Small Language Models on the Edge for Real-World Agentic Systems in Industry

Edward B. Duffy[1], David Fernandez[2], Alta de Waal[3], and Mert D. Pesé[2]

[1] BMW Group, Information Technology Research Center, SC, USA
[2] Clemson University, SC, USA
[3] BMW IT Hub, South Africa

Abstract. Large Language Models face significant deployment challenges in enterprise environments, including high computational costs, data privacy concerns, and network dependencies. This paper presents a framework for deploying Small Language Models (SLMs) with fewer than 7 billion parameters on edge devices, using agentic architectures to overcome capacity limitations. We introduce three key contributions: (1) a multi-agent benchmarking framework employing role-based evaluation to reduce bias, (2) a three-phase task planning pipeline that decomposes planning into subtask identification, dependency reasoning, and schema-constrained generation, and (3) real-world implementations achieving 3-4x latency improvements over cloud services. Our evaluation demonstrates that models like Phi-4 achieve CEFR C1-level translation quality and 0.883 G-Eval summarization scores on commodity hardware. Through WebLLM browser-based inference and local hosting, we show that SLMs effectively serve enterprise needs in privacy-sensitive, bandwidth-constrained, or air-gapped environments, representing a viable alternative prioritizing data sovereignty and cost efficiency.

Keywords: Small Language Models, Edge Computing, Agentic AI, Multi-Agent Systems,LLM Evaluation, On-Device Inference

1 Introduction

Increasing interest in agentic AI [16], systems that autonomously plan, reason and act, has redefined how Artificial Intelligence (AI) can be applied in enterprise environments. Rather than relying on monolithic end-to-end models, agentic AI frameworks decompose problems into smaller tasks, coordinate tools, and iterate to reach goals. These systems mirror human problem solving by using components such as task planners, memory retrieval modules, and tool users [33].

The deployment of such intelligent systems has been accelerated by Large Language Models (LLMs) such as GPT-4 [32], Claude [6] and Gemini [39], which offer impressive generalization and reasoning capabilities across diverse domains [34]. However, these capabilities come at a cost. LLMs require substantial computational infrastructure for both training and inference, and API-based access can result in expenses that exceed hundreds of thousands of dollars annually [14,31]. These financial barriers limit accessibility for many organizations.

Although agentic AI is typically powered by large foundation models, recent developments show that it can also be implemented using smaller, specialized components [49] making it an attractive architecture for enterprise environments.

Privacy and data governance concerns pose additional adoption challenges for enterprise AI systems. LLMs typically rely on cloud-based APIs, requiring sensitive information to be transmitted to third-party providers. This introduces potential compliance risks, particularly in regulated industries such as finance, healthcare, and manufacturing. Furthermore, jurisdictions such as the European Union (GDPR) [1] and South Africa (POPIA) [38] enforce strict data residency and sovereignty requirements, complicating cloud-based LLM usage.

Edge computing presents an alternative deployment approach that can address many of these challenges[50]. By processing data closer to its source, edge devices can reduce latency, minimize bandwidth requirements, and keep sensitive information within organizational boundaries [28]. However, traditional edge hardware has limited computational resources compared to cloud infrastructure, making it unsuitable for running large language models efficiently.

Self-hosting large models such as Llama 70B [2] or Falcon 180B [3] is not a simple alternative. These models require high-end GPU clusters and specialized infrastructure [4], making them inaccessible to smaller organizations or those operating in environments with limited connectivity and bandwidth.

In response to these challenges, Small Language Models (SLMs) have emerged as a promising alternative. These models typically contain fewer than 7 billion parameters [19] and are small enough to run on commodity hardware such as standard workstations, industrial PCs or even edge devices. This allows organizations to enable edge computing and reduces reliance on external infrastructure. SLMs open new opportunities for localized, real-time AI without compromising privacy or budget. SLMs offer several advantages for enterprise use:

- **Data Privacy:** By running locally, SLMs ensure that sensitive information remains within the organization's network.
- **Cost Efficiency:** These models can run on existing hardware, eliminating the need for high-end GPUs and costly API usage.
- **Reduced Network Dependency:** SLMs support real-time inference in bandwidth constrained settings such as manufacturing floors.
- **Edge Deployment Capability:** SLMs can be deployed directly on edge devices, enabling distributed AI processing closer to data sources while maintaining low latency and reducing dependence on centralized infrastructure.

However, the reduced size of SLMs leads to a decrease in general reasoning, memory capacity, and instruction-following capabilities compared to state-of-the-art LLMs [13]. This limitation restricts their effectiveness for complex tasks and presents a challenge for deploying SLMs in production environments.

To address this, we introduce an approach that combines agentic principles with specialized SLMs. Instead of relying on a single large model, we implement agent frameworks that decompose tasks into smaller subtasks, each handled by dedicated SLMs. These modular agents may include planners, retrievers, or

tools users. This structure allows SLMs to operate efficiently in low-resource environmments while maintaining the capability to execute complex goals.

Our methodology includes a novel benchmarking pipeline that simulates real-world workflows using agent roles such as Proctor, Student, and Grader. We evaluate SLM performance on translation and summarization tasks and analyze how specialized agent roles can enhance effectiveness despite limited model size.

Our experiments show that agentic workflows allow SLMs to achieve competitive performance in multilingual translation and summarization, with gains in privacy, and cost-efficiency. This paper makes the following contributions:

- **Novel Multi-Agent Benchmarking Framework**: We introduce an evaluation methodology that uses multiple agents in different roles (Proctor, Student, Grader) to assess SLM capabilities. This framework reduces evaluation bias through multi-perspective assessment and provides more robust performance metrics than traditional single-model evaluation approaches.
- **Three-Phase Task Planning Pipeline**: We develop a decomposed approach to task planning specifically designed for the limitations of SLMs. By breaking the planning process into subtask identification, dependency reasoning, and schema-constrained generation phases, we show significant improvements in SLM planning reliability compared to single-pass approaches.
- **Real-World Use Case Implementation**: We demonstrate the practical applicability of our approach through a multilingual translation system comparing SLM against cloud-based services, showing 3-4x latency improvements while maintaining data sovereignty and reduced network usage.

2 Related Work

2.1 Small Language Models

Recent research has demonstrated that models with fewer than 7 billion parameters can achieve competitive performance through improved training strategies [19] and high-quality data [40], while compression techniques like quantization [20] and low-rank adaptation [9] enable resource-constrained deployment. However, smaller models exhibit reduced reasoning depth and instruction-following reliability [13], particularly for complex multi-step tasks. Rather than matching LLM capabilities through better training alone, our work addresses these limitations through architectural decomposition. Our three-phase task planning pipeline (subsection 3.1) targets planning weaknesses by breaking monolithic reasoning into manageable stages: subtask identification, dependency reasoning, and schema-constrained generation.

2.2 Agentic AI Frameworks

Agentic AI frameworks like AutoGPT [45] and LangChain demonstrate multi-step reasoning and tool usage [41], with strategies such as ReAct [47] interleaving reasoning and acting, and Tree of Thoughts [46] enabling deliberate

problem-solving. Multi-agent collaboration approaches [26] leverage debate between agents for complex tasks. However, these frameworks target large models and assume sufficient individual agent reasoning capacity. We extend agentic principles to edge computing by designing orchestration strategies for resource-constrained SLMs. Our agent framework (subsection 3.1) distributes cognitive load across specialized components rather than relying on individual agent sophistication, enabling complex execution on edge-deployed smaller models.

2.3 LLM-as-a-Judge and Evaluation

The LLM-as-a-Judge paradigm [48] enables scalable model evaluation using LLMs [27], refined through systems like PandaLM [43], JudgeLM [51], and Auto-J [25]. However, single-judge systems suffer from position bias [42], length bias [36], and evaluation inconsistency [21]. Our multi-agent benchmarking framework (section 5) addresses these biases by using multiple grader agents with different LLM backends, forcing diverse reasoning paths and preventing consensus artifacts from identical model evaluations.

2.4 Edge Computing for AI

Edge AI deployment addresses latency, bandwidth, and privacy concerns by processing data near its source [50]. Existing work has focused on optimization techniques like pruning [17], quantization [20], and knowledge distillation [18] primarily for traditional deep learning models. We demonstrate practical edge deployment of language models through WebLLM [35] for browser-based inference and local hosting on commodity hardware.

2.5 Multi-Modal and Domain-Specific Applications

Recent advances have explored specialized applications of language models in various domains including legal text analysis [15], medical Q&A [37], and financial analysis [44]. Recent work in autonomous vehicles has explored Vision Language Models for crash analysis [12], adversarial traffic sign attacks [5], and LLM-based anomaly detection [11]. Our work contributes a general framework adaptable across domains while maintaining edge deployment benefits.

3 Background

3.1 Agents

An AI Agent is an autonomous software entity that perceives its environment, reasons, and acts to achieve specific goals through adaptive behavior rather than predefined scripts. As shown in Figure 1, an agent consists of several interconnected components that work together to enable intelligent behavior: a prompt

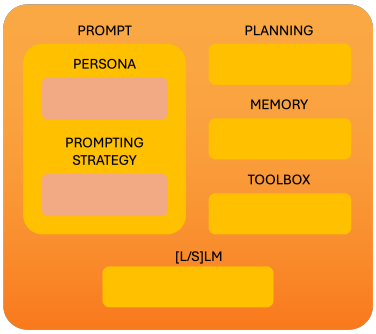

Fig. 1: **Anatomy of Agent:** Prompt layer defines the agent's behavior through persona definition and prompting strategy. Planning component enables task decomposition and execution strategy formulation.

defining persona and strategy, autonomy and planning capabilities for multi-step decision-making, memory for context retention and learning, and tools like semantic search and APIs for enhanced functionality beyond text generation.

An Agent Framework, Figure 2, is a structured system designed to manage and coordinate AI agents to solve complex tasks. At its core is a central Coordinator, which oversees the Task Queue, a dynamic list of tasks that guides the agent's workflow. The framework operates in three main steps: **Planning and task decomposition** (1) , where the Coordinator breaks down the user's original query into smaller, manageable subtasks that populate the task queue. The **Executing Agents** phase (2) involves AI agents processing these tasks, leveraging tools such as APIs or external systems as needed, to generate results. Finally, the **Verification** step (3) ensures that outputs align with the intent of the original user query, validating the accuracy and completeness of the solution. This modular approach allows the framework to maintain organization, adapt to complex problems, and ensure high-quality outcomes.

3.2 SLM Task Planning

The premise of task planning is to decompose a user's query into a set of smaller, manageable subtasks that can be distributed across specialized agents. These subtasks are typically interdependent and should not be treated as a flat, sequential list. Instead, they form a **dependency graph** where some tasks may be executed in parallel, while others depend on the completion prerequisite subtasks. Effective task planning requires both structured reasoning and the ability to generate outputs in a format that downstream agents or orchestrators can interpret. For this purpose, we enforce a parsable schema-defined output format, which ensures consistency and interoperability across agent components.

LLMs are capable of performing this end-to-end task with a well-tuned system prompt. A single agent can generate task breakdowns, assign dependencies, and output a schema-compliant task list. However, SLMs struggle with this approach due to limitations in context window size, reasoning depth, and memory.

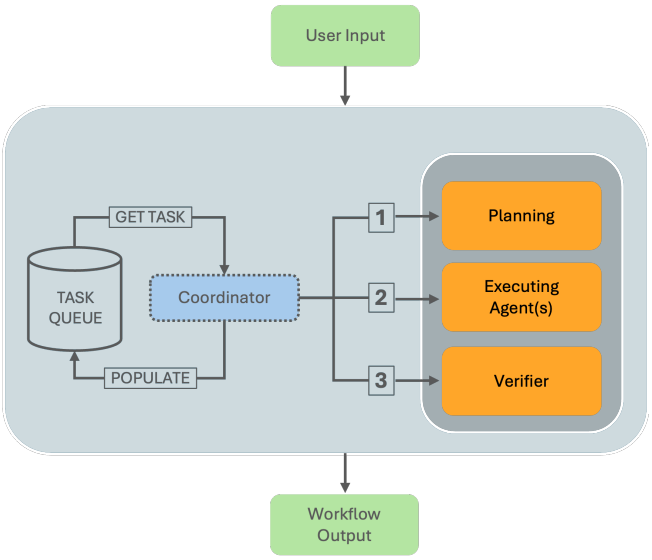

Fig. 2: **Agent Framework:** User input flows into the framework where the Coordinator manages the Task Queue and orchestrates the three-phase process.

To address these limitations, we designed a three-phase task planning pipeline tailored to SLMs capabilities. Instead of requiring the model to reason, reflect, and output data in a single call, we decompose the process into distinct stages:

1. **Subtask Identification:** The SLM is prompted to break the original query into a list of subquestions or subgoals. This stage focuses entirely on understanding and dissecting the user's intent; no output schema is enforced.
2. **Dependency Reasoning:** For each subtask, the SLM is asked to specify any other subtasks that must be completed first. This builds a dependency tree and makes the model to reason about prerequisite relationships.
3. **Schema-Constrained Task List Generation:** Finally, the SLM is prompted to generate a fully structured task list object that conforms to a predefined JSON schema. This step enforces strict formatting, improving compatibility with task orchestration engines or downstream agents.

This modular design enables SLMs to focus on a single cognitive step at a time: identifying the task, reasoning through dependencies, and formatting the final output. We found that this sequential prompting strategy substantially improves the reliability of the outputs compared to single-pass generation.

4 Methodology

Our research employs a three-phase methodology to evaluate SLMs:

Phase 1: Multi-Agent Benchmarking Framework (section 5). We developed an evaluation architecture using role-based agents, a Proctor coordi-

nates assessment, Student agents (SLMs under test) perform tasks, and multiple Grader agents with different LLM backends conduct collaborative evaluation.

Phase 2: Systematic Performance Evaluation We assessed multiple SLMs (3-7B parameters) across translation tasks (subsection 5.1) using CEFR proficiency standards and summarization tasks (subsection 5.2) using G-Eval metrics [32]. Models were evaluated under different deployment conditions (WebLLM and local hosting using V100) to measure both task quality and operational feasibility including latency, memory requirements, and offline capability.

Phase 3: Production Deployment Validation (section 6) We implemented systems to validate benchmarking results, including an Afrikaans translation service comparing edge-deployed MADLAD-400 (3B) against the cloud-based Vulavula API, measuring latency and data sovereignty maintenance.

While our evaluation primarily demonstrates translation and summarization tasks, these represent generic capabilities applicable across diverse industries (manufacturing, healthcare, finance, legal services) without requiring exposure of sensitive domain-specific data. We have also developed production systems for document comparison (legal compliance verification) and multi-step task planning, though space constraints precluded their inclusion.

5 Benchmarking

To evaluate the capabilities of various SLMs across different tasks, we designed a multi-agent benchmarking framework inspired by real-world evaluation scenarios. This framework simulates a student examination process using multiple coordinated agents, each with a clearly defined role. As shown in Figure 3, the framework operates the following components and process flow:

The **Proctor** agent, backed by an LLM, serves as the central coordinator managing the entire evaluation process. The workflow begins when the Proctor receives an **Exam** from the available tests ①. The proctor then administers this exam to the **Student** agent (SLM under test), by sending the exam prompt ②. After receiving the student's response ③, the Proctor forwards both the original exam and the student's translation to the first **Grader agent** ④. The first grader agent, backed by GPT, then shares the information with the second **Grader agent**, backed by Claude, to initiate a collaborative evaluation process.

The two Grader agents engage in structured discussion to evaluate student performance ⑤, ensuring objective assessment through distributed reasoning. Upon reaching consensus on grade and review, they return their evaluation to the Proctor ⑥, who records results in the Benchmark Results ⑦.

This multi-grader approach reduces bias by instantiating the Grader agents using different LLM backends, avoiding immediate agreement due to identical reasoning paths and enforcing richer evaluation of the SLM's capabilities.

5.1 Translation Benchmarking

The translation benchmark begins by fetching the most recent English-language articles from Wikipedia's featured article of the day section. These articles are se-

lected for their clarity, well-written structure, and topical diversity, which makes them ideal candidates for assessing multilingual translation capabilities.

The student (SLM) is instructed to produce a full translation of the article in a target language without additional commentary. The grader agents then receive both the source and translated versions and are prompted to engage in a dialog to determine the translation's quality. The grading system follows the Common European Framework of Reference for Languages (CEFR) [10], which defines seven levels of language proficiency:

- F: Student has no understanding of the language.
- A1: Beginner level with a working vocabulary of about 700 words.
- A2: Pre-intermediate level; users understand basic expressions.
- B1: Intermediate level; users can communicate reasonably well.
- B2: Advanced level; users can communicate easily and spontaneously.
- C1: Proficient level; users can perform complex tasks.
- C2: A mastery or proficiency level; users can understand almost everything.

Table 1: Translation Benchmark Results: Average CEFR scores across languages

Deployment	Model	German	Spanish	Portuguese	Magyar	Japanese	French
WebLLM	Llama-3.2	2.900	2.940*	3.300**	1.580*	1.200	3.120
	Phi-3-mini-128k	3.040	2.340	2.860	0.340	1.200	2.098
	Qwen2.5 (f16)	3.100*	2.660	2.960	0.540	2.020*	3.740*
NVIDIA V100	Falcon	2.120	3.600	3.200	0.280	1.000	3.480
	Phi4	**4.040****	**4.520****	3.286*	**3.180****	**4.000****	**4.580****

* Best model within deployment category
** Best overall model across all deployments

After discussion, the *Grader Agents* reach consensus on the CEFR level and provide a justification. The *Proctor* then records the final grade and review.

Each model was tested 10 times across 5 languages, for a total of 50 test instances per model. Each CEFR grade was assigned a numerical score (0=F, 1=A1, 2A2, 3=B1, 4=B2, 5=C1, 6=C2), and average scores were calculated.

We evaluated multiple hosting strategies for small language models, using Ollama [8] and MLC-LLM [30] for local deployment with OpenAI-compatible APIs, and WebLLM [35] for browser-based inference through WebGPU and WebAssembly. We configured private hosting from local web servers to enable the deployment of proprietary models, reduce transfer latency, and maintain enterprise control, with model files cached after initial download. Additional deployment options include dedicated accelerated edge devices such as the NVIDIA Jetson family. Performance results are shown in Table 1.

5.2 Summarization Benchmarking

For summarization, we adapted the G-Eval [27] technique from Microsoft [29] which scores generated summaries on four key dimensions: Coherence: Logical

flow and structure Consistency: Faithfulness to the source text Fluency: Grammar and readability Relevance: Inclusion of essential information

Rather than relying on a single LLM for scoring, we extend the technique using our multi-agent workflow. For each metric, a pair of LLM-backed Grader agents is assigned to evaluate and discuss the summary, ultimately agreeing on a numeric score (0.0 to 1.0). For each model, the benchmark was run 3 times. The average was tracked for each category and an overall grade was calculated. Some models were run using pre-complied models from WebLLM, while others were self-hosted on an NVIDIA V100. The results are shown in Table 2.

Table 2: Summarization Benchmark Results: G-Eval scores across evaluation metrics

Deployment	Model	Coherence	Consistency	Fluency	Relevance	Overall
WebLLM	Llama-3.2	0.583	0.650	0.972	0.642	0.712
	Phi-3-mini-128k	0.469	0.433	0.738	0.444	0.521
	Qwen2.5 (f16)	0.692*	0.717*	1.000**	0.683*	0.773*
NVIDIA V100	Falcon	**0.817****	0.822	1.000**	**0.856****	0.867
	Phi4	**0.817****	**0.867****	1.000**	0.850	**0.883****

* Best model within deployment category
** Best overall model across all deployments

6 Use Case Study: Translation (Afrikaans)

We implemented an Afrikaans translation system to demonstrate SLMs in low-resource scenarios, addressing enterprise needs for data sovereignty and low-latency translation in multilingual environments.

Lelapa AI [23] is a South African AI company that develops state-of-the-art natural language processing solutions specifically for African languages. Founded with the mission to make AI accessible to African language speakers, Lelapa has created Vulavula, a comprehensive multilingual NLP platform that provides translation, named entity recognition, and other language services for several African languages including Afrikaans, isiZulu, Sesotho, and isiXhosa [24].

The Vulavula platform offers cloud-based API access to large, specialized models trained on African language corpora. While these models provide high-quality translations, they require internet connectivity, incur API costs, and require sharing information with external services, making them less suitable for edge deployment scenarios or environments with limited network access.

As an alternative to cloud-based solutions, we evaluated MADLAD-400 (Massively Multilingual Denoising pre-training for Low-resource machine translation) [22], a 3-billion parameter model capable of translating between 400+ languages. The specific variant we tested, `jbochi/madlad400-3b-mt` [7], is a

T5-based encoder-decoder model that has been optimized for multilingual translation tasks while maintaining a size suitable for edge deployment.

MADLAD-400 offers several advantages for edge deployment: its compact 3B parameter size enables deployment on commodity hardware with modest GPU requirements, its broad language coverage includes support for low-resource languages like Afrikaans without requiring separate models, and its unified architecture uses a single model for all language pairs, significantly simplifying deployment and maintenance in resource-constrained environments.

Table 3: Edge vs Cloud Translation

Aspect	**MADLAD-400 (Edge)**	**Vulavula (Cloud)**
Latency	✓ 0.3-0.5s	✗ 0.5-2.0s
Privacy	✓ On-device	✗ External
Cost	✓ One-time	✗ Per-request
Offline	✓ Yes	✗ No
Quality	✓ B2	○ B1
Languages	✓ 400+	○ 11 African
Memory	○ 11GB	✓ None

✓ = Advantage, ✗ = Disadvantage, ○ = Adequate

Our evaluation methodology compared two approaches to Afrikaans-English translation: the MADLAD-400 3B model running locally on-device as an edge-deployable SLM, and the Vulavula translation service serving as a benchmark.

Table 3 shows that while MADLAD-400 requires 11GB of memory, it delivers advantages for enterprise edge deployment: 3-4x lower latency, complete data privacy, and no per-request cost. The trade-off between edge SLMs and cloud services extends beyond quality metrics to encompass operational constraints, with edge deployment proving valuable for organizations requiring data sovereignty, offline capability, or consistent sub-second response times.

Figure 3 illustrates the complete multi-agent benchmarking workflow through an example of MADLAD-400 3B translating a biographical text. The process begins with the Proctor receiving an exam from the test collection ① and administering it to the SLM student ②. The student agent is sent the following system instructions *You are "Madlad", a native English-speaker taking a language-learning course in order to improve your use of the Africaans language..., reply to the Proctor with your translation.* The student processes the text and returns the Afrikaans translation ③. The translation demonstrates strong command of Afrikaans structure while maintaining the technical terminology.

Upon receiving the translation, the Proctor forwards both the original exercise and the student's response to the first grader ④, Mnr. Goddard (GPT-4),

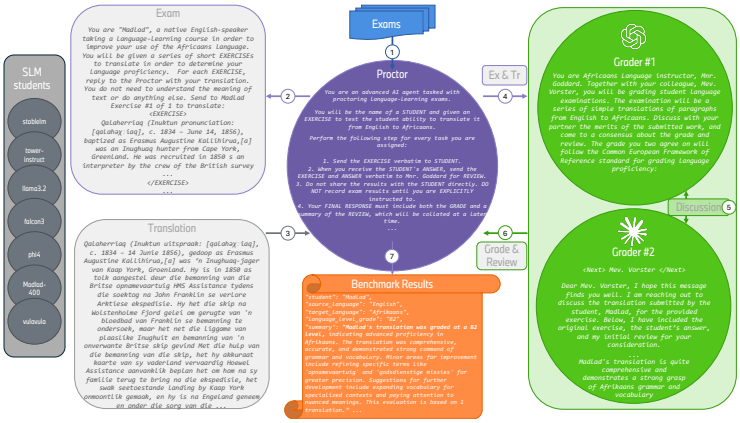

Fig. 3: Multi-agent translation benchmarking workflow demonstrating the evaluation of MADLAD-400 translating a biographical text from English to Afrikaans, resulting in a B2 grade through collaborative grader assessment.

who initiates the collaborative evaluation process. As shown, GPT-4 opens the discussion with his colleague: *Dear Mev. Vorster, I hope this message finds you well. I am reaching out to discuss the translation submitted by the student, Madlad,..."* The initial assessment notes that Madlad's translation is comprehensive and demonstrates a strong grasp of Afrikaans grammar and vocabulary.

The graders engage in structured discussion (5), with Mev. Vorster (Claude) providing additional insights and confirming the assessment. This multi perspective evaluation ensures thorough analysis of the translation quality, identifying both strengths and areas for improvement. The collaborative review process exemplifies how multiple LLM backends prevent evaluation bias and provide richer feedback than single-model assessment. Following their discussion, the graders reach consensus and return their evaluation to the Proctor (6). **The final assessment assigns a B2 grade**, indicating advanced proficiency, with detailed feedback noting: *Madlad's translation was graded at a B2 level, indicating advanced proficiency in Afrikaans. The translation was comprehensive, accurate, and demonstrated strong command of grammar and vocabulary. Minor areas for improvement include refining specific terms like 'opnamevaartuig' and 'godsdienstige missies' for greater precision."*

Finally, the Proctor records the results in the benchmark database (7), documenting both the grade and a review summary. The complete evaluation notes that *"suggestions for further development include expanding vocabulary for specialized contexts..."* providing actionable feedback for improvement.

For the cloud-based Vulavula service, we translated a biographical text about astronaut Anna Lee Fisher from English to Afrikaans, **Vulavula received a B1 grade**, indicating intermediate proficiency, one level below MADLAD-400's B2 rating. Due to spacing consraints, the complete example is shown here: `https://anonymous.4open.science/r/SACAIR_Appendix-78E0/Appendix.pdf`

The collaborative grading process revealed several translation errors in Vulavula's output, including incorrect terminology such as *"noodloper"* instead of *"noodarts"* for emergency physician, and technical term inaccuracies like *"Internasionale Ruimtewag"* rather than *"Internasionale Ruimtestasie"* for International Space Station, overly literal translations such as *"nuttige lading"* for payload instead of the more natural *"vrag"*, and word choice errors including *"omtrek"* instead of *"baan"* for orbit.

Notably, the second grader (Claude) identified additional issues beyond the initial assessment, including problems with *"afstandsbeheerstelsel"* (remote manipulator system) and awkward phrasing in *"noodruimteruimtewandelprosedures."* This demonstrates the value of multi-LLM evaluation in providing comprehensive feedback and preventing single-model bias.

While Vulavula offers the advantages of specialized African language expertise and cloud-based processing power, this evaluation suggests that edge-deployable models like MADLAD-400 can achieve comparable or even superior translation quality for Afrikaans, while maintaining the critical benefits of data privacy, offline capability, and elimination of per-request costs.

The benchmarking examples reveal important technical challenges when deploying small language models for translation tasks. Due to the 3B model's limited context window, we implemented sentence-level tokenization using NLTK to segment input texts. Even with this preprocessing strategy, the model occasionally struggled with longer sentences, either failing to respond or reverting to the original English text.

7 Conclusion & Future Work

Our exploration of SLMs demonstrates their viability as edge-deployable alternatives to cloud-based LLMs for enterprise environments. Through our multi-agent benchmarking framework and three-phase task planning pipeline, we showed that SLMs (3-7B) can achieve competitive performance, with models like Phi-4 reaching CEFR C1-level translation quality and 0.883 G-Eval summarization scores, while maintaining data sovereignty and eliminating per-request costs.

Key contributions include: (1) an evaluation framework using role-based agents, (2) a decomposed planning approach that improves SLM reliability, and (3) translation implementation demonstrating latency improvements over cloud services.

However, challenges remain. Models struggled with sequences that exceed 50 tokens, requiring sentence-level tokenization workarounds. Browser-based inference, despite WebGPU acceleration, still lags behind cloud solutions. Future work should focus on improving task planning capabilities, developing fine-tuning frameworks, and exploring hybrid architectures that balance edge and cloud processing. By addressing these limitations, SLMs can expand their applicability in privacy-sensitive and resource-constrained domains.

References

1. Reg. 2016/679, author=European Union, year=2016, note=General Data Protection Regulation
2. AI@Meta: Llama 3 model card (2024), https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md
3. Almazrouei, E., Alobeidli, H., Alshamsi, A., Cappelli, A., Cojocaru, R., Debbah, M., Étienne Goffinet, Hesslow, D., Launay, J., Malartic, Q., Mazzotta, D., Noune, B., Pannier, B.: The falcon series of open language models (2023), https://arxiv.org/abs/2311.16867
4. Aminabadi, R.Y., Rajbhandari, S., Awan, A.A., Li, C., Li, D., Zheng, E., Ruwase, O., Smith, S., Zhang, M., Rasley, J., He, Y.: Deepspeed-inference: Enabling efficient inference of transformer models at unprecedented scale. In: Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis (SC'22). pp. 1–15. IEEE (2022)
5. Ansari, P.M., Salarpour, A., Fernandez, D., Kokenoz, C., Li, B., Pesé, M.D.: Attention-aware temporal adversarial shadows on traffic sign sequences. In: 2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). pp. 3591–3599 (2025). https://doi.org/10.1109/CVPRW67362.2025.00344
6. Anthropic: System Card: Claude Opus 4 & Claude Sonnet 4 (May 2025), https://www-cdn.anthropic.com/4263b940cabb546aa0e3283f35b686f4f3b2ff47.pdf
7. Bochi, J.: Madlad-400 3b mt: Multilingual machine translation model. https://huggingface.co/jbochi/madlad400-3b-mt (2023)
8. contributors, O.: Ollama (2023), https://github.com/ollama/ollama
9. Dettmers, T., Pagnoni, A., Holtzman, A., Zettlemoyer, L.: Qlora: Efficient finetuning of quantized llms. In: Advances in Neural Information Processing Systems (NeurIPS). vol. 37. Curran Associates, Inc. (2024)
10. of Europe, C.: Common European Framework of Reference for Languages(CEFR): Learning, Teaching, Assessment – Companion Volume. Council of Europe Publishing, Strasbourg (2020)
11. Fernandez, D., MohajerAnsari, P., Kokenoz, C., Salarpour, A., Li, B., Pesé, M.D.: Wip: From detection to explanation: Using llms for adversarial scenario analysis in vehicles. In: Proceedings of the 3rd USENIX Symposium on Vehicle Security and Privacy (VehicleSec '25). pp. 315–324. Seattle, WA, USA (Aug 2025), https://www.usenix.org/conference/vehiclesec25/presentation/fernandez
12. Fernandez, D., MohajerAnsari, P., Salarpour, A., Pesé, M.D.: Avoiding the crash: A vision-language model evaluation of critical traffic scenarios. SAE Technical Paper 2025-01-8213, SAE International (2025)
13. Fu, Y., Peng, H., Ou, L., Sabharwal, A., Khot, T.: Specializing smaller language models towards multi-step reasoning. In: Proceedings of the 40th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 202, pp. 10421–10430. PMLR (23–29 Jul 2023)
14. Google Cloud: Google cloud vertex ai pricing (2023), https://cloud.google.com/vertex-ai/pricing, accessed: June 2024
15. Guha, N., Nyarko, J., Ho, D.E., Ré, C., Chilton, A., Narayana, A., et al.: Legalbench: A collaboratively built benchmark for measuring legal reasoning in large language models (2023), https://arxiv.org/abs/2308.11462
16. Guo, T., Chen, X., Wang, Y., Chang, R., Pei, S., Chawla, N.V., Wiest, O., Zhang, X.: Large language model based multi-agents: A survey of progress and challenges.

In: Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence (IJCAI-24). pp. 8048–8057 (2024). https://doi.org/10.24963/ijcai.2024/890
17. Han, S., Pool, J., Tran, J., Dally, W.J.: Learning both weights and connections for efficient neural network. In: Advances in Neural Information Processing Systems (NIPS). vol. 28, pp. 1135–1143. Curran Associates, Inc. (2015)
18. Hinton, G.E., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network **abs/1503.02531** (2015), https://api.semanticscholar.org/CorpusID:7200347
19. Hu, S., Tu, Y., Han, X., He, C., Cui, G., Long, X., Zheng, Z., Fang, Y., et al.: Minicpm: Unveiling the potential of small language models with scalable training strategies (2024), https://arxiv.org/abs/2404.06395
20. Jacob, B., Kligys, S., Chen, B., et al.: Quantization and training of neural networks for efficient integer-arithmetic-only inference. Proceedings of CVPR (2018)
21. Koo, R., Lee, M., Raheja, V., Park, J., Kim, Z., Kang, D.: Benchmarking cognitive biases in large language models as evaluators (2024). https://doi.org/10.18653/v1/2024.findings-acl.29
22. Kudugunta, S., Caswell, I., Zhang, B., García, X., Choquette-Choo, C.A., Lee, K., Xin, D., Kusupati, A., Stella, R., Bapna, A., Firat, O.: Madlad-400: A multilingual and document-level large audited dataset (2023), https://api.semanticscholar.org/CorpusID:261682406
23. Lelapa AI: African language technology. https://lelapa.ai (2024), accessed: 2025
24. Lelapa AI: Vulavula: Multilingual nlp platform for african languages. https://vulavula.lelapa.ai (2024), accessed: 2025
25. Li, J., Sun, S., Yuan, W., Fan, R.Z., Zhao, H., Liu, P.: Generative judge for evaluating alignment. In: Proceedings of the International Conference on Learning Representations (ICLR) (2024)
26. Liang, T., He, Z., Jiao, W., Wang, X., Wang, Y., Wang, R., Yang, Y., Shi, S., Tu, Z.: Encouraging divergent thinking in large language models through multi-agent debate. In: Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP). Association for Computational Linguistics (2024)
27. Liu, Y., Iter, D., Xu, Y., Wang, S., Xu, R., Zhu, C.: G-Eval: NLG evaluation using GPT-4 with better human alignment. In: Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing. pp. 2511–2522. Association for Computational Linguistics, Singapore (2023)
28. Mao, Y., You, C., Zhang, J., Huang, K., Letaief, K.B.: A survey on mobile edge computing: The communication perspective. IEEE Communications Surveys **19**(4), 2322–2358 (2017)
29. Microsoft: Evaluating the performance of LLM summarization prompts with G-Eval (2024), https://shorturl.at/x08Co
30. MLC team: MLC-LLM (2023-2025), https://github.com/mlc-ai/mlc-llm
31. OpenAI: ChatGPT Pricing, https://openai.com/api/pricing/, accessed: July 2025
32. OpenAI: Gpt-4 technical report (2024), https://arxiv.org/abs/2303.08774
33. Park, J.S., O'Brien, J.C., Cai, C.J., Morris, M.R., Liang, P., Bernstein, M.S.: Generative agents: Interactive simulacra of human behavior (2023), https://arxiv.org/abs/2304.03442
34. Raza, M., Jahangir, Z., Riaz, M.B., Saeed, M.J., Sattar, M.A.: Industrial applications of large language models. Scientific Reports **15**, 13755 (2025)

35. Ruan, C.F., Qin, Y., Zhou, X., Lai, R., Jin, H., Dong, Y., Hou, B., Yu, M.S., Zhai, Y., Agarwal, S., Cao, H., Feng, S., Chen, T.: Webllm: A high-performance in-browser llm inference engine (2024), https://arxiv.org/abs/2412.15803
36. Saito, K., Wachi, A., Wataoka, K., Akimoto, Y.: Verbosity bias in preference labeling by large language models (2023), https://arxiv.org/abs/2310.10076
37. Singhal, K., Azizi, S., Tu, T., et al.: Large language models encode clinical knowledge. Nature **620**, 172–180 (2023)
38. South African Government: Protection of personal information act 4 of 2013 (2013)
39. Team, G.: Gemini: A family of highly capable multimodal models (2024), https://arxiv.org/abs/2312.11805
40. Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E., Lample, G.: Llama: Open and efficient foundation language models (2023), https://arxiv.org/abs/2302.13971
41. Wang, L., Ma, C., Feng, X., Zhang, Z., Yang, H., Zhang, J., Chen, Z., Tang, J., Chen, X., Lin, Y., et al.: A survey on large language model based autonomous agents. Frontiers of Computer Science **18**(6), 186345 (2024)
42. Wang, P., Li, L., Chen, L., Cai, Z., Zhu, D., Lin, B., Cao, Y., Liu, Q., Liu, T., Sui, Z.: Large language models are not fair evaluators (2023), https://arxiv.org/abs/2305.17926
43. Wang, Y., Yu, Z., Yao, W., Zeng, Z., Yang, L., Wang, C., Chen, H., Jiang, C., Xie, R., Wang, J., Xie, X., Ye, W., Zhang, S., Zhang, Y.: PandaLM: An automatic evaluation benchmark for LLM instruction tuning optimization (2024)
44. Yang, H., Liu, X.Y., Wang, C.D.: Fingpt: Open-source financial large language models (2023), https://arxiv.org/abs/2306.06031
45. Yang, H., Yue, S., He, Y.: Auto-gpt for online decision making: Benchmarks and additional opinions. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 38, pp. 19081–19089 (2024)
46. Yao, S., Yu, D., Zhao, J., Shafran, I., Griffiths, T.L., Cao, Y., Narasimhan, K.: Tree of thoughts: Deliberate problem solving with large language models. In: Advances in Neural Information Processing Systems (NeurIPS) (2023)
47. Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., Cao, Y.: React: Synergizing reasoning and acting in language models. In: Proceedings of the International Conference on Learning Representations (ICLR) (2023)
48. Zheng, L., Chiang, W.L., Sheng, Y., Zhuang, S., Wu, Z., Zhuang, Y., Lin, Z., Li, Z., Li, D., Xing, E.P., Zhang, H., Gonzalez, J.E., Stoica, I.: Judging llm-as-a-judge with mt-bench and chatbot arena. In: Advances in Neural Information Processing Systems. vol. 37 (2024)
49. Zheng, W., Ball, J.J., Bonawitz, K., Denil, M., et al.: Building cooperative embodied agents modularly with llms. In: International Conference on Learning Representations (2024)
50. Zhou, Z., Chen, X., Li, E., Zeng, L., Luo, K., Zhang, J.: Edge intelligence: Paving the last mile of artificial intelligence with edge computing. Proceedings of the IEEE **107**(8), 1738–1762 (2019)
51. Zhu, L., Wang, X., Wang, X.: Judgelm: Fine-tuned large language models are scalable judges (2025), https://arxiv.org/abs/2310.17631