Small Language Models are the Future of Agentic AI

Peter Belcak[1] **Greg Heinrich**[1] **Shizhe Diao**[1] **Yonggan Fu**[1] **Xin Dong**[1]
Saurav Muralidharan[1] **Yingyan Celine Lin**[1,2] **Pavlo Molchanov**[1]
[1]NVIDIA Research [2]Georgia Institute of Technology
agents-research@nvidia.com

Abstract

Large language models (LLMs) are often praised for exhibiting near-human performance on a wide range of tasks and valued for their ability to hold a general conversation. The rise of agentic AI systems is, however, ushering in a mass of applications in which language models perform a small number of specialized tasks repetitively and with little variation.

Here we lay out the position that small language models (SLMs) are *sufficiently powerful, inherently more suitable, and necessarily more economical for many invocations in agentic systems, and are therefore the future of agentic AI*. Our argumentation is grounded in the current level of capabilities exhibited by SLMs, the common architectures of agentic systems, and the economy of LM deployment. We further argue that in situations where general-purpose conversational abilities are essential, heterogeneous agentic systems (i.e., agents invoking multiple different models) are the natural choice. We discuss the potential barriers for the adoption of SLMs in agentic systems and outline a general LLM-to-SLM agent conversion algorithm.

Our position[1], formulated as a value statement, highlights the significance of the operational and economic impact even a partial shift from LLMs to SLMs is to have on the AI agent industry. We aim to stimulate the discussion on the effective use of AI resources and hope to advance the efforts to lower the costs of AI of the present day. Calling for both contributions to and critique of our position, we commit to publishing all such correspondence at `research.nvidia.com/labs/lpr/slm-agents`.

1 Introduction

The deployment of agentic artificial intelligence is on a meteoric rise. Recent surveys show that more than a half of large IT enterprises are actively using AI agents, with 21% having adopted just within the last year [14]. Aside from the users, markets also see substantial economic value in AI agents: As of late 2024, the agentic AI sector had seen more than USD 2bn in startup funding, was valued at USD 5.2bn, and was expected to grow to nearly USD 200bn by 2034 [46, 51]. Put plainly, there is a growing expectation that AI agents will play a substantial role in the modern economy.

The core components powering most modern AI agents are (very) large language models [52, 48]. It is the LLMs that provide the foundational intelligence that enables agents to make strategic decisions about when and how to use available tools, control the flow of operations needed to complete tasks, and, if necessary, to break down complex tasks into manageable subtasks and to perform reasoning for action planning and problem-solving [52, 17]. A typical AI agent then simply communicates with a chosen LLM API endpoint by making requests to centralized cloud infrastructure that hosts these models [52].

[1]The views and positions expressed in this paper are those of the authors and do not necessarily reflect the views or positions of any entities they represent.

Preprint. Under review.

LLM API endpoints are specifically designed to serve a large volume of diverse requests using one generalist LLM. This operational model is deeply ingrained in the industry. In fact, it forms the foundation of substantial capital investment in the hosting cloud infrastructure – estimated at USD 57bn [16]. It is anticipated that this operational model will remain the cornerstone of the industry and that the large initial investment will deliver returns comparable to traditional software and internet solutions within 3-4 years [57].

In this work, we recognize the dominance of the standard operational model but verbally challenge one of its aspects, namely the custom that the agents' requests to access language intelligence are – in spite of their comparative simplicity – handled by singleton choices of generalist LLMs. We state (Section 2), argue (Section 3), and defend (Section 4) the position that the **small, rather than large, language models are the future of agentic AI**. We, however, recognize the business commitment and the now-legacy praxis that is the cause for the contrary state of the present (Section 5). In remedy, we provide an outline of a conversion algorithm for the migration of agentic applications from LLMs to SLMs (Section 6), and call for a wider discussion (Section 7). If needed to concretize our stance, we attach a set of short case studies estimating the potential extent of LLM-to-SLM replacement in selected popular open-source agents (Section B).

2 Position

2.1 Definitions

For the purpose of concretizing our position, we use the following working definitions:

WD1 A *SLM* is a LM that can fit onto a common consumer electronic device and perform inference with latency sufficiently low to be practical when serving the agentic requests of one user.

WD2 An *LLM* is a LM that is not a *SLM*.

We justify the wording of these definitions in Section A, but note that their choice has little bearing on the essence of our position. We note that as of 2025, we would be comfortable with considering most models below 10bn parameters in size to be SLMs.

We use the words *agent* and *agentic system* interchangeably, preferring the former when emphasizing the software with some agency as a whole (e.g., "as seen in popular coding agents") and the latter when highlighting the systems aspect of the agentic application as a sum of its components (e.g., "not all LMs of an agentic system are replaceable by SLMs"). For brevity, we focus on LMs as the bedrock of agentic applications and do not explicitly consider vision-language models, although we note that our position and most of our arguments readily extend to vision-language models as well.

2.2 Statement

We contend that SLMs are

V1 principally *sufficiently powerful* to handle language modeling errands of agentic applications;

V2 inherently *more operationally suitable* for use in agentic systems than LLMs;

V3 necessarily *more economical* for the vast majority of LM uses in agentic systems than their general-purpose LLM counterparts by the virtue of their smaller size;

and that on the basis of views V1–V3 SLMs are the future of agentic AI.

The phrasing of our position is deliberate. In its statement, we wish to convey that the described future development is ultimately a necessary consequence of the differences between SLMs and LLMs if the natural priorities are followed. We do not make a recommendation or try to impose an obligation — we make a statement of what we see as a faithful reflection of the community's values in this context.

2.3 Elaboration

We assert that the dominance of LLMs in the design of AI agents is both excessive and misaligned with the functional demands of most agentic use cases. While LLMs offer impressive generality and

conversational fluency, the majority of agentic *sub*tasks in deployed agentic systems are repetitive, scoped, and non-conversational—calling for models that are efficient, predictable, and inexpensive. In this context, SLMs not only suffice, but are often preferable. They offer several advantages: lower latency, reduced memory and computational requirements, and significantly lower operational costs, all while maintaining adequate task performance in constrained domains.

Our position stems from a pragmatic view of language model usage patterns within agentic architectures. These systems typically decompose complex goals into modular sub-tasks, each of which can be reliably handled by specialized or fine-tuned SLMs. We argue that insisting on LLMs for all such tasks reflects a misallocation of computational resources—one that is economically inefficient and environmentally unsustainable at scale.

Moreover, in cases where general reasoning or open-domain dialogue is essential, we advocate for heterogeneous agentic systems, where SLMs are used by default and LLMs are invoked selectively and sparingly. This modular composition — combining the precision and efficiency of SLMs with the generality of LLMs — enables the construction of agents that are both cost-effective and capable.

Ultimately, we observe that shifting the paradigm from LLM-centric to SLM-first architectures represents to many not only a technical refinement but also a Humean moral ought. As the AI community grapples with rising infrastructure costs and environmental concerns, adopting and normalizing the use of SLMs in agentic workflows can play a crucial role in promoting responsible and sustainable AI deployment.

3 Position Arguments

We support views **V1**–**V3** by the following non-exclusive arguments.

3.1 SLMs are already sufficiently powerful for use in agents

A1 SLMs are sufficiently powerful to take the place of LLMs in agentic systems. This argument stands in support of view **V1**.

Over the past few years, the capabilities of small language models have advanced significantly. Although the LM scaling laws remain observed, the scaling curve between model size and capabilities is becoming increasingly steeper, implying that the capabilities of newer small language models are much closer to those of previous large language models. Indeed, recent advances show that well-designed small language models can meet or exceed the task performance previously attributed only to much larger models.

Extensive comparisons with large models have been conducted in the individual works cited below, but not all capabilities assessed by benchmarks are essential to their deployment in the agentic context. Here we highlight their aptitude for commonsense reasoning (an indicator of basic understanding), tool calling and code generation (both indicators of the ability to correctly communicate across the model→tool/code interface; see Figure 1; [77, 78]), and instruction following (ability to correctly respond back across the code←model interface; [83]). In each case, we also quote the efficiency increase if stated by the authors.

- **Microsoft Phi series.** Phi-2 (2.7bn) achieves commonsense reasoning scores and code generation scores on par with 30bn models while running $\sim$15$\times$ faster [37]. Phi-3 small (7bn) [3] achieves language understanding and commonsense reasoning on par with and code generation scores running up to 70bn models of the same generation.
- **NVIDIA Nemotron-H family.** The 2/4.8/9bn hybrid Mamba-Transformer models achieve instruction following and code-generation accuracy comparable to dense 30bn LLMs of the same generation at an order-of-magnitude fraction of the inference FLOPs [9].
- **Huggingface SmolLM2 series.** SmolLM2 family of compact language models with sizes ranging from 125mn to 1.7bn parameters [6] each run up in their language understanding, tool calling, and instruction following performance to 14bn contemporaries while matching 70bn models of 2 years prior.

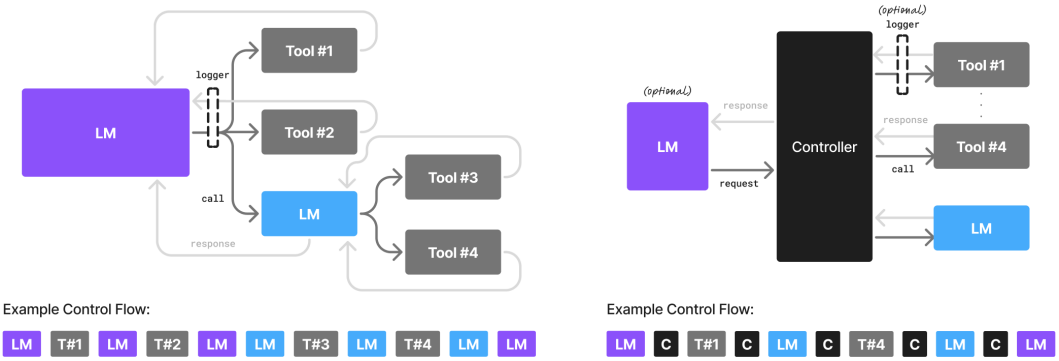

Figure 1: An illustration of agentic systems with different modes of agency. *Left: Language model agency.* The language model acts both as the HCI and the orchestrator of tool calls to carry out a task. *Right: Code agency.* The language model fills the role of the HCI (optionally) while a dedicated controller code orchestrates all interactions.

- **NVIDIA Hymba-1.5B.** This Mamba-attention hybrid-head SLM demonstrates best instruction accuracy and 3.5× greater token throughput than comparably-sized transformer models [23]. On instruction following, it outperforms larger 13bn models.
- **DeepSeek-R1-Distill series.** DeepSeek-R1-Distill is a family of reasoning models featuring 1.5-8bn sizes, trained on samples generated by DeepSeek-R1 [19]. They demonstrate strong commonsense reasoning capabilities. Notably, the DeepSeek-R1-Distill-Qwen-7B model outperforms large proprietary models such as Claude-3.5-Sonnet-1022 and GPT-4o-0513.
- **DeepMind RETRO-7.5B**: Retrieval-Enhanced Transformer (RETRO) is a 7.5bn parameter model augmented with an extensive external text database, achieving performance comparable to GPT-3 (175B) on language modeling while using 25× fewer parameters [10].
- **Salesforce xLAM-2-8B.** The 8bn model achieves state-of-the-art performance on tool calling despite is relatively modest size, surpassing frontier models like GPT-4o and Claude 3.5 [81].

Note that on top of competitive off-the-shelf performance, the reasoning capabilities of SLMs can be enhanced by light-weight selective finetuning [45] or at inference time with self-consistency, verifier feedback, or tool augmentation — e.g., Toolformer (6.7bn) outperforms GPT-3 (175bn) via API use [65], and 1-3bn models have rivaled 30bn+ LLMs on math problems via structured reasoning [84].

In sum, with modern training, prompting, and agentic augmentation techniques, capability — not the parameter count — is the binding constraint. SLMs now supply sufficient reasoning power for a substantial portion of agentic invocations, making them not just viable, but comparatively more suitable than LLMs for modular and scalable agentic systems.

3.2 SLMs are more economical in agentic systems

A2 SLMs are more economical than LLMs in agentic systems. This argument supports view **V3**.

Small models provide significant benefits in cost-efficiency, adaptability, and deployment flexibility. These advantages are specifically valuable in agentic workflows where specialization and iterative refinement are critical. Section 3.1 detailed a number of efficiency comparisons of the listed SLMs to relevant LLMs. Here we draw a more encompassing picture to support argument **A2**.

- **Inference efficiency.** Serving a 7bn SLM is 10–30× cheaper (in latency, energy consumption, and FLOPs) than a 70–175bn LLM, enabling real-time agentic responses at scale [70, 68, 36, 53]. Recent advances in inference operating systems such as NVIDIA Dynamo [24] explicitly provide support for high-throughput, low-latency SLM inference in both cloud and edge deployments. In addition, since SLMs require less or no parallelization across GPUs and nodes, the maintenance and operation of the serving infrastructure comes at a lower expense as well (see counter-argument **CA4** and argument **A13**).

- **Fine-tuning agility.** Parameter-efficient (e.g., LoRA [33] and DoRA [43]), low-resource [7], and full-parameter finetuning for SLMs require only a few GPU-hours, allowing behaviors to be added, fixed, or specialized overnight rather than over weeks [70].
- **Edge deployment.** Advances in on-device inference systems such as ChatRTX [59] demonstrate local execution of SLMs on consumer-grade GPUs, showcasing real-time, offline agentic inference with lower latency and stronger data control.
- **Parameter and embedding space utilization.** At the outset, LLMs appear to operate as monoliths involving a large amount of parameters representing swathes of compressed information in the production of their outputs. On a closer look, however, many of the embeddings passing through these systems are very sparse, engaging only a fraction of their parameters for any single input [69, 44] and being effectively compressible [8, 29]. That this behavior appears to be more subdued in SLMs [69, 75] suggests that SLMs may be fundamentally more efficient by the virtue of having a smaller proportion of their parameters contribute to the inference cost without a tangible effect on the output.

Modular system design. The position outlined in [56] presents a thorough argument in favor of composite agentic systems. Here we note that the approach of leveraging several models of varying sizes aligns well with the real-world heterogeneity of agentic tasks and is already slowly being incorporated into major software development frameworks [28]. Furthermore, this newly discovered sense for modularity in the context of agents allows for the easy addition of new skills and the ability to adapt to changing requirements, and is consistent with the push for modularity in language model design [27, 12, 40].

The above-mentioned "Lego-like" composition of agentic intelligence—scaling out by adding small, specialized experts instead of scaling up monolithic models—yields systems that are cheaper, faster to debug, easier to deploy, and better aligned with the operational diversity of real-world agents. When combined with tool calling, caching, and fine-grained routing, SLM-first architectures appear to offer the best path forward for cost-effective, modular, and sustainable agentic AI.

3.3 SLMs are more flexible

A3 SLMs possess greater operational flexibility in comparison to LLMs. This argument stands in support of views **V2** and **V3**.

Due to their small size and the associated reduction in pre-training and fine-tuning costs (Section 3.2), SLMs are inherently more flexible than their large counterparts when appearing in agentic systems. As such, it becomes much more affordable and practical to train, adapt, and deploy multiple specialized expert models for different agentic routines. This efficiency enables rapid iteration and adaptation, making it feasible to address evolving user needs, including supporting new behaviors, meeting new output formatting requirements, and complying with changing local regulation in selected markets [73, 41, 72].

Democratization. One particularly notable and desirable consequence of SLM flexibility when put in place of LLMs is the ensuing democratization of agents. When more individuals and organizations can participate in developing language models with the aim for deployment in agentic systems, the aggregate population of agents is more likely to represent a more diverse range of perspectives and societal needs. This diversity can then help with reducing the risk of systemic biases and encourage competition and innovation. With more actors entering the field to create and refine models, the field will advance more rapidly [38].

3.4 Agents expose only very narrow LM functionality

A4 Agentic applications are interfaces to a limited subset of LM capabilities. This supports views **V1** and **V2**.

An AI agent is essentially a heavily instructed and externally choreographed gateway to a language model featuring a human-computer interface and a selection of tools that, when engaged correctly, do something of utility [73]. From this perspective, the underlying large language model that was engineered to be a powerful generalist is through a set of tediously written prompts and meticulously

orchestrated context management restricted to operate within a small section of its otherwise large pallet of skills. Thus, we argue that a SLM appropriately fine-tuned for the selected prompts would suffice while having the above-mentioned benefits of increased efficiency and greater flexibility.

It could be argued back that the careful interfacing with a generalist LLM is necessary for strong performance on the narrow task because of the LLM's better understanding of the broader language and the world (alternative view **AV1**). This is addressed in Section 4.1.

3.5 Agentic interactions necessitate close behavioral alignment

A5 Agentic interactions necessitate close behavioral alignment. This aligns with view **V2**.

A typical AI agent has frequent interactions with code, be it through LM tool calling or by returning output that is to be parsed by a piece of agentic code that makes the LM call [52]. It is essential for the success of these interactions that the generated tool call and the generated output conform to strict formatting requirements imposed on it by the order, typing, and nature of the tool's parameters, and the expectation of the code invoking the LM, respectively. In such cases, it becomes unnecessary for the model to handle multiple different formats (e.g. JSON/XML/Python for tool calls and XML/YAML/Markdown/Latex for output [54]), as only one would be chosen for consistency across the agentic application. It is also undesirable for the model to make the occasional hallucinatory mistake and respond in a format different from that being expected by the "code parts" of the agentic system. It is because of this that the SLM trained with a single formatting decision enforced during its post-training or encouraged through additional fine-tuning at a low cost is preferable over a general-purpose LLM in the context of AI agents.

3.6 Agentic systems are naturally heterogeneous

A6 Agentic systems naturally allow for heterogeneity in the selection of models that they use. This aligns with view **V2**.

A language model can itself be a tool called by another language model. Likewise, every time the agent's code invokes a language model, it can, in principle, choose any language model. This is illustrated in Figure 1. We argue that incorporating multiple language models of different sizes and capabilities for queries or operations of different levels of complexity offers a natural way for the introduction of SLMs. In the context of Figure 1-*Left*, an LLM can be used for the model with the root agency, while a SLM could be used for the subordinate LM. In Figure 1-*Right*, all LMs could in principle be specialized SLMs: one for conversationality, another one for carrying out controller-defined language modeling tasks.

3.7 Agentic interactions are natural pathways for gathering data for future improvement

A7 Agentic interactions are a good source for data for future model improvement. This is fundamentally supportive of view **V2**.

As noted in Section 3.4, invocations of tools and language models during an agentic process are often accompanied by careful prompting that focuses the language model on delivering the narrow functionality that is required at the time. Each one of these invocations is itself a natural source of data for future improvement (under the necessary assumption that no non-retainable confidential data is being processed). A listener decorating the tool/model call interface can gather specialized instruction data that can later be used to produce a fine-tune an expert SLM and lower the cost of that call in the future (see `logger` in Figure 1). We argue that this avenue is enabled by the architecture of agents [52] and produces high-quality organic data (that can be further post-filtered by considering the overall success of the workflow), thus making the production of expert SLMs to stand in place of LLMs a natural step in agent deployment — not just an auxiliary effort.

4 Alternative Views

The following significant alternative views have been expressed in the academic and popular literature.

4.1 LLM generalists will always have the advantage of more general language understanding

AV1 Let $\mathcal{T}$ be a single task using general language and let L, S be a large and a small language model of the same generation, respectively. The performance of L on $\mathcal{T}$ will always trump that of S.

This alternative view disputes view **V2** and rests on the following counter-arguments:

CA1 There is a non-negligible body of empirical evidence of the superiority of large language models in general language understanding over small language models of the same generation. LLMs acquire their language understanding capabilities in accordance with scaling laws [18]. Their larger scale then enables them to demonstrate better performance across a wide array of specialized natural language tasks, including text generation, translation, and reasoning, outperforming small models trained both (a) in the same general fashion and (b) from scratch specifically for these tasks [58]. It can then be said that to claim otherwise is to contradict the LM scaling laws [32, 31].

CA2 Moreover, recent studies also purport that LLMs possess a "semantic hub" mechanism, which has been hypothesized to enable them to integrate and abstract semantic information from various modalities and languages in a generalized manner [80]. If true, LLMs could be thought to generalize knowledge across languages and domains far more effectively than smaller models, which under the same study lack the capacity for the presence of such a hub [80]. It can then argued that while small language models may be efficient for narrowly defined or highly specialized tasks, their limited scale fundamentally restricts their ability to achieve the same level of general language understanding in these specialized as LLMs because of the lack of room for the internalization of complex abstractions.

A conclusion can be then drawn that LLM generalist models will always retain the advantage of universally better performance on language tasks, no matter how narrowly defined, over small language models of the same generation. This to be to their advantage over SLMs when deployed in agentic applications.

Rebuttal. The above alternative view is the most popularly cited belief against the use of SLMs, even when only a narrow language task needs to be performed [2, 71, 30, 1].

We believe that counter-argument **CA1** is too limited to attack view **V2**, namely because

A8 Popular scaling law studies assume the model architecture to be kept constant [32, 31] within the same generation, whereas the recent work on small language model training demonstrates that there are distinct performance benefits to considering different architectures for different model sizes [23, 9].

A9 The flexibility of small language models (Section 3.3) comes to the rescue. A small language model can be easily fine-tuned for the task $\mathcal{T}$ of alternative view **AV1** to perform to the desired level of reliability. This is unaccounted for in scaling law studies.

A10 Reasoning (or, more generally, test-time compute scaling; see Section 3.2) is significantly more affordable. A small language model, still retaining its benefits of greater cross-device agility can be reasonable expected to be scalable at inference time to the desired level of reliability.

We also believe that counter-argument **CA2** is too arcane to attack view **V2** because

A11 The utility of the purported "semantic hub" shows itself when tasks or inputs at hand to be processed by the LM are complex. However, advanced agentic systems are either designed in their entirety or at least actively prompted to perform decompositions of complex problems and inputs [52, 17]. Therefore, we argue to the contrary that invocations of small language models within agentic systems would be on appropriately broken-down into sub-tasks so simple that any general abstract understanding due to the hub would be of little utility.

4.2 LLM inference will still be cheaper because of their centralization

AV2 The per-token inference cost benefit of the smallness of specialized SLMs in agentic applications is dwarfed by the economy of scale.

It could be argued that the analysis in argument **A2** that put forth in favor of view **V3** was ignorant of the wider business of AI model deployment:

CA3 It is more difficult to fully utilize and properly balance the load for an expert SLM inference endpoint than it is for a generalist LLM endpoint [70, 25].

CA4 The costs of inference infrastructure setup combined with the costs of acquiring and retaining talent for its upkeep are often omitted in inference cost calculations but figure more prominently if the deployment of (S)LMs became the responsibility of the agent service developer. Early industrial reports point to considerable costs associated with these operations [39, 13, 67].

Acknowledgment. We acknowledge that alternative view **AV2** is a valid view, with the exact economical considerations being highly case-specific. We believe that the jury is still out on alternative view **AV2**, but that several factors hint that view **V3** might prevail:

A12 Recent improvements in inference scheduling and large inference system modularization offer unprecedented levels of inference system flexibility in monolithic computing clusters [85, 60, 50], countering the traditional stance expressed in counter-argument **CA3**.

A13 The most recent analyses on the set-up costs of inference infrastructure indicate a consistent falling trend due to underlying technological reasons [82, 4].

4.3 Equally possible worlds

AV3 Both the agentic world utilizing SLMs and agentic world utilizing LLMs are equally possible worlds, but the "LLM agentic world" has a considerable head start in terms of deployment practice and optimization, and the industry inertia already funnels efforts into innovation solely in that direction.

Acknowledgment. We acknowledge alternative view **AV3** as a distinct possibility, but maintain the position that the weight of advantages described across arguments **A1**–**A7** can plausibily overturn the present state of affairs.

5 Barriers to Adoption

It would be prudent to ask oneself: If the arguments **A1**–**A7** are truly compelling, why do the ever newer generations of agents seemingly just perpetuate the status quo of using generalist LLMs?

We believe the following to be among the today's main barriers to wide-spread adoption of SLMs:

B1 **Large amounts of upfront investment in centralized LLM inference infrastructure.** As detailed in Section 1, large capital bets have been made on the centralized LLM inference being the leading paradigm in providing AI services in the future. As such, the industry has been much quicker at building the tools and infrastructure to that end, omitting any considerations for the possibility that more decentralized SLM or on-device inference might be equally feasible in the near future.

B2 **Use of generalist benchmarks in SLM training, design, and evaluation.** It must be pointed out that much of the work on SLM design and development follows the tracks of LLM design, focusing on the same generalist benchmarks in their development [47, 61]. On this point, [23] notes that if one focuses solely on benchmarks measuring the agentic utility of agents, the studied SLMs easily outperform larger models.

B3 **Lack of popular awareness.** SLMs often do not receive the level of marketing intensity and press attention LLMs do, despite their better suitability in many industrial scenarios.

We note that barriers **B1**–**B3** are practical hurdles and far from being fundamental flaws of the SLM technology in the context of agentic AI. With advanced inference scheduling systems such as Dynamo [24], barrier **B1** is being reduced to a mere effect of inertia. barrier **B2** is becoming increasingly recognized in the field [23, 37], and it would be natural for barrier **B3** to fall once the economic benefits of SLM deployment in agentic applications (argument **A2**) are better known. With the inertia of barrier **B1** in particular, we do not endeavor to give a timeline for the retreat of these barriers or the popular adoption of SLMs.

6 LLM-to-SLM Agent Conversion Algorithm

The very nature of agentic applications enables them to eventually switch from using LLM generalists to using SLM specialists at many of their interfaces. In the following steps, we outline an algorithm that describes one possible way to carry out the change of the underlying model painlessly.

- **S1 Secure usage data collection.** The initial step involves deploying instrumentation to log all non-HCI agent calls, capturing input prompts, output responses, contents of individual tool calls, and optionally latency metrics for a later targeted optimization. In terms of implementation, it is the recommended practice to set up encrypted logging pipelines with role-based access controls [55] and anonymize all data with respect to its origins before storage [74]. See `logger` in Figure 1 for an illustration.
- **S2 Data curation and filtering.** Begin collecting data through the pipelines of step **S1**. Once a satisfactory amount of data has been collected (10k-100k examples being sufficient for fine-tuning of small models as a rule of thumb [5, 22]), it is necessary to remove any PII, PHI, or any other application-specific sensitive data that could cause a data leak across user accounts once used to produce a SLM specialist. Many typical varieties of sensitive data can be detected and masked or removed using popular automated tools for dataset preparation [64, 62]. Application specific inputs (e.g. legal or internal documents) can be often be automatically paraphrased to obfuscate named entities and numerical details without compromising the general information content [11, 79, 76].
- **S3 Task clustering.** Employ unsupervised clustering techniques on the collected prompts and agent actions to identify recurring patterns of requests or internal agent operations [35, 42, 21]. These clusters help define candidate tasks for SLM specialization. The granularity of tasks will depend on the diversity of operations; common examples include intent recognition, data extraction, summarization of specific document types, or code generation with respect to tools available to the agent.
- **S4 SLM selection.** For each identified task, select one or more candidate SLMs. Criteria for selection include the SLM's inherent capabilities (e.g., instruction following, reasoning, context window size), its performance on relevant benchmarks for the task type, its licensing, and its deployment footprint (memory, computational requirements). Models of Section 3.2 serve as good starting candidates.
- **S5 Specialized SLM fine-tuning.** For each selected task and corresponding SLM candidate, prepare a task-specific dataset from the curated data collected in steps **S2** and **S3**. Then, fine-tune the chosen SLMs on these specialized datasets. PEFT techniques such as LoRA [34] or QLoRA [20] can be leveraged to reduce computational costs and memory requirements associated with fine-tuning, making the process more accessible. Full fine-tuning can also be considered if resources permit and maximal adaptation is required. In some cases, it may be beneficial to use knowledge distillation, where the specialist SLM is trained to mimic the outputs of the more powerful generalist LLM on the task-specific dataset. This can help transfer some of the more nuanced capabilities of the LLM to the SLM.
- **S6 Iteration and refinement.** One may retrain the SLMs and the router model periodically with new data to maintain performance and adapt to evolving usage patterns. This forms a continuous improvement loop, returning to step **S2** or step **S4** as appropriate.

7 Call for Discussion

The agentic AI industry is showing the signs of a promise to have a transformative effect on white collar work and beyond.

It is the view of the authors that any expense savings or improvements on the sustainability of AI infrastructure would act as a catalyst for this transformation, and that it is thus eminently desirable to explore all options for doing so.

We therefore call for both contributions to and critique of our position, to be directed to `agents@nvidia.com`, and commit to publishing all such correspondence at `research.nvidia.com/labs/lpr/slm-agents`.

References

[1] Aashima. Small language models vs. llms: Finding the right fit for your needs, October 2024. Accessed: 2025-05-09.

[2] ABBYY. Small language models vs. large language models, November 2024. Accessed: 2025-05-09.

[3] Marah Abdin, Jyoti Aneja, Hany Awadalla, Ahmed Awadallah, Ammar Ahmad Awan, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Jianmin Bao, Harkirat Behl, et al. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*, 2024.

[4] Adyog. The economics of ai training and inference: How deepseek broke the cost curve, February 2025. Accessed: 2025-05-09.

[5] Ishika Agarwal, Krishnateja Killamsetty, Lucian Popa, and Marina Danilevksy. Delift: Data efficient language model instruction fine tuning. *arXiv preprint arXiv:2411.04425*, 2024.

[6] Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, Gabriel Martín Blázquez, Guilherme Penedo, Lewis Tunstall, Andrés Marafioti, Hynek Kydlíček, Agustín Piqueres Lajarín, Vaibhav Srivastav, Joshua Lochner, Caleb Fahlgren, Xuan-Son Nguyen, Clémentine Fourrier, Ben Burtenshaw, Hugo Larcher, Haojun Zhao, Cyril Zakka, Mathieu Morlon, Colin Raffel, Leandro von Werra, and Thomas Wolf. Smollm2: When smol goes big – data-centric training of a small language model, 2025.

[7] Peter Belcak, Greg Heinrich, Jan Kautz, and Pavlo Molchanov. Minifinetuning: Low-data generation domain adaptation through corrective self-distillation. *arXiv preprint arXiv:2506.15702*, 2025.

[8] Peter Belcak and Roger Wattenhofer. Tiny transformers excel at sentence compression. *arXiv preprint arXiv:2410.23510*, 2024.

[9] Aaron Blakeman, Aarti Basant, Abhinav Khattar, Adithya Renduchintala, Akhiad Bercovich, Aleksander Ficek, Alexis Bjorlin, Ali Taghibakhshi, Amala Sanjay Deshmukh, Ameya Sunil Mahabaleshwarkar, et al. Nemotron-h: A family of accurate and efficient hybrid mamba-transformer models. *arXiv preprint arXiv:2504.03624*, 2025.

[10] Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Bogdan Damoc, Aidan Clark, Jan Kramár, et al. Improving language models by retrieving from trillions of tokens. *arXiv preprint arXiv:2112.04426*, 2022.

[11] Michael Brennan, Sadia Afroz, and Rachel Greenstadt. Adversarial stylometry: Circumventing authorship recognition to preserve privacy and anonymity. *ACM Transactions on Information and System Security (TISSEC)*, 15(3):1–22, 2012.

[12] Ruisi Cai, Saurav Muralidharan, Greg Heinrich, Hongxu Yin, Zhangyang Wang, Jan Kautz, and Pavlo Molchanov. Flextron: Many-in-one flexible large language model. In *Proceedings of the 41st International Conference on Machine Learning (ICML 2024)*, 2024.

[13] Michael Chui, Bryce Hall, Helen Mayhew, Alex Singla, and Alexander Sukharevsky. The state of ai in 2022—and a half decade in review, December 2022. Accessed: 2025-05-09.

[14] Cloudera, Inc. 96% of enterprises are expanding use of ai agents, according to latest data from cloudera, April 2025. Accessed: 2025-05-08.

[15] Planck Collaboration et al. Planck 2018 results. vi. cosmological parameters. *Astronomy & Astrophysics*, 641:A6, 2020.

[16] Colliers. 2025 data center marketplace: Balancing unprecedented opportunity with strategic risk. U.s. research report, Colliers, 2025.

[17] DAIR.AI. Llm agents, April 2024. Accessed: 2025-05-08.

[18] Badhan Chandra Das, M Hadi Amini, and Yanzhao Wu. Security and privacy challenges of large language models: A survey. *ACM Computing Surveys*, 57(6):1–39, 2025.

[19] DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025.

[20] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. *Advances in neural information processing systems*, 36:10088–10115, 2023.

[21] Shizhe Diao, Yu Yang, Yonggan Fu, Xin Dong, Dan Su, Markus Kliegl, Zijia Chen, Peter Belcak, Yoshi Suhara, Hongxu Yin, et al. Climb: Clustering-based iterative data mixture bootstrapping for language model pre-training. *arXiv preprint arXiv:2504.13161*, 2025.

[22] Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi-Min Chan, Weize Chen, et al. Parameter-efficient fine-tuning of large-scale pre-trained language models. *Nature Machine Intelligence*, 5(3):220–235, 2023.

[23] Xin Dong, Yonggan Fu, Shizhe Diao, Wonmin Byeon, Zijia Chen, Ameya Sunil Mahabaleshwarkar, Shih-Yang Liu, Matthijs Van Keirsbilck, Min-Hung Chen, Yoshi Suhara, et al. Hymba: A hybrid-head architecture for small language models. *arXiv preprint arXiv:2411.13676*, 2024.

[24] Amr Elmeleegy et al. Introducing nvidia dynamo, a low-latency distributed inference framework for scaling reasoning ai models, March 2025. NVIDIA Technical Blog.

[25] Henry Evans. Llms vs. slms: Balancing comprehensiveness and smart resource-saving, April 2025. Accessed: 2025-05-09.

[26] Barbara A Ferguson, Timothy A Dreisbach, Catherine G Parks, Gregory M Filip, and Craig L Schmitt. Coarse-scale population structure of pathogenic armillaria species in a mixed-conifer forest in the blue mountains of northeast oregon. *Canadian Journal of Forest Research*, 33(4):612–623, 2003.

[27] Yonggan Fu, Zhongzhi Yu, Junwei Li, Jiayi Qian, Yongan Zhang, Xiangchi Yuan, Dachuan Shi, Roman Yakunin, and Yingyan Celine Lin. Amoeballm: Constructing any-shape large language models for efficient and instant deployment. In *Proceedings of the 38th Annual Conference on Neural Information Processing Systems (NeurIPS 2024)*, 2024.

[28] google. GitHub - google/A2A: An open protocol enabling communication and interoperability between opaque agentic applications.

[29] David Gu, Peter Belcak, and Roger Wattenhofer. Text compression for efficient language generation. *arXiv preprint arXiv:2503.11426*, 2025.

[30] Harrison Clarke. Large language models vs. small language models, March 2024. Accessed: 2025-05-09.

[31] Danny Hernandez, Jared Kaplan, Tom Henighan, and Sam McCandlish. Scaling laws for transfer. *arXiv preprint arXiv:2102.01293*, 2021.

[32] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

[33] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. arxiv 2021. *arXiv preprint arXiv:2106.09685*, 2021.

[34] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022.

[35] Shaohan Huang, Furu Wei, Lei Cui, Xingxing Zhang, and Ming Zhou. Unsupervised fine-tuning for text clustering. In *Proceedings of the 28th international conference on computational linguistics*, pages 5530–5534, 2020.

[36] Invisible Technologies. How small language models can outperform llms, March 2025. Accessed: 2025-05-21.

[37] Mojan Javaheripi and Sébastien Bubeck. Phi-2: The surprising power of small language models, 2023. Microsoft Research Blog.

[38] Andreas Jungherr. Artificial intelligence and democracy: A conceptual framework. *Social media+ society*, 9(3):20563051231186353, 2023.

[39] Aviv Kaufmann. Understanding the total cost of inferencing large language models. Technical report, Enterprise Strategy Group, April 2024. Commissioned by Dell Technologies. Accessed: 2025-05-09.

[40] Sneha Kudugunta, Aditya Kusupati, Tim Dettmers, Kaifeng Chen, Inderjit Dhillon, Yulia Tsvetkov, Hannaneh Hajishirzi, Sham Kakade, Ali Farhadi, Prateek Jain, et al. Matformer: Nested transformer for elastic inference. *arXiv preprint arXiv:2310.07707*, 2023.

[41] Akshi Kumar. From large to small: The rise of small language models (slms) in text analytics. 2025.

[42] Luying Liu, Jianchu Kang, Jing Yu, and Zhongliang Wang. A comparative study on unsupervised feature selection methods for text clustering. In *2005 International Conference on Natural Language Processing and Knowledge Engineering*, pages 597–601. IEEE, 2005.

[43] Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. Dora: Weight-decomposed low-rank adaptation. *arXiv preprint arXiv:2402.09353*, 2024.

[44] Zichang Liu, Jue Wang, Tri Dao, Tianyi Zhou, Binhang Yuan, Zhao Song, Anshumali Shrivastava, Ce Zhang, Yuandong Tian, Christopher Re, et al. Deja vu: Contextual sparsity for efficient llms at inference time. In *International Conference on Machine Learning*, pages 22137–22176. PMLR, 2023.

[45] Nunzio Lore, Sepehr Ilami, and Babak Heydari. Large model strategic thinking, small model efficiency: transferring theory of mind in large language models. *arXiv preprint arXiv:2408.05241*, 2024.

[46] Jeff Loucks, Gillian Crossan, Baris Sarer, China Widener, and Ariane Bucaille. Autonomous generative ai agents: Under development. *Deloitte Insights*, November 2024. Accessed: 2025-05-08.

[47] Zhenyan Lu, Xiang Li, Dongqi Cai, Rongjie Yi, Fangming Liu, Xiwen Zhang, Nicholas D Lane, and Mengwei Xu. Small language models: Survey, measurements, and insights. *arXiv preprint arXiv:2409.15790*, 2024.

[48] Junyu Luo, Weizhi Zhang, Ye Yuan, Yusheng Zhao, Junwei Yang, Yiyang Gu, Bohan Wu, Binqi Chen, Ziyue Qiao, Qingqing Long, et al. Large language model agent: A survey on methodology, applications and challenges. *arXiv preprint arXiv:2503.21460*, 2025.

[49] Georgina M Mace, Paul H Harvey, and Timothy H Clutton-Brock. Brain size and ecology in small mammals. *Journal of Zoology*, 193(3):333–354, 1981.

[50] Tobias Mann. A closer look at dynamo, nvidia's 'operating system' for ai inference, March 2025. Accessed: 2025-05-09.

[51] Market.us. Global agentic ai market size, share analysis by product type, agent role, agent system, end user, region and companies – industry segment outlook, market assessment, competition scenario, trends and forecast 2025–2034, March 2025. Accessed: 2025-05-08.

[52] Tula Masterman, Sandi Besen, Mason Sawtell, and Alex Chao. The landscape of emerging ai agent architectures for reasoning, planning, and tool calling: A survey. *arXiv preprint arXiv:2404.11584*, 2024.

[53] Sourabh Mehta. How much energy do llms consume? unveiling the power behind ai, July 2024. Accessed: 2025-05-21.

[54] Meta Platforms, Inc. Model cards and prompt formats: Llama 3.3, April 2025. Accessed: 2025-05-08.

[55] Metomic. Understanding ai agents & data security, 2025. Accessed: 2025-05-13.

[56] Erik Miehling, Karthikeyan Natesan Ramamurthy, Kush R Varshney, Matthew Riemer, Djallel Bouneffouf, John T Richards, Amit Dhurandhar, Elizabeth M Daly, Michael Hind, Prasanna Sattigeri, et al. Agentic ai needs a systems theory. *arXiv preprint arXiv:2503.00237*, 2025.

[57] Morgan Stanley. Genai revenue growth and profitability, April 2025. Accessed: 2025-05-08.

[58] Humza Naveed, Asad Ullah Khan, Shi Qiu, Muhammad Saqib, Saeed Anwar, Muhammad Usman, Naveed Akhtar, Nick Barnes, and Ajmal Mian. A comprehensive overview of large language models. *arXiv preprint arXiv:2307.06435*, 2023.

[59] NVIDIA. Chatrtx, 2024. NVIDIA AI Product.

[60] NVIDIA. Nvidia dynamo: A datacenter scale distributed inference serving framework. `https://github.com/ai-dynamo/dynamo`, 2025. Accessed: 2025-05-09.

[61] Felipe Maia Polo, Lucas Weber, Leshem Choshen, Yuekai Sun, Gongjun Xu, and Mikhail Yurochkin. tinybenchmarks: evaluating llms with fewer examples. *arXiv preprint arXiv:2402.14992*, 2024.

[62] Lakshmi Radhakrishnan, Gundolf Schenk, Kathleen Muenzen, Boris Oskotsky, Habibeh Ashouri Choshali, Thomas Plunkett, Sharat Israni, and Atul J Butte. A certified de-identification system for all clinical text documents for information extraction at scale. *JAMIA open*, 6(3):ooad045, 2023.

[63] Martin J Rees. *Before the Beginning: Our Universe and Others*. Addison-Wesley, 1997.

[64] Judith Sáinz-Pardo Díaz and Álvaro López García. An open source python library for anonymizing sensitive data. *Scientific data*, 11(1):1289, 2024.

[65] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[66] J William Schopf. Microfossils of the early archean apex chert: New evidence of the antiquity of life. *Science*, 260(5108):640–646, 1993.

[67] Tanya Seda. Cloud llm cost model: Breakdown for mid-market businesses, 2024. Accessed: 2025-05-09.

[68] Olivia Shone. Explore ai models: Key differences between small language models and large language models, November 2024. Accessed: 2025-05-21.

[69] Yixin Song, Zeyu Mi, Haotong Xie, and Haibo Chen. Powerinfer: Fast large language model serving with a consumer-grade gpu. In *Proceedings of the ACM SIGOPS 30th Symposium on Operating Systems Principles*, pages 590–606, 2024.

[70] Shreyas Subramanian, Vikram Elango, and Mecit Gungor. Small language models (slms) can still pack a punch: A survey. *arXiv preprint arXiv:2501.05465*, 2025.

[71] Synergy Technical. Small language models vs. large language models, 2025. Accessed: 2025-05-09.

[72] Brian G. Thamm. Trustworthy and secure ai: How small language models strengthen data security. *Service Contractor Magazine*, October 2024. Accessed: 2025-05-08.

[73] Fali Wang, Zhiwei Zhang, Xianren Zhang, Zongyu Wu, Tzuhao Mo, Qiuhao Lu, Wanjing Wang, Rui Li, Junjie Xu, Xianfeng Tang, et al. A comprehensive survey of small language models in the era of large language models: Techniques, enhancements, applications, collaboration with llms, and trustworthiness. *arXiv preprint arXiv:2411.03350*, 2024.

[74] WorkOS. Build secure ai agents, 2025. Accessed: 2025-05-13.

[75] Zhenliang Xue, Yixin Song, Zeyu Mi, Xinrui Zheng, Yubin Xia, and Haibo Chen. Powerinfer-2: Fast large language model inference on a smartphone. *arXiv preprint arXiv:2406.06282*, 2024.

[76] Biwei Yan, Kun Li, Minghui Xu, Yueyan Dong, Yue Zhang, Zhaochun Ren, and Xiuzhen Cheng. On protecting the data privacy of large language models (llms): A survey. *arXiv preprint arXiv:2403.05156*, 2024.

[77] Fanjia Yan, Huanzhi Mao, Charlie Cheng-Jie Ji, Tianjun Zhang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. Berkeley function calling leaderboard. `https://gorilla.cs.berkeley.edu/blogs/8_berkeley_function_calling_leaderboard.html`, 2024.

[78] Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. Tau-bench: A benchmark for tool-agent-user interaction in real-world domains. *arXiv preprint arXiv:2406.12045*, 2024.

[79] Da Yu, Peter Kairouz, Sewoong Oh, and Zheng Xu. Privacy-preserving instructions for aligning large language models. *arXiv preprint arXiv:2402.13659*, 2024.

[80] Adam Zewe. Like human brains, large language models reason about diverse data in a general way. *MIT News*, February 19 2025. Accessed: 2025-05-09.

[81] Jianguo Zhang, Tian Lan, Ming Zhu, Zuxin Liu, Thai Hoang, Shirley Kokane, Weiran Yao, Juntao Tan, Akshara Prabhakar, Haolin Chen, et al. xlam: A family of large action models to empower ai agent systems. *arXiv preprint arXiv:2409.03215*, 2024.

[82] Kevin Zhang. A deep dive on ai inference startups, 2024. Accessed: 2025-05-09.

[83] Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*, 2023.

[84] Xuezhi Zhou, Nathanael Schärli, Yujie Hou, Jason Wei, Denny Zhou, Quoc V. Le, and Douwe Kiela. Least-to-most prompting enables complex reasoning in small language models. *arXiv preprint arXiv:2205.10625*, 2022.

[85] David Zier and Harry Kim. Introducing nvidia dynamo, a low-latency distributed inference framework for scaling reasoning ai models, March 2025. Accessed: 2025-05-09.

A Definitions

This appendix provides two justifications for the choice of definitions in Section 2.1.

A.1 Pragmatic argument

It is desirable to have a definition of SLMs that meets three key criteria:

- **Timelessness.** The definition should be timeless: It should avoiding dependence on hardware-specific metrics like parameter count or FLOPs, which quickly become obsolete as technology advances—what qualifies as "small" today may be "large" tomorrow.
- **Practicality.** The definition is likely to have much wider generality if it is grounded in practical use, reflecting the real-world goal of deploying SLMs on widely available consumer devices, where they can serve the user in their proximity with low-latency inference.
- **Motivation alignment.** The definition should capture the fundamental motivation that drives the training of SLMs in the first place, which is to enable capable language models that can run on-device or within significantly constrained budgets compared to LLMs.

We find definition **WD1** to possess all three. Definition **WD2** is then phrased to complement the set of all language models.

A.2 Limit argument

To explore the distinction between small and large language models in the context of agentic AI, let us adopt the uncompromising lens of an extremalist, for whom intelligence must be either maximally small or maximally large.

Imagine a super-intelligent system spanning galactic scales, marshaling all available matter to optimize its computations. Such a system, while theoretically capable of addressing profound questions would face insurmountable physical constraints. The speed of light limits communication, with round-trip delays across a galaxy potentially spanning tens of thousands of years [63]. This latency precludes real-time coordination, fragmenting the system into loosely coupled components rather than a unified "mind". At cosmological scales, spanning millions or billions of light-years, communication delays could approach or exceed the universe's age of 13.8 billion years [15]. Such a system, while vast, would be impractical for human-relevant applications, its computations unfolding over eons.

Conversely, consider an infinitely small intelligent system, reduced to the minimal substrate capable of computation. Such a system, akin to the simplest biological organisms, would lack the sensors, effectors, or computational capacity to meaningfully interact with its environment. Its intelligence would be constrained to rudimentary evolution, much like early life forms that emerged 3.5 billion years ago [66]. Yet, even in nature, scale varies dramatically: living organisms range from bacteria (hundreds of nanometers) to blue whales (up to 30 meters), the heaviest ones being limited by heat dissipation due to their high volume-to-surface ratio [26]. At the cosmic scale, all terrestrial life appears microscopic, suggesting that absolute size is less critical than functional adaptability.

Hereby: Humans, often regarded as a pinnacle of intelligence, offer a useful anchor for defining SLMs and LLMs. With a brain-to-body mass ratio surpassed only by small mammals like mice [49], humans balance computational efficiency with practical embodiment. SLMs, by analogy, are systems compact enough to run on personal devices, be trained with modest human interaction, or perform constrained, verifiable tasks. LLMs, in contrast, demand datacenter-scale infrastructure, organization-level training, and extensive validation, reflecting their computational load. The extremalist perspective hints at a profound truth: intelligence is not defined by size alone but by the balance of capability, efficiency, and context. For agentic workflows, SLMs may offer agility and accessibility, while LLMs provide depth at the cost of scale.

It is because of this apparent continuum that, if pressed to provide a *definition* of SLMs, we choose to anchor it in characteristics of a model that can be deployed in a distributed fashion with present-day technology and be interactive enough when engaging with a human to be of utility. Proceeding in such a way, the contemporary instances of the definition will evolve as the technology underpinning these models advances, making the definition sufficiently timeless to be practical.

B LLM-to-SLM Replacement Case Studies

This appendix assesses the potential extent of replacing large language model invocations with small language models in three popular open-source agents: *MetaGPT*, *Open Operator*, and *Cradle*. Each case study examines the use of LLMs, evaluates where SLMs may be viable replacements, and concludes with an estimated percentage of replaceable queries.

B.1 Case study 1: MetaGPT

Name. MetaGPT

License. Apache 2.0

Purpose. MetaGPT is a multi-agent framework designed to emulate a software company. It assigns roles such as Product Manager, Architect, Engineer, and QA Engineer to collaboratively handle tasks including requirement drafting, system design, implementation, and testing.

LLM Invocations.

- *Role-Based Actions.* Each agent role invokes LLMs to fulfill its specialized responsibilities (e.g., coding, documentation).
- *Prompt Templates.* Structured prompts used for consistent outputs.
- *Dynamic Intelligence.* Used for planning, reasoning, and adaptation.
- *Retrieval-Augmented Generation (RAG).* Retrieves relevant documents to enhance generation.

Assessment for SLM Replacement. SLMs would be well-suited for routine code generation and boilerplate tasks, as well as for producing structured responses based on predefined templates. However, they would require further fine-tuning data to reliably perform more complex tasks, such as architectural reasoning and adaptive planning or debugging, which would initially benefit from the broader contextual understanding and the generality of LLMs.

Conclusion. In the case of MetaGPT, we estimate that about 60% of its LLM queries could be reliably handled by appropriately specialized SLMs.

B.2 Case study 2: Open Operator

Name. Open Operator

License. MIT License

Purpose. Open Operator is a workflow automation agent enabling users to define behaviours of agents that can perform tasks like API calls, monitoring, and orchestration using tools and services.

LLM Invocations

- *Natural Language Processing.* Parses user intent.
- *Decision Making.* Guides execution flow.
- *Content Generation.* Writes summaries, reports.

Assessment for SLM Replacement SLMs would be well-suited for tasks such as simple command parsing and routing, as well as generating messages based on predefined templates. They could be meeting their limitations when dealing with more complex tasks that would require multi-step reasoning or the ability to maintain conversation flow and context over time—areas where LLMs would continue to offer significant advantages.

Conclusion. In the case of Open Operator, we estimate that about 40% of its LLM queries could be reliably handled by appropriately specialized SLMs.

B.3 Case study 3: Cradle

Name. Cradle

License. MIT License

Purpose Cradle is designed for General Computer Control (GCC), enabling agents to operate GUI applications via screenshot input and simulated user interaction.

LLM Invocations.

- *Interface Interpretation.* Understands visual context.
- *Task Execution Planning.* Determines sequences of GUI actions.
- *Error Handling.* Diagnoses and reacts to unexpected software states.

Assessment for SLM Replacement SLMs would be well-suited for handling repetitive GUI interaction workflows and the execution of pre-learned click sequences. However, they would face challenges when it comes to tasks involving dynamic GUI adaptation or unstructured error resolution, which would require a higher degree of contextual understanding typically provided by LLMs.

Conclusion In the case of Cradle, we estimate that about 70% of its LLM queries could be reliably handled by appropriately specialized SLMs.