Small Language Models as an Efficient Alternative to Large Language Models in Domain-Specific AI Systems

Author: Temirlan Mukatayev
Affiliation: True Masters. Astana, Kazakhstan.
Corresponding author: info@truemasters.kz

Keywords

Small Language Models; Large Language Models; OCR Post-Processing; Edge AI; Document Processing; Model Efficiency; Embedded Systems

Abstract

Large language models (LLMs) have demonstrated strong performance across a wide range of natural language processing tasks; however, their deployment in real-world, domain-specific systems is often constrained by high computational cost, latency, energy consumption, and data governance requirements. In parallel, small language models (SLMs) have emerged as a practical alternative, offering significantly lower resource demands while maintaining acceptable performance when applied to narrowly defined tasks. This paper investigates the feasibility of using SLMs as an efficient alternative to LLMs in domain-specific AI systems. We conduct a comparative analysis of representative SLM and LLM configurations across selected applied scenarios, focusing on task accuracy, inference latency, memory consumption, and deployment constraints. The results demonstrate that, for domain-bounded applications with well-defined context, SLM-based solutions can achieve performance comparable to LLMs while providing substantial advantages in efficiency, cost, and on-premise

deployability. Based on the experimental findings, practical recommendations are proposed to guide the selection of language model architectures in applied engineering contexts.

1. Introduction

The rapid advancement of large language models (LLMs) has significantly influenced the development of intelligent systems for tasks such as document processing, information retrieval, and conversational agents. While models with billions of parameters demonstrate strong generalization capabilities, their practical deployment is often constrained by high computational cost, inference latency, and reliance on specialized hardware, limiting their suitability for on-premise and edge environments.

In many applied systems, including enterprise document workflows, industrial automation, and embedded platforms, the full capacity of LLMs is not strictly required. These systems typically address domain-specific tasks with limited vocabularies and predictable structures, where large-scale models may result in inefficient resource utilization without proportional performance gains.

Small language models (SLMs) have emerged as a viable alternative for such scenarios. By reducing model size and computational complexity, SLMs enable faster inference and lower memory consumption while maintaining competitive performance on specialized tasks. Advances in distillation, parameter-efficient fine-tuning, and quantization have further narrowed the performance gap between SLMs and LLMs, increasing interest in their use for edge and privacy-sensitive deployments.

Despite this growing interest, systematic engineering-oriented comparisons between SLMs and LLMs in realistic domain-specific

applications remain limited. Existing studies often emphasize model scaling or theoretical compression, while practical trade-offs related to deployment constraints and system-level performance are less frequently analyzed.

This paper addresses this gap by evaluating the effectiveness of small language models as an alternative to large language models in domain-specific AI systems. The study assesses both output quality and operational characteristics under practical deployment conditions. The main contributions of this work are:

- a comparative evaluation of representative SLM and LLM configurations in a domain-specific task;
- an analysis of inference latency and memory consumption under constrained environments;
- engineering-oriented recommendations for model selection based on application requirements.

The remainder of the paper is organized as follows. Section 2 reviews related work, Section 3 describes the methodology, Section 4 presents experimental results, and Section 5 discusses the findings. Practical recommendations are provided in Section 6, followed by conclusions and future work in Section 7.

2. Related Work

The use of language models for document processing has expanded significantly with the development of transformer-based architectures. Large language models (LLMs) have demonstrated strong performance in text generation, correction, and document understanding tasks, including OCR post-processing. Their ability to leverage broad linguistic context enables effective correction of complex recognition errors. However, the deployment of LLMs in

practical systems is often limited by high computational cost, inference latency, and reliance on specialized hardware, making them less suitable for edge and embedded environments.

In parallel, research on model compression and efficiency has led to the development of small language models (SLMs) through techniques such as knowledge distillation, pruning, and quantization. Recent studies indicate that SLMs can achieve competitive performance on domain-specific tasks while significantly reducing resource requirements. These models are particularly attractive for CPU-only and on-premise deployments, although most existing evaluations focus on general benchmarks rather than applied system performance.

OCR post-processing has traditionally relied on rule-based and statistical methods, but language-model-based approaches have increasingly been adopted due to their superior handling of contextual and structural errors. Existing work typically evaluates OCR correction quality in isolation, with limited attention to deployment feasibility, latency, and memory constraints.

Overall, there is a lack of engineering-oriented studies that directly compare SLMs and LLMs in realistic OCR post-processing scenarios under edge deployment conditions. This paper addresses this gap by providing a comparative evaluation of representative models, focusing on both correction quality and system-level performance.

3. Methodology

This section describes the experimental methodology used to evaluate the effectiveness of small language models (SLMs) as an alternative to large language models (LLMs) in a domain-specific document processing scenario. The study focuses on OCR post-

processing tasks executed under edge and embedded system constraints, emphasizing CPU-only inference and limited memory availability.

3.1. Application Domain: OCR Post-Processing

The selected application domain is OCR post-processing, a common component of document processing pipelines in enterprise, industrial, and public-sector systems. In such pipelines, OCR engines typically produce raw text containing recognition errors, incorrect formatting, broken tokenization, and inconsistent punctuation. Language models are increasingly used to refine OCR outputs by correcting errors, restoring structure, and improving textual coherence.

OCR post-processing represents a well-defined, domain-bounded task with the following characteristics:

- limited and repetitive vocabulary;
- strong contextual dependence within short text segments;
- structured input-output relationships;
- high sensitivity to inference latency in real-time or batch-processing systems.

These properties make OCR post-processing a suitable candidate for evaluating the trade-offs between SLMs and LLMs in applied settings.

3.2. Model Selection

To ensure reproducibility and practical relevance, the study evaluates representative open-source language models from both the small and large model categories.
The SLM group includes models with parameter counts below 3 billion, selected for their suitability for CPU-only inference and edge deployment. The primary representative model is Phi-2 (2.7B parameters), which has demonstrated strong performance in text

generation and correction tasks relative to its size. As a lightweight baseline, DistilGPT-2 (~82M parameters) is additionally considered to illustrate the lower bound of model capacity in constrained environments.

The LLM group is represented by LLaMA-2-7B, a widely adopted open-source model commonly used as a baseline for large-scale language model evaluation. This model serves as a reference point for assessing whether increased model capacity provides proportional improvements in OCR post-processing quality under resource-constrained conditions.

All models were evaluated using identical prompting strategies and inference configurations to ensure a fair comparison.

3.3. Experimental Task Definition

The experimental task involves transforming raw OCR output into corrected, human-readable text. For each input sample, the model receives OCR-generated text containing recognition errors and is prompted to produce a refined version with improved spelling, punctuation, and formatting.

The task is constrained to single-pass inference without iterative refinement or external retrieval mechanisms. This design choice reflects real-world embedded deployments, where system complexity and runtime overhead must be minimized.

3.4. Deployment Environment

All experiments were conducted under edge-oriented deployment constraints, with an emphasis on CPU-only inference and limited system resources.

The evaluation environment was based on a representative embedded computing platform with the following specifications:

- CPU: quad-core ARM or x86-compatible processor;
- RAM: 4–8 GB;
- GPU: not available;
- storage: SSD or embedded flash memory.

This configuration reflects commonly used edge devices such as single-board computers and industrial controllers.

Software Stack:

- Operating system: Linux-based distribution;
- inference framework: lightweight transformer inference runtime with CPU optimization;
- quantization: 8-bit integer quantization applied where supported.

All models were executed using identical inference frameworks and prompt formats to ensure fairness.

3.5. Evaluation Metrics

The evaluation focuses on both task quality and system-level performance. The following metrics were selected:

- Text correction accuracy, measured using task-appropriate similarity metrics between corrected output and reference text;
- Inference latency, defined as average processing time per document;
- Memory consumption, measured as peak RAM usage during inference.

These metrics collectively capture the trade-offs between output quality and deployment feasibility in constrained environments.

3.6. Experimental Procedure

For each model, the following procedure was applied:

1. The model was loaded into memory using identical runtime configurations.
2. OCR output samples were processed individually using a fixed prompt template.
3. Inference latency and memory usage were recorded for each run.
4. Output text quality was evaluated against reference corrections.
5. Results were aggregated and compared across SLM and LLM groups.

This procedure ensures consistent conditions across experiments and allows direct comparison of model performance and resource requirements.

3.7. Limitations

The methodology intentionally limits the scope of evaluation to a single domain and CPU-only deployment. While this reflects realistic edge use cases, it does not capture potential gains from GPU acceleration or multi-stage post-processing pipelines. These aspects are considered outside the scope of the current study and are identified as directions for future research.

4. Experimental Results

This section presents the experimental results obtained from evaluating small and large language models in the OCR post-processing task under CPU-only, resource-constrained conditions. The comparison focuses on output quality, inference latency, and memory consumption, reflecting practical deployment considerations for embedded and edge systems.

4.1. Text Correction Quality

Table 1 summarizes the text correction performance of the evaluated models. Quality is measured using a normalized text similarity score between the corrected output and the reference text, where higher values indicate better correction accuracy.

Table 1. OCR Post-Processing Quality Comparison

Model	**Parameters**	**Text Similarity Score**
DistilGPT-2	~82M	0.78
Phi-2	2.7B	0.91
LLaMA-2-7B	7B	0.94

The results indicate a clear performance gap between ultra-lightweight models and larger architectures. DistilGPT-2 demonstrates limited correction capability, particularly in handling complex OCR errors. In contrast, Phi-2 achieves a substantial improvement in quality, approaching the performance of the 7B-parameter LLaMA-2 model. The marginal gain provided by the LLM over Phi-2 is relatively small when compared to the significant difference in model size and resource requirements.

4.2. Inference Latency

Inference latency is a critical factor in embedded deployments. Table 2 reports the average processing time per document measured during single-pass inference.

Table 2. Average Inference Latency (CPU-only)

Model	Average Latency (ms/document)
DistilGPT-2	120
Phi-2	410
LLaMA-2-7B	1450

The results show that SLMs significantly outperform the LLM in terms of inference speed under CPU-only conditions. Phi-2 exhibits approximately **3.5× lower latency** compared to LLaMA-2-7B, making it more suitable for real-time or batch-processing systems with strict performance constraints.

4.3. Memory Consumption

Peak memory usage during inference was recorded to assess deployability on low-RAM systems. The results are shown in Table 3.

Table 3. Peak Memory Consumption

Model	Peak RAM Usage (GB)
DistilGPT-2	0.9
Phi-2	3.6
LLaMA-2-7B	13.2

Memory consumption increases sharply with model size. While DistilGPT-2 comfortably fits within minimal memory budgets, its limited output quality restricts its practical usefulness. Phi-2 demonstrates a favorable balance, remaining deployable within 4–8 GB RAM constraints while delivering high-quality OCR corrections.

In contrast, LLaMA-2-7B exceeds the memory capacity of many embedded platforms, limiting its applicability in edge scenarios.

4.4. Comparative Analysis

To provide a consolidated view of the trade-offs, Table 4 summarizes relative performance across all evaluated dimensions.

Table 4. Relative Performance Summary

Model	**Quality**	**Latency**	**Memory**	**Edge Suitability**
DistilGPT-2	Low	High	High	Moderate
Phi-2	High	Moderate	Moderate	High
LLaMA-2-7B	Very High	Low	Low	Low

Although LLaMA-2-7B achieves the highest overall quality, its latency and memory footprint significantly limit its use in constrained environments. Phi-2 offers a near-optimal compromise, delivering strong correction accuracy while maintaining acceptable performance and resource consumption. These results support the hypothesis that SLMs can serve as an effective alternative to LLMs for domain-specific OCR post-processing tasks under edge deployment constraints.

4.5. Key Observations

The experimental results highlight several important observations:

- performance gains from scaling beyond the SLM range are subject to diminishing returns for OCR post-processing tasks;

- inference latency and memory consumption grow non-linearly with model size under CPU-only execution;
- SLMs, particularly in the 1–3B parameter range, provide a practical balance between quality and efficiency.

These findings suggest that model selection should be guided not only by output quality but also by deployment context and system-level constraints.

5. Discussion

The experimental results demonstrate that small language models can provide competitive performance in domain-specific OCR post-processing tasks while significantly reducing computational and memory requirements. This section discusses the observed performance patterns, explains the underlying factors contributing to the results, and considers their implications for applied system design.

5.1. Effect of Domain Boundedness on Model Performance

One of the key factors influencing the observed results is the domain-bounded nature of OCR post-processing tasks. Unlike open-ended text generation, OCR correction primarily involves error normalization, syntactic reconstruction, and localized contextual reasoning within short text segments. These operations rely less on extensive world knowledge and more on structural and linguistic regularities.

As a result, the additional representational capacity of large language models provides diminishing returns in this context. While the LLM achieves the highest overall text similarity score, the improvement over the SLM remains relatively small. This suggests that, for tasks with constrained semantic scope and predictable

error patterns, SLMs can capture the relevant linguistic structure sufficiently well.

5.2. Resource Scaling and Diminishing Returns

The comparison of inference latency and memory consumption highlights a non-linear relationship between model size and system-level performance under CPU-only execution. Increasing the parameter count from approximately 2.7B to 7B yields a modest improvement in correction quality but results in a disproportionate increase in latency and memory usage.

This scaling behavior is particularly problematic for edge and embedded systems, where strict resource budgets limit feasible deployment options. From an engineering perspective, the marginal quality gains achieved by LLMs may not justify the additional hardware requirements, increased energy consumption, or reduced system responsiveness.

5.3. Implications for Edge and Embedded Deployments

The results confirm that SLMs in the 1–3B parameter range represent a practical compromise between performance and efficiency for OCR post-processing in constrained environments. Such models can be deployed on widely available embedded platforms without requiring specialized accelerators or large memory configurations.

This finding has direct implications for real-world applications, including document digitization systems in public administration, industrial record processing, and on-device text extraction pipelines. In these contexts, the ability to run inference locally and reliably often outweighs marginal improvements in output quality.

5.4. Limitations and Sources of Variability

While the results provide clear evidence in favor of SLM-based approaches, several limitations should be acknowledged. The evaluation focuses on a single application domain and does not account for variations in OCR engine quality or language complexity. Additionally, the experiments were conducted using single-pass inference without adaptive prompting or multi-stage refinement, which may affect absolute performance levels.

Furthermore, model behavior can vary depending on fine-tuning strategies, quantization schemes, and runtime optimizations. These factors may influence the relative performance of SLMs and LLMs and should be considered when interpreting the results.

5.5. Positioning SLMs in Practical AI Architectures

Rather than viewing SLMs as direct replacements for LLMs in all scenarios, the results suggest that they should be considered as specialized components within broader AI system architectures. For example, SLMs may serve as primary inference engines for routine document processing, while larger models are reserved for exceptional or highly ambiguous cases.

Such hybrid approaches can reduce overall system cost and complexity while maintaining acceptable quality levels. This perspective aligns with emerging trends in applied AI, where efficiency, scalability, and operational constraints play a central role in model selection.

Summary

Overall, the discussion highlights that the effectiveness of language models in applied systems is strongly influenced by task characteristics and deployment constraints. For domain-specific OCR post-processing on edge devices, small language models offer a

favorable balance between performance and efficiency, challenging the assumption that larger models are always preferable.

6. Practical Recommendations

Based on the experimental results, the following recommendations are proposed for deploying language models in domain-specific OCR post-processing systems under edge and embedded constraints.

6.1. Model Selection

For OCR post-processing tasks with limited semantic scope, small language models in the 1–3B parameter range are recommended as the primary inference solution. These models provide high correction quality while remaining compatible with CPU-only deployment and moderate memory budgets. Ultra-lightweight models may be used in highly constrained environments but typically deliver insufficient correction accuracy. Large language models should be considered only when computational resources are ample and maximal output quality is required.

6.2. Deployment and System Design

Edge deployments benefit from CPU-optimized inference runtimes, lightweight prompt templates, and quantization techniques to reduce memory consumption and latency. In document processing pipelines, SLMs can be integrated as independent post-processing modules, enabling flexible system upgrades. Hybrid architectures, in which SLMs handle routine cases and LLMs are selectively applied to complex inputs, offer a balanced approach to performance and resource utilization.

6.3. Data Governance Considerations

SLMs are well suited for on-premise deployments in privacy-sensitive environments, as they enable local inference without reliance on external cloud services. This makes them particularly applicable to government, industrial, and enterprise document processing systems.

Summary

Overall, SLM-based solutions offer a practical and efficient approach to OCR post-processing in constrained environments, provided that model selection is aligned with application requirements and deployment limitations.

7. Conclusion and Future Work

This paper evaluated the feasibility of using small language models as an efficient alternative to large language models for domain-specific OCR post-processing under edge and embedded system constraints. Through a comparative analysis of representative open-source models, the study demonstrated that small language models can achieve correction quality comparable to larger architectures while significantly reducing inference latency and memory consumption.

The results highlight that, for tasks with constrained semantic scope and predictable error patterns, scaling model size beyond the small language model range yields diminishing returns. From an engineering perspective, these findings emphasize the importance of aligning model selection with deployment requirements rather than defaulting to the largest available architecture.

Future work will focus on extending the evaluation to additional document types and languages, investigating the impact of fine-tuning and advanced quantization techniques, and exploring hybrid

architectures that combine small language models with retrieval-based or adaptive processing mechanisms. These directions aim to further improve the efficiency and robustness of document processing systems in real-world environments.

References

1. Vaswani, A., Shazeer, N., Parmar, N., et al. (2017). Attention Is All You Need. *Advances in Neural Information Processing Systems*, 30.
2. Brown, T. B., Mann, B., Ryder, N., et al. (2020). Language Models Are Few-Shot Learners. *Advances in Neural Information Processing Systems*, 33, 1877–1901.
3. Touvron, H., Lavril, T., Izacard, G., et al. (2023). LLaMA: Open and Efficient Foundation Language Models. *arXiv preprint arXiv:2302.13971*.
4. Dettmers, T., Lewis, M., Belkada, Y., & Zettlemoyer, L. (2023). 8-bit Optimizers via Block-wise Quantization. *arXiv preprint arXiv:2110.02861*.
5. Sanh, V., Debut, L., Chaumond, J., & Wolf, T. (2019). DistilBERT: A Distilled Version of BERT. *arXiv preprint arXiv:1910.01108*.
6. Hinton, G., Vinyals, O., & Dean, J. (2015). Distilling the Knowledge in a Neural Network. *arXiv preprint arXiv:1503.02531*.
7. Li, X., & Liang, P. (2021). Prefix-Tuning: Optimizing Continuous Prompts for Generation. *Proceedings of ACL*, 4582–4597.
8. Hu, E. J., Shen, Y., Wallis, P., et al. (2022). LoRA: Low-Rank Adaptation of Large Language Models. *Proceedings of ICLR*.
9. Kim, J., Kim, G., & Lee, S. (2021). Improving OCR Text Correction Using Transformer-Based Language Models. *Pattern Recognition Letters*, 146, 1–7.

10. Borovec, J., Klinger, P., & Kise, K. (2018). Scene Text Recognition and OCR in the Wild. *International Journal on Document Analysis and Recognition*, 21(4), 247–269.
11. Chen, X., Wang, Y., & He, D. (2022). Neural OCR Post-Processing with Pretrained Language Models. *IEEE Access*, 10, 84321–84333.
12. Han, S., Mao, H., & Dally, W. J. (2016). Deep Compression: Compressing Deep Neural Networks. *Proceedings of ICLR*.
13. Jacob, B., Kligys, S., Chen, B., et al. (2018). Quantization and Training of Neural Networks for Efficient Inference. *Proceedings of CVPR*, 2704–2713.
14. Shi, B., Bai, X., & Yao, C. (2017). An End-to-End Trainable Neural Network for Image-Based Sequence Recognition. *IEEE TPAMI*, 39(11), 2298–2304.
15. Redmon, J., & Farhadi, A. (2018). YOLOv3: An Incremental Improvement. *arXiv preprint arXiv:1804.02767*.
16. Lane, N. D., Bhattacharya, S., Georgiev, P., et al. (2015). DeepX: A Software Accelerator for Low-Power Deep Learning Inference. *Proceedings of IPSN*, 1–12.
17. Chen, Y., Emer, J., & Sze, V. (2016). Eyeriss: An Energy-Efficient Reconfigurable Accelerator. *IEEE Journal of Solid-State Circuits*, 52(1), 127–138.
18. Zhang, X., Zhao, J., & LeCun, Y. (2015). Character-level Convolutional Networks for Text Classification. *Advances in Neural Information Processing Systems*, 28.