Accurate Classification for Government Data: A Tree-of-Thoughts-Driven Few-Shot Learning Approach

Mengxiang Zhu[1,2,3], Yunchuan Guo[1,2,3], Duo Zhang[4], Ziyan Zhou[1,2,3], Haoyang Yu[1,2,3], and Lingcui Zhang[1,2,3](✉)

[1] Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
{Zhumengxiang,Guoyunchuan,zhanglingcui}@iie.ac.cn
[2] School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
[3] State Key Laboratory of Cyberspace Security Defense, Beijing, China
[4] Beijing Electronic Science and Technology Institute, Beijing, China

Abstract. Government data is essential for social and economic development, underpinning policy-making, public services, and data-driven innovation. However, the heterogeneity of government datasets significantly increases the risk of misclassification. Currently, obtaining a robust and accurate classification model for government data is challenging due to label limitation and structural complexity, resulting in poor classification performance. To address this challenge, this paper utilizes a tree-of-thoughts-driven few-shot learning approach to propose an accurate classification for government data without requiring extensive feature engineering. In our approach, we employ a tree-of-thoughts-based structure to dynamically adjust the classification process. To enhance classification accuracy, we develop a label expansion method that further extends and clarifies the original classification terms. Additionally, we design a dynamic tree generator based on semantic clustering and association to achieve accurate classification. Experiments demonstrate that our approach achieves a precision of 85.15% in government data classification, surpassing existing methods in comparative studies.

Keywords: Government Data · LLM · Prompt Engineering · Tree of Thoughts

1 Introduction

Government Data, which integrates statistical data, social security information, environmental monitoring data, and transportation data, plays a crucial role in supporting government decision-making and owns huge economic value [1]. According to Research World [2], in government data, unstructured data represents an estimated 80 to 90 percent of all data. This highlights the importance of government data classification in ensuring that government data are effectively utilized in decision support and social services.

Existing schemes for government data classification can be roughly divided into two categories: the rule-based schemes [3, 4] and the deep learning-based schemes [5, 6]. In the first schemes, the predefined rules are used to recognize and classify data,

D.-S. Huang et al. (Eds.): ICIC 2025, LNCS 15851, pp. 368–379, 2025.
https://doi.org/10.1007/978-981-96-9849-3_31

including keyword-based rules, pattern-based rules, and structural features-based rules. For example, Song et al. [4] utilized IF-IDF information retrieval to classify government data with information retrieval rules. **However, the rule-based schemes often require precise classification rules and domain expert knowledge, which are difficult to obtain in scenarios with government data**. This means that these schemes cannot be used to efficiently classify government data. Consequently, many efforts have been spent on leveraging deep learning-based schemes to classify government data. Along this line, Mao et al. [5] utilized pre-trained language models to develop a classification framework, thereby achieving more accurate government data classification.

However, existing deep learning-based schemes cannot be directly used to classify government data for the following reasons: 1) **Limitation in Labels**: Traditional deep learning models rely heavily on large amounts of labeled data for training. In the context of government data classification, the diversity of data types and the lack of standardized classification criteria make it extremely challenging to obtain a high-quality dataset with consistent labels [7]. 2) **Low classification precision**: Even with fine-tuning, existing pre-trained models (e.g. BERT or GPT) often fail to deliver satisfactory classification accuracy. Furthermore, government data are inherently complex [8], own multiple hierarchical levels and intricate relationships among categories. Existing approaches fail to characterize the complex relationships.

To address the above challenges, we propose a Tree-of-Thoughts-driven few-shot learning approach to accurately classify government data. Our main contributions are as follows.

- To address the challenge of scarce high-quality labels in government data classification, we design a label expansion scheme that leverages large language models and external resources to automatically generate synonyms and related concepts that are semantically similar to the predefined categories, thereby enhancing label coverage and improving classification accuracy.
- To accurately extract hierarchical structures from government data, we design a Dynamic Tree Generator to automatically determine the tree's parameters, in which the number of clustered categories defines the tree's breadth, and semantic associations between categories determine its depth. This approach ensures a more accurate and structured representation of hierarchical semantics.
- Experimental results on government data demonstrate that our approach achieves a precision of 85.15% in government data classification with only 50 samples. Even in zero-shot scenarios, it maintains a high precision of 83.11%. Comparative experiments indicate that our scheme outperforms existing methods.

2 Related Work

2.1 Government Data Classification

Existing government data classification schemes can be roughly divided into two categories: the rule-based schemes and the deep-learning-based schemes. The rule-based schemes typically rely on the predefined rules to identify and classify data. For example, Yao et al. [9] combined rule-based features with knowledge-guided models for effective disease classification. Similarly, Jonathan et al. [10] utilized a taxonomy with 24 criteria

to classify government data patterns and conducted a clustering analysis that identified 16 key criteria and 6 clusters.

Different from the rule-based schemes, in the deep-learning-based schemes, neural networks are used to extract government data features for classification. Along with the line, combining a Bi-LSTM with an attention mechanism, Pan et al. [11] achieved better results than LSTM, logistic regression, and naive Bayes for classifying government data. Further, Guo et al. [12] proposed a domain knowledge-powered attention mechanism, which leverages simply organized domain knowledge to effectively classify Air Traffic Management hazardous data.

From the above analysis, the existing schemes for government data classification heavily rely on the predefined rules, a large number of the labeled or high-quality data. As a result, we cannot use the existing schemes to accurately classify government data.

2.2 Prompt Engineering

Prompt engineering refers to the process of systematically designing, optimizing, and adapting input prompts to maximize the performance and utility of large language models (LLMs) [13]. This method utilizes task-specific instructions, known as prompts, to enhance the model's capabilities without altering the core model parameters. Through this approach, the pre-trained models are seamlessly integrated into downstream tasks. From a reasoning and logical perspective, existing prompt techniques can be roughly divided into two categories: the chain-based reasoning [14, 15] and the structure-based reasoning [16, 17].

In the chain-based reasoning schemes, the linear and step-by-step reasoning chains are used to systematically decompose complex reasoning tasks into sequential or interlinked steps. Along this line, Wei et al. [14] proposed a chain-of-thought (CoT) prompting scheme to improve the performance on a variety of arithmetic, commonsense, and symbolic reasoning tasks, even outperforming fine-tuned GPT-3 with a verifier. Hu et al. [15] designed a Chain-of-Symbols (CoS) prompting scheme to represent intermediate reasoning steps in a compressed symbolic space and capture complex environments.

In the structure-based reasoning schemes, the predefined and non-linear representations (e.g. trees or graphs) are used to simulate complex logical relationships and guide the inference process and solve complex problems. Along this line, Yao et al. [16] proposed a Tree-of-Thoughts (ToT) framework as a generalization of the popular CoT method. This framework prompts language models to explore coherent units of text, called "thoughts," as intermediate steps. Further, Wang et al. [17] introduced the Chain-of-Table framework to enrich the application scenarios of non-linear reasoning methods. This framework explicitly uses tabular data as proxies for intermediate reasoning steps, enabling LLMs to dynamically plan subsequent steps.

Despite the powerful adaptability of prompt engineering, existing methods are challenging to directly apply to government data classification. To address these challenges, this paper proposes a Tree-of-Thoughts-driven few-shot learning approach that incorporates a Label Expansion Module and a Dynamic Tree Generator to accurately classify government data.

3 The Approach

3.1 Overview

As shown in Fig. 1, we propose a ToT-driven few-shot learning approach for government data to achieve precise classification. Our approach consists of three main modules: Label Expansion Module (LEM), Dynamic Tree Generator (DTG), and Tree of Thoughts Module (TTM). First, LEM takes the input classification candidates, performs normalization and online lexical database queries, and outputs expanded and combined label information. Next, DTG receives the output of the LEM, conducts semantic clustering analysis, and generates a hierarchical tree structure as its output. Finally, TTM integrates the outputs from the LEM and the DTG, adopts the TOT mechanism to process user input and produces precise final classification results.

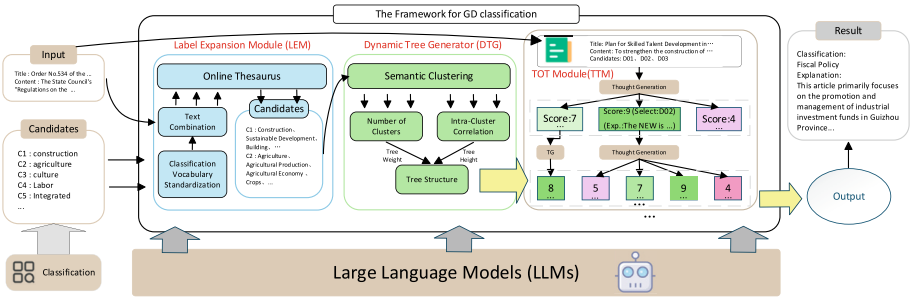


Fig. 1. System Design for Government Data Classification

3.2 LEM

To effectively improve the precision of text classification tasks, especially when handling ambiguous or under-represented categories, we design a LEM to enhance classification capability by expanding the representation of each category with semantically related terms such as synonyms, related words, and concepts.

LEM consists of three main stages:

1. Preprocessing Stage: LEM takes news articles and category labels as input, performs text preprocessing (e.g. tokenization, stopword removal and named entity recognition) and category normalization (e.g. synonym handling and minor corrections), and outputs normalized text and category labels.
2. Text Combination Stage: LEM processes the preprocessed text and normalized category labels, combines them to prepare for semantic expansion, and outputs the combined classification information.
3. Semantic Expansion Stage: LEM processes the combined classification information and uses large language models and external lexical resources (e.g. WordNet, Chinese Open Wordnet, and THUOCL) to generate synonyms and related concepts similar to the original category labels through semantic similarity retrieval. It outputs the expanded label set S.

Next, we formally define LEM. Let C denote the predefined category label set, T represent the text content of the news articles, S signify the expanded category label set, and R correspond to the external resource repository. The core task of LEM is to automatically generate a new expanded label set S from C, T, and R, where each label in S is semantically similar to the original category labels. The expansion process is defined as follows:

$$S_i = \left\{ t_j \middle| sim\left(c_i, t_j\right) > \theta \text{ and } t_j \in R \right\} \tag{1}$$

where S_i is the expanded label set for category c_i.

$sim\left(c_i, t_j\right)$ represents the semantic similarity between category c_i and term t_j, calculated using the cosine similarity of their corresponding word embeddings:

$$sim\left(c_i, t_j\right) = \frac{\overrightarrow{c_i} \cdot \overrightarrow{t_j}}{|\overrightarrow{c_i}| \cdot |\overrightarrow{t_j}|} \tag{2}$$

where t_j is a term or concept semantically similar to c_i, derived from the external resource repository R.

θ is the minimum similarity threshold, defined as:

$$\theta = \mu + \alpha \cdot \sigma \tag{3}$$

where μ and σ are the mean and standard deviation of the similarity scores between c_i and all candidate terms $t_j \in R$, and α is a tunable hyperparameter that controls the strictness of the expansion.

Through the above three stages, LEM ensures that the expanded category labels are highly accurate, relevant, and expressive, thereby effectively enhancing the performance of government data classification tasks.

3.3 DTG

To improve the model's capability in extracting hierarchical structures from government data, we introduce DTG module to dynamically cluster candidate categories by analyzing the semantic relationships between them, determining the tree's width (number of major categories) and height (hierarchical relationships within major categories), thereby enhancing the classification model's expressiveness and adaptability.

DTG module primarily comprises two stages:

1. Semantic Clustering Stage: The DTG module first employs LLM to perform semantic analysis on the candidate category set generated by LEM, aggregating semantically similar categories into major categories.
2. Hierarchy Construction Stage: After determining the number of major categories, the DTG module adopts a recursive semantic analysis method to further refine the semantic distinctions within each major category, constructing hierarchical relationships layer by layer.

DTG is formally defined as follows. Let C be the candidate category set generated by LEM. The semantic similarity between categories is calculated using the following function:

$$\mathrm{Sim}(c_i, c_j) = f_{\mathrm{LLM}}(c_i, c_j) \tag{4}$$

where f_{LLM} denotes the semantic analysis function based on a large language model (LLM).

In the clustering stage, DTG divides the candidate category set C into k major categories based on semantic similarity, each major category G_i is defined as:

$$G_i = \{c_j \mid c_j \in C, \mathrm{Sim}(c_j, c_l) \geq \theta, \forall c_l \in G_i\} \tag{5}$$

and θ is the semantic similarity threshold.

The width W of the thinking tree is defined as the number of major categories:

$$W = |G| \tag{6}$$

Within each major category, hierarchical structures are constructed through recursive semantic analysis. The height H_i of a major category is defined as:

$$H_i = \max_{c_j \in G_i} (\mathrm{depth}(c_j)) \tag{7}$$

where $\mathrm{depth}(c_j)$ represents the hierarchical level of category c_j within the tree.

Finally, DTG module integrates the results of semantic clustering and hierarchical construction to generate a TOT thinking tree with semantic relevance and hierarchical structure. The tree's width reflects high-level semantic grouping of categories, while the height reveals the hierarchical organization within major categories.

3.4 TTM

To integrate the reasoning capabilities of language models with structured input processing, we introduce TTM. This module combines outputs from LEM and DTG, processes user input through the Tree of Thoughts, and delivers highly accurate final classification results.

TTM primarily comprises three stages:

1. Thought generator (TG). Building on the thinking tree structure developed by DTG through the LLM, the Thought Generator performs a deeper exploration by identifying and focusing on the key points within each subtree. For each state, candidate thoughts are proposed based on the current state s and the number of candidates k. The core lies in dynamically integrating the subtree structures constructed by DTG to ensure each candidate has higher semantic relevance.
2. State evaluator (V). After the Thought Generator completes candidate generation, the State Evaluator assesses the quality of the state set S based on semantic similarity analysis performed by the DTG. This evaluation reflects the performance of each state s in the current task, ensuring that the final selected states are concentrated in the tree's core semantic areas.

3. Search algorithm. Breadth-First Search (BFS) retains b (b < W) of the most promising state sets at each step, ensuring the search process focuses on paths with high semantic relevance. Through iterative search and dynamic state evaluation, the algorithm efficiently explores the depth of the thinking tree and produces the final results.

Next, we formally define TTM. We use M to denote a large language model, lowercase letters $x, y, z, s, \cdots$ to denote a language sequence, and uppercase letters $S, \cdots$ to denote a collection of language sequences. Each node in the Tree of Thoughts represents a state:

$$s = [x, z_1, z_2, \ldots, z_i] \tag{8}$$

where x is the original input, and z_i represents intermediate steps for solving the problem.

The generation step in the Thought Generator (TG) for each state can be expressed as:

$$\left[z^{(1)}, \cdots, z^{(k)}\right] \sim M^{propose}\left(z_{i+1}^{(1\cdots k)} s\right) \tag{9}$$

where k is the number of candidates, and propose refers to using the propose prompt. After generating ideas, the State Evaluator evaluates the quality of the state set S, with V defined as:

$$V(M, S)(s) \sim M^{value}(v|s) \forall s \in S \tag{10}$$

where v represents the evaluation function value, reflecting the performance of state s in the current task.

Algorithm 1 TTM(x, M, TG, k, V, H, b)

REQUIRE Input x, LLM M, Thought generator TG(), size limit k, states evaluator V(), step limit H, breadth limit b

$S_0 \leftarrow \{x\}$

$for\ t\ =\ 1\ to\ H\ do$

$S_h' \leftarrow \{[s, z] \mid s \in S_{h-1}, z_h \in TG(M, s, k)\}$

$V_h \leftarrow V(M, S_h')$

$S_h \leftarrow \arg \max_{S \subset S_h, |S|=b} \sum_{s \in S} V_h(s)$

end for

return $TG\left(M \arg \max_{s \in S_H} V_H(s), 1\right)$

Finally, TTM employs the Breadth-First Search algorithm to traverse the tree of thoughts, ultimately deriving the optimal classification result. The detailed algorithm is defined as Algorithm.1.

4 Experiment Evaluation

4.1 Experimental Setup

Dataset Description. The dataset used in this study is derived from a government dataset published by Mao et al. [8], containing 5,000 samples collected from official government websites across nine provinces and autonomous regions in China. To ensure balanced representation, the dataset was reorganized into ten distinct categories. These categories include Urban Construction, Finance, Culture, Labor, Integrated Governance, Education, Agriculture, Market, Industry, and National Economy.

Experimental Environment. The experiments were conducted on a computer equipped with a 12th Gen Intel Core i7-12700 processor, 64 GB of RAM, and a 64-bit x64 architecture operating system. For software configuration, the experiments utilized LangChain[1], a framework for developing applications powered by large language models (LLMs), in conjunction with GPT-4o-mini as the supporting LLM for all tasks.

Sample Setup. To evaluate the proposed framework's performance under few-shot learning conditions, a limited number of labeled samples were randomly selected for each category to serve as prompts. Specifically, for each experimental configuration, 0, 1, or 5 samples were used per category. The test set was constructed to include 1,000 samples, while the remaining 3,500 samples constituted the training set.

Comparative Experiments. The study conducted four types of comparative experiments to assess the effectiveness of the proposed framework:

1. Comparison with Other Prompt Engineering Approaches: The experimental setup involved three distinct methods for comparison: the Baseline method, the Chain-of-Thought (CoT) method, and the proposed approach.
2. Comparison with Other Pretrained Model Approaches: This experiment compares the proposed approach with other methods that leverage pretrained models, including PET, iPET, and GD-PTCF [8].
3. Comparison with Classical Methods: The experimental setup also included a comparison with traditional deep learning models, such as classic deep learning architectures[2] and BERT-based models[3], using datasets of the same scale.

4.2 Experimental Results

Comparison with Other Prompt Engineering Approaches. This experiment evaluated the Tree-of-Thoughts-driven few-shot learning approach by comparing it with the Baseline and Chain-of-Thought (CoT) methods. Performance was assessed across three sample configurations (0, 10, and 50) using Precision, Recall, and F1-Score metrics. The results, detailed in Table 1, reflect the outcomes of these evaluations for government data classification tasks.

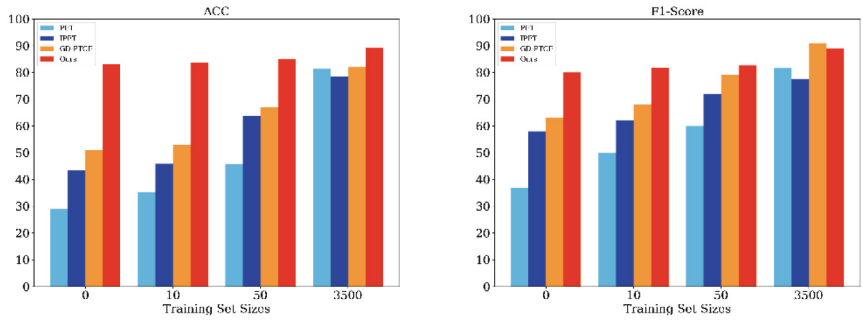
[1] https://www.langchain.com/.
[2] https://github.com/649453932/Chinese-Text-Classification-Pytorch.
[3] https://github.com/649453932/Bert-Chinese-Text-Classification-Pytorch.

Table 1. Performance comparison of different methods and sample sizes

Method	TS	Precision	Recall	F1-Score
Baseline	0	58.54	55.10	53.58
	10	61.64	57.80	56.65
	50	61.59	60.70	60.32
COT	0	60.82	59.90	58.73
	10	60.62	60.00	59.61
	50	63.21	62.50	61.90
Ours	0	83.11	79.48	80.10
	10	83.80	81.41	81.72
	50	**85.15**	**82.05**	**82.57**

The analysis reveals that as the sample size increases, the performance metrics for all methods show consistent improvement, demonstrating the positive impact of larger datasets on model effectiveness. Notably, our approach consistently outperforms both the Baseline and CoT methods across all metrics and sample sizes. For instance, with a sample size of 50, our method achieves a Precision of 85.15%, a Recall of 82.05%, and an F1-Score of 82.57%, significantly surpassing the results of the other methods. These observations highlight the strong performance of our approach, particularly in addressing the challenges of few-shot and zero-shot government data classification scenarios.

Comparison with Other Pretrained Model Approaches. This experiment assessed the performance of our approach in comparison with other pretrained model methods. The evaluation focused on average accuracy and F1-Score across multiple methods using training datasets of varying sizes, as illustrated in Fig. 2.



(a)Accuracy Across Different Training Dataset Sizes (b)F1-Score Across Different Training Dataset Sizes

Fig. 2. Compare experimental results.

The results demonstrate that our approach excels in few-shot learning scenarios, achieving higher accuracy and F1-Score compared to other methods. However, as the training set size increases to 3500, the performance across all methods converges, indicating reduced relative differences at larger scales. These findings emphasize the strength of our approach for few-shot classification tasks, while maintaining comparable performance in scenarios with larger datasets.

Comparison with Classical Methods. This experiment compared the performance of our approach with traditional deep learning models, including classic deep learning architectures and BERT-based models, using datasets of the same scale. The results, summarized in Table 2, provide a comprehensive comparison under consistent experimental conditions.

Table 2. Performance of different methods under varying TS values.

Method	TS = 0			TS = 10			TS = 50			TS = 3500		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
TextCNN	3.28	9.8	2.53	3.26	10.1	3.13	2.14	10.6	3.55	78.57	77.2	77.39
TextRNN	9.1	8.8	6.15	9.47	8.8	6.28	7.14	8.3	5.8	58.33	56.3	54.94
FastText	11.78	10.1	8.2	9.4	10.1	8.13	9.43	10.2	8.03	69.27	66.8	67.19
TextRCNN	3.64	10.7	4.67	3.58	10.7	4.65	3.62	10.7	4.68	75.75	74.3	74.54
TextRNN Att	8.66	9.8	8.23	9.25	10.1	8.55	8.28	10.6	8.18	76.33	75.5	75.63
DPCNN	1	10	1.82	1	10	1.82	1	10	1.82	75.14	72.5	72.52
Transformer	11.96	11.1	9.95	12.91	11.9	10.83	13.34	11.4	8.58	53.14	51.6	51.23
bert	6.4	7	4.64	5.96	11.1	6.19	7.77	10.3	3.04	82.2	82.2	82.13
bert CNN	3.04	10.5	4.66	2.3	7.8	2.54	2.66	7.6	2.31	82.73	82.3	82.29
bert RNN	4.12	9.1	4.92	4.04	10.1	2.63	9.56	11.6	7.69	82.17	81.7	81.7
bert RCNN	4.12	11.7	5.9	1.97	9.7	2.5	14.47	11.2	5.36	82.24	81.5	81.58
bert DPCNN	2.71	11.1	3.64	4.71	11.2	5.59	1	10	1.82	79.62	78.7	78.87
Ours	**83.11**	**79.48**	**80.1**	**83.8**	**81.41**	**81.72**	**85.15**	**82.05**	**82.57**	**89.13**	**89**	**89.01**

The experimental results demonstrate that our approach significantly outperforms traditional deep learning and BERT-based models across all training sample scales. Specifically, in few-shot scenarios (TS = 10), our method achieves an F1-Score of 81.72, compared to 8.55 (TextRNN_Att) and 6.19 (BERT-based models). Even in zero-shot scenarios (TS = 0), our method attains an F1-Score of 80.10, surpassing 9.95 (Transformer) and 4.64 (BERT-based models). Additionally, with a large-scale dataset (TS = 3500), our approach achieves an F1-Score of 89.01, outperforming TextCNN's 77.39 and bert_CNN's 82.29. These substantial improvements highlight the effectiveness, robustness, and scalability of our method for Chinese text classification under varying data conditions.

4.3 Ablation Study

To evaluate the effectiveness of the modules (LEM, DTG, and TTM) in the proposed framework, we conducted an ablation study using 10 samples for training and 1000 samples for testing. The detailed results are shown in Table. 3. Each row in the table represents a different configuration of these techniques, indicated with a check mark (✓), along with their corresponding Precision metric.

Overall, these experimental results demonstrate that combining all three modules achieves the best performance, underscoring their synergistic contributions and the overall effectiveness of the proposed framework.

Table 3. Ablation study (sample number = 10)

Number	LEM	DTG	TTM	Precision(%)
1	✓			67.05
2		✓	✓	81.18
3	✓		✓	79.74
4	✓	✓	✓	83.80

5 Conclusion

This paper designed a Tree-of-Thoughts-driven few-shot learning scheme to classify government data. In our scheme, LEM, DTG, and TTM are combined to address challenges brought by the limited labeled data and complex hierarchical structures. In the future, we will focus on several directions. The first is to enhance the scalability of our model for larger and more diverse datasets. The second is to improve the semantic expansion capabilities of LEM for even more accurate label generation and explore the integration of our approach with other domains like healthcare and finance.

Acknowledgments. This study was funded by the National Key Research and Development Program of China (No.2023YFB3106500).

References

1. Zhao, Y., Liang, Y., Yao, C., Han, X.: Key factors and generation mechanisms of open government data performance: a mixed methods study in the case of China. Gov. Inf. Q. **39**(4), 101717 (2022)
2. Heeg, R.: Possibilities and limitations of unstructured data. https://researchworld.com/articles/possibilities-and-limitations-of-unstructured-data (2022)
3. Onan, A., Korukoğlu, S., Bulut, H.: Ensemble of keyword extraction methods and classifiers in text classification. Expert Syst. Appl. **57**, 232–247 (2016)

4. Song, Y., Li, Z., He, J., Li, Z., Fang, X., Chen, D.: Employing auto-annotated data for government document classification, pp. 121–125. ACM, New York, NY, USA (2019)
5. Mao, M., Zhang, D., Xia, C., Guo, Y., Zhang, D., Li, X.: GD-PTCF: prompt-tuning based classification framework for government data. In: ICIC 2024, Part II, LNCS, vol. **14876**, pp. 211–224. Springer, Tianjin, China (2024)
6. Wang, Y., Gong, C., Ji, X., Yuan, Q.: Text classification for evaluating digital technology adoption maturity based on BERT: an evidence of Industrial AI from China. Technol. Forecast. Soc. Change **211**, 123903 (2025)
7. Minaee, S., Kalchbrenner, N., Cambria, E., Nikzad, N., Chenaghlu, M., Gao, J.: Deep learning-based text classification: a comprehensive review. ACM Comput. Surv. **54**(3), 62:1–62:40 (2022)
8. Crusoe, J.: The needs, wants, and wishes of the open government data idea – the emergence of a new understanding. EGOV 2024, LNCS, vol. **14841**, pp. 337–353. Springer, Ghent-Leuven, Belgium (2024)
9. Yao, L., Mao, C., Luo, Y.: Clinical text classification with rule-based features and knowledge-guided convolutional neural networks. BMC Med. Inform. Decis. Mak. **19-S**(3), 31–39 (2019)
10. Crusoe, J., Clarinval, A.: Classification of open government data solutions' help: A novel taxonomy and cluster analysis. In: Electronic Government – EGOV 2023. LNCS, vol. 14130, pp. 230–245. Springer, Budapest, Hungary (2023)
11. Pan, P., Chen, Y.: Automatic subject classification of public messages in e-government affairs. Data Inf. Manag. **5**(3), 336–347 (2021)
12. Guo, Z., Zeng, W., Quan, Z., Tan, X.: Domain knowledge-powered attention for air traffic management hazardous events classification. Eng. Appl. Artif. Intell. **138**, 109454 (2024)
13. Sahoo, P., Singh, A.K., Saha, S., Jain, V., Mondal, S., Chadha, A.: A systematic survey of prompt engineering in large language models: Techniques and applications. arXiv preprint arXiv:2402.07927 (2024)
14. Wei, J., et al.: Chain-of-thought prompting elicits reasoning in large language models. In: Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., Oh, A. (eds.) Advances in Neural Information Processing Systems, vol. 35, NeurIPS 2022, New Orleans, LA, USA (2022)
15. Hu, H., Lu, H., Zhang, H., Song, Y.-Z., Lam, W., Zhang, Y.: Chain-of-symbol prompting elicits planning in large language models. arXiv preprint arXiv:2305.10276 (2024)
16. Yao, S., Yu, D., Zhao, J., Shafran, I., Griffiths, T., Cao, Y., Narasimhan, K.: Tree of thoughts: deliberate problem solving with large language models. In: Oh, A., Naumann, T., Globerson, A., Saenko, K., Hardt, M., Levine, S. (eds.) Advances in Neural Information Processing Systems, vol. 36, NeurIPS 2023, New Orleans, LA, USA (2023)
17. Wang, Z., et al.: Chain-of-table: Evolving tables in the reasoning chain for table understanding. arXiv preprint arXiv:2401.04398 (2024)