Transformer-based Language Models for Text Clustering

Jianjun Yu[1]

[1]*Political science department, University of Iowa, Iowa City, IA, USA. Email: Jianjyu@uiowa.edu*

Abstract

Text clustering based on probabilistic topic models has gained significant attention among political scientists in recent years. However, commonly used probabilistic topic models like Latent Dirichlet Allocation and Structural Topic Models exhibit several limitations that hinder their application. Firstly, these models struggle to accurately classify short texts. Secondly, their performance heavily relies on user decisions regarding text preprocessing and the number of clusters selected. Lastly, model training with large text datasets can be time-consuming. In this paper, I propose a workflow that uses transformer-based language models, such as BERT and GPT, for text clustering. This method surpasses traditional topic models in terms of accuracy and time efficiency. Furthermore, it reduces the impact of user decisions on text preprocessing and facilitates easy comparison of different topic numbers' impact on research outcomes. Given the increasing popularity and application of transformer-based language models like ChatGPT, this paper encourages social scientists to explore how this state-of-the-art technology can enhance their research.[1]

Keywords: Text clustering, Topic modeling, Large language model.

1 Introduction

Text clustering is a prevalent method in political science for exploring text data. Traditional probabilistic topic modeling algorithms (PTMs), such as Latent Dirichlet Allocation (LDA), Structural Topic Models (STM), and Keyword-Assisted Topic Models (keyATM), have been widely used due to their interpretable dimension reduction capabilities (Blei, Ng, and Jordan 2003; Eshima, Imai, and Sasaki 2020; Roberts *et al.* 2014). As these models circumvent the need for labeled training data or the creation of a codebook, they offer a cost-effective solution for text classification in social science research, compared to alternatives like supervised machine learning and dictionary-based methods.

However, PTMs have notable limitations that impede their performance. First and foremost, these models struggle to accurately classify short texts, which is of growing concern as political scientists increasingly focus on the analysis of short texts like social media posts and news titles. Second, PTMs' performance hinges heavily on the user's choices regarding text preprocessing and the number of clusters. Lastly, the training process for PTMs can be time-consuming and computationally intensive, particularly when dealing with large text datasets. As political scientists increasingly delve into the realm of big data, this challenge becomes more pronounced. Additionally, the time-intensive nature of training multiple models often deters researchers from thoroughly investigating the impact of varying text preprocessing strategies and topic counts on their results.

In response to these prevailing challenges, this paper introduces a novel approach employing transformer-based language models (TLMs) for text clustering, which I call TLM-based text clustering. TLMs can generate contextualized word embeddings, paving the way for diverse text analysis applications (Brown *et al.* 2020; Devlin *et al.* 2018; Vaswani *et al.* 2017). In this paper, TLMs are utilized to create vectors that represent documents (known as document embeddings or text embeddings). I then use K-means clustering on these vectors. While the concept of clustering

Political Analysis (2023)

DOI: 10.1017/pan.xxxx.xx

Corresponding author
Jianjun Yu

Edited by
John Doe

1. This paper is a draft using PA template, not an accepted paper

documents based on embeddings is not new, TLMs offer a unique advantage. They create document embeddings that are more meaningful and, consequently, yield higher accuracy when used for clustering. This allows the proposed approach to outperform PTMs and directly address the issues associated with these models. This paper emphasizes the transformative potential of TLMs in advancing text classification methodologies within social sciences. It serves as an invitation for scholars to explore and harness the significant potential that TLMs present for their research.

In the following sections, this paper first addresses the limitations of prevalent topic models before introducing the novel domain of transformer-based language models. It then elaborates on the mechanism of TLMs and provides a detailed, step-by-step guide to employing TLMs for text clustering. The distinct advantages that TLMs offer over traditional topic models are also emphasized. It further compares the structure topic model, one of the most popular topic models in political science studies, with two TLM-based text clustering models to demonstrate the superior performance of TLM-based text clustering. While these models may not represent the pinnacle of transformer technology, the primary objective of this paper is to showcase TLM capabilities and encourage political scientists to shift from traditional methodologies towards more innovative and effective strategies. Researchers are urged to explore and utilize advanced TLMs that align with their specific research requirements.

2 The Challenge of Probabilistic Topic Models

Probabilistic topic models (PTMs) have gained significant traction in political science research (Blei, Ng, and Jordan 2003; Eshima, Imai, and Sasaki 2020; Roberts *et al.* 2014). PTMs posit that each document is a probabilistic mixture of topics, each of which represents a probability distribution over words in the corpus. Different models adopt distinct procedures for sampling topics and selecting words, but their overarching principle is to estimate the topic distribution for each document and the word distribution for each topic by analyzing word co-occurrence patterns.

Despite their widespread use in political science research, these models present challenges that hamper the reliability of derived findings and the exploration of crucial research questions. A major constraint is the difficulty in handling short texts, given that PTMs rely on a generative model of document creation. Shorter documents contain fewer word occurrences, making it more challenging to estimate their topic distributions accurately. For example, an LDA model with ten topics would consider a 200-word document as generated from 200 topic samples, each drawn randomly from a topic distribution with nine parameters. This ample number of samples enables a reliable estimation of the document's topic distribution. However, when dealing with a document containing only ten words, estimating the topic distribution becomes difficult due to the limited number of topic samples available.

To tackle the challenge of short texts in topic modeling, political scientists often resort to special text preprocessing strategies. One commonly employed strategy is the aggregation of short texts based on specific rules. For instance, Barberá *et al.* (2019) aggregated Congresspeople's tweets on a daily basis, totaling the tweets sent by members of Congress each day and categorizing them by party and chamber. This approach, however, has limited applicability as it depends on the theoretical meaningfulness of text aggregation. Alternatively, short texts might be entirely excluded from the dataset, as Ying, Montgomery, and Stewart (2022). demonstrated by removing Facebook posts with less than ten words. Although effective, this method risks losing valuable insights contained within those texts.

There are efforts in the natural language processing (NLP) community to design new topic models specifically for short-text analysis (Yan *et al.* 2013). However, their adoption in political science is limited, likely due to their complexity and time-intensive nature.

Another concern with PTMs is their dependence on users' decisions regarding text preprocessing and the selection of the number of clusters. Text preprocessing typically includes noise

reduction, tokenization, removal of stop words and punctuation, stemming and lemmatization, vocabulary control (removing high/low-frequency words), and data normalization (Grimmer, Roberts, and Stewart 2022). The choice of preprocessing techniques should be grounded in substantive knowledge and domain-specific considerations. For instance, legal documents may require special handling, such as treating "Roe v. Wade" as a single token rather than three separate tokens. Denny and Spirling (2018) demonstrated that the choices made in text preprocessing could significantly impact the selection of the number of clusters in an LDA model, subsequently influencing how the model labels the text.

Finally, the training process for PTMs can be computationally intensive and time-consuming, particularly with large text datasets. PTMs require the entire dataset to be loaded into memory, posing challenges with larger datasets. Furthermore, most topic models use relatively slow algorithms, such as Markov chain Monte Carlo (MCMC) or Expectation-Maximization (EM). When data size is large, it might take days to cluster data, which makes it harder for political scientists to explore how the different number of clusters influence their research outcomes.

These limitations of PTMs make them less suitable for current research needs, particularly as more political scientists are studying social media, where text data is often short, but the dataset size is large. To overcome these challenges, I propose the use of transformer-based language models for conducting text clustering. Transformer models, such as the ones used in state-of-the-art language models like BERT (Bidirectional Encoder Representations from Transformers) and GPT (Generative Pre-trained Transformer), have shown remarkable performance in various NLP tasks and offer advantages such as parallel processing and the ability to handle short texts effectively (Brown *et al.* 2020; Devlin *et al.* 2018). By leveraging these transformer-based models, researchers can potentially address the limitations of probability topic models and achieve more accurate and efficient text clustering.

3 What is Transformer-based Language Models?

Transformer-based language models (TLMs), such as OpenAI's GPT and Google's BERT, have gained recognition for their exceptional performance in language-related tasks (Brown *et al.* 2020; Devlin *et al.* 2018; Vaswani *et al.* 2017). For text analysis, TLMs convert a document into a sequence of word embeddings. Word embeddings are vector representations that capture the meaning of words, enabling semantically similar words to have similar vectors (Harris 1954; Rodman 2020). Unlike other word embedding generation approaches, TLMs analyze the relationships between words in a text and generate contextual word embeddings that incorporate the interdependence of words. This allows for a more accurate representation of word meanings.

During training, TLMs will learn to understand the context and relationships between words, enabling them to generate contextual embedding sequences. TLMs then utilize these contextual word embeddings to perform various text analysis tasks.

For political scientists interested in TLMs, Appendix A provides a detailed explanation of how TLMs convert texts into the contextual word embedding sequences, how to build a TLM, and conduct training.

TLMs have emerged as highly efficient and accurate models, surpassing traditional models like CNN and RNN in various machine-learning tasks. Their effectiveness has been demonstrated across various applications, including chatbots, supervised text classification, text generation, and translation (Kalyan, Rajasekharan, and Sangeetha 2021; OpenAI 2023). Furthermore, they have gained popularity in fields like radio and video analysis, as well as attracting interest from researchers working in image analysis (Arnab *et al.* 2021; Han *et al.* 2021; Huang *et al.* 2018; Parmar *et al.* 2018).

However, the adoption of transformer-based models in political science research has been limited. As of now, their application in political studies has primarily been focused on tasks such as

supervised text classification and emotion detection, utilizing models like BERT and GPT (Licht 2022; Widmann and Wich 2022). This paper explores the potential of transformer-based models in text clustering and replace commonly used topic models. By leveraging the power of transformer-based models, it is hoped that other scholars will be inspired to explore their applications in various other tasks within the field of political science.

4 Implementing Text Clustering with Transformer-Based Models

This section details the process of employing TLMs for text clustering.

4.1 Step one: Selection of a Pre-Train Transformer-based Model

The initial step in deploying TLM-based text clustering involves choosing an appropriate pre-trained model. Pre-trained models are TLMs that scholars have trained. These models have already learned general language patterns and representations from a large corpus. There are several commonly used pre-trained models, including ALBERT, BERT, ELECTRA, GPT, RoBERTa, XLNet, and T5. These models are readily available for download or can be accessed through APIs. Scholars also have the option to train their own pre-trained models tailored to their specific tasks. For example, if the subject of study involves American legal documents, researchers can collate texts such as American congress bills, court decisions, and state laws to train a novel "BERT-Law" model. This method amplifies the model's comprehension of legal documents and enhances its efficacy in analyzing such texts.

However, it's crucial to remember that training a TLM demands a significant amount of data and is a highly time-intensive process (Strubell, Ganesh, and McCallum 2019). Thus, for researchers limited by a small text corpus or budget, utilizing existing pre-trained models is preferable. Many of these models, having been developed by leading companies like Google and OpenAI, have already processed a large volume of text and are well-equipped to handle various tasks. As such, training a new TLM is typically unnecessary.

For English text clustering, I recommend the Sentence-RoBERTa model (SBERT) and the text-embedding-ada-002 model. SBERT is a pre-trained model fine-tuned to generate high-quality text embeddings—vector representations that encapsulate the meaning of texts (Reimers and Gurevych 2019). SBERT, based on the RoBERTa model (a BERT variant), employs pair-sentence data for fine-tuning. The objective is to generate sentence embeddings that yield a cosine similarity closer to 1 for texts with similar meanings and closer to -1 for texts with contradictory meanings. This text clustering based on SBERT is referred to as BERT-based text clustering.

Another favorable choice is the text-embedding-ada-002 model. It's the second-generation text embedding model based on GPT-3, recommended by OpenAI for generating text embeddings (Greene *et al.* 2022). This text clustering based on the text-embedding-ada-002 model is referred to as GPT-based text clustering. GPT-based models have gained significant attention due to the rising popularity of ChatGPT, a chatbot, since 2023, and their reported superiority over BERT-based models in several NLP tasks.

Despite the recommendation of BERT-based text clustering and GPT-based text clustering, researchers are encouraged to choose the model that best aligns with their work.

4.2 Step Two: Data Tokenization

Having selected the pre-trained model, the next phase involves tokenizing your text data. Tokenization is dividing the text into individual tokens so TLMs can convert them into vectors. Each pre-trained model utilizes a specific tokenization algorithm, dictating how the text is partitioned. Widely used tokenization algorithms encompass Byte-Pair Encoding (BPE), WordPiece, and SentencePiece. These subword tokenization strategies treat high-frequency words as whole tokens,

while breaking down low-frequency words into higher-frequency subwords. This approach helps decrease the token size and accelerates the training process.

For instance, subword tokenization algorithms may decompose the word "transformer" into "transform" and "er", as "transform" and "er" appear more frequently in English than "transformer". This type of decomposition will not interfere with the understanding of "transformer" as TLMs will learn how the meaning of "transform" is altered by the subsequent "er".

The exact implementation details may vary across different tokenizers, so it is recommended to consult the relevant papers to gain a deeper understanding of their workings. For SBERT, users need to use the RoBERTa tokenizer.

Unlike topic models, you do not need to remove any symbols, such as emojis, from the text before tokenization nor do you need to remove any tokens post-tokenization. The sole prerequisite is ensuring the inputs are texts, as opposed to pictures or video links.

4.3 Step Three: Generation of Word Embedding Sequences for Each Text

Upon tokenizing the data, the subsequent step is to generate word embedding sequences for each piece of text. Since we are not training a model, it is unnecessary to input all the data simultaneously. Researchers should consider the memory capacity of their computers and input a portion of the data at a time. They may also partition the data into several subsets and employ different computers to generate word embedding sequences for different subsets. Generating word embedding sequences using GPUs and TPUs is a highly efficient operation. In this study, I utilized Google Colab with a TPU to tokenize and then generate word embedding sequences of nearly 400,000 news titles. It took approximately just 1 minute to generate all the embedding sequences. In the Replication code of this paper, a sample code is provided to illustrate how to employ GPUs or TPUs to accelerate the estimation process. These capabilities allow researchers to generate word embedding sequences for large datasets with relative ease and efficiency.

4.4 Step Three: Generation of Text Embeddings

The subsequent phase is generating text embeddings. In contrast to word embeddings that represent individual words, text embeddings encapsulate a text's holistic meaning and context. Multiple methods are available for amalgamating word embeddings into text embeddings.

A widely adopted approach, exemplified by SBERT, is to compute the mean of all the word embeddings within a text. The resultant text embedding captures the central tendencies of the word embeddings, thus representing the comprehensive semantic content of the text.

In the case of GPT-based text clustering, OpenAI's API provides direct access to text embeddings of input text. However, the details of how OpenAI aggregates the word embedding sequence have not been disclosed.

4.5 Step Four: Clustering Text Using K-means Clustering Algorithm

Once sentence embeddings have been obtained, the K-means clustering algorithm can be applied to cluster them based on their similarities. K-means clustering is a renowned unsupervised machine learning algorithm to partition data points into groups or clusters (Forgy 1965; Lloyd 1982; MacQueen 1967). This algorithm ensures that data points within the same cluster are as closely related as possible.

Researchers need to select the number of clusters for K-means clustering, denoted as K. Two methods that can assist in selecting the appropriate K are the Elbow Curve Method and Silhouette Analysis.

The Elbow Curve Method involves calculating the sum of the squared distances between each data point and its corresponding centroid within the cluster (SSD) (Cui 2020). By plotting the

SSD values against the number of clusters, users can look for an "elbow point" where the rate of decrease in SSD slows down significantly. This point suggests a good trade-off between the number of clusters and the SSE. If the elbow point is not clear, users can examine the first difference of the SSD to identify when the decrease becomes less significant.

Silhouette Analysis is another method for determining the appropriate K. It measures how similar a data point is within its own cluster compared to other clusters (Rousseeuw 1987). The Silhouette score ranges from -1 to 1, where a higher score indicates a better choice of K. To compute the Silhouette score, one can calculate the average score for each data point and then obtain the average Silhouette score for each value of K. However, note that calculating the Silhouette score for large datasets can be computationally expensive. In practice, using a random sample of the data is recommended for calculating the Silhouette score when dealing with large datasets.

Both the Elbow Curve Method and Silhouette Analysis should be used together to select the optimal number of clusters. It's important to note that the "optimal" number of clusters may not be a single value but rather a range of values where both methods suggest reasonably good results. In such cases, users can try different values of K within the range and evaluate how the choice of K affects their clustering results. The advantage of K-means clustering compared to probabilistic topic modeling is its computational efficiency, allowing for easy exploration of different values of K.

4.6 Step Five: Generate Labels from Clusters

Upon achieving clusters using the K-means clustering algorithm, users can progress to the task of assigning labels to these clusters. Several approaches are applicable in this scenario. One method is to randomly select a handful of documents from each cluster and manually read and analyze these texts to identify a fitting label for the cluster. Another strategy is to choose a few central documents from each cluster. Each cluster possesses a center point that potentially represents the semantic meaning of the cluster. Users can select a few documents whose text vectors are closest to this center point. These documents should closely align with the cluster's meaning.

5 Superiority of TLMs-based Text Clustering Over Probabilistic Topic Models

5.1 Greater Precision

The fundamental difference between TLM-based text clustering and PTMs in performing text clustering resides in utilizing a more diverse range of text features. Unlike PTMs, which exclusively rely on word co-occurrence patterns for text clustering, TLM-based text clustering integrates multiple aspects of the text, encompassing word co-occurrence patterns, the semantic meaning of words, the sequential order of words, and contextual information. This is made feasible by harnessing the power of TLMs that extract and vectorize these diverse features from the text.

By portraying the text as high-dimensional vectors built on the extracted features, similarities between texts can be effectively gauged, thereby facilitating meaningful clustering. The K-means algorithm groups the texts based on their feature similarities, considering the multiple facets derived from the text. This comprehensive analysis enables a more detailed examination of the text and aids in identifying more nuanced clusters that are closer to human interpretation. This approach broadens the scope of text features that can be analyzed and significantly enhances the clustering process's precision.

5.2 Robust to Short Text

Traditional topic modeling techniques often face difficulties when handling short texts, given the scarcity of information available for reliable estimation of topic distribution. This limitation results from the number of available topics not providing a large enough sample size for reliably estimating the topic distribution that birthed these topics. With limited textual content, discerning

the dominant topics and their respective proportions within the document becomes challenging.

In comparison, clustering methodologies utilizing TLMs do not encounter the same issues with short texts. Transformer-based models can effectively extract meaningful features and representations, such as word meaning, semantic relationships, and contextual information, from texts, irrespective of their length. This enables accurate clustering grounded on a more comprehensive range of information beyond simple word co-occurrence patterns.

5.3 Simplified Text Preprocessing

The preprocessing stage in text clustering using transformer-based models offers a significant benefit of simplifying numerous preprocessing decisions, thereby reducing the burden on social scientists. As mentioned earlier, text preprocessing in PTMs can be intricate, necessitating a substantial degree of domain-specific knowledge and considerations. Transformer-based models, however, streamline this process, rendering it more accessible and user-friendly.

The complexity of text preprocessing in probabilistic topic modeling largely arises from its reliance on word co-occurrence patterns to attribute topic distributions to documents and word distributions to topics. Consequently, words appearing with high or low frequency may not provide valuable information for the models to cluster documents effectively. These words can burden the topic model estimation process, prompting the removal of such high-frequency and low-frequency words. However, defining a standard threshold to identify excessively frequent or infrequent words is subjective and lacks a universally recognized criterion. This condition necessitates researchers to make arbitrary decisions based on personal judgment, potentially introducing variability and uncertainty during the preprocessing stage.

Another complication in the text preprocessing of PTMs stems from the fact that PTMs do not consider the semantic content of a word or a phrase. The tokenization process merely converts words into numerical IDs devoid of any semantic understanding. Hence, the same word or phrase used in different contexts will be assigned the same ID, while different words with similar meanings will receive unrelated IDs. This limitation hampers the ability of PTMs to understand that a word may possess different meanings in various contexts and that different words might hold similar meanings.

For instance, PTMs cannot discern the difference between using "white" and "house" separately and the phrase "White House." To address this, researchers often employ n-grams as tokens. When tokenized, N-grams, ordered sets of n words, receive separate token IDs, enabling PTMs to perceive "White House" as a single token rather than two separate ones. This necessitates the development of strategies to generate the n-gram list and include them in the tokenization process (Rule, Cointet, and Bearman 2015).

Contrastingly, TLMs present a more streamlined and simplified approach to text preprocessing. These models are proficient in extracting meaningful word, phrase, and sentence vectors that capture semantic relationships. Hence, they can effectively comprehend the differing meanings of the same word in various contexts and identify similarities among distinct words. This eliminates the need for additional preprocessing steps like creating an n-gram list, stemming, lemmatization, vocabulary control, and data normalization. The only requirement in text preprocessing for TLM-based text clustering is ensuring that the inputs are indeed texts. Subsequently, the associated tokenization algorithm will automatically convert the text into tokens, eliminating the need for human judgment.

5.4 Flexibility to Explore Different Cluster Numbers

The speed and computational efficiency of transformer-based large language models for text clustering offer significant advantages, particularly when dealing with large-scale datasets in social

science research. The performance of TLM-based text clustering enables users to readily explore how varying the number of clusters may impact their analysis, thereby promoting the production of more robust findings.

TLM-based text clustering mainly involves two steps. The first step is generating text embeddings, a process that is not computationally intensive even for a large dataset. This is because it does not require any training process. Instead, a pre-trained transformer-based large language model is chosen, and this model is used to generate the text vectors. Given that the model's parameters are pre-trained, there's no need for additional updates. Users simply employ the model to create document embeddings, a process that can be swiftly accomplished using GPUs. Transformer-based models are designed to efficiently handle large data, thereby making the processing of sizeable datasets manageable.

After obtaining the document embeddings, users can experiment with determining the optimal number of clusters for K-means clustering. K-means is also a much faster clustering algorithm than PTMs. Moreover, since the dimension of text embeddings is the same regardless of the length of the documents, the speed of the K-means cluster based on TLMs is independent of the document length. However, for PTMs, the time consumed is proportional to the length of the documents (Blei, Ng, and Jordan 2003). Therefore, when the document length is long, the time difference between TLM-based text clustering and PTMs will be much larger.

In the subsequent section, I will showcase the time difference using real-world data to underscore the speed and efficiency advantage of TLM-based clustering over traditional topic modeling approaches.

6 Performance Evaluation

This section provides a comparative performance analysis between TLM-based text clustering and the Structural Topic Model (STM), a widely adopted topic modeling technique in the realm of political science (Roberts *et al.* 2014). The objective is to underscore the superior efficacy of TLM-based text clustering, especially when dealing with short text data and social media data.

6.1 Datasets

The performance comparison is conducted using two datasets. The first dataset, dubbed the News Aggregator Data Set, is sourced from the Center for Machine Learning and Intelligent Systems at the University of California, Irvine (UCI data) (Gasparetti 2016). Released in 2016, this dataset comprises 422,419 news titles. The documents within this dataset are characteristically short, averaging at about 9.25 words. These titles have been manually categorized into four clusters: business, science and technology, entertainment, and health. Upon removing duplications and erroneous data entries[2], a total of 398,142 news titles were included in this study.

The second dataset is the Twitter Topic Classification data (single topic) compiled by Cardiff NLP from the School of Computer Science and Informatics at Cardiff University (TTC data) (Antypas *et al.* 2022). This dataset was initially used to train a supervised text classifier. For the purpose of this study, I aggregated the training, validation, and testing datasets to create a labeled dataset of 6,984 tweets. As STM's performance tends to decline when handling short texts, I eliminated tweets consisting of fewer than ten words. This enhanced STM's performance and allowed for the demonstration of the superior efficacy of transformer-based text clustering even when short texts are excluded. Following this filtering process, 6,152 tweets were retained. These tweets were manually labeled into six clusters: arts and culture, business and entrepreneurs, science and technology, gaming and sports, pop culture, and daily life.

Both datasets have been manually labeled based on their content themes, providing a ground

2. For instance, some rows of the dataset lack news titles, while some others contain URLs instead of titles

truth against which the performance of topic modeling and TLM-based text clustering can be evaluated.

6.2 Models for Comparison

The performance comparison involves three models: the STM, BERT-based text clustering, and GPT-based text clustering.

The Structural Topic Model (STM) was proposed by Roberts *et al.* (2014) and has since gained popularity as a topic model within political science circles. It was selected for this comparison given its extensive application in political science research. In terms of TLM-based text clustering, two methods are proposed. The first method involves BERT-based text clustering, and the second method employs GPT-based text clustering. Both models have been introduced earlier in this study.

6.3 Preprocessing

Text preprocessing is a prerequisite for topic models. For both datasets used in this comparison, only English words were retained, and stop words were removed utilizing the Natural Language Toolkit (NLTK), a popular Python NLP package. However, I refrained from removing low or high-frequency words. Each text document was then tokenized into individual words.

For the BERT-based text clustering model, the RoBERTa tokenizer was employed during the preprocessing phase. For the GPT-based text clustering model, no additional preprocessing was deemed necessary since the OpenAI API can directly process raw text as input, thereby obviating the need for extra preprocessing steps.

6.4 Generation of Text Embeddings

Before proceeding with text clustering, it was necessary to generate text embeddings for TLM-based text clustering. To generate embeddings for the BERT-based clustering models, I used Google Colab's TPU. I timed how long the TPU required to generate word embeddings for the UCI dataset. Since the TTC dataset is quite small, I did not record the time for it. The TPU took 106 seconds to generate all text embeddings and then required an additional 18 minutes to transfer the data from TPU memory (RAM) back to RAM for further analysis. In total, this process cost me around 0.5 dollars. It's important to note that the time to transfer data from TPU RAM to RAM is proportional to the number of documents, not the memory size of the data because the text embedding for each document is of the same length. Therefore, increasing the length of the documents will not increase the time required to transfer data from TPU RAM to RAM.

To generate text embeddings for GPT-based text clustering, I employed OpenAI's API. The process took approximately 21 hours for the entire dataset, costing me around 2.5 dollars.

6.5 Selection of the Optimal Number of Clusters

Selecting the optimal number of clusters is an important step in text clustering. In this research, two different approaches are used to determine the number of clusters. The first approach considers the dataset's actual number of manually labeled clusters. For the UCI data, there are four clusters, while for the Twitter data, there are six clusters. The second approach involves selecting the optimal number of clusters based on model performance.

To select the optimal number of clusters for STM, I use coherence and perplexity scores. For TLM-based models, the Elbow curve method and silhouette analysis are employed. Due to the computational complexity of silhouette analysis on large datasets, a random sample of 5000 data points is selected for calculating the Silhouette score for the UCI data.

Based on the evaluation metrics, the optimal number of clusters for BERT-based text clustering is 25 for the UCI dataset and 15 for the TTC dataset. For GPT-based text clustering, the optimal

Table 1. Time Spent Training Different Models

Models	K=10	K=25	K=50	K=100
BERT-based text clustering	33 seconds	72 seconds	133 seconds	3 mins 13 seconds
Structure topic model	16 mins	24 mins	32 mins	1 hour and 4 mins

number of clusters is 15 for the UCI dataset and 15 for the TTC dataset. Finally, for STM, the optimal number of clusters is 25 for the UCI dataset and 10 for the TTC dataset.

In Appendix B, I provide more details on how the optimal number of clusters was selected. It is important to note that the metrics used in this paper, such as the coherence and perplexity scores, only provide an approximate range for the optimal topic numbers. Therefore, selecting the optimal number of clusters can vary based on researchers' preferences and specific requirements. However, the main objective of this study is to demonstrate the superiority of TLM-based text clustering over PTMs. Slight variations in the optimal number of clusters should not significantly impact the overall findings. Additionally, to mitigate the influence of different optimal numbers, I also calculate the performance of models using the optimal number of clusters for other models. This analysis further underscores the superior performance of TLM-based models over STM, even when the number of clusters is specifically chosen for STM.

6.6 Evaluation Strategy

The performance evaluation of each clustering method involves assigning labels to each cluster based on its dominant topic. When the chosen number of clusters aligns with the actual topic count in the dataset, the cluster is labeled with its predominant topic. For example, if the STM clusters the UCI data into four groups and the dominant topic of the first group is 'science,' this group is labeled as 'science.'

However, when the chosen number of clusters does not correspond to the actual topic count in the dataset, clusters are grouped based on their prevalent topic, and these aggregated clusters are labeled accordingly. For instance, if the STM clusters the UCI data into 25 groups, all clusters with a majority of texts related to 'business' are combined, and this merged cluster is labeled as 'business.' This approach of aggregation reflects the typical usage of unsupervised topic models in research studies.

Once the labels for each cluster are assigned, the performance of each method is evaluated using accuracy and F1 score metrics. These metrics provide an assessment of how well the clustering method captures the true topics and assigns the correct labels to the clusters.

6.7 Results

It's important to consider their computational time before delving into the accuracy and F1 score comparisons among different models. Table 1 presents the time spent in estimating clusters using BERT-based text clustering and STM on the UCI dataset. Both models run on a single core of a Intel Xeon Processor 2.20 GHz Processor on Google Colab. I do not include the time spent using GPT-based text clustering because it also uses K-meaning clustering and the time spent will not be very different from BERT-based text clustering.

The temporal discrepancy between the two methods is pronounced. On the UCI dataset, K-means clustering requires only 33 seconds to assign data to 10 clusters and 3 minutes to partition data into 100 clusters. On the other hand, STM takes much more time, requiring 16 minutes to estimate 10 clusters and over an hour to estimate 100 clusters.

In practical research applications, different numbers of clusters are tested to ascertain the optimal quantity for clustering. This practice accentuates the temporal differences between using

Table 2. Accuracy and F1 score on UCI data

	Accuracy			F1 score		
Models	**K=4**	**K=15**	**K=25**	**K=4**	**K=15**	**K=25**
BERT-based text clustering	0.67	0.82	**0.83**	0.63	0.82	**0.83**
GPT-based text clustering	0.67	**0.82**	0.83	0.63	**0.82**	0.83
Structure topic model	0.54	0.66	**0.68**	0.51	0.65	**0.67**

TLM-based text clustering and PTMs. Moreover, in this study, estimating an STM with 100 clusters only takes an hour, primarily due to the short average length of documents. The average length of documents in the UCI dataset is around nine words. Given that STM estimation time is proportional to the average document length, a dataset of similar size but with an average document length of 200 words could necessitate around 20 hours to train an STM with 100 topics! Under such conditions, researchers would have to rely on a high-performance computing center (HPC) with ample RAM and multiple cores to compute several STMs simultaneously. In contrast, while TLM-based text clustering may require slightly more time to generate text embeddings, the time to generate clusters remains constant. Text embeddings can still be generated on Google Colab for just 1 or 2 dollars, and several topics can be tested within minutes.

The comparative analysis presented in Tables 2 and 3 underscores the performance of BERT-based text clustering, GPT-based text clustering, and structured Topic Model (STM) on UCI and TTC datasets. These tables depict the accuracy and F1 scores of the models for varying numbers of clusters, with the optimal number of clusters denoted in bold. The results distinctly show that TLM-based text clustering outperforms STM on both datasets.

For the UCI dataset, STM exhibits accuracy and F1 scores ranging from 0.5 to 0.6 across all numbers of clusters, suggesting limited effectiveness in clustering short text. In contrast, TLM-based text clustering consistently demonstrates superior performance. When the selected number of clusters matches the human-clustered categories, TLM-based text clustering achieves approximately 0.1 higher accuracy and F1 scores compared to STM. Furthermore, when comparing the models' accuracy and F1 scores for their optimal number of clusters, TLM-based text clustering surpasses STM by approximately 0.15. The performance gap is even more pronounced for the TTC dataset, with TLM-based text clustering surpassing STM by around 0.2 in accuracy and F1 scores. This discrepancy may be attributed to the smaller size of the TTC dataset, which limits STM's ability to learn sufficient word co-occurrence patterns. In contrast, TLM-based text clustering, which relies less on specific data characteristics, demonstrates robustness and performs well even with relatively smaller datasets.

These findings underscore the superiority of TLM-based text clustering over STM in terms of clustering accuracy and F1 scores. TLM-based models offer more effective and reliable results for clustering text data. Additionally, the comparative analysis suggests that the performance of BERT-based text clustering and GPT-based text clustering is comparable. Given the substantial computational resources required for GPT-based text clustering, it is recommended to utilize BERT-based text clustering in most cases due to its similar performance and lower computational costs.

In summary, the results indicate that TLM-based text clustering, specifically BERT-based text clustering, outperforms STM in terms of accuracy and F1 scores. These models provide more accurate and reliable clustering results, making them valuable tools for social science researchers.

Table 3. Accuracy and F1 score on TTC data

	Accuracy			F1 score		
Models	**K=6**	**K=10**	**K=15**	**K=6**	**K=10**	**K=15**
BERT-based text clustering	0.69	0.73	**0.74**	0.60	0.73	**0.71**
GPT-based text clustering	0.73	0.75	**0.74**	0.70	0.74	**0.72**
Structure topic model	0.51	**0.51**	0.54	0.48	**0.47**	0.50

7 Conclusion

In summary, this paper provides a comparative analysis of the performance of transformer-based large language models and traditional methodologies, particularly in the realm of text clustering. The study accentuates the advantages of TLM-based text clustering, specifically utilizing models like BERT, over traditional topic modeling techniques such as the Structured Topic Model. The findings bear substantial implications for researchers in the discipline of political science, who are increasingly dependent on comprehensive text data analysis.

As transformer-based large language models become increasingly prevalent in the field of text analysis and consistently outperform traditional methods, this paper delivers an in-depth exploration of what TLMs are and how they can be effectively employed in political science studies. It is hoped that this exposition will aid researchers in further harnessing the power of this innovative approach to bolster their research endeavors.

Data Availability Statement: Replication data and code are all ready to share.

References

Antypas, D., A. Ushio, J. Camacho-Collados, L. Neves, V. Silva, and F. Barbieri. 2022. "Twitter Topic Classification." *arXiv preprint arXiv:2209.09824.*

Arnab, A., M. Dehghani, G. Heigold, C. Sun, M. Lučić, and C. Schmid. 2021. "Vivit: A video vision transformer." In *Proceedings of the IEEE/CVF international conference on computer vision,* 6836–6846.

Barberá, P., A. Casas, J. Nagler, P. J. Egan, R. Bonneau, J. T. Jost, and J. A. Tucker. 2019. "Who leads? Who follows? Measuring issue attention and agenda setting by legislators and the mass public using social media data." *American Political Science Review* 113 (4): 883–901.

Blei, D. M., A. Y. Ng, and M. I. Jordan. 2003. "Latent dirichlet allocation." *Journal of machine Learning research* 3 (Jan): 993–1022.

Brown, T., B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, *et al.* 2020. "Language models are few-shot learners." *Advances in neural information processing systems* 33:1877–1901.

Cui, M. 2020. "Introduction to the k-means clustering algorithm based on the elbow method." *Accounting, Auditing and Finance* 1 (1): 5–8.

Denny, M. J., and A. Spirling. 2018. "Text preprocessing for unsupervised learning: Why it matters, when it misleads, and what to do about it." *Political Analysis* 26 (2): 168–189.

Devlin, J., M.-W. Chang, K. Lee, and K. Toutanova. 2018. "Bert: Pre-training of deep bidirectional transformers for language understanding." *arXiv preprint arXiv:1810.04805.*

Eshima, S., K. Imai, and T. Sasaki. 2020. "Keyword assisted topic models." *arXiv preprint arXiv:2004.05964.*

Forgy, E. W. 1965. “Cluster analysis of multivariate data: Efficiency vs. interpretability of classifications.” *biometrics* 21:768–769.

Gasparetti, F. 2016. *News Aggregator.* UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5F61C.

Greene, R., T. Sanders, L. Weng, and A. Neelakantan. 2022. “New and improved embedding model.” Accessed June 7, 2023. https://openai.com/blog/new-and-improved-embedding-model.

Grimmer, J., M. E. Roberts, and B. M. Stewart. 2022. *Text as data: A new framework for machine learning and the social sciences.* Princeton University Press.

Han, K., A. Xiao, E. Wu, J. Guo, C. Xu, and Y. Wang. 2021. “Transformer in transformer.” *Advances in Neural Information Processing Systems* 34:15908–15919.

Harris, Z. S. 1954. “Distributional structure.” *Word* 10 (2-3): 146–162.

Huang, C.-Z. A., A. Vaswani, J. Uszkoreit, N. Shazeer, I. Simon, C. Hawthorne, A. M. Dai, M. D. Hoffman, M. Dinculescu, and D. Eck. 2018. “Music transformer.” *arXiv preprint arXiv:1809.04281.*

Kalyan, K. S., A. Rajasekharan, and S. Sangeetha. 2021. “Ammus: A survey of transformer-based pretrained models in natural language processing.” *arXiv preprint arXiv:2108.05542.*

Licht, H. 2022. “Cross-Lingual Classification of Political Texts Using Multilingual Sentence Embeddings.” *Political Analysis,* 1–14.

Lloyd, S. 1982. “Least squares quantization in PCM.” *IEEE transactions on information theory* 28 (2): 129–137.

MacQueen, J. 1967. “Classification and analysis of multivariate observations.” In *5th Berkeley Symp. Math. Statist. Probability,* 281–297. University of California Los Angeles LA USA.

OpenAI. 2023. “GPT-4 Technical Report.” *ArXiv* abs/2303.08774.

Parmar, N., A. Vaswani, J. Uszkoreit, L. Kaiser, N. Shazeer, A. Ku, and D. Tran. 2018. “Image transformer.” In *International conference on machine learning,* 4055–4064. PMLR.

Reimers, N., and I. Gurevych. 2019. “Sentence-bert: Sentence embeddings using siamese bert-networks.” *arXiv preprint arXiv:1908.10084.*

Roberts, M. E., B. M. Stewart, D. Tingley, C. Lucas, J. Leder-Luis, S. K. Gadarian, B. Albertson, and D. G. Rand. 2014. “Structural topic models for open-ended survey responses.” *American journal of political science* 58 (4): 1064–1082.

Rodman, E. 2020. “A timely intervention: Tracking the changing meanings of political concepts with word vectors.” *Political Analysis* 28 (1): 87–111.

Rousseeuw, P. J. 1987. “Silhouettes: a graphical aid to the interpretation and validation of cluster analysis.” *Journal of computational and applied mathematics* 20:53–65.

Rule, A., J.-P. Cointet, and P. S. Bearman. 2015. “Lexical shifts, substantive changes, and continuity in State of the Union discourse, 1790–2014.” *Proceedings of the National Academy of Sciences* 112 (35): 10837–10844.

Strubell, E., A. Ganesh, and A. McCallum. 2019. “Energy and policy considerations for deep learning in NLP.” *arXiv preprint arXiv:1906.02243.*

Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. 2017. “Attention is all you need.” *Advances in neural information processing systems* 30.

Widmann, T., and M. Wich. 2022. “Creating and comparing dictionary, word embedding, and transformer-based models to measure discrete emotions in german political text.” *Political Analysis,* 1–16.

Yan, X., J. Guo, Y. Lan, and X. Cheng. 2013. “A biterm topic model for short texts.” In *Proceedings of the 22nd international conference on World Wide Web,* 1445–1456.

Ying, L., J. M. Montgomery, and B. M. Stewart. 2022. “Topics, concepts, and measurement: A crowdsourced procedure for validating topics as measures.” *Political Analysis* 30 (4): 570–589.