Small Language Models in the Real World: Insights from Industrial Text Classification

Lujun Li[1], Lama Sleem[1], Niccolo' Gentile[2], Geoffrey Nichil[2], Radu State[1]
[1]University of Luxembourg, [2]Foyer S.A.,
Correspondence: lujun.li@uni.lu

Abstract

With the emergence of ChatGPT, Transformer models have significantly advanced text classification and related tasks. Decoder-only models such as Llama exhibit strong performance and flexibility, yet they suffer from inefficiency on inference due to token-by-token generation, and their effectiveness in text classification tasks heavily depends on prompt quality. Moreover, their substantial GPU resource requirements often limit widespread adoption. Thus, the question of whether smaller language models are capable of effectively handling text classification tasks emerges as a topic of significant interest. However, the selection of appropriate models and methodologies remains largely underexplored. In this paper, we conduct a comprehensive evaluation of prompt engineering and supervised fine-tuning methods for transformer-based text classification. Specifically, we focus on practical industrial scenarios, including email classification, legal document categorization, and the classification of extremely long academic texts. We examine the strengths and limitations of smaller models, with particular attention to both their performance and their efficiency in **V**ideo **R**andom-**A**ccess **M**emory (VRAM) utilization, thereby providing valuable insights for the local deployment and application of compact models in industrial settings[1].

1 Introduction

Text classification is a fundamental task in natural language processing (NLP) that involves the automatic assignment of textual documents, regardless of length, to predefined categories (Taha et al., 2024). With the exponential growth of digital textual data, the significance of this task has increased considerably. Efficient classification methods have become increasingly valuable in both academic research and industrial applications, while the complexity of classification has also escalated (Collins et al., 2018). The field has evolved from basic sentiment analysis of entire texts to more advanced approaches such as multi-label classification and hierarchical classification of long documents(Wang et al., 2023b). These advancements have led to greater demands for customization and higher classification efficiency, particularly in industrial applications. In scenarios with abundant labeled data, certain encoder-only models can be quickly trained and deployed. However, in cases with limited or no labeled samples, BERT-like models (Devlin et al., 2018) often struggle to achieve satisfactory performance. For localized industrial deployments, achieving optimal results typically requires large-scale models like Llama-3.1-70B-Instruct, which demands significant GPU resources. This makes their widespread use in industrial text classification less practical compared to models like BERT, as dedicating high-memory GPUs solely for classification is often infeasible.

As a consequence, this study aims to investigate the limitations of transformer models, with a particular focus on the performance of Small Language Models (SLMs) and exploring best practices to address industrial text classification challenges effectively. To achieve this, we center our research around three key questions:

- **RQ1**: Can SLMs perform classification without any task-specific training?
- **RQ2**: What are the strengths and limitations of various methods applied to text classification using SLMs?
- **RQ3**: How can the trade-off between computational efficiency and classification performance be optimized, and how can SLMs be more effectively deployed in practice?

The remainder of this paper is organized as follows. Section 2 reviews related work and text clas-

[1] https://github.com/DobricLilujun/agentCLS/

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 6: Industry Track), pages 971–982

sification approaches; Section 3 presents the experimental methodology applied to industrial datasets; Section 4 provides a detailed analysis of the results; and Section 5 concludes the study with key findings and future directions.

2 Related Work

2.1 Different Types of Transformers

Transformers have demonstrated remarkable efficacy in classification tasks (Zhao et al., 2023), primarily due to their ability to comprehend multilingual texts and generate linguistically nuanced and stylistically personalized outputs (Zhao et al., 2024). Across encoder-decoder architectures of LLMs, three primary paradigms emerge:

1. The sequence to sequence framework (Naveed et al., 2024) maps an input sequence to a hidden space, enabling various downstream tasks by appending additional components of the neural network, such as the classifier head. This framework encompasses a range of models, including T5 (Raffel et al., 2019), and BART (Lewis et al., 2019), which have been extensively employed in applications such as machine translation and text summarization.

2. Encoder-only models, such as BERT (Devlin et al., 2019), are designed to focus on understanding and processing input text to extract meaningful representations. They demonstrated superior performance in tasks such as named entity recognition (NER: (Liu et al., 2021)), surpassing other state-of-the-art (SOTA) models. Additionally, models like RoBERTa (Robustly Optimized BERT (Liu et al., 2019)) and ModernBERT (Warner et al., 2024) (149M parameters) are optimized for lightweight deployment due to their smaller size.

3. Decoder-only models, with a more compact structure (Gao et al., 2022), extract linguistic knowledge from large corpora and generate translations auto-regressively. They have shown strong performance in text generation (Hendy et al., 2023; Brown et al., 2020a). The rapid growth of language models is driven by decoder-only architectures, known for their versatility, reasoning, and problem-solving abilities. Their decoding mechanism allows them to handle nearly all NLP tasks. Notable examples include Meta's Llama series (Touvron et al., 2023) and Google's Gemma series (Team et al., 2024), along with newly released reasoning models such as DeepSeek (Liu et al., 2024), which enhance logical problem-solving by leveraging hard-coded reasoning chains.

2.2 Background

The earliest systematic studies on text classification included probabilistic model-based methods such as Naive Bayes (Joachims, 1998). He was the first to apply Support Vector Machines (SVM) to text classification tasks. With the advent of neural networks, early research primarily utilized embeddings and simple neural network architectures for text classification. Subsequently, (Kim, 2014) proposed a convolutional neural network-based approach for text classification, significantly improving classification performance at sentence-level feature extraction. In addition, classification models based on Recurrent Neural Networks (RNNs) have also shown remarkable performance, demonstrating greater robustness under distribution shifts (Yogatama et al., 2017). However, they still struggle to effectively handle complex scenarios in classification tasks such as long texts(Du et al., 2020). Later, the emergence of attention architectures led to extensive experimentation in various applications.

The advent of transformer-based architectures in 2018, particularly BERT, brought about a paradigm shift in natural language classification tasks, resulting in considerable performance enhancements (Kora and Mohammed, 2023; Pawar et al., 2024). Some knowledge distillation approaches (Nityasya et al., 2022) have also been explored to compress large BERT models into smaller, faster, and more efficient versions that can retain up to 97% of the original model's classification performance. This observation has motivated our interest in directly using small open source models, which often achieve performance comparable to that of large models after distillation (Zhu et al., 2024). For long text classification, specialized bidirectional models such as Longformer (Beltagy et al., 2020) and LegalBERT (Chalkidis et al., 2020) have emerged in recent years, capable of handling ultra-long documents and showing excellent performance. Nevertheless, their adoption in industry remains limited, primarily due to substantial GPU resource requirements and the need for custom CUDA kernels to support sliding-window attention, which also introduces compatibility challenges with the Huggingface Transformers framework.

Regarding SLMs, (Lepagnol et al., 2024) explored the zero-shot text classification capabilities of small language models, highlighting their potential in classification tasks. Recent advance-

ments in text classification have primarily focused on two key approaches: prompt engineering and **S**upervised **F**ine **T**uning(SFT).

Prompt engineering involves crafting well-structured inputs to guide LLMs in producing more personalized responses. Recent research has shown that sophisticated prompt engineering techniques can sometimes compete with or even outperform fine-tuned models(Sahoo et al., 2025). In both industry and academia, models such as BERT and Llama are commonly used to assess downstream tasks. Nevertheless, there is a notable absence of extensive comparative research on various prompt engineering and SFT techniques for SLMs, aimed at identifying the most effective practices for industrial applications. Furthermore, publicly available datasets are frequently subject to inherent biases resulting from prior exposure during pre-training, which means that models being evaluated may have already been trained on portions of the test set, thereby introducing the possibility of biases.

3 Experiments On Industrial Cases

3.1 Methods

To address the challenges outlined in the related work, we trained models on datasets of varying difficulty levels, including a proprietary, real-world industrial dataset. Regarding model selection, we primarily focused on decoder-only architectures while incorporating a subset of encoder-only models for validation. In addition, we explore various prompt engineering techniques and examine the impact of different prompt tuning methods, focusing on classification task.

Table 1 presents an overview of different templates and prompt strategies, where all prompts are designed to enforce a structured output format. The base prompt closely resembles a direct label mapping approach, where the model outputs the label it deems most appropriate. Few-shot prompts extend this by incorporating examples alongside descriptions. Furthermore, Chain-of-Thought (COT) and Chain-of-Draft (COD) prompts serve to evaluate the reasoning capabilities of SLMs to some extent.

In the training process, we primarily employ three distinct methods: 1) SFT, which modifies only the weights of the classification heads added at the end of the model using labeled data; 2) Soft Prompt Tuning (SPT), which involves optimizing input prompts to continuously guide the model towards correct behavior based on labeled data; and 3) Prefix Tuning (PT), which incorporates a learnable prefix tensor into each attention layer.

These approaches enhance the model's classification performance while keeping most of the model weights frozen, which are widely used in industrial use cases.

Methods Types	Methods	Reference
Prompt Engineering	Base Prompts	(Ye et al., 2024)
Prompt Engineering	Few-Shot Prompts	(Brown et al., 2020b)
Prompt Engineering	Chain-of-Thought (COT)	(Wei et al., 2022)
Prompt Engineering	Self-consistency COT	(Wang et al., 2023a)
Prompt Engineering	Chain-of-Draft (COD)	(Xu et al., 2025)
Fine Tuning	Supervised Fine-tuning	(Parthasarathy et al., 2024)
Soft Prompt Tuning	Parameter Efficient Fine-tuning	(Lester et al., 2021)
Prefix Tuning	Parameter Efficient Fine-tuning	(Li and Liang, 2021)

Table 1: Classification methods based on the transformer architecture investigated in this study.

3.2 Datasets

In this study, we primarily utilized three datasets for our experiments, as shown in Table 2. First, we used the EURLEX57K dataset (Chalkidis et al., 2019), which was released by (Chalkidis et al., 2019) and contains 57,000 new legislative documents. We adopted the document type as the classification label, which includes Regulation, Decision, and Directive. Additionally, we employed the Long Document Dataset (He et al., 2019), a relatively more challenging dataset that consists of a large amount of literature text extracted from PDFs, categorized into 11 different classes, such as cs.AI (Artificial Intelligence), cs.CE (Computational Engineering), and so on. The main difficulty lies in the length of the documents and the challenge of classifying them into over 11 labels, which significantly increases the complexity of the task.

In addition, we possess a proprietary, closed-source dataset derived from email correspondence between our partner company and its clients. The primary business requirement is to analyze historical interactions with each client—written in a mixture of English, French, German, and Luxembourgish—to determine whether the most recent emails in the thread are reminders. Consequently, the task involves identifying the optimal position within the text and determining whether that position conveys a "reminder" meaning, resulting in a binary labeling scheme. It also requires a comprehensive understanding of long email threads written in mixed languages, including low-resource ones, and making a final decision based on the contextual

Dataset	Abbreviation	Words / D	# Train	# Validation	# Labels	Subject
EURLEX57K	EUR	720	3039	900	3	EU Legislation
Long Document Dataset	LDD	10378	15682	3300	11	Academy
Insurance Email	IE	724	2015	1000	2	Email History

Table 2: The table below presents the statistics of the three datasets used in our experiments. `Words/D` denotes the average number of words per document, `#Train` represents the number of training samples, `#Validation` refers to the number of validation samples, and `#Labels` indicates the number of unique labels in the dataset. Each dataset corresponds to a different domain of text. Notably, the LDD dataset exhibits a larger number of labels and a higher word count per document, which increases the difficulty of the classification task.

meaning at the identified position.

The main challenges associated with this dataset are: 1. Semantic decision-making is heavily based on the content of the most recent emails exchanged with the client, with older emails primarily serving as background context. This characteristic places the most crucial textual information towards the beginning of the sequence, which contrasts with typical datasets where classification decisions are based on the overall semantics of the entire text. 2. The dataset inherently contains long texts with uneven length distributions with information extracted from images. All nontextual data has been processed using OCR to extract textual content. By incorporating this real-world industrial dataset, we improve the persuasiveness and robustness of our model and methods evaluations.

3.3 SLM Models

Fine-tuning on classification typically refers to the application of transfer learning when a task is associated with a certain amount of labeled data. This approach capitalizes on the semantic representation capabilities of a pre-trained model by incorporating a lightweight linear layer for classification, denoted as classification heads. During training, the model parameters are kept frozen, while only the newly introduced classification network is optimized to achieve the classification objective. In this study, we adopt SLMs including **Llama-3.2-1B**, **Llama-3.2-1B** and **ModernBERT-base** as the foundational models. Additionally, Llama-3.3-70B-Instruct and GPT-4o mini are used as foundation model baselines for performance comparison. More details are shown in the Appendix A.

3.4 Experimental Settings & Metrics

We employ **Accuracy**, **F1-score** as performance metrics to evaluate different methods across all models. For the **fine-tuning** approach, we standardize the learning rate to **1e-6** and train all models for **10 epochs** to ensure controlled variable conditions. To evaluate the efficiency of different methods and analyze resource usage, we track GPU hours (GHs) and GPU RAM hours (GRHs). GPU hours represent the total computational time a model utilizes GPU clusters, while GPU RAM hours quantify cumulative memory consumption during execution. These metrics provide insights into computational cost and resource efficiency. As prompt engineering primarily affects inference time and pretraining duration is unknown, we measure only its inference stage.

The prompts used from different strategy methods were well designed as shown in the appendix B. When it comes to self-consistency COT, several different paths of thinking should be set, and in this study, we explicitly set it to 3. To control for variables, we standardize the batch size to 8 and set the number of training epochs to 10, selecting the checkpoint with the lowest evaluation loss. For both SPT and PT, we configure the number of virtual tokens to 128. In general, all models are trained with a maximum context length of 4096 tokens.

4 Results

4.1 Main Performance

Additional models were used to validate the test set in order to provide a reference performance for State-of-the-Art (SOTA) models. However, ChatGPT was not evaluated on the IE dataset due to potential data leakage concerns. In contrast, Llama-3.3-70B-Instruct was run locally, allowing for GPU resource estimation and comprehensive metric evaluation. As presented in Table 3, the highest prompt engineering performance was achieved by ChatGPT-o1 mini. Meanwhile, in the IE dataset, which serves as our industrial database, an accuracy score of 0.800 was achieved by Llama-3.3-70B-Instruct. Regarding SLMs, we

Table 3: The main results include validation performance on three datasets under different prompt engineering and SFT conditions. ACC represents accuracy, GH indicates GPU hours, and GRH refers to GPU RAM hours for memory usage. Prefix-tuning is unsupported on ModernBERT-base due to model structure incompatibility.

Methods Type	Methods	Models	EUR				LDD				IE			
			ACC ↑	F1 ↑	GH ↓	GRH ↓	ACC ↑	F1 ↑	GH ↓	GRH ↓	ACC ↑	F1 ↑	GH ↓	GRH ↓
		GPT-4o-mini	**0.833**	**0.767**	N/A	N/A	0.682	0.698	N/A	N/A	N/A	N/A	N/A	N/A
		Llama-3.3-70B-Instruct	0.398	0.287	0.157	26.443	0.500	0.333	0.188	31.651	**0.800**	**0.799**	0.517	86.772
Prompt Engineering	Base prompt	Llama-3.2-1B-Instruct	0.330	0.319	0.010	0.263	0.186	0.159	0.775	19.981	0.500	0.370	0.040	1.034
		Llama-3.2-3B-Instruct	0.346	0.220	0.030	1.167	0.314	0.301	0.313	12.385	0.500	0.333	0.047	1.847
	Few-shot Prompt	Llama-3.2-1B-Instruct	0.387	0.377	0.022	0.578	0.132	0.113	0.574	14.804	0.488	0.338	0.038	0.972
		Llama-3.2-3B-Instruct	0.506	0.499	0.024	0.931	0.471	0.491	0.136	5.376	0.500	0.333	0.044	1.756
	Chain-of-Thought	Llama-3.2-1B-Instruct	0.463	0.438	0.181	4.659	0.181	0.167	1.248	32.171	0.501	0.339	0.189	4.873
		Llama-3.2-3B-Instruct	0.341	0.293	0.427	16.906	0.365	0.334	0.722	28.544	0.491	0.401	0.519	20.538
	Self-consistency COT	Llama-3.2-1B-Instruct	0.433	0.411	0.582	14.997	0.178	0.168	4.231	109.086	0.500	0.333	0.597	15.392
		Llama-3.2-3B-Instruct	0.419	0.338	0.982	38.836	0.167	0.168	2.321	91.821	0.510	0.333	0.991	39.192
	Chain-of-Draft	Llama-3.2-1B-Instruct	0.408	0.395	0.061	1.560	0.226	0.226	0.376	9.702	0.499	0.336	0.105	2.705
		Llama-3.2-3B-Instruct	0.351	0.332	0.055	2.191	0.425	0.437	0.390	15.431	0.499	0.335	0.113	4.458
Supervised Fine-Tuning	Soft Prompt Tuning (SPT)	Llama-3.2-1B-Instruct	0.643	0.533	0.848	22.977	0.442	0.429	4.589	124.827	0.506	0.381	0.594	15.914
		Llama-3.2-3B-Instruct	0.641	0.524	2.926	169.812	0.136	0.135	8.303	481.701	0.526	0.475	1.396	76.384
		ModernBERT-base	0.332	0.171	0.533	11.903	0.207	0.184	1.374	26.394	0.500	0.333	0.566	12.667
	Prefix Tuning (PT)	Llama-3.2-1B-Instruct	0.330	0.266	1.580	42.947	0.112	0.107	7.826	212.826	0.502	0.371	0.463	12.530
		Llama-3.2-3B-Instruct	0.320	0.300	1.360	83.864	0.128	0.117	16.532	1040.624	0.588	0.536	2.999	172.257
		ModernBERT-base	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
	Fine-tuning (FT)	Llama-3.2-1B-Instruct	**0.999**	**0.999**	0.508	13.813	**0.892**	**0.890**	1.698	40.631	**0.865**	**0.863**	1.008	27.474
		Llama-3.2-3B-Instruct	**0.998**	**0.998**	1.750	92.118	**0.904**	**0.903**	3.764	226.869	**0.960**	**0.960**	1.949	123.109
		ModernBERT-base	0.333	0.167	0.132	1.849	0.810	0.811	1.762	24.018	0.514	0.408	0.104	1.476

found the results particularly intriguing, especially in the context of prompt engineering. Given the relatively small size of these models, we did not expect them to achieve high performance. The final results for the 1B and 3B models aligned with our expectations, performing roughly at the level of random guessing. Interestingly, both the 3B and even the 1B models demonstrated a strong preference for few-shot prompting. This approach led to an improvement of over 10% compared to the base prompt on the EUR and LDD datasets, highlighting the importance of few-shot learning in the application of SLMs, as also emphasized in (Brown et al., 2020a). Furthermore, we observed that both COD and COT provided limited improvements. In fact, on the LDD dataset, COD performed worse than COT and was nearly on par with the base prompt. Therefore, the use of COD and COT is not recommended as a solution for classification tasks in SLMs.

In the context of SFT, we observed that SPT outperformed prefix tuning by a significant margin, although it also required substantially more training time. Prefix tuning introduces a trainable part at every layer within the model, whereas SPT only incorporates a soft prompt at the input level. It is possible that SPT better preserves the original language understanding of the model, as it does not alter the overall architecture. In contrast, prefix-tuning's modifications to the attention structure may disrupt the model's inherent linguistic comprehension. Additionally, supervised fine-tuning, which adds a classification head to the end of the model, demonstrated the highest overall performance. Notably, ModernBERT achieved a performance of approximately 0.810 of accuracy on the LDD dataset while requiring less training time and GPU memory, making it a promising candidate for academic English text classification. Limited exposure to French, other multilingual languages, and domain-specific corpora during training (Warner et al., 2024) led to weaker performance on the IE dataset (primarily in French) and EUR (a domain-specific corpus).

4.2 Exploratory Results

4.2.1 Does data matter?

Experiments were conducted to examine the impact of data volume, primarily using SFT, the best method in our research scope. We randomly selected 50, 150, and 1500 samples as training data. The results, as shown in Figure 1, indicate that on the relatively simple EU dataset, the model can achieve good performance even with a small amount of data after multiple training iterations, with the primary determinant of performance being the model itself. However, for more complex and challenging datasets such as LDD and IE, the amount of training data directly determines performance. Furthermore, we observed that models of different sizes exhibit only minor differences in classification performance. Therefore, data volume has a direct impact on classification performance in difficult datasets, which ultimately defines the performance bottleneck instead of the model itself.

4.2.2 Larger Models?

As observed in Table 4, the performance gains from larger models are also minimal. For example, in the

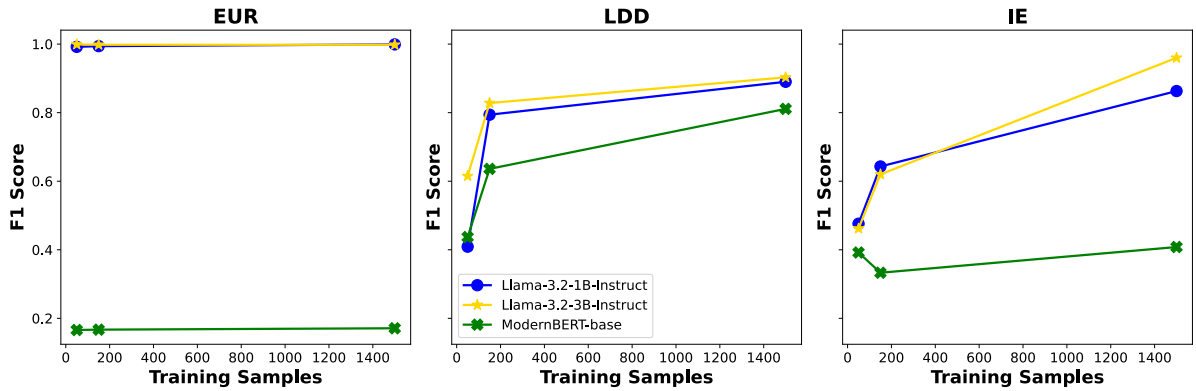

Figure 1: Impact of Data Volume on Model Performance.

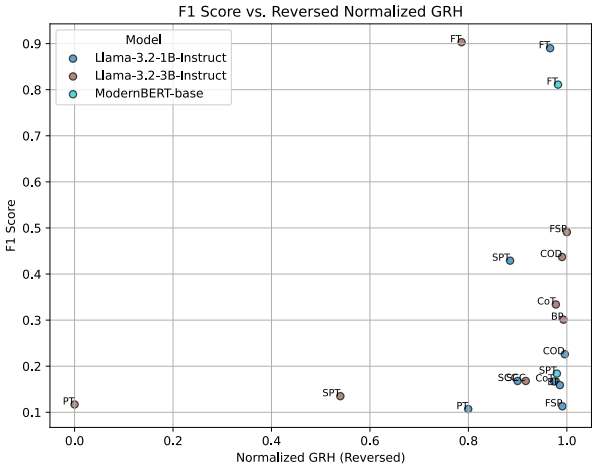

Figure 2: Reversed efficiency on LDD datasets

Table 4: This table compares the performance of ModernBERT-Base ("Base") and ModernBERT-Large ("Large") on the same dataset.

Models	EUR		LDD		IE	
	ACC	F1	ACC	F1	ACC	F1
Base	0.333	0.167	0.810	0.811	0.514	0.408
Large	0.333	0.168	0.828	0.829	0.539	0.424

LDD dataset, ModernBERT-large only improves by about 2% over the base model. In particular, on the EUR, larger models do not show significant performance gains. This is highly related to the domain relevance of the model's pre-training data. For example, in the ModernBERT paper, it is mentioned that the model is trained on a large amount of academic English data, which leads to high performance on LDD. The IE dataset, which includes French, German, and English, results in accuracy around 0.5. In the EUR dataset, performance is especially poor and increasing the model size does not improve results. This shows that SFT models for classification do not enhance semantic understanding, but guide comprehension and classification. Thus, the model should be thoroughly investigated before industrial deployment, and decoder-only SLMs are sufficient for classification tasks if they excel at understanding the dataset's domain knowledge.

4.2.3 Deeper Header?

In our primary experimental setting, we adhere to the definition of a "Header" as implemented in the Transformers library, referring to a single linear layer serving as the classification head. To further explore potential improvements using different levels of header, we experimented with replacing the standard single-layer header with a multi-layer linear architecture incorporating ReLU activations. Specifically, we constructed classification heads with 2 to 5 linear layers (hidden dimension = 256) and fine-tuned Llama-3.2-1B-Instruct model accordingly. As shown in Table 5, the results indicate that increasing the depth of the classification head yields only marginal gains, with performance plateauing beyond three layers. These findings suggest that deeper header architectures offer limited benefit in enhancing the classification accuracy or F1 score in this context.

# Layers	1	2	3	4	5
ACC	0.89	0.91	0.92	0.91	0.91
F1	0.89	0.91	0.92	0.91	0.91

Table 5: Impact of classification head depth on performance, evaluated on the LDD dataset using Llama-3.2-1B-Instruct. "# Layers" refers to the number of stacked linear layers in the classification head.

4.3 Efficiency

We particularly focus on model efficiency from training to inference, with a specific emphasis on VRAM usage, which is the primary limiting factor for deployment in industrial settings. As shown in Figure 2, the x-axis represents the reverse normalized GRH score, while the y-axis represents the F1 Score. Therefore, points located further towards the top-right indicate higher efficiency. It is clear that the three FT models exhibit the highest efficiency, while the prompt engineering methods, although very efficient in terms of GPU RAM usage, significantly lag behind in performance. Therefore, for local deployment, fine-tuning of SLMs is the optimal approach for enhancing both efficiency and accuracy. Additionally, we can observe that from 1B to 3B models, there is only a marginal improvement in model accuracy, while GPU time consumption increases. Hence, fine-tuning the 1B model could be the optimal solution when considering efficiency.

4.4 Research Questions

For RQ1, "*Can SLMs perform classification without any task-specific training?*", we found that text classification using SLMs faces several key challenges. Smaller models tend to exhibit limited logical reasoning capabilities and are more susceptible to generating hallucinations while encountering long text. Moreover, the performance ceiling is strongly influenced by the amount of available training data, while the intrinsic properties of the SLMs themselves also play a critical role in shaping classification outcomes.

Regarding RQ2, "*What are the strengths and limitations of various methods applied to text classification using SLMs?*", prompt engineering can demonstrate substantial flexibility and customization; however, its performance on SLMs remains significantly limited. Notably, various prompt engineering strategies, such as COT or COD, sometimes negatively influence model performance. If employing prompts engineering on SLMs is necessary, it is recommended to utilize few-shot prompting rather than COT or COD as shown in Table 3. In contrast, SFT shows excellent performance on decoder-only models, whereas SPT and PT achieve moderate effectiveness. Nevertheless, both approaches generally yield superior results compared to prompt engineering.

For RQ3, "*How can the trade-off between computational efficiency and classification performance be optimized, and how can SLMs be more effectively deployed in practice?*", we found that although training the model consumes significant GPU resources, the SLMs are essentially unusable in their current form due to the lack of inference capability. We also tested Llama-3.3-70B-Instruct, which, although capable of achieving 80% accuracy in IE, still produces uncertain output. Therefore, FT transformers remains the only viable solution on SLMs which is portable and light weight. Finally, the limited capacity of SLMs creates a bottleneck on performance and the amount of labeled data also remains a key limitation. For real application, it is crucial to focus not only on data quality but also on the model's inherent characteristics, such as multilingual comprehension. If resources are relatively abundant, opting for decoder-only models such as the Llama series would be a better choice, which has a good support on both languages and different domain knowledge.

5 Conclusion

In this study, we present a comprehensive evaluation of lightweight models on text classification. We systematically investigate nearly all major approaches, including prompt engineering and supervised fine-tuning. Our experimental setup spans three benchmark datasets, including a real-world industrial scenario involving email history classification.

Our findings indicate that while the volume of training data has a significant impact on classification performance, the model's intrinsic understanding of domain-specific textual content also plays a critical role and can become a major bottleneck in achieving high accuracy. Furthermore, we observe that increasing the size of the model or the depth of the classification head yields only marginal performance improvements.

Finally, we analyze the VRAM efficiency of different models across the entire classification pipeline, offering practical insights into their suitability for real-world deployment. These results are particularly relevant for industrial applications, where both high precision and computational efficiency are essential, providing guidance in selecting the appropriate models, classification strategies, and computational resources to optimize under real-world constraints.

6 Limitations

This paper comprehensively evaluates Transformer-based classification methods on industrial datasets, providing valuable insights for real-world deployment. However, the impact of the number of virtual tokens in SFT has not been thoroughly explored. It is possible that increasing the number of virtual tokens could yield better results.

Furthermore, we observed that the performance of the ModernBERT-base model on the EUR dataset is particularly poor. However, due to the limited understanding of its pretraining data volume and composition, further research is needed to analyze the language understanding capabilities of ModernBERT-base. Since our training does not enhance the model's intrinsic language understanding, the model's inherent linguistic comprehension plays a crucial role in classification tasks. Additionally, more SLMs should be evaluated, such as Gemma-2B, to obtain a more comprehensive understanding of the results.

Acknowledgments

This research has benefited from the collaboration and support of our industrial partner, academic institutions, and contributors. We thank the "AI & Data Studio" team for their insights, guidance, and provision of essential computational resources (NVIDIA H100 GPUs), which were crucial for the experiments. The mentorship from faculty members and feedback from postdoctoral researchers have greatly improved the study's rigor. Additionally, we confirm that no AI-generated text was used in preparing this manuscript. Our draft complies with the European General Data Protection Regulation (GDPR) data policy.

References

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *CoRR*, abs/2004.05150.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020a. Language models are few-shot learners. *Preprint*, arXiv:2005.14165.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020b. Language models are few-shot learners. *CoRR*, abs/2005.14165.

Ilias Chalkidis, Emmanouil Fergadiotis, Prodromos Malakasiotis, and Ion Androutsopoulos. 2019. Large-scale multi-label text classification on EU legislation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6314–6322, Florence, Italy. Association for Computational Linguistics.

Ilias Chalkidis, Manos Fergadiotis, Prodromos Malakasiotis, Nikolaos Aletras, and Ion Androutsopoulos. 2020. LEGAL-BERT: The muppets straight out of law school. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2898–2904, Online. Association for Computational Linguistics.

Edward Collins, Nikolai Rozanov, and Bingbing Zhang. 2018. Evolutionary data measures: Understanding the difficulty of text classification tasks. *CoRR*, abs/1811.01910.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *Preprint*, arXiv:1810.04805.

Jinhua Du, Yan Huang, and Karo Moilanen. 2020. Pointing to select: A fast pointer-LSTM for long text classification. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6184–6193, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Yingbo Gao, Christian Herold, Zijian Yang, and Hermann Ney. 2022. Is encoder-decoder redundant for neural machine translation? *Preprint*, arXiv:2210.11807.

Jun He, Liqun Wang, Liu Liu, Jiao Feng, and Hao Wu. 2019. Long document classification from local word glimpses via recurrent attention learning. *IEEE Access*, 7:40707–40718.

Amr Hendy, Mohamed Abdelrehim, Amr Sharaf, Vikas Raunak, Mohamed Gabr, Hitokazu Matsushita,

Young Jin Kim, Mohamed Afify, and Hany Hassan Awadalla. 2023. How good are gpt models at machine translation? a comprehensive evaluation. *Preprint*, arXiv:2302.09210.

Thorsten Joachims. 1998. Text categorization with support vector machines. *Proc. European Conf. Machine Learning (ECML'98)*.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *CoRR*, abs/1408.5882.

Rania Kora and Ammar Mohammed. 2023. A comprehensive review on transformers models for text classification. In *2023 International Mobile, Intelligent, and Ubiquitous Computing Conference (MIUCC)*, pages 1–7.

Pierre Lepagnol, Thomas Gerald, Sahar Ghannay, Christophe Servan, and Sophie Rosset. 2024. Small language models are good too: An empirical study of zero-shot classification. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 14923–14936, Torino, Italia. ELRA and ICCL.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. *CoRR*, abs/2104.08691.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2019. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *CoRR*, abs/1910.13461.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *CoRR*, abs/2101.00190.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. 2024. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Zihan Liu, Feijun Jiang, Yuxiang Hu, Chen Shi, and Pascale Fung. 2021. NER-BERT: A pre-trained model for low-resource entity tagging. *CoRR*, abs/2112.00405.

Humza Naveed, Asad Ullah Khan, Shi Qiu, Muhammad Saqib, Saeed Anwar, Muhammad Usman, Naveed Akhtar, Nick Barnes, and Ajmal Mian. 2024. A comprehensive overview of large language models. *Preprint*, arXiv:2307.06435.

Made Nindyatama Nityasya, Haryo Akbarianto Wibowo, Rendi Chevi, Radityo Eko Prasojo, and Alham Fikri Aji. 2022. Which student is best? a comprehensive knowledge distillation exam for task-specific bert models. *Preprint*, arXiv:2201.00558.

Venkatesh Balavadhani Parthasarathy, Ahtsham Zafar, Aafaq Khan, and Arsalan Shahid. 2024. The ultimate guide to fine-tuning llms from basics to breakthroughs: An exhaustive review of technologies, research, best practices, applied research challenges and opportunities. *Preprint*, arXiv:2408.13296.

Sachin Pawar, Nitin Ramrakhiyani, Anubhav Sinha, Manoj Apte, and Girish Palshikar. 2024. Why generate when you can discriminate? a novel technique for text classification using language models. In *Findings of the Association for Computational Linguistics: EACL 2024*, pages 1099–1114, St. Julian's, Malta. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *CoRR*, abs/1910.10683.

Pranab Sahoo, Ayush Kumar Singh, Sriparna Saha, Vinija Jain, Samrat Mondal, and Aman Chadha. 2025. A systematic survey of prompt engineering in large language models: Techniques and applications. *Preprint*, arXiv:2402.07927.

Kamal Taha, Paul D. Yoo, Chan Yeun, Dirar Homouz, and Aya Taha. 2024. A comprehensive survey of text classification techniques and their research applications: Observational and experimental insights. *Computer Science Review*, 54:100664.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, et al. 2024. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models. *Preprint*, arXiv:2302.13971.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023a. Self-consistency improves chain of thought reasoning in language models. *Preprint*, arXiv:2203.11171.

Yue Wang, Dan Qiao, Juntao Li, Jinxiong Chang, Qishen Zhang, Zhongyi Liu, Guannan Zhang, and Min Zhang. 2023b. Towards better hierarchical text classification with data generation. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 7722–7739, Toronto, Canada. Association for Computational Linguistics.

Benjamin Warner, Antoine Chaffin, Benjamin Clavié, Orion Weller, Oskar Hallström, Said Taghadouini, Alexis Gallagher, Raja Biswas, Faisal Ladhak, Tom Aarsen, Nathan Cooper, Griffin Adams, Jeremy Howard, and Iacopo Poli. 2024. Smarter, better, faster, longer: A modern bidirectional encoder for fast, memory efficient, and long context finetuning and inference. *Preprint*, arXiv:2412.13663.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed H. Chi, Quoc Le, and Denny Zhou. 2022. Chain of thought prompting elicits reasoning in large language models. *CoRR*, abs/2201.11903.

Silei Xu, Wenhao Xie, Lingxiao Zhao, and Pengcheng He. 2025. Chain of draft: Thinking faster by writing less. *Preprint*, arXiv:2502.18600.

Qinyuan Ye, Maxamed Axmed, Reid Pryzant, and Fereshte Khani. 2024. Prompt engineering a prompt engineer. *Preprint*, arXiv:2311.05661.

Dani Yogatama, Chris Dyer, Wang Ling, and Phil Blunsom. 2017. Generative and discriminative text classification with recurrent neural networks. *Preprint*, arXiv:1703.01898.

Yang Zhao, Jiajun Zhang, and Chengqing Zong. 2023. Transformer: A general framework from machine translation to others. *Machine Intelligence Research*, 20(4):514–538.

Yiran Zhao, Wenxuan Zhang, Guizhen Chen, Kenji Kawaguchi, and Lidong Bing. 2024. How do large language models handle multilingualism? *Preprint*, arXiv:2402.18815.

Xunyu Zhu, Jian Li, Yong Liu, Can Ma, and Weiping Wang. 2024. A survey on model compression for large language models. *Transactions of the Association for Computational Linguistics*, 12:1556–1577.

A Experiment Details

In this study, we examine three distinct models in all text classification methods, along with several larger models, as presented in Table 6.

We primarily utilized the AutoModelForSequenceClassification from Transformers to train our model for classification tasks. The main principle involves adding a linear mapping head for model classification, where the input dimension corresponds to the output dimension of the LLMs. For instance, in the case of Llama-3.2-1B-Instruct, its output features are 2048, which serve as the input features for the linear mapping head. The output features' dimension, on the other hand, corresponds to the number of classification labels.

During training, the orignal weights of the pretrained model are kept frozen, while only the newly introduced classification head is optimized to achieve the final classification objective. In this study,the optimization process is guided by **BCEWithLogitsLoss**, which serves as the loss function throughout the training.

B Prompt Example

The base prompt template for the EUR dataset is shown below. Basically, it requires the models to provide three labels with a classification answer at the end, following a separator ####.

> Return the classification answer after a separator ####. Do not return any preamble, explanation, or reasoning.
> Classify the **input** text into one of the following categories based on the descriptions provided, and explicitly provide the output classification at the end.
> Categories: 1. **Decision** - Choose this category if the text involves making a choice or selecting an option. 2. **Directive** - Use this category if the text instructs or commands an action. 3. **Regulation** - Appropriate for texts that stipulate rules or guidelines.
> <<<START OF INPUT>>>
> {input}
> <<<END OF INPUT>>>

In the LDD dataset, there will be 11 labels, each representing the category of an academic subject, while the input will be the document version of academic articles. The base prompt template for the LDD dataset is shown below.

Model	Ctx Len	Release	VRAM Train(GB)	VRAM Infer(GB)
Llama-3.2-1B-Instruct	128k	Sep 25, 2024	27.36	25.78
Llama-3.2-3B-Instruct	128k	Sep 25, 2024	65.52	39.55
ModernBERT-base	8,192	Dec 19, 2024	12.82	1.72
ModernBERT-large	8,192	Dec 19, 2024	25.48	3.35
Llama-3.3-70B-Instruct	128k	Mar 14, 2025	N/A	168
GPT4o-mini	32k	Jul 18, 2024	N/A	N/A

Table 6: Table of Model Specifications with GPU Memory Requirements. In this table, "Ctx" Len refers to the maximum context length, "Release" denotes the model's release date, "VRAM Train (GB)" indicates the amount of VRAM required for training each model with a batch size of 8 and a context length of 4096, and "VRAM Infer (GB)" specifies the VRAM needed to load the model and perform inference.

Return the classification answer after a separator ####. Do not return any preamble, explanation, or reasoning.
Classify the **input** text into one of the following categories based on the descriptions provided, and explicitly provide the output classification at the end.
Categories:
- **cs.AI**: Involves topics related to Artificial Intelligence. - **cs.CE**: Related to Computational Engineering. - **cs.CV**: Pertains to Computer Vision. - **cs.DS**: Concerns Data Structures. - **cs.IT**: Deals with Information Theory. - **cs.NE**: Focuses on Neural and Evolutionary Computing. - **cs.PL**: Involves Programming Languages. - **cs.SY**: Related to Systems and Control. - **math.AC**: Pertains to Commutative Algebra. - **math.GR**: Involves Group Theory. - **math.ST**: Related to Statistics Theory.
<<<START OF INPUT>>>
{input}
<<<END OF INPUT>>>

In the real-world IE dataset, we used authentic email history records from the industry as the data source, with labels manually identified by experts from our industrial partners.

Particularly of interest, we consider Self-consistency COT method to further validate the model's logical reasoning ability. In this approach, the model first generates three different reasoning chains using a COT prompt. Then, the reasoning chains, along with the question, are presented to the model, which selects the most consistent reasoning chain and ultimately identifies the correct classification label.

Return the classification answer after a separator ####. Do not return any preamble, explanation, or reasoning.
You will be provided three thinking paths for answering the text classification question, and the conclusions from the three paths will be compared. If two or more paths arrive at the same classification result, that will be selected as the most consistent answer; if all three paths differ, answer with the most plausible classification based on the overall reasoning. The self consistency prompt template is shown below.
Question:
{question}
Path 1: {path 1}
Path 2: {path 2}
Path 3: {path 3}

C Additional Results

We conducted a comprehensive evaluation of various prompt engineering techniques on the relatively large-scale model, Llama-3.1-8B-Instruct, with the aim of achieving competitive performance in comparison to other SLMs. As shown in Table 7, despite leveraging an 8-billion parameter model, attaining satisfactory accuracy proved challenging. Notably, the performance improvements achieved through COT and COD strategies were significantly more substantial, markedly outperforming those obtained via Few-shot Prompting. This suggests that for larger models, COT and COD methodologies should be prioritized, whereas few-shot prompting remains the optimal approach for smaller models.

Methods	Models	EUR		LDD		IE	
		ACC	F1	ACC	F1	ACC	F1
	GPT4o-mini	**0.833**	**0.767**	0.682	0.698	-	-
	Llama-3.3-70B-Instruct	0.398	0.287	0.500	0.333	0.800	0.799
Base prompt	Llama-3.1-8B-Instruct	0.216	0.193	0.554	0.596	0.500	0.333
Few-shot Prompt	Llama-3.1-8B-Instruct	0.494	0.460	0.456	0.490	0.530	0.408
Chain-of-Thought	Llama-3.1-8B-Instruct	0.503	0.465	0.650	0.656	0.514	0.423
Self-consistency COT	Llama-3.1-8B-Instruct	0.568	0.528	0.231	0.248	0.500	0.333
Chain-of-Draft	Llama-3.1-8B-Instruct	0.422	0.375	**0.622**	**0.635**	0.498	0.332

Table 7: This table presents the performance results of all prompt engineering tests conducted on the larger-scale model, Llama-3.1-8B-Instruct.

Furthermore, it is important to highlight the poor performance of Self-Consistency COT on the LDD dataset. This limitation is primarily attributed to the excessively long text sequences within LDD, which induce hallucination effects in the model. Given that Self-Consistency COT involves generating three separate reasoning chains, the input length increases considerably, leading to a noticeable degradation in performance. In contrast, COD demonstrates comparable performance to GPT-4o-mini on the LDD dataset, indicating its potential as a promising area for further investigation.