A LIGHTWEIGHT MULTI-EXPERT GENERATIVE LANGUAGE MODEL SYSTEM FOR ENGINEERING INFORMATION AND KNOWLEDGE EXTRACTION

Bogdan Bogachov¹, Yaoyao Fiona Zhao^{1,*}

¹McGill University, Montreal, QC, Canada

ABSTRACT

Despite recent advancements in domain adaptation techniques for large language models, these methods remain computationally intensive, and the resulting models can still exhibit hallucination issues. Most existing adaptation methods do not prioritize reducing the computational resources required for fineprioritize reducing the computational resources required for finetuning and inference of language models. Hallucination issues
have gradually decreased with each new model release. However,
they remain prevalent in engineering contexts, where generating
well-structured text with minimal errors and inconsistencies is
critical. This work introduces a novel approach called the Small
Language Graph (SLG), which is a lightweight adaptation solution designed to address the two key challenges outlined above.
The system is structured in the form of a graph, where each node
represents a lightweight expert—a small language model finetuned on specific and concise texts. The results of this study have
shown that SLG was able to surpass conventional fine-tuning
methods on the Exact Match metric by 3 times. Additionally methods on the Exact Match metric by 3 times. Additionally, the fine-tuning process was 1.7 times faster compared to that of a larger stand-alone language model. These findings introduce a potential for small to medium-sized engineering companies to confidently use generative AI technologies, such as LLMs, without the necessity to invest in expensive computational resources. Also, the graph architecture and the small size of expert nodes offer a possible opportunity for distributed AI systems, thus poten-clusters.

Keywords: Large Language Model, Fine-tuning, Adaptation, Small Language Model, Small Language Graph, Generative ΑI

1. INTRODUCTION

In recent years, Large Language Models (LLMs) have experienced a surge in popularity due to their ability to process and generate extensive amounts of data in response to user-defined queries. Major technology companies have been competing to deliver the most advanced LLMs on the market, resulting in

models equipped with vast amounts of publicly available online knowledge. The most prominent examples of such systems in use are closed-source ChatGPT [1], Gemini [2], and open-source Llama models [3]. These systems can serve as effective assistants in domains grounded in well-established knowledge, where relevant information is readily or easily accessible through opensource data such as mathematics, law, and biology.

On the other hand, LLM systems may sometimes lack the necessary knowledge to answer a user query—particularly when the requested information was not included in the training data. To cope with this difficulty, agents were introduced. In general terms, agents [4-6] act as "helpers" to LLM systems, capable of performing fact-checking, retrieving up-to-date and reliable information from the internet, and mitigating hallucination issues.

However, LLMs and LLM systems with agents struggle in narrow and specific domains such as design and manufacturing. As is widely known, the efficacy of LLMs is directly linked to the volume and quality of data available for training and fine-tuning. The key to producing efficient models is high-quality data [7, 8]. Yet, taking various factors into account, including security [9], data in many design and manufacturing sub-fields is not publicly accessible, leading to challenges in obtaining domain-specific information. It can be argued that publicly available data is sufficient for developing state-of-the-art LLMs, and that transfer learning [10, 11] enables near-optimal data processing and generation capabilities for end users. However, these models are not entirely reliable in specific applications and are prone to hallucinations even when agents are employed due to the inaccessibility of proprietary data.

Now, understanding the necessity of such systems in engineering domains is a critical aspect of this discussion. When properly adapted, these systems have the potential to substantially reduce the man-hours required for routine tasks (such as searching standard repair procedures for aerospace components), thereby freeing up the workforce to focus on more creative and value-added engineering activities. This could significantly increase the productivity of engineering firms. Furthermore, the financial aspect must also be considered. While readily available LLM models or systems could be adapted for engineering appli-

^{*}Corresponding author: yaoyao.zhao@mcgill.ca

cations, most rely on costly cloud computing services or require the deployment of high-end on-premises servers. The majority of small to medium-sized engineering companies will not be able to afford such costly technologies. Therefore, there is a clear need for lightweight LLM adaptation techniques tailored to specific domains, aimed at reducing hallucinations and enhancing their accessibility for engineering applications.

In this research, the problem stated above is addressed by introducing SLG, a system comprised of transformer-based [12] language model experts, which are based on fine-tuned Llama-3.2-1B-Instruct models [13, 14]. The reasoning behind choosing a graph system instead of fine-tuning a stand-alone LLM is due to the hallucination problem, since in any engineering domain, word inaccuracies or ambiguities are highly undesirable. Transformerbased [12] models lack reasoning skills [15] because, while being trained, they simply learn underlying word patterns in training data. Thus, during inference, the word generation process is purely probabilistic. The probabilistic nature of LLMs introduces a high risk of generating words that could not necessarily be related to the question of an engineer. One of the main reasons why this situation could happen is due to training data overlap. This issue, referred to as "knowledge overshadowing" [16], describes how overlapping contexts in the training data can blend together, making it difficult for an LLM to distinguish between identical or similar words with different meanings.

In SLG, the use of relatively small expert models, such as Llama-3.2-1B-Instruct [13], enables small to medium-sized engineering firms to deploy generative AI technologies locally. Additionally, the graph-based nature of SLG enhances text generation accuracy by leveraging expert nodes trained on focused, domain-specific data segments.

The remainder of the paper is structured as follows. Section 2 discusses the related work. Section 3 explains in detail the proposed methodology and the architecture of the SLG system. Experiments are detailed in Section 4. Limitations and future work are introduced in Section 5. Finally, the conclusions and discussion are listed in Section 6.

2. RELATED WORK

This study proposes the following classification of technologies used to tackle the problem of LLM adaptation in engineering domains: prompt engineering, fine-tuning, and Retrieval-Augmented Generation (RAG).

2.1. Prompt engineering

Prompt engineering offers several advantages, including easy access to preferred LLM systems, rapid interaction, swift generation of desired information, and the ability for users to focus on creative tasks rather than the meticulous process of searching for and extracting knowledge. Ready-to-use models are accessible online through platforms, such as OpenAI [1], Gemini [2], etc. These platforms are user-friendly and provide access to their basic models free of charge. Studies conducted on prompt engineering [17–19] as a method to augment human knowledge have shown the usefulness of LLMs to tackle text generation tasks and speed up workflows. Among the advantages of prompt engineering are ease of access to the LLM systems of choice, fast interaction,

quick generation of requested information, and the possibility for users to concentrate on creativity rather than on the scrutinized process of knowledge search and extraction. However, this method has significant drawbacks. LLM systems like ChatGPT [1] are prone to bias and hallucinations [20]. Also, as specified in [17, 18], LLMs are sensitive to the quality of user prompts. Prompt sensitivity leads to high variability in LLM responses to similar questions that are phrased differently. Moreover, LLM systems lack the cognitive ability to truly understand context and rely solely on probability distributions when generating text [15]. Agents [4–6] offer a partial solution to the issues outlined above; however, they cannot address cases where user queries involve knowledge that is not accessible online.

2.2. Fine-tuning

One approach to overcoming the limitation of inaccessible online knowledge is to ingest proprietary or non-public data into a pre-trained LLM. The most commonly known way of ingestion is fine-tuning. From a macro perspective, fine-tuning techniques can be classified into two major approaches: fine-tuning by means of modifying a base pre-trained model and fine-tuning by means of adding new layers or adapters on top of a base pre-trained model while keeping a base model unchanged.

The full fine-tuning method described in [21] proves its efficiency against prompt engineering. The authors used LaMDA-PT [22] as a backbone model. Its fine-tuned variant outperformed the backbone model by equipping it with additional knowledge. However, the study specifies several limitations. The most significant one is the high computational cost induced by updating all 137 billion parameters of the model.

In contrast, a notable example of LLM adaptation through the addition of extra layers atop a backbone model is Hierarchical Domain Adaptation (HDA), as introduced in [23]. HDA [23] leverages a pre-trained model and trains multiple domain-specific adapters, which are attached one at a time on top of the base model depending on a task being performed. Another similar method is Low-Rank Adaptation of LLMs, or LoRA [24]. Similar to HDA [23], LoRA [24] introduces additional layers on top of a frozen backbone model. LoRA employs a bottleneck architecture that substantially reduces the number of trainable parameters, enabling faster training and inference with minimal added latency.

It is worth noting that the above-mentioned reported literature lacks hallucination tests. Since all of the described methods involve fine-tuning LLMs on data from whole domains, knowledge overshadowing [16] mentioned in Section 1 could occur, thus invoking hallucinations.

2.3. RAG systems

One of the most impactful technologies potentially able to solve the hallucination phenomenon in LLMs is RAG. Originally introduced in [25], this method was welcomed by researchers and professionals around the world not only as a way to fight hallucinations but also as a strong option to augment knowledge of any LLM [26, 27].

To achieve this, RAG chunks textual information, converts chunks to dense vectors, and stores them in a vector database.

During inference, relevant chunks in the form of vectors are retrieved. Retrieval is achieved by comparing a vectorized user query with vectors in the vector database created previously. Top-k vectors are then selected to be passed as context to a generator LLM, which composes a response. This approach can significantly enhance the knowledge of an LLM and reduce hallucinations by enabling access to a dynamically updated vector database containing the most current information.

One of the latest developments in RAG was shared in [28]. This research introduces a retrieval method based on questions using atomic units to improve the retrieval step in RAG systems. This approach enhances recall by breaking text chunks into smaller atomic statements and generating synthetic questions to match user queries more accurately. However, an assumption is made that each query has a single answerable chunk. Also, it does not handle multi-hop retrieval and has only been tested on small-scale datasets.

Consequently, it implies that RAG is not a panacea for all deficiencies of LLMs. This methodology struggles with noisy data and is sporadically incapable of providing negative rejection, an ability to refuse answering a question when retrieved documents lack relevant information [29].

2.4. LLMs in engineering

One of the most recent works devoted to adapting LLMs in engineering domains employing prompt engineering [30] introduces a novel method to extract aviation accident causality information. The approach presented in this paper is compared with existing LLM-based information extraction methods and is reported to outperform them by achieving higher accuracy, requiring less annotated data, and handling unstructured text more effectively. However, this method struggles with processing ambiguous texts and requires high computational resources.

LLMs' fine-tuning, presented by [31], showcases a solid method tailored to solve engineering problems. This paper introduces a set of MechBERT models, LLMs based on Bidirectional Encoder Representations for Transformer (BERT). The models were pre-trained on stress—strain scientific literature and further fine-tuned for general English-language question-answering tasks to improve information extraction of mechanical properties. The resultant models outperformed other models in the BERT family while being smaller and faster. However, despite the performance increase in the domain of interest, the models showed limited performance on general-language tasks.

Finally, [32] offers promising insights into using RAG in engineering. This paper proposes a RAG-based tool to extract information from documents encompassing multiple domains. The tool provides a high level of semantic understanding, flexibility in domain adaptation, and integration. Nevertheless, the proposed technique is overly reliant on complex models, and it lacks standardized evaluation metrics.

Motivated by the limitations outlined in the preceding subsections, there is a clear need to develop a method that combines computational efficiency with high accuracy while effectively addressing domain-specific tasks.

3. METHODOLOGY

The methodology used to create SLG is split into two main portions: dataset preparation and the SLG system construction.

3.1. Dataset

Since this work is aimed at finding a lightweight LLM adaptation solution tailored to maximize accuracy while generating engineering data, any text-based engineering document is sufficient as a dataset. In this research, a Structural Repair Manual (SRM) of Cessna aircraft is used [33].

Increasing LLM generation accuracy could involve multiple approaches. One of them is aiming to reduce hallucinations. As it was mentioned earlier, one of the reasons for the hallucination phenomenon is data overshadowing [16]. In an oversimplified way, this phenomenon can be described as data overlapping, as shown in Figure 1.

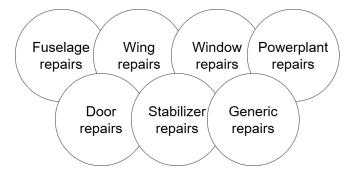


FIGURE 1: DATA OVERLAPPING ILLUSTRATION.

An example of such overlapping could happen when two or more engineering procedures have identical beginnings but different endings, as shown in Table 1.

TABLE 1: EXAMPLE OF DATA OVERLAPPING FROM [33].

	Sentence	
Sentence 1	Damage which would involve a typical skin repair can be described as damage the requires modification, such as material replacement or patching.	
Sentence 2	Damage which would involve a control surface repair: After the repair is completed, the control surface balance must be checked as described in Flight Control Surface Balancing.	

To avoid overlapping, training data chunks were isolated from each other. A schematic example of an ideal training dataset split would look as shown in Figure 2, where each bubble represents a small chunk of the whole training dataset. Each chunk is used to fine-tune only one expert in the SLG system. This way, each expert receives isolated knowledge, thus eliminating data overlap.

To achieve such data division, each training data chunk has to have a logical beginning and a logical ending. Usually, text-based engineering documentation has a well-defined structure split by sections and subsections; Cessna's SRM [33] is not an

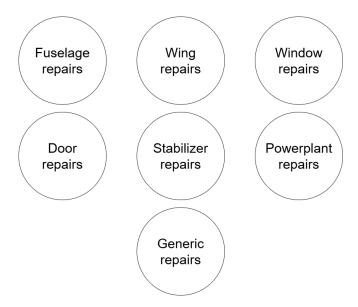


FIGURE 2: SCHEMATIC REPRESENTATION OF ISOLATED TRAINING DATA.

exception. This feature of engineering documentation simplifies the data preparation process. The text is split into chunks by subsections. Subsequently, each chunk is fed into Llama-3.3-70B-Instruct LLM [34], asking it to generate questions for the text. Thus, question-answer pairs are created, which are used for model fine-tuning and testing.

It is important to note that this data chunking method is wellsuited to most engineering documentation due to its structured nature, making SLG applicable across a wide range of engineering domains.

Also, it is essential to highlight that all image data was removed from the dataset due to the text-only focus of this specific research.

3.2. SLG

The methodology used to build the SLG system is based on graphs, as shown in Figure 3. In the flowchart, it is assumed that the user's query is about fuselage repairs. The process follows the green arrows. The query is first directed to the orchestrator, which then queries the fuselage repairs expert. A response is returned to the user, and the process concludes at the end block.

The system is built on the Llama-3.2-1B-Instruct [13] as its backbone LLM, which is fine-tuned using LoRA [24] to serve both as the orchestrator and the expert nodes.

A dataset used for the orchestrator differs from the one used for experts. In both cases, the datasets share identical questions; however, the answers used for expert fine-tuning are actual engineering procedures, while for the orchestrator, the answers are the expert names. The expert names bear the names of the engineering document subsections. This approach allows the orchestrator to directly return the name of an appropriate expert and send the user's query to it. Refer to a question-answer example in Table 2.

To fine-tune experts, the same backbone Llama-3.2-1B-Instruct [13] LLM is used. The model is fine-tuned separately on each isolated dataset described in Subsection 3.1 using LoRA

[24]. The fine-tuned models are then connected using a graph approach utilizing the LangGraph library [35]; thus, each model is represented by a node, and the orchestrator extends edges to each of the experts.

To perform inference within the SLG system, the orchestrator receives a user query, processes it, and routes it to the most relevant expert node for response generation. A chosen expert produces an answer, which is returned to the user.

A detailed evaluation of the proposed method is presented in Section 4, with Table 4 providing a comprehensive list of all hyperparameters used.

4. EXPERIMENTS

This section describes the experimentation setup, followed by the fine-tuning strategy of all tested models.

4.1. Experimentation setup

It is implied by model metrics on different benchmarks that Llama-3.1-8B-Instruct LLM [36] exhibits better performance than Llama-3.2-1B-Instruct LLM [13].

Since SLG is based on a small LLM - Llama-3.2-1B-Instruct [13], to prove the potential of SLG, it is proposed to compare it with fine-tuned Llama-3.1-8B-Instruct LLM [36] and fine-tuned Llama-3.2-1B-Instruct LLM [13]. The core objective of this experimental setup is to demonstrate that the fine-tuned multi-expert SLG system outperforms both a larger stand-alone fine-tuned Llama-3.1-8B-Instruct LLM [36] and a size-matched stand-alone fine-tuned Llama-3.2-1B-Instruct LLM [13].

All models are tested using a test dataset described in Subsection 3.1 by comparing generated answers to ground truth answers.

ROUGE-L, Exact Match (EM), and METEOR are used as evaluation metrics in this research, where ROUGE-L measures the longest common subsequence between the generated and reference texts, EM checks for an exact string match between the prediction and the reference, and METEOR evaluates based on unigram matches while considering synonyms, stemming, and word order.

4.2. Fine-tuning strategy

LoRA [24] is chosen as a fine-tuning technique in this research. The finetuning pipeline and hyperparameters are shared among all models, namely, Llama-3.2-1B-Instruct LLM [13], Llama-3.1-8B-Instruct LLM [36], SLG. This approach allows a fair comparison by fixing all variables.

The experiments in this study are divided into four categories, each focusing on tuning a specific hyperparameter in the following sequence: learning rate, LoRA [24] rank, gradient accumulation, and LoRA [24] alpha.

Table 3 lists all combinations of tuned hyperparameters. The values in bold indicate which hyperparameter is tuned at each specific row. After tuning, the values exhibiting the best performance are fixed and highlighted in green. All other hyperparameters are fixed and listed in Table 4.

For the full fine-tuning pipeline refer to 'finetune.py' [37] module in the SLG repository.

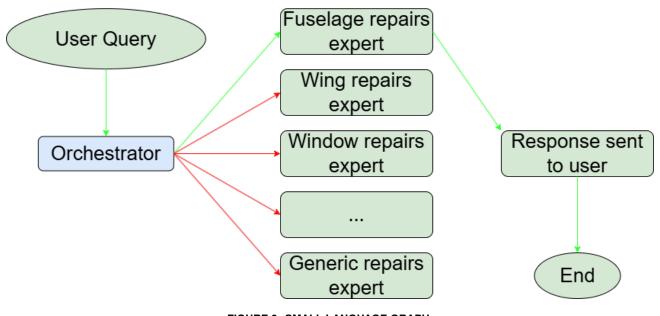


FIGURE 3: SMALL LANGUAGE GRAPH.

TABLE 2: SLG EXPERTS AND ORCHESTRATOR QUESTION-ANSWER PAIRS EXAMPLE [33].

Orchestrator answer	Common question	Expert answer
WING DAMAGE CLASSIFICATION	What are the key factors that determine whether damage to the wing fuel bay spars or ribs can be addressed through repair or requires replacement, considering the criteria outlined for negligible, repairable, and replacement-necessitating damage?	Wing Fuel Bay Spars/Rib Damage Criteria. Negligible damage: Any smooth dents in the wing fuel spar and ribs that have no evidence of tears, cracks, or penetrations – which are not stress wrinkles and do not change (oil can, or pop in and out) with internal pressure – are considered negligible damage

4.3. Results

Overall, the initial experimental results demonstrate the efficiency of the SLG system, built on smaller Llama-3.2-1B-Instruct models [13], outperforming both the stand-alone Llama-3.1-8B-Instruct [36] and the stand-alone Llama-3.2-1B-Instruct [13] models.

Figure 4 illustrates the experimentation evolution. The charts are organized as follows: rows iterate over tuned hyperparameters, while columns iterate over evaluation metrics. Rows one to four showcase comparisons of learning rate, LoRA rank, gradient accumulation, and LoRA alpha against the corresponding metrics. Columns one to three depict comparisons of ROUGE-L, EM, and METEOR across the corresponding hyperparameters.

Table 6 showcases the best experiment results, where R-L, EM, and M stand for ROUGE-L, EM, and METEOR, respectively. While ROUGE-L and METEOR metrics demonstrate similar performance on all compared models, the EM metric indicates that SLG can achieve 3 times better results. Among the three used metrics, EM is the most powerful indication that SLG has the potential to better resist hallucinations by producing text exactly matching the engineering ground truth answers.

In addition, all SLG experts and its orchestrator are trained 1.7 times faster than one stand-alone Llama-3.1-8B-Instruct LLM

[36], as demonstrated in Table 5.

Furthermore, SLG has the potential to exhibit better performance on all three metrics if the orchestrator node is improved. It was discovered that the orchestrator did not always direct user queries to the appropriate expert, thus decreasing the performance of SLG. The success rate of the orchestrator is approximately 70% and is subject to improvement in the future iterations.

Lastly, yet importantly, SLG is able to be fine-tuned and inferred on only one NVIDIA RTX 4090 (24GB VRAM) Graphics Processing Unit (GPU), which makes the system undoubtedly lightweight.

5. LIMITATIONS AND FUTURE WORK

Although SLG demonstrated significant potential in generating engineering texts, it has certain limitations and requires future adjustments.

One notable constraint of this research is its limit to only two models for comparisons, namely, Llama-3.1-8B-Instruct LLM [36] and Llama-3.2-1B-Instruct LLM [13]. It is planned to conduct more extensive comparisons by including the bigger Llama-3.3-70B-Instruct LLM [34] and RAG [25]. As described in Subsection 2.3, RAG is a very powerful technique that enables LLMs to access up-to-date information and augment their contexts before generating text. Llama-3.3-70B-Instruct LLM [34], on the

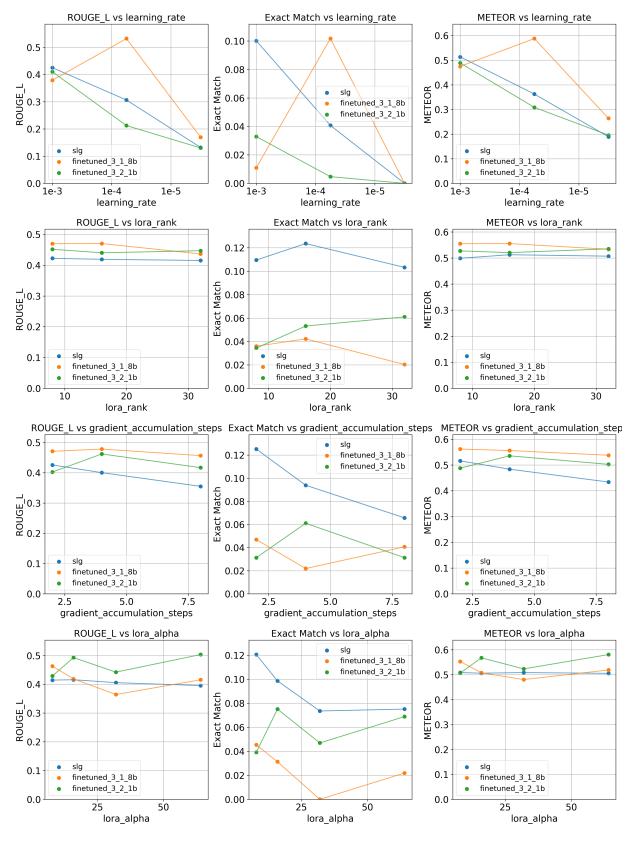


FIGURE 4: EXPERIMENT CHARTS.

TABLE 3: TUNED HYPERPARAMETERS.

Experiment #	Learning rate	LoRA rank	Gradient accumulation	LoRA alpha	
1	1e-5	4	2	8	
2	1e-4	4	2	8	
3	1e-3	4	2	8	
4	1e-3	8	2	8	
5	1e-3	16	2	8	
6	1e-3	32	2	8	
7	1e-3	16	2	8	
8	1e-3	16	4	8	
9	1e-3	16	8	8	
10	1e-3	16	2	8	
11	1e-3	16	2	16	
12	1e-3	16	2	32	
13	1e-3	16	2	64	

TABLE 4: HYPERPARAMETERS USED FOR FINE-TUNING.

Hyperparameter	Value	
LoRA alpha	Refer to Table 3	
LoRA r	Refer to Table 3	
LoRA dropout	0.05	
LoRA task_type	CAUSAL_LM	
learning_rate	Refer to Table 3	
gradient_accumulation_steps	Refer to Table 3	
weight_decay	0.001	
label_smoothing_factor	0.01	
optim	adamw_torch	
num_train_epochs	25 (early stopped)	
early_stopping_patience	3	
eval_strategy	epoch	
save_strategy	epoch	
fp16	True	
per_device_train_batch_size	2	
per_device_eval_batch_size	2	
adam_beta1	0.9	
adam_beta2	0.999	
max_grad_norm	0.5	
warmup_ratio	0.03	
lr_scheduler_type	linear	
load_best_model_at_end	True	
save_total_limit	4	

other hand, demonstrates better results than GPT-40 on most benchmarks [38]; thus, it is a great candidate for comparisons. Also, the experimentation in this research focuses on tuning 4 hyperparameters only, while it is beneficial to extend the experimentation towards other potentially significant hyperparameters, namely, weight decay, learning rate scheduler, warmup ratio, and max gradient norm.

Another shortcoming lies in the limited hallucinations check. This study uses EM as a prevailing metric to showcase the superiority of SLG in resisting hallucinations in comparison to standalone LLMs; however, human evaluation and fact-checking could

TABLE 5: FINE-TUNING TIME COMPARISON.

Model	Average fine-tuning time	
SLG	3475 seconds	
Llama-3.1-8B [36]	5891 seconds	
Llama-3.2-1B [13]	2163 seconds	

TABLE 6: BEST EXPERIMENT METRICS.

Model	R-L	EM	M
SLG	0.41	0.12	0.50
Llama-3.1-8B [36]	0.46	0.05	0.55
Llama-3.2-1B [13]	0.43	0.04	0.51

be a more exhaustive way to estimate how well SLG can avoid hallucinations.

A further limitation involves the absence of images in the training data due to the pure text-based focus of the study. It is an important aspect to consider in future works, since image data is essential in engineering.

It is important to acknowledge that the proposed version of SLG is not a full-scale chatbot, does not have memory, and does not keep conversational context. Each user query is a stand-alone question that does not lead to further communication after receiving an answer from the system. Also, as it was mentioned in Subsection 4.3, the orchestrator node does not always direct user queries to an appropriate expert. This issue could be overcome by converting SLG into a full-scale chatbot system, which would equip a user with the possibility to send clarifying prompts to the system and provide the orchestrator with the necessary information to make a proper decision. Also, an aggregator node could be added to the pipeline to collect text generated by experts into one piece of information in cases when the orchestrator would split a user query among multiple experts. A generic expert node could be a solid addition to the system too, in cases when the orchestrator would not find an appropriate expert at all.

6. CONCLUSIONS AND DISCUSSION

This research proposes a lightweight SLG system tailored for engineering domains to enhance engineers' knowledge and accelerate their workflows. By offloading repetitive tasks, the system enables engineers to focus on more creative and value-driven activities.

SLG employs ultra-small language models as nodes within a graph-based architecture. This design has demonstrated both efficiency and strong potential for mitigating hallucinations in LLMs by constraining each expert node to a narrowly defined knowledge domain. This knowledge isolation strategy minimizes data overlap, thereby reducing the risk of hallucinations. Using EM as the primary evaluation metric, SLG achieved results three times better than those of the larger stand-alone model, Llama-3.1-8B-Instruct.

As it was reported in 4.1 that the Llama-3.1-8B-Instruct LLM outperforms the Llama-3.2-1B-Instruct LLM when used individually. Therefore, the threefold performance improvement achieved by SLG is particularly significant—it demonstrates that a system composed of multiple smaller and individually less capable Llama-3.2-1B-Instruct models can collectively outperform a much larger standalone model. Moreover, despite comprising multiple expert models, SLG achieves 1.7 times faster training than the Llama-3.1-8B-Instruct and requires substantially fewer computational resources, owing to the lightweight nature of its constituent models.

This finding opens the door to building larger, more scalable systems based on the SLG architecture. In particular, it points to the potential of distributed AI systems composed of small language models, such as Llama-3.2-1B-Instruct, where individual users contribute expert nodes running on personal devices like laptops or smartphones. Given that these expert models require minimal computational resources, the network can scale virtually without limit. Such an approach could eventually reduce the reliance on expensive compute clusters, shifting the paradigm toward decentralized AI infrastructure. This vision draws parallels with existing distributed systems, such as peer-to-peer file sharing enabled by the BitTorrent protocol [39].

7. ACKNOWLEDGMENT

This work is funded by McGill Engineering Doctoral Award (MEDA) with additional funding support from Natural Sciences and Engineering Research Council of Canada Discovery Grant RGPIN-2018-05971.

The authors thank Digital Research Alliance of Canada for providing computational resources.

REFERENCES

- [1] OpenAI. "OpenAI." https://openai.com. Accessed: February 13, 2025.
- [2] Google. "Gemini." https://gemini.google.com/app. Accessed: February 13, 2025.
- [3] Touvron, Hugo, Lavril, Thibaut, Izacard, Gautier, Martinet, Xavier, Lachaux, Marie-Anne, Lacroix, Timothée, Rozière, Baptiste, Goyal, Naman, Hambro, Eric, Azhar, Faisal

- et al. "Llama: Open and efficient foundation language models. CoRR, abs/2302.13971, 2023. doi: 10.48550." *arXiv* preprint arXiv.2302.13971 (2023).
- [4] Sheng, Alex. "From Language Models to Practical Self-Improving Computer Agents." *arXiv* (2024)URL 2404. 11964, URL https://arxiv.org/abs/2404.11964.
- [5] Xi, Zhiheng, Chen, Wenxiang, Guo, Xin, He, Wei, Ding, Yiwen and et el. "The Rise and Potential of Large Language Model Based Agents: A Survey." arXiv (2023)URL 2309. 07864, URL https://arxiv.org/abs/2309.07864.
- [6] Yao, Shunyu, Zhao, Jeffrey, Yu, Dian, Du, Nan, Shafran, Izhak, Narasimhan, Karthik and Cao, Yuan. "ReAct Synergizing Reasoning and Acting in Language Models." Conference paper. 2023. International Conference on Learning Representations, ICLR. URL https://www.scopus.com/inward/record.uri?eid=2-s2.0-85199863002&partnerID= 40&md5=7cd72e9c58ecd294bec0f951cfda3ef8. Cited by: 361.
- [7] Luo, Jianxi. "Data-driven innovation: What is it?" *IEEE Transactions on Engineering Management* Vol. 70 No. 2 (2022): pp. 784–790.
- [8] Han, Ji, Forbes, Hannah, Shi, Feng, Hao, Jia and Schaefer, Dirk. "A data-driven approach for creative concept generation and evaluation." *Proceedings of the Design Society: DESIGN Conference*, Vol. 1: pp. 167–176. 2020. Cambridge University Press.
- [9] Behnia, Rouzbeh, Ebrahimi, Mohammadreza Reza, Pacheco, Jason and Padmanabhan, Balaji. "EW-Tune: A Framework for Privately Fine-Tuning Large Language Models with Differential Privacy." 2022 IEEE International Conference on Data Mining Workshops (ICDMW): pp. 560– 566. 2022. IEEE.
- [10] Raffel, Colin, Shazeer, Noam, Roberts, Adam, Lee, Katherine, Narang, Sharan, Matena, Michael, Zhou, Yanqi, Li, Wei and Liu, Peter J. "Exploring the limits of transfer learning with a unified text-to-text transformer." *Journal of machine learning research* Vol. 21 No. 140 (2020): pp. 1–67.
- [11] Radford, Alec, Wu, Jeffrey, Child, Rewon, Luan, David, Amodei, Dario, Sutskever, Ilya et al. "Language models are unsupervised multitask learners." *OpenAI blog* Vol. 1 No. 8 (2019): p. 9.
- [12] Vaswani, Ashish, Shazeer, Noam, Parmar, Niki, Uszkoreit, Jakob, Jones, Llion, Gomez, Aidan N, Kaiser, Ł ukasz and Polosukhin, Illia. "Attention is All you Need." *Proceedings* of the 31st International Conference on Neural Information Processing Systems. 2017.
- [13] Meta. "meta-llama/Llama-3.2-1B-Instruct." https://huggingface.co/meta-llama/Llama-3.2-1B-Instruct. Accessed: February 26, 2025.
- [14] Grattafiori, Aaron, Dubey, Abhimanyu, Jauhri, Abhinav, Pandey, Abhinav, Kadian, Abhishek and et al. "The Llama 3 Herd of Models." *arXiv* (2024)URL 2407.21783, URL https://arxiv.org/abs/2407.21783.
- [15] Zečević, Matej, Willig, Moritz, Dhami, Devendra Singh and Kersting, Kristian. "Causal Parrots: Large Language Models May Talk Causality But Are Not Causal."

- Transactions on Machine Learning Research (2023)URL https://openreview.net/forum?id=tv46tCzs83.
- [16] Zhang, Yuji, Li, Sha, Liu, Jiateng, Yu, Pengfei, Fung, Yi R., Li, Jing, Li, Manling and Ji, Heng. "Knowledge Overshadowing Causes Amalgamated Hallucination in Large Language Models." (2024). URL 2407.08039, URL https://arxiv.org/abs/2407.08039.
- [17] Ma, Kevin, Grandi, Daniele, McComb, Christopher and Goucher-Lambert, Kosa. "Conceptual Design Generation Using Large Language Models." Vol. 6. 2023. URL https://www.scopus.com/inward/record.uri?eid=2-s2.0-85178515903&doi=10.1115%2fdetc2023-116838&partnerID=40&md5=217f1f20efe1f68b5e3fb387cc5d7da1.
- [18] Bouschery, Sebastian G., Blazevic, Vera and Piller, Frank T. "Augmenting human innovation teams with artificial intelligence: Exploring transformer-based language models." *Journal of Product Innovation Management* Vol. 40 No. 2 (2023): p. 139 153. URL https://www.scopus.com/inward/record.uri?eid=2-s2.0-85146930847&doi=10.1111%2fjpim.12656&partnerID=40&md5=4733282820d49cfc3fc26edae328f859.
- [19] Korzynski, Pawel, Mazurek, Grzegorz, Krzypkowska, Pamela and Kurasinski, Artur. "Artificial intelligence prompt engineering as a new digital competence: Analysis of generative AI technologies such as ChatGPT." *Entrepreneurial Business and Economics Review* Vol. 11 No. 3 (2023): p. 25 37. URL https://www.scopus.com/inward/record.uri?eid=2-s2.0-85175052409&doi=10.15678%2fEBER.2023.110302&partnerID=40&md5=8de874dbb66448a57ec1c9a540e46b25.
- [20] Bang, Yejin, Lee, Nayeon, Dai, Wenliang, Su, Dan, Wilie, Bryan, Lovenia, Holy, Ji, Ziwei, Yu, Tiehzheng, Chung, Willy, Do, Quyet, Yan, Xu and Fung, Pascale. "A Multitask, Multilingual, Multimodal Evaluation of ChatGPT on Reasoning, Hallucination, and Interactivity." (2023). DOI 10.48550/arXiv.2302.04023.
- [21] Wei, Jason, Bosma, Maarten, Zhao, Vincent Y., Guu, Kelvin, Yu, Adams Wei, Lester, Brian, Du, Nan, Dai, Andrew M. and Le, Quoc V. "Finetuned Language Models Are Zero-Shot Learners." *ICLR 2022 10th International Conference on Learning Representations* (2022)URL https://www.scopus.com/inward/record.uri?eid=2-s2.0-85150358531&partnerID=40&md5=ca65257522f14c86b051226262d53989.
- [22] Thoppilan, Romal, Freitas, Daniel De, Hall, Jamie, Shazeer, Noam M., Kulshreshtha, Apoorv and et al. "LaMDA: Language Models for Dialog Applications." *ArXiv* Vol. abs/2201.08239 (2022). URL https://api.semanticscholar.org/CorpusID:246063428.
- [23] Chronopoulou, Alexandra, Peters, Matthew E. and Dodge, Jesse. "Efficient Hierarchical Domain Adaptation for Pretrained Language Models." NAACL 2022 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference (2022):
 p. 1336 1351URL https://www.scopus.com/inward/

- record.uri?eid=2-s2.0-85136099333&partnerID=40&md5=90fafe1805750ba01dcefee7eb0d926d.
- [24] Hu, Edward, Shen, Yelong, Wallis, Phillip, Allen-Zhu, Zeyuan, Li, Yuanzhi, Wang, Shean, Wang, Lu and Chen, Weizhu. "LoRA: Low-Rank Adaptation of Large Language Models." *ICLR* 2022 10th International Conference on Learning Representations (2022)URL https://www.scopus.com/inward/record.uri?eid=2-s2.0-85150379710&partnerID=40&md5=1247c3667a6304c3edabb3e2c0c3094b.
- [25] Lewis, Patrick, Perez, Ethan, Piktus, Aleksandra, Petroni, Fabio, Karpukhin, Vladimir, Goyal, Naman, Küttler, Heinrich, Lewis, Mike, Yih, Wen-Tau, Rocktäschel, Tim, Riedel, Sebastian and Kiela, Douwe. "Retrieval-augmented generation for knowledge-intensive NLP tasks." *Advances in Neural Information Processing Systems* Vol. 2020-December (2020). URL https://www.scopus.com/inward/record.uri?eid=2-s2.0-85108449607&partnerID=40&md5=c6711ab215f5fdba9b0d4a8449d7a25a.
- [26] Li, Siran, Stenzel, Linus, Eickhoff, Carsten and Ali Bahrainian, Seyed. "Enhancing Retrieval-Augmented Generation: A Study of Best Practices." Vol. Part F206484-1: p. 6705 6717. 2025. URL https://www.scopus.com/inward/record.uri?eid=2-s2.0-85218493737&partnerID=40&md5=b24733a11a304cf916d4324d98d5e1b2.
- [27] Liu, Siru, McCoy, Allison B and Wright, Adam. "Improving large language model applications in biomedicine with retrieval-augmented generation: a systematic review, meta-analysis, and clinical development guidelines." *Journal of the American Medical Informatics Association* (2025): p. ocaf008DOI 10.1093/jamia/ocaf008. URL https://academic.oup.com/jamia/advance-article-pdf/doi/10.1093/jamia/ocaf008/61442713/ocaf008.pdf, URL https://doi.org/10.1093/jamia/ocaf008.
- [28] Raina, Vatsal and Gales, Mark. "Question-Based Retrieval using Atomic Units for Enterprise RAG.": p. 219 233. 2024. URL https://www.scopus.com/inward/record.uri?eid=2-s2.0-85216930459&partnerID=40&md5=0fba8949ac025ab64333e07b555615d5.
- [29] Chen, Jiawei, Lin, Hongyu, Han, Xianpei and Sun, Le. "Benchmarking Large Language Models in Retrieval-Augmented Generation." Vol. 38. 16: p. 17754 17762. 2024. URL https://www.scopus.com/inward/record.uri?eid=2-s2.0-85189613527&doi=10.1609%2faaai.v38i16.29728&partnerID=40&md5=6f78ac42d80af63bf434a065136db443.
- [30] Chen, Lu, Xu, Jihui, Wu, Tianyu and Liu, Jie. "Information Extraction of Aviation Accident Causation Knowledge Graph: An LLM-Based Approach." *Electronics (Switzerland)* Vol. 13 No. 19 (2024). DOI 10.3390/electronics13193936. URL https://www.scopus.com/inward/record.uri?eid=2-s2.0-85208786490&doi=10.3390%2felectronics13193936&partnerID=40&md5=aedfad1f20d73c13339d249d0d1e0723. Cited by: 1; All Open Access, Gold Open Access.
- [31] Kumar, Pankaj, Kabra, Saurabh and Cole, Jacqueline M. "MechBERT: Language Models for Extracting Chemical

- and Property Relationships about Mechanical Stress and Strain." *Journal of Chemical Information and Modeling* Vol. 65 No. 4 (2025): p. 1873 1888. DOI 10.1021/acs.jcim.4c00857. URL https://www.scopus.com/inward/record.uri?eid=2-s2.0-85216721913&doi=10.1021%2facs.jcim.4c00857&partnerID=40&md5=a2bec78c0a77062ffb5b716a9f76124a. Cited by: 0; All Open Access, Hybrid Gold Open Access.
- [32] Joshi, Raghav, Bubna, Yash, Sahana, M. and Shruthiba, A. "An Approach to Intelligent Information Extraction and Utilization from Diverse Documents." Conference paper. 2024. DOI 10.1109/CSITSS64042.2024.10816908. URL https://www.scopus.com/inward/record.uri?eid=2-s2.0-85216926987&doi=10.1109% 2fCSITSS64042.2024.10816908&partnerID=40&md5= 80eec4445ca534540378b17544e3014d. Cited by: 0.
- [33] COMPANY, CESSNA AIRCRAFT. "Single Engine Models 172, 182, T182, 206 AND T206 1996 And On." http://www.aeroelectric.com/Reference_Docs/Cessna/cessna-maintenance-manuals/CessnaSingle_1996on_

- structural_repair_MM_SESR04.pdf. Accessed: August 04, 2024.
- [34] Meta. "meta-llama/Llama-3.3-70B-Instruct." https://huggingface.co/meta-llama/Llama-3.3-70B-Instruct. Accessed: February 26, 2025.
- [35] LangChain. "LangGraph." https://www.langchain.com/langgraph. Accessed: March 07, 2025.
- [36] Meta. "meta-llama/Llama-3.1-8B-Instruct." https://huggingface.co/meta-llama/Llama-3.1-8B-Instruct. Accessed: February 26, 2025.
- [37] Bogachov, Bogdan. "fine-tune: full fine-tuning pipeline." (2025). URL https://github.com/bogdanbogachov/eng_llm/blob/main/finetune/finetune.py. Accessed: March 20, 2025.
- [38] Meta. "Llama." https://www.llama.com/. Accessed: March 21, 2025.
- [39] Pouwelse, J.A., Garbacki, Pawel, Epema, D. and Sips, Henk. "The Bittorrent P2P File-Sharing System: Measurements and Analysis." Vol. 3640: pp. 205–216. 2005. DOI 10.1007/11558989_19.