Mini Project
Classification of Stack Overflow Posts

Group:
Craig Atkinson a1669436
Lalitphan Sae-teoh a1932456
Pratham Maharjan a1944180

The University of Adelaide

4533_COMP_SCI_7417_7717

Applied Natural Language Processing

Lecturer: Dr. Orvila Sarker

Table of Contents

1. Abstract

The development of Natural Language Processing (NLP) software will involve many challenges, however many are already documented on the website Stack Overflow. This project is to identify and categorize these common issues by applying automated topic modelling techniques to a dataset of NLP-related questions sourced from Stack Overflow via API. Three methods using transformer sentence level embeddings were used: SBERT combined with DBSCAN clustering, BERTopic, and Top2Vec. The BERTopic method was found to be the best at automatically classifying a large range of posts, categorising 56% of the posts (~13,300) into 207 meaningful topics (33 major topics with greater than 100 posts), achieving a silhouette score of 0.48. SBERT method classified only 2% of posts into 5 clusters (silhouette 0.49) and required manual topic interpretation, while Top2Vec yielded weaker cluster separation (silhouette 0.13). This project successfully tested different topic modelling methods and was able to organise many NLP related posts into useful categories of common issues for NLP developers.

2. Introduction

The objective of this mini project is to develop a database of common issues faced by developers of Natural Language Processing (NLP) software. Many developers discuss these issues on the website Stack Overflow since 2008. Stack Overflow contains thousands of technical questions and answers from real software developers and is now a huge database of common issues that developers have faced, with tens of thousands of NLP related posts.

2.1. Data Categorisation

The database of technical questions can be organised by classifying posts that are similar, so that a list of common issues can be created. Literature research was conducted to determine how an automated classification system can be developed in Python. Rivas et al. compared several different methods for text classification of online reviews of doctors' offices. The researchers classified the reviews into similar topics to extract the main topic of complaint in each review. Word embedding and clustering methods were used including word2vec and k-means clustering. Once the reviews were clustered the topic names were created based on frequent terms that appeared such as 'waiting times', or 'professional'.

Zhang et al. (2020): "Sentiment Analysis for Software Engineering" This paper explored transformer-based models specifically for categorising software engineering text content. It was found that pre-trained transformers (BERT, RoBERTa, RNN) substantially outperformed traditional ML methods, such as LDA or SVM algorithms. Transformer embedding models such as BERT take the whole document as a sequence and this allows them to capture the technical context and therefore produce more meaningful embeddings for topic clustering. Therefore, it was shown that transformer models can be an effective method for processing technical software content and should be considered for the classification of Stack Overflow questions.

2.2. Bert

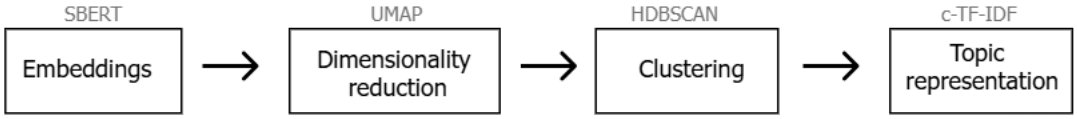
BERT (Bidirectional Encoder Representations from Transformers) is a transformer method of creating embeddings for whole documents, which has been shown to be very effective at understanding context and semantics. BERT is a pretrained foundational model that has been trained on a huge amount of text, most of which is from the internet sources such as Wikipedia, and online forums. However, it is also quite slow at generating the embeddings especially on computers without GPUs such as the ones used in this assignment. Due to the number of documents in the dataset and the time limitations of the project, some faster variations of generating word embeddings will be considered.

Sentence-BERT (SBERT) is a variation of BERT that is designed to work well with smaller documents of text and was found to run faster on CPU only computers (Reimers & Gurevych 2019). The code for the SBERT model is provided in a python library called sentence-transformers, and the pretrained models are stored on the Huggingface website. The authors of the original SBERT paper provided the pretrained model used in this assignment, called "sentence-transformers/paraphrase-multilingual-mpnet-base-v2", which was founded by the authors to be very good for generating embeddings of short paragraphs for used in clustering tasks. However, one limitation of using this SBERT pretrained model is that the input sequence size is limited to 128 tokens. For classifying Stack Overflow posts, SBERT is particularly useful because it can understand more context of technical questions, is not just finding similar words. Once the embeddings are created, clustering can be performed on the list of embeddings, so that posts of similar vectors are grouped together.

BERTopic is a topic modelling approach that enhances the traditional process by deriving coherent topic representations through a class-based variation of TF-IDF. In particular, BERTopic creates document embeddings using pre-trained language models based on transformers, clusters these embeddings, and then produces topic representations via the class-based TF-IDF method. BERTopic yields coherent topics and remains competitive across various benchmarks, including both classical models and more recent clustering methods in topic modelling. (Grootendorst, 2022)

There are three main steps for BERTopic to generates topic representations. Each document is converted into its embedding representation in first step using a pre-trained language model. Next, before clustering these embeddings, their dimensionality is reduced to optimize the clustering process. Finally, topic representations are extracted from the clusters of documents using a custom class-based variation of TF-IDF.

Figure 1 : BERTopic Pipeline (Grootendorst, 2022)



For this mini project, the embedding step that apply to BERTopic model is "sentence-transformers/all-MiniLM-L12-v2", which is used for capturing the semantics information by encoding a sentence and short paragraph. It is useful for on tasks like information retrieval, clustering or sentence similarity. It offers a balance between performance and efficiency, making it suitable for large-scale text data. However, there is a limitation that if input text longer than 256 tokens, it will be truncated by default.

2.3. Top2Vec

Top2Vec is a topic modelling approach developed by Angelov (2020). The model leverages semantic embeddings to uncover topics instead of relying on traditional word-frequency counts. Unlike methods like LDA, it doesn't require setting the number of topics beforehand or performing heavy preprocessing steps such as removing stop words or stemming. Top2Vec identifies topics by clustering documents within a semantic vector space, then selects the most representative words for each cluster. This report walks through the core components of the model: document embedding, dimensionality reduction using UMAP, clustering with HDBSCAN, and topic generation. It also details how both documents and words are embedded into the same vector space, how dense regions in this space represent topics, and how hierarchical topic reduction can optionally be used to merge closely related topics. Top2Vec jointly embeds document and word vectors in a high-dimensional semantic space. This is usually done with a Doc2Vec model, a pretrained embedding model such as the Universal Sentence Encoder has been used in this case. The Doc2Vec method generalizes Word2Vec to learn a vector for every document in the same semantic space of word vectors. So, the vector of each document is close to the vectors of words that describe its content. This joint embedding captures semantic meaning i.e. the distance between any document and word is a measure of their semantic similarity. For example, a document regarding climate change will sit close to word vectors such as "carbon" or "policy," assuming those words matter in the document.

The purpose behind embedding documents and words together is that it creates a semantic space in which it is possible to compare documents to documents, words to words, and documents to words

directly. This is an important property for Top2Vec, since it can be used to find topics by finding groups of similar document vectors and then labelling those groups with the word vectors that are closest to them.

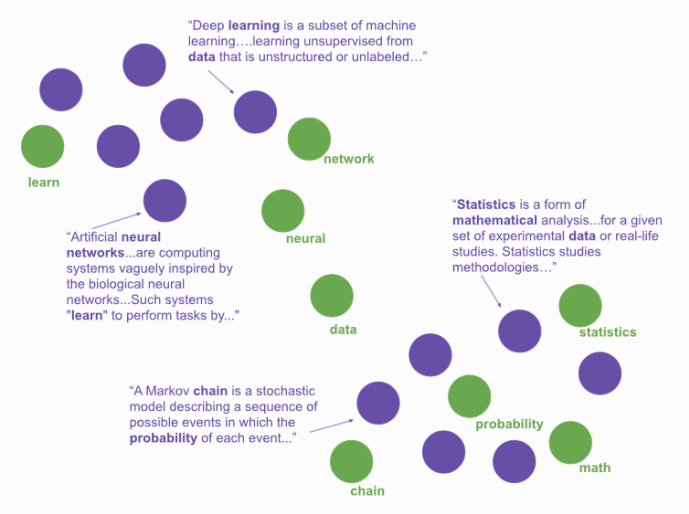


Figure 2: Top2Vec Algorithm Diagram (Angelov, 2020).

Top2Vec combines these semantic embeddings to capture contextual and syntactic relationships that bag-of-words models cannot. It takes the order and meaning of words into consideration, and so common stop-words do not dominate the representation and therefore makes stop-word removal or lemmatization unnecessary as they become scattered noise in vector space.

Top2Vec also implements an optional hierarchical topic reduction feature to structure and organize topics, which is useful if the initial clustering reveals a very high number of distinct and fine-grained topics. Angelov found that they could be hierarchically clustered or merged to create higher-level and broader topics. In this case, the topic reduction strategy used was to iteratively merge the smallest topic with its most similar topic until we have our desired lower number of topics. The similarity between topics is evaluated with the distance (e.g. cosine distance) between their topic vectors. In each merge, the two topic vectors (the smallest topic vector and the topic vector of the nearest neighbour topic) are consolidated into a single new topic vector – in this case merging may be done by averaging, weighted through sample size for the two topic clusters. After each merge, topic sizes are updated, and this process repeats creating a hierarchy in which specific and small topics are rolled up into more general and larger topics. The hierarchical clustering of topics continues until ultimately a specified number of topics are created.

3. Data Collection

Stack Overflow content is organised into questions and answers, and each question is assigned a list of tags by the question author. The author also can mark an answer as "accepted", which in most cases will be the answer that is the most correct, as the author indicated that this answer solved their issue. For this assignment, the Stack Overflow API was used to retrieve all the questions that contained tags that were considered relevant to Natural Language Processing topics. The API returns questions in JSON format, where each question has the fields: title, body, tags, accepted_answer_id, score, view_count,

creation_date, and link. The fields called title and body, are in HTML form, so that the authors' text format is preserved. Another API called is then made to get the accepted_answer content, which uses the accepted_answer_id to return the one answer that was marked as accepted, this is also return in HTML form. A Pandas DataFrame was then used to assemble the questions and answer together in a single row. The API returns a maximum of 100 questions at a time, so python loop was used to fetch all the questions that contain the specified tag. This was then saved in CSV format to make it easier to share.

The first tag used with the API was "NLP", from total 20,581 questions provided 8,530 questions with accepted answers. To retrieve more accepted answers, the related "NLP" tags were applied, where the related "NLP" tags were selected from top 20 tags that were tagged together with "NLP".

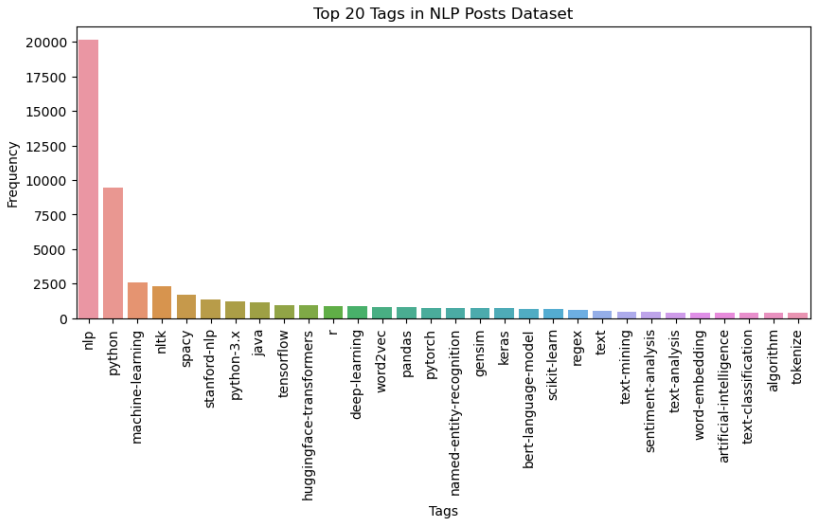


Figure 3: Top 20 Tags in NLP Post Dataset

For additional post retrieval, some of the common tags were excluded because they were broad terms that would include many topics that are not meaningful to NLP, for example "python", "machine-learning" tags. Figure 4 shows the number of total posts that were retrieved for each tag.

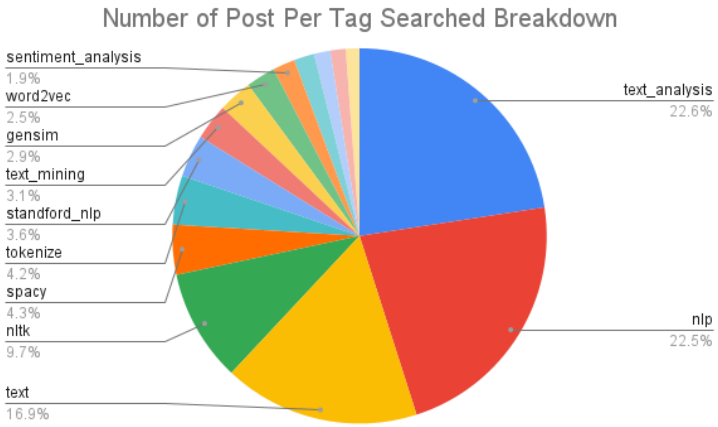


Figure 4: Breakdown of the number of posts returned by each API call, by tag.

The top 3 proportion are “text-analysis”, “nlp” and “text” tags. After merging the questions from all the selected tag and removing all the duplicate posts, there were a total of 23,873 posts, which saved as the Comma-Separated-Values (csv) file: “/data/questions_data_nlp_combine_top_tags.csv”.

3.1. Data Dictionary

Column	Description	Data Type	Example
Title	Post title.	String	Is there an algorithm that tells the semantic similarity of two phrases
Description	Post detail.	String	input: phrase 1, phrase 2 output: semantic similarity value (between 0 and 1), or the probability these two phrases are talking about the same thing
Tags	Tags of the post and related topic.	String	algorithm, nlp, semantics
Accepted Answer	Answers that accepted by the post owners.	String	<hr>\r\n\r\n You might want to check out this paper: \ r\n\r\n Sentence similarity based on semantic nets and corpus statistics (PDF)
Answer Score	Score of the accepted answers voted by Stack Overflow users.	float	44.0
Question Score	Score of the post voted by Stack Overflow users.	float	65.0
Question Views	Number of the post view.	float	49881.0
Creation Time	Post creation time.	Datetime	15/09/2008 12:26
Link	Link to the post.	String	https://stackoverflow.com/questions/62328/is-there-an-algorithm-that-tells-the-semantic-similarity-of-two-phrases

3.2. Exploratory Data Analysis

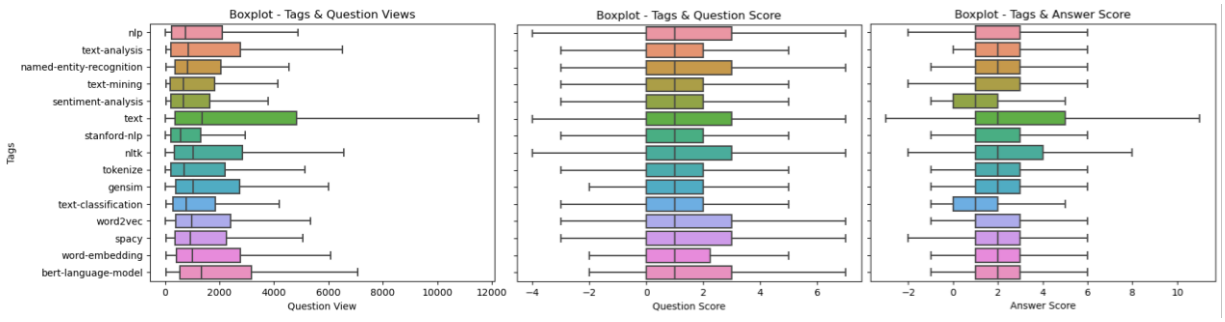


Figure 5: Boxplot between Tags, Question Views, Question Score and Answer Score.

Exploratory Data analysis was performed after merging the dataset. The boxplot shown in figure 5, includes the distribution of question views, question score and answer score for each unique tag. Most popular posts were with the "Text" and "NLP" tag which shows that both tags might be too common for this dataset. "NLTK" tag has the third most engagement from all question views, questions score and answer score making it the most popular NLP tool. "standford-nlp" tag has the lowest average question views, and the highest was "Tokenize".

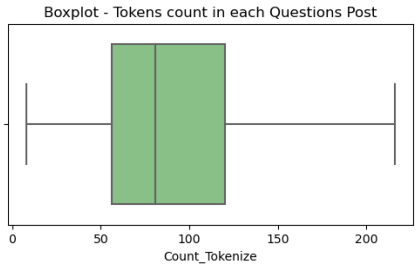


Figure 6: Boxplot Tokens count in each Question Post

After tokenising the questions content, it was found that the average length of question was 85 tokens, which is below the maximum input sequence amounts of SBERT and BERTopic methods for most of the posts. However, there is a small number of posts that are higher number of tokens that the maximum of SBERT input sequence of 128 tokens, which will be cut short in the embedding stage for this method.

4. Data Preprocessing

Before performing topic modelling, it is essential to preprocess the raw text data to improve the quality and consistency of the input. In this mini project, Text column for the target to perform categorization are combining from the Title and Description fields from the merged dataset

4.1. Text Cleaning

The following text cleaning and preprocessing techniques were applied to this column:

Technique	Description	Why It's Important
Removing Code Blocks, URLs, HTML Tags, Special Characters, Digits, Extra Whitespaces, 1 character	All code snippets, links, HTML tags, punctuation, numeric values, and excessive whitespace were removed using regex-based methods.	These elements introduce noise and are often irrelevant for semantic understanding in topic modelling. Removing them ensures the model focuses on meaningful language patterns.
Expanding Contractions	Common contractions (e.g. I'm --> I am) were expanded to their full forms using contractions library.	Expanding contractions standardizes text and improves token matching across different text samples, increasing consistency.
Lowercasing	All text was converted to lowercase.	This reduces redundancy and ensures consistent token matching.

Removing Stop Words	Common non-informative words were removed using NLTK's stop word list.	Stop words do not carry useful meaning for topic modelling and can dilute the topic signal. Removing them helps highlight more informative content.
Lemmatization	Words were lemmatized to their base or dictionary form using WordNetLemmatizer in NLTK.	Lemmatization reduces words to a standard base form, which groups similar meanings under a single token, enhancing semantic coherence across documents.

After the data preprocessing, there are 3 additional fields which are Clean_Text_Init, Clean_Text_Stopword and Clean_Text_Stopword_Lemma. As this project will apply 3 methods for topic categorisation and each method might need different approaches of data. The processed dataset was saved to "/data/ questions_data_clean_stopword_lemma_v4".

5. Data Visualisation

5.1. Word Cloud

Word Clouds are a visual tool that can provide an easy way to explore commonly used NLP terms in titles of Stack Overflow posts. In this assignment, the Python library wordcloud was used as it was found to be an easy-to-use tool that can generate a Word Cloud image from a corpus of text. The parameters of the wordcloud function allow for customisation of the image and the method of how terms are selected. The following parameters for the word cloud were selected for this assignment: top 100 terms in the corpus by frequency, arranged with text size from 10-150 based on relative frequency rank, and a bigram Dunning likelihood collocation score threshold value of 30. The bigram Dunning likelihood collocation is used to identify terms that are most likely a bigram (two-word phrases) rather than just two adjacent words (Mueller, 2020). The Word Cloud in Figure 7 was created from "title" column and was pre-processed by text cleaning process and stop word removal (refer to 4.1). Additionally, some custom words were also removed such as "text", "python", "using", "word", "file", as they were identified as being too common and have a broad meaning. From the Word Cloud below, some of the terms identified as bigram phrases including "Standford NLP", "regular expression" and "sentiment analysis".

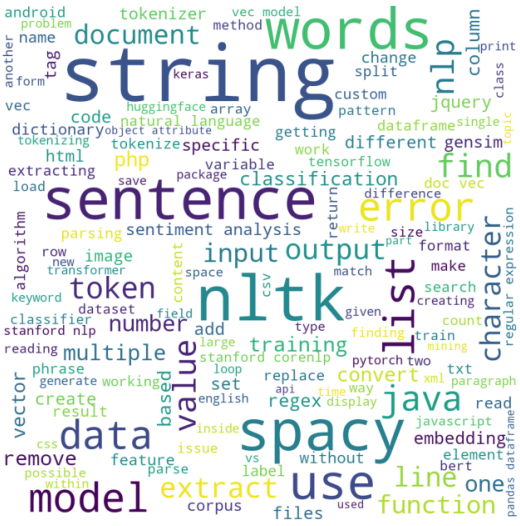


Figure 7: Word Cloud of the Titles of Stack Overflow Posts in the Dataset

6. Data Categorization Implementation

6.1. SBERT with DBSCAN Clustering

A similar method to the paper by Reimers & Gurevych (2019) was implemented to create embeddings of each of the text the posts. The text that was used was the "clean_text_init" column which is a combination of the title and question of the post, which was cleaned but stop words were not removed as it may change the way the transformer understands the text. The SBERT method was used from the sentence-transformer Python module using the pretrained model, 'sentence-transformers/paraphrase-multilingual-mpnet-base-v2'. A parameter searching method was used to find the best parameters for the clustering. The best parameter was found to be an EPS value of 0.22 which resulted in 5 clusters having 425 posts assigned and had a silhouette score of 0.49.

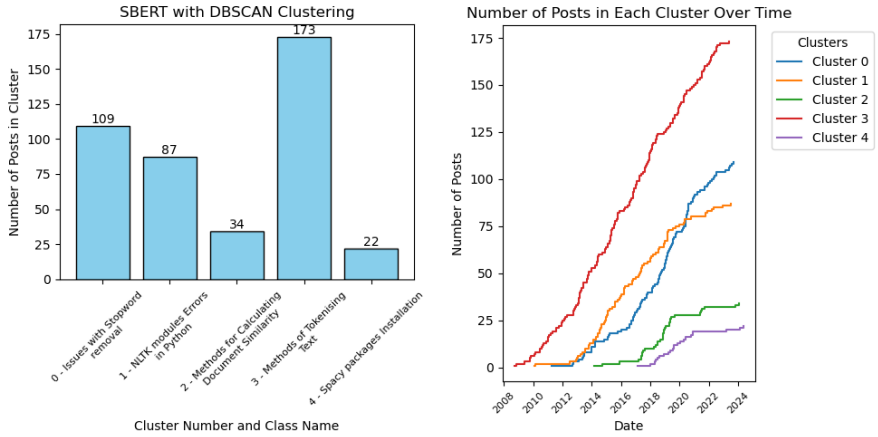


Figure 8: Number of Posts in Each Cluster using the SBERT method and each cluster evolution

The 5 clusters were then examined (Human Evaluation) to determine the Class Name.

Cluster	Class Name	Example Post	Link to Example
0	Issues with Stopword removal	How to remove Stopwords from CSV file using NLTK?	https://stackoverflow.com/questions/56122854/how-to-remove-stopwords-from-csv-file-using-nltk
1	NLTK module Errors in Python	Error installing NLTK on Win 7	https://stackoverflow.com/questions/43769655/error-installing-nltk-on-win-7
2	Methods for Calculating Document Similarity	Document similarity in Spacy vs Word2Vec	https://stackoverflow.com/questions/49767270/document-similarity-in-spacy-vs-word2vec
3	Methods of Tokenising Text	Tokenizer not working	https://stackoverflow.com/questions/30671040/tokenizer-not-working
4	Spacy Packages Installation	Can't install spacy python3.7	https://stackoverflow.com/questions/64319629/cant-install-spacy-python3-7

More examples of 10 posts of each topic can be find in the GitHub Folder "./classification examples/"

6.2. BERTopic

The pretrained model "sentence-transformers/all-MiniLM-L12-v2" is utilised for sentence embedding. The dimension reduction and clustering steps are used the default of BERTopic model which are UMAP and HDBSCAN and multiple "n_gram_range" settings were applied for this experiment.

Model	n_gram	# Topics	# Classified Posts (% from total posts)	#Topics with >100 Posts
Base Model	(1,1)	204	13,344 (56%)	38
Unigram + Bigram	(1,2)	184	13,981 (59%)	32
Unigram + Bigram + Trigram	(1,3)	206	12,221 (51%)	37
Bigram Only	(2,2)	188	13,471 (56%)	33

From human evaluation on comparison, the Bigram Only model was selected for further refinement due to its improved clarity and topic cohesiveness. To enhance topic representations for Bigram Only model, the following components were integrated into the BERTopic pipeline:

Representation Model	Why It's Important
KeyBERTInspired	Helps generate keywords that are more semantically aligned with the topic content by leveraging contextual embeddings.
MaximalMarginalRelevance (MRR)	Promotes diversity in keyword selection, reducing redundancy.
PartOfSpeech (POS) Tagging	Filters and prioritizes syntactically meaningful words (e.g., nouns and adjectives), improving interpretability.

These additions significantly improved topic labelling accuracy and expressiveness upon manual verification. Final model results as follows:

Topics	# Classified Posts	# Topics with >100 Posts
207	13,298 (56%)	33

The 33 most prominent topics (each with over 100 posts) are focused to represent key insights. These topics include common NLP issues and frequently discussed tools and libraries.

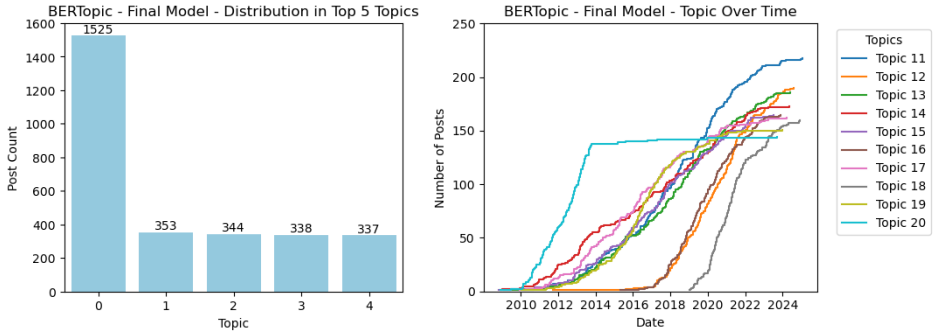


Figure 9: Number of Posts in Top 5 Topics using the BERTopic method and example topics evolution.

Full topic list is available in the appendix and examples of 10 posts of each topic can be find at "./classification examples/bertopic_topic_example.html"

The topic name is derived from the top n words that BERTopic selected from cTF-IDF. Below are examples of top 4 topics from the final model.

Topic	Topic Name	Example Post	Link to Example
0	0_text file_txt file_line text_read text	counting the number of lines in a text file	https://stackoverflow.com/questions/3482064/counting-the-number-of-lines-in-a-text-file
1	1_text classification_scikit learn_test set_training set	How to train a model to classify input to one or more classes	https://stackoverflow.com/questions/62677043/how-to-train-a-model-to-classify-input-to-one-or-more-classes
2	2_import nltk_install nltk_nltk download_nltk data	Installing megam for NLTK on Windows	https://stackoverflow.com/questions/46852316/installing-megam-for-nltk-on-windows
3	3_using jquery_text input_input field_get text	Remove all HTMLtags in a string (with the jquery text() function)	https://stackoverflow.com/questions/7889765/remove-all-htmltags-in-a-string-with-the-jquery-text-function
4	4_sentiment analysis_positive negative_sentiment score_analysis using	How to calculate the quality of comments using NLP	https://stackoverflow.com/questions/76927485/how-to-calculate-the-quality-of-comments-using-nlp

For more details of BERTopic visualization and examples please refer to BERTopic appendix.

6.3. Top2Vec

Since Top2Vec does not require too much preprocessing of the data, only the combination of the title and description of the dataset was used without steps like stop word removal or lemmatization. The pretrained model "universal-sentence-encoder" was used for sentence embedding with ngram_voca having set to true to include multi-word expressions in the vocaubalry . The number of topics was not given as a set parameter but left for the model to generate which later generated somewhere around 80-90 topics in different settings with around 60 topics having more than 100 posts. As for the dimensionality reduction and clustering, UMAP and HDBSCAN has been used respectively.

After running several hit and run trails, the parameters were selected for the model that reduces the clusters to 7 having more than 500 posts. The value of cluster_selection_epsilon, min_cluster_size, min_samples has been updated to group clusters together and reduce the total number of clusters.

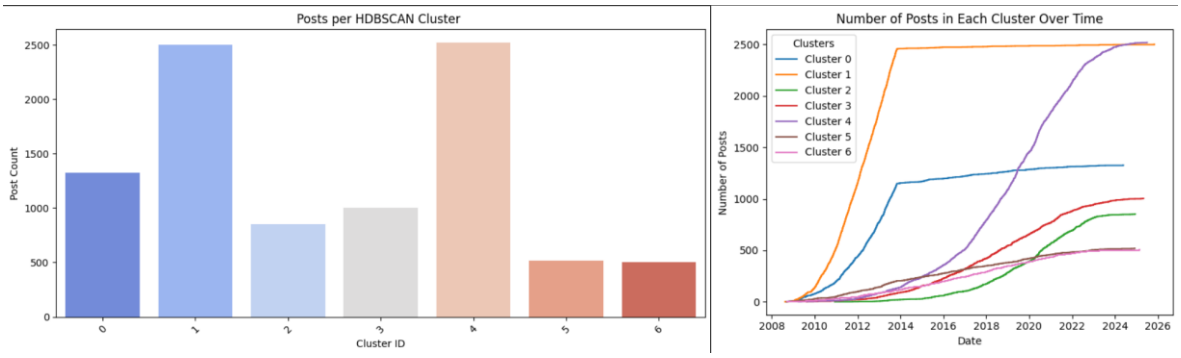


Figure 10: Number of Posts in Top 5 Topics using the Top2Vec method.

Examples of the posts in each cluster are:

Cluster	Class Name	Example	Link
0	Reading and Processing Text Files	What is the default chunker for NLTK toolkit in Python?,	https://stackoverflow.com/questions/1687510/what-is-the-default-chunker-for-nltk-toolkit-in-python
1	Text Extraction and HTML/Image Handling '	Stop-word elimination and stemmer in python	https://stackoverflow.com/questions/3882921/stop-word-elimination-and-stemmer-in-python
2	Text Processing in DataFrames	Word Co-occurrence - find the co-occurence of a term in a set of n-grams	https://stackoverflow.com/questions/6510338/word-co-occurrence-find-the-co-occurence-of-a-term-in-a-set-of-n-grams
3	Python NLP Library Installation and Errors	Error while installing spacy,	https://stackoverflow.com/questions/48465968/error-while-installing-spacy
4	Training NLP Models with Text Data	Sentiment analysis with NLTK python for sentences using sample data or webservice?	https://stackoverflow.com/questions/2832394/sentiment-analysis-with-nltk-python-for-sentences-using-sample-data-or-webservic
5	Regex and String Manipulation in Text	Storing list in a pandas DataFrame column	https://stackoverflow.com/questions/38710061/storing-list-in-a-pandas-dataframe-column
6	Basic Sentence and Word-Level NLP Tasks	Calculating Cosine Similarity of two Vectors of Different Size	https://stackoverflow.com/questions/12497394/calculating-cosine-similarity-of-two-vectors-of-different-size

Further examples of the 10 posts of each topic can be found at GitHub at classification_example/Top2Vec_example.HTML.

6.4. Result and Discussion

As they are cluster based so we tried to use silhouette score to compare the performance of cluster generation. Best model of each method is listed below.

	SBERT	BERTopic	Top2Vec
Number of Clusters	5	207	7
Number of Posts Classified	425 (2%)	13,298 (56%)	9216 (39%)
Silhouette Score	0.49	0.48	0.13
Strengths	Created well defined clusters	Created clusters with meaningful topic name Convenient no need to implement from scratch	Can group small cluster together with similar cluster to reduce the number of topics Top2Vec had better results on smaller texts such as social media posts.
Weakness	Input sequence is limited to 128 tokens. Human evaluation needed to create meaningful class names.	Too many clusters, the reduce topics feature still not work well.	Human effort is required to provide meaningful names to the classes.

7. Conclusion

The BERTopic was therefore shown to have the best results for the automatic classifying Stack Overflow posts, as it was able to classify 56% of the posts (~13,300) into 207 meaningful topics (33 major topics with greater than 100 posts), and it had decently distinct topics due to low cluster overlap, which was indicated by the silhouette score of 0.48. SBERT method in comparison, only classified only 2% of posts into 5 clusters (silhouette 0.49) and it also required manual topic interpretation, while Top2Vec yielded weaker cluster separation (silhouette 0.13), and so the topics were not very distinct from each other.

This project successfully tested different topic modelling methods to organise relevant Stack Overflow content into useful categories of common issues for NLP developers. In the future, additional methods of clustering and embedding can be examined, and compared to the results to try to improve the clustering silhouette score closer to 1, which would indicate excellent cluster separation, and therefore, more distinct topic classifications.

8. References

Mueller, A 2020, WordCloud for Python documentation. [online] Available at: https://amueller.github.io/word_cloud/ [Accessed 9 April 2025].

Vargo, CJ 2025, 'The Computational Content Analyst: Using Machine Learning to Classify Media Messages', First edition., Routledge, New York, New York.

Reimers, N & Gurevych, I 2019, 'Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks' arXiv.Org.

Grootendorst, M 2022, 'BERTopic: Neural topic modeling with a class-based TF-IDF procedure', arXiv.Org.

Angelov, D. 2020. 'Top2Vec: Distributed Representations of Topics', arXiv.Org.

Egger, R & Yu, J 2022, 'A Topic Modeling Comparison Between LDA, NMF, Top2Vec, and BERTopic to Demystify Twitter Posts', *Frontiers in Sociology*, vol. 7, pp. 886498–886498.

Le, QV & Mikolov, T 2014, 'Distributed Representations of Sentences and Documents'.

9. Appendix

GitHub Repository: https://github.com/lalitphan-rainie/COMPSCI7417_NLP_Assignment2/

BERTopic appendix

Group Member Task Breakdown

Task	Group Member Assigned
Data Collection (API)	Craig & Lalitphan
Data Pre-processing	Pratham
Exploratory Data Analysis	Lalitphan
Post Titles Word Cloud	Craig & Pratham
Categorisation Method 1: SBERT + DBSCAN	Craig
Categorisation Method 2: BERTopic	Lalitphan
Categorisation Method 3: top2vec	Pratham
Categorisation Method Evaluation	Craig, Lalitphan & Pratham
Final Categorisation Visualisations	Craig, Lalitphan & Pratham