94, id2word=id2word\_l, workers=12, passes=5. The corpus here represents the corpus of documents on which LDA model is trained, num\_topics represent number of topics LDA should generate. The parameter id2word specifies the dictionary on which model should be trained. The parameter workers specify the number of CPU cores to be used to train the model and pass represent the number of times the model will iterate over the entire corpus during training. The trained model is stored in the variable, using which we can explore the topics and topics assigned to each document.

#### 4.5.2.1 Visualization

For visualizing the topics generated we will use python library pyLDAvis. This library is used for interactive topic model visualization. We will be visualizing the similarity between the topic using intertopic distance map. The intertopic distance map visualizes the relationship between the topics based on their similarity. We will also visualize the most relevant words for the given topics using wordcloud. We will visualize top 3 topics using the wordcloud. We will also use print\_topics() method to show the most significant topics that are discovered by the LDA topic model.

#### 4.5.2.2 Coherence and Perplexity Score

Perplexity is the measure which indicates the performance of topic model on how well it predicts the given or unseen data. LDAMulticore has an in-built function log\_perplexity() to calculate log perplexity of the LDA model. It has a parameter corpus which indicates the corpus on which model is trained.

For evaluating coherence of LDA topic model, we will use CoherenceModel from Gensim library. It builds a model to evaluate topic coherence score. The constructor takes trained topic model, dictionary provided the model doesn’t contain one, as the parameter. Since we are using c\_v score measure to calculate coherence score, we need to pass tokenized text on which model is trained as parameter. For evaluating perplexity of LDA model, we will use an in-built function called log\_perplexity which takes corpus as parameter

#### 4.5.2.2 Hyperparameter Tuning

To determine the optimal values for each parameter of the model, we will use the Optuna framework for hyperparameter tuning. Optuna is an automated software framework designed for hyperparameter optimization. We have opted for Optuna due to its capability to dynamically construct a search space for hyperparameters. In the section we have explained the two basic concepts of optuna trial and study. We will find the optimal values for parameter num\_topics, alpha and eta.

* The parameter num\_topics specifies the number of latent topics the model should discover from the training corpus.
* The parameter alpha specifies the Dirichlet priori belief on document-topic distribution and where optimal value for alpha is 50/T where T is number of topics. A lower value of alpha suggests that smaller number of topics in document mixture and higher value indicates higher number of topics in the document.
* The parameter eta indicates the Dirichlet priori belief on topic-word distribution and the optimal value is 0.1 or 200/W, where W is number of words in vocabulary. A lower value of eta suggests that topics will have fewer representative words and the higher value of eta suggests that more words will be representing the topics.

The range of possible values for each hyperparameter is shown in table 2

|  |  |  |
| --- | --- | --- |
| S No. | Parameters | Values |
| 1. | num\_topics | [90,94] |
| 2. | alpha | [0.01, 1.0] |
| 3. | eta | [0.01,1.0] |

##### Table 2 List of hyperparameter values used in optuna for LDA

We are passing parameter n\_trials = 10. This means that it will try a different set of values from the specified value in 10 runs.

#### 4.5.2.3 Calculating accuracy, precision, recall, F1

To calculate the performance metrics, we will use the same labelled dataset which was used to evaluate BERTopic. First, we will convert the categorical labels into integer values using LabelEncoder from the scikit library. We will add the encoded value for each label into dataframe. Then we will divide the dataset into train and test set using train\_test\_split in the ratio of 70% and 30% for train and test dataset. Then we will tokenize the twitter and remove the stopwords, links, hashtag, punctuations and then lemmatize the tokenized text for both train and test dataset.

After preprocessing the train and test dataset, a dictionary will be created using the tokens in train dataset by implementing the function create\_dictionary. After creating dictionary, a corpus for train and test dataset is created to convert text into BoW representation. This is by applying the doc2bow method on the tokenized data.

After creating corpus and dictionary for data, we will train an LDAMulticore model on training corpus and the training dictionary. We will pass num\_topics=6, workers=12, passes=5 as the parameter while creating an instance of LDAMulticore. After training we will get the 6-topic distribution for each tweet. Then for each topic discovered we will find top 10 words that represent the topics and create a dictionary with topic id as key and the list of top 10 words as the values for each topic. After, this we will find the topic distribution for each document and assign the topic with highest probability as the main topic for each document and append this to dataframe. After this we will map each topic assigned to document to a list of representative words for that topic and create a separate column in the dataframe. A user defined function check\_value\_in\_list() which accepts label and the predicted output as the parameter and checks if the labelled topic exists among the words identified as representative of the topic by the model, If the labeled word is found in the representative words, the function will assign the label as the predicted topic. However, if the labeled word is not present, the function will select the words with the highest probabilities within each topic, as these words are more likely to appear frequently in the documents for each respective topic, and assign this set of words as the predicted topic instead.

We will then evaluate the output of the classifier using performance metrics. We will evaluate accuracy, precision, recall and F1 using the scikit library. When computing each score, the predicted and label values are provided as input. The average="weighted" parameter determines the way the scores from individual classes are aggregated to yield an overarching metric.

### 4.5.3 Top2Vec

Top2Vec leverages the joint document and word semantic embedding to find the underlying hidden topics in the documents. We will be using the Top2Vec core class from Top2Vec library for training the Top2Vec model and finding the underlying topics in the tweets. We will create an instance of Top2Vec model and initialize the model by passing the tweets to be analyzed. The model utilizes semantic embedding and cluster documents into hierarchical manner.

#### 4.5.3.1 Visualization

To visualize the topic generated by Top2Vec model, we will use wordcloud to visualize the prominent words for each topic. We will try to visualize the most relevant word for each topic generated by top2vec model.

#### 4.5.3.2 Coherence Score

To calculate the coherence score we will use the corpus and the dictionary generated from the tokens in the LDA model. Then we will extract the words relevant for each topic identified by the Top2Vec. We will only consider the words relevant to the topic and ignore each word score and topic number. We will convert all the words representing a topic into a list of lists and treat each element in list as the topics generated by the model. To calculate coherence score, we will initialize an instance of CoherenceModel. We will pass a list of topics, text on which topic model was trained, corpus and dictionary generated from the dataset as the parameters. We are using c\_v measure to evaluate the coherence score.

#### 4.5.3.3 Hyperparameter Tuning

For hyperparameter tuning we are using optuna framework. Optuna has been discussed in the above section. We are using optuna to perform hyperparameter tuning because it efficiently searches large area for hyperparameters. We will discuss the hyperparameter that is optimized, and the values passed for each hyperparameter.

* min\_count : This parameter specifies frequency of the occurrence of words. The words with total frequency less than this parameter will be ignored. The default value for min\_count is 50. For smaller corpora the value for this parameter should be small
* workers: This parameter specifies the number of threads to use to train the model. The higher value means more threads for training which will lead to faster training.

The range of possible values for each hyperparameter is shown in table 3

|  |  |  |
| --- | --- | --- |
| S No. | Parameters | Values |
| 1. | min\_count | (5,20) |
| 2. | workers | (4,8) |

##### Table 3 List of hyperparameter values used in optuna for Top2Vec

We are passing parameter n\_trials = 10. This means that it will try a different set of values from the specified value in 10 runs.

**Chapter 5**

# RESULTS & ANALYSIS

We performed topic modelling to detect the underlying topics in the tweets. This task is done using 3 topic models BERTopic, LDA and Top2Vec. In this chapter we will cover the results for all the three topic models and analyze the result to answer the three research questions. This chapter is divided into four sections. Firstly, we will investigate the results obtained by implementing topic modelling on Twitter dataset using BERTopic. In the second section, we will discuss the output of LDA model trained on the dataset. In the third section we will look into the results of Top2Vec. In the last section we will analyze and compare the results of three models to answer the three research questions. All three topic models were performed on the 5000 tweets on US presidential election 2020

## 5.1 BERTopic

BERTopic was executed on the 5000 tweets. The output of the BERTopic contains the discovered topics, information about generated topics, keywords associated with each topic, and documents allocated to each topic. The model generated 90 topics for 5000 tweets. The first topic with topic\_id =-1 represents the ‘outlier’ or ‘noise’ documents that topic model couldn’t confidently allocate the topic to. The rest of the 90 topics are the topics that topic model assigned to the documents.

### 5.1.1 Intertopic Distance Map

The topics generated by the BERTopic and the relationship between different topics in multi-dimensional space can be visualized using intertopic distance map. Semantically similar topics are placed closer to each other on the map. Clusters of topics represent groups of related topics; it suggests the topics cover semantically similar content. The 90 topics are clustered into 13 clusters which means many topics are semantically similar and related topics. The intertopic distance map for BERTopic is shown in Fig 9.

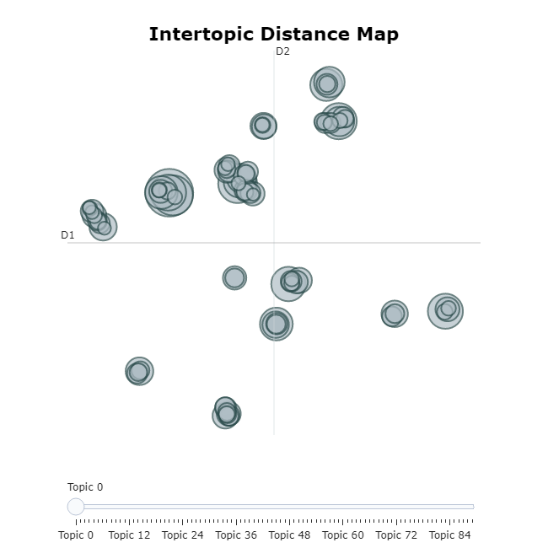


Fig 9: *Intertopic distance map for topics generated by BERTopic*

### 5.1.2 Hierarchical Clustering

To visualize the hierarchical clustering, the clusters and the merging of clusters can be represented by dendrogram. The distance at which the clusters merge indicates the distance at which similar topics merge. The clusters merging at smaller distance means the topics are more similar. The vertical line represents the clusters formed at different levels. The length of vertical lines indicates the size of cluster. In the figure we can see that topics that are more or less likely related to covid-19 infection and vaccination are clustered together. Topics that are concerned about racism are denoted together and form one cluster. We can see from figure 10 that the general affairs like Iowa rally, corruption, stimulus package and so on which were concerned with Donald Trump, one of the presidential candidates are grouped together. The topics that discussed the social media, media and foreign affairs were denoted together under one category. The anti- Trump topics are grouped together.

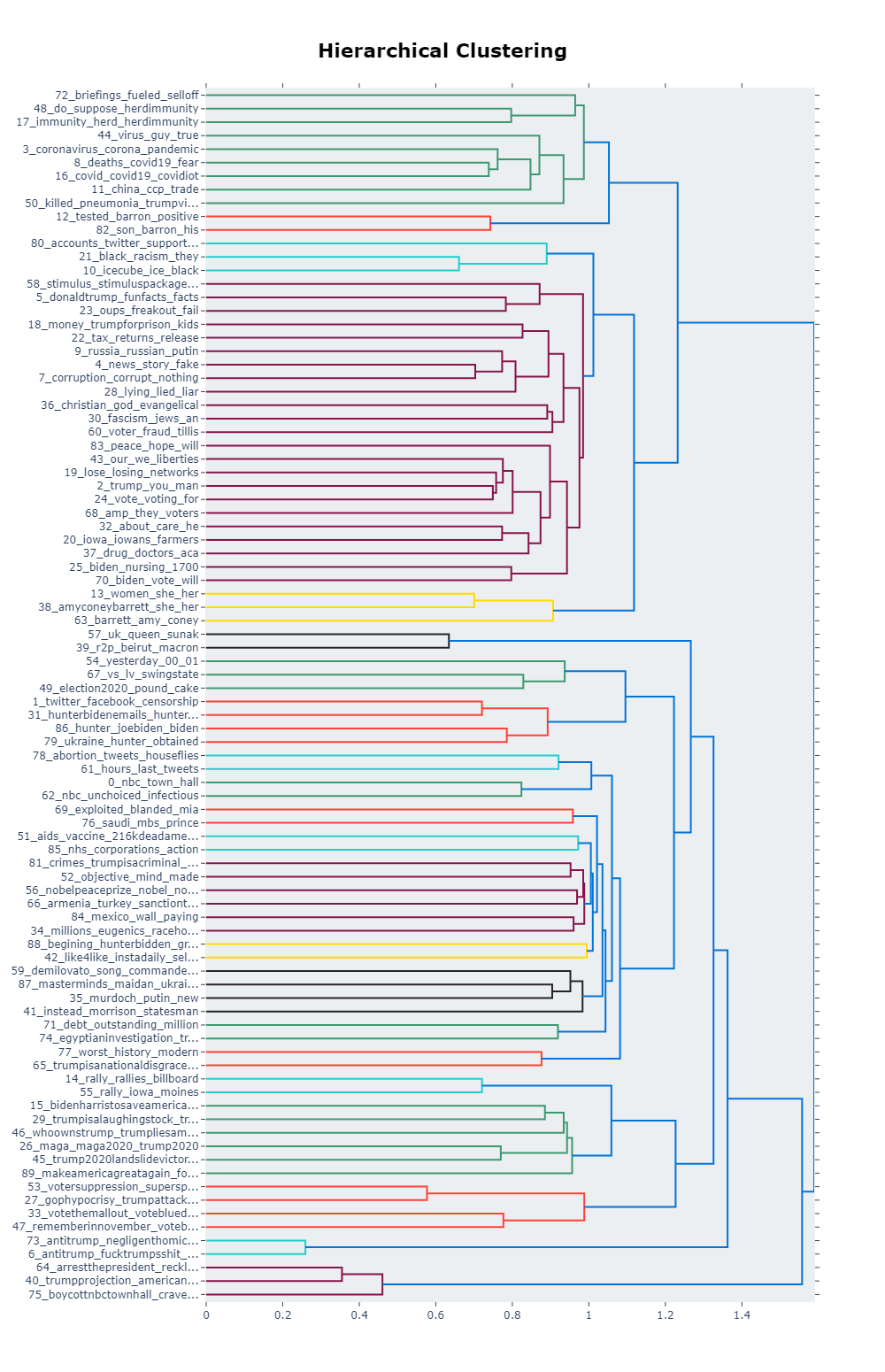


Fig 10: *Hierarchical Clustering for topics obtained by BERTopic*

### 5.1.3 Barchart Representation and Word Cloud

We will understand the distribution of each word in the topics by visualizing the probabilities of each word using the barchart for the top 15 topics. We can see from fig 11 that representative words with higher probabilities in top 15 topics are nbc, twitter, trump, coronavirus, news, donaldtrump, antitrump, corruption, deaths, russia, icecube, china, tested, women, rally. This able to understand what the topic is concerned about.

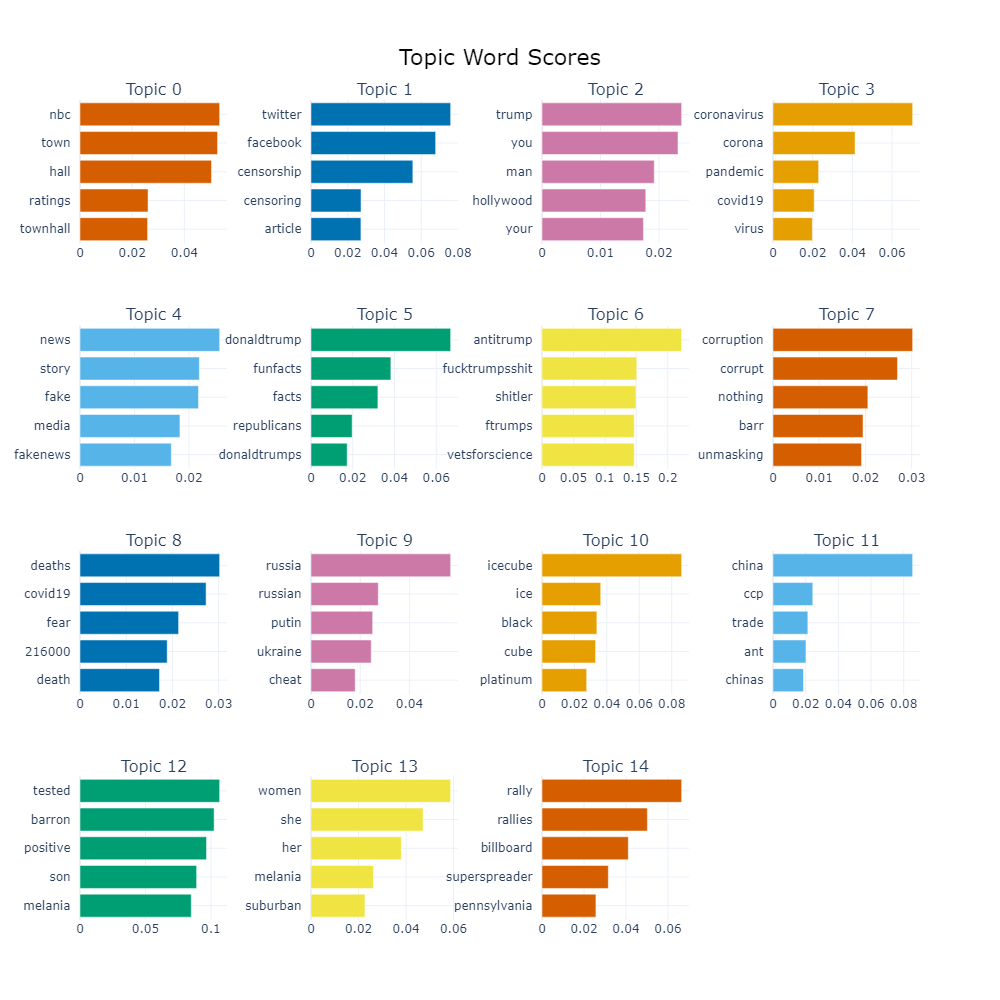


Fig 11: *Barchart for representative words for topics obtained by BERTopic*

Word Cloud helps us to understand keywords visually representing the most frequent words within each topic. We obtained wordcloud for the top 3 topic which is shown in Fig 12. From word cloud we can see that twitter, trump, coronavirus are the words that occur frequently in the documents which represent the topics.

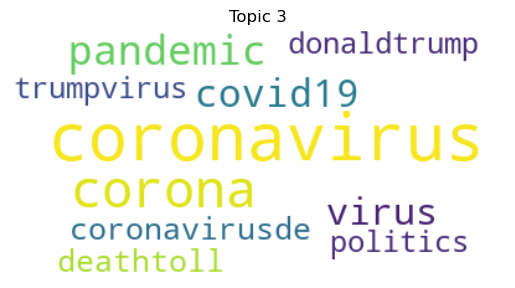
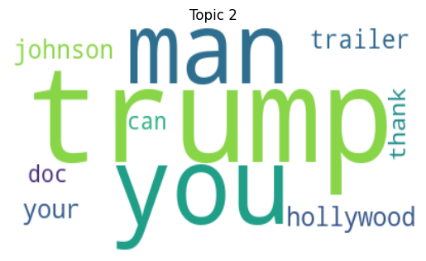
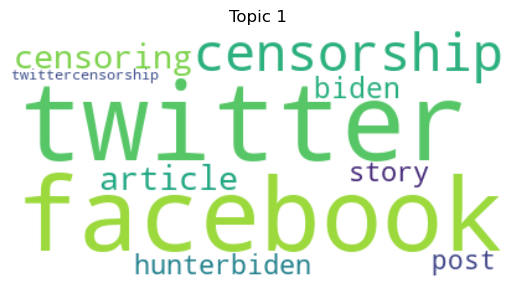


Fig 12: *Word Cloud for topics obtained by BERTopic*

### 5.1.4 Coherence score and Performance metrics

Coherence score for the BERTopic without hyperparameter tuning is obtained as 0.744898 which means the topics obtained as the output of the BERTopic is more coherent and interpretable to the human. After performing hyperparameter tuning the coherence score for the optimal value of parameter increased to 0.763571. The optimal value of parameters is min\_topic\_size': 8, 'nr\_topics': 94, 'top\_n\_words': 10. Fig 13 and Fig 14 shows the output of hyperparameter tuning and the coherence score before and after hyperparameter tuning.



Fig 13: *Optimal values for parameters for BERTopic after hyperparameter tuning*

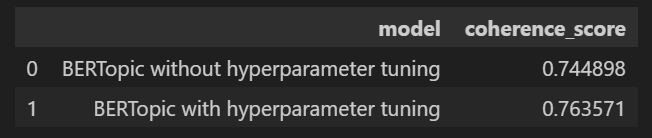


Fig 14: *Coherence score obtained for BERTopic*

We also have evaluated the BERTopic based on the accuracy, recall, precision, F1 and classification report of the model for 300 tweets. BERTopic achieved accuracy, recall, precision, F1 as 0.88, 0.87, 0.88 and 0.869 respectively which is shown in Fig 15.

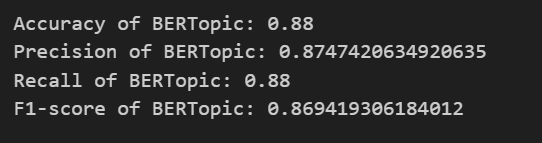


Fig 15: *Accuracy, Precision, Recall. F1 obtained for BERTopic*

The classification report for the BERTopic obtained is depicted in Fig 16.

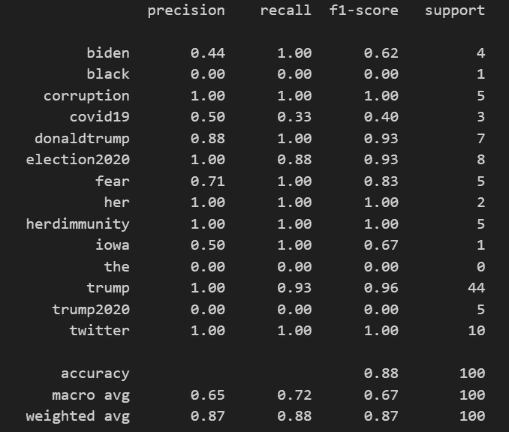


Fig 16: *Classification Report obtained for BERTopic*

The confusion metric we got for BERTopic is shown in Fig 17. The 14 topics generated is represented as integer numbered from 0 to 13

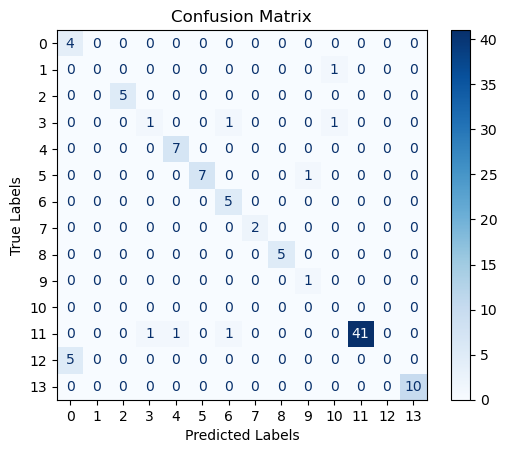


Fig 17: *Confusion metrics obtained for BERTopic*

**5.2 LDA**

In this section we will visualize the topic distribution generated by the LDA topic model. LDA was trained on the dataset of 5000 tweets. The LDA generated the topic, and words representing the documents. We evaluated the LDA model using performance metrics and visualized the output of LDA which is discussed in the following section.

### 5.2.1 Topics

LDA generated topic distribution for each document and the topic-word distribution. We will look into the top 10 that were generated by LDA and the representative word distribution for each of these topics. In Fig 18 shows the top 10 and the word distribution for each topic.

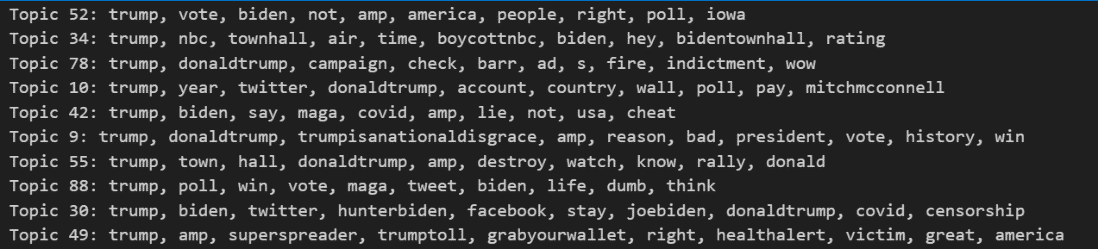
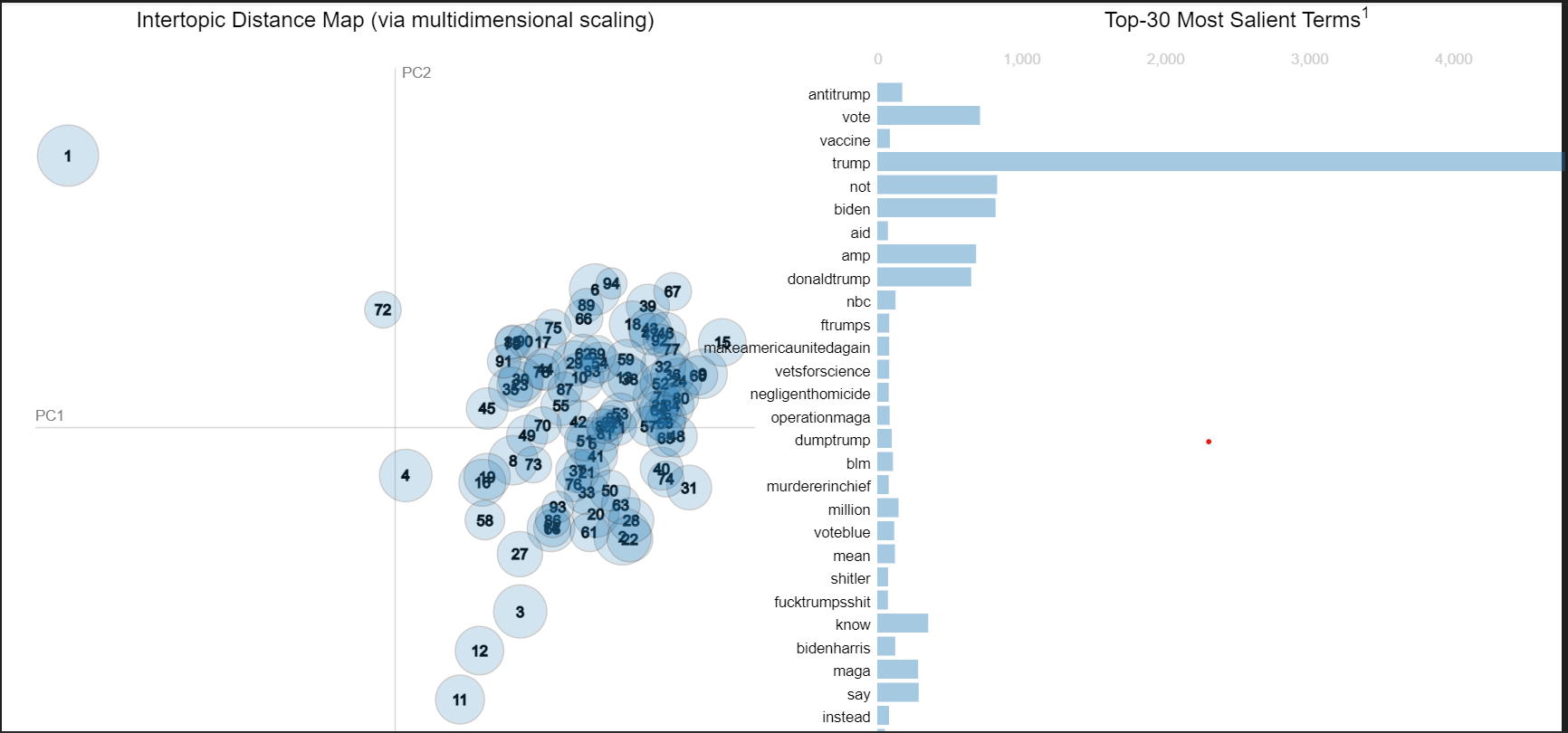


Fig 18: *Topics and representative words achieved by LDA*

### 5.2.2 Intertopic Distance Map

Intertopic distance map provides insight into word distribution per topic and the relationship between topics. Semantically similar topics are placed closer to each other on the map. The distance between clusters represents the cosine similarity between the topics. The most representative term for each topic is depicted by Top-30 Most Salient Terms. Fig 19 represents the intertopic distance map.



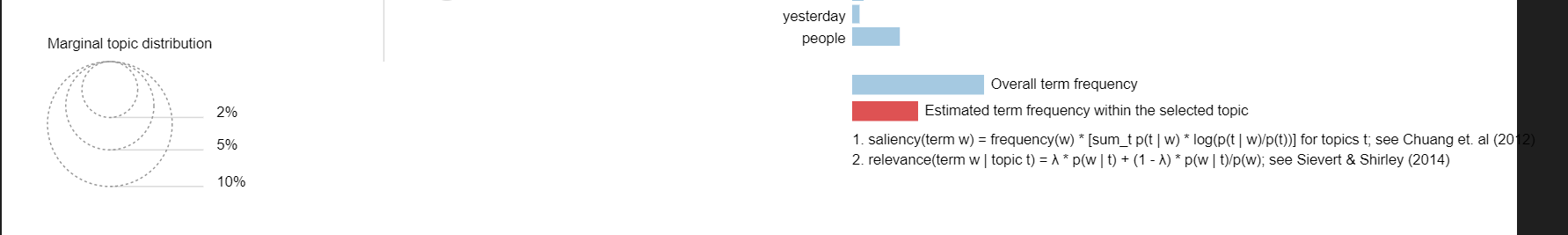
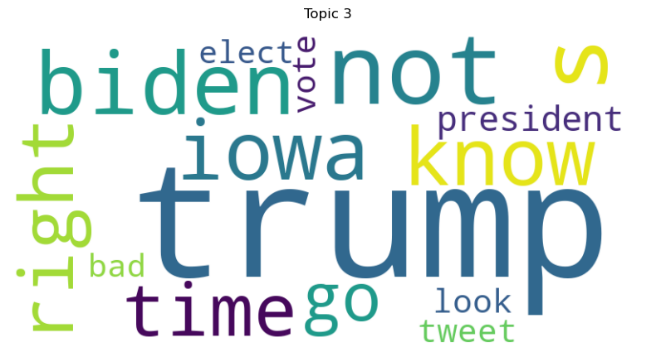
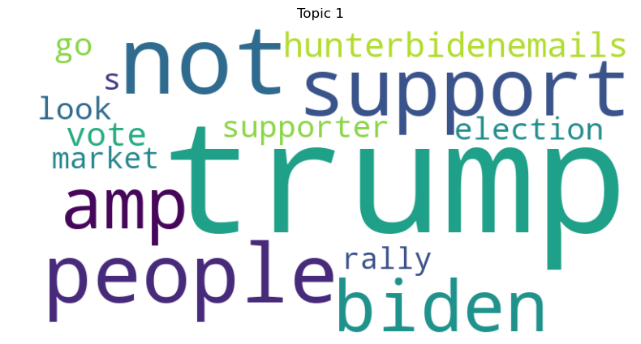


Fig 19*: Intertopic Distance map for topics obtained by LDA*

### 5.2.3 Word Cloud

We visualize the top 3 topics and the representative words of the three topics as the word cloud. Fig 20 shows the three wordcloud for the top 3 topic. The size of each word in the cloud reflects its frequency or weight within the topic. We can observe that in Topic 1, Topic2, topic3 the keyword trump is the word that occur in the highest frequency in all 3 topics.



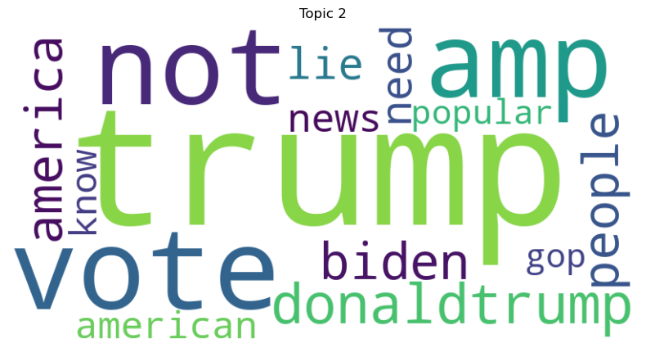


Fig 20: *Word Cloud for the topics discovered by LDA*

### 5.2.4 Coherence score and Performance metrics

The quality of topics generated by the topic model is evaluated on the basis of coherence score. It evaluates the interpretability of topics generated by topic model. The coherence score obtained without optimizing hyperparameters is 0.3913 and the coherence score achieved after optimizing the parameters is 0.395032 which is represented in Fig. The optimal value of the hyperparameters is num\_topics': 92, 'workers': 12, 'passes': 7. The fig 21 and fig 22 shows the output we obtained after tuning hyperparameter and calculate the coherence score for both values.



Fig 21: *Optimal values for hyperparameters of LDA*

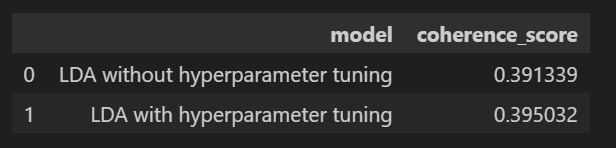


Fig 22: *Coherence score for LDA*

The performance of the LDA model was assessed using key evaluation metrics including accuracy, recall, precision, and F1-score. These metrics provide insights into the effectiveness of the model's classification of documents into topics. Our LDA model achieved an overall accuracy of 0.64, indicating that 64% of the documents were correctly assigned to their corresponding topics. The recall score of 0.64 signifies that the model successfully identified 64% of the relevant documents for each topic. The precision value of 0.49 indicates that when the model assigned a document to a topic, it was correct 49% of the time. The F1-score, which considers both precision and recall, was 0.53, suggesting a balance between accurate classification and comprehensive coverage of relevant documents. Fig 23 and Fig24 shows the output of calculating performance metrics and confusion metrics respectively.

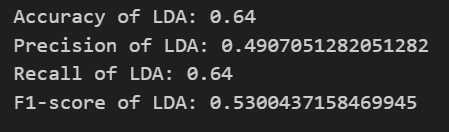


Fig 23: *Accuracy, Precision, Recall. F1 obtained for LDA*



Fig 24: *Classification Report obtained for LDA*

We have trained LDA model on 300 tweets and split it in test and train set in the ration of 80:20. We performed LDA and LDA identified 5 topics among 300 tweets. We got the classification and report and the confusion matrix shown in fig 24 and fig 25 which indicates how accurately each class is identified. We can see model performed best for topic related to biden,fear and twitter. The values we got for each class is different than expected since the LDA only consider BOW representation and doesn’t look into semantic meaning, so words with high frequency are taken as representative words this leads to low precision, recall and F1 value.

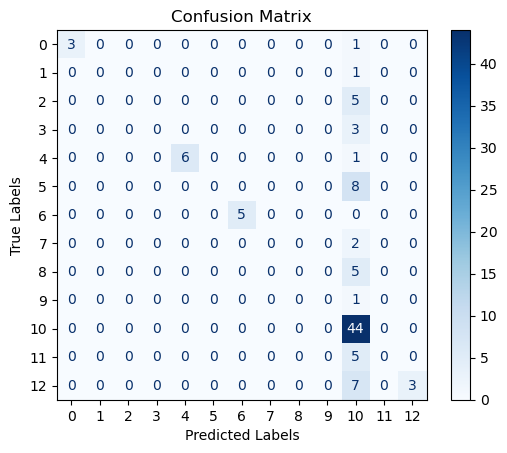


Fig 25: *Confusion metrics obtained for LDA*

## 5.3 Top2Vec

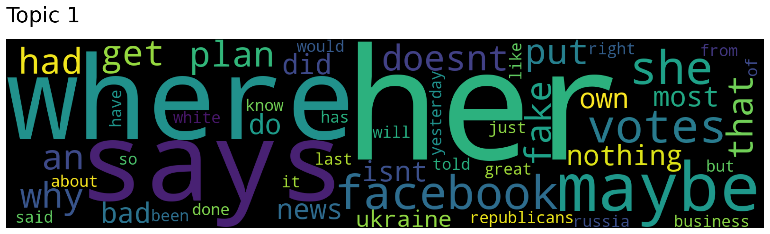
In this section we will look into the output obtained for visualizing Top2Vec topics. Top2Vec model was trained on the given dataset. We evaluated the Top2Vec model on the basis of the coherence score.

### 5.3.1 Topics generated

LDA generated 6 topics for the 5000 tweets. The Topic 1 is regarding the trump and his policies. Topic 2 identifies the crimes and issues related to trump. Topic 3 is all tweets about antitrump. Topic 4 is all about covid 19 and the government policies about the vaccination. Topic 4 is

### 5.3.2 Word Cloud

We are visualizing the top 3 topics and the words that represent each of these 3 topics. Fig 26 shows the word cloud for each of these topics. The keyword ‘her’ occurs in the higher frequency in topic 1, ‘shitler’ occurs most in topic 2. In topic 3 has keyword ‘come’ occurs the most.





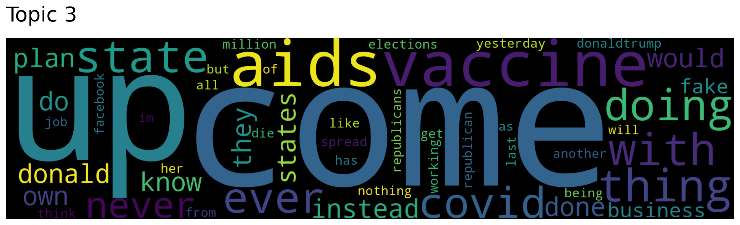


Fig 26: *Word Cloud for representative word of top 3 topics in Top2Vec model*

### 5.3.3 Coherence Score

We are calculating the coherence score to evaluate the top2Vec. The coherence score obtained without performing hyperparameters tuning is 0.446822 and the coherence score achieved after training LDA with optimal values of hyperparameter is 0.540627 which is represented in Fig. The optimal value of the hyperparameters is 'min\_count': 16, 'workers': 4. Fig 27 shows the optimal value of the parameters.



Fig 27: *Optimal values of hyperparameters of Top2Vec*

The fig 28 shows the output we obtained after tuning hyperparameter and calculate the coherence score for both values.

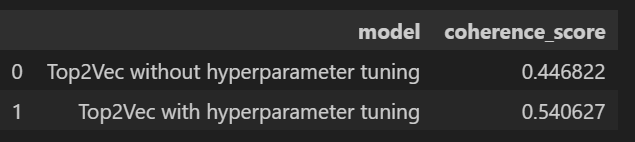


Fig 28 : *Coherence score of Top2Vec with and without hyperparameter tuning*

## 5.4 Comparison of BERTopic, LDA and Top2Vec

In this section we will compare all three topic models to answer our research questions.

For the first questions, we perform BERTopic on the 5000 tweets. BERTopic identified 90 topics among the tweets. This answers our first research questions to find the underlying themes that were discussed for the US presidential election 2020 among the people on Twitter.

To compare and analyze the performance of each topic model we calculate the coherence score of each topic model with optimal parameters. In Fig 29 shows the barchart that compares the coherence score for each topic model.

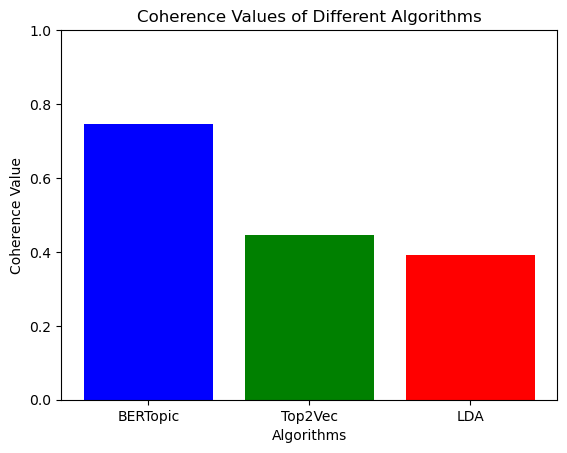


Fig 29: *Comparison of Coherence score for each of the topic models using Barchart*

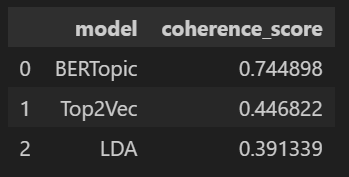


Fig 30: *Coherence score of BERTopic, LDA and top2vec*

From the figure 30 we can see that BERTopic has the highest coherence score than the other two models. This indicates that BERTopic gives a more coherent topic than Top2Vec, LDA. It also ensures that BERTopic gives better topic qualities. This is because BERTopic leverages pre-trained BERT based embedding model to generate high-quality semantic embeddings that capture the contextual meaning and relationships between words in a more sophisticated way compared to other topic models. Top2Vec has a higher coherence score than LDA because it utilizes doc2bow representation to generate joint document-word embedding. This takes order of the words and semantic meaning into account to discover a coherent topic. So, these answers the second research question about performance of each topic model.

For the third question whether we will achieve better topic modeling using contextual embedding topic models than traditional topic model. We will evaluate the contextual topic model i.e. BERTopic and traditional topic model LDA using performance metrics. We will compare the results of BERTopic and LDA using accuracy, precision, recall, F1. We will plot a barchart for these scores. Fig 32 shows barchart and Fig 31 shows the values of the performance metrics.

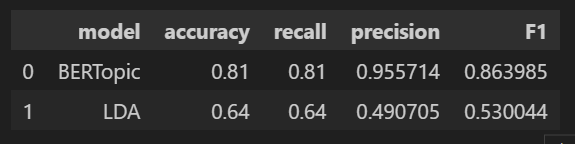


Fig 31: *Accuracy, recall, precision and F1 for BERTopic, LDA*

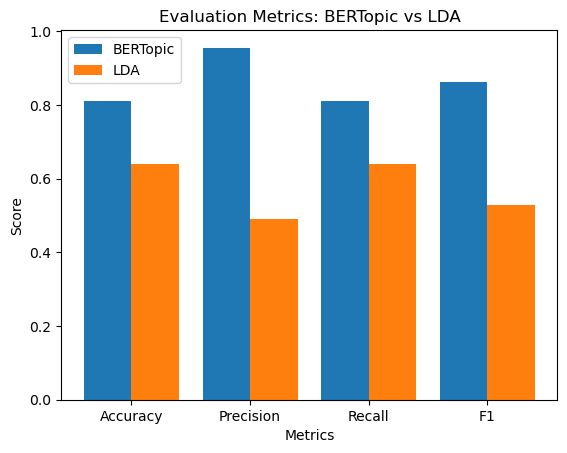


Fig 32*: Barchart for comparison of BERTopic and LDA accuracy, recall, precision and F1*

From barchart we can see that BERTopic gives a better score than the traditional LDA approach. Accuracy, precision, recall and F1 score of BERTopic is higher than LDA model. This suggests that BERTopic accurately assign topics to each document rather than LDA. This is due to the reason that BERTopic utilizes embedding that captures semantic meaning of the tweets whereas LDA uses BoW representation that doesn’t take word order into account so lower values of accuracy, precision, recall and F1. This helps us to understand that contextual embedding topic model performs better than traditional approaches and answers third research questions.

## 5.5 Limitations

Comparative assessments of topic modelling approaches such as BERTopic, Top2Vec, and LDA are critical for understanding their strengths and drawbacks. However, it is critical to recognize the limitations of such analyses in order to guarantee that the results are accurately interpreted and applied. Here are some of the drawbacks of these methodologies' comparative analyses:

1. Data Dependence: The performance of topic modelling methods is strongly dependent on the type and quality of the data. When applied to different datasets, comparative analyses may produce varied results, making it difficult to generalize conclusions.
2. Sensitivity of the parameter: Each subject modelling approach has a unique set of hyperparameters that must be optimized for best performance. Small adjustments in these factors significantly affect results.
3. Coherence Metrics: Coherence metrics are frequently used in comparative assessments to assess topic quality. These measurements have limits and may not fully capture a topic's interpretability and importance.
4. Interpretability: Coherence metrics do not take into account topic interpretability, which is critical for many applications. Comparative assessments may fail to reflect how easily users may comprehend and apply the generated themes.
5. Model complexity: Top2Vec, and LDA have varying levels of complexity. BERT-based models, for example, are more computationally intensive. So, performing topic modelling in a large dataset is time consuming and requires more computational resources.

These are some of the limitations that we were facing while performing the comparative analysis of the topic models.

**Chapter 6**

# CONCLUSION & FUTURE WORKS

## 6.1 Conclusion

Topic modelling is an NLP technique to uncover the underlying themes or topics from the corpus of text documents. It is a valuable tool for extracting meaningful insights from large volumes of text data. Discovering the hidden topics is important to give insight into the documents and understand the theme of the document. The aim of this project is to compare and analyze the three-topic model BERTopic, Top2Vec and LDA and find the underlying topics in tweets related to 2020 US Presidential election. We performed the topic modelling on 5000 tweets for 2020 US Presidential election. We generated the topic, and topic-word distribution. Finally, this research conducted a thorough comparison of three popular topic modelling techniques: BERTopic, LDA (Latent Dirichlet Allocation), and Top2Vec. These models were evaluated using a variety of measures, including coherence, Scores. We also evaluated performance of contextual topic model (BERTopic) and traditional topic model (LDA) using performance metrics.

Our examination into coherence found that BERTopic has a considerable advantage in developing coherent topics, implying that it has the potential to generate more interpretable and contextually relevant topic clusters. Top2Vec produced a competitive result in this regard, showing its continuous relevance and usefulness in topic modelling tasks. While LDA showed potential in terms of identifying semantically relevant themes, LDA, while being a classic approach, trailed behind the other two models in terms of coherence. And while investigating performance metrics it was noted that the contextual topic modelling BERTopic produced more accurate topics for each tweet than LDA. This points to its utility in applications requiring exact topic categorization, such as information retrieval systems. Accuracy, recall, precision, and F1-score were used to evaluate the models' topic classification performance.

It is critical to emphasize that the best subject modelling technique depends on the specific requirements of the work at hand. BERTopic performed admirably in terms of coherence and precision, making it an excellent candidate for applications requiring interpretable and exact topic representation. Top2Vec revealed its strength in capturing semantic links but may require more development to improve coherence. LDA proved to be a reliable and interpretable technique, appropriate for different text analysis jobs, whereas LDA proved to be a reliable and interpretable method, suitable for various text analysis tasks.

In conclusion, this dissertation contributes valuable insights into the comparative analysis of BERTopic, LDA, and Top2Vec, emphasizing the importance of coherence and contextual embeddings in topic modeling. Researchers and practitioners can leverage these findings to make informed decisions regarding the selection of a topic modeling technique based on their specific project needs. Furthermore, ongoing research and development in the field of natural language processing will likely continue to refine and expand the capabilities of these models, offering exciting opportunities for future advancements in topic modeling.

## 6.2 Future Works

There are some intriguing routes for additional research and exploration in this domain based on the comparative analysis of BERTopic, LDA, and Top2Vec applied to 2020 US presidential election tweets:

* **Real-time Election Monitoring**: Create a real-time election monitoring system that tracks and analyses tweets during future elections using these topic modelling techniques. This would provide for timely insights into changing public attitude and debate.
* **Multi-lingual and Cross-cultural Analysis**: Extend the analysis to include tweets in different languages and investigate how these topic modelling tools work in cross-cultural contexts, particularly during international elections or global events.
* **Social Network Analysis**: Incorporate network analysis into the research framework to investigate interactions on social media platforms between users, political candidates, and themes. Examine the function of influencers in the dissemination of election-related content.
* **User Profiling**: Create ways for creating user profiles based on the content of their tweets during an election. This can help us better understand voter demographics and preferences.
* **Multi-modal Data Fusion**: Combine textual analysis with image and video content analysis to gain a thorough knowledge of election-related social media information.

Researchers can contribute to the advancement of topic modelling techniques in the context of election analysis by pursuing these additional works, providing valuable insights into the dynamics of political discourse on social media and its implications for democratic processes and political decision-making.

# REFERENCES

1. A.A.A. Karim and R.A. Sameer, “Image Classification Using Bag of Visual Words (BoVW),” Journal of Al-Nahrain University-Science, vol. 21, pp. 76-82, 2018.
2. Quoc V Le, Tomas Mikolov, “Distributed Representations of Sentences and Documents”, Proceedings of the 31 st International Conference on Machine Learning, Beijing, China, 2014. JMLR: W&CP volume 32
3. David M. Blei, Andrew Y. Ng, Michael I. Jordan, “Latent Dirichlet Allocation”, Journal of Machine Learning Research 3 (2003) 993-1022, 102
4. Thomas L. Griffiths and Mark Steyvers. 2004. Finding scientific topics. Proc. Nat. Acad. Sci. 101, suppl 1 (2004), 5228– 5235. DOI:https://doi.org/10.1073/pnas.0307752101
5. Uttam Chauhan, Apurva Shah, “Topic Modeling Using Latent Dirichlet allocation: A Survey” ACM Computing Surveys, Vol. 54, No. 7, Article 145, 2021
6. Josip Kunsabo and Jasminka Dobša, “A systematic literature review on topic modelling and sentiment analysis” , Conference: International Scientific Conference Central European Conference on Information and Intelligent Systems, 2022
7. Dimo Angelov, “Top2Vec: Distributed Representation of Topics”, arXiv:2008.09470 , 2020
8. Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. arXiv preprint arXiv:1802.03426, 2018
9. Claudia Malzer, Marcus Baum, “A Hybrid Approach To Hierarchical Density-based Cluster Selection”, 2020 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI), 2020
10. Maarten Grootendorst, “Bertopic: Neural topic modeling with a class-based tf-idf procedure”, 2022
11. Maarten Grootendorst. Bertopic algorithm. https://maartengr.github.io/BERTopic/algorithm/ algorithm.html#visual-overview, 2021. Accessed: July 21, 2023.
12. Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks, 2019
13. David Alfred Ostrowski, “Using Latent Dirichlet Allocation for Topic Modelling in Twitter”, Proceedings of the 2015 IEEE 9th International Conference on Semantic Computing,2015
14. Phillip Ma , Qing Zeng-Treitler and Stuart J. Nelson, “USE OF TWO TOPIC MODELING METHODS TO INVESTIGATE COVID VACCINE HESITANCY”, International Conferences ICT, Society, and Human Beings, 2021
15. Toshihi Koyanase , “Optuna: A hyperparameter optimization framework”, Github, https://github.com/optuna/optuna