12-2021

Enhancing the Performance of Text Mining

Farah Mahmoud Al Shanik
falshan@g.clemson.edu

Enhancing the Performance of Text Mining

A Dissertation
Presented to
the Graduate School of
Clemson University

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy
Computer Science

by
Farah Mahmoud Alshanik
December 2021

Accepted by:
Dr. Amy Apon, Committee Chair
Dr. Ilya Safro, Co-chair
Dr. Alexander Herzog
Dr. Nina Hubig

Abstract

The amount of text data produced in science, finance, social media, and medicine is growing at an unprecedented pace. The raw text data typically introduces major computational and analytical obstacles (e.g., extremely high dimensionality) to data mining and machine learning algorithms. Besides, the growth in the size of text data makes the search process more difficult for information retrieval systems, making retrieving relevant results to match the users' search queries challenging. Moreover, the availability of text data in different languages creates the need to develop new methods to analyze multilingual topics to help policymakers in governmental and health systems to make risk decisions and to create policies to respond to public health crises, natural disasters, and political or social movements. The goal of this thesis is to develop new methods that handle computational and analytical problems for complex high-dimensional text data, develop a new query expansion approach to enhance the performance of information retrieval systems, and to present new techniques for analyzing multilingual topics using a translation service.

First, in the field of dimensionality reduction, we develop a new method for detecting and eliminating domain-based words. In this method, we use three different datasets and five classifiers for testing and evaluating the performance of our new approach before and after eliminating domain-based words. We compare the performance of our approach with other feature selection methods. We find that the new approach improves the performance of the binary classifier and reduces the dimensionality of the feature space by 90%. Also, our approach reduces the execution time of the classifier and outperforms one of the feature selection methods.

Second, in the field of information retrieval, we design and implement a method that integrates words from a current stream with *external data sources* in order to predict the occurrence of relevant words that have not yet appeared in the primary source. This algorithm enables the construction of new queries that effectively capture emergent events that a user may not have anticipated when initiating the data collection stream. The added value of using the external data sources appears when we have a stream of data and we want to predict something that has not yet happened instead of using only the stream that is limited to the available information at a specific time. We compare the performance of our approach with two alternative approaches. The first approach (static) expands user queries with words extracted from a probabilistic topic model of the stream. The second approach (emergent) reinforces user queries with emergent words extracted from the stream. We find that our method outperforms alternative approaches, exhibiting particularly good results in identifying future emergent topics.

Third, in the field of the multilingual text, we present a strategy to analyze the similarity between multilingual topics in English and Arabic tweets surrounding the 2020 COVID-19 pandemic. We make a descriptive comparison between topics in Arabic and English tweets about COVID-19 using tweets collected in the same way and filtered using the same keywords. We analyze Twitter's discussion to understand the evolution of topics over time and reveal topic similarity among tweets across the datasets. We use probabilistic topic modeling to identify and extract the key topics of Twitter's discussion in Arabic and English tweets. We use two methods to analyze the similarity between multilingual topics. The first method (full-text topic modeling approach) translates all text to English and then runs topic modeling to find similar topics. The second method (term-based topic modeling approach) runs topic modeling on the text before translation then translates the top keywords in each topic to find similar topics. We find similar topics related to COVID-19 pandemic covered in English and Arabic tweets for certain time intervals. Results indicate that the term-based topic modeling approach can reduce the cost compared to the full-text topic modeling approach and still have comparable results in finding similar topics. The computational time to translate the terms is significantly lower than the translation of the full text.

Dedication

> ”Behind every young child who believes in himself is a parent who believed first.”
>
> *Matthew Jacobson*

This dissertation is dedicated to my parents Mahmoud Alshanik and Sana’a Bakeer for their endless love, prayers, encouragement, sacrifices, and support throughout my whole life. Without their loving support, I could not have made it this far. Words cannot express my feelings and gratitude.

I would like also to dedicate this dissertation to my brothers Samer, Mohamad, Dea’a, and my dearest sister Lana for their endless love and support.

I would like also to dedicate this dissertation to the loving memory of my grandmother, you will always be remembered.

Acknowledgments

First and foremost, Alhamdulillah, I want to thank ALLAH for lighting my way and giving me countless blessings, strength, knowledge, ability, and opportunity to complete this work successfully.

I would like to express my deepest appreciation to my advisor, Dr. Amy Apon. Thanks to her supportive attitude, amazing guidance, perfect time management, endless encouragement, and consistently wise advice, she was a true source of inspiration. I would also like to express my gratitude to committee members Dr. Ilya Safro, Dr. Alexander Herzog, and Dr. Nina Hubig for generously offering their time, support, and guidance.

I would like to thank Clemson University for affording me the un-imaginable opportunity to complete my Masters' and PhD degree.

I would also like to thank my best friends Sereen Majdalaweyh, Lana Alshanik, Alaa Assad, Qabas Sabrah, and Ethar Qawasmeh, who constantly supported me.

Last but not least, I would like to thank my family and friends for standing by my side and helping me achieve my long-lasting dream. Words can not express how grateful I am to my parents for all of the sacrifices that they have made on my behalf.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

The massive growth in the amount of text available and the need for fast preprocessing, retrieving, and analyzing multilingual text has led to an interest in developing methods to handle the inherited challenges in text data. Raw text data typically introduces major computational and analytical obstacles (e.g., extremely high dimensionality) to data mining and machine learning algorithms. The unprecedented growth in raw text data is making the retrieval process more difficult. The information retrieval system needs to filter a massive amount of text data to return a set of results that match a users' search query. This problem becomes more significant when using a static query to filter the data from high-volume, high-velocity, real-time sources. Moreover, the availability of text data in different languages introduces significant challenges (e.g., hiring people with language skills) to the analysis process, which make it extremely complex and costly. The goals of this dissertation are to develop a method that handles computational and analytical problems for complex high-dimensional text data, develop a new query expansion approach to enhance the performance of information retrieval systems, and present new techniques for analyzing multilingual topics using a translation service.

The first research topic in our thesis focuses on developing a method to detect and eliminate domain-based words for text preprocessing to handle high dimensionality problems in the text data. Removing words that are not informative enough is a crucial storage-saving technique in text indexing and results in improved computational efficiency. Typically, a generic stop word list is applied to a dataset regardless of the domain. However, many common words are different from one domain to another but have no significance within a particular domain. Eliminating domain-specific common words in a corpus using our method result in reduces the dimensionality of the feature space, and improves the performance of binary classification.

The second research topic of our thesis focuses on enhancing the performance of the information retrieval system using streaming data. Information retrieval systems typically use query expansion techniques to enhance the initial user query, e.g., by adding inflected forms, cognates, and related words manually retrieved from the text. [87, 88]. In this research, we propose a novel query expansion technique that addresses the challenges of analyzing data from high-volume, high-velocity social media streams. We develop a query expansion technique that integrates words from the current stream with external data sources to predict the occurrence of relevant words that have not appeared in the stream yet. The external data sources enrich user queries with words that do not appear in the stream yet but are highly correlated with emergent words in the existing stream. By adding these words, our system can construct queries that capture emergent events that were not anticipated when initiating the data collection stream.

The third research topic of our thesis focuses on analyzing the similarity of topics in multilingual text using semantic similarity for two languages (English and Arabic) during the COVID-19 pandemic within three months using tweets collected in the same way and filtered using the same keywords. We utilizes topic modeling with Google translation API and compares two methods for identifying common-language topics with a translation happening at different step in each method (full-text topic modeling, term-based topic modeling). The full-text topic modeling translate the entire text from one language to another, then run LDA on the translated text. The term-based topic modeling run LDA on the text before translation, then use a translation service to translate the LDA words from one language to another. The results indicate that the term-based topic modeling approach is much cost-efficient than the full-text topic modeling approach and still have comparable results in finding similar topics.

1.1 Thesis Statement

The goal of this thesis is to provide methods for effective text preprocessing, retrieving, and analyzing multilingual text to solve different challenges inherent in the text data. We have designed and implemented novel methods for reducing text dimensionality through text preprocessing, improving the performance of the information retrieval system through dynamic query expansion over streaming data, and analyzing the similarity of multilingual text through topic modeling and translation service. For the dimensionality reduction method, we investigate the hypothesis that eliminating the domain-based words in a corpus can reduce the dimensionality of the feature space and enhance the performance of text mining tasks. A major contribution of this method is that we provide evidence to show that our method is capable of extracting domain-based words in different datasets and that it is able to reduce the dimensionality of the corpus by 90%. For the query expansion, the key hypothesis we investigate is that the proposed query expansion method can detect emergent events in streaming data and enrich the query with novel words from an external source. A major contribution of this method is that we give evidence that our query expansion methods can enhance the performance of the information retrieval and significantly improve the quality of search queries by adding more relevant words. Also by using an external data source, we show that the proposed methods more efficient than the static query expansion method. For an automated multilingual text analysis, the key hypothesis we investigate is that the proposed methods (full-text topic modeling, term-based topic modeling) in addition to topic modeling can find the similarity between multiple languages. A major contribution of this work is that we give evidence that using term-based method can reduce the cost of the translation process and translation execution time.

1.2 Research Contribution

In this thesis, we develop novel methods to enhance the performance of the text mining tasks of binary classification, information retrieval, and multilingual text analysis. In this chapter, we provide a summary of our contributions.

Accelerating Text Mining Using Domain-Specific Stop Word Lists

We present a novel mathematical approach for detecting domain-specific words called the hyperplane-based approach. This new approach depends on the notion of low dimensional representation of the word in vector space and its distance from the hyperplane, where the domain-specific words are defined as the words with the shortest distance from the separating hyperplane. The performance of the proposed approach is quantified by the accuracy and the execution time of five classifiers: (1) Naive Bayes, (2) Random Forest, (3) Logistic Regression, (4) Thunder SVM, and (5) CART. This approach is validated using 138 sub-corpora from three datasets (Hansard Speeches, IMDB Movie Review, and Pubmed). Also, it is compared with two feature selection approaches, namely χ^2 and MI. For each feature selection method including the hyperplane-based approach and each corpus, various word elimination percentages are considered to find the optimal elimination percentage for each classifier and approach. The hyperplane-based approach generally improves the performance of the classifier and it achieved comparable performance with the χ^2 and MI. *However, our experiments indicate that qualitatively the eliminated words significantly differ from other approaches. In addition, the method is more robust to the erroneous elimination of important words.* The performance of our approach varies with respect to the classifier and the elimination percentages. For example, the Naive Bayes classifier presented the best improvement of accuracy before and after the elimination using our approach, and the optimal elimination percentage per our approach is 90% for all datasets. Finally, the proposed approach plays a key role in reducing the dimensionality of the corpus, which means reducing the classification execution time. The implementation of the hyperplane-based approach in different datasets is straight-forward and merging the hyperplane based-domain words with other feature selection domain words is a future research direction.

Enriching Query Expansion Using External Data Source

We present a novel approach for expanding search queries called the proactive query expansion method. This new method depends on adding novel words to the search query from an external data source, where the words are chosen using either nearest neighbor words or highest frequency words to each word that appears in the five LDA topics and DEC words. Two major experiments were performed: (1) We compared the performance of our proposed query expansion methods: method2 (i.e., query expansion using LDA and DEC), method3 and 4 (i.e., query expansion using LDA, DEC and historical data) with the reference model (i.e., method1, query expansion using LDA). The performance of the proposed approach is quantified by quality indicators of the streaming data which are the tweet count, hashtag count, and hashtag clustering. (2) We tested the effectiveness of our proposed methods to predict future emerging events or to predict the conversations from previous time intervals using a different set of hashtags. Our experiment performed on 20.5 million tweets, the primary stream, covers fifteen days from April 17th–May 3rd, 2015. For our external source, the historical data, that is, the secondary stream, we collected news from CNN and New York Times which covers one year before the event happened (2014). Generally, the historical query expansion methods (Method 3 and 4) improve the performance of the information retrieval and achieve higher performance compared with Method1 and Method2. Additionally, the experiments indicate that our approach can enhance the quality of the results for all the topics. Besides, our proposed methods are more concise comparing to Method1. Finally, the proposed methods play a key role in enhancing the performance of the search query, which means providing the user with more relative and concise results of interest.

Lost in translation: Estimating multi-lingual linguistic context

We combine topic modeling with a translation service to develop two methods (full-text topic modeling, term-based topic modeling) for analyzing multilingual topic similarity using two languages (English and Arabic). The full-text topic modeling approach translates all text in the Arabic language to the English language then runs topic modeling. The term-based topic modeling approach runs topic modeling on the Arabic text before translation and then translates the key words in each Arabic topic to English. After running topic modeling and making the translation we compare the similarity between English and Arabic topics using semantic similarity. We use the notion of low dimensional representation of the word in vector space to find the cosine similarity between topics in two languages. We find that there are similar topics in English and Arabic tweets. In addition, we find that both methods have the same similarity results. However, the term-based topic modeling approach is more efficient in term of cost. The performance of the two approaches is quantified by the similarity of topics in both languages and the cost in terms of translation execution time. These approaches are validated using multilingual COVID-19 tweets. The term-based topic modeling approach generally achieves comparable performance of the full-text topic modeling in terms of topics similarity. The term-based topic modeling approach is better than full-text topic modeling in terms of cost, and needs less time to translate the text. *Our experiments indicate that the costs of the translation makes the term-based topic modeling a valuable addition to multilingual text analysis.* Finally, the proposed approach plays a key role in reducing the translation execution time.

Chapter 2

Background and Literature Review

This chapter introduces the background of this research work. The provided background covers essential knowledge to understanding this dissertation and the three proposed novel ML algorithms.

2.1 Accelerating Text Mining Using Domain-Specific Stop Word Lists

Stop word removal has been an active area of research and several studies have sought an automated method for generating a stop word list for different languages. Some studies try to find a generic stop word list that is domain independent. Others focus on finding domain-specific words that are document-dependent [1].One method, term frequency, was the first automated process for identifying and extracting stop words.

The term frequency approach produces valuable results for the words that have the highest frequency in the corpus, but term frequency does not take into account the words that occur frequently in only some documents. Also, the term frequency does not consider the words that rarely occur in the corpus but still meaningless for the classification [2].

Our work differs from this approach by returning the domain-specific common words with respect to the document collection.

In [3] the authors used term frequency for word elimination and studied its effect on the similarity of Hindi text documents. They found that the removal of stop words based on term frequency decreased the similarity of documents. In [4] authors used term frequency with Punjabi texts. They identified 615 stop words in a corpus of more than 11 million words. However, in later work they found that the frequency count cannot be taken as the true measure of stop word identification [5]. To overcome the drawback of the frequency words the authors used a statistical model based on the distribution of words in documents along with the frequency count to generate an aggregated stop word list for the Punjabi language.

Word entropy is another method. It was used in [2] to generate web-specific stop lists for web document analysis. This research aimed to eliminate common web-specific terms such as ’email’, ’contact’, etc. The words with the lowest entropy were considered to be stop words. The method was evaluated using the web document dataset and the BankSearch dataset. The authors successfully extracted the web-specific stop list from the web document dataset. However, the results on the BankSearch dataset, which consists of 10 different categories, showed a bias towards low entropies for words that are category-related and frequent in a large number of documents. The authors claimed that to use the word entropy the dataset should be unbiased towards any subject category. In [6] the authors used word entropy to generate a Mongolian stopword list.

Other research has combined the term frequency and word entropy approaches. In [7] the authors used strategies focused on statistical methods and knowledge-based measures for creating generic stop words for Hindi text.

The objective is to measure the information content of each word in the corpus, which is measured using word entropy. The main advantage of this method is to overcome the problem inherited by manually picking stop words, which takes a lot of time. Authors in [8] aggregate term frequency and entropy value to construct the first stop word list for the Amharic text.

A Chinese corpus consisting of Xinhua News and People’s Daily News is used in [9] to find a generic stop word list aggregated from two models: a statistical model that measures the combination of mean probability and variance of probability, and an information model that measures the informativity of words using the words’ entropy in the corpus. The generated Chinese stop word list had a high intersection with the English stop words. Also, compared with other Chinese stop lists, this method was more effective, and it was faster than manual generation[9]. The methodology was used to generate an Arabic stop word list in [10] with superior performance.

Authors in [11] present the term-based random sampling approach in which they generated a list of stop words based on the importance of terms using Kullback-Leibler divergence measure, which models how informative a term is. They show that term-based random sampling outperforms the rank-frequency approach in terms of computational effort to derive the stop word list. However, using the term-based random sampling approach on DOTGOV and the WT10G collections to obtain the stop word list did not provide better results than baseline approaches.

A dictionary approach is used in [12] to generate the first Sanskrit stop word list. However, this method of generating the stop word list is both resource intensive and time consuming[8]. The hyperplane-based approach does not require a dictionary to return the domain-specific words. A rule-based approach using static eleven rules was used in [13] to automatically develop a stop word list for the Gujarati language. The main limitation of this approach, as described in the paper, appears when the stop words contain more than three characters. In the hyperplane-based approach there is no limitation based on the length of the domain-specific word.

A method based on the weighted χ^2 was used to generate the Chinese stop word list[14]. The method considers the high document frequency and the low statistical correlations with all the classification categories. This is the first list that uses the dependent relationship between a word and all categories in a set of documents. This approach inputs a threshold to specify the size of the stop word list. The Naive Bayes classifier tested the performance of the generated stop words before and after elimination, and tested stop word lists of various lengths. The study eliminated the words based on some threshold. However, there is no guidance on the optimal elimination percentages of the detected domain-specific words. In our work we have performed extensive experiments to find and provide guidance on the best percentage of elimination.

2.2 Thinking Outside the Box: Enriching Query Expansion Using External Data Source

Query expansion has been an active area of research and several studies have sought an automated method to deal with the word mismatch in information retrieval. Authors in [15] perform automatic query expansion using three representative techniques. The first technique is the global analysis based on the method introduced by [16]. The global analysis technique creates a thesaurus-like database with a ranked list of phrases for a given query. The method is known as the global analysis approach because the association database it uses considers the entire collection of documents, and the process is frequently computationally intensive. The task in [15] is different from our task in which we use the streaming data as a primary stream to the query. This means the thesaurus-like database used in [16] is not directly applicable. Besides, the database in this solution requires to be updated with each new tweet, which makes the method fail in large-scale data [17].

The second approach introduced by [15] is the local feedback method, which overcomes the drawback of the global analysis by using the documents in the query results to generate a list of top-ranked words instead of using the entire corpus. The efficacy of this method crucially depends on the quality of the query result itself. The reliability of the local feedback method, therefore, remains an issue even it is less expensive to perform [17]. Local context analysis is the third technique introduced by [15], which uses a combination of the global analysis approach and the local feedback approach. It uses the ranked query results to identify the top concepts (noun groups share the same semantic meaning). Based on the distance of each concept to the original query in the global thesaurus and their TF-IDF scores, the local context analysis picks new words. This method achieves better performance than using either global analysis or local feedback separately. However, it also requires a static metric to rank the documents in the query result.

For querying streaming social data, the metrics to evaluate the goodness of the query results are often dynamically changing and may comprise a mixture of various sub-metrics [17]. Hence, it is not feasible to directly use the local context analysis method introduced by [15]. The query expansion method proposed in [18] uses tweet data as the query platform. The approach in [18] is similar to our query expansion work in that they employ a time-based indicator to deal with the data streaming's dynamic nature, which is similar to our method of measuring query quality using a dynamic metric. Authors in [18] use repost count and followers of posts as an indicator of a tweet's quality which changes with time. Our approach uses tweet count and hashtags information as a quality indicator of query results.

Many studies use different techniques to detect the emergent events in the streaming data [19, 20, 21, 22, 23]. The application of topic models has been an active area of research in informational retrieval, and several studies have sought an automated method for using topic models in query expansion. The authors in [24] apply Latent Dirichlet Allocation (LDA) to improve the retrieval results using cluster-based models. Authors in [25] use Mixture of Unigrams [26], LDA [27], and the Pachinko Allocation Model [28] to integrate the topic models into the retrieval process. Authors in [29] use topic models for re-ranking initial retrieval results. Another approach [30] uses

LDA in query expansion to improve the performance of relevance feedback. Authors in [31] use topic modeling to enhance the search and recommendation functionalities of Enterprise Social Software. The study in [32] expands the initial query by using the latent topic information on the documents retrieved at the first search. Similar to this line of work, we propose a method that uses topic distributions of the targeted documents in addition to an external data source to expand the query. To the best of our knowledge, there is no existing study that uses topic modeling with external sources to expand the initial query in streaming data.

2.3 Lost in translation: Estimating multi-lingual linguistic context

Automatic text analysis has been an active area of research, and several studies have sought an automated method to deal with multilingual text [33, 34, 35, 36, 37, 38, 39]. Topic modeling is one of the commonly used approaches for text analysis [40, 41]. Most studies to date use topic modeling to extract underneath discussion themes or topics from a large set of text written in one language, especially in English [42, 43]. More precisely, in the field of public health many studies use topic models [44] for understanding healthcare reviews [45, 46], identifying infectious disease [47, 48], and studying health behaviors [49, 50]. However, comparative analysis of multilingual text help to overcome the language barriers of the communication process and helps to remove the boundaries of understanding cross-cultural differences [51, 52], making risk decisions and policies to respond to public health crises, natural disasters, and political or social movements much more accessible. However, analyzing multilingual texts require people with language skills to collect and analyzes the data, which makes the analysis process complex, labor-intensive, and costly [37]. Using automated text analysis and machine translation to translate texts from one language to another helps to overcome these challenges and exchange information between different countries for effective communication. The approaches in [38, 53] use machine translation and topic modeling to study public communication.

There are two approaches to perform multilingual analysis using topic modeling [37]. The first approach applies topic models on the multilingual text then matches the topics. The second approach translates the text into common language and then applies a topic model. Using the first approach requires additional information to bring the topics together [54]. The second approach is much easier given the availability of automated translation services. However, using the second approach can be costly, often requiring that a massive amount of text be translated from one language to another using paid translation services. Authors in [38] show that machine translation services and topic modeling are useful tools for analyzing multilingual text. However, in their analysis they focus on the translation of the full documents, which makes the translation process costly in terms of time and money. The authors in [37] argue that it is more cost effective and time saving if only translating the unique words in the corpus using the document term matrix that used in topic modeling process. However, translating the document term matrix ignore the the syntactic structure which may affect the translation process [37]. Authors in [53, 37] concluded that there is no lost in translation by using the document term matrix which confirmed that the translation of the individual terms of a document-term matrices can be a useful shortcut for the translation of larger corpora. This thesis continues along this path but instead of translating the document term matrix we only translate the top words in the estimated topics.

This research studies the potential of the use of topic modeling with a translation service using two approaches for comparing the similarity between two languages during the COVID-19 pandemic. One of the suggested approaches uses only the words from the estimated topic instead of translating the full text to reduce the cost of the translation process. The effectiveness of the suggested approach is examined by determining whether translating the topics after applying a topic modeling achieves the same performance as translating the full text then applying a topic modeling. Also, it is examined whether topic modeling coupled with machine translation can reduce the cost of the translation process.

2.4 Related Technologies

The low-dimensional representation method, word2vec, utilizes the skip-gram model. The word2vec maps words into a low-dimensional space, thus revealing non-trivial context based relationships between words. Many syntactic and semantic relationships between words can be defined by using simple algebraic operations on the word vectors. For example, the word vector ”King” - word vector ”Man” + word vector ”Woman” results in a word vector similar to vector representation of the word vector ”Queen”[55].

Naive Bayes classifier [56], a probabilistic model based on Bayes theorem, is a supervised learning technique broadly used for text classification. The method is, in practice, extremely sensitive to preprocessed data quality and other factors. Here we mostly use it as a baseline technique to examine the sensitivity of the preprocessing.

Support Vector Machine (SVM) is a quadratic optimization-based technique that, in its simplest form, maximizes the margin between classes using the optimized separating hyperplane. While having fast performance, the linear SVM is known to be sensitive to hard classification problems. Slower nonlinear (aka kernel based) SVM often produces higher quality results but requires suboptimal heuristics to achieve linearly scalability [57, 58]. Here we use the Thunder SVM package [59] which provides a good quality/performance trade-off.

We also use three other broadly applicable classification techniques, namely, Logistic Regression [60], Random Forest [61], and Classification and Regression Trees (CART) [62]. These are methods of comparable quality when applied in the text mining domain. Their implementation is available inside Python's module called Scikit-learn [63].

The low-dimensional representation model, fastText, utilizes the skip-gram model. representation [64]. The fastText maps words into a low-dimensional space and uses subword information, revealing non-trivial context based relationships between them. Many syntactic and semantic relationships between words can be defined by using simple algebraic operations on the word vectors.

Dynamic Eigenvector Centrality (DEC) [22], is a graph based technique. It extracts the emergent words from a document stream based on dynamic semantic graphs. DEC, in its simplest form, constructs a network in which the nodes and undirected edges correspond to words and co-occurrences of words in a stream of documents, respectively. DEC use the constructed network to rank and extract top-ranked emergent words.

Latent Dirichlet Allocation (LDA), a generative probabilistic model, is an unsupervised learning method used for extracting latent topics from a large set of documents. LDA is uses a three-level Bayesian model to fit the generative process, it represents documents as random mixtures over latent topics and represents each topic as a distribution over words [27].

Chapter 3

Accelerating Text Mining Using Domain-Specific Stop Word Lists

Text preprocessing is an essential step in text mining. Removing words that can negatively impact the quality of prediction algorithms or are not informative enough is a crucial storage-saving technique in text indexing and results in improved computational efficiency. Typically, a generic stop word list is applied to a dataset regardless of the domain. However, many common words are different from one domain to another but have no significance within a particular domain. Eliminating domain-specific common words in a corpus reduces the dimensionality of the feature space, and improves the performance of text mining tasks. In this thesis, we present a novel mathematical approach for the automatic extraction of domain-specific words called the hyperplane-based approach. This new approach depends on the notion of low dimensional representation of the word in vector space and its distance from hyperplane. The hyperplane-based approach can significantly reduce text dimensionality by eliminating irrelevant features. We compare the hyperplane-based approach with other feature selection methods, namely χ^2 and mutual information. An experimental study is performed on three different datasets and five classification algorithms, and measure the dimensionality reduction and the increase in the classification performance. Results indicate that the hyperplane-based approach can reduce the dimensionality of the corpus by 90% and outperforms mutual information. The computational time to identify the domain-specific words is significantly lower than mutual information.

Reproducibility: code and results can be found at `https://github.com/FarahAlshanik/Domain-Specific-Word-List`

3.1 Motivation

The amount of raw text data produced in science, finance, social media, and medicine is growing at an unprecedented pace. Without effective preprocessing, the raw text data typically introduces major computational and analytical obstacles (e.g., extremely high dimensionality) to data mining and machine learning algorithms. Hence, text preprocessing has become an essential step to better manage and handle various challenges inherent with the raw data before it is ready for text mining tasks. Stop word removal is one of the most important steps in text preprocessing. Stop words are those that appear frequently and commonly and contribute little analytical meaning and value in text mining tasks [65, 66].

Elimination of stop words reduces text count in the corpus typically by 35% to 45% and makes the application of text mining methods more effective[67]. In the majority of text mining tasks, preprocessing begins with eliminating stop words without considering the domain of application. For example, in the popular Python Natural Language ToolKit (nltk) module, the English stop words list contains frequent words such as “the”, “and”, “to”, “a”, and “is” [68]. However, there may exist many domain-specific words that do not add value in a text mining application but that increase the computational time. For instance, the word “protein” is likely to add little value in text mining of bioinformatics articles, but not for a collection of political articles. Reduction in running time is also extremely important for such tasks as streaming, sliding window text mining, and large-scale semantic knowledge network mining [22, 69, 70, 71] and becomes even more sensitive problem with the analysis of long texts [72, 73].

The objective of this research is to present the hyperplane-based approach for detecting and eliminating domain-specific common words in a corpus of documents in addition to (or in some cases instead of) commonly-used stop words. Our approach aims to enhance the performance of binary text classification and to reduce the dimensionality. In binary text classification, the documents are grouped into two classes. The trained classifier’s role is to predict the document class. However, the domain-specific words often mislead the classifiers’ prediction. The hyperplane-based approach employs the principle of low dimensional representation of words using the word2vec skip-gram model. The main idea of our approach is to project in the vector space both centroids of the two classes and each word in the corpus. The method constructs a hyperplane perpendicular to the normalized vector between the centroids of the two classes such that the words closer to the perpendicular hyperplane are selected as domain-specific common words.

To justify the proposed approach and to demonstrate the contextual difference with other traditional feature elimination methods, we eliminate a varying count of the detected domain-specific words from the corpus and observe the accuracy of five different classifiers: Naive Bayes, Logistic Regression, Random Forest, Support Vector Machine (SVM), and Classification and Regression Trees (CART). We evaluate the five approaches on three different datasets: (1) Pubmed [74], (2) IMDB movie reviews[75], and (3) Hansard Speeches 1979–2018[76].

Two major experiments were performed. First, we observed the performance of our approach before and after eliminating domain-specific words on the classification accuracy and execution time. Secondly, we compared the classifier accuracy and validate our results with other feature selection methods namely χ^2 and mutual information. The classifiers are used to predict the journal name for the Pubmed dataset, sentiment for the IMDB movie review dataset, and party for the Hansard Speeches. Also, we found the size of the overlap of the remaining words after eliminating the domain-specific words between our approach and the χ^2 or mutual information.

The results show that the hyperplace-based approach has improved performance in terms of accuracy and execution time with respect to feature selection methods. Through our proposed approach we identify that a significant number of stopwords can be removed without reducing the accuracy of any classifier we considered. On the contrary, we determine that after removing 90% of the words within the Hansard Speeches dataset according to our method, the accuracy of a Random Forest classifier increases by 4%, and a Naive Bayes classifier increases by 12%.

3.2 Domain-specific word detection

Simplified Example

The basics of our approach are best understood with a simple example on a synthetic dataset. Suppose a synthetic binary class dataset of total of 40,000 documents is generated in which the two classes have an equal number of documents. The length of each document from both classes is 300 words. The words are chosen randomly from predefined dictionary that has been created for this experiment. Dictionaries of classes A and B consist of 2000 words each represented by w_i, and v_i, respectively, where $1 \leq i \leq 2000$. The classes are disjoint and any classifier will result in a 100% accuracy. Now, in order to include common words between the classes, a dictionary of 300 words m_i is created, $1 \leq i \leq 300$. Then, 10 m_i words are randomly added to each document in the two class. The word2vec skip-gram is applied on the synthetic data corpus to project the words into a vector space using 100-dimensional word vectors. After that the centroid of each class is calculated by averaging the word embedding in each class. We found that the common words m_i exist in the middle between the two classes as shown in Fig. 3.1 (blue points). Accordingly, we decided to create a perpendicular hyperplane on the normalized vector between the centroid of the two classes (pink and gold). Additionally, we calculated the distance between each unique word in the corpus and the hyperplane, such that the words that have the shortest distance will be the domain-specific common words, and the words that have the longest distance will be the words that are used to distinguish between the two classes (red).

Overview of the Approach

We introduce the hyperplane-based approach to detect the domain-specific words. It is based on the distance of the word from the separating hyperplane with the notion of a low dimensional vector representation of the words using the word2vec model[77]. The domain-specific words are determined as the words that have the *shortest distance* from the hyperplane.

This approach aims to enhance the performance of binary text classification and reduce the execution time of the classifier. The goal is to eliminate the features (i.e., words) that are not used to distinguish between the two classes. The flowchart of the approach is presented in Fig. 3.2. The hyperplane-based method consists of four main steps: 1) text prepossessing, 2) centroid embedding, 3) computing the tokens' distances from an orthogonal hyperplane, and 4) sorting and detecting the domain-specific common words list. These steps are explained in the following subsections.

Text Preprocessing

The text preproccesing step starts by tokenizing the text, then converting the words to lower case, and finally removing the special characters as shown in Fig. 3.2. Traditional stop words (such as those that are given in Python nltk module) are not removed. Rather, some or all of the stop words are anticipated to be removed as a result of our method which indeed happens in the end of

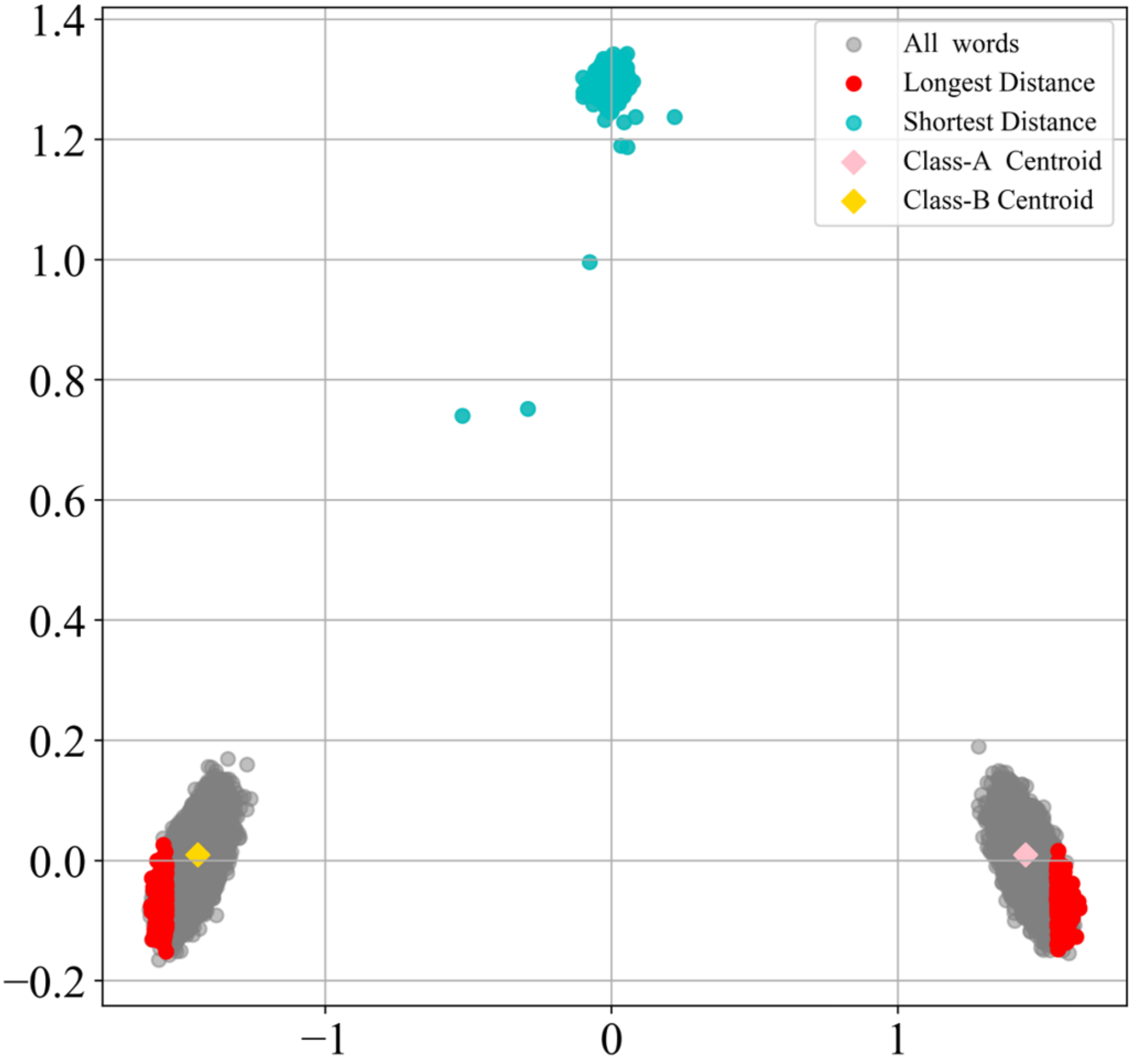


Figure 3.1: PCA-based visualization of the synthetic dataset embedding. The closest words to hyperplane are marked in blue (300 words). The words that have a largest distance are marked in red (300 words), the centroid embedding of the class A and class B are marked in pink and gold, respectively

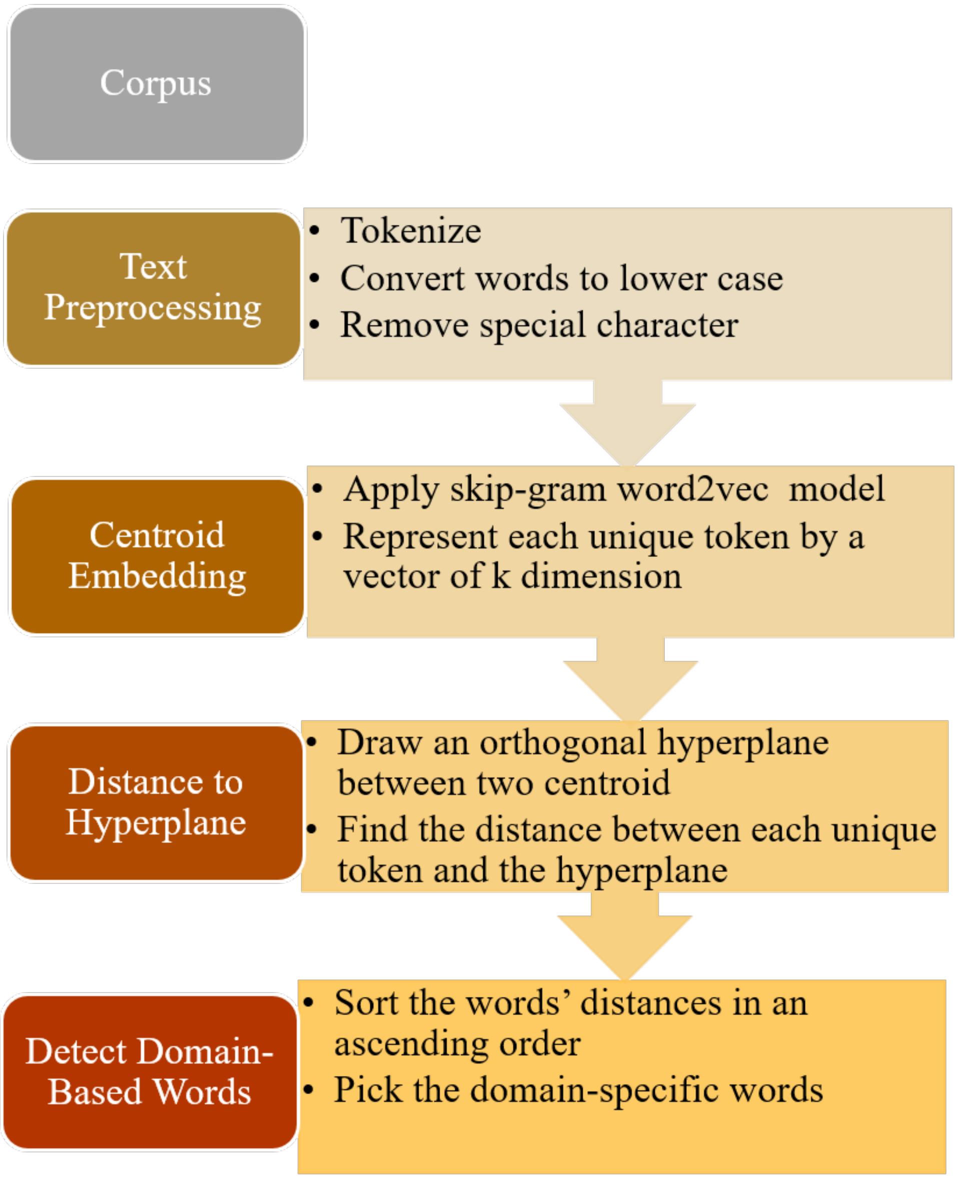

Corpus
Text Preprocessing
• Tokenize
• Convert words to lower case
• Remove special character
Centroid Embedding
• Apply skip-gram word2vec model
• Represent each unique token by a vector of k dimension
Distance to Hyperplane
• Draw an orthogonal hyperplane between two centroid
• Find the distance between each unique token and the hyperplane
Detect Domain-Based Words
• Sort the words' distances in an ascending order
• Pick the domain-specific words

Figure 3.2: Hyperplane-based approach steps

the entire process.

Centroid Embedding

The centroids of class are the averages over the corresponding word embeddings, precomputed using the word2vec skip-gram model. The result is that the semantically similar words are mapped together in a vector space. This representation is also known as distributed numerical representations of word features.

Given two disjoint classes of n documents, namely, A and B. The centroid of a class is defined using the following steps. First, the word2vec skip-gram model is utilized to construct a k-dimensional representation of each word, i.e., each embedded word, or token, t is represented as a vector $emb(t) = (t_1, t_2, t_3, ..., t_k)$. Then, the centroid of each class is computed as the average of the embeddings of words from that class:

$$Center_X = \frac{\sum\limits_{t \in X} emb(t)}{M}, \tag{3.1}$$

where X is a class, $emb(t)$ is an embedding function, and M is the total number of unique words in class X. In Fig. 3.3, we visualize the example of embedding using the 2D PCA dimensionality reduction from the initial 100-dimensional embedding space. In this example, the corpus of documents for classification contains abstracts of two journals, *Cell* and *Journal of Prosthetic Dentistry*, extracted from the Pubmed dataset. The figure illustrates the two-dimensional visualization of the individual journals' centroids and the stop words in nltk module, represented by diamonds and stars, respectively.

Distance from Hyperplane

In this thesis, the separating hyperplane non-strictly defines the boundary between the two classes and their centroids, namely, class A ($Center_A$) and class B ($Center_B$). It is orthogonal to the line connecting the centroids, and passes through its middle. It is defined by the linear equation:

$$w'x + b = 0 \tag{3.2}$$

where w is the slope of the plane, b the offset. The first step is to find the slope of the plane, i.e.

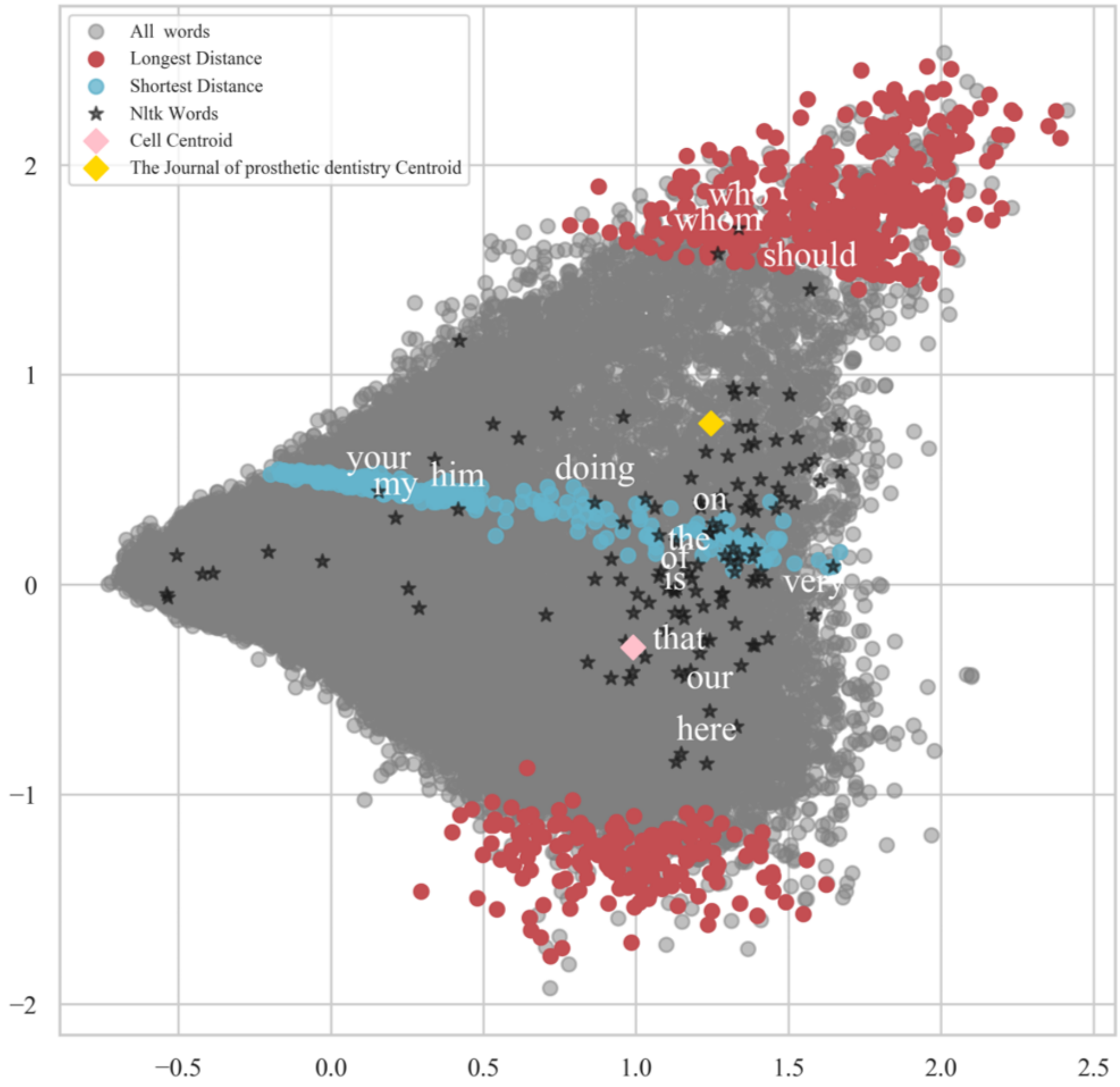


Figure 3.3: PCA-based visualization of the Cell and Journal of Prosthetic Dentistry from Pubmed dataset. The closest words to hyperplane are marked in blue (300 words), the words that have a largest distance are marked in red (300 words), the centroid embedding of the cell and Journal of Prosthetic Dentistry journals are marked in pink and gold respectively, some nltk stop words are plotted in white.

the normal vector perpendicular to the plane (w), which is defined as the difference between the two centroids A and B as appear in equation (3.3).

$$w = Center(A) - Center(B) \tag{3.3}$$

The offset point (b) will be equal to the negative dot product of the normal vector by the coordinate of any point on the plane as expressed in equation (3.4).

$$b = -w \cdot x_0 \tag{3.4}$$

The predefined point at this stage is the mid point between the two centroids (x_0), which can be calculated using:

$$x_0 = \frac{Center_A - Center_B}{2} \tag{3.5}$$

After defining the hyperplane equation, the distance between any point (embedded words in space) to the hyperplane is calculated as:

$$d = \frac{|w \cdot x_0 + b|}{||w||} \tag{3.6}$$

Fig. 3.3 shows that the shortest distance words (blue points) are clustered around the hyperplane which are the domain-specific words. The longest distance words (red points) are far away from the hyperplane which are the words that used to distinguish between the two classes.

Fig. 3.3 also includes the 127 stop words from the Python nltk (black stars). It can be noticed that the majority of these words are near the hyperplane, however, few of them defined as longest distance words. To explain such behavior, a sample of nltk stop words which positioned on various distances with respect to the hyperplane is picked as shown in Fig. 3.3. These words are: ”our”, “here”, “an”, “is”, “that”, “of”, “the”, “your”, “him”, “my”, “whom”, “doing”, “who”, “should”, and “very”. Then, the frequency of these words is calculated as shown in Fig. 3.4. The histogram shows a high discrepancy in the longest distance words’ frequency between the two classes (two journals) such as “who”, “should”, and “whom”, that means these words are most likely to appear in one journal rather than the other.

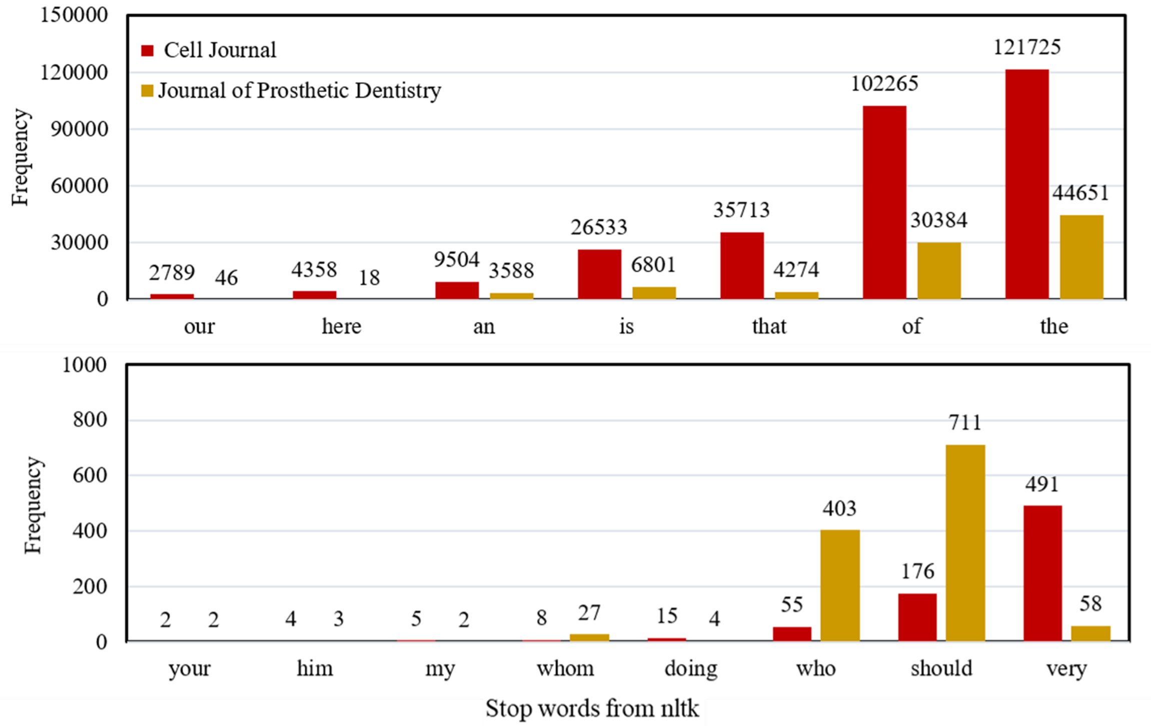


Figure 3.4: nltk stop Words frequency in Cell and Journal of Prosthetic Dentistry.

In this surprising example, some differences are quite significant and indicate that they can be used in determining the classes of papers. Thus, some of them should not be eliminated by simple traditional stop word removal.

Detect domain-specific words

Domain-specific words can be defined in this approach as the words with the shortest distance from the hyperplane. Therefore, the final step is to sort the words' distances in ascending order such that the words that have the shortest distance from the hyperplane can be detected.

3.3 Data Sources for Experiments

We tested our approach on three datasets. These are chosen to vary in size and complexity for testing the limitations of the approach.

Hansard Speeches 1979–2018: This is a public dataset of speeches from the UK House of Commons, extracted from Hansard, the official public records [76]. This data set includes the text for every speech made in the House of Commons between the 1979 general election and the end of 2017, with information for each speaker, including their party affiliation. We reduce the set of speeches to speakers from the two largest parties, Labour and Conservative. The resulting dataset consists of 1,608,012 speeches. We picked the Hansard Speeches because the baseline accuracy of the classifiers applied to these data is much lower than the other datasets, meaning the classification task is more difficult.

IMDB Movie Review Dataset: This is a dataset for binary sentiment classification containing 50,000 positive and negative reviews[75].

Pubmed: The Pubmed dataset[74] provides public information by the National Library of Medicine (NLM). It contains more than 26 million citations for biomedical literature from MEDLINE, life science journals, and online books. Our knowledge base starts as XML files provided by Pubmed, from which we extract each publication abstract, year, and journal. In this work we focus on the journal abstracts, treating journal names as the categories into which we categorize the abstracts. To ensure higher quality in the word embeddings, we only consider journals with at least 10,000 abstracts. From these journals, many pairs were generated then tested using the Logistic Regression classifier. The pairs that have an accuracy lower than 98% were picked to efficiently test the proposed approach. The curated dataset contains 100 pairs of journals and 2,694,790 abstracts.

3.4 Validation experiments and results

We conduct our experiments in two parts: First, the validation of the proposed approach is studied on the classification accuracy by eliminating three types of words: (1) the words with the shortest distance from the hyperplane (domain-specific words); (2) the words with the longest distance from the hyperplane (distinguishable words); (3) random words. Second, we compare the hyperplane-based approach against two feature selection methods: χ^2 and Mutual Information (MI).

χ^2 is used as a dimensionality reduction feature selection method which evaluates the independence of the feature and the class. The higher the χ^2 value, the more class information the feature contains. Mutual Information (MI) is another dimensionality reduction feature selection method. MI is used to measure the dependencies between two random variables, which are class and feature. The higher the MI value, the more information content exists between the feature and the class. Accordingly, a low χ^2 or MI value indicates that a word is a domain-specific common word. The most distinguishable words are those that have a high χ^2 or MI. For our proposed hyperplane method, domain-specific words are those that have the shortest distance from the hyperplane.

Our experimental scheme is illustrated in Fig. 3.5. In this study, 138 different sub-corpora from the Hansard speeches, IMDB movie review, and Pubmed datasets are tested per different feature selection methods, percentages of domain-specific words elimination, and classifiers. For each feature selection approach, 11 percentages of domain-specific words are eliminated in increments of 10 from 0% to 99%. The 0% is selected as a reference category because it represents the performance of a classifier before any words are eliminated.

For each case, five classifiers are applied: Naive Bayes, Thunder SVM, Logistic Regression, Random Forest, and CART. The performance of a classifier is quantified by its prediction accuracy, calculated as the ratio of correctly predicted instances to all instances in the experiment, and estimated using 10-fold cross validation. Overall, we conduct a total of 37,950 experiments to test the hyperplane-based approach.

Hansard Speeches

The Hansard data spans the time period from 1979–2018. We calculate the accuracy for all five classifiers, three feature selection approaches, and 11 elimination percentages for each year. We then calculate average accuracy across all years included in our data. Hyperplane-based approach validation starts with comparing the performance of the classifiers when eliminating three criteria: the shortest distance, longest distance, and random words. Figures. 3.6, 3.7, and 3.8 presents the average accuracy of classifiers per each selection criterion. As expected the highest accuracy is obtained when removing the domain-specific words (shortest distance), and the lowest accuracy when removing the distinguishable words (longest distance). Also, when the percentage of distinguishable words elimination increases, accuracy drops significantly. The accuracy of the classifiers when removing random words is in between the accuracy when removing the shortest or longest distance words. Additionally, we note that the accuracy of all classifiers when eliminating from 10 to 90% of words will maintain about the same level if not increase compared to the reference category (0% elimination).

Figures. 3.9, 3.10, and 3.11 presents the average accuracy of the classifiers per hyperplane-based, χ^2 and MI approach on the Hansard Speeches dataset. The hyperplane-based approach improved the average accuracy of the Naive Bayes classifier by 8% (from 0.66 to 0.74) when eliminating 90% of domain-specific words with respect to the reference category (0% elimination). Further, our

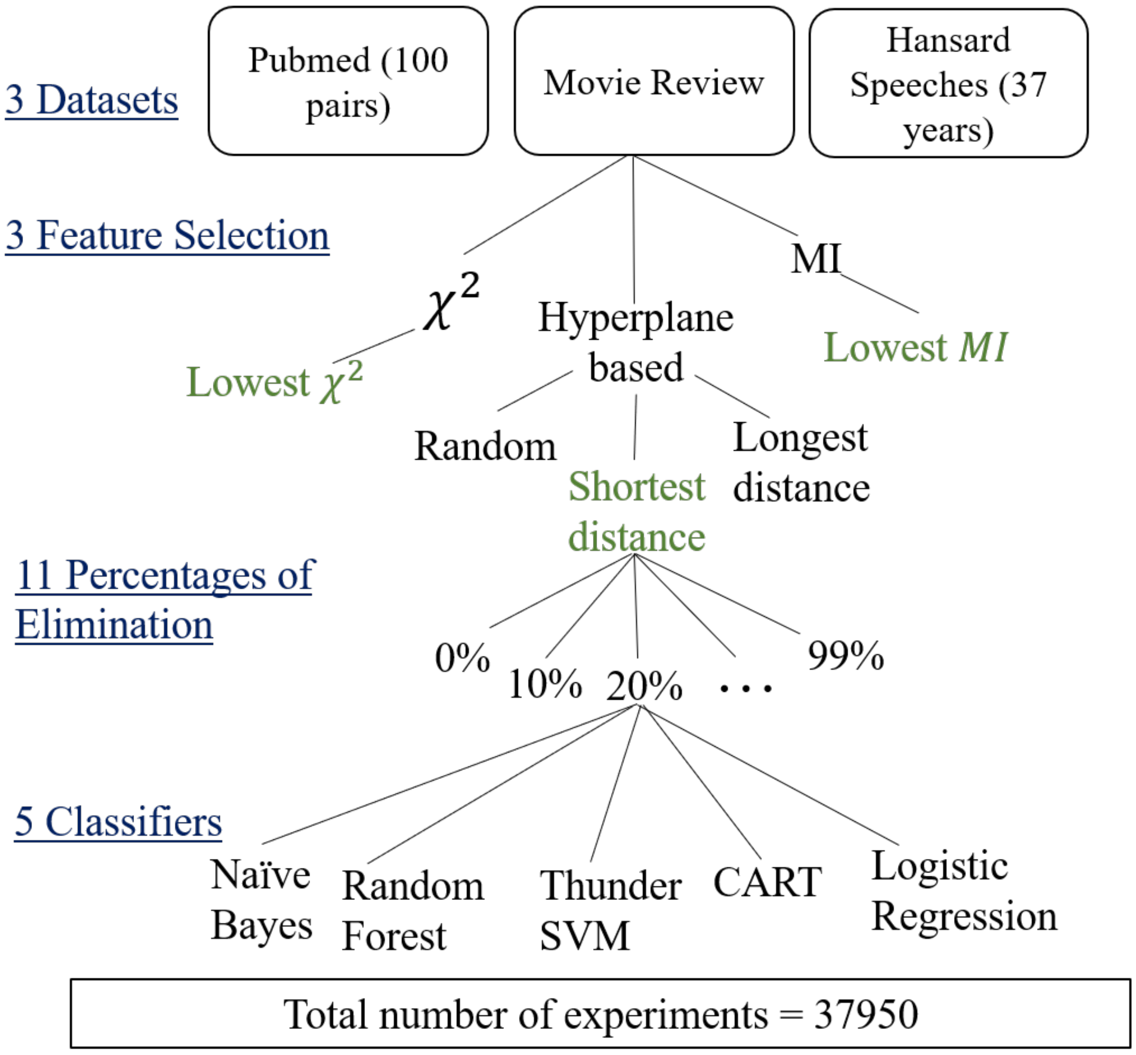


Figure 3.5: Experimental scheme

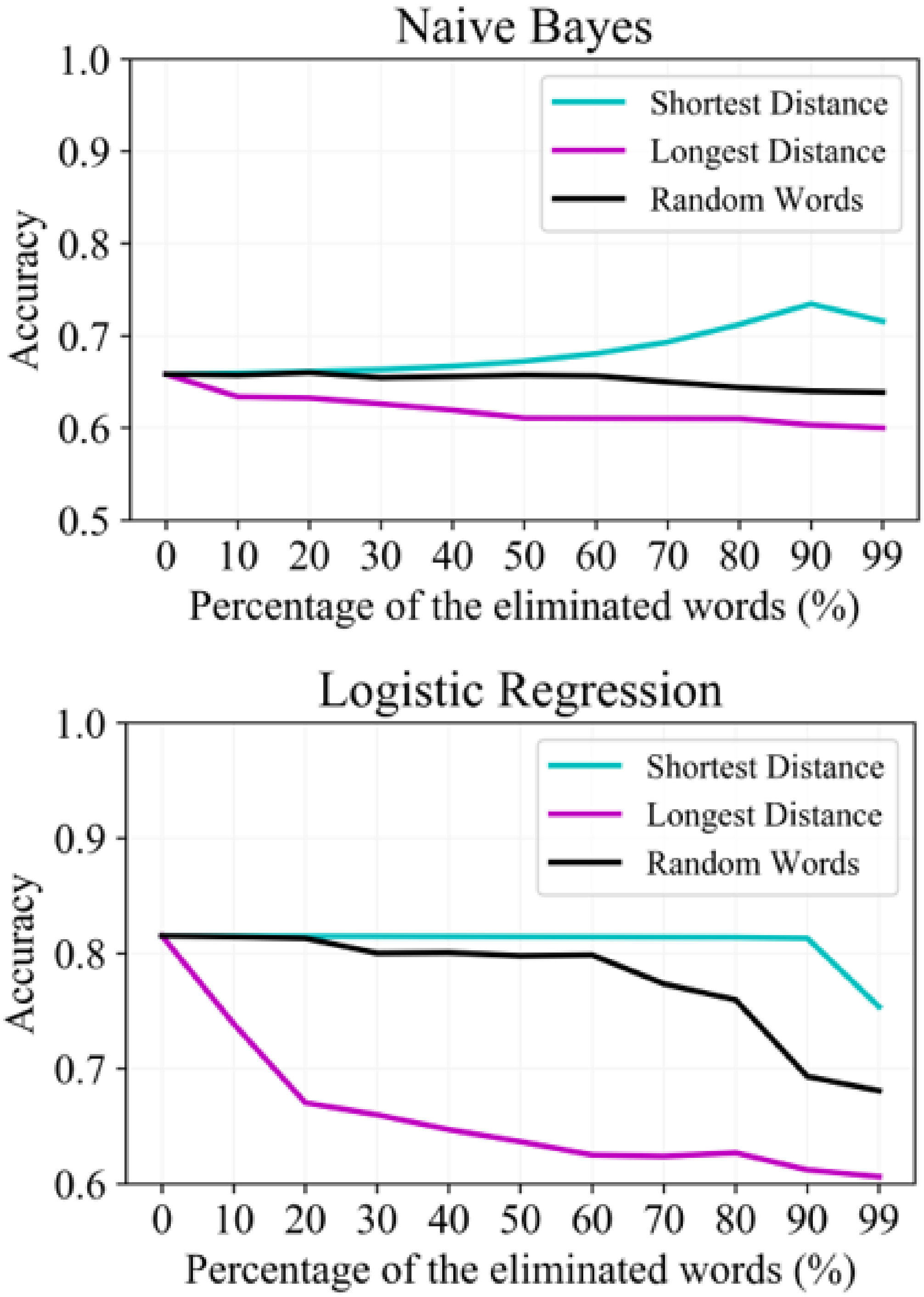


Figure 3.6: Classifier performance on Hansard Speeches dataset when eliminating different percentages of words based on three different criteria: shortest distance from hyperplane, longest distance from hyperplane, and random word elimination.

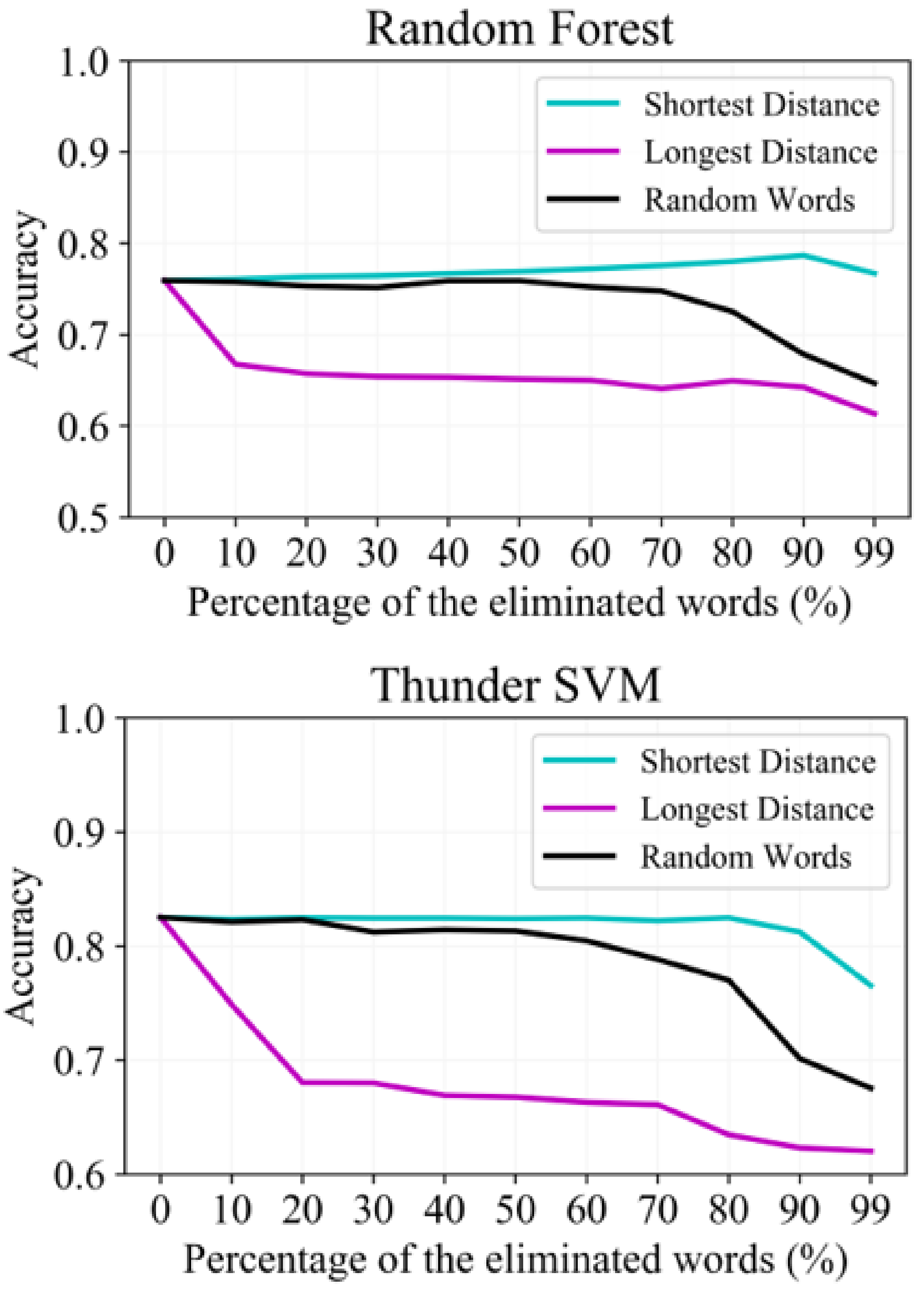


Figure 3.7: Classifier performance on Hansard Speeches dataset when eliminating different percentages of words based on three different criteria: shortest distance from hyperplane, longest distance from hyperplane, and random word elimination.

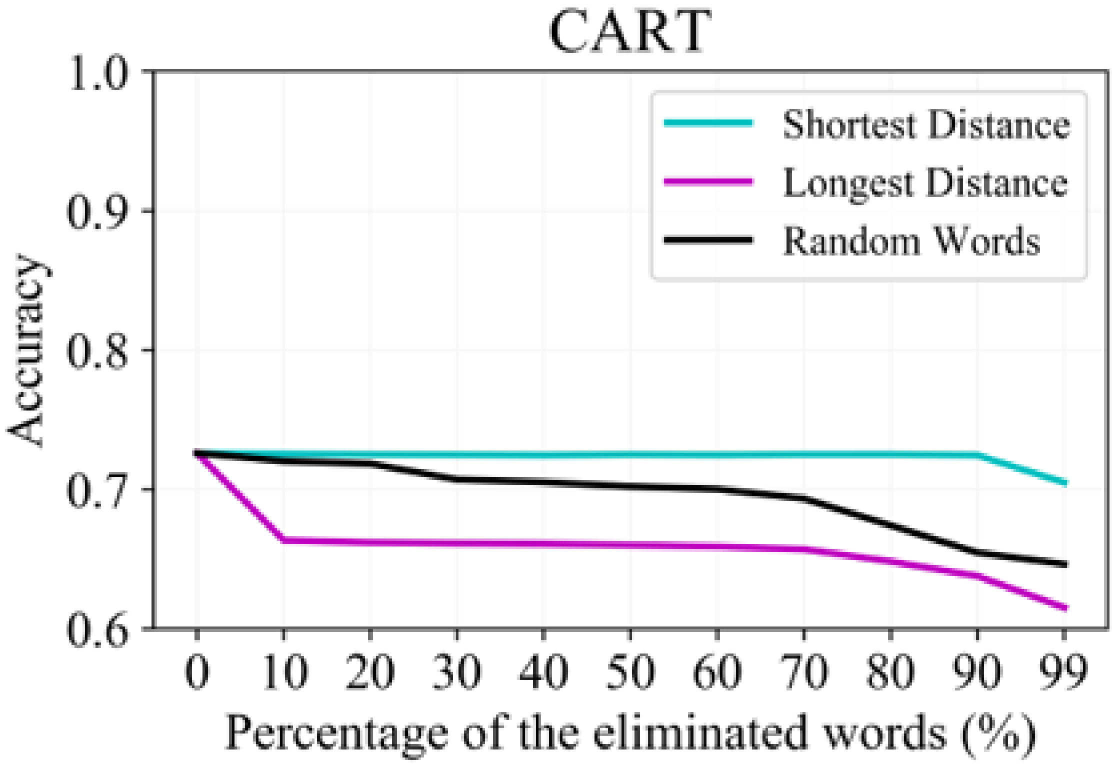


Figure 3.8: Classifier performance on Hansard Speeches dataset when eliminating different percentages of words based on three different criteria: shortest distance from hyperplane, longest distance from hyperplane, and random word elimination.

proposed approach outperforms the χ^2 and MI approaches from 60% to 90% elimination. The average accuracy of the Naive Bayes classifier increased at 99% elimination with respect to the reference experiment in all approaches, where the hyperplane-based approach had the same average accuracy as the χ^2 and higher than MI approach. The average accuracy of the Random Forest classifier is increased by 2%, and 3% when removing 90% of domain-specific words using our proposed approach, and χ^2 approach, respectively. MI, in conrast, increased the Random Forest accuracy with less than 1%. The accuracy of other classifiers (Logistic Regression, Thunder SVM, and CART) stay the same after any elimination percentage. In the case of removing 90% of domain-specific words, the χ^2 approach and hyperplane-based enhanced the accuracy of both Naive Bayes and Random Forest classifiers compared with the reference experiment.

To further investigate the performance of our proposed hyperplane-based approach, we break the estimation of accuracy down by year instead of calculating the average across years. Fig. 3.12 shows trends in accuracy for two classifiers, Naive Bayes and Random Forest, for two elimination categories: 90% of domain-specific words vs 0% elimination (our reference category). We notice that the accuracy of both classifiers using our hyperplane-based approach outperform the χ^2 for the years from 1980 to 1999, and outperforms MI for all the years. Our approach further increases the performance of Naive Bayes by 12% for the year 1992 from 0.64 to 0.76. For Random Forest, our proposed approach increases the accuracy by 4% in 1981.

Overall, the three approaches play a key role in reducing the dimensionality of the corpus, which means reducing the execution time of the classification problem, and the hyperplane-based approach achieved comparable results with χ^2 and MI. We demonstrate the impact of feature elimination on execution time in Tables 4.3. Table 4.3 presents the average execution time for all years of the five classifiers per elimination percentage. We can see from this table that when the percentage of elimination increases, the drop in the average execution time increases dramatically. We further compare the execution time for the three elimination strategies hyperplane-based, χ^2, and MI. We find that the χ^2 approach has the lowest execution time 8 seconds, while MI has the highest. Our approach outperforms the MI, which only needs 152 seconds to generate the list of words for each year, while MI needs 1743 seconds, more than ten times longer.

Finally, we investigate the extent to which our approach generates word lists that are different from those generated by the two other approaches. To this end, we look at the intersection of the word sets generated by the three approaches, comparing our method against the other two

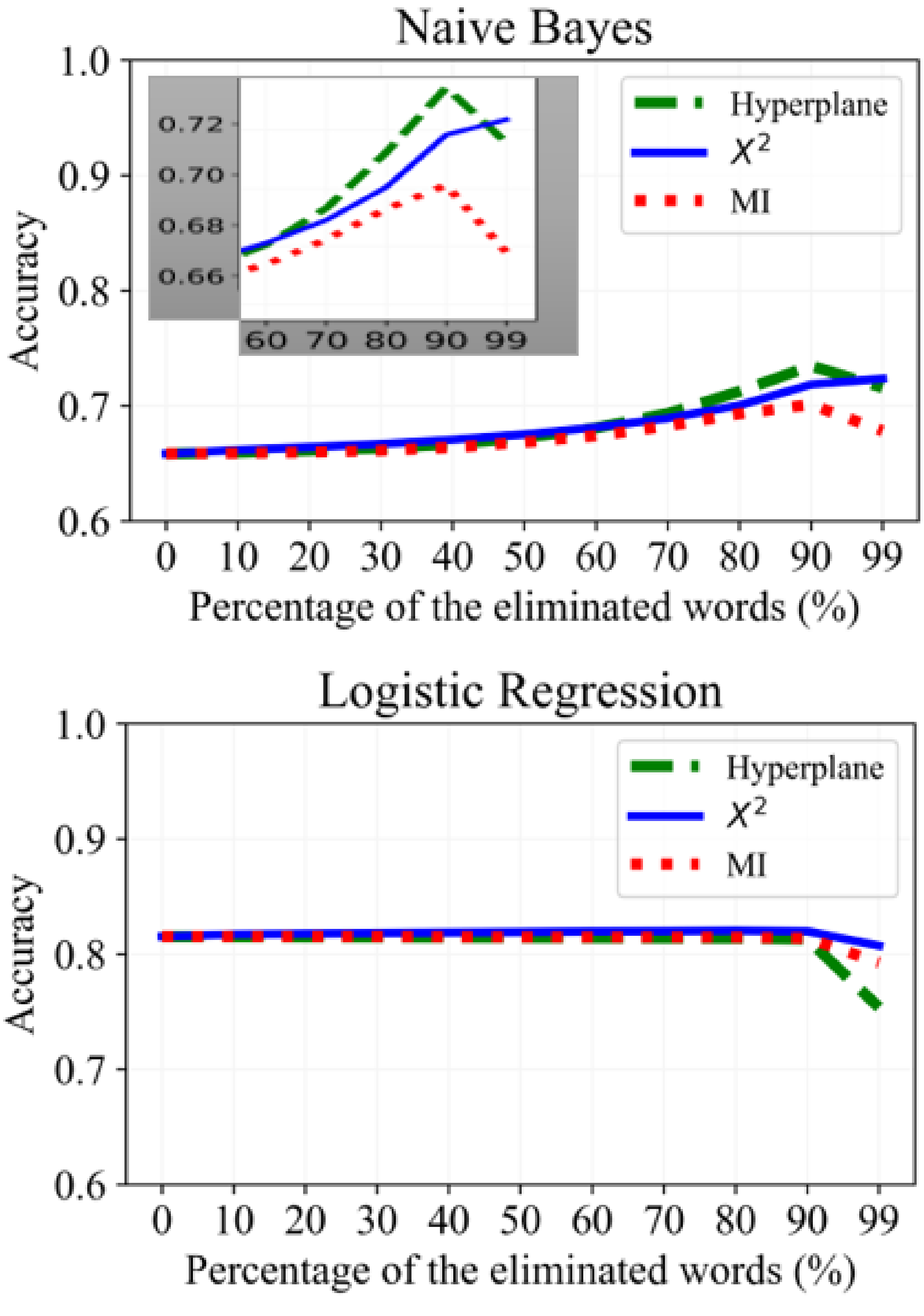


Figure 3.9: Average classifier accuracy comparison on Hansard Speeches for all the years, using three different approach: hyperplane-based, χ^2 and MI by eliminating different percentages of words starting from shortest to longest distance from hyperplane, lowest to highest χ^2 value and lowest to highest mutual information.

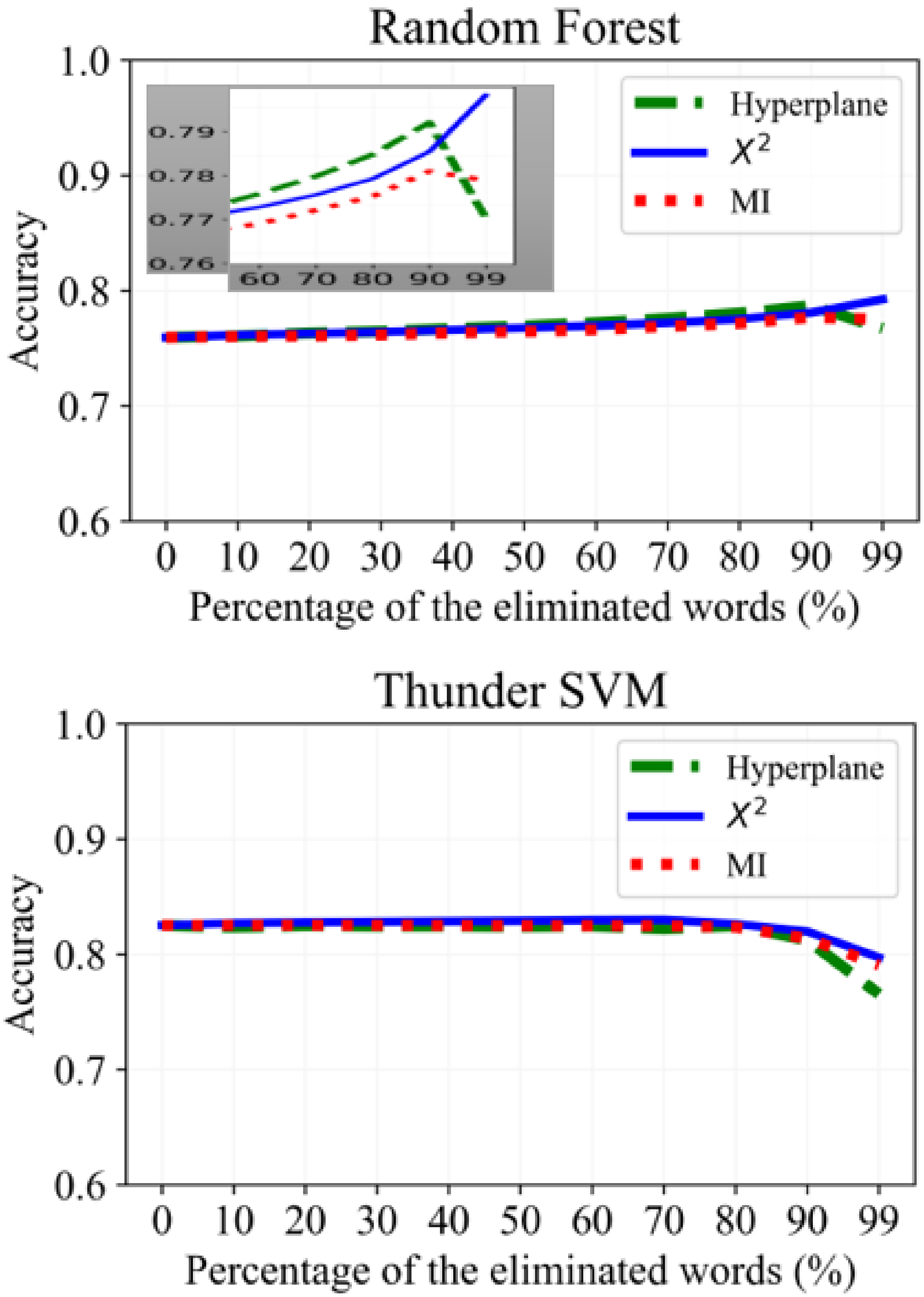


Figure 3.10: Average classifier accuracy comparison on Hansard Speeches for all the years, using three different approach: hyperplane-based, χ^2 and MI by eliminating different percentages of words starting from shortest to longest distance from hyperplane, lowest to highest χ^2 value and lowest to highest mutual information.

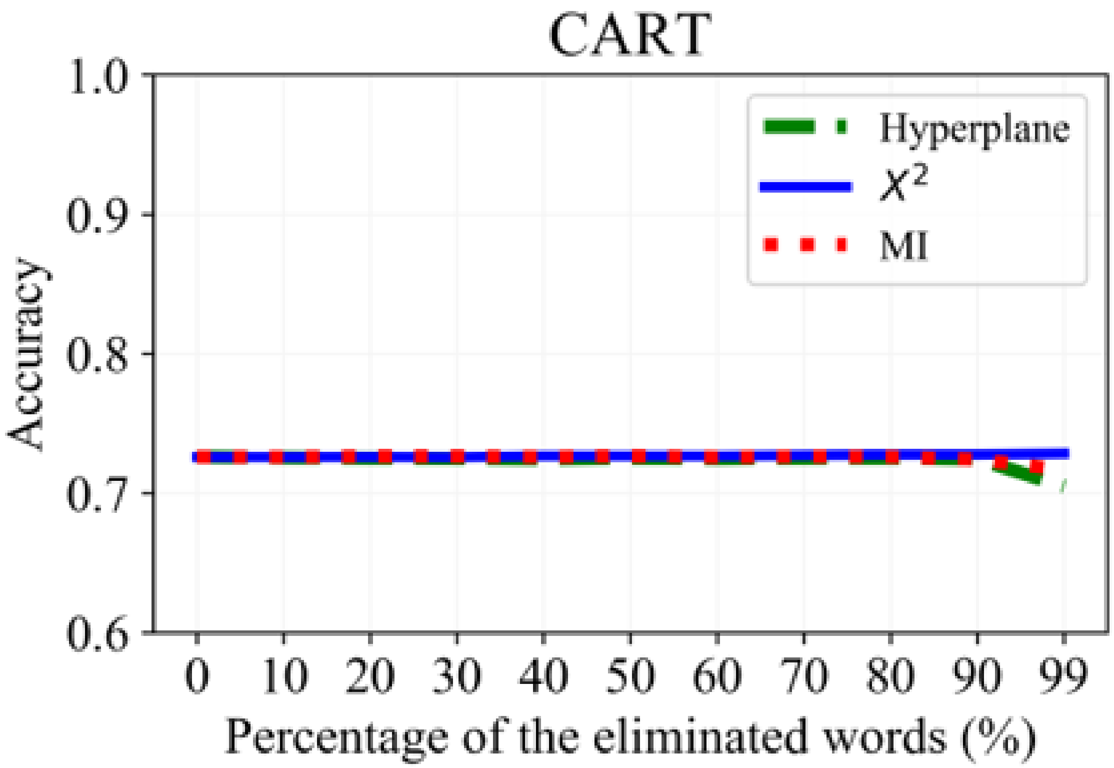


Figure 3.11: Average classifier accuracy comparison on Hansard Speeches for all the years, using three different approach: hyperplane-based, χ^2 and MI by eliminating different percentages of words starting from shortest to longest distance from hyperplane, lowest to highest χ^2 value and lowest to highest mutual information.

Table 3.1: Average Execution Time for each classifier in seconds

Percentage of elimination	NB	LR	RF	SVM	CART
0%	0.37	35	1296	733	533
10%	0.36	31	1260	686	525
20%	0.35	28	1191	661	521
30%	0.33	26	1127	643	506
40%	0.33	24	1099	626	502
50%	0.31	22	1034	611	488
60%	0.29	21	989	602	480
70%	0.28	19	977	589	477
80%	0.27	18	946	575	475
90%	0.25	15	830	556	448
99%	0.09	4	565	213	77

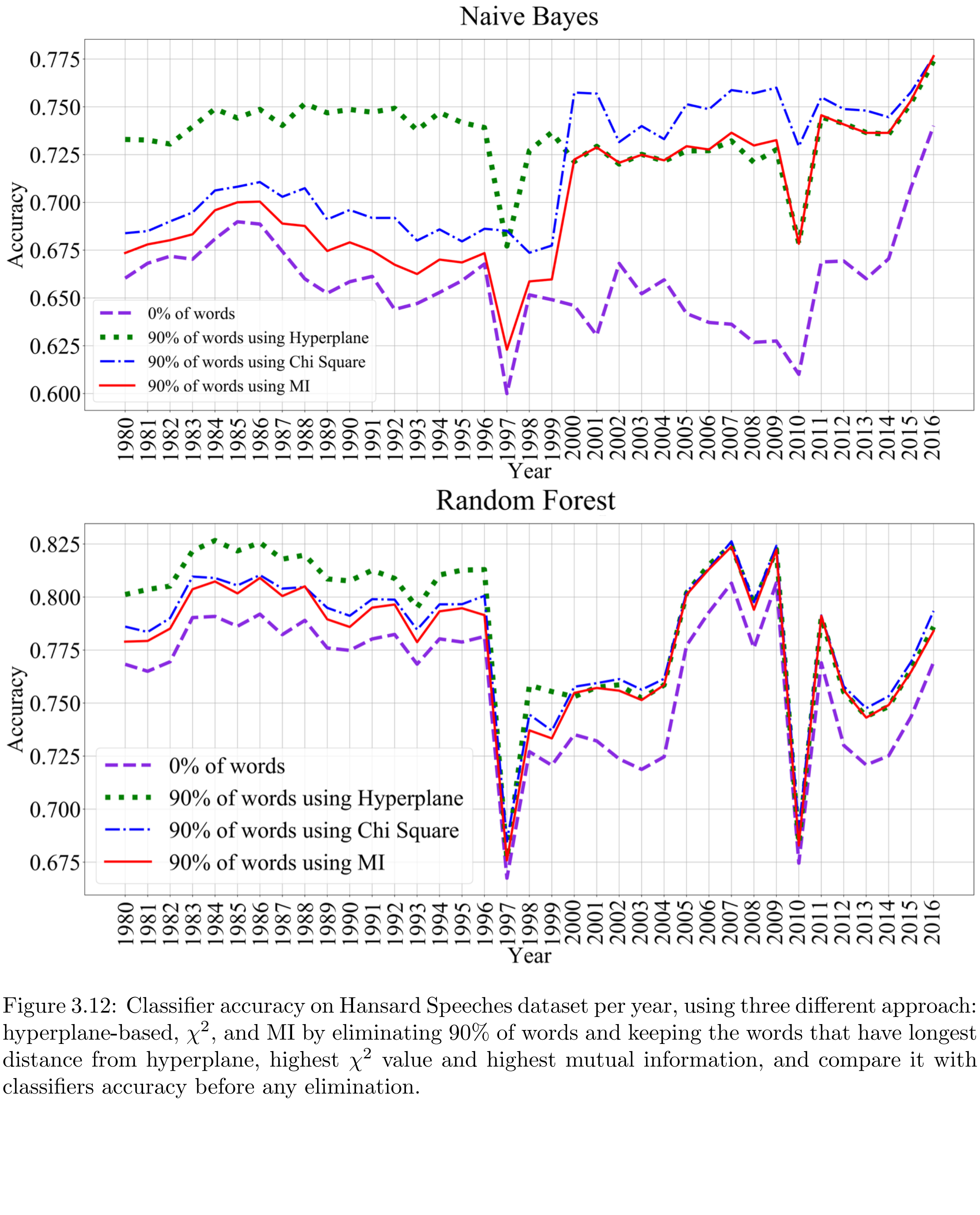


Figure 3.12: Classifier accuracy on Hansard Speeches dataset per year, using three different approach: hyperplane-based, χ^2, and MI by eliminating 90% of words and keeping the words that have longest distance from hyperplane, highest χ^2 value and highest mutual information, and compare it with classifiers accuracy before any elimination.

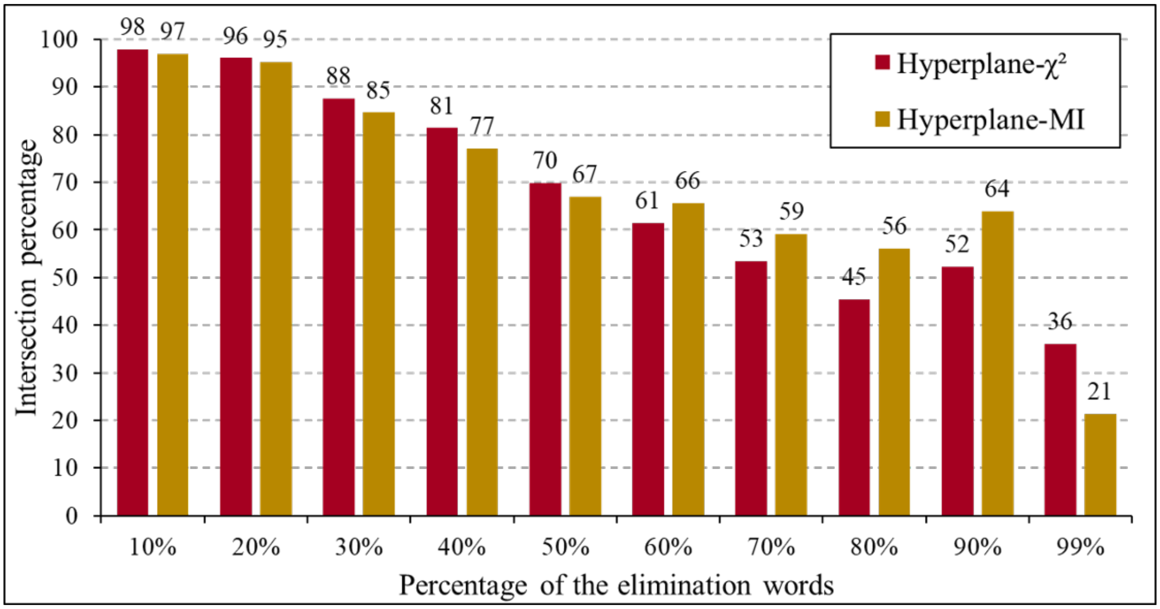


Figure 3.13: Intersection between Hyperplane-based approach with the χ^2 and MI approach against the range of elimination percentages.

for different percentage categories of elimination – see Fig. 3.13. The intersection is defined by the percentage of identical words between two approaches that remain after elimination, divided by the total number of remaining words after elimination. Fig. 3.13 shows that the overlap between the methodologies is not high when eliminating 99% of words, which means that our hyperplane-based approach extracts words that are different from those extracted by the other two approaches while maintaining the same accuracy. We present a sample of the words that used to distinguish between two classes after eliminating 99% of words in Table 3.2. We present examples of the remaining words after eliminating 99% of words per approach for the year 1980. These are the words with the largest distance from the hyperplane, highest χ^2, and highest MI.

IMDB Movie Review Dataset

The accuracy of the five classifiers with respect to the range elimination percentages for the shortest distance, longest distance, and random words estimated from this dataset show the same trend as the Hansard Speeches dataset, as is illustrated in Figures. 3.14, 3.15, and 3.16.

Figures. 3.17, 3.18, and 3.19 shows the classifier performance of testing the IMDB movie review dataset for the hyperplane-based approach and the two feature selection methods using five classifiers. It can be noticed that the accuracy for the Naive Bayes and Thunder SVM classifiers

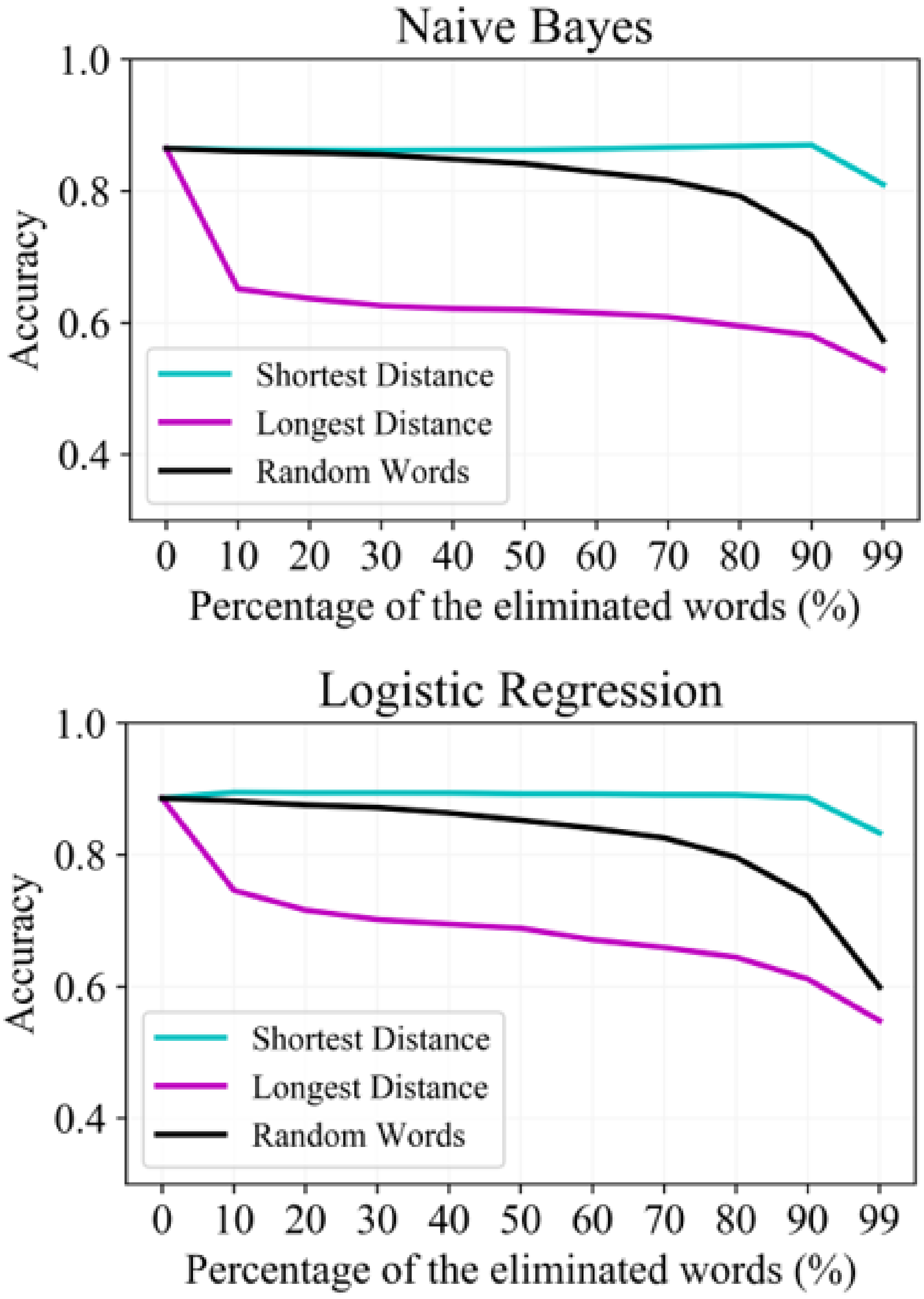


Figure 3.14: Classifiers performance on IMDB movie review dataset by eliminating different percentage of words that have shortest, longest distance from hyperplane and by eliminating random words.

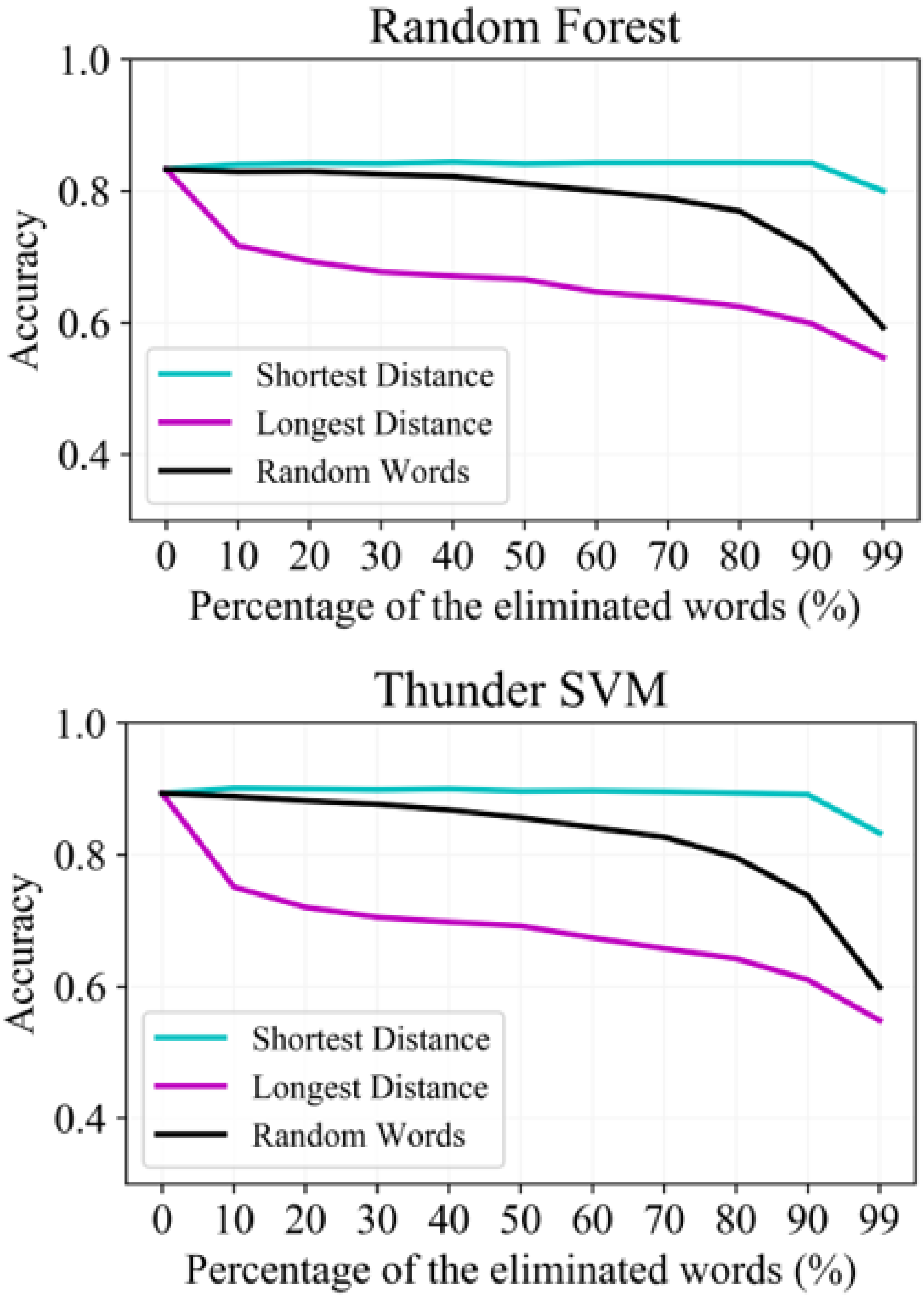


Figure 3.15: Classifiers performance on IMDB movie review dataset by eliminating different percentage of words that have shortest, longest distance from hyperplane and by eliminating random words.

Table 3.2: Presents words from Hansard Speeches dataset for the year 1980 that are deemed most important by the hyperplane, χ^2, and Mutual Information

Hyperplane words	χ^2	MI words
sparkbrook	minister	not
disservice	friend	for
benchers	secretary	hon
dispatch	state	and
duchy	he	is
sidcup	you	in
engagements	state	that
orme	conservative	to
mislead	she	of

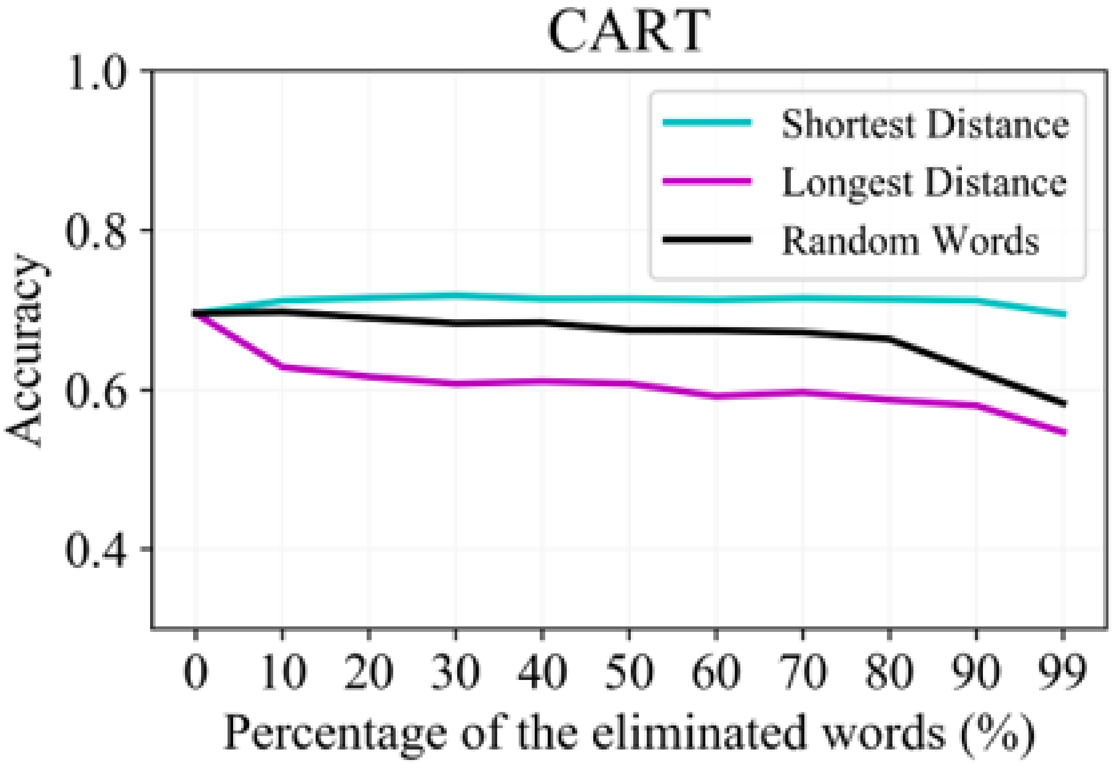


Figure 3.16: Classifiers performance on IMDB movie review dataset by eliminating different percentage of words that have shortest, longest distance from hyperplane and by eliminating random words.

are improved by using the three approaches when eliminating 90% of words. The hyperplane-based approach performs better than the MI approach when eliminating 90% of the domain-specific words under Naive Bayes. The χ^2 is slightly higher than the hyperplane-based approach and the MI at all elimination percentages under Naive Bayes. The accuracy of the other classifiers (Logistic Regression, Random Forest, and CART) is stable before and after eliminating 90% of words using all approaches.

When removing 99% of domain-specific words, only χ^2 maintains the same accuracy, while the performance of the other two approaches drops. We note that the drop in performance cannot be explained by the small number of data points at this elimination level alone. After eliminating 99% of the words, there are 2,221 words remaining, with an average of 45-80 words per review, depending on which approach is being used, which should be enough distinguishable words to classify the data. As such, the drop in performance is a shortcoming of the two methods, though only at this high level of word elimination.

The intersection of the remaining words between the proposed approach with the χ^2 and MI approach against the range of elimination show the same trend as we observed in the Hansard Speeches dataset, which is illustrated in Fig. 3.20. We also again present examples of the remaining words after eliminating 99% in Table 3.3.

Table 3.3: Presents words from IMDB Movie Review Dataset that are deemed most important by the hyperplane, χ^2, and Mutual Information

Hyperplane words	χ^2 words	MI words
waste	bad	the
renting	worst	and
crap	waste	of
stupid	awful	to
suck	great	this
bother	terrible	is
unwatchable	horrible	in
pile	excellent	It

Pubmed

The curated Pubmed dataset includes abstracts from 100 pairs of different journals. For each pair, the accuracy of all classifiers is calculated per all elimination percentages. Then, the average accuracy for all pairs is calculated and presented. Figures. 3.21, 3.22, and 3.23 presents the

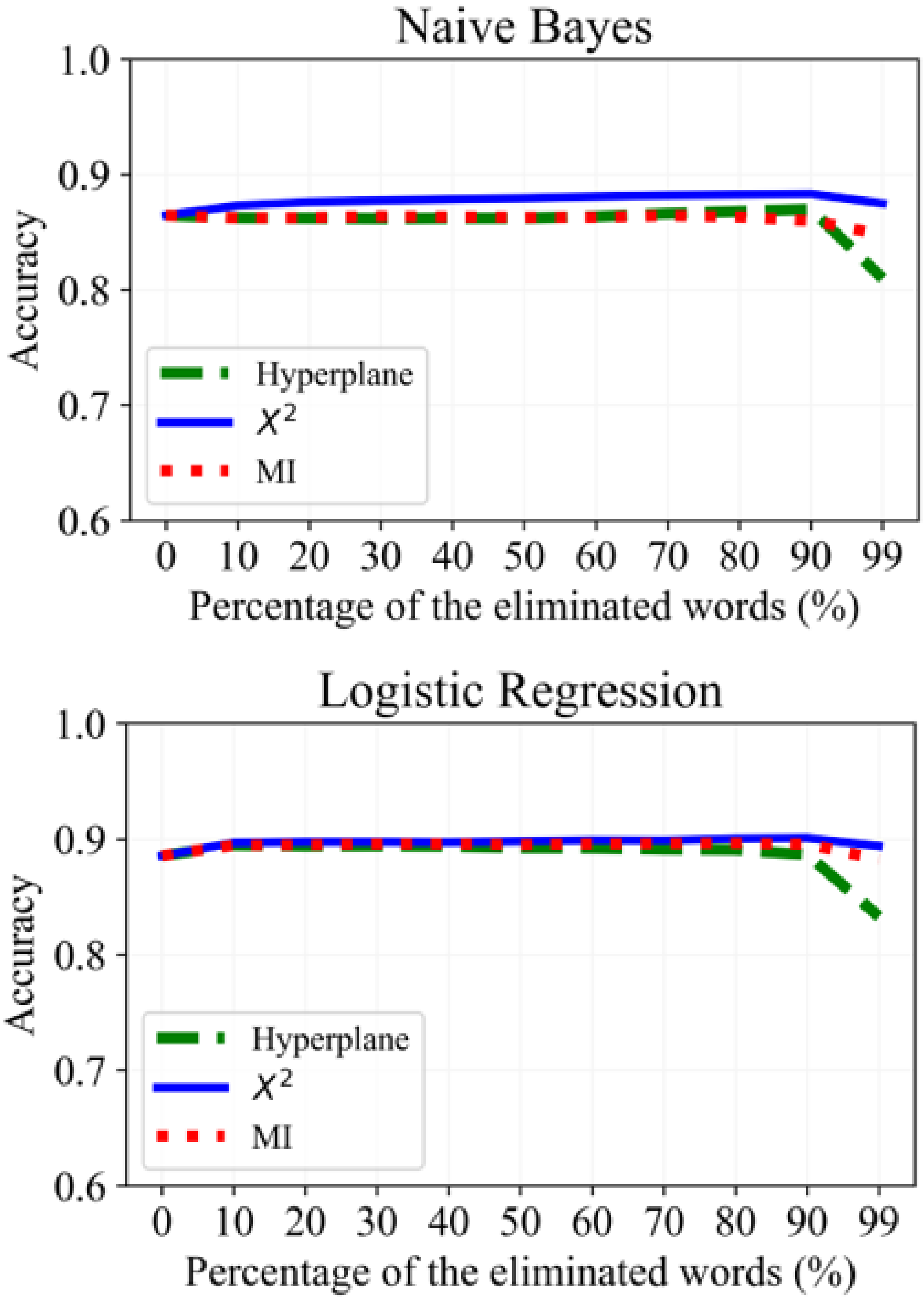


Figure 3.17: Classifier performance comparison on IMDB movie review dataset using three different approach: hyperplane-based, χ^2 and MI by eliminating different percentages of words starting from shortest to longest distance from hyperplane, lowest to highest χ^2 value and lowest to highest mutual information.

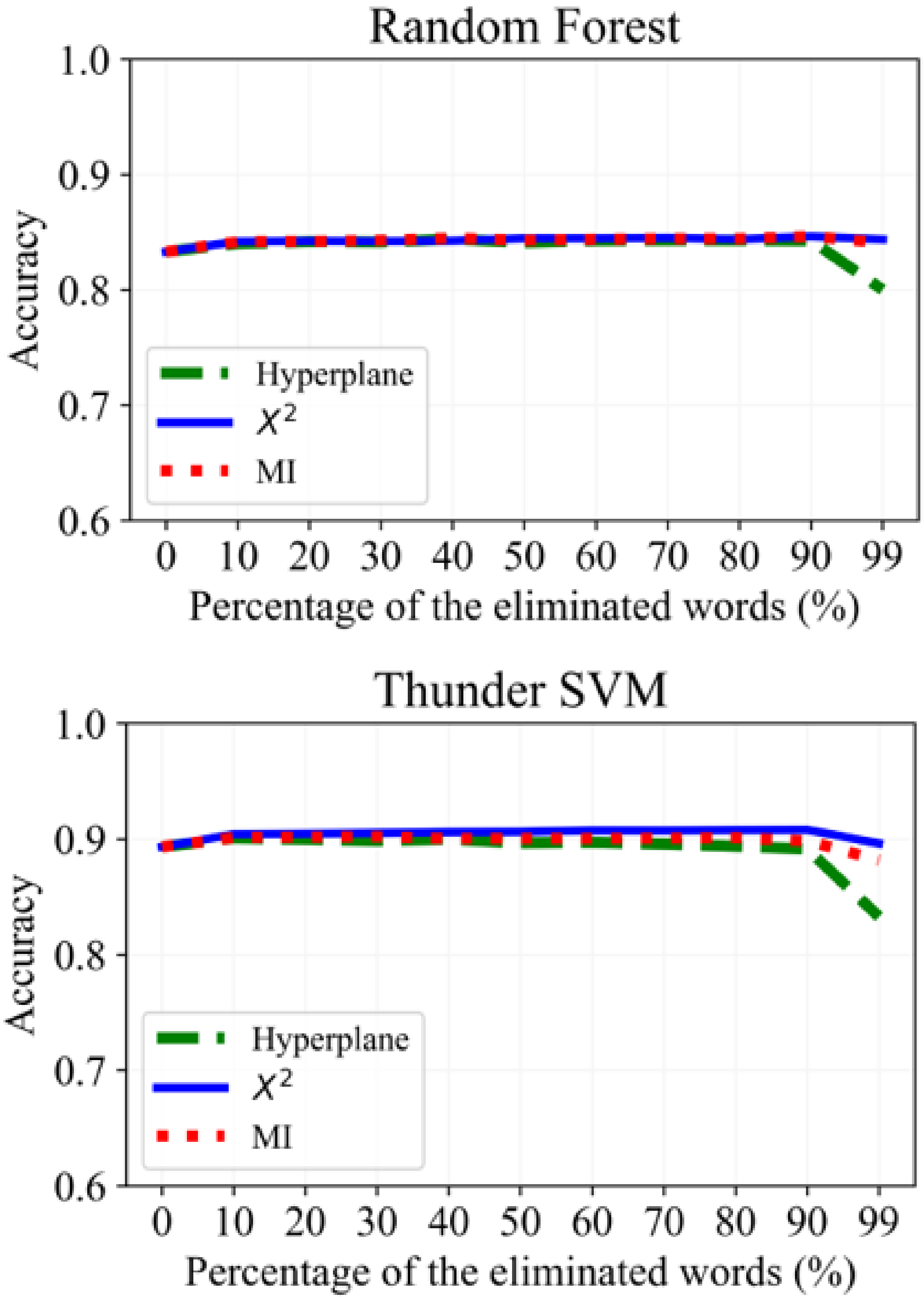


Figure 3.18: Classifier performance comparison on IMDB movie review dataset using three different approach: hyperplane-based, χ^2 and MI by eliminating different percentages of words starting from shortest to longest distance from hyperplane, lowest to highest χ^2 value and lowest to highest mutual information.

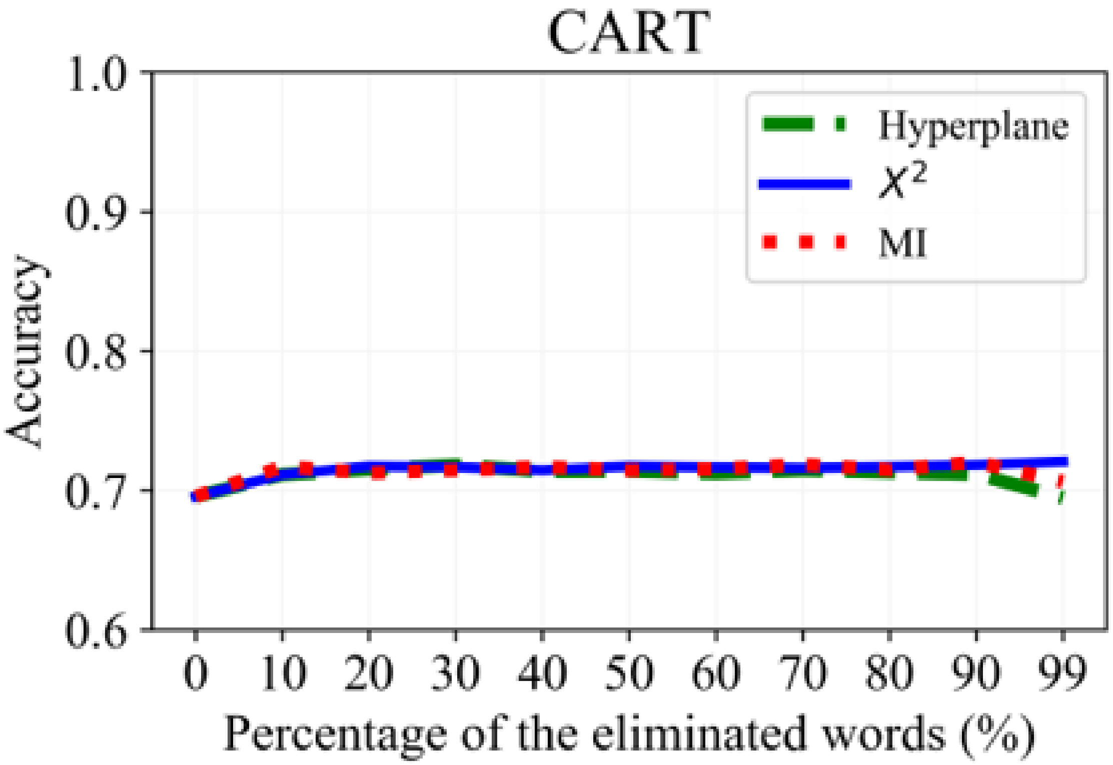


Figure 3.19: Classifier performance comparison on IMDB movie review dataset using three different approach: hyperplane-based, χ^2 and MI by eliminating different percentages of words starting from shortest to longest distance from hyperplane, lowest to highest χ^2 value and lowest to highest mutual information.

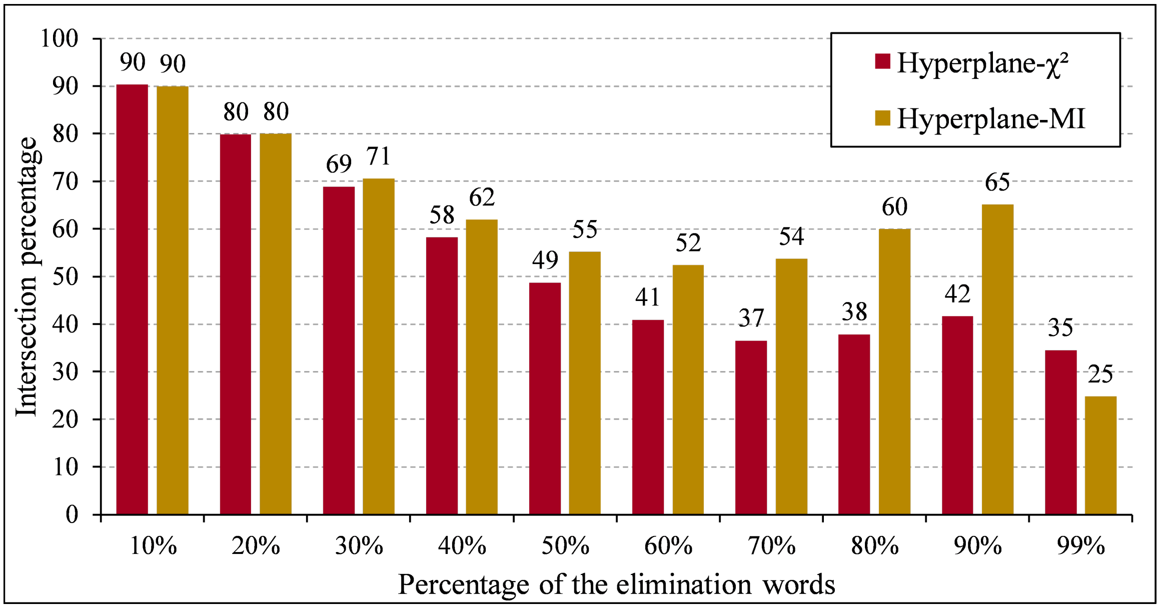


Figure 3.20: Intersection between Hyperplane-based approach with the χ^2 and MI approach against the range of elimination percentages.

average accuracy of the five classifiers with respect to the range of elimination percentages for the shortest distance, longest distance, and random words to the Pubmed dataset have the same trend to the Hansard Speeches dataset and the IMDB movie review dataset.

Figures. 3.24, 3.25, and 3.26 shows the average classifier accuracy of testing the 100 pairs of Pubmed dataset for the hyperplane-based and two feature selection methods and five classifiers. The classifying accuracy for the Naive Bayes is improved by using the hyperplane-based and χ^2. The hyperplane-based approach outperforms the MI approach when eliminating 90% of the domain-specific words for Naive Bayes, Logistic Regression and Thunder SVM. The accuracy of the other classifiers (Random Forest, and CART) is stable before and after eliminating 90% of words using all approaches.

We present the intersection of the remaining words between the proposed approach with the χ^2 and MI approach against the range of elimination on one pair, Cell and Journal of Prosthetic Dentistry, in Fig. 3.27. We find that the intersections have the same trend as those observed on the two other datasets. We present a sample of the remaining words after eliminating 99% of words in Table 3.4.

Table 3.4: Presents words of the Cell and Journal of Prosthetic Dentistry from Pubmed dataset that are deemed most important by the hyperplane, χ^2, and Mutual Information

Hyper plane words	χ^2 words	MI words
computeraided	denture	the
restorations	cell	of
provisional	we	and
castings	dental	in
impressions	implant	to
crowns	resin	that
fabrication	teeth	is

3.5 Conclusion

This study proposes a novel mathematical approach for detecting domain-specific words called the hyperplane-based approach. This new approach depends on the notion of low dimensional representation of the word in vector space and its distance from the hyperplane, where the domain-specific words are defined as the words with the shortest distance from the separating hyperplane. The performance of the proposed approach is quantified by the accuracy and the

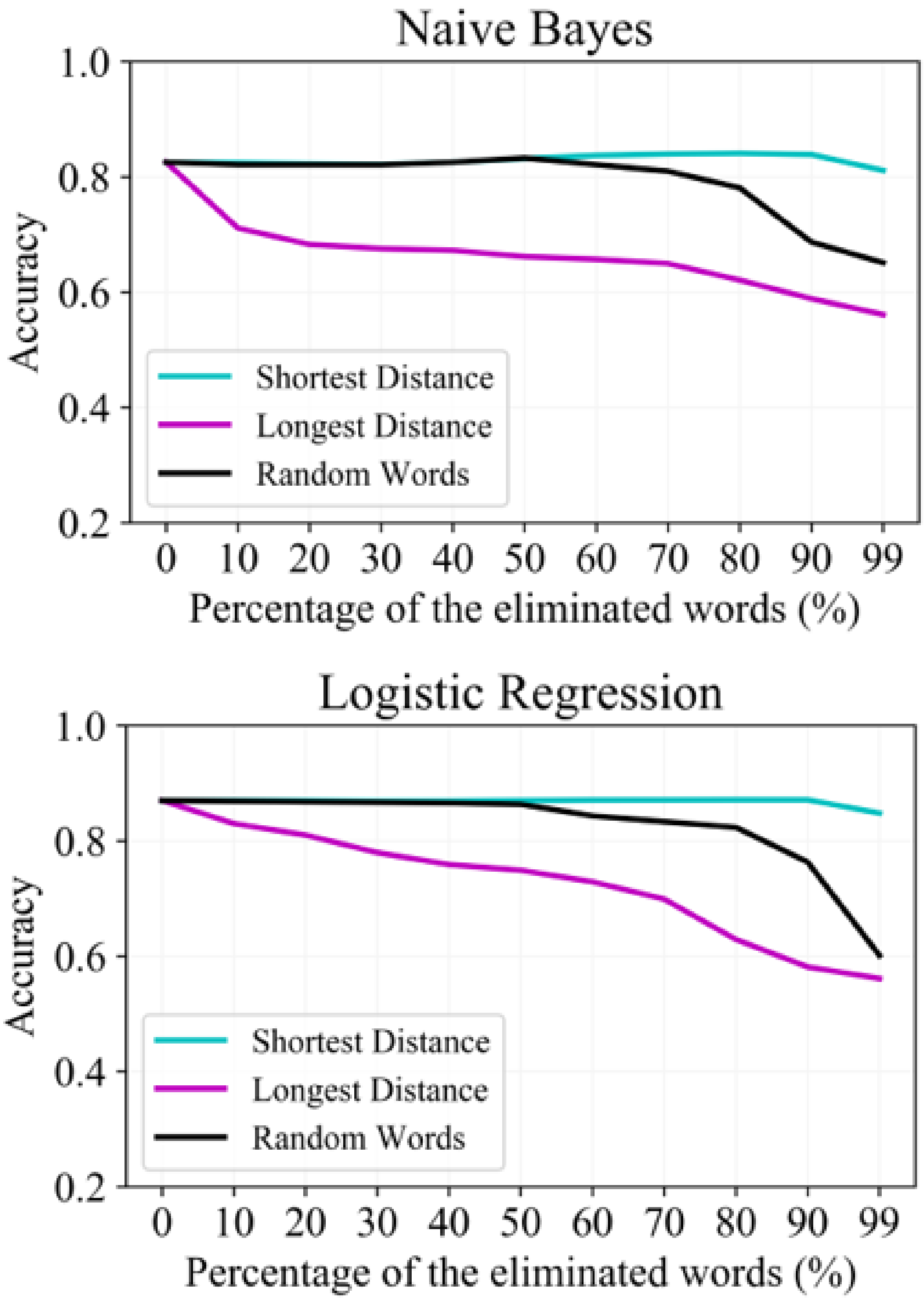


Figure 3.21: Classifiers performance on 100 pairs of Pubmed dataset by eliminating different percentage of words that have shortest, longest distance from hyperplane and by eliminating random words.

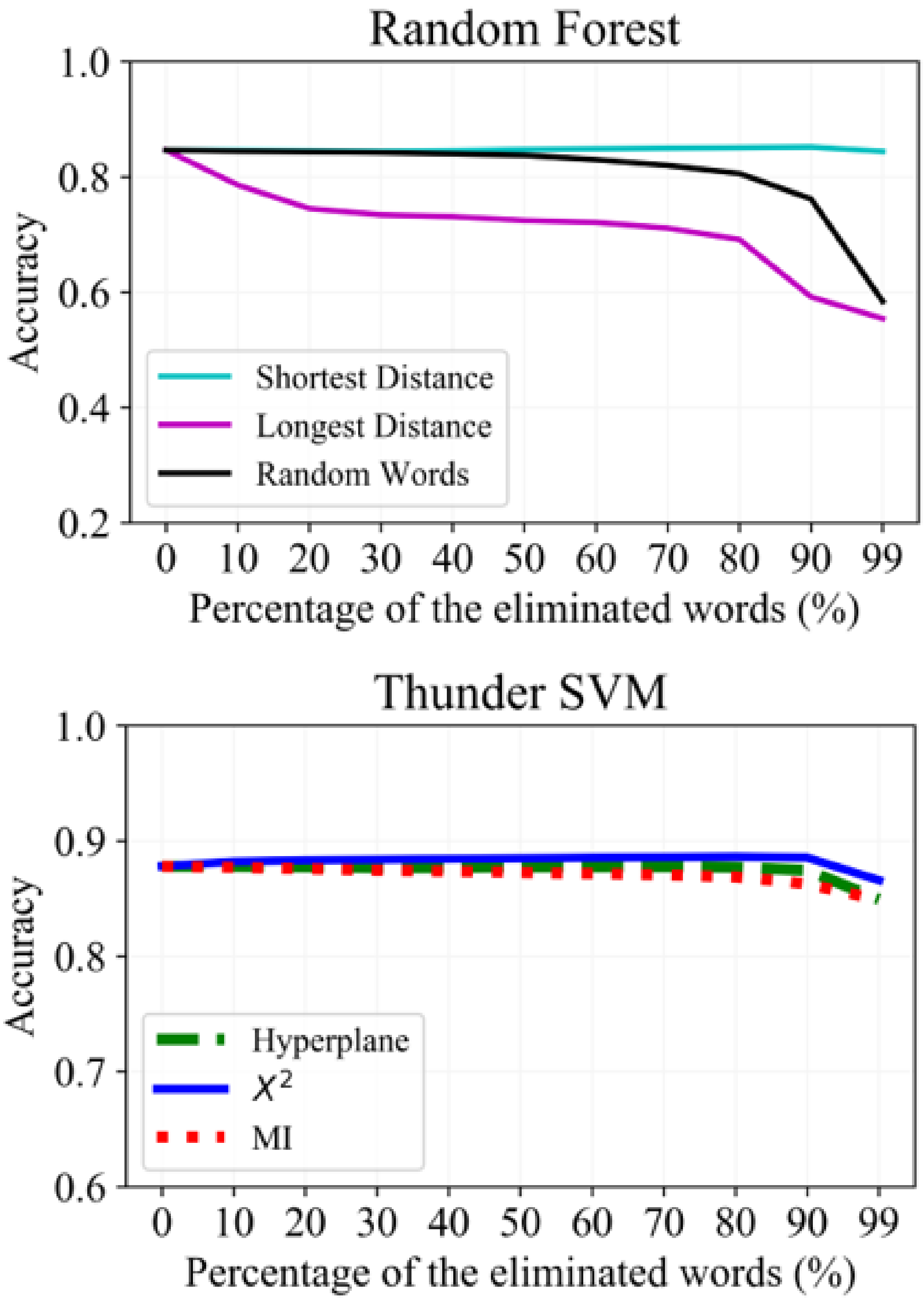


Figure 3.22: Classifiers performance on 100 pairs of Pubmed dataset by eliminating different percentage of words that have shortest, longest distance from hyperplane and by eliminating random word.

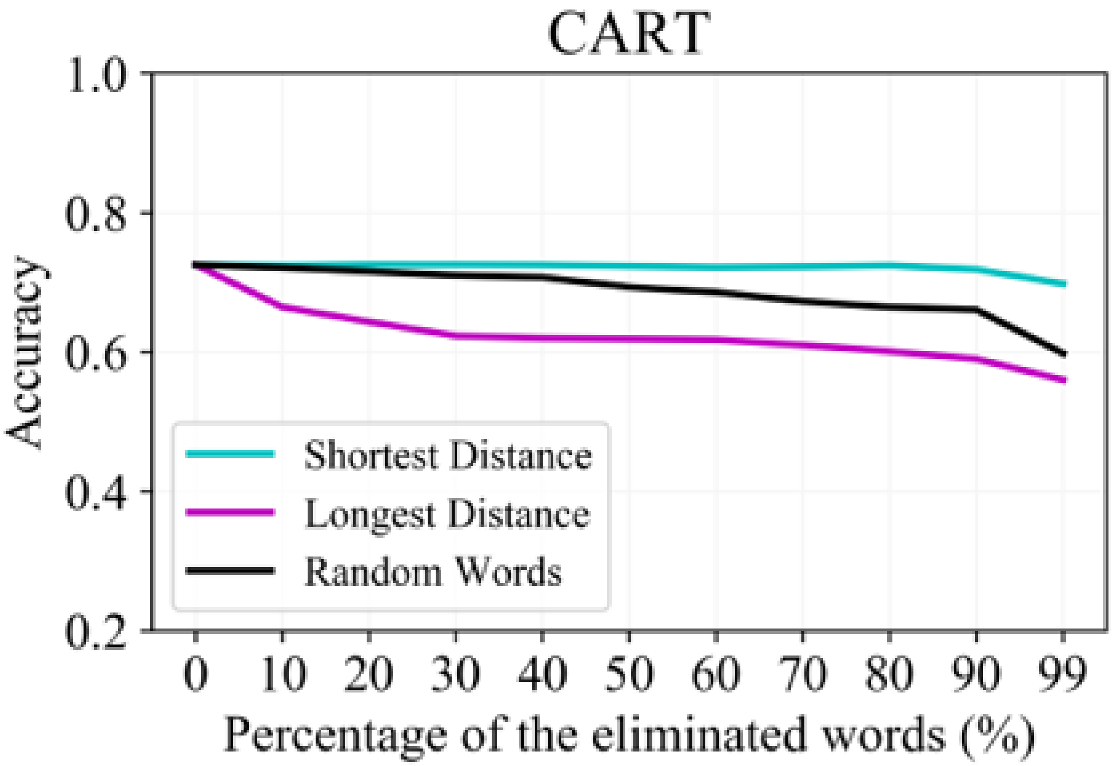


Figure 3.23: Classifiers performance on 100 pairs of Pubmed dataset by eliminating different percentage of words that have shortest, longest distance from hyperplane and by eliminating random words.

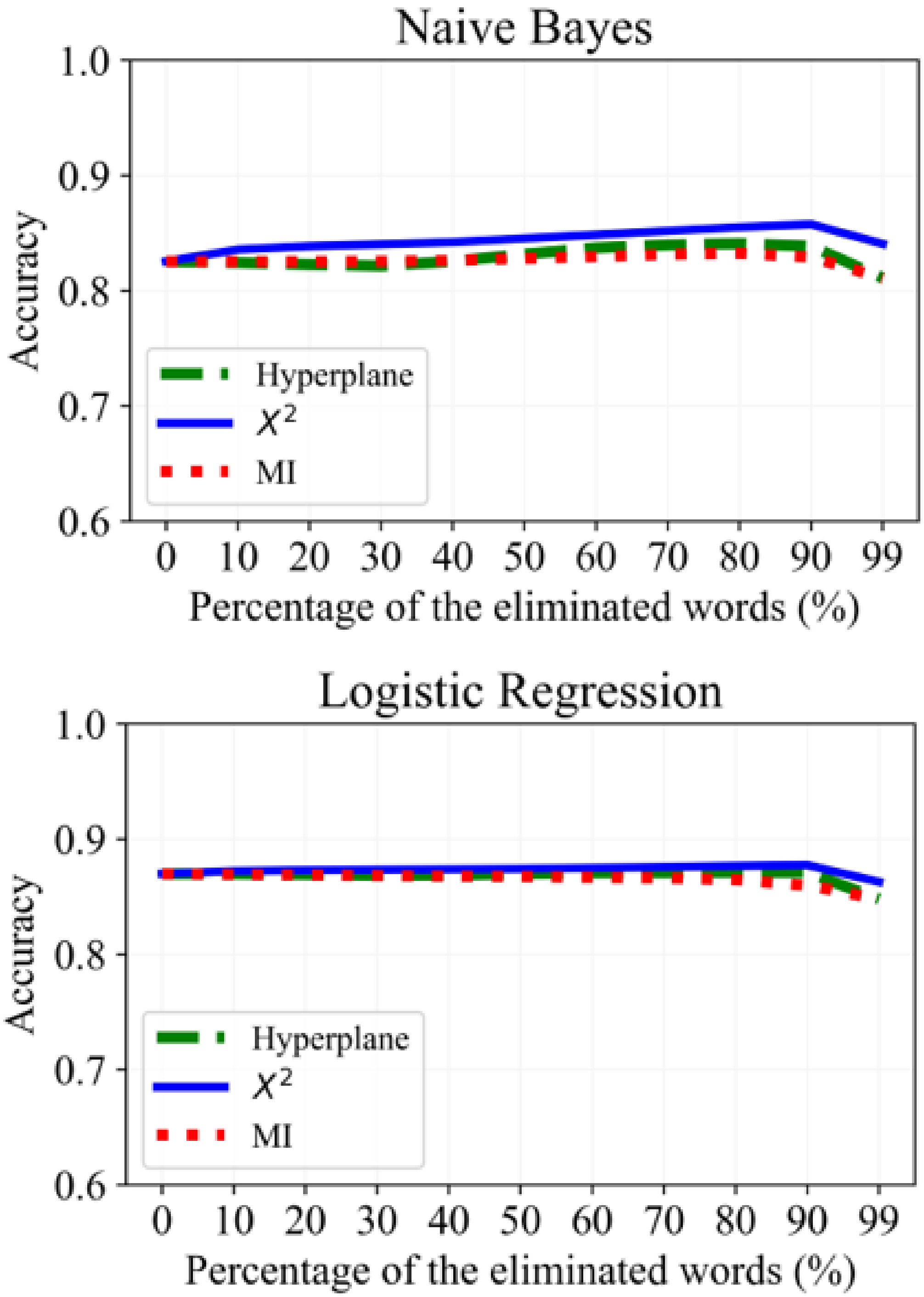


Figure 3.24: Classifier performance comparison on 100 pairs of Pubmed datasets using three different approach: hyperplane-based, χ^2 and MI by eliminating different percentages of words starting from shortest to longest distance from hyperplane, lowest to highest χ^2 value and lowest to highest mutual information.

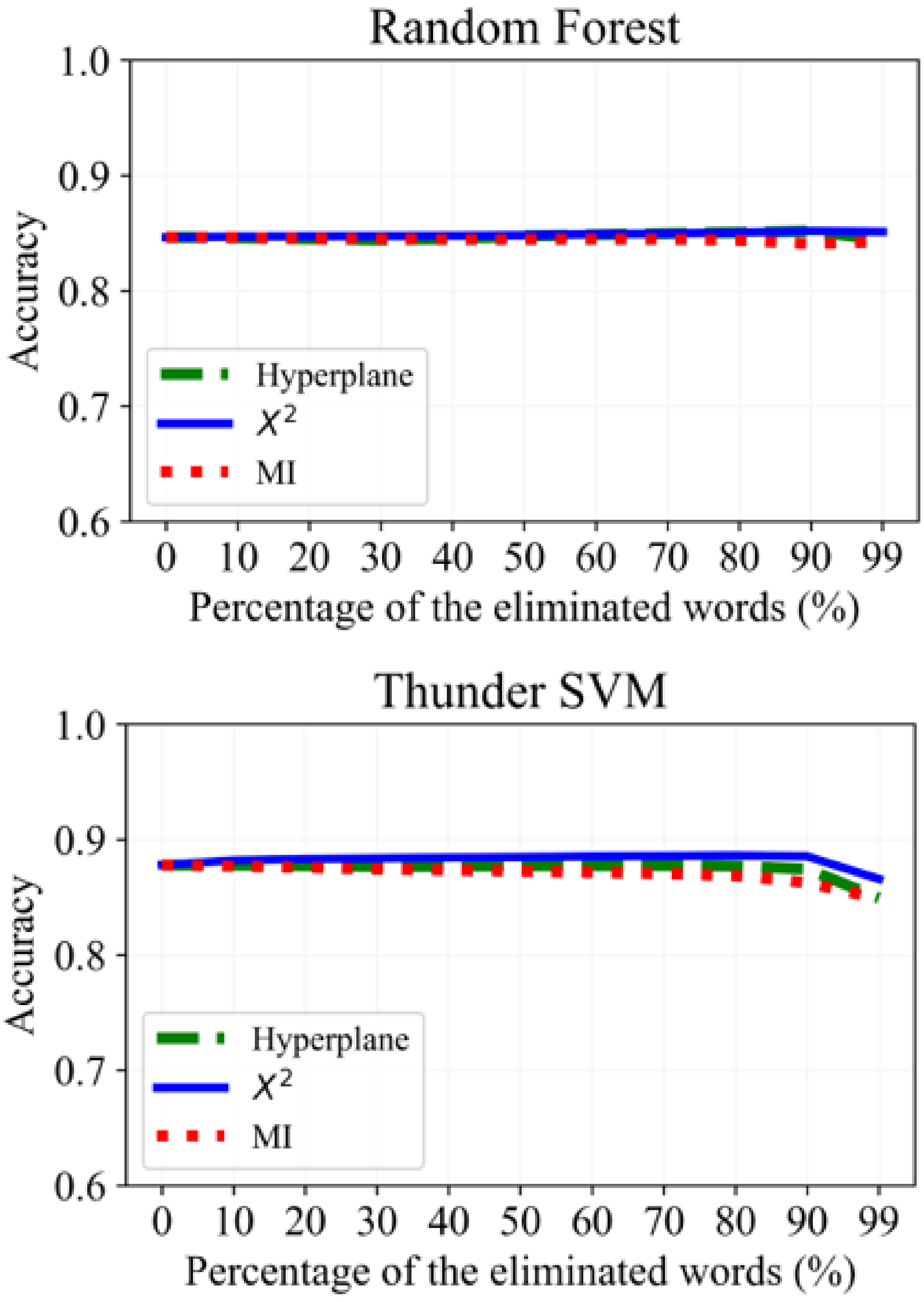


Figure 3.25: Classifier performance comparison on 100 pairs of Pubmed datasets using three different approach: hyperplane-based, χ^2 and MI by eliminating different percentages of words starting from shortest to longest distance from hyperplane, lowest to highest χ^2 value and lowest to highest mutual information.

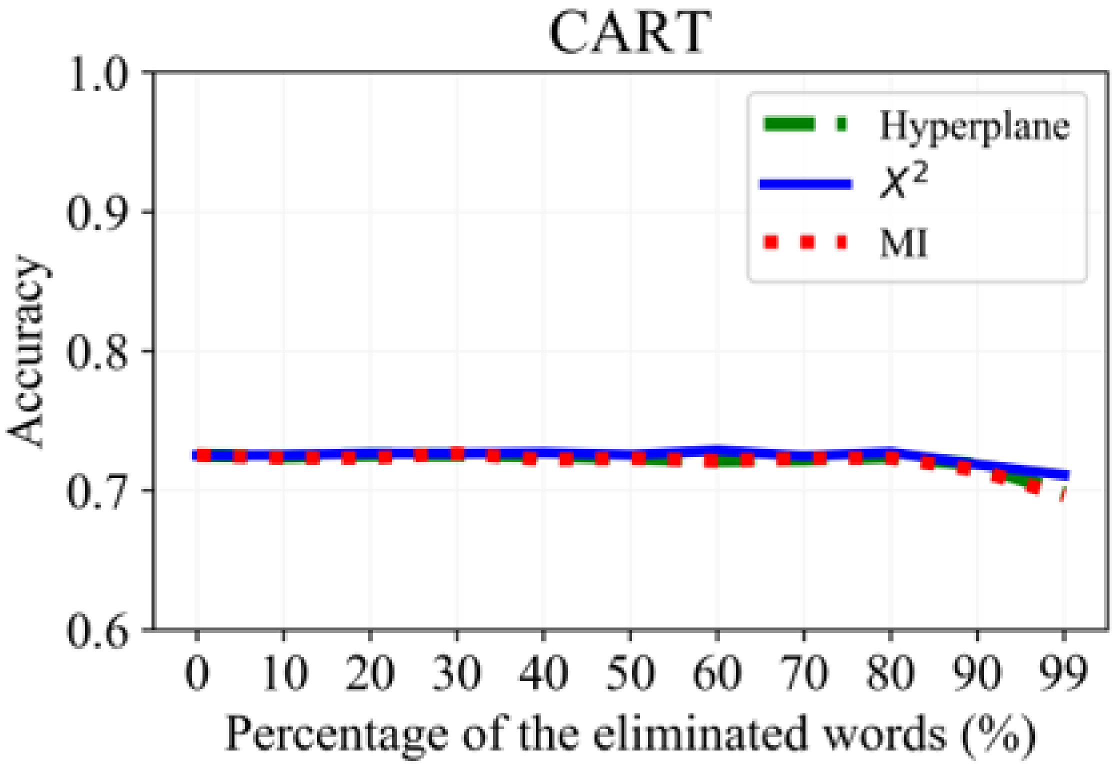


Figure 3.26: Classifier performance comparison on 100 pairs of Pubmed datasets using three different approach: hyperplane-based, χ^2 and MI by eliminating different percentages of words starting from shortest to longest distance from hyperplane, lowest to highest χ^2 value and lowest to highest mutual information.

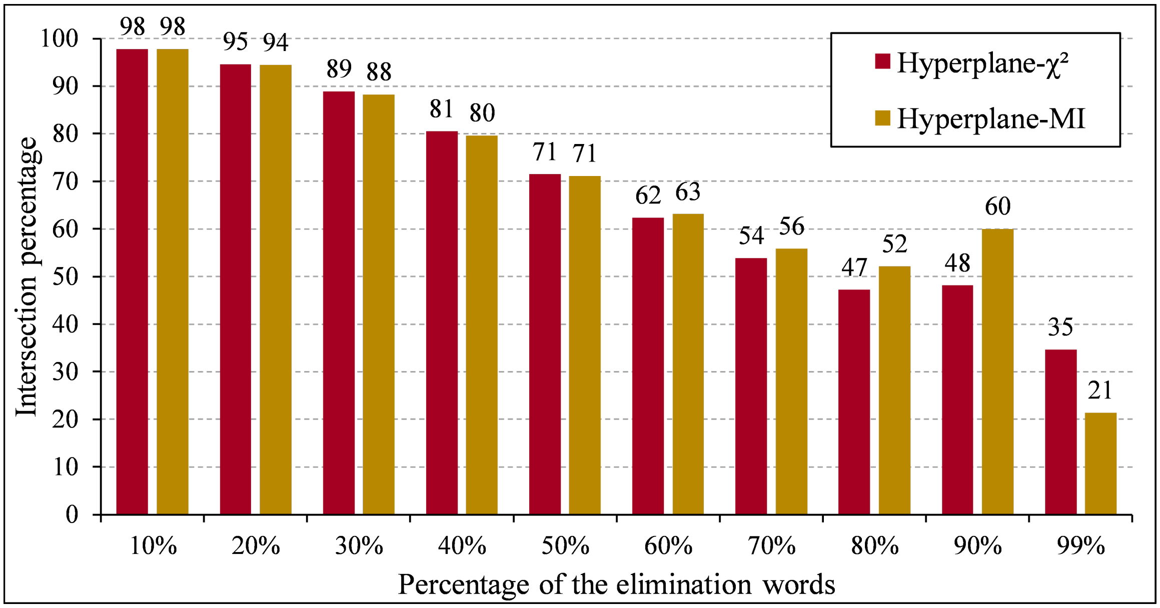


Figure 3.27: Intersection between Hyperplane-based aproach with the χ^2 and MI approach against the range of elimination percentages.

execution time of five classifiers: (1) Naive Bayes, (2) Random Forest, (3) Logistic Regression, (4) Thunder SVM, and (5) CART. This approach is validated using 138 sub-corpora from three datasets (Hansard Speeches, IMDB Movie Review, and Pubmed). Also, it is compared with two feature selection approaches, namely χ^2 and MI. For each feature selection method including the hyperplane-based approach and each corpus, various word elimination percentages are considered to find the optimal elimination percentage for each classifier and approach. The hyperplane-based approach generally improves the performance of the classifier and it achieved comparable performance with the χ^2 and MI. *However, our experiments indicate that qualitatively the eliminated words significantly differ from other approaches. In addition, the method is more robust to the erroneous elimination of important words.* The performance of our approach varies with respect to the classifier and the elimination percentages. For example, the Naive Bayes classifier presented the best improvement of accuracy before and after the elimination using our approach, and the optimal elimination percentage per our approach is 90% for all datasets. Finally, the proposed approach plays a key role in reducing the dimensionality of the corpus, which means reducing the classification execution time. The implementation of the hyperplane-based approach in different datasets is straight-forward and merging the hyperplane based-domain words with other feature selection domain words is a future research recommendation. Reproducibility: code and results can be found at `https://github.com/FarahAlshanik/Domain-Specific-Word-List`

Chapter 4

Thinking Outside the Box: Enriching Query Expansion Using External Data Source

Query expansion is the process of reformulating the original query by adding relevant words. Choosing which terms to add in order to improve the performance of the query expansion methods or to enhance the quality of the retrieved results is an important aspect of any information retrieval system. Adding words that can positively impact the quality of the search query or are informative enough play an important role in returning or gathering relevant documents that cover a certain topic can result in improving the efficiency of the information retrieval system. Typically, query expansion techniques are used to add or substitute words to a given search query to collect relevant data. In this thesis, we design and implement a pipeline of automated query expansion. We outline several tools using different methods to expand the query. Our methods depend on targeting emergent events in streaming data over time and finding the hidden topics from targeted documents using probabilistic topic models. We employ Dynamic Eigenvector Centrality to trigger the emergent events, and the Latent Dirichlet Allocation to discover the topics. Also, we use an external data source as a secondary stream to supplement the primary stream with relevant words and expand the query using the words from both primary and secondary streams. An experimental study is performed on Twitter data (primary stream) related to the events that happened during protests in

Baltimore in 2015. The quality of the retrieved results was measured using a quality indicator of the streaming data: tweets count, hashtag count, and hashtag clustering. Results indicate that adding words from the secondary stream can significantly improve the quality of search queries and return more relevant information that covers a certain topic. Another experimental study is performed to measure the effectiveness of the proposed methods to predict future emerging events or to predict the conversations from previous time intervals using the precision of a different set of hashtags. Results show that our method increases the precision.

4.1 Motivation

Social media streaming data (e.g., Twitter messages or Facebook posts) have become a primary source to analyze public opinion [78], track user sentiment [79, 80], or study emergent safety events, such as public health crises [81, 82, 83], natural disasters [84, 85, 86, 87], and political or social movements [88, 89]. Queries used to draw data from these high-volume, high-velocity, real-time sources typically require a set of words to filter the data. For example, tracking the public sentiment surrounding the COVID-19 pandemic on Twitter may rely on words such as "covid", "corona", and "lockdown" to filter out relevant messages. Such queries initiated by a static word list can be problematic because they reflect the domain expertise of users, and therefore reflect their biases or jargon, which can result in the exclusion of words required to retrieve relevant information. Static words also fail to keep up with changes in language and emergent words which results in incomplete data.

Information retrieval systems typically use query expansion techniques to enhance the initial user query, e.g., by adding inflected forms, cognates, and related words manually retrieved from the text. [90, 91]. We propose a novel query expansion technique that addresses the challenges of analyzing data from high-volume, high-velocity social media streams. We argue that effectively filtering a data stream in an environment in which language and terms can rapidly change does not necessarily rely only on information from the stream itself. Instead, we develop a query expansion technique that integrates words from the current stream with *external data sources* (in our experiments, newspaper archives) in order to predict the occurrence of relevant words that have not appeared in the stream yet. This algorithm enables the construction of new queries that effectively capture emergent events that a user may not have anticipated when initiating the data collection stream. The purpose of using the *external data sources* appears when we have a stream of data and we want to predict something that has not happened yet instead of using only the stream that is limited to the available information at a specific time. The external data source will enrich queries with words that cover future topics. For example, using the static word ”protest” to filter stream at a specific time x will fail to retrieve the related information of the word ”protest”. Generally, the “protest” results in violence, looting, and sometimes in curfew but these words or ”events” will appear in the future as a result of protest and do not appear in the stream yet. In order to retrieve the information that correlated with protest, we need to augment the query with information from an external source. In another example, COVID-19 has impacted the lives of millions of people across the globe in different ways. Travel restrictions and business closures are some of the events that happened in response to the COVID-19 pandemic. Predicting such events from a stream of data at a specific time using static query requires an external source with information related to similar event happened in the past to enrich the query with correlated information. For example, if we have an external source (news articles) related to the MERS coronavirus that has appeared in news since 2012, and we want to filter a stream (tweets about COVID-19) at a specific time x using the static query “coronavirus”. The MERS coronavirus appeared in news along with some business closures and travel restrictions. Relying only on the information from the stream for query expansion means that the static word “coronavirus”” might be unable to predict the events that may appear in the future at time $x + n$ as a result of the COVID-19 pandemic. Using the external source about the MERS coronavirus allows our system to search for old words that were related to “coronavirus” to augment our static query “coronavirus” with future events or topics such as travel restrictions and business closures. Our system should be able to predict the related information of the static word based on the available external source and augment the static query with dynamic words that may appear in the future and have a correlation with a static query in such a way.

We demonstrate the validity and effectiveness of our approach with an analysis of Twitter messages surrounding the 2015 Baltimore protests. Our data consists of more than 20.5 million tweets collected over two weeks. We use this data to evaluate the performance of our query expansion method against two alternative approaches. the first approach expands user queries with words extracted from a probabilistic topic model of the stream. The second approach reinforces them with emergent words extracted from the stream. We conduct several experiments in which we evaluate the performance of our query expansion along three metrics applied to the retrieved results:

volume (measured by tweet count), relevance (measured by hashtag count), and conciseness (measured through hashtags clustering). We find that our methods outperform alternative approaches, exhibiting particularly good results in identifying future emergent topics.

The remainder of this chapter is organized as follows: The system overview of our proactive query expansion method and three methods for expanding query in addition to the reference model are described in Section 4.2. Algorithm is described in Section 4.3. Evaluation and Results are provided in Section 4.5, and Section 4.6 respectively and section 4.6 concludes the chapter.

4.2 Proactive Query Expansion Method

System Overview

We introduce the proactive query expansion approach to detect emerging events in streaming data. Our approach utilizes external data sources to expand and enrich user proactive queries with words that do not appear in the stream yet but are highly correlated with emergent words in the existing stream. By adding these proactive words, our system can construct queries that capture emergent events that were not anticipated when initiating the data collection stream. In Fig. 4.1 we illustrate our proactive query expansion system. First, the initial data stream (in our application it is Twitter) is being monitored for emergent events to trigger query expansion process using the method introduced by [22] who use Dynamic Eigenvector Centrality to detect emergent words. If an emergent event is detected, a new set of queries is triggered in step 2 by collecting relevant words from the initial data stream using LDA. In step 3, we use DEC to identify the emergent words in the initial stream and combine these words with LDA words in step 4 to construct new queries. In step 5, we identify proactive words from an external data source. These words are determined by using two methods that in the past were correlated with words extracted from the initial data stream. In the final step (step 6), in addition to the LDA words and DEC words each query is expanded with proactive words from the external data source.

Query Expansion Methods

At the core of our system is the proactive query expansion method, which combines words retrieved from the primary stream with novel words extracted from the external source (archival

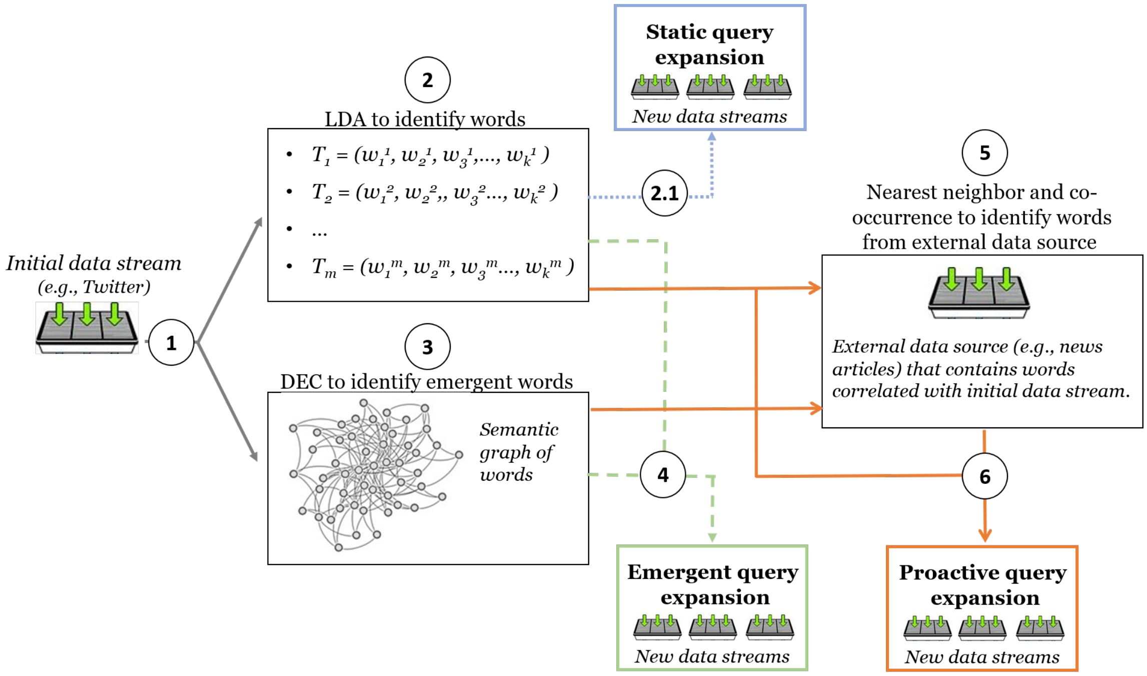

Static query expansion
New data streams
2
LDA to identify words
• $T_1 = (w_1^1, w_2^1, w_3^1,..., w_k^1)$
• $T_2 = (w_1^2, w_2^2,, w_3^2..., w_k^2)$
• ...
• $T_m = (w_1^m, w_2^m, w_3^m..., w_k^m)$
2.1
5
Nearest neighbor and co-occurrence to identify words from external data source
Initial data stream
(e.g., Twitter)
1
3
DEC to identify emergent words
Semantic graph of words
External data source (e.g., news articles) that contains words correlated with initial data stream.
4
6
Emergent query expansion
New data streams
Proactive query expansion
New data streams

Figure 4.1: Simplify illustration of the core logic of how queries constructed

Step 1. Detect emergent events in the initial stream using DEC [22] to trigger query expansion process.

Step 2. Use LDA to identify words in the initial stream.

Step 2.1 Use LDA words to construct **static** queries.

Step 3. Use DEC to identify emergent words in the initial stream.

Step 4. Combine LDA and DEC words to construct **emergent** queries.

Step 5. Use nearest neighbor and co-occurrence applied to external data source to identify words that are correlated with LDA/DEC words extracted from the initial stream.

Step 6. Combine LDA, DEC and words from external data source to construct **proactive** queries using (vector space or co-occurrence).

news in our application) with the goal to anticipate words that have not appeared yet. Our method belongs to the class of approaches that employ LDA to build new queries.

A large stream of literature uses LDA for query expansion [31, 32]. LDA estimates latent topics from a given corpus, where each topic represents a ranked list of the words included in the corpus. Overall, using LDA for query expansion means that each topic discovered by LDA serves as a new query restricted to the top-ranked words in each topic. words in LDA are ranked by their relative importance for each topic. However, when applied to dynamic data, this approach ignores emergent words which is a significant disadvantage. A key characteristic of high-volume, high-velocity streaming data, such as Twitter, is that topics can change rapidly. Relying on LDA alone for query expansion means that the extracted words might be unable to capture emergent topics.

Several studies propose different techniques to detect emergent events in streaming data [92, 93, 21, 22]. In our system, we use the Dynamic Eigenvector Centrality (DEC) method to detect emergent words [22] because of its ability to detect meaningful, less noisy, and more interpretable information in data streams than frequency-based measures used in [92, 93]. A natural improvement of LDA in the context of streaming data is therefore to combine words extracted with LDA with words identified as emergent based on the DEC method.

Combining LDA with DEC to construct new queries (LDA-DEC) overcomes the static nature of LDA and, we argue, is better suited to construct queries for streaming data where topics are rapidly changing. However, this method still only relies on words extracted from the current stream, which means that the resulting queries will potentially miss words that have not appeared in the stream yet. For this reason we propose the proactive query expansion which extends the LDA-DEC approach by adding words that, historically, were correlated with the words identified from the stream. We detect these correlated words through two different methods, both of them rely on the same external data source. In our first method, we construct a low-dimensional representation of the external data. We then use this vector space of proactive words to identify words that are close to those detected by the combined LDA-DEC method, using nearest neighbor search as proximity measure. In our second approach, we select proactive words based on their co-occurrence frequency with the LDA and DEC words.

The above discussion leads to three alternative query expansion methods in addition to the reference method for streaming data, which are summarized in Table 4.1. Method 1, Static, only uses the information from the current stream, which we include here as a benchmark case against which we will evaluate the other methods. Method 2, Emergent, combines the LDA words with the emergent words identified with DEC. Method 3, Proactive using vector space (VS), is our first proactive query expansion that adds words from the vector space constructed from the external data source. Method 4, Proactive using co-occurrence, is our second proactive query expansion that adds words from the external data source based on their co-occurrence frequency.

Table 4.1: Summary of Proposed Query Expansion Methods

	Method for word Identification			
	LDA	DEC	External Vector Space (VS)	External Co-Occurrence (CO)
Method 1 – Static	✓	-	-	-
Method 2 – Emergent	✓	✓	-	-
Method 3 – proactive VS	✓	✓	✓	-
Method 4 – proactive CO	✓	✓	-	✓

4.3 Algorithm

In this section we introduce the proactive query expansion and give a detailed description of each query expansion method. We also compare the proactive query expansion method to the reference method (labeled ”static”) and the emergent method (”emergent”). In all query expansion methods, we define the algorithms (Algorithms 1 to 4) with reference to using data from Twitter as an example of streaming data. Table 4.2 summarizes the notations that will appear in the algorithms.

We process a data stream by discretizing it into time intervals (called window), each of length w units (in our implementation we use minutes). We use the DEC metric introduced in [22] to extract the top-ranked emergent words. We trigger the query expansion if the set of emergent words in the current window indicate that an emergent event is occurring. To this end, we calculate the Jaccard similarity (J_s) of the top d DEC words between the current window and the P-previous windows. If the Jaccard similarity is less than or equal to some threshold th, we assume an emergent event is occurring that will require a new search query with new keywords not currently captured in the query that has initiated the current stream. At this point in the stream, we execute Steps 1–6 from Figure 4.1 to construct the following three queries:

Table 4.2: Table of Notation

S_1	:	Primary tweet stream
S_2	:	External data source
T	:	A set of topics result from LDA
J_s	:	Jaccard similarity
th	:	Specific threshold
w	:	Time interval (window)
n	:	Number of windows
d	:	Number of top-ranked DEC words
s	:	A tweet in the primary tweet stream
t	:	A given topic in T
m	:	Number of LDA topics
k	:	Number of top-ranked LDA words
l	:	Query length
Q	:	Query result for all topics
q_t	:	Query result associated with a given topic t
V	:	External semantic vector space
F	:	External bi-grams dictionary
D	:	DEC words for time interval w
i	:	Number of nearest neighbor words to return for a specific word
j	:	number of highest frequency words to return for a specific word

Static: Query expansion using LDA words. LDA is used to generate m topics from the current time windows. Each topic represents a ranked list of the words included in the current window which reveals a discussion theme for the topic. Using LDA for query expansion means using each topic estimated with LDA as a new query restricted to the top-ranked words in each topic. The algorithm procedure of this method is described in Algorithm 1. After generating a set of topics, we expand the query and return the results that satisfy each expanded query, such that for a given document s from the primary stream S_1 and a topic t from a set of topics T, if we can find any l LDA words in document s, then we add s into the query result (q_t). Finally, the aggregated query results Q for all topics will be returned by the end of the procedure.

Algorithm 1 Query Expansion Using LDA words

```
Inputs
    $Q$, Query result for all topics
    $T$, A set of topics result from LDA
    $S_1$, Primary tweet stream
end Inputs
procedure STATIC($Q$,$T$,$S_1$)
    $Q \leftarrow \emptyset$
    for $t$ in $T$ do                ▷ for each topic $t$ return the query result associated with it
        $q_t \leftarrow \emptyset$                ▷ initialize query result associated with topic $t$
        for $s$ in $S_1$ do    ▷ for each tweet $s$ in the primary stream $S_1$ check if the tweet has the LDA words
            $cnt$=0                ▷ count the number of LDA words in the tweets $s$
            for $v$ in $t$ do        ▷ for each word in the topic $t$ check if the word $v$ in the tweet $s$
                if $v$ in $s$ then
                    $cnt$+=1            ▷ increase the number of LDA words in tweets $s$
                    if $cnt \geq l$ then ▷ if $cnt \geq$ the length of the query $l$, add the tweet $s$ to the $q_t$
                        $q_t \leftarrow q_t \cup s$            ▷ add the tweet $s$ to the query result $q_t$
                    end if
                end if
            end for
        end for
        $Q \leftarrow Q \cup$                ▷ query results for all topics $q_t$
    end for
    return $Q$
end procedure
```

Emergent: Query expansion using LDA words and DEC words. We propose a query expansion method that combines words extracted with LDA with words identified as emergent based on the DEC method for a specific time window. Combining LDA with DEC to construct new queries overcomes the static nature of LDA and, we argue, is better suited to construct queries for streaming data where topics might rapidly change. For each topic t and a time window w, we add d

top-ranked DEC words from D that do not appear in the k top-ranked LDA words for topic t using function Dec(w,t,D,d) which saves the returned words in Dec_w. We used this condition to avoid redundant queries because we found that some top DEC words also appear in the top LDA words. By adding the DEC words to the topics, we will guarantee that the emerging topics are included in the query results. After adding the DEC words to each topic, we expand the query and return the results that satisfy each expanded query, such that for a given document s from the primary stream S_1, and a topic t from a set of topics T, if we can find any l LDA and DEC words in the document s, then we add document s into query result q_t. Finally, the aggregated query results Q for all topics will be returned by the end of the procedure.

Algorithm 2 Query Expansion Using LDA words and DEC words

```
Inputs
    $w$, Time interval
    $D$, DEC words for time interval $w$
    $d$, Number of top-ranked DEC words to return
end Inputs
procedure EMERGENT($Q$,$T$,$w$,$S_1$,$d$,$D$)
    $Q \leftarrow \emptyset$
    for $t$ in $T$ do            ▷ for each topic $t$ return the query result associated with it
        $q_t \leftarrow \emptyset$            ▷ initialize query result associated with topic $t$
        $DEC_w \leftarrow$DEC($w$,$t$,$d$ ,$D$)    ▷ return $d$ top-ranked DEC words that do not appear in the top LDA words of topic $t$ for time interval $w$
        $t \leftarrow t \cup DEC_w$            ▷ attach top $d$ DEC words to the topic $t$
        for $s$ in $S_1$ do   ▷ for each tweet $s$ in $S_1$ check if the tweet has the LDA and DEC words
            $cnt$=0            ▷ count the number of LDA and DEC words in the tweets $s$
            for $v$ in $t$ do
                if $v$ in $s$ then    ▷ for each word in the topic $t$ check if the word $v$ in the tweet $s$
                    $cnt$+=1            ▷ increase the number of LDA and DEC words in tweets $s$
                    if $cnt \geq l$ then   ▷ if $cnt \geq$ the length of the query $l$, add the tweet $s$ to the $q_t$
                        $q_t \leftarrow q_t \cup s$            ▷ add the tweet $s$ to the query result $q_t$
                    end if
                end if
            end for
        end for
        $Q \leftarrow Q \cup q_t$
    end for
    return $Q$
end procedure
```

Proactive Vector Space: Query expansion using LDA words, DEC words, and Vector Space. We extend the emergent query by using external data to add words that are correlated with the words identified from the stream but potentially haven't appeared in the initial stream yet. This method overcomes the limitation of the LDA-DEC method which only relies on words extracted from the current stream. Using this method allows us to capture future events or words that have not appeared in the stream yet. We used the fastText model [64] to generate a vector space V to find the words' nearest neighbor to each LDA and DEC word. The fastText model is utilized to construct an n-dimensional representation of each word in the external data called word embedding, each embedded word is represented as a vector of n dimension. After representing each word by a vector, we used the fastText nearest neighbor method to find the closest words in space to a given word. The nearest neighbor method allows us to capture the semantic information of a given word. To find nearest neighbor words for a target word, this method computes the cosine similarity between the target word and all words in the vocabulary using the vector representation of the words. As an example of this process, Table 4.3 presents the top 10 nearest neighbor words for two words: ”curfew” and ”looting”. These words represent key moments in the primary stream. As appear in the table, the top 3 nearest neighbor words for the word “curfew” are: “lockdown”, “nightfal”, and “impos” which means the static query “curfew” will be augmented by these words. For more investigation about the appearance of these words in the stream, we found that the word “curfew” appears in the stream at 2015-04-28 05:46:05, “@cbsbaltimore: A curfew is in place in #Baltimore overnight from 10 p.m. to 5 a.m. this week.”, the word lockdown appears after an hour and 15 minutes in a different time interval of the word “curfew” after two widows, which means our system can capture relevant words that have not appeared in the stream yet. The word “lockdown” appears at 2015-04-28 06:51:19, “Where the have the whole city on lockdown. No one let out and National guard or police on every block of the city”. The other word “nightfal” appears after 12 windows of the word “curfew” at 2015-04-28 08:59:38 , “Let's see what loot I get in this weeks nightfall! Post it if anything good!” and at 2015-04-28 16:40:08, “Hearing that #Baltimore police made 235 arrests last night as nightfall approaches.” Finally, the word “impos” is the stemming of the word imposed appear in the stream after one window of the word “curfew” at 2015-04-28 06:02:13, “Baltimore schools are closed today and a 10 p.m. to 5 a.m. curfew will be imposed tonight.” All these examples prove that our system can enrich the static query with words that capture future events. It is worth noticing that the word ”curfew” is also correlated with the word ”violence” as appears in the table, this word (”violence”) appears after 17 window from the word ”curfew” at 2015-04-28 10:09:42, ”You want to end the riots? Stop police violence! Stop bailing out man slaughtering officers! Stop police shooting people!.” The top 3 nearest neighbor words for the word “looting” show the same trend as the top 3 words nearest neighbor of the word “curfew”.

Algorithm 3 represents the process of the proposed method. The algorithm starts by adding the d top-ranked DEC words that do not appear in the top k LDA words to the topic t as explained in Algorithm 2. Then the function nearestNeighbors(V,w_t,i) is used to find the i nearest words closest to each word from topic t in the vector space V. The resulting words are then saved in a list called *nearest*. For each topic t we then attach the words that do not appear in topic t from the *nearest* list that is saved in W_v. After adding the nearest neighbors words W_v to topic t, we expand the query and return the results such that for a given document s from the primary stream S_1, if we can find any l LDA, DEC, and nearest neighbor words in the document s , then we add document s into query result q_t. Finally, the aggregated query results Q for all topics will be returned by the end of the procedure.

Table 4.3: 10-nearest neighbor words of word curfew and looting

Nearest Neighbor Words for "curfew"	Nearest Neighbor Words for "looting"
lockdown	vandal
nightfal	arson
impos	ransack
riot	quiktrip
loot	destruct
midnight	riot
polic	violenc
violence	protest
rioter	kill
protest	pillag

Proactive Co-occurrence Query expansion using LDA words, DEC words, and Co-occurrence frequency. This method is the same as the previous method except that the most relevant words in the external data is defined using the words' highest frequency. This method returns a set of words that have the highest number of occurrences for a certain LDA and DEC word. We built a dictionary F that consist of bi-grams (pair of adjacent words) from the external source, then we compute the frequency for all the bi-grams in the external data to return the j highest word frequency related to each LDA and DEC word. Table 4.4 presents the top 10 highest number of co-occurrence words for the two words "curfew" and "looting". As appear in the table, the top 3 highest number of co-occurrence words for the word "curfew" are: "militari", " impos", and "nationwid". As we mentioned before the word "curfew" appears in the stream at 2015-04-28 05:46:05. The word "militari" appears after 20 minutes in a different time interval of the word "curfew" after

Algorithm 3 Query expansion using LDA words, DEC words, and Vector Space

```
Inputs
    $Q$, Query result for all topics
    $S_1$, Primary tweet stream
    $T$, A set of topics result from LDA
    $w$, Time interval
    $D$, DEC words for time interval $w$
    $d$, Number of top-ranked DEC words to return
    $V$, External semantic vector space
    $i$, Number of nearest neighbor words to return
end Inputs
procedure PROACTIVE VECTOR SPACE($T$,$V$,$w$,$S_1$,$d$,$D$,$i$)
    $Q \leftarrow \emptyset$
    $W_v \leftarrow [\,]$
    for $t$ in $T$ do    ▷ for each topic $t$ return the query result associated with it
        $DEC_w \leftarrow DEC(w,t,d,\ D)$    ▷ return $d$ top-ranked DEC words that do not appear in the top LDA words of topic $t$ for time interval $w$
        $t \leftarrow t \cup DEC_w$    ▷ attach top $d$ DEC words to the topic $t$
        $qt \leftarrow \emptyset$    ▷ initialize query result associated with topic $t$
        for $w_t$ in $t$ do    ▷ for each word $w_t$ in $t$ return $i$ nearest neighbor words
            $nearest \leftarrow$ nearestNeighbors($V$,$w_t$,$i$)    ▷ return $i$ nearest neighbor words to the word $w_t$ that do not appear in the LDA and DEC words of topic $t$ for time interval $w$
            for $newWord$ in $nearest$ do
                if $newWord$ not in $t$ then    ▷ for each nearest neighbor word check if it are already in $t$
                    $W_v \leftarrow W_v \cup newWord$
                end if
            end for
        end for
        $t \leftarrow t \cup W_v$    ▷ attach nearest neighbor words to $t$
        for $s$ in $S_1$ do    ▷ for each tweet $s$ in $S_1$ check if the tweet has the LDA, DEC, and nearest neighbor words
            $cnt$=0 ▷ count the number of LDA,DEC, and nearest neighbor words in the tweets $s$
            for $v$ in $t$ do    ▷ for each word in the topic $t$ check if the word $v$ in the tweet $s$
                if $v$ in $s$ then
                    $cnt$+=1 ▷ increase the number of of LDA DEC and nearest neighbor words in tweets $s$
                    if $cnt \geq l$ then ▷ if $cnt \geq$ the length of the query $l$, add the tweet $s$ to the $q_t$
                        $q_t \leftarrow q_t \cup s$    ▷ add the tweet $s$ to the query result $q_t$
                    end if
                end if
            end for
        end for
        $Q \leftarrow Q \cup q_t$
    end for
    return $Q$
end procedure
```

one window. which proves that our system can capture relevant words that have not appeared in the stream yet using the words co-occurrence. The word “militari” appears at 2015-04-28 06:06:53, “Don’t forget. The heavily militarized police presence formed in response to PEACEFUL student protests. Not a violent one.” The other word “impos” is the same word that returned using the proactive vector space which means this word has the highest words co-occurrence and the highest cosine similarity to the word “curfew”. Finally, the word “nationwid” appears in the stream after one window of the word “curfew” at 2015-04-28 06:00:13, ”The police are ridiculous to EVERYBODY. Most people can agree with that. The police force needs to change nationwide.” The top 3 highest number of occurrences words for the word “looting” show the same trend as the highest words co-occurrence for the word “curfew”.

Algorithm 4 represents the process of the proposed method. The algorithm starts by adding the d top-ranked DEC words that do not appear in the top k LDA words to the topic t as explained in Algorithm 2. The Algorithm then identifies the j highest frequency words for each word in topic t. The Function highestFreq(F,w_t,j) is then used to attach the highest frequency words W_f to each topic t, with the resulting words saved in a list called $freq$. For each topic t, we then attach the words that do not appear in topic t from $freq$ list that is saved in W_v. After adding the highest frequency words W_v to topic t, we expand the query and return the results such that for a given document s from the primary stream S_1, if we can find any l LDA, DEC, and highest frequency words in the document s, then we add document s into query result q_t.Finally, the aggregated query results Q for all topics are returned by the end of the procedure.

Table 4.4: 10 Highest number of occurrences words of the two words ”curfew” and ”looting”

Highest Word Frequency for ”curfew”	Highest Word Frequency for ”looting”
militari	secur
impos	destruct
nationwid	extens
overnight	sporad
hour	systemat
violat	store
began	riot
mandatori	violenc
citywid	widespread
citi	vandal

Algorithm 4 Query expansion using LDA words, DEC words, and Co-occurrence frequency

```
Inputs
    $Q$, Query result for all topics
    $S_1$, Primary tweet stream
    $T$, A set of topics result from LDA
    $w$, Time interval
    $D$, DEC words for time interval $w$
    $d$, Number of top-ranked DEC words to return
    $F$, External bi-grams dictionary
    $j$, Number of nearest neighbor words to return for a specific word
end Inputs
procedure PROACTIVE CO-OCCURRENCE($T$,$F$,$w$,$S_1$,$d$,$D$,$j$)
    $Q \leftarrow \emptyset$
    $W_f \leftarrow [\,]$
    for $t$ in $T$ do    ▷ for each topic $t$ return the query result associated with it
        $DEC_w \leftarrow DEC(w,t,d,\ D)$    ▷ return $d$ top-ranked DEC words that do not appear in the top LDA words of topic $t$ for time interval $w$
        $t \leftarrow t \cup DEC_w$    ▷ attach top $d$ DEC words to the topic $t$
        $qt \leftarrow \emptyset$    ▷ initialize query result associated with topic $t$
        for $w_t$ in $t$ do    ▷ for each word $w_t$ in $t$ return $j$ highest frequency words
            freq $\leftarrow$ highestFreq($F$,$w_t$,$j$)    ▷ return $j$ highest frequency words to the word $w_t$ that do not appear in the LDA and DEC words of topic $t$ for time interval $w$
            for $newWord$ in freq do
                if $newWord$ not in $t$ then    ▷ for each highest frequency word check if it are already in $t$
                    $W_v \leftarrow W_v \cup newWord$
                end if
            end for
        end for
        $t \leftarrow t \cup W_v$    ▷ attach highest frequency words to $t$
        for $s$ in $S_1$ do    ▷ for each tweet $s$ in $S_1$ check if the tweet has the LDA, DEC, and highest frequency words
            $cnt$=0 ▷ count the number of LDA, DEC, and highest frequency words in the tweets $s$
            for $v$ in $t$ do    ▷ for each word in the topic $t$ check if the word $v$ in the tweet $s$
                if $v$ in $s$ then
                    $cnt$+=1 ▷ increase the number of of LDA DEC and nearest neighbor words in tweets $s$
                    if $cnt \geq l$ then ▷ if $cnt \geq$ the length of the query $l$, add the tweet $s$ to the $q_t$
                        $q_t \leftarrow q_t \cup s$    ▷  add the tweet $s$ to the query result $q_t$
                    end if
                end if
            end for
        end for
        $Q \leftarrow Q \cup q_t$
    end for
    return $Q$
end procedure
```

4.4 Evaluation

In this section, we will give an overview of the evaluation of our query expansion methods. First, we describe the data that we use to detect the emergent topics and expand the query. Second, we give a detailed description of the evaluation of our query expansion methods.

Data Description

We use Twitter data collected for one specific public safety event: the 2015 Baltimore protests in response to the death of a Baltimore resident Freddie Gray. The death of Gray caused a series of protests and violence, which led to a whole city curfew on the evening of April 28th. We purchased archived tweets from Gnip, a company that provides access to the full archive of public Twitter data. We used broad search words to collect tweets in order to create a noisy data stream that covers tweets related to the Baltimore events as well as unrelated events.Our data set was collected with the following search words: joseph kent, freddie gray, eric garner, ferguson, curfew, police, riot, protests, loot, looting, #purge, #baltimore, #baltimoreriots, #baltimoreuprising, #freddiegray, #josephkent, #blacklivesmatter, #onebaltimore, rioter, charge, charged, murder, homicide, mosby, corporal, #mayday, justice, #blackspring, #freddiegray's, cops, unjustified, spinal, broken spine, arrested, thugs, thug, #marilynmosby, #wakeupamerica, freddie, racist, racism, #baltimoreprotest, propaganda, officers, knife. A logical OR expression was used to filter the words, i.e., for example, by keeping the term Baltimore, we obtained all tweets related to the city, and not necessary to the event [22]. The Tweet data is preprocessed before applying any query expansion methods; we removed the stop words, URLs, numbers, and all non-English tweets. Our data set comprises 20.5 million tweets covering fifteen days from April 17th to May 3rd, 2015. Because of its noisy nature, the stream is ideal to evaluate our method's ability to detect emergent topics and expand the query.

For our external source, we decided to use an archival news article published one year before the Baltimore events in order to predict the occurrence of relevant words that have not appeared in the stream yet. Using an archival external source allows our system to search for old words that related to a static query to augment it with correlated information to predict future events or topics. We chose New York Times (NYT) and CNN as our source of external data because these sources have a public API that can be used to crawl the archived news articles. We obtained 30,456 articles from NYT and 14,145 from CNN.

Evaluation

How good is our proposed query expansion method in expanding queries and predicting future topics? In this section, we answer this question by conducting two types of experiments to evaluate our methods using Twitter data from the 2015 Baltimore protests. For both experiments, we simulate real-time stream processing by constructing a primary stream from the full data. This primary stream consists of all tweets that contain the word "police", a total of 5.1 million tweets. We divide this primary stream into 15-minute time intervals, which we call "windows". On each window, we use the DEC metric to determine during which windows an emergent event occurred. More precisely, we calculate the Jaccard similarity (J_s) between the top 200 DEC words between the current window and the 3-previous windows. If this similarity is less than or equal to 15%, we assume that an emergent event has occurred. We chose low Jaccard similarity because it indicates we have found relatively unique set of emergent words and these words have not appeared in the stream yet. Using this metric, the primary stream resulted in 373 windows with emergent events out of the 1573 windows. Then LDA was used to extract a set of 5 topics in the targeted time window (intervals have emergent events). Then, we use the top 20 words from each topic to form the top-ranked words.

In order to relate our results to reality, we identified three key events from timelines published by news outlets [94] to pick some time intervals to use in our experiments. Therefore, in addition to the first-time interval triggered by our algorithm (time interval 16), we used the time intervals 155, 781, and 1065 based on the events that happened in Baltimore. Time interval 155 at 7:00 am, April 19, captures the tweets about the death of Freddie Gray. Time interval 781 on April 25, has the tweets about looting, violence, and protest. Time interval 1065 at 10:00 pm on April 28, has the Baltimore curfew tweets. Each experiment is applied at these time intervals. The two experiments are explained in the following subsections:

Experiment 1: Quantity and Quality of Retrieved Data

Experiment 1 compares the performance of our proactive query expansion methods proactive VS and proactive CO with respect to the reference methods static and emergent using streaming data quality indicators for a certain time interval. Each method returns a set of tweets called query result (Q). The following quality indicators metrics are:

Volume (measured by tweet count): This metric finds the total number of tweets matching a specific query condition from a certain time interval to the end of the primary stream.

Relevance (measured by hashtag count): This metric finds the total number of hashtags in the tweets matching a specific query condition from a certain time interval to the end of the primary stream.

Conciseness (measured through hashtags clustering): This metric clusters the tweets matching a specific query condition from a certain time interval to the end of the stream. In order to determine how concise the stream is, we use the hashtags to check if the query results return similar hashtags based on the number of clusters needs to cluster them. We find the number of clusters using k-means [95]. K-means is used to cluster the query results for each method such that: the data points are the set of tweets, and the features of the cluster are the hashtags. The lower the optimal number of clusters, the more concise and specific the stream is. To find the optimal number of clusters (k), we used the elbow method [96, 97]. Where the optimal number of clusters is represented in a graph as is an inflection point (elbow) using the average distortion score. The distortion score is the sum of squared differences of each point to its assigned center. In this experiment, the distortion score is computed from k = 2 to k = 15 clusters. Fig. 4.2 shows the graphical representation of the elbow test, the inflection point or the elbow in this test is 8 which is the optimal number of clusters.

The higher the tweets count and hashtag count, the better and more relevant result is and the lower number of clusters, the more concise and specific the result is.

Experiment 2: Predictive Power of Retrieved Data

Experiment 2 tests the effectiveness of our proposed methods to predict future emerging events or to predict the conversations from previous time intervals using a different set of hashtags. For example, we know that in time interval $x + n$ there was a curfew. Are our methods able to predict the curfew from time interval x which is related to protests? Will the protest lead to curfew? We picked and used a different set of hashtags matching the events that happened in Baltimore to determine if our methods can predict them from previous time intervals. The question here is which hashtag to use and how to validate? We used two categories of hashtags, the highest frequency hashtag, and the lowest frequency hashtags. In this experiment, we used a different set of hashtags that are related to some events that happened in Baltimore in response to the death of Freddie Gray. And for the validation, we used precision as an indicator of the effectiveness of our methods in

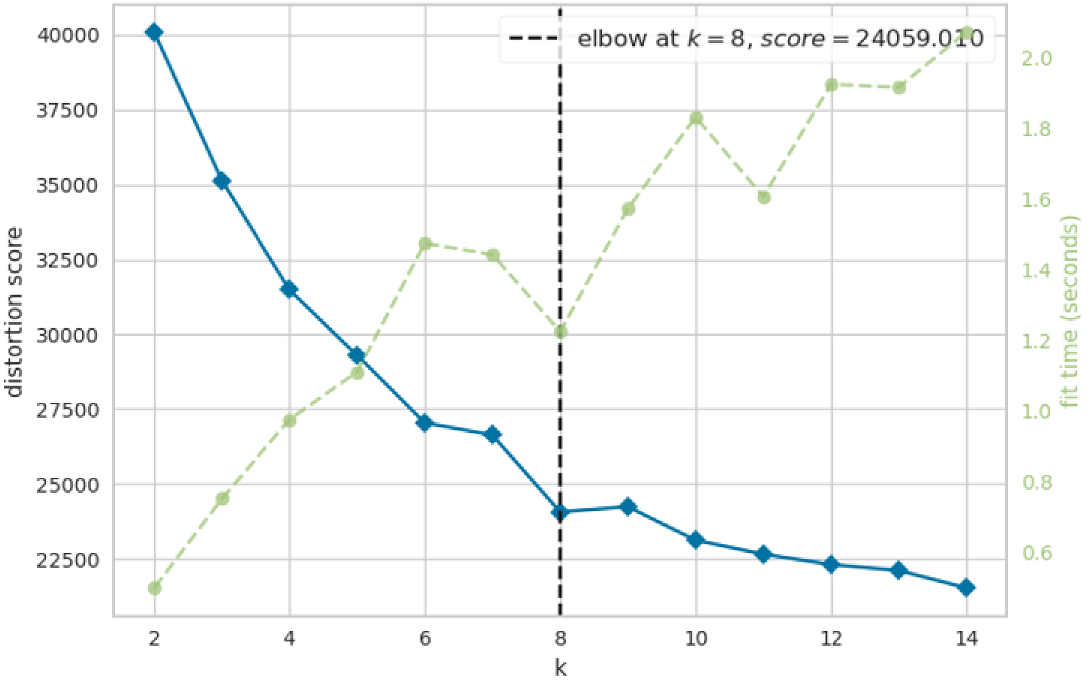


Figure 4.2: Elbow Method

predicting events. The precision is defined as the count of a certain hashtag from the query results divided by the total number of the same hashtag from the start of a certain time interval to the end of the stream.

4.5 Results

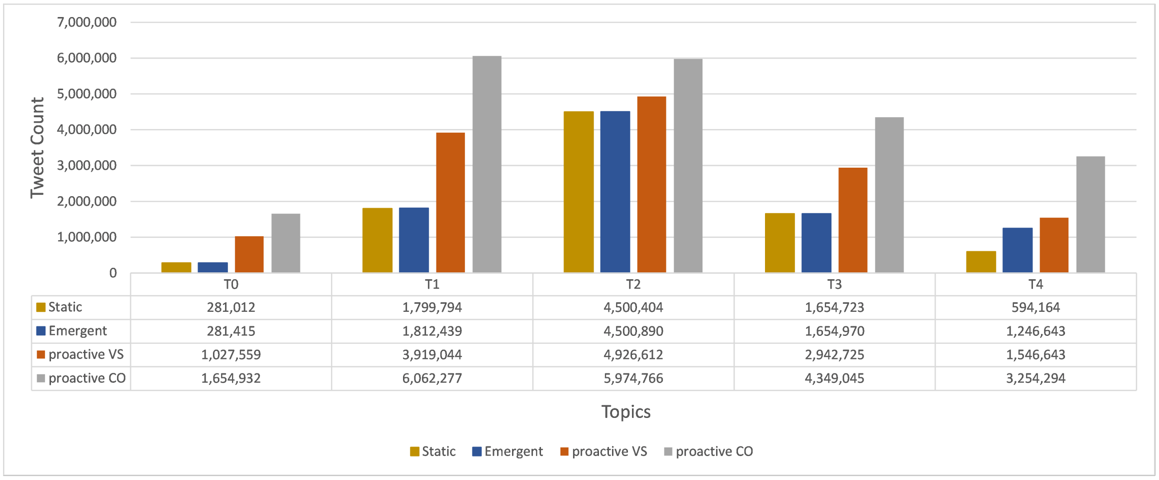
This section compare the results of the four methods based on volume, relevance, and conciseness which we defined above.

Experiment 1

In this experiment, we compared the performance of proactive VS and proactive CO with respect to the reference methods static and emergent using streaming data quality indicators; tweet count, hashtag count, and hashtag clustering, for different time intervals (16, 155, 781, and 1065).

In terms of volume (i.e, tweet count), figures 4.3, 4.4, 4.5, and 4.6 show the number of tweets for the query results per five topics using the four methods at time intervals 16, 155, 781, and 1065, respectively. For all time intervals, we found that proactive VS and proactive CO significantly outperform emergent and static methods. In other words, proactive VS and proactive CO can return more tweets than the others per each topic, which means adding new words from external sources covers more data. For instance, at time interval 16 under the topic (T0), proactive VS and proactive CO returned 3.65 and 5.88 times more tweets than static, respectively. Also, proactive CO returns more tweets than any other method which proves its efficiency.

In Figure. 4.4, it can be seen that topics two and four covers more tweets than other topics per all methods at time interval 155. It indicates that there is an important event that happened at this time interval. To further investigate why these two topics have the highest number of tweets, we looked at the LDA words for these topics (T2, and T4). As a result, the LDA words included: Freddie Gray, died, killed, black, arrested, beaten, and officer. These words were the majority of the tweets in our stream at that time.

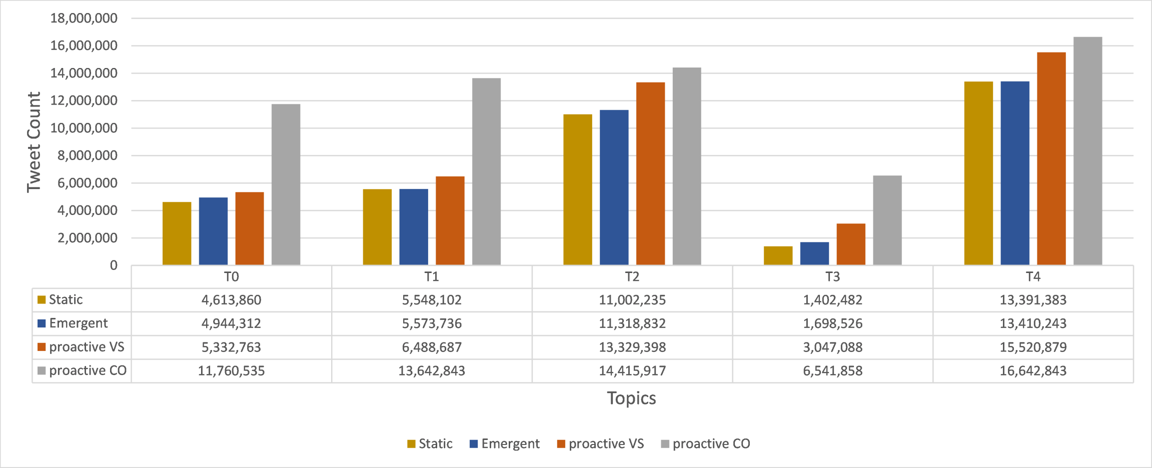


	T0	T1	T2	T3	T4
Static	281,012	1,799,794	4,500,404	1,654,723	594,164
Emergent	281,415	1,812,439	4,500,890	1,654,970	1,246,643
proactive VS	1,027,559	3,919,044	4,926,612	2,942,725	1,546,643
proactive CO	1,654,932	6,062,277	5,974,766	4,349,045	3,254,294

Figure 4.3: Tweet Count Over Time interval 16 of Query Results

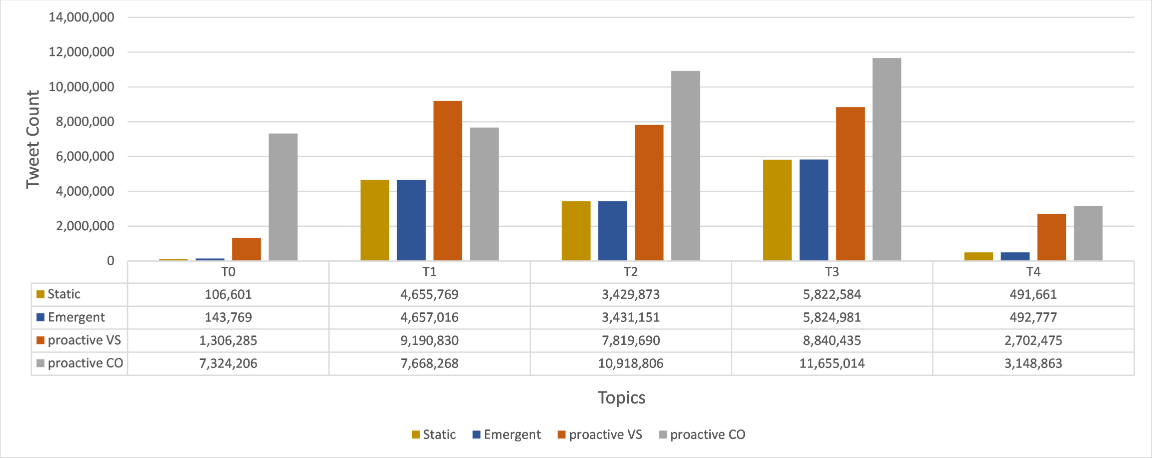
In terms of relevance (i.e, hashtag count), figures 4.7, 4.8, 4.9, and 4.10 show the number of hashtags for the query result associated with each of the five topics and generated using the four methods for all time intervals. As we expected, a higher number of hashtags is associated with proactive VS and proactive CO comparing with static and emergent methods.

In term of conciseness (i.e, quality of hashtag clustering), figures4.11, 4.12, 4.13, 4.14 show the optimal number of clusters k for the query result using the k-means elbow method per each of the five topics per each of the four methods for all time intervals. For all topics and all-time intervals, proactive VS and proactive CO return more concise tweets, despite the fact they return more tweets than other methods.



	T0	T1	T2	T3	T4
■ Static	4,613,860	5,548,102	11,002,235	1,402,482	13,391,383
■ Emergent	4,944,312	5,573,736	11,318,832	1,698,526	13,410,243
■ proactive VS	5,332,763	6,488,687	13,329,398	3,047,088	15,520,879
■ proactive CO	11,760,535	13,642,843	14,415,917	6,541,858	16,642,843

Figure 4.4: Tweet Count Over Time interval 155 of Query Results



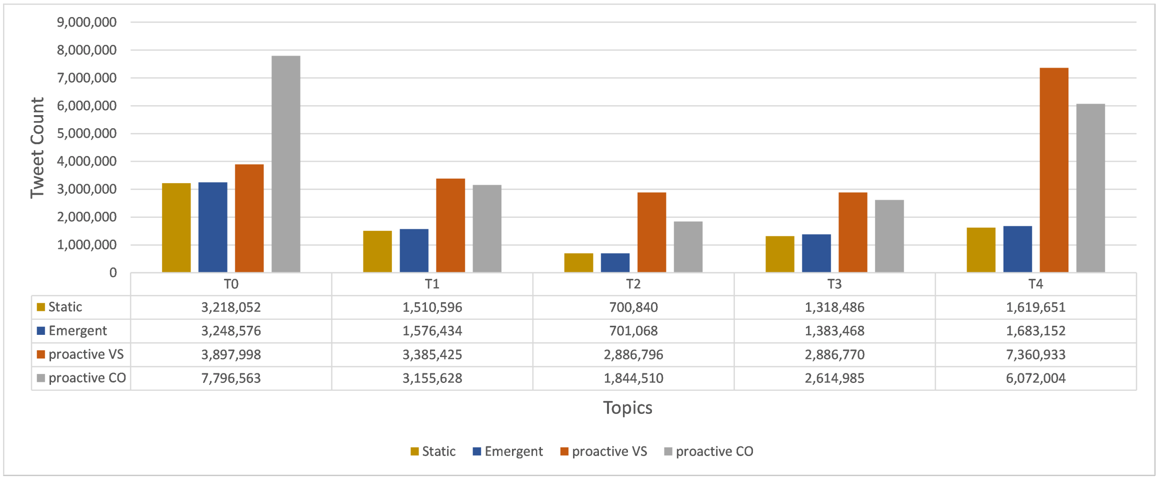
	T0	T1	T2	T3	T4
■ Static	106,601	4,655,769	3,429,873	5,822,584	491,661
■ Emergent	143,769	4,657,016	3,431,151	5,824,981	492,777
■ proactive VS	1,306,285	9,190,830	7,819,690	8,840,435	2,702,475
■ proactive CO	7,324,206	7,668,268	10,918,806	11,655,014	3,148,863

Figure 4.5: Tweet Count Over Time interval 781 of Query Results

For example, In Figure.4.11 for topic 0 at time interval 16, we can cluster the tweets using 6 and 5 clusters using proactive VS and proactive CO respectively, while we need 10 and 8 clusters to cluster them using static and emergent respectively.

Experiment 2

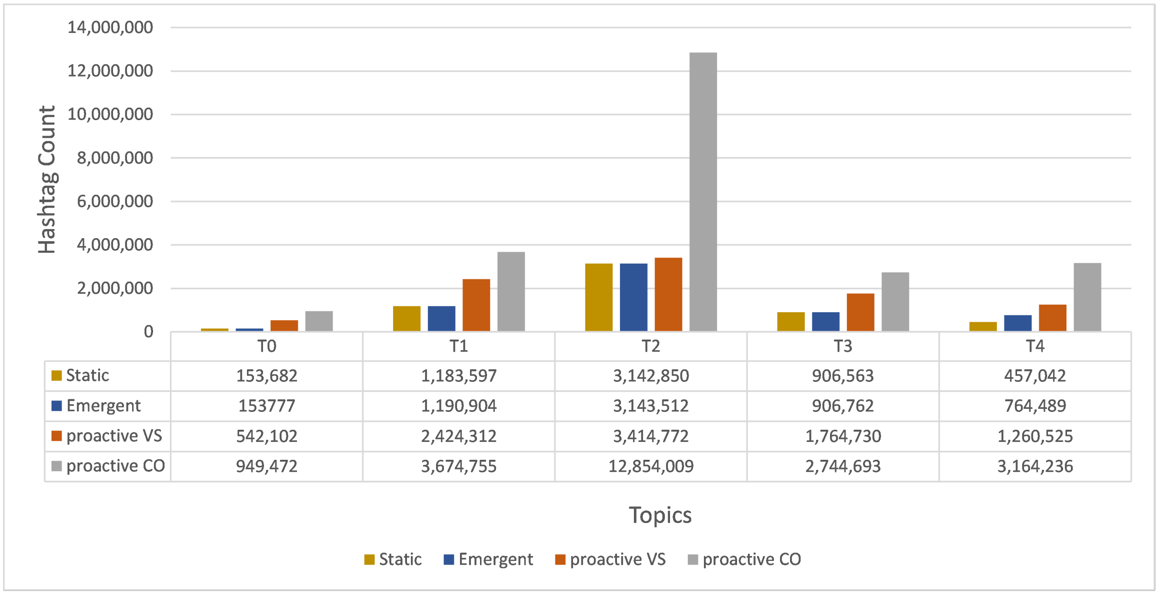
In this experiment we test the effectiveness of our proposed methods to predict future emerging events or to predict the conversations from previous time intervals using the precision of a different set of hashtags. For example, if we are interested in predicting the events that might happen as a result of the death of Frediy Gray. Let us assume that we use the time interval x which



	T0	T1	T2	T3	T4
Static	3,218,052	1,510,596	700,840	1,318,486	1,619,651
Emergent	3,248,576	1,576,434	701,068	1,383,468	1,683,152
proactive VS	3,897,998	3,385,425	2,886,796	2,886,770	7,360,933
proactive CO	7,796,563	3,155,628	1,844,510	2,614,985	6,072,004

Figure 4.6: Tweet Count Over Time interval 1065 of Query Results

has the tweets about this event (death of Frediy Gray) to return the query results (set of tweets) that satisfies the expanded query for that interval, and we will use hashtags from time interval $x+n$ that has tweets about the events happened based on his death. To predict the events that happened in time interval $x+n$ from time interval x, we picked a random set of hashtags consists of the highest and lowest frequency hashtags from time interval $x+n$ such as '#protest' to check if the query results from time interval x contain these hashtags using the precision. We define precision as a fraction of the relevant hashtags. For example, the precision to predict #protest that appears in time interval $x+n$ from time interval x is the count of the hashtag "#protest" in the query results at time interval x divided by the count of the hashtag "#protest" from the start of the time interval x to the end of primary stream. If the precision is near or equal to one indicates the effectiveness of the method to predict future events as shown in Figures 4.15 and 4.16 represents the results of the four query expansion methods in predicting the events that happened in time interval 781 from time interval 155 using highest and lowest hashtags. It is noticed that the precision of our proactive VS and proactive CO is higher than other methods static and emergent for all the topics. For example, as shown in Figure.4.15 (d), proactive CO predicts the topics about the future event "protest" that appears in topics 1, 2, and 4 which indicates its effectiveness. Additionally, we can say proactive VS and proactive CO can target all the events that appear in streaming data even if it has a few hashtags that describe it. For instance, hashtag "#health" appears 12 times compared to hashtag "#baltimore" that appears 1207 in the time interval 781, however, our methods were able to capture the topics that have these events as appear in 4.16 (a).



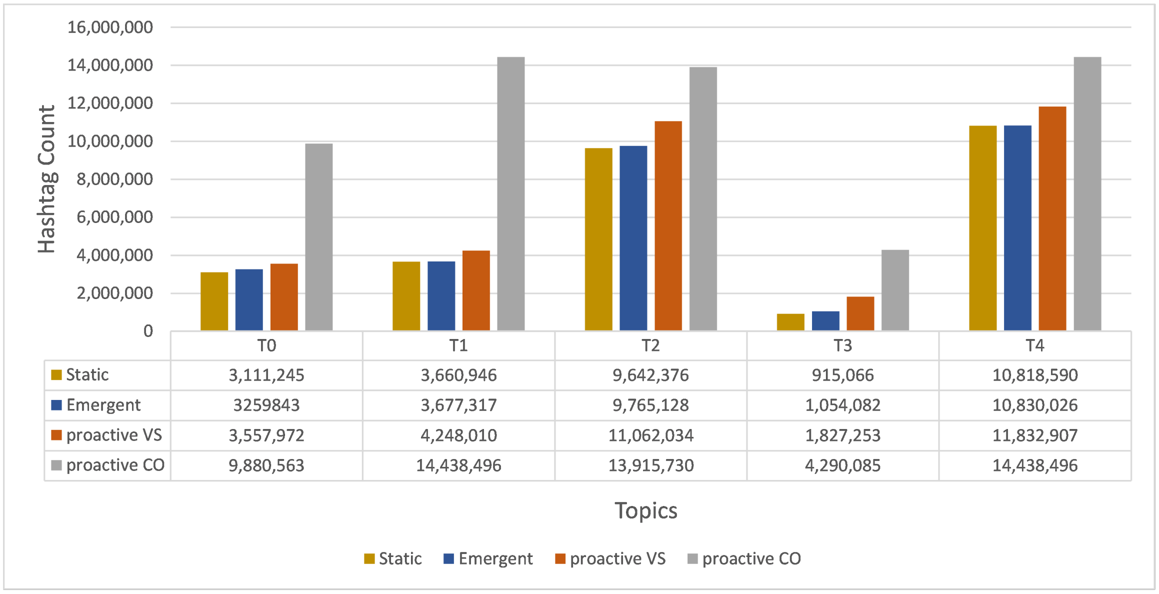
	T0	T1	T2	T3	T4
Static	153,682	1,183,597	3,142,850	906,563	457,042
Emergent	153777	1,190,904	3,143,512	906,762	764,489
proactive VS	542,102	2,424,312	3,414,772	1,764,730	1,260,525
proactive CO	949,472	3,674,755	12,854,009	2,744,693	3,164,236

Figure 4.7: Hashtag Count Over Time interval 16 of Query Results

Figures 4.17 and 4.18 present the precision for predicting the events that happened in the time interval 1065 from the time interval 781 using the highest and lowest frequency hashtags. We find that our proactive query expansion methods have the same trend as those observed in the two Figures 4.17 and 4.18.

4.6 Conclusion

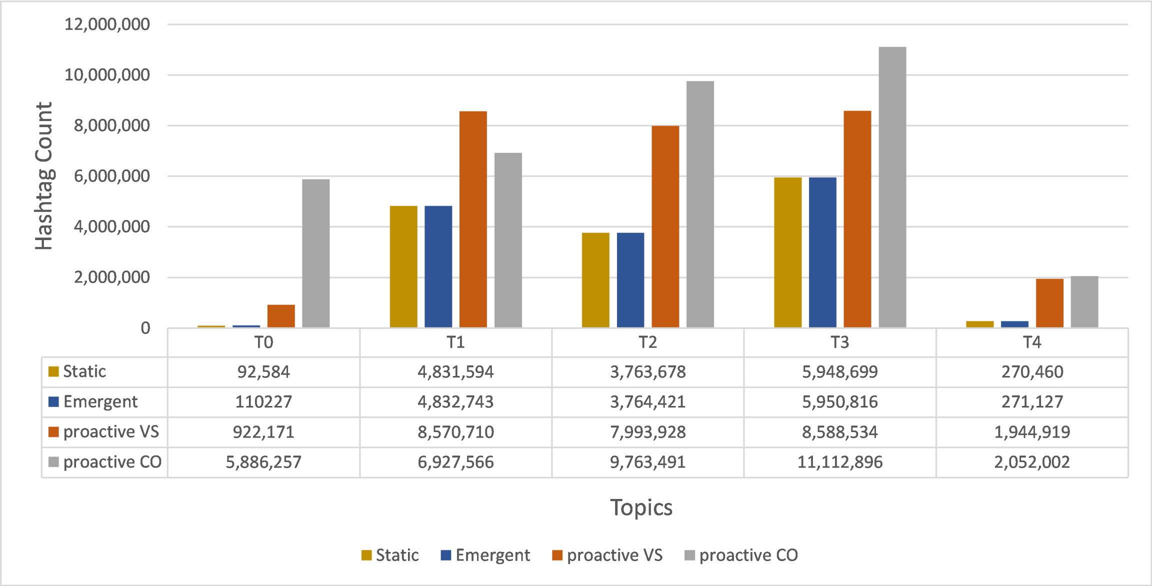
This study proposed a novel approach for expanding queries called the proactive query expansion method. This new method depends on adding novel words to the search query from an external data source, where the words are chosen using either nearest neighbor words or highest frequency words to each word that appears in the five LDA topics and DEC words. Two major experiments were performed: (1) we compared the performance of our proposed query expansion methods: Proactive VS and Proactive CO (query expansion using LDA, DEC, and external data) with the reference methods (Static: query expansion using LDA) and Emergent (query expansion using LDA and DEC). The performance of the proposed approach is quantified by quality indicators of the streaming data which are the tweet count, hashtag count, and hashtag clustering. (2) We tested the effectiveness of our proposed methods to predict future emerging events or to predict the conversations from previous time intervals using a different set of hashtags. Our experiment



	T0	T1	T2	T3	T4
🟨 Static	3,111,245	3,660,946	9,642,376	915,066	10,818,590
🟦 Emergent	3259843	3,677,317	9,765,128	1,054,082	10,830,026
🟧 proactive VS	3,557,972	4,248,010	11,062,034	1,827,253	11,832,907
⬜ proactive CO	9,880,563	14,438,496	13,915,730	4,290,085	14,438,496

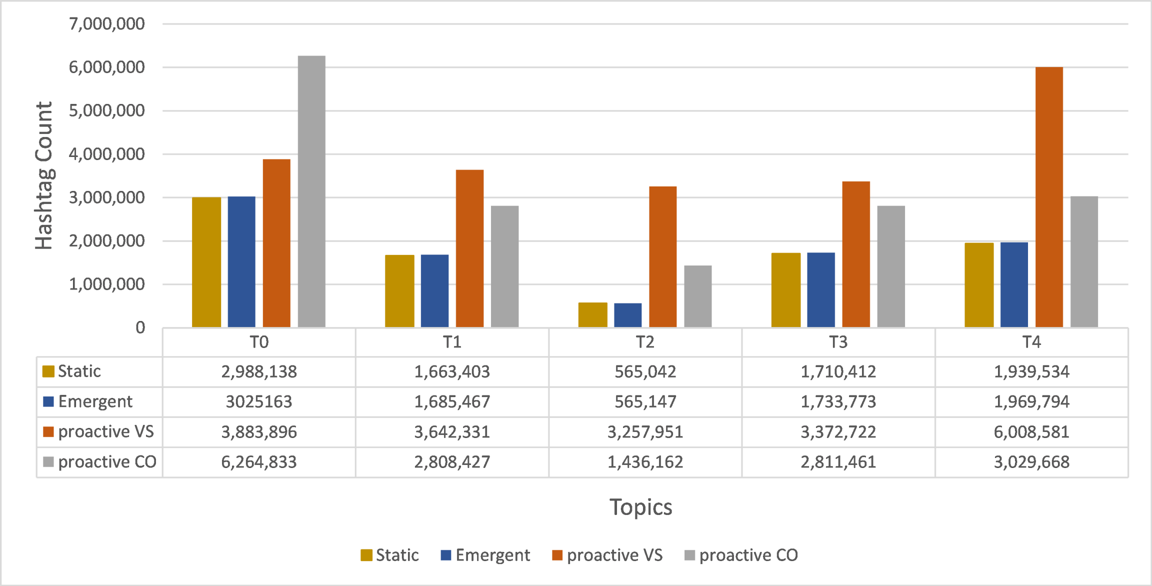
Figure 4.8: Hashtag Count Over Time interval 155 of Query Results

performed on 20.5 million tweets (primary stream) covers fifteen days from April 17–May 3, 2015. For our external source, the external data (secondary stream), we collected news from CNN and New York Times which covers one year before the event happened (2014). Generally, the proactive query expansion methods (Proactive VS and Proactive CO) improve the performance of the information retrieval and achieve higher performance compared with Static and Emergent. Additionally, the experiments indicate that our approach can enhance the quality of the results for all the topics. Besides, our proposed methods are more concise comparing to Static and Emergent. Finally, the proposed methods play a key role in enhancing the performance of the search query, which means providing the user with more relative and concise results of interest.



	T0	T1	T2	T3	T4
■ Static	92,584	4,831,594	3,763,678	5,948,699	270,460
■ Emergent	110227	4,832,743	3,764,421	5,950,816	271,127
■ proactive VS	922,171	8,570,710	7,993,928	8,588,534	1,944,919
■ proactive CO	5,886,257	6,927,566	9,763,491	11,112,896	2,052,002

Figure 4.9: Hashtag Count Over Time interval 781 of Query Results



	T0	T1	T2	T3	T4
■ Static	2,988,138	1,663,403	565,042	1,710,412	1,939,534
■ Emergent	3025163	1,685,467	565,147	1,733,773	1,969,794
■ proactive VS	3,883,896	3,642,331	3,257,951	3,372,722	6,008,581
■ proactive CO	6,264,833	2,808,427	1,436,162	2,811,461	3,029,668

Figure 4.10: Hashtag Count Over Time interval 1065 of Query Results

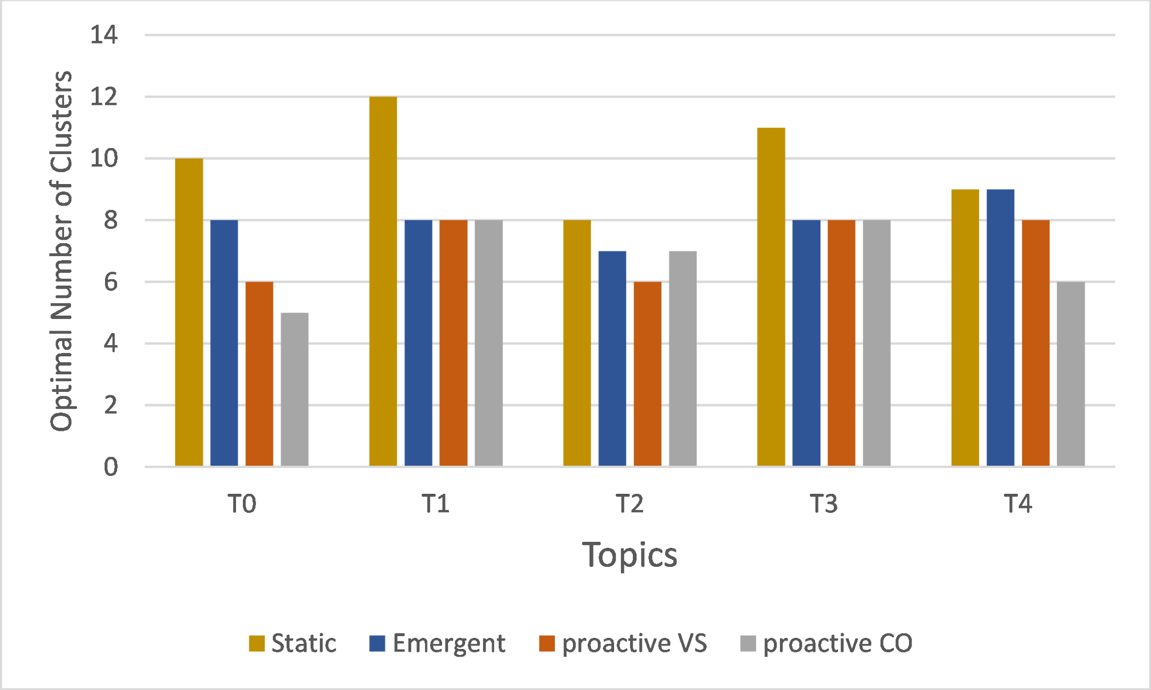


Figure 4.11: Optimal Number of Clusters Over Time interval 16 of Query Results

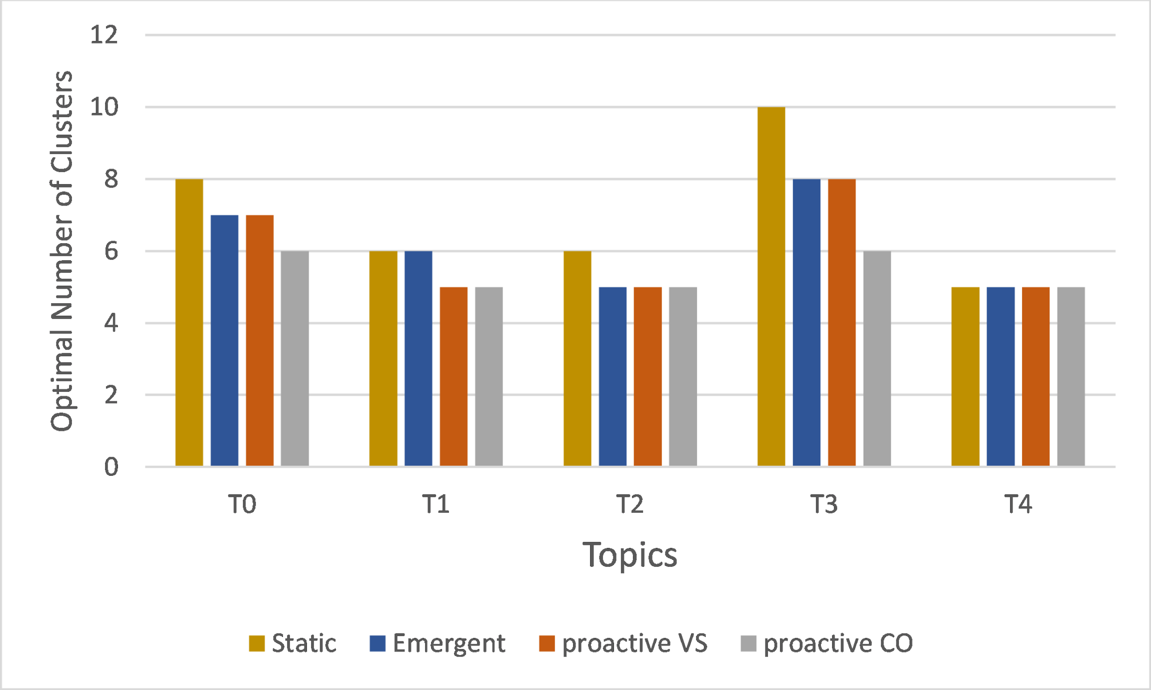


Figure 4.12: Optimal Number of Clusters Over Time interval 155 of Query Results

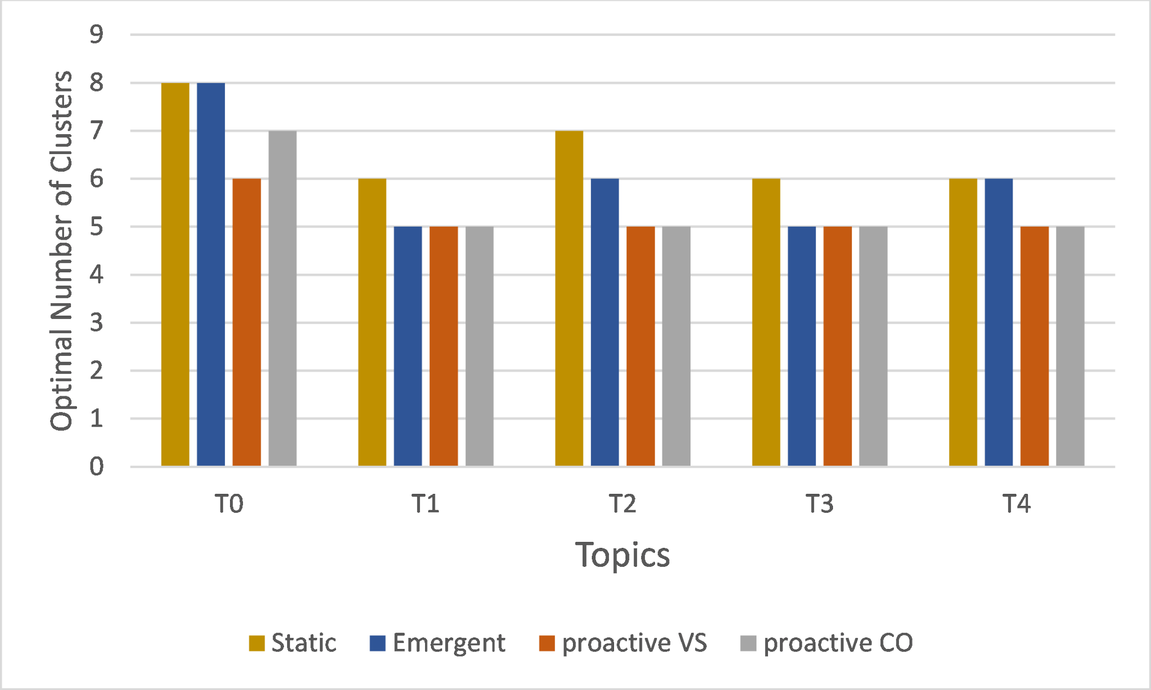


Figure 4.13: Optimal Number of Clusters Over Time interval 781 of Query Results

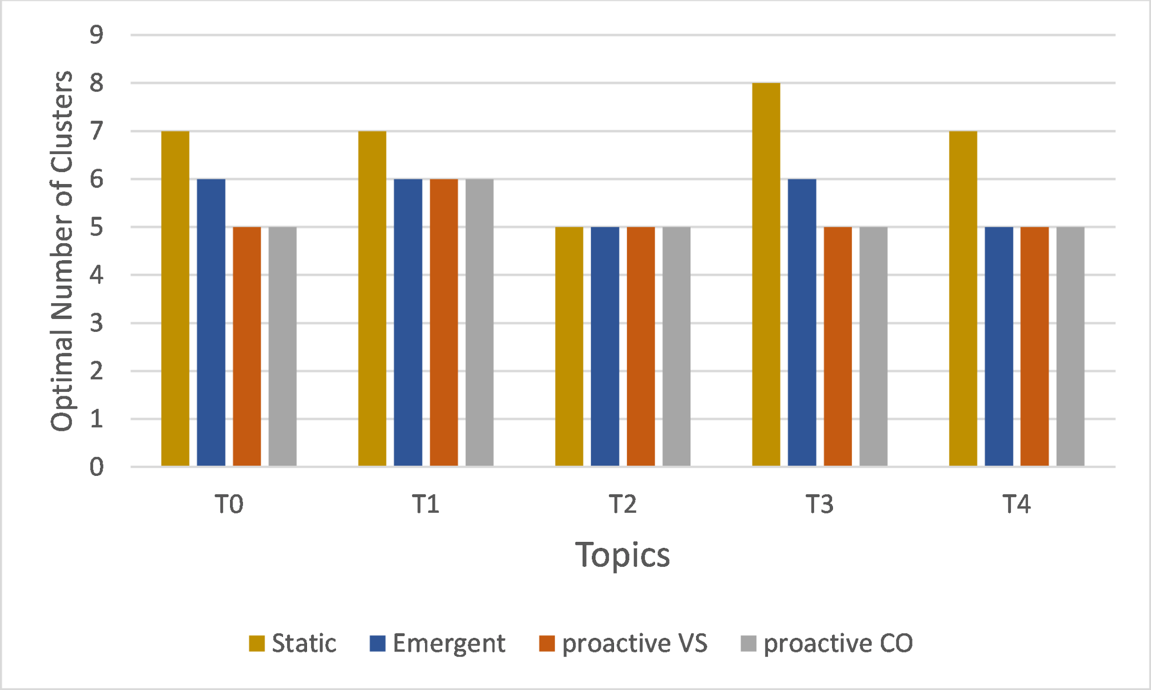
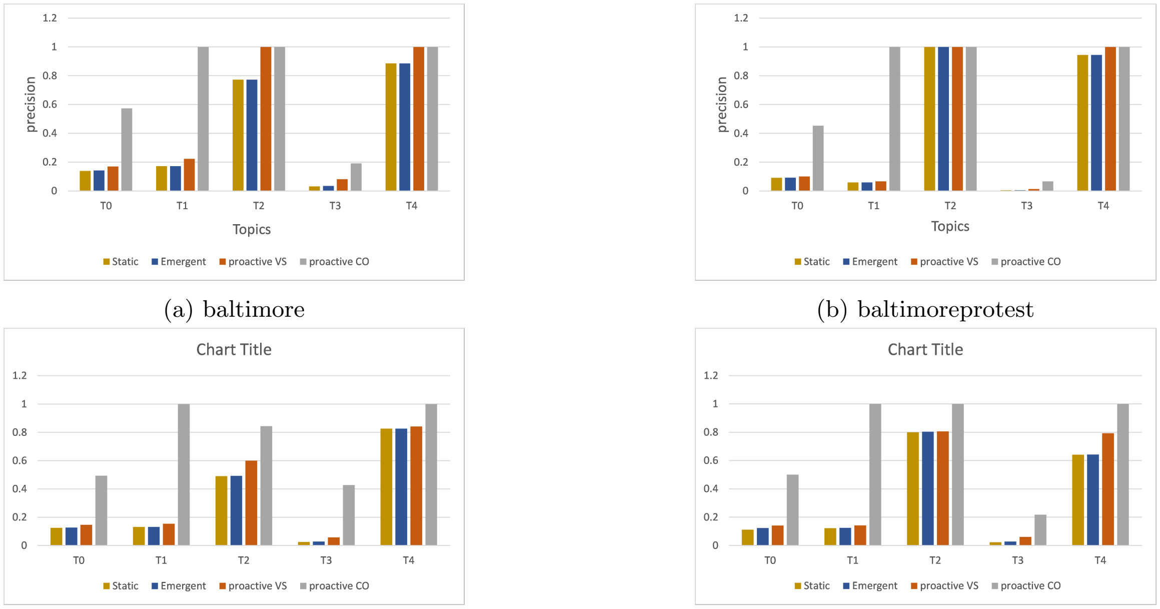


Figure 4.14: Optimal Number of Clusters Over Time interval 1065 of Query Results



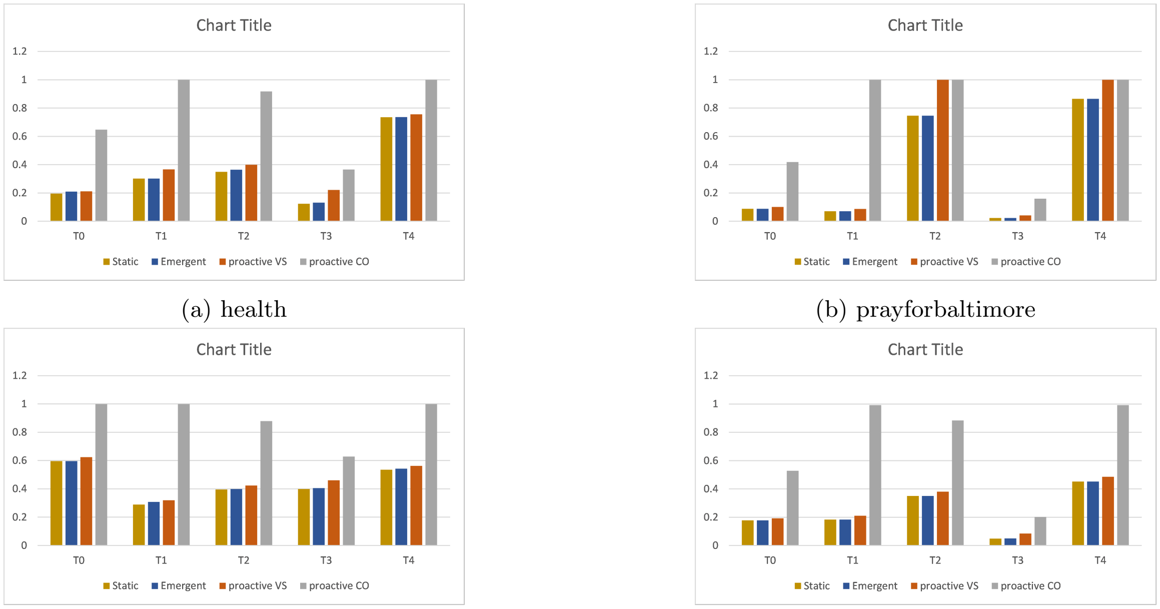
(a) baltimore

(b) baltimoreprotest

(c) blacklivesmatter

(d) protest

Figure 4.15: Prediction the events that happened in time interval 781 from time interval 155 using the highest frequency hashtags



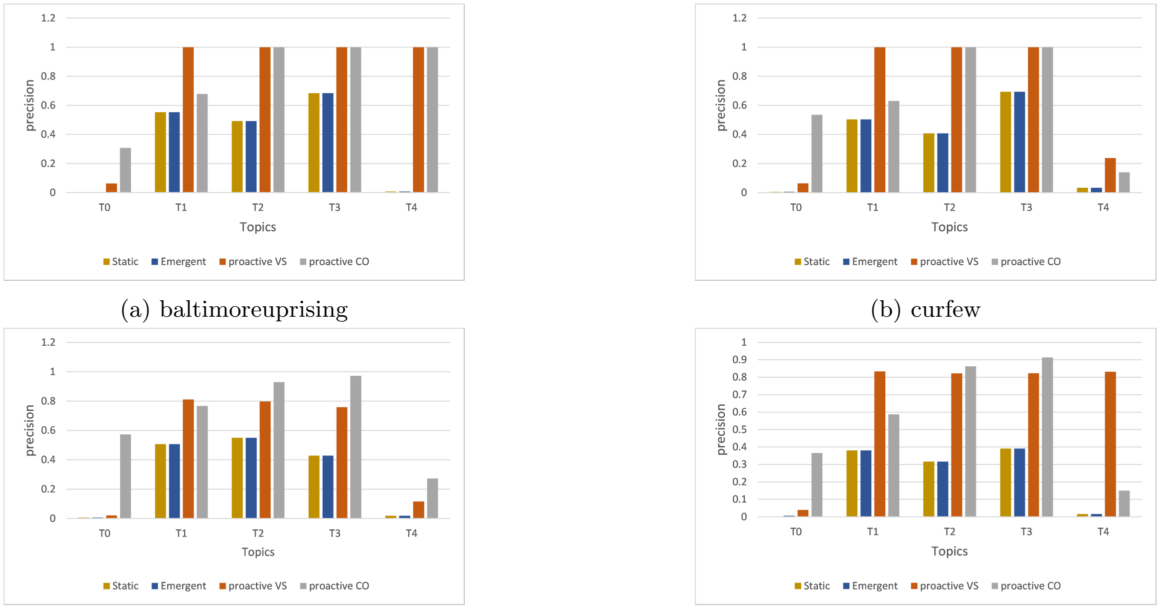
(a) health

(b) prayforbaltimore

(c) murder

(d) racism

Figure 4.16: Prediction the events that happened in time interval 781 from time interval 155 using the lowest frequency hashtags



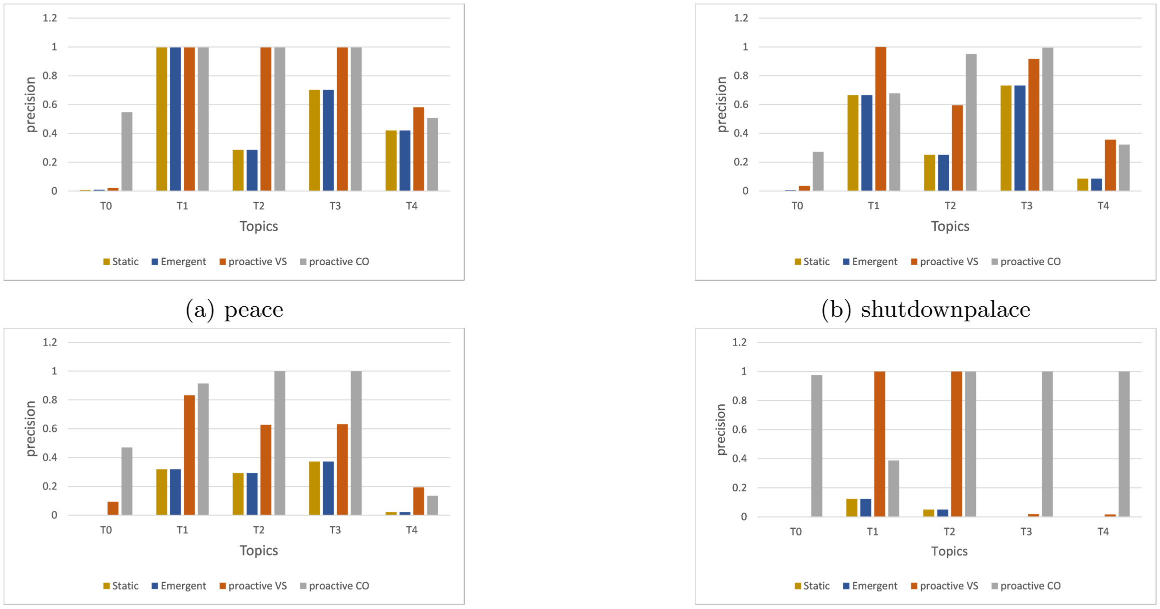
(a) baltimoreuprising

(b) curfew

(c) baltimorecurfew

(d) breaking

Figure 4.17: Prediction the events that happened in time interval 1065 from time interval 781 using the highest frequency hashtags



(a) peace

(b) shutdownpalace

(c) peaceinbaltimore

(d) policelivesmatter

Figure 4.18: Prediction the events that happened in time interval 1065 from interval 781 using the lowest frequency hashtags

Chapter 5

Lost in translation: Estimating multi-lingual linguistic context

This study evaluates the use of topic modeling with machine translation for analyzing topic similarity across two different languages using tweets surrounding the 2020 COVID-19 public health crisis as a case study. We use topic modeling through Latent Dirichlet Allocation (LDA) to generate different topics for English and Arabic tweets. Two methods for identifying common-language topics are used with a translation happening at different step in each method (full-text topic modeling, term-based topic modeling). First, we use a state-of-the art translation service to translate the entire text from one language to another, then run LDA on the translated text. We call this method *full-text topic modeling.* Second, we run LDA on the text before translation, then use a translation service to translate the LDA words from one language to another. We call this method *term-based topic modeling.* We compute semantic similarity using a low-dimensional representation of words to find similarities between topics in different languages. The performance, cost, and the execution time of the translation process are examined. The similarity between tweets in English and Arabic languages is compared. The results of Google translation services using the two methods are compared. First, the results indicate that the term-based topic modeling approach is more cost-efficient than the full-text topic modeling approach and still have comparable results in finding similar topics. Second, the computational time to translate the terms is significantly lower than that of the full text translation. It is concluded that the cost-effectiveness makes the term-based topic modeling a valuable addition to multilingual text analysis.

5.1 Motivation

As the size and scope of multilingual data grow, automated text analysis becomes necessary to allow policymakers in governmental and health system to make risk decisions and policies to respond to public health crises, natural disasters, and political or social movements. Analyzing large corpora of multi-lingual texts is challenging. Without effective automated text analysis methods, the multilingual text typically introduces significant challenges (e.g., hiring people with language skills) to the analysis process, which makes it extremely complex and costly [37]. Twitter data is one of the high-velocity data used to spread the news and other information, so questions arise about how the information contained in Twitter differs from one language to another. To answer this question, our research focuses on analyzing multilingual topics for two languages (English and Arabic) during the COVID-19 pandemic within three months using tweets collected in the same way and filtered using the same keywords. However, analyzing multilingual text across different countries where people speak different languages requires a translation technique to convert the text from one language to another. Typically, translation API (e.g., Google Translation API, DeepL, etc.) is applied to translate all text to English before applying any data analysis [38]. However, these APIs are not free, which requires considering costs as factors that affect the translation process. This thesis utilizes topic models with Google translation API and compares two translation methods. One translates the full-text and the other translates terms. The discussion leads to three questions: Which translation method should we use? Do we get lost in translation by converting text from one language to another? Which translation method is more effective in terms of cost and time? This thesis makes a step towards answering these questions by conducting an experiment to analyze the similarity between multilingual topics in Arabic and English tweets. We compare topics in English and Arabic tweets surrounding the 2020 COVID-19 pandemic using tweets collected in the same way and filtered using the same keywords. We use a probabilistic topic model to identify and extract the key topics of Twitter's discussion in Arabic and English tweets. We use two methods to analyze the similarity between multilingual topics. The first method (full-text topic modeling approach) translates all text to English and then runs topic modeling to find similar topics. The second method (term-based topic modeling approach) runs topic modeling on text before translation then translates the top keywords to find similar topics. We do our experiments using a multilingual COVID-19 Twitter dataset called GeoCoV19, which has $524,353,432$ tweets posted over 90 days since February 1, 2020 [98]. The dataset contains only the IDs of the tweets. Twarc API is a commonly-used tool to collect tweets [99]. It is utilized to collect the tweet's text from the available tweet's ID. The Google Translate API [100, 101] is used to translate from one language to another. In addition, we use well-known Python packages for text stemming and cleaning for English tweets. We use a set of available methods for preprocessing Arabic tweets such as Farasa [102]. Results indicate that the term-based topic modeling approach can reduce the cost compared to the full-text topic modeling approach and still have comparable results in finding similar topics. The computational time to translate the terms is significantly lower than that of the full text translation.

5.2 Algorithm

This thesis utilizes topic modeling through Latent Dirichlet Allocation (LDA) and a machine translation API to introduce two methods, full-text topic modeling, and term-based topic modeling. We analyze the similarity between topics in two languages (English and Arabic) during the COVID-19 pandemic and compare the two methods. In the full-text topic modeling, we translate the entire Arabic text (the full set of tweets) to English then run LDA to generate a set of topics, each of which has x top-ranked LDA words. For the term-based topic modeling, we run LDA on the Arabic text, and then use a translation service to translate the Arabic topics into English (the x top words in each topic). The LDA is also used to generate a set of topics for English text. We use the topics from English text and topics from Arabic text after translation either by full-text or term-based to compare the similarity between the two languages. We study semantic similarity using a low-dimensional representation of words in vector space precomputed using word2vec skip-gram model [77] to find similarities between topics. This representation is also known as distributed numerical representations of word features. This approach maps semantically similar words together in a vector space. Many syntactic and semantic relationships between words can be defined using simple algebraic operations on the word vectors.

The flowchart of the two methods is presented in Fig. 5.1. The two methods consist of six main steps: 1) Filter Text, 2) Text Prepossessing, 3) Machine Translation, 4) Topic Modeling, 5) Machine Translation of Terms, and 6) Topics Comparison. These steps are explained in the following

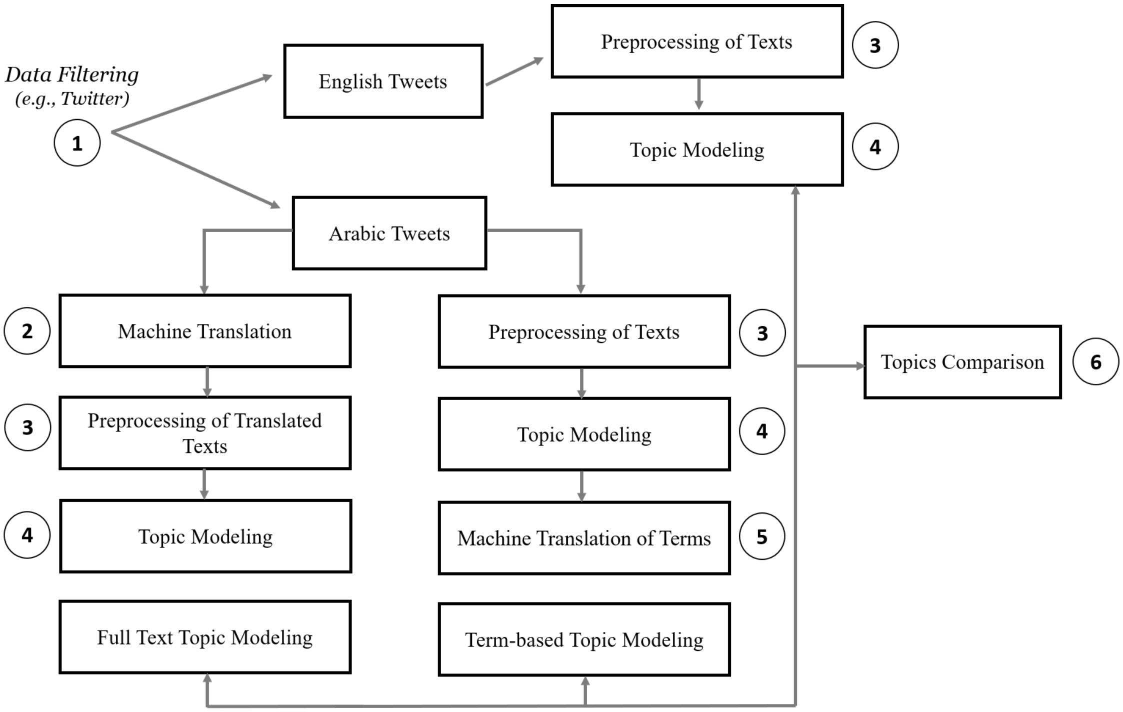


Figure 5.1: Simplified illustration of the two methods

Step 1. Filter the multilingual text based on language to English and Arabic texts

Step 2. Use Google translation API to translate Arabic text to English.

Step 3. Preprocess text in each language.

Step 4. Use LDA to extract topics in English tweets, translated tweets, and Arabic tweets.

Step 5. Use Google translation API to translate Arabic LDA topics to English.

Step 6. Use semantic cosine similarity to compare the similarity between topics in the two languages using the full-text and term-based modeling techniques.

subsections.

Data Filtering

For our study, we use a multilingual COVID-19 Twitter dataset called GeoCoV19, which has hundreds of millions of tweets posted over 90 days from February 1, 2020, until May 1, 2020, using hundreds of multilingual hashtags and keywords [98]. The dataset contains only the ID of the tweets. Twarc API is a widely used tool to collect tweets [99], it is utilized to collect the tweet's text from the available tweet's ID. After collecting all the tweets, We filter them based on the language to return English and Arabic tweets. We use the Arabic language because it is one of the most frequently used languages in Twitter with a still growing representation [103] and the official language for 22 countries.

Machine Translation

We use Google Translation API (Application Programming Interface) [100, 101] to translate text from one language to another. In the full-text topic modeling approach, we translate all the Arabic text to English using Python by passing the Arabic texts to the API after importing the translate service from google.cloud in a Python script. We choose Google Translation API because of its translation quality which is much better than other online translation services [104]. Also, Google Translation API does not have any restriction on the text length while DeepL's API has [37]. To use Google translation API, we create an account and purchase the translation service from Google. The cost of using Google translation API is US $20 per 1 million characters. The estimated cost to translate 5.1 million Arabic tweets, which has 970,801,329 characters, is US $19,420, so we decided to translate one entire month. We chose March because it has the highest number of tweets. Many emergent events happened during the month (e.g., Trump declares COVID-19 a national emergency, travel ban on Non-US citizens traveling from Europe goes into effect, World Health Organization declared the coronavirus a global pandemic).

Accordingly, the Arabic Tweet:

'آخر الإحصائيات عن فيروس كورونا الجديد الصين نحو ١٤ ألف مصاب والوفيات ٣٠٤'

is translated like this:

'The latest statistics about the new Corona virus China have about 14 thousand infected and 304 deaths'

Text Preprocessing

Text preprocessing is an essential step in text mining. Removing words that can negatively impact the quality of prediction algorithms or are not informative enough is a crucial storage-saving technique in text indexing and results in improved computational efficiency [105]. In our analysis, we have English and Arabic text, and both of them need to preprocess to eliminate the noise. We used Arabic text in two forms: the raw Arabic text without translation (Arabic Text) and translated Arabic text to English (Translated Arabic Text). Both English text and the Translated

Arabic Text have the same preprocessing steps: tokenizing the text, then converting the words to lower case, removing the special characters, punctuation, emojis, URLs, and stemming the words. For Arabic Text: The raw Arabic text has been tokenized, white spaces, punctuation marks, special characters, emojis, and URLs have been removed. We use a set of available methods for preprocessing Arabic text such as Farasa [102] for normalizing Arabic characters, removing diacritics, removing punctuations, and removing repeating characters

Topic Modeling

To evaluate the usefulness and efficiency of the two methods (full text topic modeling and term-based topic modeling), we estimated topic modeling on English text, Translated Arabic Text, and Arabic Text separately using the LDA model [27]. LDA is a generative probabilistic model, used for extracting latent topics from a large set of documents. The LDA uses a three-level Bayesian model to fit the generative process, it represents documents as random mixtures over latent topics and represents each topic as a distribution over words [27]. To run the LDA model we need to set a few parameters one of them is the number of topics. The number of topics (k) must be known and fixed before running the LDA and any probabilistic topic modeling methods [106]. We kept the parameters identical in all the models that we run on the English text, Translated Arabic Text, and Arabic Text. The number of topics was kept constant. In our experiments, We chose different values for the number of topics to compare the models such that we selected a range of k between 5 to 20 in an increment of 5.

Machine Translation of Terms

As we mentioned before, we use Google Translation API to translate text from one language to another. In the term-based topic modeling approach, we translate the top-ranked words in each topic to English. Translating a few words from one language to another has apparent advantages over translating the full-text, which requires less cost in terms of money and translation time.

Topics Comparison

We aim to compare topics estimated from the English text to topics estimated in the Arabic text using the two methods. Typically, perplexity is a measure used to evaluate the quality of topic models estimated on the same dataset [107]. However, perplexity is not applicable in our comparison since we need to compare the similarity of two models estimated on two different datasets. We instead use a metric that captures the semantic similarity between two topic models. We compare the top-ranked words in different topics using the semantic cosine similarity. We have all the topics in English text, Translated Arabic Text, and Arabic Text to this stage. The topics are a set of vocabulary sorted based on their probability distribution in a document. We use the x top-ranked words in each topic, where the typical x value ranges from 5 to 20. Table 5.1 provides an example of the top 20 words for the first LDA topic estimated on the English text, Translated Arabic Text, and Arabic Text. We can say both languages have similar tweets if they produce topics similar to each other based on the semantic cosine similarity between topics. For the term-based topic modeling to be a valuable addition to multilingual text analysis, the similarity between its topics and the English topics should be the same as the similarity between the topics estimated using full-text topic modeling. The question here is how we can find the similarity between these topics? To answer this question, we use a metric that captures the semantic similarity between two topics. We compare the top-ranked words in different topics using semantic cosine similarity. [108, 17] suggested a procedure that uses Sørensen–Dice coefficient to evaluate topic meaning similarity. [108, 17] compare the top-ranked words of two topics and judge their similarity in terms of the words they include. However, using the Dice coefficient means that the topic similarity is measured by finding the overlap between two keyword lists (two topics), which does not consider any semantic similarity between words. We modify the procedure in [108, 17] by using the semantic cosine similarity instead of Dice coefficient. We use the notion of a low dimensional vector representation of the words using the word2vec model [77]. The word2vec maps words into a low-dimensional space revealing non-trivial context-based relationships between them. This results that the semantically similar words are mapped together in a vector space. This representation is also known as distributed numerical representations of word features. To find the topic similarity, first, the word2vec skip-gram model is utilized to construct a n-dimensional representation of each word in the whole datasets (English and Translated English Text), i.e., each embedded word, or token is represented as a vector emb(t) = $(t1,t2,t3,...,tn)$. Then, we use cosine similarity to find similar topics. To find the similarity between two topics, we greedily match words to each other based on the maximum cosine similarity. More precisely, for two equally-sized sets of topics, we first match the word pair with the maximum cosine similarity, then repeat this process with the unassigned words until all words are matched. Our measure of topic similarity is then the average cosine similarity of overall selected words pairs. For example, let A denote the word embedding of a set of x top words (w_1,w_2,..w_x) from a topic estimated from English text, and let B (v_1,v_2,v_3..v_x) denote the word embedding of a set of x top words estimated from Arabic text. The cosine similarity between two words (embedding words) is the cosine of the angle between two vectors, such that the cosine similarity between two vectors w1 and v1 is,

$$cos(w1, v1) = \frac{w1 \cdot v1}{||w1|| \cdot ||B||} \tag{5.1}$$

In this example, we will find the cosine similarity between each pair of embedding words in the two topics, eliminate the words with the maximum cosine similarity, and then repeat the process to match all the pairs. The cosine similarity of the two topics is the average of the maximum cosine similarities between words in the two topics. To this stage, we will find the similarity between all the topics pairs. To find the similarity between topics from different languages, we assign each topic from one language to a topic from another language. It is a matching problem. We here follow an approach suggested in [109, 108] for this particular case.

Table 5.1: Presents top words for the First topic estimated on the tweets for the first week of March using English tweets, full-text modeling approach, and Term-based modeling approach

English	Full-text modeling approach	Term-based modeling approach
coronaviru	thi	corona
cancel	coronaviru	covid
due	wa	health
fear	time	viru
amp	one	ministri
concern	diseas	coronaviru
start	ha	global
outbreak	way	protect
ha	say	organis
toilet	day	new
paper	thousand	moham
amid	vaccin	releas
event	year	son
game	becaus	covid
life	know	viru
confer	see	protect
market	use	muzzl
god	region	educ
know	okaz	prevent

5.3 Validation experiments and results

We conduct our experiments in two parts: first, we compare the two methods based on the topic similarity in reference to the English tweets such that we compare the similarity between the topics estimated on English tweets with the topics estimated on Translated Arabic Text using the full-text topic modeling and the similarity between the topics estimated on English tweets with the topics estimated on Arabic Text using term-based topic modeling. Second, we use the cost-effectiveness and the execution time as a factors of translation to compare the full-text and the term-based topic modeling approach. For both approaches, we use the tweets from March 2020 to conduct our experiments. We split the month into 4 weeks to evaluate the similarity between the two languages in different time intervals. In the full-text topic modeling approach, we translated all the tweets for March, and then we applied the LDA model on each week for both English tweets and the Translated Arabic Text. Then, We evaluate the LDA topics estimated from the English tweets in terms of their cosine similarity with the topics estimated from the Translated Arabic Text. For the term-based topic modeling approach, we apply the LDA model on each week of Arabic tweets (Arabic Text), then we translate the topics in each week. Then, We evaluate the LDA topics estimated from the English tweets in terms of their cosine similarity with the topics estimated from the Arabic Text.

Figures 5.2-5.5 show the average cosine similarity for each LDA model using different values of number of topics for the four weeks using the full-text modeling approach and the term-based modeling approach. As appear in Figure 5.2, the average cosine similarity range from 0.69 (using 20 topics) to 0.72 (using 5 topics) using the full-text modeling approach for the first week of March. In terms of topic similarity, the average cosine similarity of the matched topic pair between the two languages is larger than 50% in all the 5 models which means that both languages cover the same topics. Using the term-based topic modeling approach, the average cosine similarity range from 0.67 (using 20 topics) to 0.68 (using 5 topics) for the same week. Also, the average cosine similarity of the matched topic pair between the two languages is larger than 50% in all the 5 models which means that both languages cover the same topics. By looking at the similarity of the two methods we notice that the term-based approach has comparable results to the full-text approach which make it an efficient approach in translation. For further investigation, we look at the topics pair between the two languages using the two approaches. Table 5.2 is the pair that has the highest cosine similarity

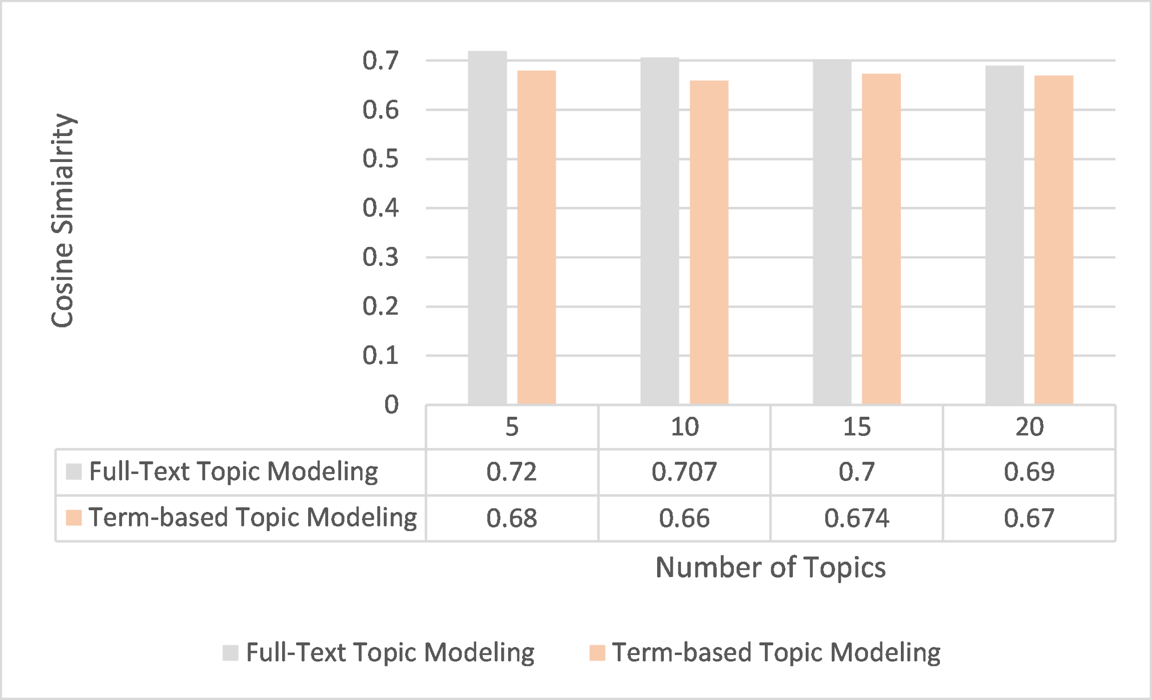
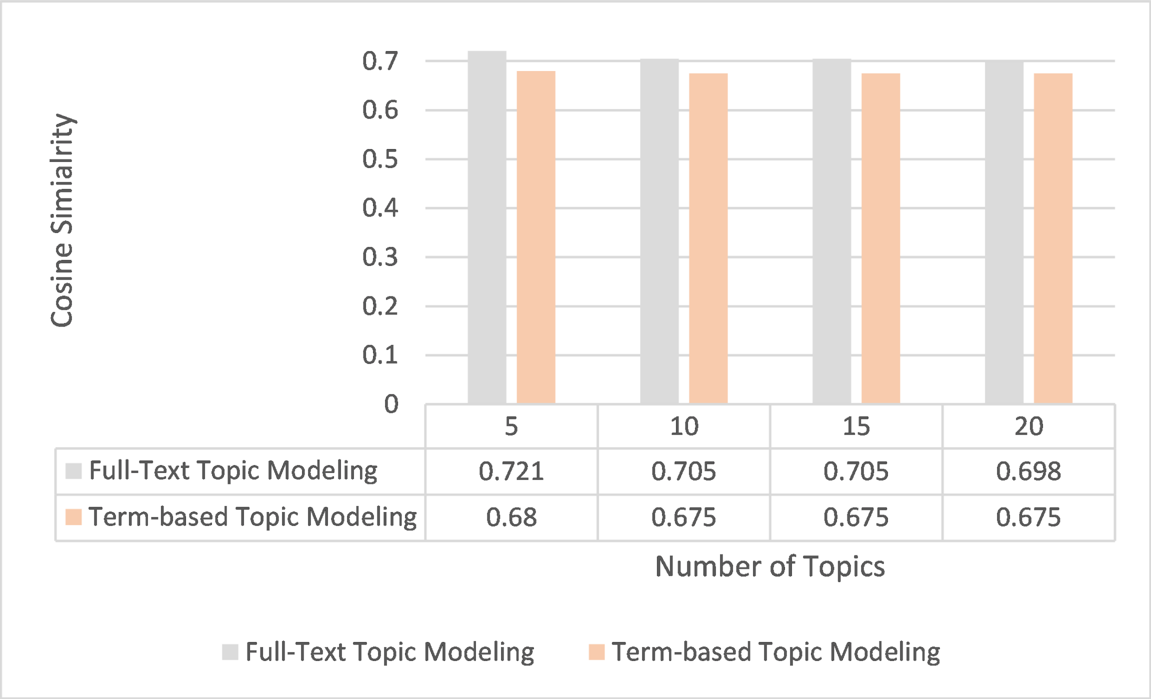


Figure 5.2: Maximum Cosine Similarity between English and Translated Arabic Text using full-text topic modeling and Maximum Cosine Similarity between English and Arabic Text using term-based topic modeling on the first week of March

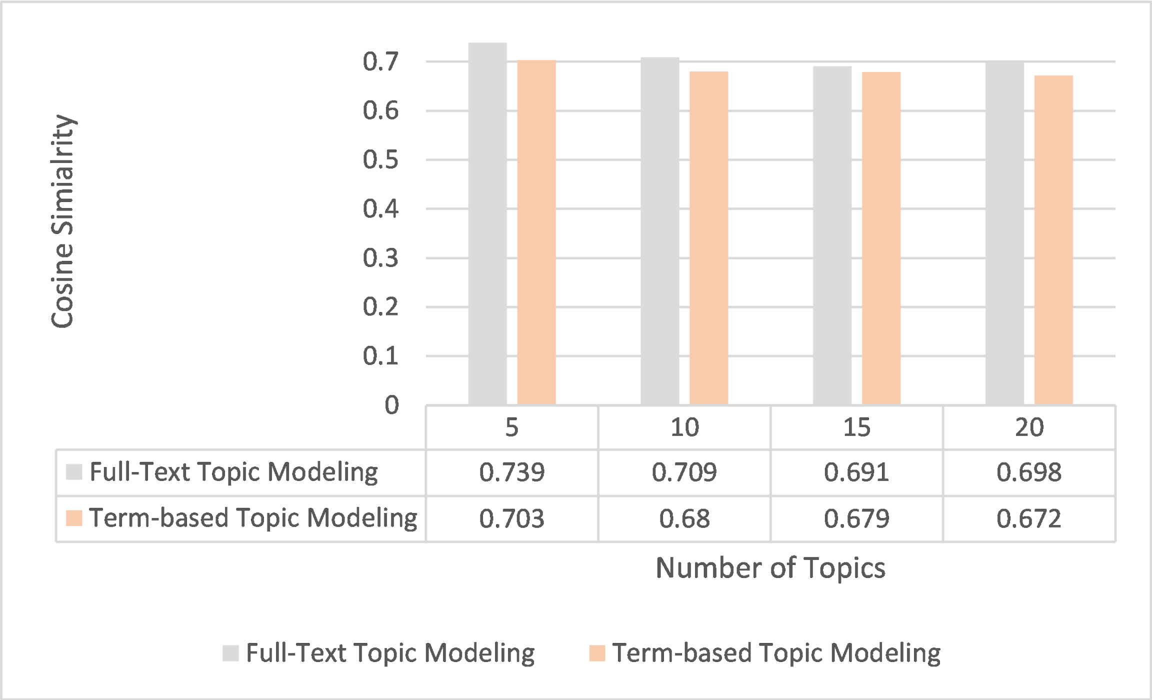
(0.7956) between English tweets and the Translated Arabic tweets for the first week of March. The pair has around 80% cosine similarity which means both topics are similar to each other. Table 5.3 is the pair that has the highest cosine similarity (0.763) between English tweets and the Arabic tweets for the first week of March using term-based topic modeling. The same as the full-text modeling approach this pair has 76% cosine similarity which indicated that both languages covers the same topic. Figures 5.3-5.5 show the same trend as we observed in Figure5.3.

Since we are talking about the semantic cosine similarity we also plot the word embedding for the matched pair from English and Arabic tweets using full-text modeling approach that has the highest cosine similarity. Fig. 5.6, visualize the embedding using the 2D PCA dimensionality reduction from the initial 100-dimensional embedding space. In this example, the corpus of documents contains tweets from English and Translated Arabic Text. The figure illustrates the two-dimensional visualization of the pairs of topic that has the highest cosine similarity for the first week of March. As we can see from this figure the word embedding of the words in the matched pair are in the top of each other or very close to each other which indicated that there is a high semantic similarity



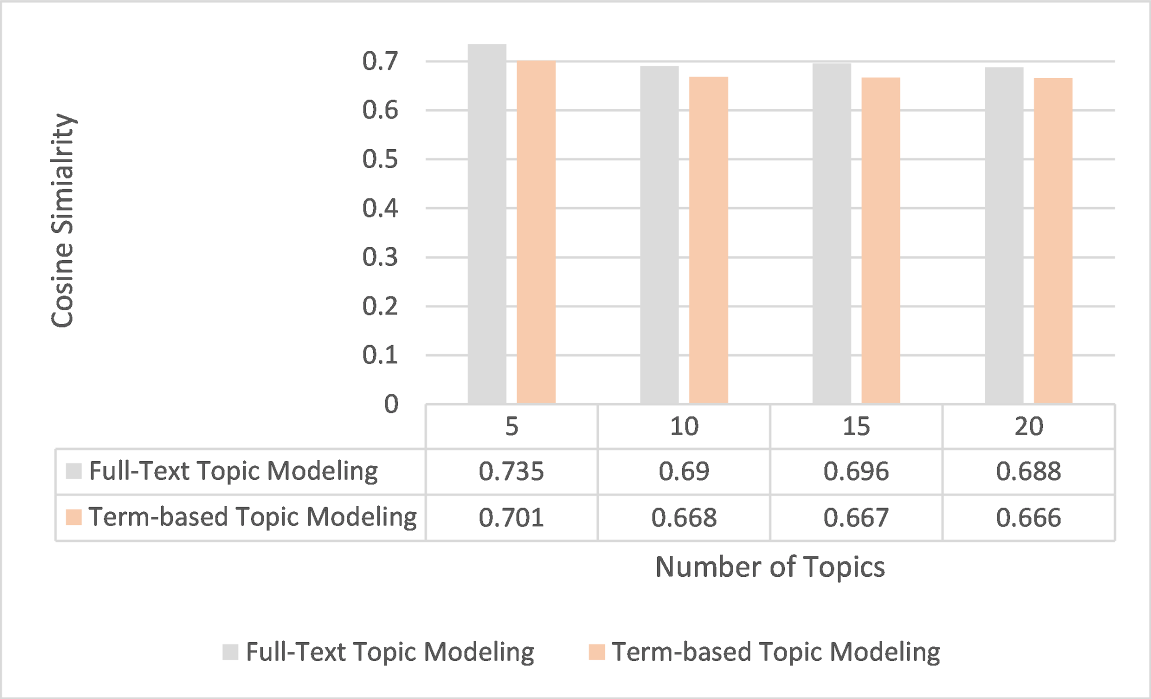
	5	10	15	20
Full-Text Topic Modeling	0.721	0.705	0.705	0.698
Term-based Topic Modeling	0.68	0.675	0.675	0.675

Figure 5.3: Maximum Cosine Similarity between English and Translated Arabic Text using full-text topic modeling and Maximum Cosine Similarity between English and Arabic Text using term-based topic modeling on the second week of March



	5	10	15	20
Full-Text Topic Modeling	0.739	0.709	0.691	0.698
Term-based Topic Modeling	0.703	0.68	0.679	0.672

Figure 5.4: Maximum Cosine Similarity between English and Translated Arabic Text using full text topic modeling and Maximum Cosine Similarity between English and Arabic Text using term-based topic modeling on the third week of March



	5	10	15	20
Full-Text Topic Modeling	0.735	0.69	0.696	0.688
Term-based Topic Modeling	0.701	0.668	0.667	0.666

Figure 5.5: Maximum Cosine Similarity between English and Translated Arabic Text using full text topic modeling and Maximum Cosine Similarity between English and Arabic Text using term-based topic modeling on the fourth week of March

Table 5.2: Presents top words for the first topic estimated on the tweets for the first week of March using English tweets and full-text modeling approach

English	Full-text modeling approach
ha	ha
coronaviru	coronaviru
whi	becaus
becaus	vaccin
peopl	see
one	okaz
amp	day
thi	diseas
like	thousand
flu	one
realli	use
go	thi
know	time
would	region
viru	know
get	period
wa	way
die	wa
onli	say
think	year

between theses two topics.

Figures. 5.7-5.9 illustrate the two-dimensional visualization of the pairs of topic that has the highest cosine similarity using the full-text topic modeling approach for the second, third and fourth weeks of March respectively. Figure 5.9 illustrates the topic that has the highest cosine similarity for the fourth week of March and this topic has 0.85 similarity which is the highest cosine similarity among all the weeks. For more investigation we look at the words of the matched topics: As it appear in table. 5.4 we can see that we have 8 words in overlap and most of the other words are semantically similar like the words 'coronaviru' and 'covid', 'pandem' and 'epidem', and 'us' and 'america' which give an evidence that the semantic cosine similarity is much better than the Dice coefficient that only consider the overlap between words.

Figures. 5.10-5.13 illustrate the two-dimensional visualization of the pairs of topic that has the highest cosine similarity using the term-based topic modeling approach for the first, second, third and fourth weeks of March respectively.

In terms of the cost we found that the total amount of money for translating a full month

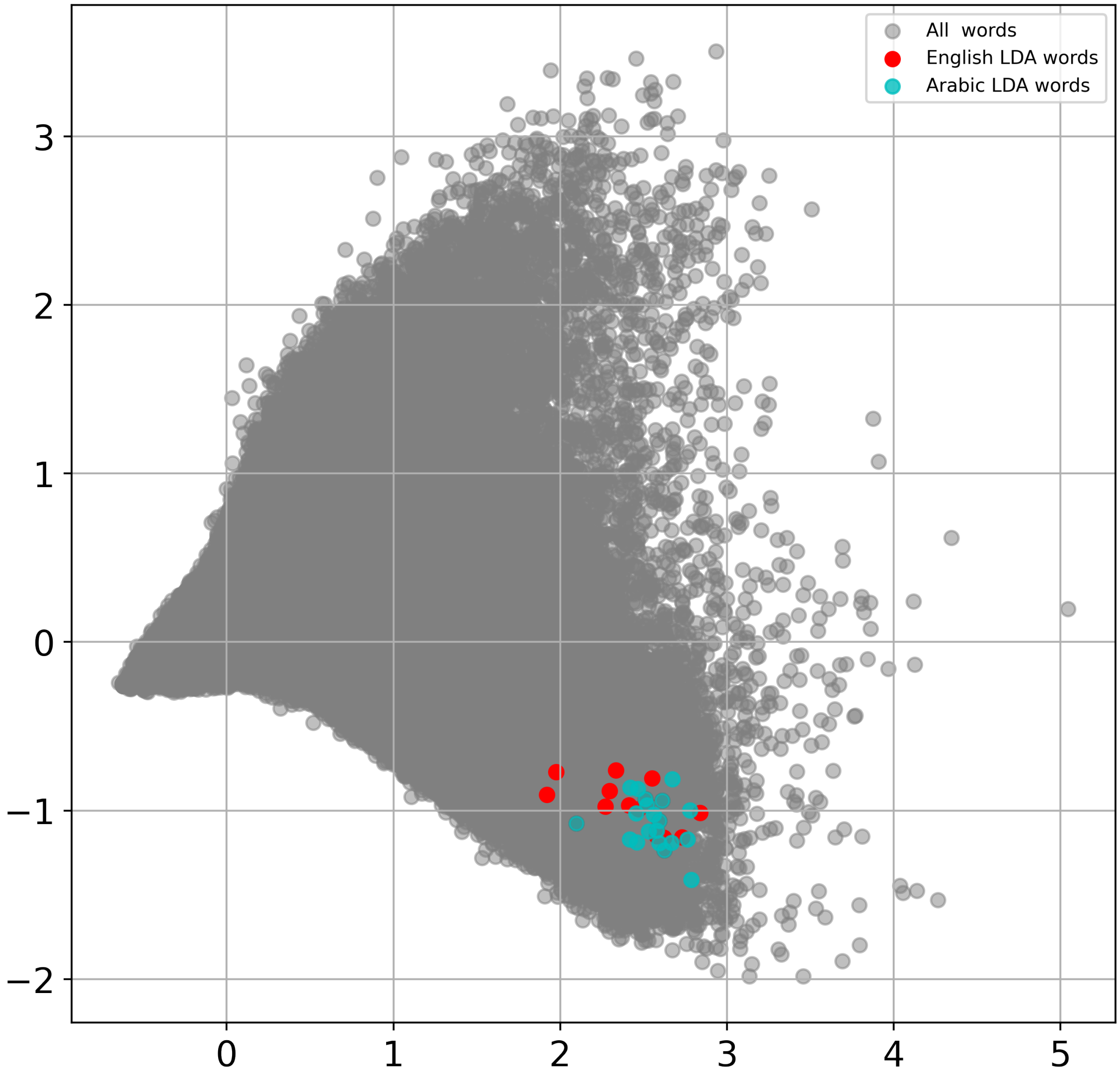


Figure 5.6: PCA-based visualization of the pair of topic that has the highest cosine similarity between English tweets and Arabic tweets using full-text topic modeling for the first week of March. The top 20 LDA words for the first topic of English tweets marked in blue. The top 20 LDA words for the first topic of Arabic tweets marked in red

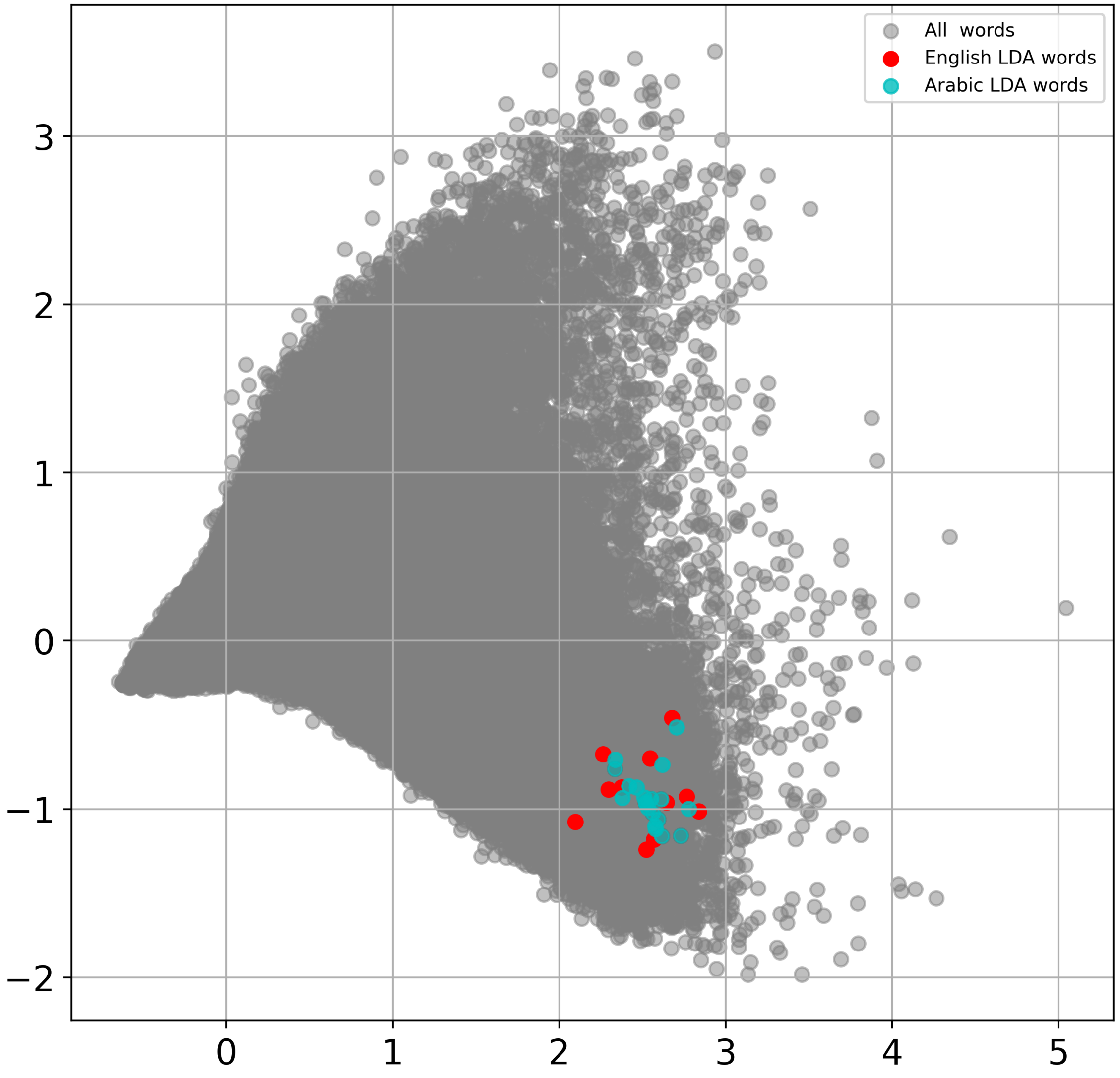


Figure 5.7: PCA-based visualization of the pair of topic that has the highest cosine similarity between English tweets and Arabic tweets using full-text topic modeling for the second week of March. The top 20 LDA words for the topic of English tweets marked in blue. The top 20 LDA words for the topic of Arabic tweets marked in red

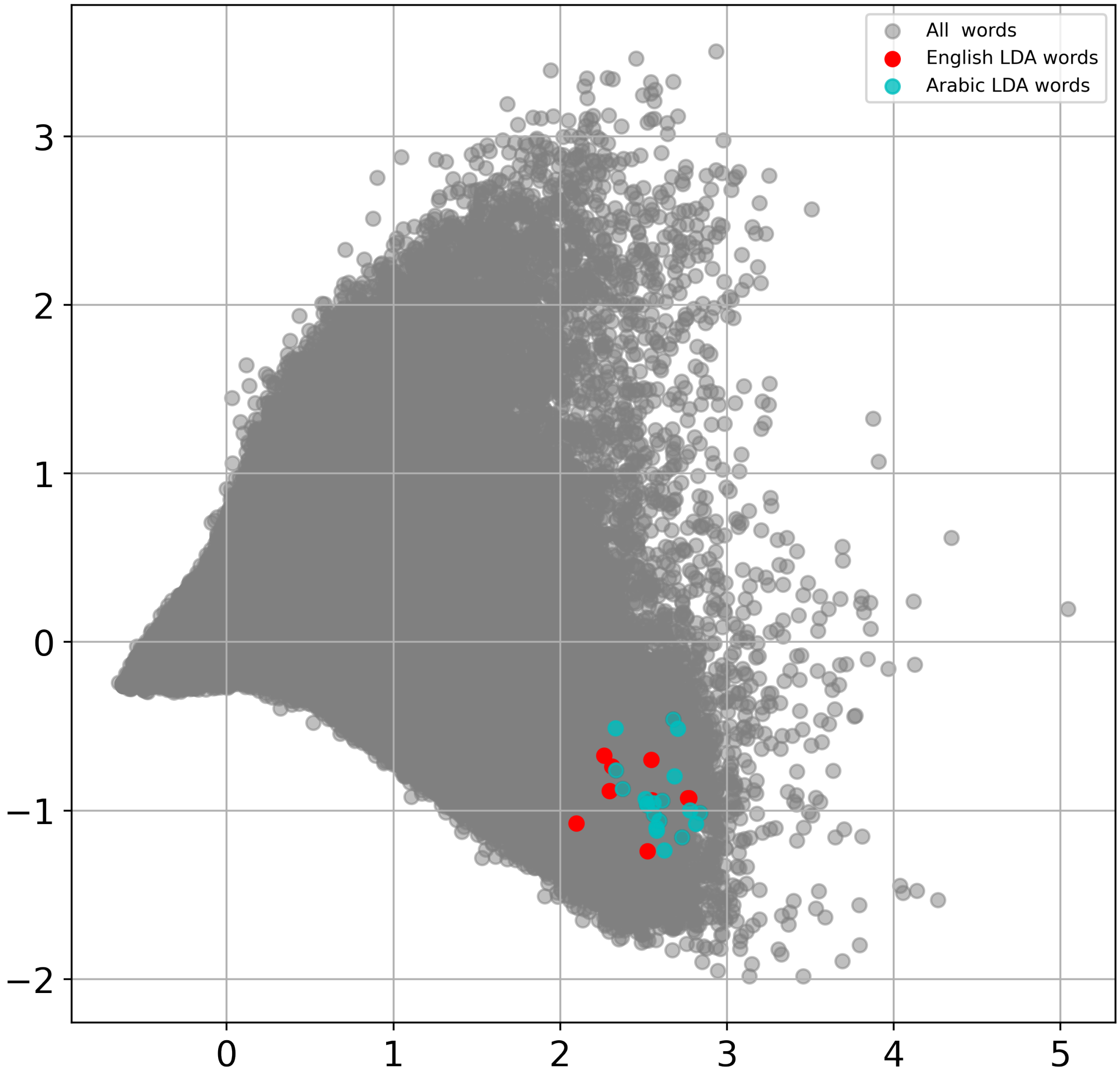


Figure 5.8: PCA-based visualization of the pair of topic that has the highest cosine similarity between English tweets and Arabic tweets using full-text topic modeling for the third week of March. The top 20 LDA words for the topic of English tweets marked in blue. The top 20 LDA words for the topic of Arabic tweets marked in red

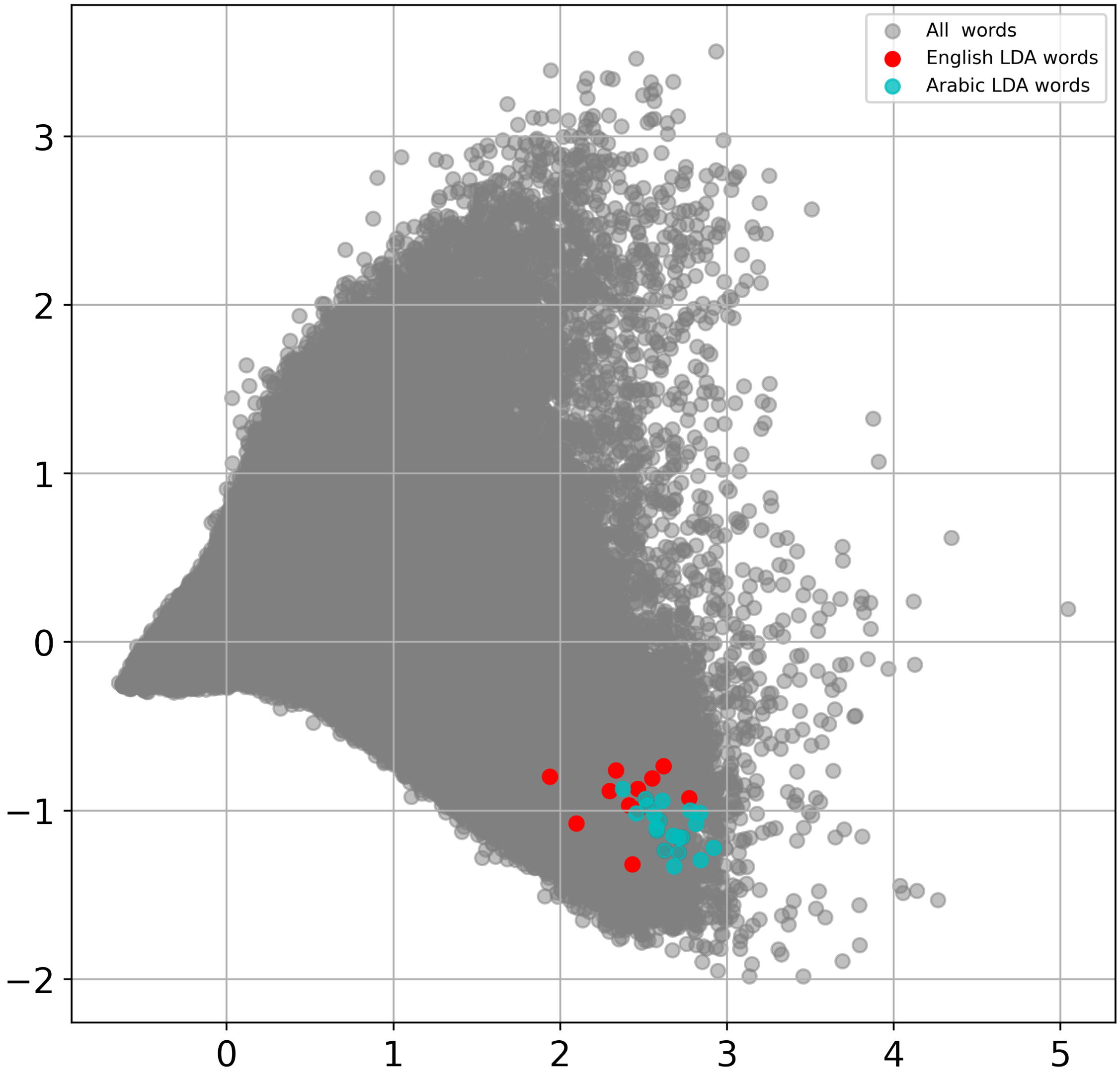


Figure 5.9: PCA-based visualization of the pair of topic that has the highest cosine similarity between English tweets and Arabic tweets using full-text topic modeling for the fourth week of March. The top 20 LDA words for the topic of English tweets marked in blue. The top 20 LDA words for the topic of Arabic tweets marked in red (0.76)

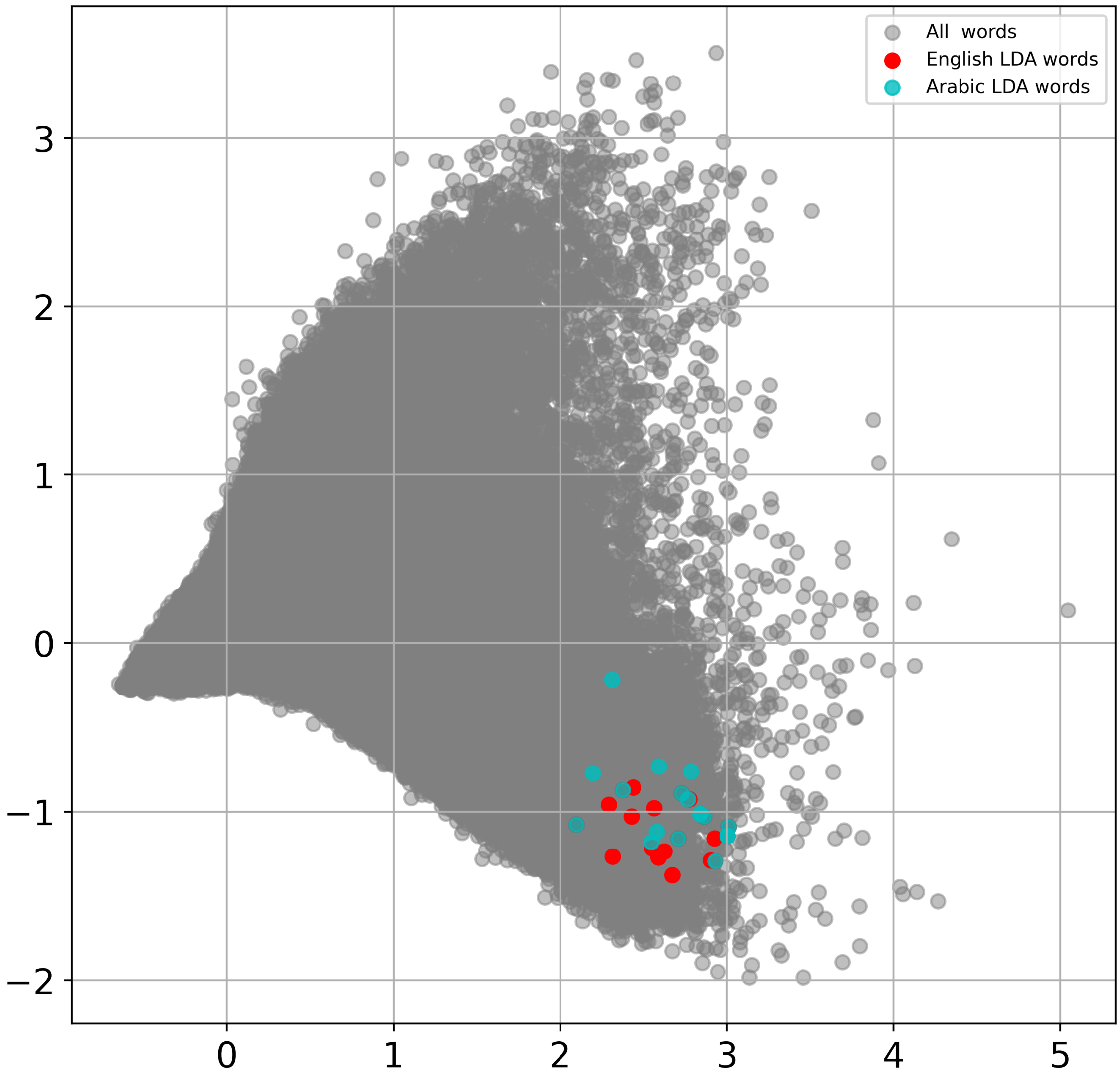


Figure 5.10: PCA-based visualization of the pair of topic that has the highest cosine similarity between English tweets and Arabic tweets using term-based topic modeling for the first week of March. The top 20 LDA words for the first topic of English tweets marked in blue. The top 20 LDA words for the first topic of Arabic tweets marked in red

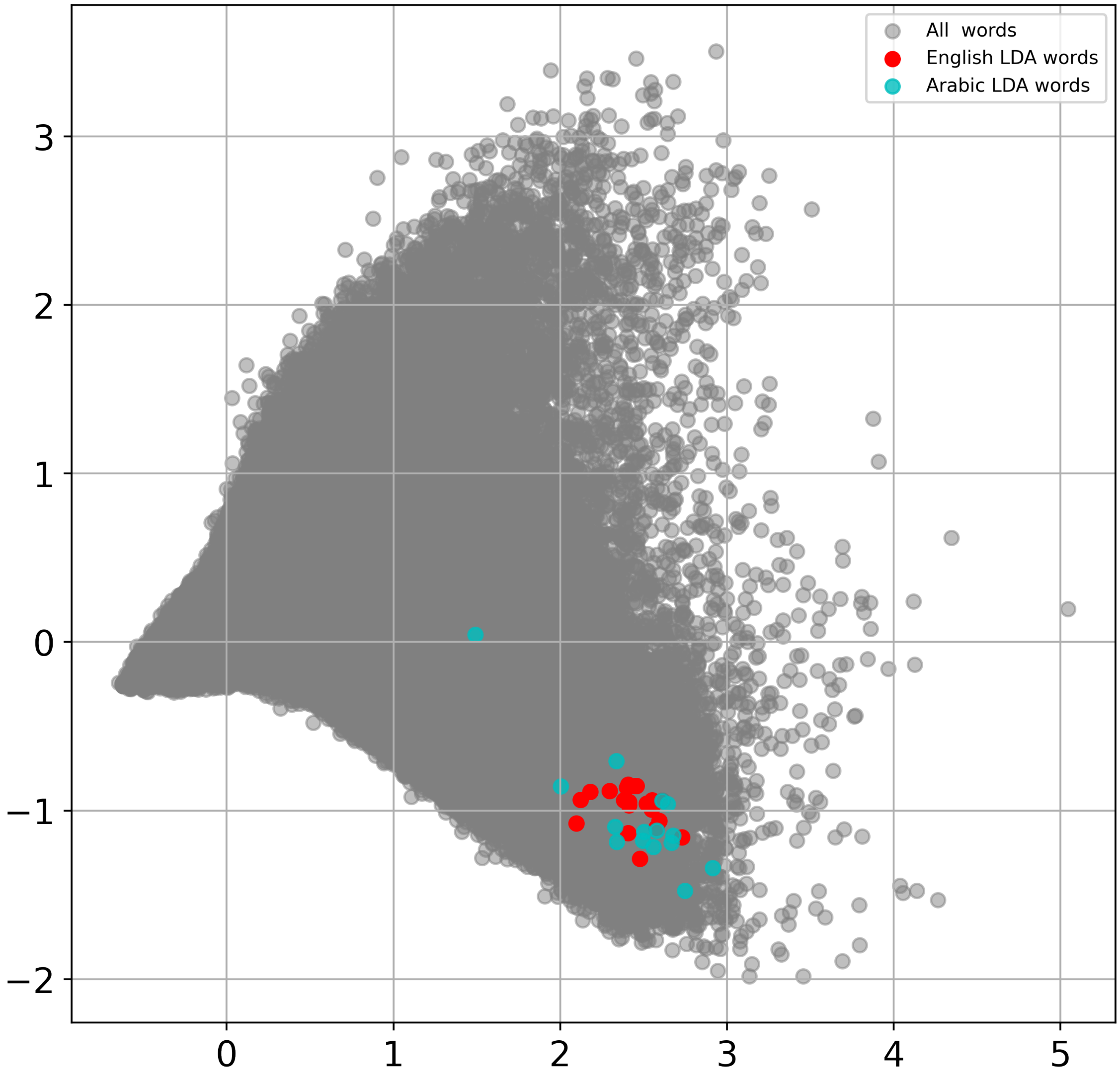


Figure 5.11: PCA-based visualization of the pair of topic that has the highest cosine similarity between English tweets and Arabic tweets using term-based topic modeling for the second week of March. The top 20 LDA words for the first topic of English tweets marked in blue. The top 20 LDA words for the first topic of Arabic tweets marked in red

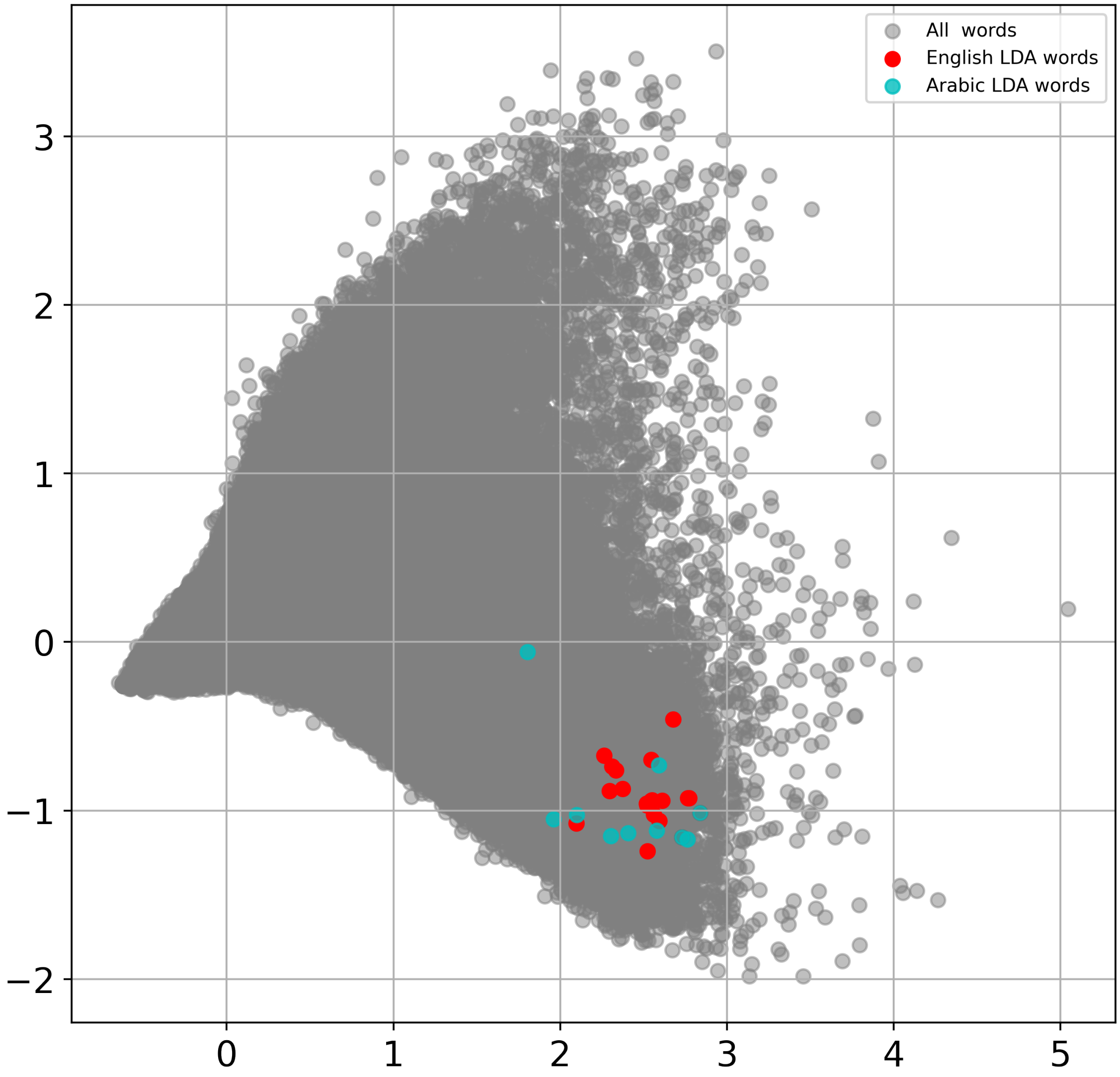


Figure 5.12: PCA-based visualization of the pair of topic that has the highest cosine similarity between English tweets and Arabic tweets using term-based topic modeling for the third week of March. The top 20 LDA words for the first topic of English tweets marked in blue. The top 20 LDA words for the first topic of Arabic tweets marked in red

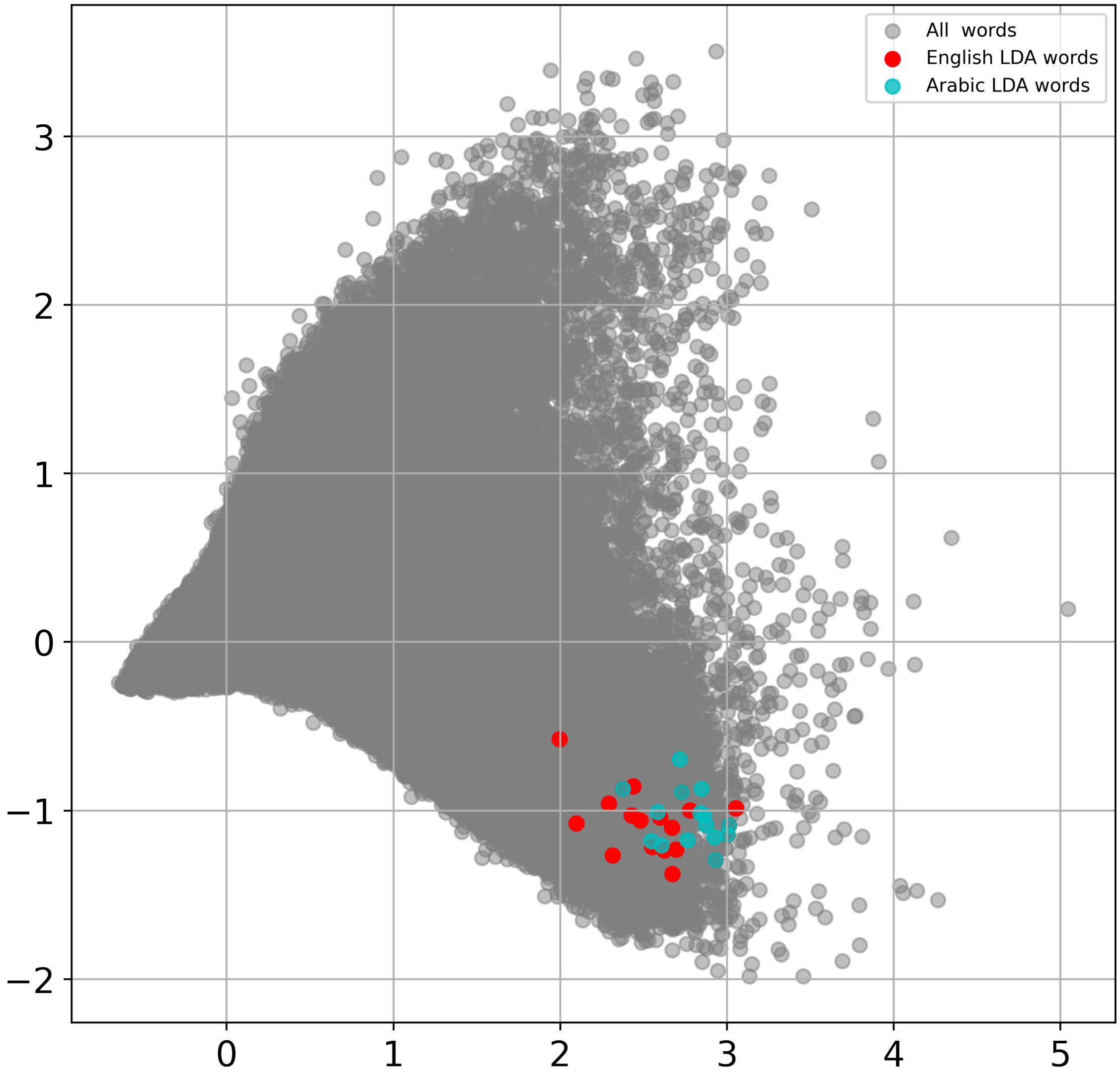


Figure 5.13: PCA-based visualization of the pair of topic that has the highest cosine similarity between English tweets and Arabic tweets using term-based topic modeling for the fourth week of March. The top 20 LDA words for the first topic of English tweets marked in blue. The top 20 LDA words for the first topic of Arabic tweets marked in red

Table 5.3: Presents top words for the first topic estimated on the tweets for the first week of March using English tweets and term-based modeling approach

English	Term-based modeling approach
case	corona
ha	covid
china	viru
confirm	coronaviru
report	condit
covid	kuwait
health	viru
news	number
us	new
new	infect
number	becaus
posit	china
coronaviru	death
first	egypt
death	case
infect	case
updat	healthi
spread	new
test	scientist
state	crown

using Google Translation API was US $1392 where we were able to translate the top LDA words for free using the same API. The Google Translation API allow us to translate up to 500,000 characters for free per month. For the translation execution time for the full-text approach it takes couple days to finish the entire translation process while it takes couple of minutes to translate the top LDA words using the term-based approach. From here we can say that the term-based topic modeling have a comparable results for the similarity between the two languages and it is more efficient in terms of cost and execution time than the full-text approach.

5.4 Conclusion

We combine topic modeling with two methods for identifying common-language topics with a translation happening at different step in each method (full-text topic modeling and term-based topic modeling) using two languages (English and Arabic). Full-text topic modeling approach translate all text in Arabic language to English language then run topic modeling on the translated text. Term-based topic modeling approach run topic modeling on the text before translation then translate the

Table 5.4: Presents top 20 LDA words for the matched pair that has the highest cosine similarity among all weeks of March using full text modeling approach

English	Full-text modeling approach
hi	year
know	world
thi	day
say	thi
would	know
coronaviru	america
ha	epidem
peopl	ha
china	peopl
die	china
wa	time
pandem	wa
realdonaldtrump	trump
go	doe
whi	countri
like	viru
trump	covid
us	one
get	human
becaus	becaus

top-key words in each Arabic topic to English. After run topic modeling and making the translation we compare the similarity between English and Arabic topics using semantic similarity where we use the notion of low dimensional representation of the word in vector space to find the cosine similarity between topics in two languages. We found that there are similar topics in English and Arabic tweets. In addition, we find that both methods have the same similarity results which make using the term-based topic modeling approach more efficient in terms of cost and translation execution time. The performance of the two methods is quantified by the similarity of topics in both languages and the cost in terms of translation execution time and cost-effectiveness. These methods is validated using multilingual COVID-19 tweets. For each approach, we apply topic modeling and finding the semantic cosine similarity between topics in each language. The term-based topic modeling approach generally achieved comparable performance of the full-text topic modeling in terms of topics similarity. The term-based topic modeling approach better than full-text topic modeling in terms of cost which needs less time to translate the text. *Our experiments indicate that the costs of the translation makes the term-based topic modeling a valuable addition to multilingual text analysis.* Finally, the proposed approach plays a key role in reducing the cost of the the translation and reducing the translation execution time.

Bibliography

[1] Jashanjot Kaur and P Kaur Buttar. A systematic review on stopword removal algorithms. *Int. J. Futur. Revolut. Comput. Sci. Commun. Eng*, 4(4), 2018.

[2] M. P. Sinka and D. W. Corne. Towards modernised and web-specific stoplists for web document analysis. In *Proceedings IEEE/WIC International Conference on Web Intelligence (WI 2003)*, pages 396–402, 2003.

[3] Urvashi Garg, Vishal Goyal, Urvashi Garg, and Vishal Goyal. Effect of stop word removal on document similarity for hindi text. *An Int. Jounal Eng. Sci.*, 2(December), 2014.

[4] Vishal Gupta and Gurpreet Singh Lehal. Preprocessing phase of punjabi language text summarization. In *International Conference on Information Systems for Indian Languages*, pages 250–253. Springer, 2011.

[5] Rajeev Puri, RPS Bedi, and Vishal Goyal. Automated stopwords identification in punjabi documents. *An Int. J. Eng. Sci.*, 8(June):119–125, 2013.

[6] Gong Zheng and GUAN Gaowa. A comparative study on between mongolian stop words and english stop words. *Journal of chinese information processing*, 4:35–38, 2011.

[7] Ruby Rani and DK Lobiyal. Automatic construction of generic stop words list for hindi text. *Procedia computer science*, 132:362–370, 2018.

[8] Sileshi Girmaw Miretie and Vijayshri Khedkar. Automatic generation of stopwords in the amharic text. *International Journal of Computer Applications*, 975:8887, 2018.

[9] Feng Zou, Fu Lee Wang, Xiaotie Deng, Song Han, and Lu Sheng Wang. Stop word list construction and application in chinese language processing. *WSEAS Transactions on Information Science and Applications*, 3(6):1036–1044, 2006.

[10] A Alajmi, EM Saad, and RR Darwish. Toward an arabic stop-words list generation. *International Journal of Computer Applications*, 46(8):8–13, 2012.

[11] Rachel Tsz-Wai Lo, Ben He, and Iadh Ounis. Automatically building a stopword list for an information retrieval system. In *Journal on Digital Information Management: Special Issue on the 5th Dutch-Belgian Information Retrieval Workshop (DIR)*, volume 5, pages 17–24, 2005.

[12] Jaideepsinh K Raulji and Jatinderkumar R Saini. Stop-word removal algorithm and its implementation for sanskrit language. *International Journal of Computer Applications*, 150(2):15–17, 2016.

[13] Rajnish M Rakholia and Jatinderkumar R Saini. A rule-based approach to identify stop words for gujarati language. In *Proceedings of the 5th International Conference on Frontiers in Intelligent Computing: Theory and Applications*, pages 797–806. Springer, 2017.

[14] L. Hao and L. Hao. Automatic identification of stop words in chinese text classification. In *2008 International Conference on Computer Science and Software Engineering*, volume 1, pages 718–722, 2008.

[15] Jinxi Xu and W Bruce Croft. Quary expansion using local and global document analysis. In *Acm sigir forum*, volume 51, pages 168–175. ACM New York, NY, USA, 2017.

[16] Yufeng Jing and W Bruce Croft. *An association thesaurus for information retrieval.* Citeseer, 1994.

[17] Yuheng Du. Streaming infrastructure and natural language modeling with application to streaming big data. 2019.

[18] Kamran Massoudi, Manos Tsagkias, Maarten De Rijke, and Wouter Weerkamp. Incorporating query expansion and quality indicators in searching microblog posts. In *European Conference on Information Retrieval*, pages 362–367. Springer, 2011.

[19] Zafar Saeed, Rabeeh Ayaz Abbasi, Imran Razzak, Onaiza Maqbool, Abida Sadaf, and Guandong Xu. Enhanced heartbeat graph for emerging event detection on twitter using time series networks. *Expert Systems with Applications*, 136:115–132, 2019.

[20] Xi Chen, Xiangmin Zhou, Timos Sellis, and Xue Li. Social event detection with retweeting behavior correlation. *Expert Systems with Applications*, 114:516–523, 2018.

[21] Mariam Adedoyin-Olowe, Mohamed Medhat Gaber, Carlos M Dancausa, Frederic Stahl, and João Bártolo Gomes. A rule dynamics approach to event detection in twitter with its application to sports and politics. *Expert Systems with Applications*, 55:351–360, 2016.

[22] Neela Avudaiappan, Alexander Herzog, Sneha Kadam, Yuheng Du, Jason Thatche, and Ilya Safro. Detecting and summarizing emergent events in microblogs and social media streams by dynamic centralities. In *2017 IEEE International Conference on Big Data (Big Data)*, pages 1627–1634. IEEE, 2017.

[23] Qi Chen, Wei Wang, Kaizhu Huang, Suparna De, and Frans Coenen. Multi-modal generative adversarial networks for traffic event detection in smart cities. *Expert Systems with Applications*, page 114939, 2021.

[24] Xing Wei and W Bruce Croft. Lda-based document models for ad-hoc retrieval. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 178–185, 2006.

[25] Xing Yi and James Allan. Evaluating topic models for information retrieval. In *Proceedings of the 17th ACM conference on Information and knowledge management*, pages 1431–1432, 2008.

[26] Jinxi Xu and W Bruce Croft. Cluster-based language models for distributed retrieval. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 254–261, 1999.

[27] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

[28] Wei Li and Andrew McCallum. Pachinko allocation: Dag-structured mixture models of topic correlations. In *Proceedings of the 23rd international conference on Machine learning*, pages 577–584, 2006.

[29] Dong Zhou and Vincent Wade. Latent document re-ranking. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1571–1580, 2009.

[30] Zheng Ye, Jimmy Xiangji Huang, and Hongfei Lin. Finding a good query-related topic for boosting pseudo-relevance feedback. *Journal of the American Society for Information Science and Technology*, 62(4):748–760, 2011.

[31] Konstantinos Christidis, Gregoris Mentzas, and Dimitris Apostolou. Using latent topics to enhance search and recommendation in enterprise social software. *Expert Systems with Applications*, 39(10):9297–9307, 2012.

[32] Midori Serizawa and Ichiro Kobayashi. A study on query expansion based on topic distributions of retrieved documents. In *International Conference on Intelligent Text Processing and Computational Linguistics*, pages 369–379. Springer, 2013.

[33] Matthew A Baum and Yuri M Zhukov. Media ownership and news coverage of international conflict. *Political Communication*, 36(1):36–63, 2019.

[34] Isaac Chun-Hai Fung, Jing Zeng, Chung-Hong Chan, Hai Liang, Jingjing Yin, Zhaochong Liu, Zion Tsz Ho Tse, and King-Wa Fu. Twitter and middle east respiratory syndrome, south korea, 2015: A multi-lingual study. *Infection, disease & health*, 23(1):10–16, 2018.

[35] Fabienne Lind, Jakob-Moritz Eberl, Tobias Heidenreich, and Hajo G Boomgaarden. Computational communication science— when the journey is as important as the goal: A roadmap to multilingual dictionary construction. *International Journal of Communication*, 13:21, 2019.

[36] Sven-Oliver Proksch, Will Lowe, Jens Wäckerle, and Stuart Soroka. Multilingual sentiment analysis: A new approach to measuring conflict in legislative speeches. *Legislative Studies Quarterly*, 44(1):97–131, 2019.

[37] Ueli Reber. Overcoming language barriers: Assessing the potential of machine translation and topic modeling for the comparative analysis of multilingual text corpora. *Communication methods and measures*, 13(2):102–125, 2019.

[38] Erik De Vries, Martijn Schoonvelde, and Gijs Schumacher. No longer lost in translation: Evidence that google translate works for comparative bag-of-words text applications. *Political Analysis*, 26(4):417–430, 2018.

[39] Chung-Hong Chan, Jing Zeng, Hartmut Wessler, Marc Jungblut, Kasper Welbers, Joseph W Bajjalieh, Wouter Van Atteveldt, and Scott L Althaus. Reproducible extraction of cross-lingual topics (rectr). *Communication Methods and Measures*, 14(4):285–305, 2020.

[40] Daniel Maier, Annie Waldherr, Peter Miltner, Gregor Wiedemann, Andreas Niekler, Alexa Keinert, Barbara Pfetsch, Gerhard Heyer, Ueli Reber, Thomas Häussler, et al. Applying lda topic modeling in communication research: Toward a valid and reliable methodology. *Communication Methods and Measures*, 12(2-3):93–118, 2018.

[41] Fabienne Lind, Jakob-Moritz Eberl, Olga Eisele, Tobias Heidenreich, Sebastian Galyga, and Hajo G Boomgaarden. Building the bridge: Topic modeling for comparative research. *Communication Methods and Measures*, pages 1–19, 2021.

[42] Jelle W Boumans and Damian Trilling. Taking stock of the toolkit: An overview of relevant automated content analysis approaches and techniques for digital journalism scholars. *Digital journalism*, 4(1):8–23, 2016.

[43] Bo Pang. lee, l.(2008). opinion mining and sentiment analysis. *Foundations and trends in information retrieval*, 2(1-2):1–135, 2008.

[44] Dasha Pruss, Yoshinari Fujinuma, Ashlynn R Daughton, Michael J Paul, Brad Arnot, Danielle Albers Szafir, and Jordan Boyd-Graber. Zika discourse in the americas: A multilingual topic analysis of twitter. *PloS one*, 14(5):e0216922, 2019.

[45] Samuel Brody and Noémie Elhadad. Detecting salient aspects in online reviews of health providers. In *AMIA Annual Symposium Proceedings*, volume 2010, page 202. American Medical Informatics Association, 2010.

[46] Byron C Wallace, Michael J Paul, Urmimala Sarkar, Thomas A Trikalinos, and Mark Dredze. A large-scale quantitative analysis of latent factors and sentiment in online doctor reviews. *Journal of the American Medical Informatics Association*, 21(6):1098–1103, 2014.

[47] Michael J Paul and Mark Dredze. You are what you tweet: Analyzing twitter for public health. In *Fifth international AAAI conference on weblogs and social media*, 2011.

[48] Liangzhe Chen, KSM Tozammel Hossain, Patrick Butler, Naren Ramakrishnan, and B Aditya Prakash. Syndromic surveillance of flu on twitter using weakly supervised temporal topic models. *Data mining and knowledge discovery*, 30(3):681–710, 2016.

[49] Kyle W Prier, Matthew S Smith, Christophe Giraud-Carrier, and Carl L Hanson. Identifying health-related topics on twitter. In *International conference on social computing, behavioral-cultural modeling, and prediction*, pages 18–25. Springer, 2011.

[50] Sachin Muralidhara and Michael J Paul. # healthy selfies: exploration of health topics on instagram. *JMIR public health and surveillance*, 4(2):e10150, 2018.

[51] Hajo G Boomgaarden and Hyunjin Song. Media use and its effects in a cross-national perspective. *KZfSS Kölner Zeitschrift für Soziologie und Sozialpsychologie*, 71(1):545–571, 2019.

[52] Frank Esser and Rens Vliegenthart. Comparative research methods. *The international encyclopedia of communication research methods*, pages 1–22, 2017.

[53] Christopher Lucas, Richard A Nielsen, Margaret E Roberts, Brandon M Stewart, Alex Storer, and Dustin Tingley. Computer-assisted text analysis for comparative politics. *Political Analysis*, 23(2):254–277, 2015.

[54] David Mimno, Hanna Wallach, Jason Naradowsky, David A Smith, and Andrew McCallum. Polylingual topic models. In *Proceedings of the 2009 conference on empirical methods in natural language processing*, pages 880–889, 2009.

[55] Tomáš Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 conference of the north american chapter of the association for computational linguistics: Human language technologies*, pages 746–751, 2013.

[56] David D Lewis. Naive (bayes) at forty: The independence assumption in information retrieval. In *European conference on machine learning*, pages 4–15. Springer, 1998.

[57] Ehsan Sadrfaridpour, Talayeh Razzaghi, and Ilya Safro. Engineering fast multilevel support vector machines. *Machine Learning*, 108(11):1879–1917, 2019.

[58] Talayeh Razzaghi and Ilya Safro. Scalable multilevel support vector machines. *Procedia Computer Science*, 51:2683–2687, 2015.

[59] Zeyi Wen, Jiashuai Shi, Qinbin Li, Bingsheng He, and Jian Chen. ThunderSVM: A fast SVM library on GPUs and CPUs. *Journal of Machine Learning Research*, 19:797–801, 2018.

[60] Jake Lever, Martin Krzywinski, and Naomi Altman. Logistic regression, 2016.

[61] Tin Kam Ho. Random decision forests. In *Proceedings of 3rd international conference on document analysis and recognition*, volume 1, pages 278–282. IEEE, 1995.

[62] Leo Breiman, Jerome Friedman, Charles J Stone, and Richard A Olshen. *Classification and regression trees.* CRC press, 1984.

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

[64] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017.

[65] Feng Zou, Fu Lee Wang, Xiaotie Deng, Song Han, and Lu Sheng Wang. Automatic construction of chinese stop word list. In *Proceedings of the 5th WSEAS international conference on Applied computer science*, pages 1010–1015, 2006.

[66] Tze Yuang Chong, Rafael E Banchs, and Eng Siong Chng. An empirical evaluation of stop word removal in statistical machine translation. In *Proceedings of the Joint Workshop on Exploiting Synergies between Information Retrieval and Machine Translation (ESIRMT) and Hybrid Approaches to Machine Translation (HyTra)*, pages 30–37, 2012.

[67] Vandana Jha, N Manjunath, P Deepa Shenoy, and KR Venugopal. Hsra: Hindi stopword removal algorithm. In *2016 international conference on microelectronics, computing and communications (MicroCom)*, pages 1–5. IEEE, 2016.

[68] Timothy C Bell, John G Cleary, and Ian H Witten. *Text compression.* Prentice-Hall, Inc., 1990.

[69] C. Gropp, A. Herzog, I. Safro, P. W. Wilson, and A. W. Apon. Clustered latent Dirichlet allocation for scientific discovery. In *2019 IEEE International Conference on Big Data (Big Data)*, pages 4503–4511, 2019.

[70] Justin Sybrandt, Ilya Tyagin, Michael Shtutman, and Ilya Safro. Agatha: Automatic graph mining and transformer based hypothesis generation approach. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 2757–2764, 2020.

[71] Justin Sybrandt, Michael Shtutman, and Ilya Safro. Moliere: Automatic biomedical hypothesis generation system. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1633–1642, 2017.

[72] Justin Sybrandt, Angelo Carrabba, Alexander Herzog, and Ilya Safro. Are abstracts enough for hypothesis generation? In *2018 IEEE International Conference on Big Data (Big Data)*, pages 1504–1513. IEEE, 2018.

[73] David Westergaard, Hans-Henrik Stærfeldt, Christian Tønsberg, Lars Juhl Jensen, and Søren Brunak. A comprehensive and quantitative comparison of text-mining in 15 million full-text articles versus their corresponding abstracts. *PLoS computational biology*, 14(2):e1005962, 2018.

[74] 2016. PubMed. (2016). `https://www.ncbi.nlm.nih.gov/pubmed/`.

[75] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA, June 2011. Association for Computational Linguistics.

[76] Evan Odell. Hansard Speeches V2.6.0 [dataset].

[77] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[78] Eleonora D'Andrea, Pietro Ducange, Alessio Bechini, Alessandro Renda, and Francesco Marcelloni. Monitoring the public opinion about the vaccination topic from tweets analysis. *Expert Systems with Applications*, 116:209–226, 2019.

[79] Anastasia Giachanou and Fabio Crestani. Like it or not: A survey of twitter sentiment analysis methods. *ACM Computing Surveys (CSUR)*, 49(2):1–41, 2016.

[80] Eugenio Martínez-Cámara, M Teresa Martín-Valdivia, L Alfonso Urena-López, and A Rturo Montejo-Ráez. Sentiment analysis in twitter. *Natural Language Engineering*, 20(1):1–28, 2014.

[81] Jeanine PD Guidry, Yan Jin, Caroline A Orr, Marcus Messner, and Shana Meganck. Ebola on instagram and twitter: How health organizations address the health crisis in their social media engagement. *Public relations review*, 43(3):477–486, 2017.

[82] Elizabeth M Glowacki, Allison J Lazard, Gary B Wilcox, Michael Mackert, and Jay M Bernhardt. Identifying the public's concerns and the centers for disease control and prevention's reactions during a health crisis: An analysis of a zika live twitter chat. *American journal of infection control*, 44(12):1709–1711, 2016.

[83] Yuliya V Bolotova, Jie Lou, and Ilya Safro. Detecting and monitoring foodborne illness outbreaks: Twitter communications and the 2015 us salmonella outbreak linked to imported cucumbers. *arXiv preprint arXiv:1708.07534*, 2017.

[84] Susmita Roy. The impact of natural disasters on violent crime (working paper). *Retrieved from the New Zealand Association of Economists (NZAE)*, 2010.

[85] Clarissa C David, Jonathan Corpus Ong, and Erika Fille T Legara. Tweeting supertyphoon haiyan: Evolving functions of twitter during and after a disaster event. *PloS one*, 11(3):e0150190, 2016.

[86] Irfan Ullah, Sharifullah Khan, Muhammad Imran, and Young-Koo Lee. Rweetminer: Automatic identification and categorization of help requests on twitter during disasters. *Expert Systems with Applications*, 176:114787, 2021.

[87] Nina Hubig, Philip Fengler, Andreas Züfle, Ruixin Yang, and Stephan Günnemann. Detection and prediction of natural hazards using large-scale environmental data. In Michael Gertz, Matthias Renz, Xiaofang Zhou, Erik G. Hoel, Wei-Shinn Ku, Agnès Voisard, Chengyang Zhang, Haiquan Chen, Liang Tang, Yan Huang, Chang-Tien Lu, and Siva Ravada, editors, *Advances in Spatial and Temporal Databases - 15th International Symposium, SSTD 2017, Arlington, VA, USA, August 21-23, 2017, Proceedings*, volume 10411 of *Lecture Notes in Computer Science*, pages 300–316. Springer, 2017.

[88] Cristian Vaccari, Augusto Valeriani, Pablo Barberá, Rich Bonneau, John T Jost, Jonathan Nagler, and Joshua A Tucker. Political expression and action on social media: Exploring the relationship between lower-and higher-threshold political activities among twitter users in italy. *Journal of Computer-Mediated Communication*, 20(2):221–239, 2015.

[89] Kate Keib, Itai Himelboim, and Jeong-Yeob Han. Important tweets matter: Predicting retweets in the# blacklivesmatter talk on twitter. *Computers in human behavior*, 85:106–115, 2018.

[90] Hinrich Schütze and Jan O Pedersen. A cooccurrence-based thesaurus and two applications to information retrieval. *Information Processing & Management*, 33(3):307–318, 1997.

[91] Jiuling Zhang, Beixing Deng, and Xing Li. Concept based query expansion using wordnet. In *2009 International e-Conference on Advanced Science and Technology*, pages 52–55. IEEE, 2009.

[92] Ankan Saha and Vikas Sindhwani. Learning evolving and emerging topics in social media: a dynamic nmf approach with temporal regularization. In *Proceedings of the fifth ACM international conference on Web search and data mining*, pages 693–702, 2012.

[93] James Benhardus and Jugal Kalita. Streaming trend detection in twitter. *International Journal of Web Based Communities*, 9(1):122–139, 2013.

[94] Erik Ortiz. Freddie gray: From baltimore arrest to protests, a timeline of the case. *NBC News*, 2015.

[95] James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA, 1967.

[96] Benjamin Bengfort, Rebecca Bilbro, Nathan Danielsen, Larry Gray, Kristen McIntyre, Prema Roman, Zijie Poh, et al. Yellowbrick, 2018.

[97] Seba Susan and Jatin Malhotra. Learning interpretable hidden state structures for handwritten numeral recognition. In *2020 4th International Conference on Computational Intelligence and Networks (CINE)*, pages 1–6. IEEE, 2020.

[98] Umair Qazi, Muhammad Imran, and Ferda Ofli. Geocov19: a dataset of hundreds of millions of multilingual covid-19 tweets with location information. *SIGSPATIAL Special*, 12(1):6–15, 2020.

[99] DocNow. GitHub. 2020. Twarc URL: `https://github.com/DocNow/twarc/`.

[100] googletrans. URL: `https://pypi.org/project/googletrans/`.

[101] Susan Lotz and Alta Van Rensburg. Translation technology explored: Has a three-year maturation period done google translate any good? *Stellenbosch papers in linguistics plus*, 43:235–259, 2014.

[102] Ahmed Abdelali, Kareem Darwish, Nadir Durrani, and Hamdy Mubarak. Farasa: A fast and furious segmenter for arabic. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: Demonstrations*, pages 11–16, 2016.

[103] Jinning Li, Yirui Gao, Xiaofeng Gao, Yan Shi, and Guihai Chen. Senti2pop: sentiment-aware topic popularity prediction on social media. In *2019 IEEE International Conference on Data Mining (ICDM)*, pages 1174–1179. IEEE, 2019.

[104] Stephen Hampshire and Carmen Porta Salvia. Translation and the internet: evaluating the quality of free online machine translators. *Quaderns: revista de traducció*, pages 197–209, 2010.

[105] Farah Alshanik, Amy Apon, Alexander Herzog, Ilya Safro, and Justin Sybrandt. Accelerating text mining using domain-specific stop word lists. In *2020 IEEE International Conference on Big Data (Big Data)*, pages 2639–2648. IEEE, 2020.

[106] David M Blei. Probabilistic topic models. *Communications of the ACM*, 55(4):77–84, 2012.

[107] Hanna M Wallach, Iain Murray, Ruslan Salakhutdinov, and David Mimno. Evaluation methods for topic models. In *Proceedings of the 26th annual international conference on machine learning*, pages 1105–1112, 2009.

[108] Chris Gropp, Alexander Herzog, Ilya Safro, Paul W Wilson, and Amy W Apon. Scalable dynamic topic modeling with clustered latent dirichlet allocation (clda). *arXiv preprint arXiv:1610.07703*, 2016.

[109] Yuheng Du, Alexander Herzog, Andre Luckow, Ramu Nerella, Christopher Gropp, and Amy Apon. Representativeness of latent dirichlet allocation topics estimated from data samples with application to common crawl. In *2017 IEEE International Conference on Big Data (Big Data)*, pages 1418–1427. IEEE, 2017.