# trCOMP4432 Group Project Report

21100602d LIU Yuzhou

21100038d YUAN Yunchen

21099695D LU Zhoudao

## 1. Introduction

For this project, we will leverage 'A million News Headlines' dataset to do three different tasks, topic modelling (Unsupervised learning), classification (supervised learning) and stock price prediction (Application). In addition to traditional machine learning and natural language processing (NLP) methods, we have made many attempts to combine deep learning, sentiment analysis and large language models (LLM) in both depth and breadth.

The remainder of this report is structured as follows. Section 2 analyze the basic information of the dataset; section 3 show how to preprocess the original texts; Section 4 apply machine learning and deep learning for topic modeling; Section 5 combines LLM to do classification tasks and Section 6 try to use texts to predict stock price movement based on sentiments and topics.

## 2. Data Understanding

| | |
|---|---|
| count | 6882.000000 |
| mean | 180.788143 |
| std | 84.916556 |
| min | 1.000000 |
| 25% | 108.000000 |
| 50% | 191.500000 |
| 75% | 245.000000 |
| max | 384.000000 |

Name: count, dtype: float64

**Table 1:** Number of data statistics

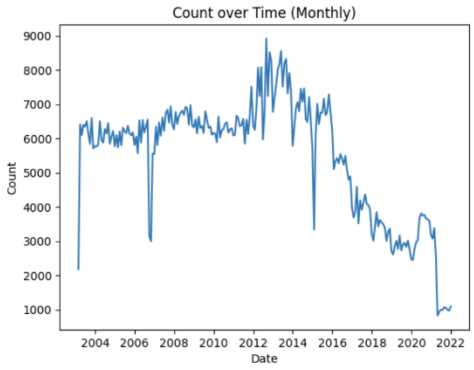



**Figure 1**: Number of data statistic

*Million News Headlines* is a public dataset in Kaggle, which contains data of news headlines published over a period of nineteen years. This news dataset is a summarized historical record of noteworthy events

**Figure 2:** Wordcloud for words

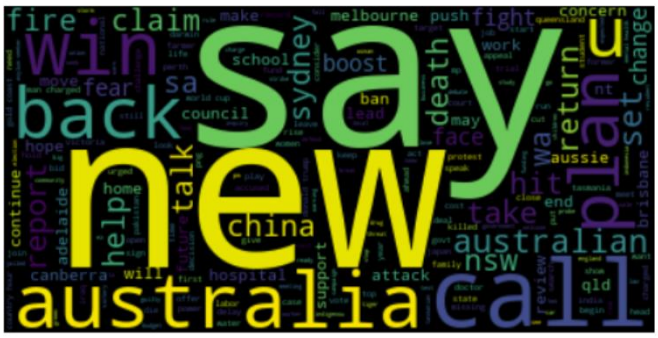
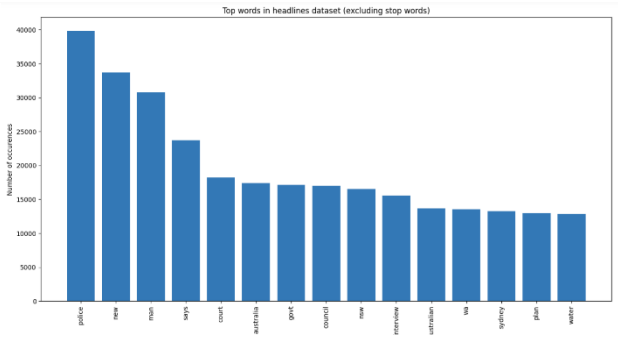



**Figure 3:** word count




In the globe from 2003-02-19 to 2021-12-31 with a more granular focus on Australia. There are 1, 244, 184 headlines in total. An average of 180 news items a day. The highest was 384 entries on 2012-08-24, but the lowest was only one on 2017-02-09 and 2021-08-06. The quantity based on days /months is presented in table 1 and Figure 1.

Moreover, all the headlines have 8, 158, 765 words, 118, 046 of them are unique. Figure 2 is a WordCloud for the words; 'news', 'say', 'australia' and so on are very popular; Figure 3 shows the top 15 most frequent words in the dataset excluding the non-sense stop words (e.g. a, an, the). 'police', 'news', 'man' appear frequently:

## 3. Data Preprocessing

### *3.1 Tokenization*

Tokenization is a fundamental task in natural language processing that involves dividing text into discrete units called tokens. Stop words are commonly used words that are often considered insignificant or have little contribution to the overall meaning of a sentence or document, like 'the' 'is', 'and' and 'in'; Punctuations are some symbols like ',', '.' And '!'. We consider stop words and punctuations are meaningless so that removing them.

### *3.2 Stemming*

Stemming is a text normalization technique in natural language processing that aims to reduce words to their base or root form. For example, 'manager', 'management' and 'managing' can be considered as one word when analyzing in case the number of different words is too high.

### *3.3 Lemmatization*

lemmatization takes into account the morphological analysis of words and applies linguistic rules to determine the lemma. For example, "cats" is lemmatized to "cat," "running" to "run," and "playing" to "play."

The preprocessing code is listed in Figure 4. We also listed sample data after precprocessing in Table 2:

**Figure 4:** Preprocessing code

```
def process(headline):
    tokens = tokenize(headline)

    stemmer = SnowballStemmer('english')
    stem = stem_tokens(tokens, stemmer)

    le=WordNetLemmatizer()
    tokens=[le.lemmatize(w) for w in stem]
    cleaned_text=" ".join(tokens)
    return cleaned_text
```

**Table 2:** sample preprocessing data

| | publish_date | headline_text | headline_cleaned_text |
|---|---|---|---|
| 0 | 20030219 | aba decides against community broadcasting lic... | decid against communiti broadcast licenc |
| 1 | 20030219 | act fire witnesses must be aware of defamation | fire must awar defam |
| 2 | 20030219 | a g calls for infrastructure protection summit | call infrastructur protect summit |
| 3 | 20030219 | air nz staff in aust strike for pay rise | staff aust strike rise |
| 4 | 20030219 | air nz strike to affect australian travellers | strike affect australian travel |

## 4. Topic Modeling

### 4.1 Overall

#### *4.1.1 Introduction*

Topic modeling is an unsupervised learning that cluster texts and modeling topics. Unlike classification, topic modeling doesn't have any labels. It is the process of extracting topics from unlabeled data, and the extracted topics are represented by keywords. The whole process is divided into 4 steps:

1. embedding
2. dimensionality reduction
3. clustering
4. topic representation learning.

The goal is to extract a generalization of the topic uniform to the content and to ensure enough differences between the different topics to measure the model's performance by the two Metrics TC and TD.

#### *4.1.2 Metric explanation*

Here we use **Topic diversity (TD)** and **Topic coherence (TC)** to evaluate the performance of our model. While TD is calculated based on the **Normalized Pointwise Mutual Information (NPMI)** directly. For TC, here we choose to use cv score to evaluate the coherence of the topic, the score is calculated based on NPMI value, and the value ranges from [0, 1], a larger value indicates a better performance of the model.

### 4.2 Traditional Topic Modeling

In the following, experiments and analyses have been conducted using traditional topic modelling methods and deep learning based topic modelling methods respectively.

#### *4.2.1 LSA*

Before applying LSA to cluster the documents, we first need to calculate the tf-idf score for different words, which can effectively indicate the strength of the word in determining which topic it belongs to.

$$\mathrm{tfidf}(t, d, D) = \mathrm{tf}(t, d) \cdot \mathrm{idf}(t, D)$$

Then it will use the SVD method to reduce the dimensionality of the matrix, which means to project the document and the word into the same latent space. And then, based on the result, LSA will automatically operate to cluster the topics into several clusters (By selecting n largest singular values). Then we can select the top-k words to represent different topics.

### *4.2.2 LDA*

LDA (Latent Dirichlet Allocation) is an unsupervised text topic model used to discover hidden thematic structures within a collection of texts. LSA is a technique based on matrix decomposition, LDA is a generative model. LDA represents text data as a document-word matrix, LSA represents text data as a word-document matrix. There are still other differences make the result different.

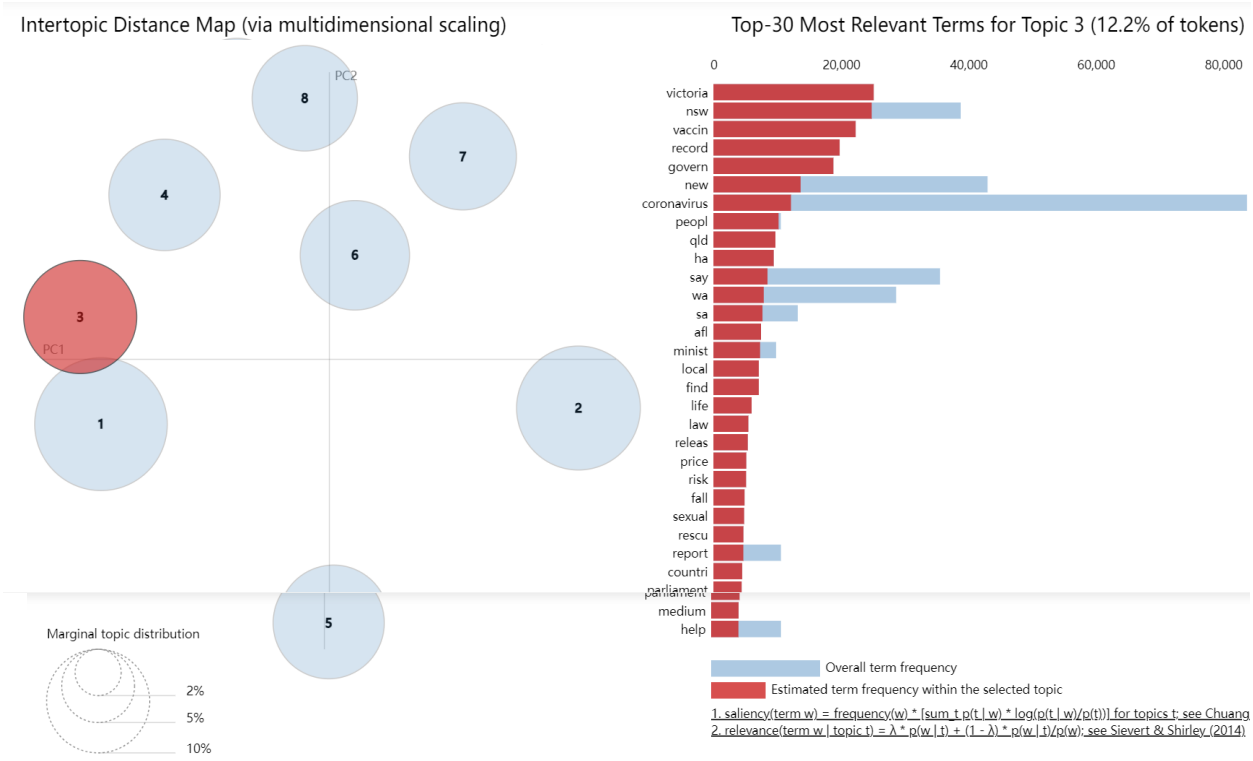



**Figure 5:** LDA visualise

### *4.2.3 Result*

The following is the result of the clustering result of LSA and LDA methods. And the table shows the cluster of topics. And we select the top 6 words to represent different topics.

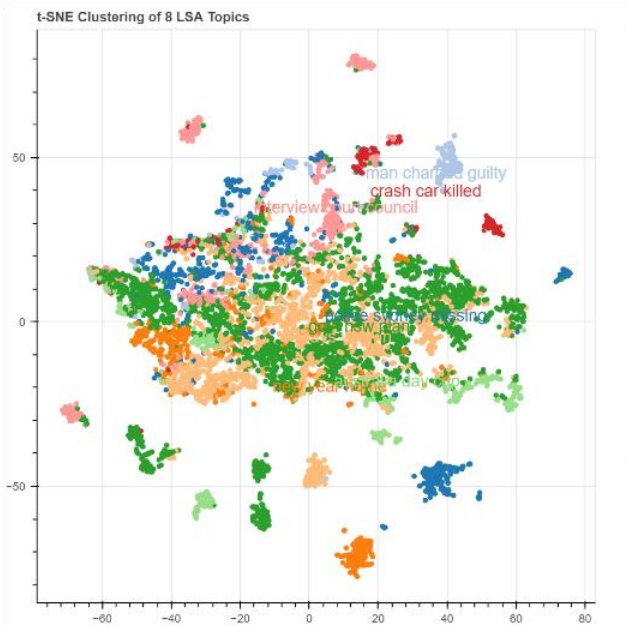



**Figure 6:** LSA TSNE g

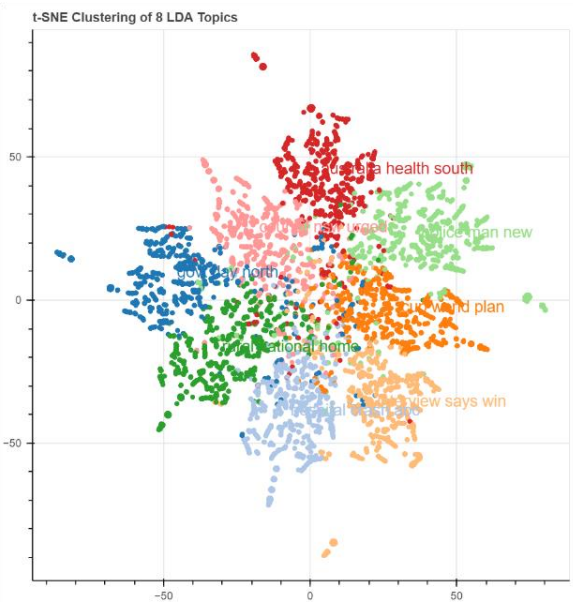



**Figure 7:** LDA TSNE

| Topics | Top 6 words | Num headlines |
|---|---|---|
| 1 | police sydney death missing adelaide search | 960 |
| 2 | man charged guilty sex jailed child | 1228 |
| 3 | new help year home deal indigenous | 2788 |
| 4 | says health minister sa election workers | 2137 |
| 5 | govt nsw water qld plan rural | 987 |
| 6 | australia day cup world coronavirus south | 1026 |
| 7 | crash car dies killed hospital road | 442 |
| 8 | interview council court australian report trial | 432 |

**Table 3:** LSA Table

| Topics | Top 6 words | Num headlines |
|---|---|---|
| 1 | govt day north sydney market nt | 1230 |
| 2 | hospital crash abc port trump budget | 1334 |
| 3 | cup world plan coast gold centre | 1211 |
| 4 | interview says win iraq minister claims | 1190 |
| 5 | rural national home school missing farmers | 1480 |
| 6 | police man new crash charged car | 1215 |
| 7 | australia health south qld water court | 1210 |
| 8 | council nsw urged workers government tax | 1130 |

**Table 4:** LDA Table

| **Method** | **TC** | **TD** |
|---|---|---|

| | | |
|---|---|---|
| **LSA** | 0.4083 | 0.9375 |
| **LDA** | 0.3130 | 0.9687 |

## 4.3 Deep Topic Modeling

### *4.3.1 Introduction*

This section investigates the application of deep learning methods in topic modelling. In the following, the three steps of embeddings, dimension reduction and clustering are compared, to see the difference between different deep learning methods. The optimized topic modeling model is obtained through combinatorial analysis, the topic representation is also modelled. Only 100,000 pieces of data are used in this part due to the consideration of computing efficiency.

### *4.3.2 deep sentence embeddings*

The first step of deep Topic modelling is to transform the headings into vector of embeddings, we have selected different embedding methods such as sentence transformer, clip, etc. for performance comparison. In order to get comprehensive information, we performed topic modelling on all models in a uniform configuration and compared the obtained two metrics TC, TD. Small scale clustering k means clustering, LSA dimensionality reduction methods), large scale clustering, large scale clustering (HDNSCAN cluster with hierarchy structure and UMAP dimensionality reduction methods) were tested respectively. In these two classical configurations, topic modelling is performed, the obtained TC, TD metrics are compared, and the basic embedding model is determined through a comprehensive analysis.

We have tested 8 category of embeddings.

| Type of model | Description | Original task |
|---|---|---|
| **Sota sentence transformer** (all-MiniLM-L6-v2) | A variant of BERT model which has 6 transformer layers | have been trained on a large corpus of text data |
| **Multilingual transformer** (paraphrase-multilingual-mpnet-base-v2) | multilingual variant of the MobileBERT Pretraining model | specifically designed for paraphrase detection, to capture semantic similarity between sentences across multiple languages. |
| **Small sentence transformer** (paraphrase-albert-small-v2) | a variant of the ALBERT (A Lite BERT) model | specifically for paraphrase detection |
| **Semantic search** (multi-qa-mpnet-base-dot-v1) | a variant of the MPNet (MobileBERT Pretraining) | multi-turn question-answering tasks |
| **Bing search** (msmarco-bert-base-dot-v5) | a variant of the BERT (Bidirectional Encoder Representations from Transformers) model that has been fine-tuned on the MS MARCO dataset | machine reading comprehension and question-answering tasks |
| **Clip** (clip-ViT-L-14) | a variant of the Contrastive Language-Image Pretraining | associate images and their textual descriptions by jointly |

| | | |
|---|---|---|
| | (CLIP) model that utilizes the Vision Transformer (ViT) architecture | training on large-scale image-text pairs |
| **Google questions** (nq-distilbert-base-v1) | a variant of the DistilBERT model that has been trained specifically for the Natural Questions (NQ) dataset | designed for question-answering tasks |
| **Meta questions** (facebook-dpr-ctx_encoder-multiset-base) | part of the Dense Passage Retrieval (DPR) framework developed by Facebook AI Research | designed for large-scale passage retrieval tasks and question-answering systems |

**Table 5:** Embedding methods

#### *4.3.3 Experiment results*

| | Small Clustering (8) | | Large Clustering (>800) | |
|---|---|---|---|---|
| | TC | TD | TC | TD |
| sota_sentence_transformer | 0.3764 | 0.85 | 0.3866 | 0.856 |
| small_sentence_transformer | 0.3917 | 0.825 | 0.3896 | 0.8615 |
| multilingual_transformer | 0.3755 | **0.8875** | 0.3762 | 0.8384 |
| semantic_search | 0.3626 | 0.875 | 0.3856 | 0.8509 |
| bing_search | **0.4143** | 0.8375 | **0.4088** | 0.8583 |
| google_questions | 0.3317 | 0.775 | 0.3725 | **0.8798** |
| meta_questions | 0.3573 | 0.6875 | 0.398 | 0.8779 |
| clip | 0.3655 | 0.725 | 0.3968 | 0.8484 |
| tf-idf | 0.4512 | 0.775 | 0.4398 | 0.8791 |

**Table 6:** Performance of different methods

It seems that the **Bing search** embedding is the best among all the methods, while multilingual transformer and google question also generate a descent result.

It is noticeable that, all the **deep learning models** tested in regular configuration perform worse than basic embedding methods such as **tf-idf** or even **word count**. This is due to the fact that the text used for training by the deep embedding method has distributional differences from our news headline data. For news headlines, the content is often very short and highly refined, and some keywords have a huge impact on the semantics. The representation and general text have considerable differences in both content and form. In this way, the deep embedding model trained based on general text may not be more effective than traditional methods without finetune. However, our analysis is still based on deep learning methods, because if the embedding is re-trained on a new dataset by methods such as finetune, the performance often exceeds that of the traditional method. (Therefore, the subsequent model selection should not only consider the current performance, but also analyses whether it is suitable as a base model for finetune).

#### *4.3.4 Visualize diagrams*

We represent the topics as vector embeddings, perform similarity calculation among different topics to generate heat map, then evaluate the degree of diversity of topic modelling by visualizing the similarity of different topics.

The darker the color, the higher the similarity between the topics and the worse the diversity of topic modelling.

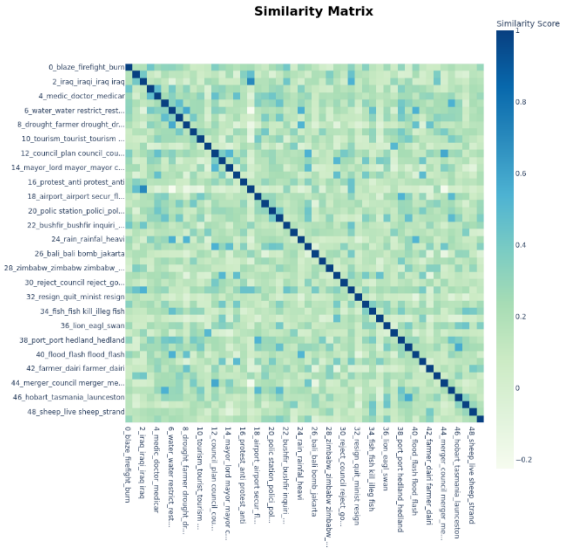



**Figure 8：** small sentence transformer

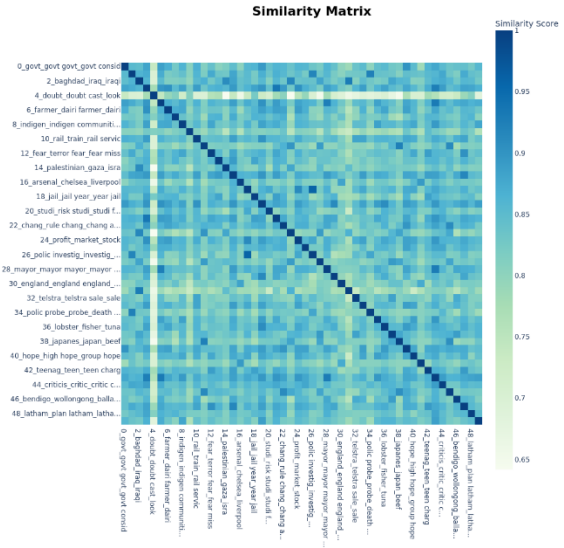



**Figure 9:** meta question transformer

We represent TOPIC as a weighted form of words, and can observe TOP content through TOP word scores. There is a clear thematic distinction between different topics. However, problem still exists, the repetitions in top words: for example, "rail" and "rail line" with similar semantics, "Iraq" and "Iraqi" with different forms but the same semantics".

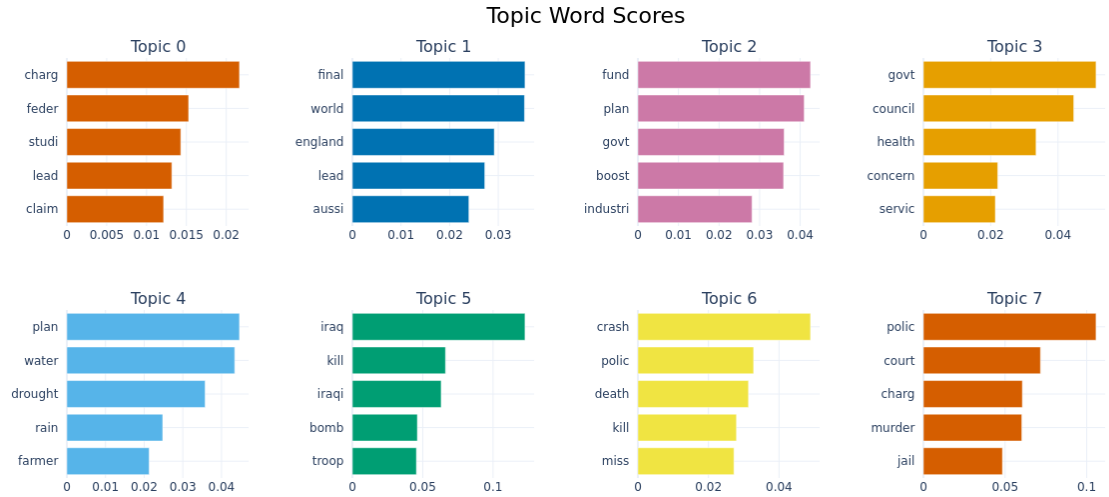



**Figure 10：** top 5 topic words scores for small transformer

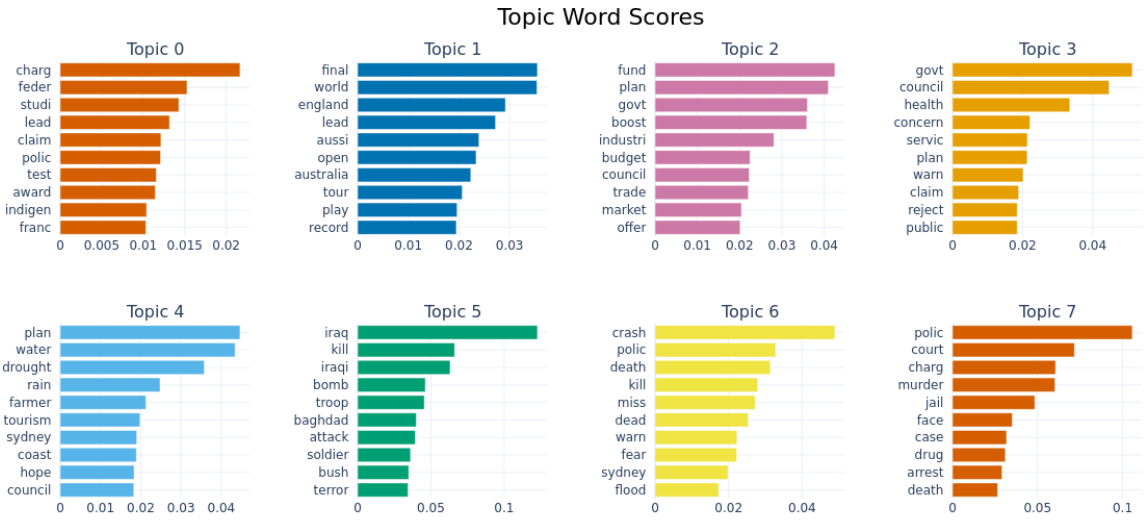



**Figure 11：** top 10 topic words scores for small transformer

We also found that the categories extracted by different embeddings have similarity, indicating that the topic modelling approach has reliability to some extent.

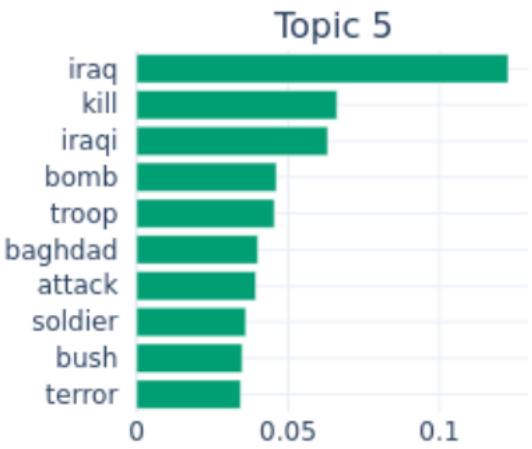



**Figure 12：** Small Sentence transformer

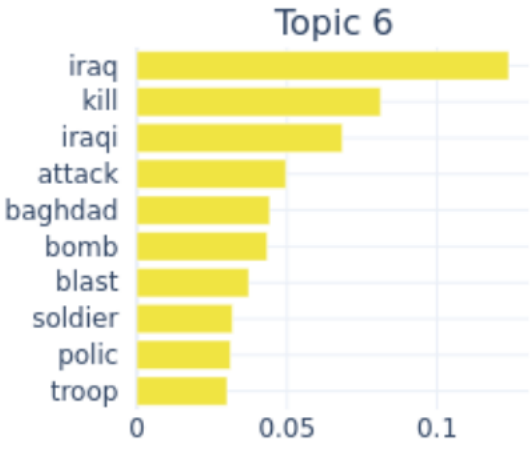



**Figure 13：** Google Questions

We can also use word cloud to count the TOPIC CLUSTER content. Observe that the clusters obtained under different embeddings have similarity to some extent.

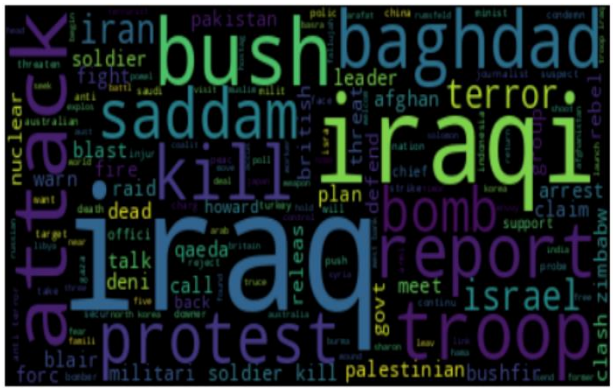



**Figure 14：** Small Sentence transformer

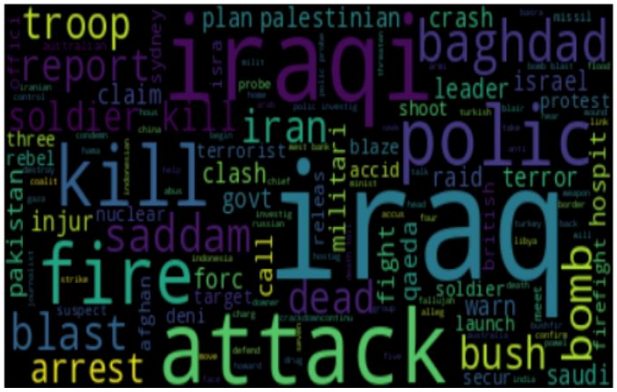



**Figure 15：** Google Question

In order to get a more visualizable representation of the theme of topic. We let GPT4 summarize the top-30 keywords of different topics into a descriptive text, and then use the Stable Diffusion model for image generation. The images generated for the two topics have high similarity.

**Figure 16:** Small Sentence transformer.

**Figure 17：** Google Question

***4.3.5 Model selection***

Selected：Small sentence transformer。

After consideration, we finally chose Small sentence transformer as the base model for subsequent analysis. On the one hand, it shows relatively good performance in both small clustering and large clustering configurations, the model structure also has some generality. On the other hand, it has a small number of parameters, which makes it less costly to perform further finetune calculations. Even with unsupervised training using masks, the amount of data we have is not that large, the use of overly complex models can easily lead to overfitting and catastrophic forgetting.

| | Model name | Model Size | TC small | TC large | TD small | TD large | Performance |
|---|---|---|---|---|---|---|---|
| multilingual transformer | paraphrase-multilingual-mpnet-base-v2 | 970 MB | 0.3755 | 0.3762 | **0.8875** | 0.8384 | 65.83 |
| sota_sentence_transformer | all-mpnet-base-v2 | 420 MB | 0.3764 | 0.3866 | 0.85 | 0.856 | **69.57** |
| small_sentence_transformer | paraphrase-MiniLM-L3-v2 | **61 M** | **0.3917** | **0.3896** | 0.825 | **0.8615** | 62.29 |

**Table 7：** model comparison

Ps: performance refers to the performance of the model as sentence embedding tested on 14 datasets

### 4.4 Dimension Reduction methods

Methods Introduction. We test several dimensionality reduction methods using small sentence transformer embedding, 50 k means clustering. As it shows in the table below.

| | TC | TD |
|---|---|---|
| **LSA** | 0.3467 | 0.532 |
| **DictionaryLearning** | 0.3387 | 0.508 |
| **umap** | **0.3907** | **0.904** |
| **PCA** | 0.3341 | 0.59 |

**Table 8:** DR method comparision

**Dictionary Learning:** a technique used to represent data as a sparse linear combination of basic elements from a learned dictionary.

**Umap:** a dimensionality reduction technique commonly used for visualizing high-dimensional data in a lower-dimensional space.

Umap's dimensionality reduction method works much better than the others. This can also be seen in the heat map.

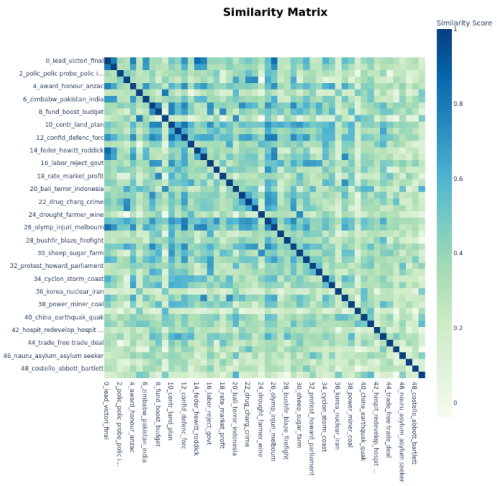



**Figure 18：** umap heat map

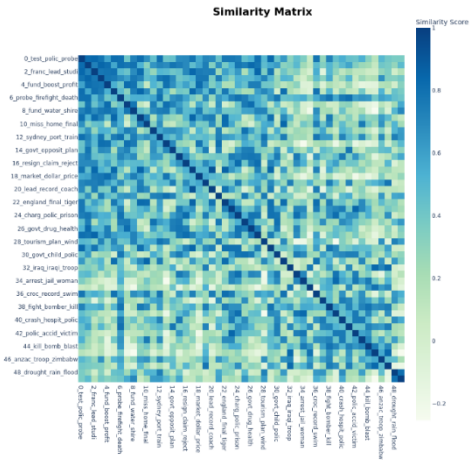



**Figure 19：** LSA heat map

### 4.5 Clustering methods

In this section we explore different clustering methods. The **small transformer embeddings** and **umap dimensionality reduction** are selected for testing different clustering models and different number of

clustering. We tested the effect of the number of categories on topic modelling for **HDBSCAN model** and **K means clustering**, respectively.

#### *4.5.1 Cluster methods*

**HDBSCAN (Hierarchical Density-Based Spatial Clustering of Applications with Noise)** is a density-based spatial clustering algorithm that automatically discovers the structure of the clusters in a dataset and does not require a predetermined number of clusters. While **K-means clustering** divides the sample points in the dataset into K different clusters, minimizes the distance of points in same cluster, enlarges the distance between different clusters. Among them, **HDBSCAN** can deal with noisy data and irregular shape clustering, also, with higher tolerance for outliers (but will exclude outlier, affecting a certain applicability). **K-means** is computational efficiency, can effectively deal with large-scale datasets, and performs quite well on regular shape of clusters. But it has less tolerance for noisy data and the irregular shape of clusters. Also, the tolerance for outliers is low.

#### *4.5.2 Number of clusters*

Even through unsupervised learning doesn't need labels, many topics modeling method need to determine the number of clusters. We use elbow function to judge what number is better at the beginning. elbow method is a commonly used technique to determine the optimal number of clusters in clustering algorithms. We use the embeddings to analyze within-cluster sum of squares (WCSS). By the Figure 6, 8 is a nice choice. We also compare the clustering effect with 50 and 800 topics in the following comparison. Besides, we also select top 8 topics from 50 clusters and 50 clusters from 800 clusters for more information because the difference between two clustering methods.

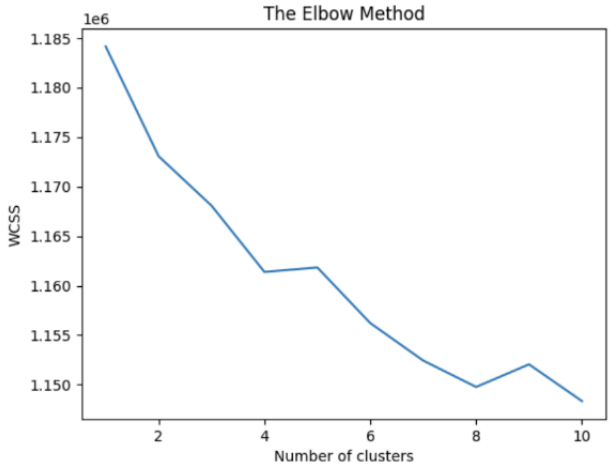



**Figure 20:** Elbow Method

#### *4.5.3 Experiment results*

In the following test for **TC metrics,** we find that selecting part of the topic content from a larger number of clustering has greater consistency for the same number of clusters. Also, **HDNSCAN** shows better results than **k means** on large-scale clustering. However, the **k means** method has relatively stable performance at different scales.

| | 8 | 50 to 8 | 50 | 800 to 50 | 800 |
|---|---|---|---|---|---|
| **k means** | 0.4127 | 0.4452 | 0.3758 | 0.4104 | 0.4248 |
| **HDNSCAN** | | | | 0.4034 | 0.4338 |

**Table 9:** Result of k-means and HDNSCAN

We also use **UMAP** to downscale the results to 2D for visual observation. We can see the difference in the distribution of results between different models, as well as different CLUSTER methods.

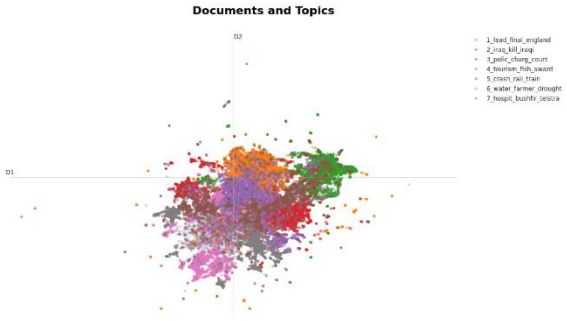



**Figure 21:** 8 k means cluster

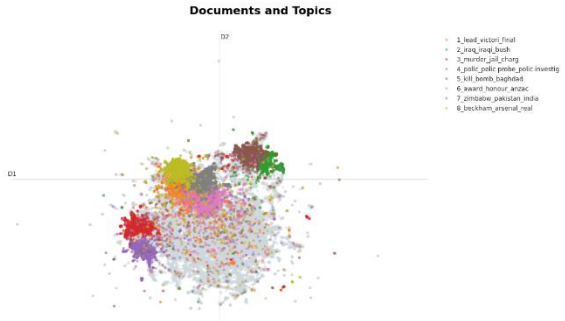



**Figure 22:** 50 to 8 k means cluster

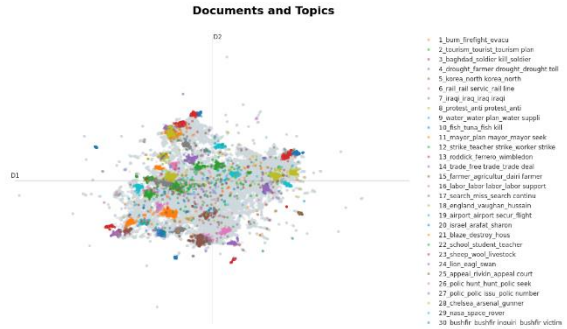



**Figure 23:** 800 to 50 k means cluster

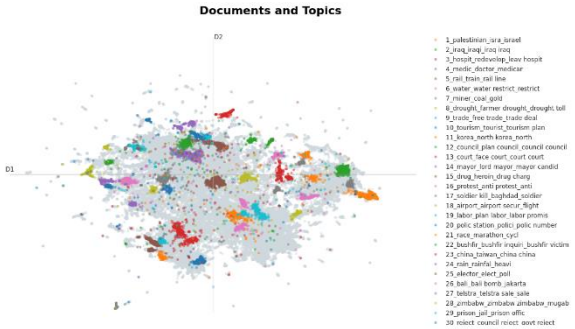



**Figure 24:** 800 to 50 HDBSCAN

### *4.5.4 Topic representation*

Each topic is a cluster of headings. We use the topic representation model to represent the cluster's content summarized into several keywords and weights.

Topics can be described using a series of words, starting with the most important word. Each word is assigned a c-TF-IDF score, which indicates how well it represents the topic. As more words are added to the description, the total c-TF-IDF score increases, but this increase becomes smaller as less relevant words are included.

To illustrate this, we can plot the c-TF-IDF scores for each topic against the rank of the word in the description. The words with the highest scores will have a rank of 1, and will be plotted on the x-axis. The y-axis will show the corresponding c-TF-IDF scores. This plot will show how the c-TF-IDF score decreases as less relevant words are added to the description. It can be used to determine the optimal number of words to include in a topic description using the elbow method.

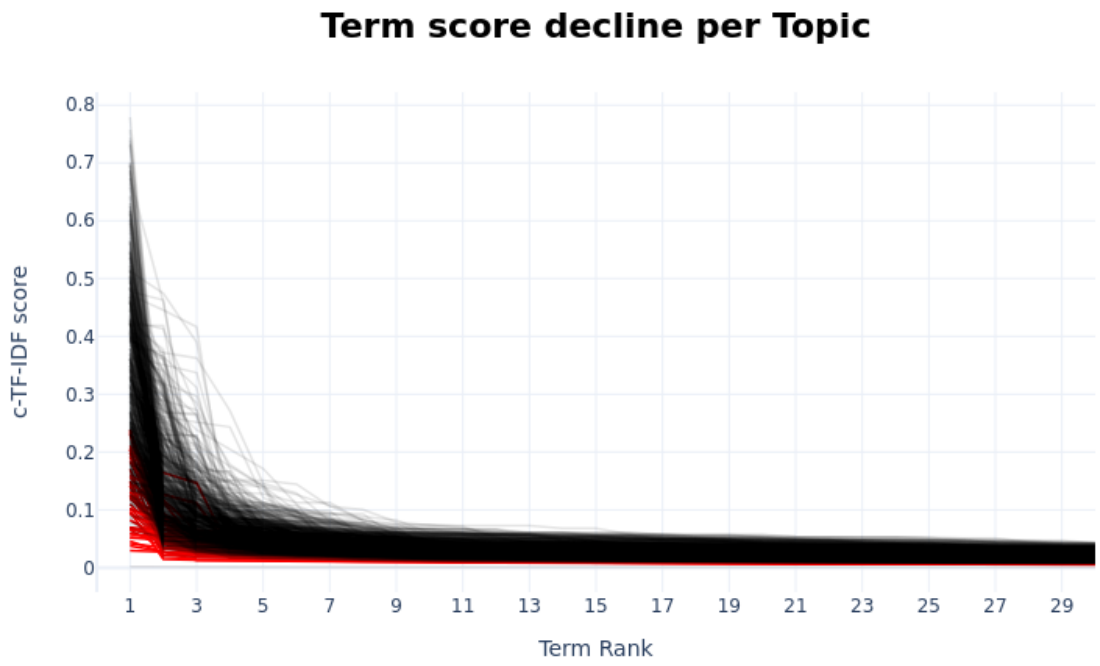



**Figure 25:** Term score decline per topic

It can be observe that most topics reach stability at about ten TERMs, so we ended up using 10 TERMs for each topic.

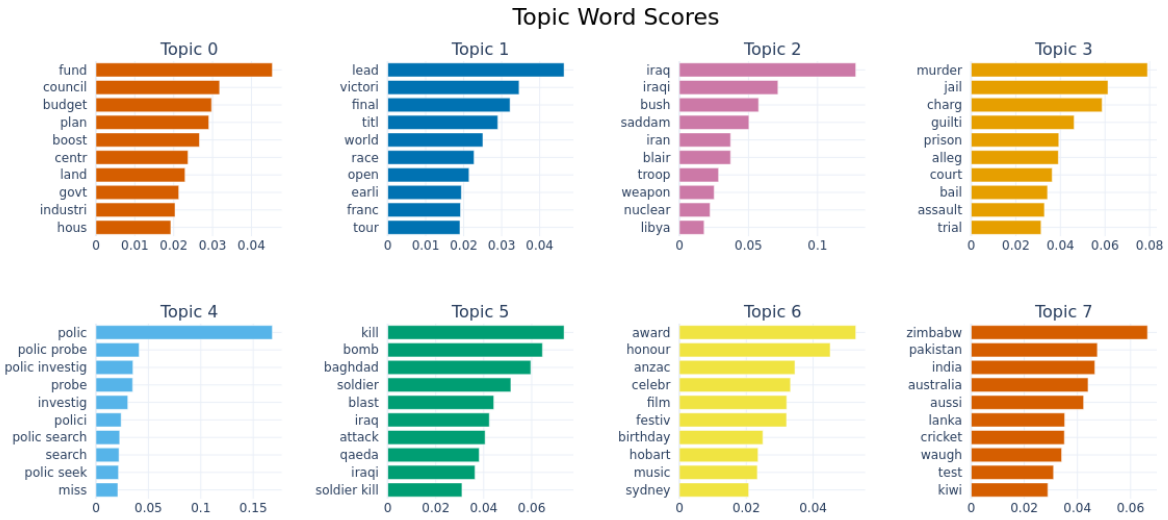



**Figure 26:** Topic word scores

For the relationships between classes, we visualize them through the hierarchical relationships during the clustering generation process

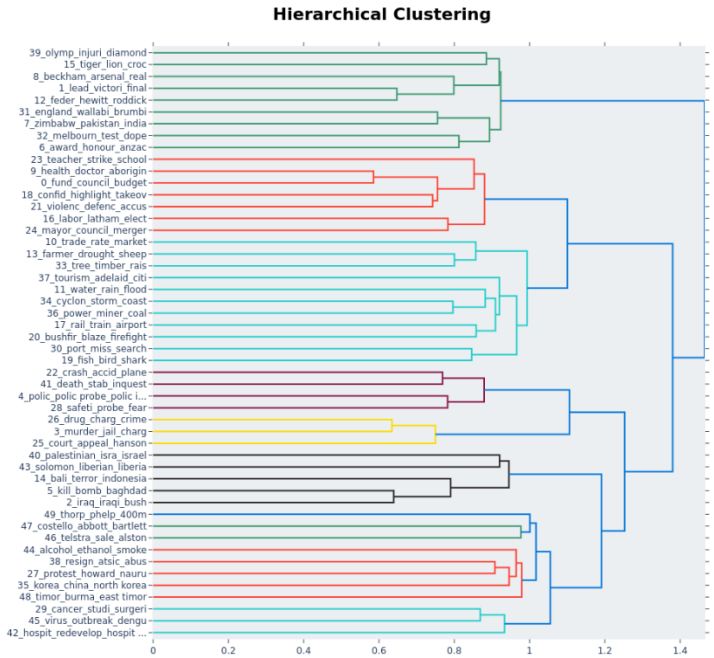



**Figure 30:** Hierachy relationship

In addition, the relationship between classes can be observe by the distance between centers

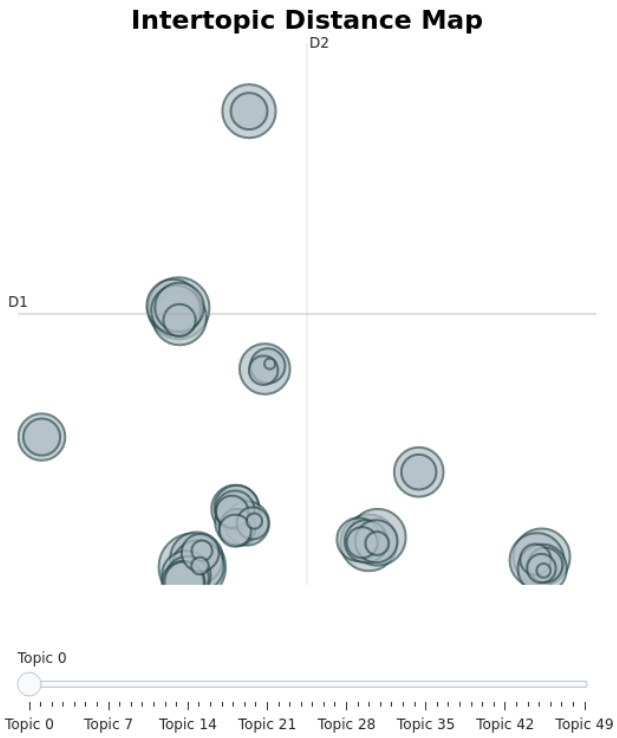



**Figure 31:** Intertopic Distance Ma

Then we also try to let chatgpt4 summarize the generated topics into a sentence, then use a diffusion model to generate an image to represent the topic. This can give us a more direct visualization of the topics.

We use the **Topic 0** as an example to describe how the images are generated, as shown below.


**Figure 32:** Topic 0

**Prompt generated by gpt 4:** Generate an image depicting a council meeting where the government is planning to fund a health boost. The mayor is present, discussing the rate of

services, labor trade, and the budget. The group is considering changes due to the rise in the central housing market. Union teachers are on strike, and the minister is rejecting their demands. The school is filled with indigenous workers. The market election is a major concern for the industrial sector.

**Figure 33:** Topic 1

**Figure 34:** Topic 2


**Figure 35:** Topic 3

**Figure 36:** Topic 4


**Figure 37:** Topic 5

**Figure 38:** Topic 6


**Figure 39:** Topic 7


**Figure 32:** Topic 0

## 5. Topic Classification

Since the dataset is not labeled, it is not possible to train a model on this dataset. We will focus on how to perform classification tasks on unlabeled datasets. The first idea is to directly apply a pre-trained classification model. Although the dataset used to train the model is different from the distribution of the new dataset, it is still relevant. Second, we use the pre-trained embedding model and compare the title embedding and category embedding to get the category based on the distance. Third, the topics extracted by topic modeling are used as attributes for zero shot

learning. Fourth, a special training strategy is adopted to perform semi-supervised learning and unsupervised learning for the classification model in the case of a small number of labels or no labels. Due to limited time, the first two methods are mainly explored in the following. There are three core goals, first, the attempt of task transfer, domain transfer on unlabeled data. Second, we discuss how to use large-scale unlabeled data sets for model training to improve model performance. Third, how to annotate the dataset with relatively high accuracy with a small amount of human labor.

## 5.1 data collection and preprocessing

Nowadays, large language model (LLM) is widely used for many different topics. Natural Language Processing (NLP) is one of the closest tasks to LLM, so using LLM is also a good idea. For unlabeled data, LLM also can help to label data and use the resulting dataset for supervised learning tasks. Sometime even if the LLM isn't label well, we can modify them by hand and fine-tuning the result again, so that the LLM can get a larger and better dataset.

In this project, we also try using GPT to help us label the data. Considering the cost, we randomly selected 1,000 of the first 100,000 pieces of data to be labelled. We use GPT-4 because it is the newest one. We also ask GPT to give us a confidence score about how confidence GPT thinks the answer is correct. The prompt is as follows:

> Here is a news headline: {headline}, please judge which topic of news it belongs to (World, Sports, Business, Sci/Tech and other), and a decimal from 0 to 1 to show how sure you are of this answer (eg: 0.4).The example of the output format is 'World, 0.8'. Please do not output anything else.

This is the samples for the results. For **case study**, GPT is 100 percent sure 'capello quits as englands manager' is sports news, which supervising us. We don't know that Capello is a coach, and labelling it manually by us isn't necessarily better than labelling it by GPT. We consider the GPT judgement as the correct result and will use it in our following classification task.

| | publish_date | headline_text | GPT_categ | GPT_confidence_score |
|---|---|---|---|---|
| 663090 | 20120102 | christchurch mayor bob parker speaks about nz | World | 0.9 |
| 671156 | 20120209 | capello quits as englands manager | Sports | 1 |
| 563459 | 20100915 | robotic dairy calms cows but gives farmer nightmare | Sci/Tech | 0.9 |
| 487170 | 20090909 | crime stats deliver mixed results | World | 0.7 |
| 239218 | 20060526 | grim winter rain outlook for eastern australia | World | 0.9 |
| 1083129 | 20170524 | firefighters concerned about new cancer legislation | Other | 0.9 |
| 816489 | 20130904 | media call brad haddin | Sports | 0.9 |

**Table 10:** Samples for GPT results

We only selected the data with GPT confidence higher than 0.9 as real samples, and then made a relative balance between different categories. Although there is some bias in the labeling of this data, there is usually some noisy data in the data, which is inevitable in the normal course of collecting data, and you only need a rough standard for comparing the accuracy of different models. Through manual partial inspection of the data, it was found that the annotations of GPT were basically without much error.

We used GPT4 to label 1000 samples, and after data cleaning and class balancing, we left 600 samples to form a test dataset to evaluate the performance of the model.

For the convenience of evaluation, we also delete the class “Others” which containing the news that does not match with any of the four classes.

| | World | Sports | Business | Sci/Tech | Others | Overall |
|---|---|---|---|---|---|---|
| **Original** | 517 | 222 | 137 | 49 | 74 | 1000 |
| **After Filtering** | 244 | 206 | 116 | 34 | 0 | 600 |

**Table 11:** Samples fall in different clusters

## 5.2 base text classification model

### *5.2.1 XLNet*

- **Overall**

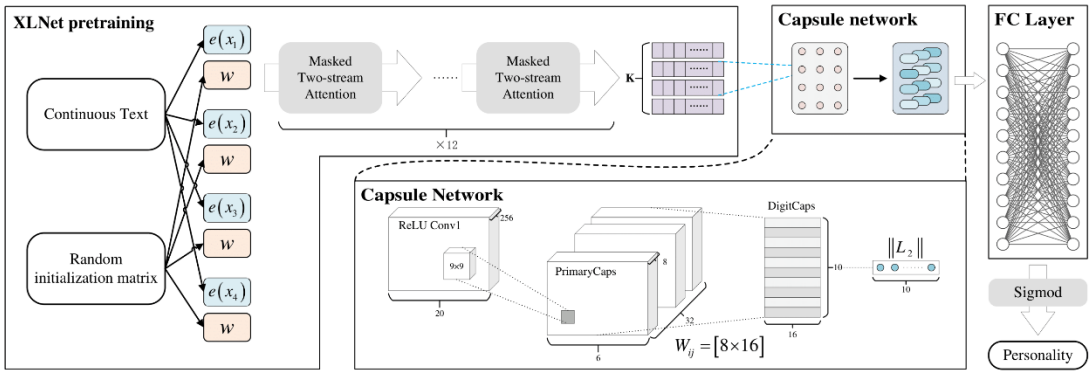



**Figure 40:** XLNet structure

XLNet is a kind of autoregressive model. Empirically, XLNet outperforms BERT on various language understanding tasks, including question answering, natural language inference, sentiment analysis, and document ranking.

- **Loglikelihood**

Unlike Bert which is built directly based on auto-encoder, XLNet is built based on permutation language modeling. Which means to considering all possible permutations of the factorization order and maximize the loglikelihood of a position to be a specific word. By doing so, it takes the context of bidirectional into consideration.

$$\max_{\theta} \quad \mathbb{E}_{\mathbf{z}\sim\mathcal{Z}_T}\left[\sum_{t=1}^{T}\log p_\theta\left(x_{z_t} \mid \mathbf{x}_{\mathbf{z}<t}\right)\right]$$

**Figure 41:** Loglikelyhood optimization

- **Two stream self-attention**

The XLNet also applied two stream self attention, which is a variant of the original one. It can capture different types of dependencies within a sequence.  It divides the self attention into two separate streams, each with a distinct purpose. For the case of XLNet, one of the stream can see self, but the ither cannot.

- **Training**

We trained this XLNet model based on a dataset names AG News, the AGnews20 dataset is compound of 4 classes of news: World (1), Sports (2), Business (3), Sci/Tech (4). The accuracy of the XLnet model for unseen news is around 83%. Which is quite decent. The training loss is shown in the figure below. We can see it converges fast. To overcome the problem of overfitting, we apply the early stopping strategy, if after 5 epochs, there is no improvement compared to the best result on validation set, we will stop our training.

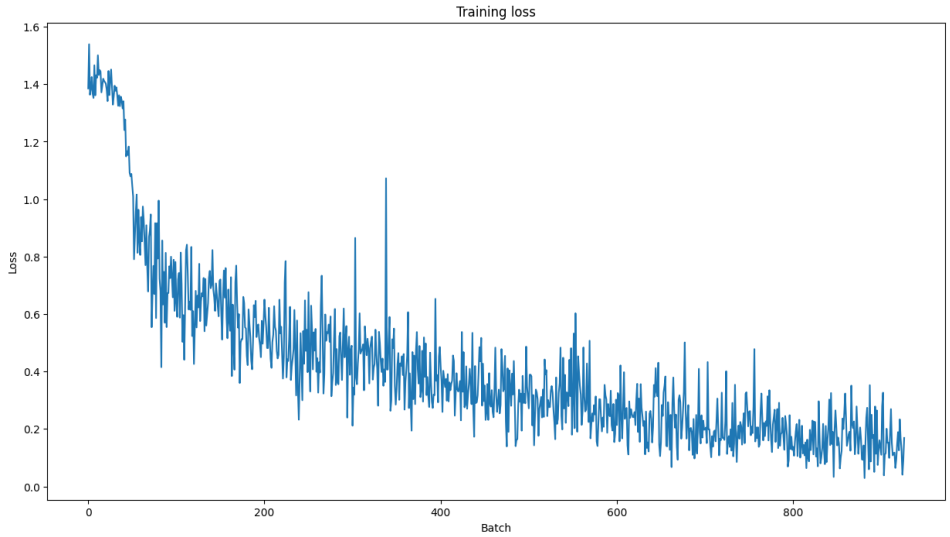



**Figure 42:** Loss of XLnet training

- **Evaluation**

Then we input the millions of headlines dataset to the trained Xlnet model to generate the classification result. To evaluate the performance of the performance, we directly apply the classification generated by chatgpt4 as ground truth, which is quite a decent model (Maybe some minor mistake, which can be tolerated).

## 5.3 basic embedding distance method

In this part, we use the pre-trained embedding model for classification. The core idea is to generate a vector embedding for each class, and then generate the embedding of the title data, which is used as the query to find the nearest category embedding, and the category to which it belongs is used as the final classification result. The assumption here is that the pre-trained embedding model can make a generalization of the semantics and be discriminative in the vector space. Texts with similar semantics are mapped to close ones. But here, we need to generate a prompt for the category first. Through experiments, the way to generate the category prompt has a great impact on the classification effect.

First, the category labels are used as prompt to generate category embeddings.

Class prompt 1: {label} Query: {data}
Example: class 1: World class prompt: "World". Query: "cracked tanker enters sydney harbour"

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 1 | 0.10 | 0.01 | 0.02 | 244 |
| 2 | 0.93 | 0.20 | 0.33 | 206 |
| 3 | 0.51 | 0.23 | 0.32 | 116 |
| 4 | 0.07 | 1.00 | 0.13 | 34 |
| accuracy | | | 0.17 | 600 |
| macro avg | 0.40 | 0.36 | 0.20 | 600 |
| weighted avg | 0.46 | 0.17 | 0.19 | 600 |

**Table 12:** classification performance

The problem that eventually arises is that the classification effect is not balanced across classes. The overall effect is not high.

Class prompt 2: A news in the topic of {label}
Example： class 1: World → class prompt: “A news in the topic of World”

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 1 | 0.67 | 0.16 | 0.25 | 244 |
| 2 | 0.93 | 0.34 | 0.50 | 206 |
| 3 | 0.58 | 0.19 | 0.29 | 116 |
| 4 | 0.08 | 1.00 | 0.15 | 34 |
| accuracy | | | 0.27 | 600 |
| macro avg | 0.56 | 0.42 | 0.30 | 600 |
| weighted avg | 0.71 | 0.27 | 0.34 | 600 |

**Table 13:** classification performance

Even though the imbalance problem still exists, the classification effect has been greatly improved.

### 5.4 experimental results

| | F1-score | Recall | accuracy |
|---|---|---|---|
| XlNet | 0.7595 | 0.7533 | 0.7533 |
| Word embedding | 0.19 | 0.17 | 0.46 |
| Sentence embedding | 0.34 | 0.27 | 0.71 |

**Table 14:** Experiment result

For the pre-trained classification model, it faces the task transfer problem with different training tasks and test tasks, and the domain transfer problem with different data distribution. For unsupervised methods, it is necessary to face the problem of lack of data annotation. In terms of accuracy, basic unsupervised methods can sometimes reach the accuracy of classification models. But in comparison, the training and application costs are much lower than supervised methods.

## 6.Sentiment Analysis and Stock Price Prediction

**Table 15:** Samples for sentiment score

| headline_cleaned_text | senti_score | compound | senti_label |
|---|---|---|---|
| aba decid against communiti broadcast licenc | {'neg': 0.0, 'neu': 1.0, 'pos': 0.0, 'compound... | 0.0000 | neutral |
| act fire wit must be awar of defam | {'neg': 0.255, 'neu': 0.745, 'pos': 0.0, 'comp... | -0.3400 | negative |
| a g call for infrastructur protect summit | {'neg': 0.0, 'neu': 0.606, 'pos': 0.394, 'comp... | 0.3818 | positive |
| air nz staff in aust strike for pay rise | {'neg': 0.293, 'neu': 0.707, 'pos': 0.0, 'comp... | -0.2263 | negative |

**Table 16:** Count for sentiment

| positive_count | negative_count | neutral_count | date |
|---|---|---|---|
| 30 | 72 | 96 | 2003-02-19 |
| 40 | 83 | 127 | 2003-02-20 |
| 47 | 86 | 117 | 2003-02-21 |
| 22 | 37 | 67 | 2003-02-22 |
| 22 | 49 | 65 | 2003-02-23 |

Sentiment analysis is also a common task for natural language processing (NLP). It is also widely studied because public emotions and opinions about specific events can affect the stock movement more profoundly than events themselves. For example, Bag of Words method can be used to analyze the sentiments of different headlines. 'nltk.corpus' has some built-in corpus about sentiment and we can use this to go over the sentiment data of the text. In our try, we use 'SentimentIntensityAnalyzer' to generate sentiment polarity. 'compound' is an emotional score between -1 and 1.

If 'compound' is larger then 0.05, we consider the headline is 'positive'; if less than -0.05, 'negative' hea dline is got. There are 664, 911 neutral news, 372, 254 negative news and 207,019 positive news. Mean while, we also count different situations of sentiments in different days. The samples listed in **Table 16.**

The count for sentiments over time is as follows. Neutral emotions are always the most. This may be because news is relatively objective and less likely to contain a lot of emotion:

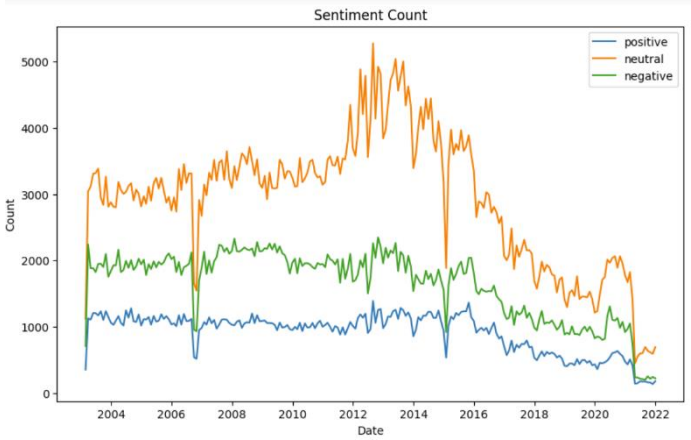



**Figure 43:** Sentiment Count every month

We also want to analyze the relationship between sentiments and stock movement. Because the news is not focused on a specific company, but rather on the world, we chose the overall indices like SP500 (a stock market index that measures the performance of 500 large publicly traded companies listed on stock exchanges in the US), NASDAQ Composite (an index that represents the performance of all the common

stocks and similar securities listed on the NASDAQ stock market) and S&P/ASX 200 (an index that represents the performance of the top 200 companies listed on the Australian Securities Exchange (ASX)).

### 6.1 Simplest Data Mining Method

We add up the compounds for each day's news, and if they are greater than or equal to 0, we consider that day's sentiment bias to be positive and mark it as 1. If it is less than 0, we mark it as –1.

| date | compound | senti_label |
|---|---|---|
| 2003-02-19 | -19.0758 | -1 |
| 2003-02-20 | -23.2749 | -1 |
| 2003-02-21 | -19.7626 | -1 |
| 2003-02-22 | -9.0097 | -1 |
| 2003-02-23 | -15.7253 | -1 |

**Table 17:** Compound and sentiment label

| | |
|---|---|
| one-day lag for S&P/ASX 200 | 0.4666106677866443 |
| three-day lag for S&P/ASX 200 | 0.4435111297774045 |
| five-day lag for S&P/ASX 200 | 0.43490130197396054 |
| one-month lag for S&P/ASX 200 | 0.36497270054598907 |
| one-day lag for SP500 | 0.45468245425188375 |
| three-day lag for SP500 | 0.4219590958019376 |
| five-day lag for SP500 | 0.4129171151776103 |
| one-month lag for SP500 | 0.33627556512378903 |
| one-day lag for NASDAQ Composite | 0.4486544671689989 |
| three-day lag for NASDAQ Composite | 0.4243272335844995 |
| five-day lag for NASDAQ Composite | 0.4139935414424112 |
| one-month lag for NASDAQ Composite | 0.33541442411194833 |

**Table 18:** Accuracy list 1

We have also taken into account the lag effect of the news and the fact that the news may not have any impact until a few days after its publication. We consider different scenarios for one day, 3 days, 5 days and 30 days. If return is larger or equal to 0, label 1, else label 0. However, the result is not so good. All results are less than 0.5, meaning that the effect is weaker than random guessing. This method is too rough, let's try machine learning method to show the importance of machine learning.

### 6.2 Machine Learning Methods

| date | positive_count | negative_count | neutral_count | compound |
|---|---|---|---|---|
| 2003-02-19 | 0.151515 | 0.363636 | 0.484848 | -19.0758 |
| 2003-02-20 | 0.160000 | 0.332000 | 0.508000 | -23.2749 |
| 2003-02-21 | 0.188000 | 0.344000 | 0.468000 | -19.7626 |
| 2003-02-22 | 0.174603 | 0.293651 | 0.531746 | -9.0097 |
| 2003-02-23 | 0.161765 | 0.360294 | 0.477941 | -15.7253 |

**Table 19:** Training dataset

Features engineering is very important to traditional machine learning, so we should build features at first. Based on Sentiment, we get 4 features (‘positive_count’, ‘negative_count’, ‘neutral_count ’ and ‘compound’) every day. Because the number of news varies from day to day, so we divide the count by

the number of news items in the day for normalization. We use the first 100,000 news items (the data from 2003-02-19~2004-06-30) for training and testing.

We have trained three traditional machine learning methods, XGBoost, Logistic Regression, Extra Tree. No matter what methods we choose, Most of the accuracies are large than 0.5 and simplest data mining method. This demonstrates the superiority of machine learning. The different methods have their own merits, but overall the best ones are Logistic Regression. The most accurate prediction for S&P /ASX 200 as far as stocks are concerned. This may be because all the news is from the Australia media. The highest accuracy has increased to 0.8, which is a huge improvement. Meanwhile, Long-term predictions are generally more accurate than short-term predictions, contrary to the results of the simplest data mining method.

| | S&P/ASX 200 | | | | SP500 | | | | NASDAQ Composite | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Lag1 | Lag3 | Lag5 | Lag30 | Lag1 | Lag3 | Lag5 | Lag30 | Lag1 | Lag3 | Lag5 | Lag30 |
| XGBoost | 0.51 | **0.65** | **0.65** | 0.74 | 0.43 | 0.53 | 0.53 | 0.63 | **0.52** | 0.50 | 0.60 | **0.75** |
| Logistic Regression | **0.59** | 0.62 | 0.63 | **0.81** | **0.53** | **0.62** | **0.59** | **0.72** | 0.49 | **0.54** | **0.63** | 0.74 |
| ExtraTree | 0.54 | 0.59 | 0.59 | 0.76 | 0.47 | 0.50 | 0.49 | 0.69 | 0.47 | **0.54** | 0.56 | 0.74 |

**Table 20:** Accuracies for stock price prediction based on sentiments

We also consider adding more features to take a try. For our topic modeling part, we have judged what topics the headlines are. We use 'google question' for topic modeling to get more features. We want to use the results to get more features. We first determined the topics of the first 100,000 headlines, then we count the number of different topics everyday. Same as sentiment score, divide the count by the number of news items in a day for normalization.

| date | count_topic_1 | count_topic_2 | count_topic_3 | count_topic_4 | count_topic_5 | count_topic_6 | count_topic_7 | count_topic_8 |
|---|---|---|---|---|---|---|---|---|
| 2003-02-19 | 0.161616 | 0.191919 | 0.131313 | 0.101010 | 0.095960 | 0.106061 | 0.111111 | 0.101010 |
| 2003-02-20 | 0.136000 | 0.140000 | 0.148000 | 0.100000 | 0.104000 | 0.136000 | 0.084000 | 0.152000 |
| 2003-02-21 | 0.140000 | 0.204000 | 0.112000 | 0.120000 | 0.132000 | 0.092000 | 0.044000 | 0.156000 |
| 2003-02-22 | 0.230159 | 0.079365 | 0.134921 | 0.063492 | 0.087302 | 0.142857 | 0.119048 | 0.142857 |
| 2003-02-23 | 0.213235 | 0.088235 | 0.117647 | 0.066176 | 0.073529 | 0.161765 | 0.095588 | 0.183824 |

**Table 21:** Training dataset 2

We use XGBoost to make a comparison to compare the effects of different features. Overall, only sentiment's features situation is the best. But not a group of features is always the best. Sometimes sentiment and topic features can lead to a better accuracy.

| | S&P/ASX 200 | | | | SP500 | | | | NASDAQ Composite | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Lag1 | Lag3 | Lag5 | Lag30 | Lag1 | Lag3 | Lag5 | Lag30 | Lag1 | Lag3 | Lag5 | Lag30 |
| sentiment | 0.51 | **0.65** | **0.65** | 0.74 | 0.43 | 0.53 | 0.53 | 0.63 | **0.52** | 0.50 | **0.60** | **0.75** |
| topic | 0.44 | 0.60 | 0.54 | 0.76 | **0.47** | 0.47 | **0.62** | **0.68** | 0.49 | **0.56** | 0.47 | 0.71 |
| Sentiment+ | **0.57** | 0.54 | 0.57 | **0.79** | 0.46 | **0.56** | 0.57 | 0.65 | 0.50 | 0.50 | 0.51 | 0.70 |

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| topic | | | | | | | | | | | | |

**Table 22:** Accuracies for stock price prediction based on sentiments + topics

We also want to use embedding data to take a try. We combine the daily news into one text. There are 384 values for each embedding if using 'SentenceTransformer'. We try Logistic Regression and XGBoost to do predictions. Interestingly, embeddings features led to both worse and better results. The most accurate is S &P /ASX 200 at lag 30.

| | S&P/ASX 200 | | | | SP500 | | | | NASDAQ Composite | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Lag1 | Lag3 | Lag5 | Lag30 | Lag1 | Lag3 | Lag5 | Lag30 | Lag1 | Lag3 | Lag5 | Lag30 |
| XGBoost | 0.53 | 0.56 | 0.57 | **0.82** | **0.54** | 0.47 | **0.59** | **0.74** | 0.50 | 0.41 | 0.56 | 0.69 |
| Logistic Regression | **0.56** | **0.62** | **0.63** | 0.81 | 0.46 | **0.56** | 0.53 | 0.72 | **0.51** | **0.5** | **0.60** | **0.73** |

**Table 23:** Accuracies for stock price prediction based on embedding+ machine learning

We just use the original traditional machine learnings and don't do any fine-tuning. Fine tuning, deep learning or considering multiple days of news for the sake of time dependency may lead to better results. We didn't try the next step because of time constraints.

## 7. Conclusion

This project applied different embedding methods on the dataset, including traditional embedding methods such as ti-idf as well as deep learning methods such as sentence transformer. Then the unsupervised dimension reduction and clustering attempts are carried out. By combining embedding method, dimension reduction and clustering, the topic representation learning is carried out. We conduct topic modeling task, which is to extract continuous and diverse topics from the data, and optimize the performance by combinatorial analysis. Then, we use the GPT4 labeled data as the test set, try to perform classification tasks on the data set, and test the classification model and embedding distance. We use the sentiment extraction model to analyze the sentiment polarity of news. Finally, the results of sentiment analysis and topic modeling are used to train a new model to perform the stock prediction task.

Our project extends from the attempt of basic methods such as dimensionality reduction to the basic task of topic modeling, and finally applies to the actual scene of stock prediction, reflecting the inheritance relationship between tasks and models, which has a certain breadth. From the perspective of method, we discuss the application of models such as task transfer and domain transfer. Firstly, the differences among supervised learning, semi-supervised learning and unsupervised learning are considered. Thinking about different ways of learning like zero shot.

## 8.Further work

The project ended up with many limitations due to time and engineering constraints.

In **topic modeling**, due to the differences in tasks, the embedding method of deep learning model does not reflect the superiority, and does not exceed the traditional methods such as tf-idf in performance. After that, the fintuning of embedding model can be further performed. Unsupervised learning is performed on the dataset by using masks and other methods to improve the embedding ability.

In addition, we did not explore the differences between different topic representation learning methods, and the current single-model representation can be extended to multi-model representation, or large language models can be combined to generalize topics. In addition, topic modeling is currently represented by keyword weighting, and topic representation learning can also be tried to represent topic into vector embedding, which extends topic modeling to more downstream tasks.

In **topic classification**, only the most basic classification model methods and embedding methods are tested, but no optimization attempts are made. In addition, it can be further investigated to use the results of topic modeling as attributes for zero shot learning. Specifically, we label the distribution of topic categories according to the topic keyword, and then calculate the predicted class according to the distribution of query data in different topics. Alternatively, a new model can be used to learn the mapping relationship from topic representation to class.

In addition, we did not use the data of the dataset for model training. After that, we can try to design a model training strategy by combining manual labeling, human feedback and GPT labeling, and think about how to effectively use large-scale unlabeled data to improve the performance of classification models in the case of weak labeling. At the same time, we can complete the labeling of large-scale data sets with a small labor cost.

On the **stock prediction task,** we only selected the basic features and models. As input, we only use topic ratio and sentiment polarity as inputs, but these cannot make a good generalization of news semantics. Ideally, we should consider all the original news data and topic representation to learn a more complex relationship from news semantics to stock prediction. In terms of the model, we only chose the most basic model, and only made predictions through the news data of the day, without considering the time series and integrating historical information. After that, the model can be replaced with transformer, LSTM and other models for time series analysis.

## 9. Reference:


1. https://paperswithcode.com/dataset/ag-news
2. https://chat.openai.com
3. https://www.sbert.net/docs/pretrained_models.html
4. https://github.com/MaartenGr/BERTopic_evaluation
5. https://github.com/MaartenGr/BERTopic

6. https://colab.research.google.com/drive/16gx06PVffJwS4pRhysCmc5qbPm26vsY8

7. https://www.kaggle.com/code/rcushen/topic-modelling-with-lsa-and-lda

8. https://finance.yahoo.com/quote/%5ESPX/

9. Grootendorst M. BERTopic: Neural topic modeling with a class-based TF-IDF procedure[J]. arXiv preprint arXiv:2203.05794, 2022.

10. Yang Z, Dai Z, Yang Y, et al. Xlnet: Generalized autoregressive pretraining for language understanding[J]. Advances in neural information processing systems, 2019, 32.

11. Röder M, Both A, Hinneburg A. Exploring the space of topic coherence measures[C]//Proceedings of the eighth ACM international conference on Web search and data mining. 2015: 399-408.