OrchestraLLM: Efficient Orchestration of Language Models for Dialogue State Tracking

Anonymous ACL submission

Abstract

Large language models (LLMs) have revolutionized the landscape of Natural Language Processing, but are computationally expensive. To reduce the cost without sacrificing performance, previous studies have explored various approaches to harness the potential of Small Language Models (SLMs) as cost-effective alternatives to their larger counterparts. Driven by findings that SLMs and LLMs exhibit complementary strengths in a structured knowledge extraction task, this work presents a novel SLM/LLM routing framework designed to improve computational efficiency and enhance task performance. In dialogue state tracking tasks, the proposed routing framework enhances performance substantially compared to relying solely on LLMs, while reducing the computational costs by over 50%.

1 Introduction

Large Language Models (LLMs) have become versatile tools capable of tackling a wide range of tasks with only a few training examples. However, their expanding sizes have brought escalating computational demands. In contrast, more efficient Small Language Models (SLMs) often require a substantial amount of fine-tuning data to become truly effective. This work addresses scenarios where only limited task-specific data is available, making fine-tuned SLMs less dependable. Our objective is to develop a routing framework that orchestrates SLMs and LLMs, enhancing task performance while reducing computational costs.

Task-oriented dialogue is crucial for efficient human-computer interaction, enabling systems to understand and assist with specific tasks like booking flights or scheduling meetings. Task-oriented dialogues involving structured data typically rely on Dialogue State Tracking (DST), where user intent is extracted from the dialogue history between a user and the agent in the form of slot values associated with a predefined schema. Fine-tuned SLMs have been used in DST for a few years, including both autoregressive LMs (Ham et al., 2020; Hosseini-Asl et al., 2020; Peng et al., 2020) and sequence-to-sequence LMs (Lee et al., 2021; Su et al., 2022; Bang et al., 2023; Imrattanatrai and Fukuda, 2023; Wang et al., 2023). LLMs have been used for few-shot in-context learning in DST (Xie et al., 2022; Hudeček and Dušek, 2023; Hu et al., 2022; King and Flanigan, 2023a) where LLMs are prompted with human-authored task descriptions or in-context exemplars. In our work, we seek to take advantage of the effectiveness of LLMs with a small amount of training data but reduce the cost.

Strategies that leverage both SLMs and LLMs have been developed to mitigate the computational demands of LLMs. Cascade-based approaches direct a query to an LLM when it cannot be resolved by an SLM (Chen et al., 2023; Madaan et al., 2023). These approaches introduce latency and computational redundancy since they consistently query SLMs. Other approaches use binary classifiers to predict the most appropriate LM to utilize (Kag et al., 2022; Šakota et al., 2023). A limitation of the classifier-based approaches is the necessity for retraining when introducing new models.

In this work, we propose a dynamic routing framework, **`OrchestraLLM`** (illustrated in Figure 1), that leverages small (fine-tuned) and large LM experts. Hypothesizing that examples with similar semantic embeddings are of the same difficulty level, we select an appropriate expert based on embedding distances between the testing instance and instances in expert pools. The expert pools contain examples representing the types of

Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), pages 1434–1445

dialogue contexts where the different LMs provide more reliable answers. After retrieving the top k nearest examples, an expert is selected based on the majority vote. Unlike cascade-based and classifier-based approaches, the proposed framework eliminates the need for router training, though hand-labeled data is needed for creating the expert pools. In addition, the retriever can be fine-tuned with target task labels or expert information to achieve more efficient and accurate routing.

In summary, the key contribution of this work is the introduction of a novel switching model designed to reduce the computational costs associated with LLMs while simultaneously enhancing performance. Experimental results on two different multi-domain DST benchmarks (MultiWOZ (Budzianowski et al., 2018; Ye et al., 2022) and SGD (Rastogi et al., 2020)) demonstrate that `OrchestraLLM` capitalizes on the proficiencies of different experts, outperforming LLM systems while also achieving a substantial reduction of over 50% in computational costs.

2 Dialogue State Tracking

In this work, we focus on combining general-purpose LLMs and task-specific SLMs to achieve better efficiency for dialogue state tracking (DST). In the following, we first provide the necessary task setups and then detail the two representative DST models using LLMs and SLMs respectively.

A task-oriented dialogue (TOD) consists of a sequence of exchanges between two parties, each of which is initialized by the user and followed by a response from the system. Here, we denote each exchange as a turn leading to a sequence, $U_1, A_1, ..., U_T, A_T$, where U_t and A_t represent the user utterance and the system response, respectively. For the t-th turn, the user provides a new utterance U_t, and the system agent responds with utterance A_t. At turn t, the corresponding dialogue context is $C_t = \{U_1, A_1, \ldots, A_{t-1}, U_t\}$, which excludes the latest system response A_t. The goal of DST is to extract task-relevant information as structured representations (dialogue states) from user-system utterances so that the user requests can be fulfilled accordingly. To facilitate this, there is typically a task-specific schema provided. In a multi-domain scenario considered in this paper, the schema contains M domains $\mathcal{D} = \{d_1, \ldots, d_M\}$ and N slots $\mathcal{S} = \{s_1, \ldots, s_N\}$ to track. DST_t, the dialogue state at turn t, defines the current mappings from pairs of (d_m, s_n) into a value v based on dialogue context C_t. Specifically,

$$DST_t = \{(d_m, s_n, v^t_{mn}) | v^t_{ij} \neq \texttt{null}\},$$

only containing the non-null slots accumulated so far. Instead of directly predicting the entire dialogue state from scratch, we build dialogue state predictions based on the turn-level belief (TLB) as done by Hu et al. (2022), which allows a more flexible combination of LLMs and SLMs. At turn t, the DST model only predicts TLB_t, where new expressed slots or slots with updated values are used to get the latest DST_t via aggregating all previous TLBs.[1]

In the literature, task-specific SLM-based DST models are typically fine-tuned with full-parameter updates while DST models using LLMs are realized via few-shot in-context learning. We discuss the two different DST models considered below.

LLM DST. *IC-DST* (Hu et al., 2022) is an in-context learning (ICL) framework that enables few-shot DST with LLMs. The prediction is the change in each turn pair instead of the accumulated dialogue states. To obtain the accumulated dialogue states, the turn changes are aggregated across turns. Dialogue states of previous turns are used as a summary of the context history and it allows to fit in more exemplars which is crucial for ICL performance. Concretely, given the schema table, K in-context exemplars, dialogue states of the previous turn, and the input instance (most recent agent-user utterance pair), the LLM outputs

$$TLB_t = \text{LLM}(T, E_{1:K}, DST_{t-1}, A_{t-1}, U_t) \quad (1)$$

where T is the schema table for all domains, E_k are examples of pairs of turn changes and associated outputs.

SLM DST. Here, we develop a prompt-based DST model (denoted as *Prompt-DST*) with SLM (T5 (Raffel et al., 2020)). The input of Prompt-DST is similar to IC-DST, except that the in-context exemplars are excluded. Specifically, given the

[1]We replace previous values with updated ones for slots present in prior TLBs.

schema prompt-augmented input, the model outputs

$$TLB_t = \text{SLM}(T, DST_{t-1}, A_{t-1}, U_t). \quad (2)$$

Here, the model is trained using the learning objective by maximizing the log-likelihood of slot values $v_t(d_m, s_n)$ for the current TLB, *i.e.,*

$$\max \log P(TLB_t|T, DST_{t-1}, A_{t-1}, U_t). \quad (3)$$

During inference, a greedy decoding procedure is directly applied, *i.e.,* only the most likely token in the given model vocabulary is predicted at each decoding step.

3 Routing Approach

Here, we present our approach for routing with `OrchestraLLM` applied to the DST task. The overall framework is illustrated in Figure 1. We denote different DST models as **experts**. Given a new input instance (the triplet $(DST_{t-1}, A_{t-1}, U_t)$), `OrchestraLLM` first computes its semantic embedding, compares it with exemplar embeddings of triplets from each **expert pool** using a cosine distance, and retrieves the top-K exemplars. The router assigns the input to an expert based on majority vote. While our approach draws inspiration from the work of Jang et al. (2023), it is important to note that their approach primarily focuses on optimizing task performance in zero-shot task transfer scenarios, whereas our emphasis lies in improving computational efficiency within the few-shot learning settings.

3.1 Expert Pool Construction

For each dialogue in a small held-out set, the SLM and LLM experts are used to predict the TLB at each user turn (TLB_t) individually. If both experts correctly predict the TLB, the instance triplet is included in the SLM pool. When only one expert correctly predicts the TLB, the instance is assigned to that expert's pool. Instances that are not correctly predicted are not used in either pool.

3.2 Triplet Representation Learning

Similar to recent work on dense retrieval (Karpukhin et al., 2020), the retriever uses a bi-encoder architecture, which encodes dialogues with labels and predictions into embedding space. Throughout the work, SenBERT (Reimers and Gurevych, 2019) is used as the backbone embedding model. The bi-encoder is fine-tuned using a small set of dialogues, the same as that used to construct the expert pools. We use a contrastive loss such that the similarity between a positive example pair is high and the similarity between a negative example pair is low. Three different methods for constructing positive and negative pairs are explored: task-aware, expert-aware, and their combination.

Task-Aware Supervision identifies positive and negative instance pairs for training by first computing pairwise similarity for each sample in the hold-out set. Then, the l highest and lowest scoring pairs are used as positive and negative examples, respectively. The similarity function leverages the gold annotations of the hold-out set dialogues. Given two instances, a and b, the similarity is a weighted combination of the slot-value similarity of the previous state (DST) and the current TLB:

$$\frac{1}{2} Sim_{DST} + Sim_{TLB}.$$

Let $TLB_x = \{(s_1^x, v_1^x), \cdots, (s_m^x, v_m^x)\}$ be the TLB of instance x. Following Hu et al. (2022), the slot-value pair similarity is

$$F_{\textit{slot-value}} = F(\{(s_1^a, v_1^a), \cdots, (s_m^a, v_m^a)\}, \{(s_1^b, v_1^b), \cdots, (s_n^b, v_n^b)\}).$$

and the slot similarity is

$$F_{slot} = F(\{s_1^a, \cdots, s_m^a\}, \{s_1^b, \cdots, s_n^b\}).$$

where F is the standard definition of F1 score *i.e.,* $F = \frac{2PR}{P+R}$, in which P is precision, and R is recall. The similarity score between TLB_a and TLB_b is

$$Sim(TLB_a, TLB_b) = F_{\textit{slot-value}} + F_{slot} - 1$$

The context history similarity Sim_{DST} is defined in the same way.

Expert-Aware Supervision first groups instances in the hold-out set according to which expert gave the most accurate prediction. (For ties, the SLM is chosen.) We then compute pairwise triplet similarities using an off-the-shelf embedder (*e.g.,* SenBERT). The l highest scoring pairs with the same expert label are positive examples, and

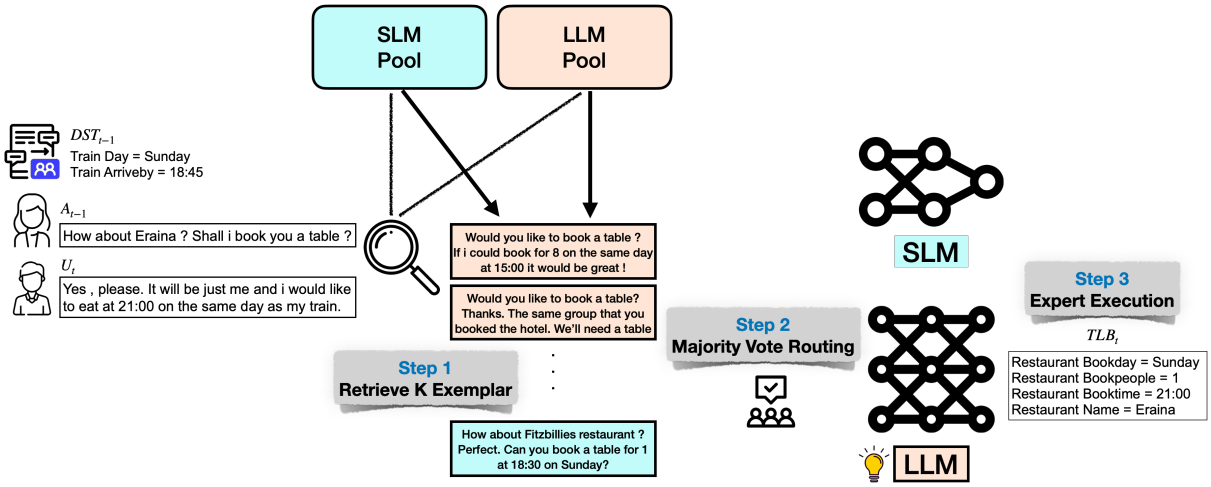

Figure 1: Illustration of OrchestraLLM. LMs are orchestrated by a retrieval-based dynamic router. During inference, the testing instance queries the expert pools to retrieve top k similar examples. Subsequently, a LM expert is selected based on the majority vote.

Dataset	MultiWOZ	SGD
# Domains	8	41
# Dialogues	8438	16142
# Total Turns	113556	329964
Avg. Turns per Dial.	13.46	20.44
Avg. Toks per Turn	13.13	9.75
# Slots	24	214
# Slot Values	4510	14139

Table 1: Experiment data summary. The numbers are computed on training splits of the datasets.

the l lowest scoring pairs with different expert label are negative examples.

Task+Expert-Aware Supervision simply pools both sets of positive and negative pairs.

Note that task-aware supervision is agnostic to what experts are used in routing, so the embedding model need not be retrained as experts are added or updated. Expert-aware supervision will require updating the embedding model if the experts change. In all cases, the expert pools will need to be updated with changes to the experts.

4 Experiments

4.1 Datasets

We use two datasets detailed below for experiments. A summary of DST datasets is reported in Table 1.

MultiWOZ (Budzianowski et al., 2018) is a multi-domain task-oriented dialogue dataset that contains over 10K human-human written dialogues across 8 domains and has been one of the most popular benchmarks in the DST literature. After the publication of Budzianowski et al. (2018), many works improve the label qualities, *e.g.,* MultiWOZ 2.1 (Eric et al., 2020) and MultiWOZ 2.4 (Ye et al., 2021). We experiment using the most recent version, MultiWOZ 2.4.

SGD (Rastogi et al., 2020) is a task-oriented dialogue dataset that contains over 16k multi-domain conversations spanning 41 domains, featuring out-of-domain evaluation. 15 out of 21 domains in the test set are not present in the training set and 77% of the dialogue turns in the test set contain at least one domain not present in the training set.

4.2 Experimental Setting

In this work, we consider a **few-shot** set up for DST. Following the multi-domain experiment setting from Wu et al. (2020), we randomly sample 5% of training data from MultiWOZ and SGD respectively for training the expert models.

Model and Hyperparameter Setting. For Prompt-DST, we use T5-base and T5-large as the backbone model for MWOZ and SGD respectively, as the latter is more complex in terms of schema and more dialogue turns. For IC-DST, we use ChatGPT as the backbone model[2] with 10 in-context exemplars. We initialize the routing retriever from

[2]Accessed: August–October 2023, Version: gpt-3.5-turbo-0301.

SenBERT (all-mpnet-base-v2). We run inference on 100 dialogues randomly sampled from validation sets of MWOZ and SGD as the held-out sets. The same 100 dialogues are used to train the retriever. For all experiments, $l = 25$ is used for the positive and negative examples for contrastive learning. During inference, we randomly sample 100 turns from the held-out sets to serve as SLM pool and LLM pool respectively for MWOZ experiments and 300 turns for SGD experiments. We use $k = 10$ for the majority vote and break the tie by favoring SLM.

4.3 Evaluation

4.3.1 Accuracy

Conventionally, DST systems are evaluated by joint goal accuracy (JGA) on accumulated dialogue states (Henderson et al., 2014). This metric assesses the correctness of the dialogue state at each turn and deems it accurate only if all slot values within every domain precisely match the ground-truth values. It is difficult to accurately assess how well a system performs on single turns with DST JGA. Therefore we also report turn-level belief (TLB JGA) (Dey et al., 2022).

4.3.2 Efficiency

Floating-point operations per Second (FLOPs) represent the number of floating-point arithmetic operations (additions and multiplications) a model performs in one pass. FLOPs are often used to estimate the computational cost or workload required for training or inference. Training a large model requires a significant number of backward passes, which are more expensive than forward passes, yet inference is a continuous process that happens whenever the model is in use, thus accruing more cost over time. NVIDIA (Leopold, 2019) and Amazon (Barr, 2019) report around 90% of the ML workload is inference processing in their cloud services. Therefore, we choose to report FLOPs for inference time usage.

We estimate the aggregate computational cost, measured in TeraFLOPs, required for performing inference across the entire testing dataset. It is important to note that IC-DST relies on ChatGPT, a model that is not publicly accessible, thus precluding a direct evaluation of its computational efficiency. Based on prevailing conjecture within the public domain, ChatGPT is presumed to be a fine-tuned iteration of the GPT-3 model with a substantial parameter count of 175 billion (Brown et al., 2020). To estimate the computational requirements, we conduct FLOPs measurements on the GPT-2 (Radford et al.) model and subsequently scale these measurements in accordance with the parameter size differential between GPT-2 and ChatGPT. The computational cost of the retriever, measured in FLOPs, for each turn instance, is approximately 0.02 TeraFLOPs. This computational load becomes negligible when considered in conjunction with ChatGPT in the `OrchestraLLM`. ChatGPT requires approximately 3000 TeraFLOPs for each turn instance.

4.4 Baselines

Classification-Based Routing

We compare our routing framework with existing classification-based approaches to model switching, such as those proposed by Šakota et al. (2023) and Kag et al. (2022). These existing approaches typically train a binary classifier to serve as the router. We train BERT (Devlin et al., 2019) (bert-base-cased) with the expert labels in the hold-out set of dialogues with a binary classification objective to do routing as a baseline.

Cascade-Based Routing

Cascade-based approaches Chen et al. (2023); Madaan et al. (2023) typically query a SLM and redirect the instance to a LLM if the small language model is not confident enough. We choose to utilize the normalized sequence level probability of SLM output as the confidence measure. We tune the probability threshold on the hold-out-set and use the threshold to determine whether to redirect the instance to LLM during inference.

4.5 Results

4.5.1 MultiWOZ

We demonstrate the MultiWOZ experiments in a few-shot setting in Table 2. We use 5% of dialogues in the training set for finetuning Prompt-DST and retriever of IC-DST. Prompt-DST and IC-DST perform inference on another 100 dialogues from the validation set, documenting the turns each expert specializes in. We randomly select 100 turns from these dialogues for each expert to serve as expert pools for dynamic routing.

Models	Router	Assignment Ratio	TeraFLOPs	TLB JGA	DST JGA
DST Baselines					
Prompt-DST	N/A	N/A	272	73.43	46.06
IC-DST (Hu et al., 2022)	N/A	N/A	22 M	78.21	49.68
DS2 - T5 (Shin et al., 2022)∗	N/A	N/A	N/A	N/A	49.89
Routing Baselines					
Prompt-DST & IC-DST	Oracle	73% Prompt-DST	5.94 M	88.07	65.39
Prompt-DST & IC-DST	Classification-Based	91% Prompt-DST	1.98 M	77.60	47.58
Prompt-DST & IC-DST	Cascade-Based	13 % Prompt-DST	19.14 M	80.40	51.46
Our Retrieval-Based Routing DST					
`OrchestraLLM`	SenBERT	60% Prompt-DST	8.8 M	80.74	50.19
`OrchestraLLM`	Task-Aware	55% Prompt-DST	9.9 M	82.43	52.53
`OrchestraLLM`	Expert-Aware	78% Prompt-DST	4.8 M	81.02	50.65
`OrchestraLLM`	Task+Expert-Aware	62% Prompt-DST	8.3 M	**82.46**	**52.68**

Table 2: Results on MultiWOZ 2.4. The TeraFLOPs are computed on inference passes on the entire testing set. We report the percentage of turns routed to Prompt-DST in the assignment ratio column. ∗ marks numbers reported in Hu et al. (2022).

Models	Router	Assignment Ratio	TeraFLOPs	TLB JGA	DST JGA
DST Baselines					
Prompt-DST	N/A	N/A	8882	62.21	28.38
IC-DST (Hu et al., 2022)	N/A	N/A	121 M	63.86	33.15
Routing Baselines					
Prompt-DST & IC-DST	Oracle	62% Prompt-DST	45.98 M	77.48	47.50
Prompt-DST & IC-DST	Classification-Based	38% Prompt-DST	75.02 M	66.94	31.86
Prompt-DST & IC-DST	Cascade-Based	7.9% Prompt-DST	111.34 M	64.17	32.75
Our Retrieval-Based Routing DST					
`OrchestraLLM`	SenBERT	50% Prompt-DST	60.50 M	65.97	32.75
`OrchestraLLM`	Task-Aware	55% Prompt-DST	54.45 M	67.25	32.78
`OrchestraLLM`	Expert-Aware	54% Prompt-DST	55.66 M	67.34	32.95
`OrchestraLLM`	Task+Expert-Aware	57% Prompt-DST	52.03 M	**68.09**	**33.07**

Table 3: Results on SGD. The TeraFLOPs are computed on inference passes on the entire testing set. We report the percentage of turns routed to Prompt-DST in the assignment ratio column.

As expected, IC-DST outperforms Prompt-DST in the few-shot setting, indicating that the LLM is more generalizable than the fine-tuned SLM. The BERT-based classification router struggles to effectively harness the capabilities of both models. To establish an upper performance bound for the learned router, we introduce the oracle router, which aggregates predictions from both LLM and SLM when either model is correct, with a preference for SLM whenever available. Even with a vanilla SenBERT as a retriever, `OrchestraLLM` outperforms IC-DST while saving 60% calls to LLM, demonstrating the effectiveness of our proposed framework. Further finetuning the retriever with the proposed task-aware contrastive examples routes examples more effectively and improves DST JGA around 3% compared to IC-DST. With additional expert-aware training of the retriever, we can further save around 7% traffic to LLM with superior performance compared with IC-DST. In spite of its compact size, Prompt-DST is finetuned to align with specific in-domain knowledge and task-specific artifacts (*e.g.,* schema constraints and customized labeling strategies). Conversely, IC-DST is enriched with an extensive repertoire of knowledge acquired during the pretraining phase of LLM, endowing it with contextual reasoning capabilities and an enhanced grasp of common-sense knowledge (Section 5.2). Since these two models are complementary, an effective integration can surpass the performance of the IC-DST model.

4.5.2 SGD

To evaluate our system under out-of-domain scenarios, we show experimental results in a few-shot

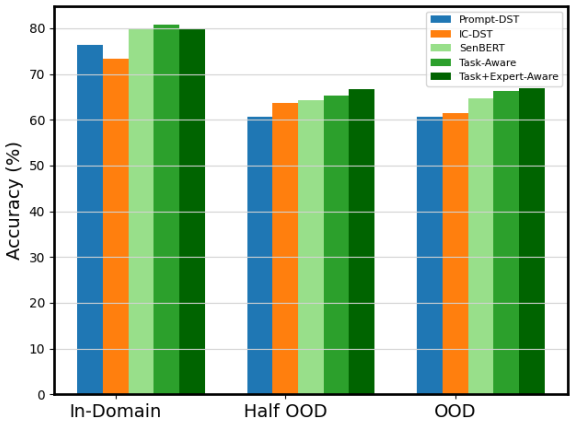

Figure 2: Cross-domain generalization results on SGD. We denote *In-Domain* when all of the testing domains are in the training set and denote *OOD* when all of the testing domains are not in the training set. For all other dialogues, we categorize them as *Half OOD*. We report TLB JGA for all settings. Green bars indicate `OrchestraLLM` with different retrievers.

setting on SGD in Table 3. We use 5% of dialogues in the training set for finetuning Prompt-DST and the retriever of IC-DST. Prompt-DST and IC-DST performed inference on another 100 dialogues in the validation set to serve as expert pool. We randomly select 300 turns from each expert to serve as expert pools for dynamic routing.

As we observe in MultiWOZ, incorporating an off-the-shelf SenBERT as the router improves the TLB score and also saves around 50% of computes. Finetuning SenBERT with the task-aware objective improves efficiency by 5% and increases both the TLB and DST scores. With the additional expert-aware supervision, more turns are routed to SLM and improves TLB score. This setting outperforms IC-DST by over 4% TLB JGA and saves 57% FLOPs, demonstrating that our router is universal enough to support cross-domain assignment and successfully improves system accuracy.

5 Analysis

5.1 Cross-Domain Generalization

Out-of-Domain (OOD) in SGD To assess the effectiveness of `OrchestraLLM` in generalizing to unseen domains, we present breakdown results on SGD in Figure 2. First, we observe that Prompt-DST performs better than IC-DST on in-domain dialogues but lags behind IC-DST on all other types of dialogues. This suggests that the generalization ability of Large Language Models (LLMs) is superior to Small Language Models (SLMs). All variants of `OrchestraLLM` outperform IC-DST in OOD scenarios, demonstrating the router's capability to effectively dispatch instances even when they are out of the domain.

Cross-Dataset Retriever We further evaluate our proposed framework in a more challenging scenario where the router and backbone models are trained in different datasets. We train the retriever model on MWOZ holdout set dialogues and evaluate the framework on SGD testing dialogues and vice versa. Notably, our routing framework can still effectively orchestrate two LLMs with a retriever trained with a different dataset and outperforms IC-DST while also achieving computational cost savings of approximately 54% on MultiWOZ and 43% on SGD (only slightly less savings than the domain-matched model).

5.2 Specialty of SLM and LLM

To better understand the complementary nature of the LMs, we inspected examples to identify their specialties. We provide representative examples from the expert pools in Table 4. One common mistake made by LLM is failing to adhere to the schema. In this example, LLM simply copies the text ("affordable") from the turn as a DST prediction, while SLM is capable of grounding the value in the schema-specific format ("cheap"). However, we identify two strengths that LLM possesses over SLM. Firstly, it excels in handling commonsense knowledge, for example, it can infer the correct number of guests staying at the guest house from the context ("me and my mum"). Secondly, it demonstrates proficiency in long-context reasoning. When there is a reference to previous context across domains, LLM consistently makes the correct inference, while SLM often overlooks the context and produces random values.

6 Related Work

6.1 Sample-Adaptive Inference

For allocating variable levels of computational resources for processing different input samples, two predominant categories of approaches have emerged in this domain: early exiting (Liu et al.,

Example from SLM pool	
DST of Previous Turn	*restaurant-area: centre*
Test turn	[system] Do you have a cuisine or price range in mind? [user] Yes, something in the affordable price range. Also, do any of them serve Singaporean food ?
SLM Prediction	*restaurant-food: Singaporean, restaurant-pricerange: cheap*
LLM Prediction	*restaurant-food: Singaporean, restaurant-pricerange: affordable*
Example from LLM Pool	
DST of Previous Turn	*hotel-name=Alpha Milton guest house*
Test turn	[system] Would you like to book a room? [user] That would be a massive help if you can do that for me! It's me and my mum and we'll be there for 2 nights.
SLM Prediction	*hotel-bookstay: 2, hotel-bookpeople: 1*
LLM Prediction	*hotel-bookstay: 2, hotel-bookpeople: 2*
DST of Previous Turn	*restaurant-name=Cocum, restaurant-area: west*
Test turn	[system] Can I be of any further assistance today? [user] Yes, I am also looking for a 3-star hotel located in the same area as the restaurant.
SLM Prediction	*hotel-stars: 3, hotel-area: centre*
LLM Prediction	*hotel-stars: 3, hotel-area: west*

Table 4: Representative examples from SLM and LLM pool. Red color text indicates the errors made by LMs.

2020; Xin et al., 2021; Zhou et al., 2020) and token dropping (Goyal et al., 2020; Guan et al., 2022; Kim and Cho, 2021). Salehi et al. (2023) also studies to direct different samples to sub-networks with varying widths. Our proposed routing framework also embraces sample-adaptive inference. However, it distinguishes itself by leveraging not just a single model but a combination of models.

6.2 Model Switching

There has been a growing body of research focusing on the concept of model switching, wherein input examples are intelligently routed between small and large models based on their individual complexity levels. For instance, Madaan et al. (2023) proposes a methodology that leverages an external meta-verifier to ascertain the correctness of predictions made by a SLM and to determine whether an example warrants routing to a LLM. In contrast, our approach does not necessitate the use of additional verifiers. Another set of related approaches, exemplified by the work of Šakota et al. (2023); Kag et al. (2022), involves training binary classifiers to categorize examples as suitable for SLM or LLM processing. This approach requires the router to be trained on labeled data where language models have made predictions. In contrast, our methodology exhibits the ability to leverage off-the-shelf retriever, enhancing its versatility.

6.3 Few-Shot Dialogue state tracking

To reduce the need for labeled data in DST, many approaches are proposed for few-shot DST (Li et al., 2021; Lin et al., 2021; Shin et al., 2022; Hu et al., 2022). The state-of-the-art few-shot DST model is (King and Flanigan, 2023b), in which the authors reformulate DST as a Python programming task and leverages Codex (Chen et al., 2021) as the backbone LLM, which is no longer accessible. Additionally, their approach involves multiple decoding passes for a single turn and relies on probability scores of tokens, which might not always be readily available.

7 Conclusion

We introduce `OrchestraLLM`, a routing framework that seamlessly integrates a SLM and a LLM, orchestrated by a retrieval-based router. During inference, a dynamic router guides instances to either LM based on their semantic embedding distances with the retrieved LM exemplars, leveraging the expertise of both SLM and LLM. Our evaluation on DST demonstrates that `OrchestraLLM` outperforms LLM-based systems while also achieving computational cost savings of over 50%. This research represents a significant step towards efficient collaboration of language models, particularly in a multi-turn human-computer interaction system such as task-oriented dialogue.

8 Limitations

Our study demonstrates the benefits of combining SLM and LLM for improved task performance while managing computational costs, particularly in the context of dialogue state tracking tasks. However, it's important to acknowledge that the applicability of our approach may not extend seamlessly to all types of tasks in the broader NLP domain. Additionally, in our current framework, we focus on leveraging a SLM and a LLM. However, real-world applications often involve a wide array of diverse tasks, each potentially requiring LMs with varying expertise. As a future avenue of research, we intend to explore the orchestration of multiple LMs simultaneously.

References

Namo Bang, Jeehyun Lee, and Myoung-Wan Koo. 2023. Task-optimized adapters for an end-to-end task-oriented dialogue system. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 7355–7369, Toronto, Canada. Association for Computational Linguistics.

Jeff Barr. 2019. Amazon ec2 update. Blog Post. Accessed: 2021-06-01.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gasic. 2018. Multiwoz-a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5016–5026.

Lingjiao Chen, Matei Zaharia, and James Zou. 2023. Frugalgpt: How to use large language models while reducing cost and improving performance. *arXiv preprint arXiv:2305.05176*.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. 2022. Llm. int8 (): 8-bit matrix multiplication for transformers at scale. *arXiv preprint arXiv:2208.07339*.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning of quantized llms. *arXiv preprint arXiv:2305.14314*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Suvodip Dey, Ramamohan Kummara, and Maunendra Desarkar. 2022. Towards fair evaluation of dialogue state tracking by flexible incorporation of turn-level performances. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 318–324.

Mihail Eric, Rahul Goel, Shachi Paul, Abhishek Sethi, Sanchit Agarwal, Shuyang Gao, Adarsh Kumar, Anuj Goyal, Peter Ku, and Dilek Hakkani-Tur. 2020. Multiwoz 2.1: A consolidated multi-domain dialogue dataset with state corrections and state tracking baselines. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 422–428.

Angela Fan, Edouard Grave, and Armand Joulin. 2019. Reducing transformer depth on demand with structured dropout. In *International Conference on Learning Representations*.

Saurabh Goyal, Anamitra Roy Choudhury, Saurabh Raje, Venkatesan Chakaravarthy, Yogish Sabharwal, and Ashish Verma. 2020. Power-bert: Accelerating bert inference via progressive word-vector elimination. In *International Conference on Machine Learning*, pages 3690–3699. PMLR.

Yue Guan, Zhengyi Li, Jingwen Leng, Zhouhan Lin, and Minyi Guo. 2022. Transkimmer: Transformer learns to layer-wise skim. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7275–7286.

Donghoon Ham, Jeong-Gwan Lee, Youngsoo Jang, and Kee-Eung Kim. 2020. End-to-end neural pipeline for goal-oriented dialogue systems using gpt-2. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 583–592.

Matthew Henderson, Blaise Thomson, and Jason D. Williams. 2014. The second dialog state tracking challenge. In *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and*

Dialogue (SIGDIAL), pages 263–272, Philadelphia, PA, U.S.A. Association for Computational Linguistics.

Ehsan Hosseini-Asl, Bryan McCann, Chien-Sheng Wu, Semih Yavuz, and Richard Socher. 2020. A simple language model for task-oriented dialogue. *Advances in Neural Information Processing Systems*, 33:20179–20191.

Yushi Hu, Chia-Hsuan Lee, Tianbao Xie, Tao Yu, Noah A. Smith, and Mari Ostendorf. 2022. In-context learning for few-shot dialogue state tracking. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 2627–2643, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Vojtěch Hudeček and Ondřej Dušek. 2023. Are large language models all you need for task-oriented dialogue? In *Proceedings of the 24th Meeting of the Special Interest Group on Discourse and Dialogue*, pages 216–228.

Wiradee Imrattanatrai and Ken Fukuda. 2023. End-to-end task-oriented dialogue systems based on schema. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 10148–10161, Toronto, Canada. Association for Computational Linguistics.

Joel Jang, Seungone Kim, Seonghyeon Ye, Doyoung Kim, Lajanugen Logeswaran, Moontae Lee, Kyungjae Lee, and Minjoon Seo. 2023. Exploring the benefits of training expert language models over instruction tuning. *arXiv preprint arXiv:2302.03202*.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2020. Tinybert: Distilling bert for natural language understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4163–4174.

Anil Kag, Igor Fedorov, Aditya Gangrade, Paul Whatmough, and Venkatesh Saligrama. 2022. Efficient edge inference by selective query. In *The Eleventh International Conference on Learning Representations*.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Gyuwan Kim and Kyunghyun Cho. 2021. Length-adaptive transformer: Train once with length drop, use anytime with search. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6501–6511, Online. Association for Computational Linguistics.

Brendan King and Jeffrey Flanigan. 2023a. Diverse retrieval-augmented in-context learning for dialogue state tracking. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 5570–5585.

Brendan King and Jeffrey Flanigan. 2023b. Diverse retrieval-augmented in-context learning for dialogue state tracking. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 5570–5585, Toronto, Canada. Association for Computational Linguistics.

François Lagunas, Ella Charlaix, Victor Sanh, and Alexander M Rush. 2021. Block pruning for faster transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10619–10629.

Chia-Hsuan Lee, Hao Cheng, and Mari Ostendorf. 2021. Dialogue state tracking with a language model using schema-driven prompting. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4937–4949.

George Leopold. 2019. Aws to offer nvidia's t4 gpus for ai inferencing. Online. Accessed: 2021-06-01.

Shuyang Li, Jin Cao, Mukund Sridhar, Henghui Zhu, Shang-Wen Li, Wael Hamza, and Julian McAuley. 2021. Zero-shot generalization in dialog state tracking through generative question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1063–1074, Online. Association for Computational Linguistics.

Zhaojiang Lin, Bing Liu, Seungwhan Moon, Paul A Crook, Zhenpeng Zhou, Zhiguang Wang, Zhou Yu, Andrea Madotto, Eunjoon Cho, and Rajen Subba. 2021. Leveraging slot descriptions for zero-shot cross-domain dialogue state tracking. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5640–5648.

Weijie Liu, Peng Zhou, Zhiruo Wang, Zhe Zhao, Haotang Deng, and Qi Ju. 2020. FastBERT: a self-distilling BERT with adaptive inference time. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6035–6044, Online. Association for Computational Linguistics.

Aman Madaan, Pranjal Aggarwal, Ankit Anand, Srividya Pranavi Potharaju, Swaroop Mishra, Pei Zhou, Aditya Gupta, Dheeraj Rajagopal, Karthik Kappaganthu, Yiming Yang, et al. 2023. Automix: Automatically mixing language models. *arXiv preprint arXiv:2310.12963*.

Paul Michel, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one? *Advances in neural information processing systems*, 32.

Baolin Peng, Chunyuan Li, Jinchao Li, Shahin Shayandeh, Lars Liden, and Jianfeng Gao. 2020. Soloist: Few-shot task-oriented dialog with a single pre-trained auto-regressive model.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551.

Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 2020. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 8689–8696.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Marija Šakota, Maxime Peyrard, and Robert West. 2023. Fly-swat or cannon? cost-effective language model choice via meta-modeling. *arXiv preprint arXiv:2308.06077*.

Mohammadreza Salehi, Sachin Mehta, Aditya Kusupati, Ali Farhadi, and Hannaneh Hajishirzi. 2023. Sharcs: Efficient transformers through routing with dynamic width sub-networks. *arXiv preprint arXiv:2310.12126*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Sheng Shen, Zhen Dong, Jiayu Ye, Linjian Ma, Zhewei Yao, Amir Gholami, Michael W Mahoney, and Kurt Keutzer. 2020. Q-bert: Hessian based ultra low precision quantization of bert. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8815–8821.

Jamin Shin, Hangyeol Yu, Hyeongdon Moon, Andrea Madotto, and Juneyoung Park. 2022. Dialogue summaries as dialogue states (DS2), template-guided summarization for few-shot dialogue state tracking. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3824–3846, Dublin, Ireland. Association for Computational Linguistics.

Yixuan Su, Lei Shu, Elman Mansimov, Arshit Gupta, Deng Cai, Yi-An Lai, and Yi Zhang. 2022. Multi-task pre-training for plug-and-play task-oriented dialogue system. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4661–4676.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Well-read students learn better: On the importance of pre-training compact models. *arXiv preprint arXiv:1908.08962*.

Qingyue Wang, Liang Ding, Yanan Cao, Yibing Zhan, Zheng Lin, Shi Wang, Dacheng Tao, and Li Guo. 2023. Divide, conquer, and combine: Mixture of semantic-independent experts for zero-shot dialogue state tracking. *arXiv preprint arXiv:2306.00434*.

Ziheng Wang, Jeremy Wohlwend, and Tao Lei. 2020. Structured pruning of large language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6151–6162.

Chien-Sheng Wu, Steven CH Hoi, and Caiming Xiong. 2020. Improving limited labeled dialogue state tracking with self-supervision. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4462–4472.

Mengzhou Xia, Zexuan Zhong, and Danqi Chen. 2022. Structured pruning learns compact and accurate models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1513–1528.

Tianbao Xie, Chen Henry Wu, Peng Shi, Ruiqi Zhong, Torsten Scholak, Michihiro Yasunaga, Chien-Sheng Wu, Ming Zhong, Pengcheng Yin, Sida I Wang, et al. 2022. Unifiedskg: Unifying and multi-tasking structured knowledge grounding with text-to-text language models. *arXiv preprint arXiv:2201.05966*.

Ji Xin, Raphael Tang, Yaoliang Yu, and Jimmy Lin. 2021. BERxiT: Early exiting for BERT with better fine-tuning and extension to regression. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 91–104, Online. Association for Computational Linguistics.

Fanghua Ye, Jarana Manotumruksa, and Emine Yilmaz. 2021. Multiwoz 2.4: A multi-domain task-oriented dialogue dataset with essential annotation corrections to improve state tracking evaluation. *arXiv preprint arXiv:2104.00773*.

Fanghua Ye, Jarana Manotumruksa, and Emine Yilmaz. 2022. MultiWOZ 2.4: A multi-domain task-oriented dialogue dataset with essential annotation corrections to improve state tracking evaluation. In *Proceedings of the 23rd Annual Meeting of the Special Interest*

Group on Discourse and Dialogue, pages 351–360, Edinburgh, UK. Association for Computational Linguistics.

Wangchunshu Zhou, Canwen Xu, Tao Ge, Julian McAuley, Ke Xu, and Furu Wei. 2020. Bert loses patience: Fast and robust inference with early exit. *Advances in Neural Information Processing Systems*, 33:18330–18341.

A Related Work

A.1 Sample-Adaptive Inference

To enable variable levels of computational resources to be allocated for processing different input samples, two predominant categories of approaches have emerged in this domain: early exiting and token dropping. Early exiting strategies typically incorporate additional classifiers at intermediate layers within a model, determining whether a given input example should terminate its processing prematurely and abstain from propagating to subsequent layers (Liu et al., 2020; Xin et al., 2021; Zhou et al., 2020). On the other hand, token-dropping techniques aim to dynamically reduce the length of the input sequence, achieved by selectively excluding certain tokens from the input sequence, which are subsequently not passed on to subsequent layers in the model (Goyal et al., 2020; Guan et al., 2022; Kim and Cho, 2021). Salehi et al. (2023) also studies to direct different samples to sub-networks with varying widths. Our proposed routing framework also embraces sample-adaptive inference. However, it distinguishes itself by leveraging not just a single model but a combination of models.

A.2 Model Compression

Model compression primarily falls into three major paradigms: pruning, distillation, and quantization. Pruning strategies are primarily devised to reduce computational overhead by selecting and retaining a subnetwork within a larger model (Fan et al., 2019; Michel et al., 2019; Wang et al., 2020; Lagunas et al., 2021; Xia et al., 2022). In contrast, distillation techniques entail the training of a compact student model, with the objective of imparting the knowledge and performance of a larger teacher model (Sanh et al., 2019; Turc et al., 2019; Jiao et al., 2020). Finally, quantization methods aim to diminish memory demands by representing model parameters with fewer bits, thereby trading off a degree of precision for enhanced efficiency (Shen et al., 2020; Dettmers et al., 2022, 2023). Note that the aforementioned methods are characterized as "static" in nature, as they primarily focus on the optimization of fixed model architectures for each data point. In contrast, the routing framework introduced in this work adopts a dynamic perspective.