Dynamic Strategy Planning for Efficient Question Answering with Large Language Models

Tanmay Parekh[*†] **Pradyot Prakash**[‡] **Alexander Radovic**[‡]
Akshay Shekher[‡] **Denis Savenkov**[‡]
[†]Univeristy of California, Los Angeles [‡]Meta AI
tparekh@cs.ucla.edu, {pradyot, alexradovic, shekher, denxx}@meta.com

Abstract

Research has shown the effectiveness of reasoning (e.g., Chain-of-Thought), planning (e.g., SelfAsk), and retrieval augmented generation strategies to improve the performance of Large Language Models (LLMs) on various tasks, such as question answering. However, using a single fixed strategy to answer different kinds of questions is suboptimal in performance and inefficient in terms of generated output tokens and performed retrievals. In our work, we propose a novel technique DyPlan, to induce a dynamic strategy selection process in LLMs, to improve performance and reduce computational costs in question-answering. DyPlan incorporates an initial decision step to select the most suitable strategy conditioned on the input question and guides the LLM's response generation accordingly. We extend DyPlan to DyPlan-verify, adding an internal verification and correction process to further enrich the generated answer. Experiments on three prominent multi-hop question answering (MHQA) datasets reveal how DyPlan can improve model performance by 7-13% while reducing the computational cost by 11-32% relative to the best baseline model. Code for this work can be found at https://github.com/facebookresearch/dyplan.

1 Introduction

Question-answering (QA) for large language models (LLMs) spans a range of question types, from simple queries to those requiring reasoning, external knowledge, step-by-step planning, or a combination of these strategies. For example (Figure 1), modern LLMs can easily answer *Who was the first president of USA?* but may need some reasoning to figure out *At what age did Roger Federer win his first Grand Slam Title?*, while *Who's contending the 2024 US Presidential Elections?* requires the model to retrieve up-to-date external information.

[*]Work completed as part of an internship at Meta.

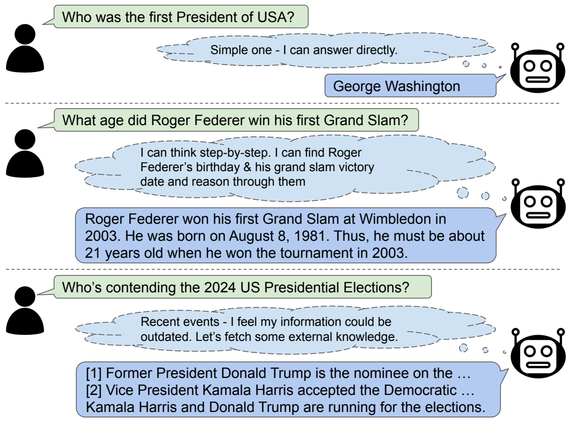

Figure 1: Illustration of dynamically deciding appropriate strategies (indicated by the clouds) conditioned on the input questions.

To this end, previous works have investigated various strategies to induce reasoning, such as Chain of Thought (Wei et al., 2022), Tree-of-Thought (Yao et al., 2023a); or planning, such as SelfAsk (Press et al., 2023), Decomposed Prompting (Khot et al., 2022), StepBack Prompting (Zheng et al., 2024a); or incorporating external knowledge through Retrieval Augmented Generation (Lewis et al., 2020) and Knowledge Graphs (Pan et al., 2024).

However, employing a single strategy for all different types of questions is sub-optimal as well as quite cost-ineffective in terms of generated tokens and retrievals. As humans, we rather employ a dynamic decision phase to first determine the most effective strategy before answering the given question. Similarly, we expect that if the model possesses sufficient self-knowledge to directly answer the question, then 'thinking step-by-step' and expending tokens on reasoning is unnecessary. In other cases, a model may not have enough confidence to answer directly and should spend some computation on additional reasoning. However, not having enough information about the topic should warrant external retrievals instead.

Findings of the Association for Computational Linguistics: NAACL 2025, pages 6053–6074
April 29 - May 4, 2025

To this end, we propose to induce a human-like cognitive ability in LLMs through our novel technique, **DyPlan** (**Dy**namic **Plan**ning). As illustrated in Figure 1, DyPlan introduces an initial decision step to select the most suitable strategy conditioned on the input question and then guides the LLM's response generation to use this strategy. To achieve this behavior, we utilize a multi-turn training paradigm, where the LLM is fine-tuned and calibrated by its own generations. DyPlan provides a computationally cost-effective and adaptive solution that efficiently leverages the strengths of various techniques while minimizing computational overhead.

However, there is no complete certainty that the chosen strategy will succeed, as reasoning can be wrong, or retrieved information may turn out to be irrelevant or limited. At such times, humans usually evaluate and re-select a new strategy to rectify any potential mistakes. We emulate this internal assessment in LLMs by extending DyPlan as **DyPlan-verify** (**Dy**namic **Plan**ning & **Verify**). Specifically, we add a self-verification step after response generation, which gauges the model's confidence in the provided answer. If verification fails, the model is prompted to re-select a different strategy, and this cycle can repeat. Overall, DyPlan-verify can achieve higher quality improvements at the cost of additional inference computations.

To evaluate the efficacy of our proposed techniques, we benchmark them on three QA datasets - HotpotQA (Yang et al., 2018), 2WikiMultihopQA (Ho et al., 2020), and Musique (Trivedi et al., 2022). We consider four primary strategies: (1) *Direct* answering directly, (2) *Reason* utilizing Chain-of-Thought (Wei et al., 2022), (3) *Plan* leveraging SelfAsk (Press et al., 2023), and (4) *Retrieval* using external knowledge with RAG (Lewis et al., 2020). We use the LLaMa3-8B model (Dubey et al., 2024) as our base model. We majorly compare against fine-tuned LLMs utilizing fixed strategies along with other ensemble and dynamic thinking baselines. Results reveal that DyPlan reduces computational costs by 26-32% along with performance gains of 7% averaged across the datasets over the best baseline. DyPlan-verify further improves performance to an average of 12-13% while providing 11-19% computational cost reductions. Analyses reveal how DyPlan is better calibrated and generalizable and provide insights into its decision-making and verification ability.

In conclusion, we make these contributions: (1) we propose dynamic strategy planning through DyPlan to improve the performance and computational cost-efficiency of LLMs for QA, (2) we extend DyPlan to DyPlan-verify introducing verification of correctness to further boost model performance, (3) we conduct extensive experiments and analyses using four major strategies on three complex QA datasets to demonstrate DyPlan's cost-effectiveness and strong performance.

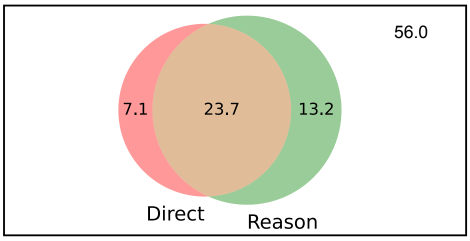

Figure 2: Venn Diagram representing the F1 contribution of Direct and Reason strategies for HotpotQA.

2 Methodology

To mimic cost-effective human cognitive thinking in LLMs, we propose our novel technique - **DyPlan**. Unlike traditional approaches that rely on a single fixed strategy for all questions, our technique employs dynamic strategy planning to determine the most effective approach for each question. We extend DyPlan to **DyPlan-verify** by incorporating additional verification and re-attempting the question with alternative strategies if necessary. We first motivate the potential impact of dynamic strategy planning in § 2.1 and later provide specific details about our techniques.

2.1 Motivation

Dynamic strategy planning can help reduce inference computational costs by simply selecting lower-cost strategies to answer simpler questions. In our work, we additionally posit that it can improve model performance as strategy selection can act as an ensembling method. We verify this hypothesis through a simple analysis using two strategies of *Direct* and *Reason*. Direct prompts the LLM to directly provide the answer to the question while Reason utilizes Chain-of-Thought (Wei et al., 2022) to answer step-by-step. Utilizing these strategies, we prompt Llama3-8B-Instruct (Dubey et al., 2024), evaluate using F1 score on 1000 samples from 2WikiMultihopQA (Ho et al., 2020) dataset, and show their performances as a Venn diagram in Figure 2. Generally, one can assume that adding

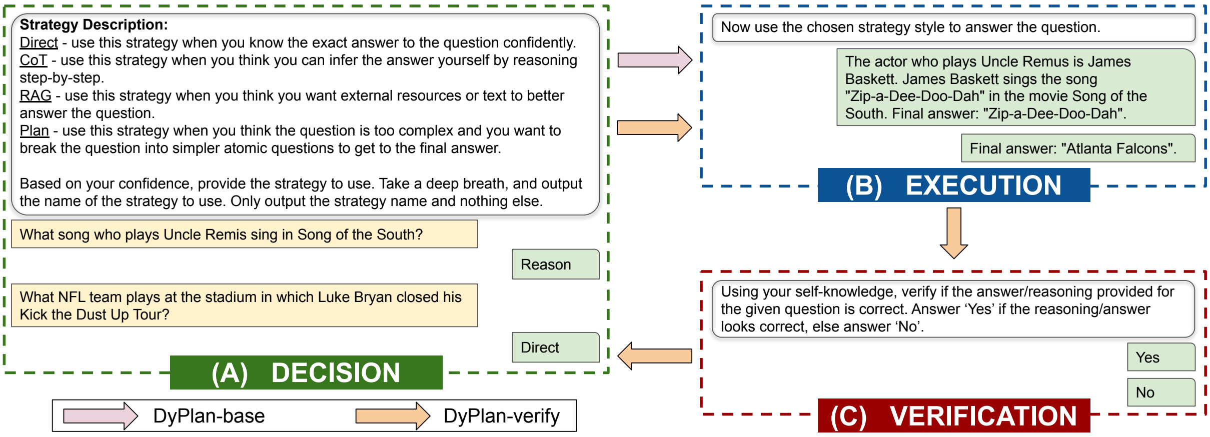

Figure 3: The different components in DyPlan along with the pipeline flow for two example questions. The Decision component chooses an appropriate strategy from a pool of strategies. The Execution component runs the chosen strategy. The Verification component (used in DyPlan-verify) self-verifies the correctness of the provided answer.

Question: Which band, Letters to Cleo or Screaming Trees, had more members? **Gold Answer:** Letters to Cleo
Direct: Final answer: "Letters to Cleo"
Reason: Letters to Cleo had 4 members: Brian Desveaux, Michael Hubbard, Ken Ray, and Stacy Jones. Screaming Trees had 4 members: Mark Lanegan, Gary Lee Conner, Van Conner, and Barrett Martin. Both bands had the same number of members, 4. Final answer: "same"

Table 1: Qualitative example highlighting how LLMs can yield the right answer by directly answering the question, while reasoning can lead to incorrect answers.

reasoning should only allow us to further answer harder questions while still being able to answer all questions that Direct strategy answers correctly. However, there is a significant contribution of 7.1% F1 where Reason is incorrect, but Direct is correct (for example, a mistake or hallucination can lead to an incorrect result). We provide a qualitative example in Table 1 highlighting this behavior (more examples in Table 24). We notice similar patterns across datasets and strategies (shown in § B.1). Such patterns shed light on how the choice of an appropriate strategy can also improve model performance.

2.2 DyPlan Components

Our techniques majorly utilize three components in a plug-and-play manner: (1) Decision, which selects a strategy to follow; (2) Execution, which generates the answer using the chosen strategy; and (3) Verification, which evaluates the answer's correctness. We describe them in detail below and provide a high-level overview diagram in Figure 3.

Decision: The Decision component is the core of our technique, responsible for dynamically selecting the optimal strategy for a given question. This is achieved by presenting an LLM with a strategy pool with their descriptions and prompting it to leverage its self-confidence to choose the most suitable and efficient strategy. This component provides an opportunity to optimize efficiency while still enabling powerful reasoning to improve performance when possible, unlike OpenAI's o1 model,[1] which currently applies thinking even for simple questions. We provide an illustration prompt of this component in Figure 3(A).

Execution: The Execution component involves prompting the model to apply the selected strategy from the Decision component to generate an answer to the question. Although analogous to fixed-strategy prompting, our approach is different since we enable dynamic strategy execution based on the previously chosen strategy. We provide an illustration Execution prompt in Figure 3(B).

Verification: The Verification component is optional and only part of our extended technique DyPlan-verify. Intuitively, when we make a decision to choose a certain strategy, like reasoning or retrieval, we cannot be certain it will be successful, as reasoning may fail and retrieval may get irrelevant results. Therefore, DyPlan should have the ability to correct the course as long as we have some more budget before having to present the final answer. This component assesses the validity of the Execution output by prompting the LLM to lever-

[1] https://platform.openai.com/docs/models/o1

age its self-knowledge and confidence to evaluate the answer's reasonableness and correctness. To minimize computational cost, we implement this component by asking the LLM to simply output yes/no, as shown in Figure 3(C).

2.3 DyPlan Pipeline Flow

DyPlan majorly achieves computational cost minimization by dynamic decision-making in the Decision phase and restricting generation output space for each component. The base version of DyPlan (also referred to as DyPlan-base) employs a low-cost Decision-Execution pipeline (pink arrow in Figure 3). On the other hand, our extension technique DyPlan-verify utilizes an iterative loop of Decision-Execution-Verification (orange arrow in Figure 3). If verification fails, the pipeline reverts to the Decision component to select an alternative strategy; otherwise, it exits the loop with the execution answer. This iterative loop runs for a preset number of rounds based on the inference budget or until the Decision runs out of usable strategies. Both pipelines are implemented using multi-turn chat, with each component corresponding to a single turn.

3 Data Creation for Finetuning

To adhere LLMs with the DyPlan pipeline, we fine-tune the LLM on DyPlan-specific data. To ensure zero human annotation cost, we propose automatic data creation utilizing an existing QA dataset $\mathcal{D}$ and a strategy set $\mathcal{S}$ comprising n strategies. We order $\mathcal{S} = [s_1, \ldots, s_n]$ by strategy preference, with s_1 being the most preferred and s_n the least preferred (e.g. in order of computational cost-efficiency/performance). For each strategy $s \in \mathcal{S}$, we prompt the base LLM on all datapoints $d \in \mathcal{D}$ and evaluate the results against the ground truth. This yields two disjoint subsets for each s: D_p^s, comprising datapoints where s produced the correct answer, and D_n^s, comprising the remaining datapoints where s failed to produce the correct answer. Utilizing the base LLM (instead of distilling from larger LLMs) ensures a stronger model self-calibration. Using the positive and negative subsets, we create component-specific data for DyPlan (described below) and train the LLM on the combination of all the component data. We conduct training only on the last-turn response for the multi-turn training instances.

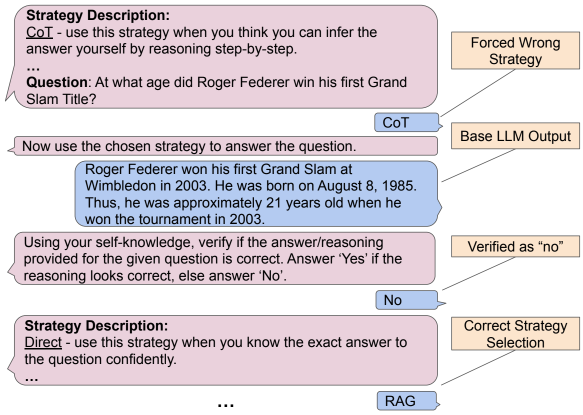

Figure 4: Illustration of an automatically created multi-turn data instance utilizing a forced wrong strategy in the first round with the correct strategy in the second round. LLM is only trained on the second round.

Decision: We define an optimal mapping function $f^* : \mathcal{D} \to \mathcal{S}$ that assigns each training datapoint $d \in \mathcal{D}$ to the first strategy $s \in \mathcal{S}$ (according to the preference order) that yields the correct answer. If none of the strategies produce the correct answer, d is mapped to the least preferred strategy s_n. The Decision component's training data consists of mapped input-output pairs $(d, f^*(d))$.

Execution: The input here is a multi-turn chat where the first turn (Decision) selects a strategy s. In the second turn (Execution), the output is set as the base LLM generation using strategy s. Utilizing the base LLM response aids efficient and faster model training. To minimize noise, we utilize only the positive data D_p^s for each strategy s.

Verification: For this component, we create binary training data by mapping positive data D_p^s to "yes" and negative data D_n^s to "no". Multi-turn traces are generated by forcing the selection of strategy s in the first turn (Decision) and using the base LLM response in the second turn (Execution).

Multi-round data: To facilitate multiple rounds of the Decision-Execution-Verification pipeline for DyPlan-verify, we generate additional training data for each component using a reduced dataset $\mathcal{D}' \subset \mathcal{D}$. Specifically, $\mathcal{D}'$ comprises subsets $\mathcal{D}_{n,p}^{s_i,s_j}$, where each datapoint $d \in \mathcal{D}_{n,p}^{s_i,s_j}$ satisfies $d \in \mathcal{D}_n^{s_i}$ and $d \in \mathcal{D}_p^{s_j}$. In other words, strategy s_i is incorrect for d and is used as the wrong strategy in the first round, while the correct strategy s_j is used in the second round. We provide an illustration of such a two-round training instance in Figure 4.

4 Experimentation Details

We describe the benchmarking datasets and the evaluation metrics. Next, we discuss the strategies, baselines, and, finally, the implementation details.

Benchmarking Datasets: LLMs seem to perform well on simpler question-answering datasets like SQuAD (Mavi et al., 2024). Instead, we consider three complex Wikipedia-based multi-hop QA (MHQA) datasets to benchmark the performance of our technique, namely HotpotQA (Yang et al., 2018), 2WikiMultihopQA (2WikiQA) (Ho et al., 2020), and Musique (Trivedi et al., 2022). HotpotQA was one of the first human-created MHQA datasets with upto 2-hop reasoning questions. 2WikiMultihopQA further improved over HotpotQA by improving the complexity and reasoning depth of the questions. Musique is a rule-based constructed dataset created by composing different single-hop questions. These unique challenges of each dataset aid extensive benchmarking. We utilize 1000 samples from the development sets of these datasets as the main evaluation dataset.

Evaluation Metrics: We evaluate the models on two major dimensions of *performance* and *computational cost*. For performance, we utilize **Exact Match (EM)** and **F1 score** evaluated against the ground truth - higher the better. For computational cost, we consider the **number of generated tokens (# T)** and **number of retrievals (# R)** - lower the better. For DyPlan, we report the aggregated cost metrics across the turns.

Strategies: We focus on four major themes of strategies, as follows:

1. *Direct*: LLM is prompted to directly provide the final answer. This is the cheapest strategy in terms of computational cost.
2. *Reason*: LLM is prompted to reason to reach the final answer. We utilize Chain-of-Thought (CoT) (Wei et al., 2022) to reason step-by-step. This strategy is more expensive than *Direct* in terms of generated tokens.
3. *Plan*: LLM is prompted to decompose the question as part of planning and reason through the atomic questions to reach the final answer. We utilize SelfAsk (Press et al., 2023) as a prototype for this strategy. This is the most expensive in terms of generated tokens.
4. *Retrieval*: Following RAG (Lewis et al., 2020), using the question as the query, three external passages retrieved from Wikipedia are fed to the LLM. LLM is prompted to reason to reach the final answer. This strategy is expensive in terms of retrievals.

Baseline Models: As baselines, we consider: (1) *Fixed-base* prompts the base LLM with a single fixed strategy, (2) *Fixed-sft* prompts a LLM fine-tuned on the fixed strategy using the positive base LLM traces, (3) *Classifier* trains an external classifier to choose the strategy and chooses the corresponding fine-tuned LLM response, (4) *Ensemble* simply outputs the majority ensemble using all the Fixed-sft strategy responses.

Additionally, we consider some similar works utilizing dynamic decision-making as reference such as: (5) *ReAct* (Yao et al., 2023b) uses thoughts-actions-observation tuples to guide model generation. (6) *DRAGIN* (Su et al., 2024) utilizes dynamic retrieval based on model entropy. Both these baselines are orthogonal to our work and can be utilized in a complementary manner as individual strategies for DyPlan. We majorly compare the cost-effectiveness of DyPlan with these techniques.

Implementation Details: For all experiments, we utilize the LLaMa3-8B-Instruct model (Dubey et al., 2024) as the base LLM. We set the strategy order in increasing order of model performance as Direct-Plan-Reason-Retrieval for training data creation. We use Low-Rank Adaptation (Hu et al., 2022) with rank 32 using LLaMa-Factory (Zheng et al., 2024c) for fine-tuning the base LLM. We utilize code from DRAGIN (Su et al., 2024) to implement the fixed strategy baselines as well as evaluate our techniques. Our reported numbers are averaged scores over three runs. Additional details and hyperparameters are provided in Appendix A.

5 Results

We present our main results comparing DyPlan utilizing all the strategies with other baselines in Table 2. We utilize the best-performing Fixed-sft Retrieval model as the reference baseline for comparisons. We also aggregate these metrics across datasets and plot Performance (F1 score) v/s Efficiency (weighted sum of # T and # R)[2] in Figure 5.

[2]Weights are determined based on pricing of input and output tokens for GPT4o-mini.

Technique	HotpotQA				2WikiMultihopQA				Musique			
	EM	F1	# T	# R	EM	F1	# T	# R	EM	F1	# T	# R
Fixed-base Direct	23.8	32.3	95	0	32.1	37.4	65	0	2.3	9.3	99	0
Fixed-base Reason	27.2	37.5	124	0	19.7	27.4	65	0	7.2	16.7	129	0
Fixed-base Plan	24.1	33.8	203	0	25.4	31.5	197	0	5.8	13.4	203	0
Fixed-base Retrieval	36.1	47.9	185	1	31.6	40.4	101	1	9.6	18	187	1
Fixed-sft Direct	24.1	34.3	9	0	32.6	38.4	10	0	2.4	9	17	0
Fixed-sft Reason	27.6	37.9	53	0	29.3	35.6	77	0	7.6	16.4	63	0
Fixed-sft Plan	26.3	36	105	0	26.9	34.7	116	0	6.6	15	117	0
Fixed-sft Retrieval [ref]	36.8	48.6	53	1	32.8	40.0	56	1	9.3	18.4	88	1
Classifier	32.6	43.9	**34**	**0.59**	36.0	43.1	**28**	**0.45**	8.0	17.5	82	**0.90**
Ensemble	35.9	47.5	220	1	35.7	42.8	260	1	8.8	18.1	279	1
DyPlan-base (ours)	36.1	47.6	42	0.76	37.8	46.0	**28**	0.48	10.1	19.8	**65**	0.98
DyPlan-verify (ours)	36.7	48.5	53	0.79	**40.5**	**49.6**	45	0.65	10.8	20.4	77	0.99
ReAct	20.5	27.5	255	3.91	27.9	32.3	226	3.01	4.4	8.5	290	5.10
DRAGIN	**38.9**	**50.2**	724	2.23	32.7	41.8	272	1.67	**11.9**	**22.0**	993	3.03

Table 2: The main results comparing DyPlan and DyPlan-verify with other baselines. We mark the best and second-best metrics in **bold** and underline. [ref] indicates the main reference baseline.

Technique	HotpotQA				2WikiMultihopQA				Musique			
	EM	F1	# T	# R	EM	F1	# T	# R	EM	F1	# T	# R
Fixed-sft Direct	24.1	34.3	9	-	32.6	38.4	10	-	2.4	9	17	-
Fixed-sft Reason [ref1]	27.6	37.9	53	-	29.3	35.6	77	-	7.6	16.4	63	-
Fixed-sft Plan	26.3	36	105	-	26.9	34.7	116	-	6.6	15	117	-
Fixed-sft Retrieval [ref2]	36.8	48.6	53	1	32.8	40.0	56	1	9.3	18.4	88	1
Strategy Combination: Direct - Plan - Reason						**Reference: Fixed-sft Reason**						
Classifier	26.3	36.4	73	-	31.7	38.1	57	-	6.3	14.6	115	-
Ensemble	27.6	37.9	167	-	29.3	35.6	203	-	7.6	16.4	191	-
DyPlan-base (ours)	28.0	38.2	**47**	-	33.5	41.3	**36**	-	**8.1**	16.7	**67**	-
DyPlan-verify (ours)	**28.3**	**38.8**	59	-	**37.4**	**43.8**	68	-	7.9	**16.8**	164	-
Strategy Combination: Reason - Retrieval						**Reference: Fixed-sft Retrieval**						
Classifier	32.7	44.3	53	**0.52**	31.4	38.2	57	**0.54**	9.1	18.2	88	0.98
Ensemble	36.8	48.6	106	1	32.8	40.0	134	1	9.3	18.4	151	1
DyPlan-base (ours)	35.5	47.3	**51**	0.71	34.5	43.8	**50**	0.6	**10.8**	20.5	**65**	**0.97**
DyPlan-verify (ours)	**37.2**	**49.4**	92	0.79	**37.4**	**46.0**	57	0.74	10.6	**20.6**	71	**0.97**

Table 3: Performance and computational cost metrics for two strategy combinations of Direct-Plan-Reason ([ref1] is reference) and Reason-Retrieval ([ref2] is reference). We mark the best and second-best metrics in **bold** and underline.

We note that external classifiers help reduce computational cost but don't improve model performance - demonstrating the difficulty of the task. Dynamic decision-making frameworks like DRAGIN and Ensemble improve performance but are 2-5x more expensive. To this end, DyPlan provides the best balance with an average reduction of **32% token and 26% retrieval cost** along with relative performance improvements of **7% EM and 7% F1**. DyPlan-verify further improves performance with average relative gains of **13% EM and 12% F1** while reducing the token and retrieval cost by **11% and 19%** respectively. In the best case scenario on 2WikiMultihopQA, DyPlan shows 16% performance gains with 52% computational cost reductions, and DyPlan-verify shows 24% performance gains while reducing costs by 35%. Overall, DyPlan provides strong computational cost reductions along with decent performance gains.

5.1 Other strategy combinations

To demonstrate the generalizability of our technique across strategy combinations, we consider two additional combinations of strategies. The first combination - Direct, Plan, Reason - explores the ability of LLMs to utilize only their self-knowledge to answer the question. The second combination - Reason and Retrieval - explores the LLM's calibration to decide if it requires any external information to answer the question. We show the results

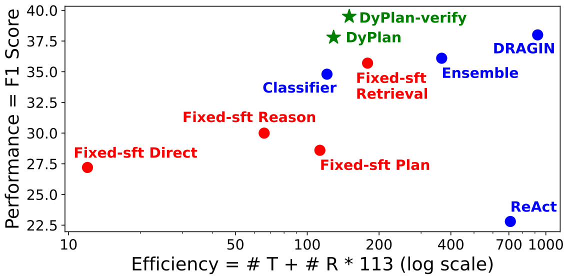

Figure 5: Performance v/s Inference Efficiency for various techniques. Our technique DyPlan (in green) provides the best performance while also reducing the inference costs relative to Fixed-sft Retrieval baseline.

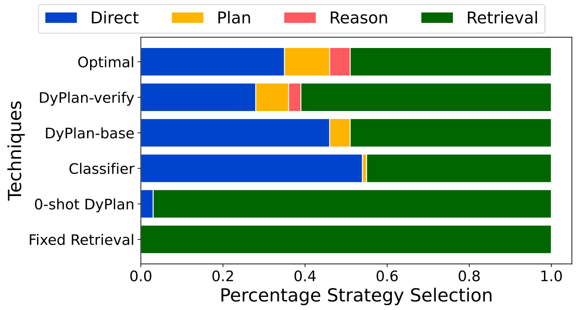

Figure 6: Comparing the strategy planning distribution of various techniques with optimal policy for 2WikiQA.

for these combinations in Table 3. Similar to the main results, we observe the superior performance of DyPlan and DyPlan-verify in terms of model performance (average relative gains of 6%-10%) as well as computational cost reduction (average reduction of 13%-20% tokens and 17%-24% retrievals).

6 Analysis

We conduct additional experiments to better understand the quality of DyPlan decision-making and verification and its generalization across datasets.

6.1 Calibration Analysis

In § 3, we defined an optimal policy f^* for each question as a strategy to pick the most cost-effective technique that yields the correct answer. Here, we analyze model calibration, that is, how well its decisions align with the optimal policy at test time (additional details are provided in § B).

6.1.1 Decision component of DyPlan

We compare the strategy planning distribution of various techniques with the optimal policy for 2WikiMultihopQA in Figure 6. The major difference is the usage of Plan and Reason which are nearly 0% for Fixed/Classifier approaches. On the other hand, DyPlan-base and DyPlan-verify are closer to the optimal distribution. We quantify this proximity of the probability distributions in terms of KL divergence. DyPlan-base and DyPlan-verify achieve low scores of 0.066 and 0.014, respectively, while the classifier baseline has a high divergence score of 0.35. Finally, for a stronger sanity check, we compute the accuracies of the strategy choice (relative to optimal policy) of DyPlan in Table 4. The high improvements relative to other baselines highlight the better strategy planning and stronger calibration of DyPlan, while throwing light towards further possible improvements.

Technique	Accuracy
Random	25.8%
Majority	30.8%
Classifier	48.1%
DyPlan	61.7%

Table 4: Accuracy of the Decision component of DyPlan in choosing the right strategy evaluated using the optimal policy on the HotpotQA dataset.

Dataset	KL-pre	KL-post	Reject %	Ver Prec
HotpotQA	0.281	0.068	8%	80%
2WikiQA	0.138	0.014	13%	71%
Musique	0.240	0.001	16%	97%

Table 5: Studying the impact of verification in DyPlan-verify by evaluating the strategy usage KL divergence with the optimal policy pre (KL-pre) and post (KL-post) verification, the % datapoints verified as “no" (Reject %) and the verification precision of “no" (Ver Prec).

6.1.2 Verification analysis of DyPlan-verify

We study the verification precision, answer rejection rate (% datapoints verified as “no"), and the change in strategy distribution pre and post-verification (in terms of KL divergence relative to optimal strategy) to gain a deeper understanding of the impact of verification in DyPlan-verify. We provide these statistics in Table 5. The huge drops in KL-divergence post-verification demonstrate how verification aids better alignment to the optimal policy and, thus, improves model calibration. The low rejection rate ensures the computational cost doesn’t increase significantly, while the high verification precision underlines the strong utility of the verification step.

Model	EM	F1	# T	# R
Fixed-base (Retrieval)	31.6	40.4	101	1
Fixed-sft (Retrieval)	32.8	40.0	56	1
0-shot DyPlan	32.1	40.6	100	0.97
Few-shot DyPlan	28.7	37.0	93	0.79
Fine-tuned DyPlan	**37.8**	**46.0**	**28**	**0.48**

Table 6: Ablation analysis on 2WikiMultihopQA for the need to fine-tune LLMs to incorporate DyPlan.

Model	HotpotQA		2WikiQA		Musique	
	EM	F1	EM	F1	EM	F1
DyPlan-base	36.1	47.6	37.8	46.0	10.1	19.8
DyPlan-verify	36.7	48.5	40.5	49.6	10.8	20.4
Upper Bound	47.0	60.5	51.3	60.2	14.2	23.0
+ Δ	**10.3**	**12.0**	**10.8**	**10.6**	**3.4**	**2.6**

Table 7: Empirical upper bounds for possible improvements of DyPlan using an oracle Decision component with base LLM responses for Execution. Δ indicates the potential improvement gap.

6.1.3 Fine-tuning improves calibration

Choosing the right strategy in a 0-shot way is difficult, as models don't surely know what they know and don't know (Yin et al., 2023a). We analyze the 0-shot DyPlan strategy planning in Figure 6 and note how the 0-shot model mostly resorts to the most expensive strategy, while fine-tuning helps to learn the patterns between questions and model capabilities. We also compare the model performance of 0-shot / few-shot DyPlan with fine-tuned DyPlan in Table 6 for 2WikiMultihopQA. Clearly, the non-fine-tuned models fail to improve over the fixed strategy baseline, but fine-tuning provides strong performance gains - demonstrating how fine-tuning strongly improves calibration for strategy planning.

6.1.4 Optimal Policy Upper Bound

We study the upper bound for DyPlan to motivate possibilities of future improvements. Specifically, we replace the DyPlan's Decision component with the optimal policy and use the corresponding fixed strategy base LLM outputs for Execution. We compare this upper bound with DyPlan in Table 7 with Δ, indicating further potential improvement. While Musique exhibits a low Δ of 2-3 F1 points, the larger Δ of 10-12% F1 for the other datasets provides promise to further explore strategy planning.

6.2 Generalization Analysis

To assess the generalization of DyPlan, we fine-tune it on combined data from the three benchmark datasets. To ensure a fair comparison, the combined data comprises 20k datapoints (same as individual data) with equal shares from the three datasets. We compare the performance of this combined-data fine-tuned model with the individual-data fine-tuned model in Figure 7 and note how the performances are nearly similar for both models. The cost analysis for the combined data model (Table 20 in § B.4) reveals at-par levels of computational costs as well. Thus, this study reveals how the gains provided by DyPlan/DyPlan-verify are generalizable and not overfitting to a single dataset.

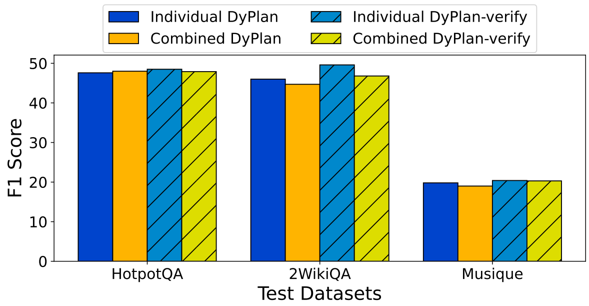

Figure 7: Assessing the generalizability of DyPlan by comparing the performance (F1 score) of combined-data training with individal-data training.

Strategy Ordering	EM	F1	# T	# R
DyPlan				
Direct, Plan, Reason, Retrieve	36.1	47.6	42	0.76
Direct, Reason, Plan, Retrieve	35.4	46.8	40	0.69
Reason, Direct, Plan, Retrieve	36.1	47.7	42	0.72
DyPlan-verify				
Direct, Plan, Reason, Retrieve	36.7	48.5	53	0.78
Direct, Reason, Plan, Retrieve	36.5	48.5	51	0.72
Reason, Direct, Plan, Retrieve	37.2	48.5	58	0.76

Table 8: Impact of different strategy orderings for DyPlan on downstream performance and computational costs on the HotpotQA dataset.

6.3 Analyzing the Order of Strategies

In this analysis, we study the impact of different strategy ordering S for DyPlan. Specifically, we consider three different orderings for HotpotQA and show the results in Table 8. Differently ordering the strategies (e.g., Direct, Reason, Plan, Retrieve) can help further reduce the computational cost by 4-7%, but it can also reduce the performance by 1-2%. Similarly, the performance can be optimized by a different ordering (e.g., Reason, Direct, Plan, Retrieve), with improvements upto 1-

2% while incurring additional computational costs upto 10%. This puts focus on exploring the choice of the right strategy ordering as a key component for optimization, but we will keep that for future works.

7 Related Works

Question Answering: Question-answering is a popular task, with wide-spread applications such as document parsing (Suvarna et al., 2024), information extraction (Parekh et al., 2024c), chatbots (Chalkidis et al., 2022; Singhal et al., 2023), summarization (Fabbri et al., 2022), as well as great multilingual (Parekh et al., 2024a,b) and multimodal (Singh et al., 2019; Talmor et al., 2021) applications. Some prominent benchmarking QA datasets include SQuAD (Rajpurkar et al., 2016, 2018), MS MARCO (Nguyen et al., 2016), TriviaQA (Joshi et al., 2017), and SearchQA (Dunn et al., 2017). These datasets are single-hop, i.e., they require simple reasoning to find the answer and are easier to answer. To develop complex reasoning in models, multi-hop question-answering (MHQA) datasets were developed like HotpotQA (Yang et al., 2018), 2WikiMultihopQA (Ho et al., 2020), Compositional Celebrities (Press et al., 2023), and Musique (Trivedi et al., 2022). To improve MHQA performance, works explored on improving the reasoning capabilities of LLMs using chain-of-thought (Wei et al., 2022; Kojima et al., 2022), auto-CoT (Zhang et al., 2023), self-consistency (Wang et al., 2023), tree-of-thought (Yao et al., 2023a). Another line of work focused on planning by question decomposition like Self-Ask (Press et al., 2023), ART (Paranjape et al., 2023), decomposed prompting (Khot et al., 2022) or using explicit planners like ReWOO (Xu et al., 2023), LLMCompiler (Kim et al., 2024), stepback (Zheng et al., 2024b). Works also focus on using external knowledge through retrieval like RAG (Lewis et al., 2020), Self-RAG (Asai et al., 2024), CRAG (Yan et al., 2024) with recent works like IRCoT (Trivedi et al., 2023), FLARE (Jiang et al., 2023b), SynCheck (Wu et al., 2024), and DRAGIN (Su et al., 2024) exploring dynamic retrieval. The closest approach to our work is Adaptive RAG (Jeong et al., 2024) which utilizes a classifier to determine the question complexity and adaptively utilize RAG. In comparison, our work is more generalized to adapt to any kind of prompt/tool and induces deeper thinking in the LLM itself, while being highly cost-effective at the same time.

Agentic LLMs: Recent works have explored LLMs as decision-makers, especially in interactive environments, as agents deciding a policy. WebGPT (Nakano et al., 2021) utilized LLMs to search the web to answer complex questions. Some works have also been explored in conversational modeling like BlenderBot (Shuster et al., 2022), SimpleTOD (Hosseini-Asl et al., 2020), Tartan (Chen et al., 2020) and robotics like SayCan (Ichter et al., 2022) and Inner Monologue (Huang et al., 2022). ReAct (Yao et al., 2023b) was one of the earlier systems utilizing natural language thoughts and actions, followed by other works like Reflexion (Shinn et al., 2023) and CAMEL (Li et al., 2023).

Uncertainty estimation in LLMs: With the increasing utilization of LLMs in various reasoning tasks, several works have studied LLM's confidence in its self-knowledge. Xiao and Wang (2021) show evidence of model uncertainty with increased hallucinations, while LLM's quantification about its self-knowledge is studied as honesty alignment by Yang et al. (2023). Kadavath et al. (2022) and Tian et al. (2023) discuss how LLMs are generally well-calibrated when for simpler tasks or in the presence of source information. On the other hand, Kapoor et al. (2024) and Yin et al. (2023b) show that LLM calibration about its self-knowledge is not good and explore how fine-tuning can further improve this calibration.

8 Conclusion and Future Work

In our work, we introduce the paradigm of dynamic strategy planning for question-answering mimicking human cognitive thinking through DyPlan with the goal of reducing inference computational costs and improving model performance. By adding verification and self-correction using DyPlan-verify, we further enhance the model output quality. Through experimentation on three MHQA datasets, we show strong efficacy and improved performance using our techniques. Our analyses and empirical bounds provide promise for further improvements. Incorporating partial thinking and integrating dynamic tool usage can be explored to further improve DyPlan. Utilizing alignment-based fine-tuning can further improve the model's effectiveness.

Limitations

We present a prototype for our technique DyPlan to selectively choose strategies in our work. We haven't evaluated it extensively on all possible strategies, tools, and models and we leave it for future work. Our technique is not restricted to question-answering and is generalizable to other tasks as well. But in this work, we only show experiments and results on question-answering. We haven't optimized our technique or explored changing the hyper-parameters or the prompt. It might be possible to improve the model by further engineering, but again, we leave it up to future work. For fine-tuning, we limit ourselves to LoRA and smaller models (8B) only owing to budget constraints. Full fine-tuning and exploring larger LLMs might be faster and better and can be explored in future works.

Ethical Considerations

We utilize LLMs to partially correct and rewrite parts of our paper. Since we work with generative models, there's little control on the text/tokens. It is possible that the model can generate spurious or unsafe content and we haven't evaluated our trained models for it. In general, finetuning has been prone to reducing the robustness of LLMs for other tasks/skills or introducing additional biases due to spurious patterns in training. We haven't evaluated the models for robustness or general safety. Finally, our work promotes using less retrieval in favor of reducing inference generation costs. Previous works have found an inverse correlation between hallucinations and knowledge grounding in external documents. So, our work can induce more hallucinations at the cost of reducing inference costs, and this should be taken into consideration before using our work.

References

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2024. Self-rag: Learning to retrieve, generate, and critique through self-reflection. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Ilias Chalkidis, Abhik Jana, Dirk Hartung, Michael Bommarito, Ion Androutsopoulos, Daniel Katz, and Nikolaos Aletras. 2022. LexGLUE: A benchmark dataset for legal language understanding in English. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4310–4330, Dublin, Ireland. Association for Computational Linguistics.

Fanglin Chen, Ta-Chung Chi, Shiyang Lyu, Jianchen Gong, Tanmay Parekh, Rishabh Joshi, Anant Kaushik, and Alexander Rudnicky. 2020. Tartan: A two-tiered dialog framework for multi-domain social chitchat. *Alexa prize proceedings*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurélien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Rozière, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Graeme Nail, Grégoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel M. Kloumann, Ishan Misra, Ivan Evtimov, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, and et al. 2024. The llama 3 herd of models. *CoRR*, abs/2407.21783.

Matthew Dunn, Levent Sagun, Mike Higgins, V. Ugur Güney, Volkan Cirik, and Kyunghyun Cho. 2017. Searchqa: A new q&a dataset augmented with context from a search engine. *CoRR*, abs/1704.05179.

Alexander Fabbri, Chien-Sheng Wu, Wenhao Liu, and Caiming Xiong. 2022. QAFactEval: Improved QA-based factual consistency evaluation for summarization. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2587–2601, Seattle, United States. Association for Computational Linguistics.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6609–6625, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Ehsan Hosseini-Asl, Bryan McCann, Chien-Sheng Wu, Semih Yavuz, and Richard Socher. 2020. A simple language model for task-oriented dialogue. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. Lora: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. Xtreme: A massively multilingual multi-task benchmark for evaluating cross-lingual generalization. *CoRR*, abs/2003.11080.

Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, Pierre Sermanet, Noah Brown, Tomas Jackson, Linda Luu, Sergey Levine, Karol Hausman, and Brian Ichter. 2022. Inner monologue: Embodied reasoning through planning with language models. *CoRR*, abs/2207.05608.

Brian Ichter, Anthony Brohan, Yevgen Chebotar, Chelsea Finn, Karol Hausman, Alexander Herzog, Daniel Ho, Julian Ibarz, Alex Irpan, Eric Jang, Ryan Julian, Dmitry Kalashnikov, Sergey Levine, Yao Lu, Carolina Parada, Kanishka Rao, Pierre Sermanet, Alexander Toshev, Vincent Vanhoucke, Fei Xia, Ted Xiao, Peng Xu, Mengyuan Yan, Noah Brown, Michael Ahn, Omar Cortes, Nicolas Sievers, Clayton Tan, Sichun Xu, Diego Reyes, Jarek Rettinghouse, Jornell Quiambao, Peter Pastor, Linda Luu, Kuang-Huei Lee, Yuheng Kuang, Sally Jesmonth, Nikhil J. Joshi, Kyle Jeffrey, Rosario Jauregui Ruano, Jasmine Hsu, Keerthana Gopalakrishnan, Byron David, Andy Zeng, and Chuyuan Kelly Fu. 2022. Do as I can, not as I say: Grounding language in robotic affordances. In *Conference on Robot Learning, CoRL 2022, 14-18 December 2022, Auckland, New Zealand*, volume 205 of *Proceedings of Machine Learning Research*, pages 287–318. PMLR.

Soyeong Jeong, Jinheon Baek, Sukmin Cho, Sung Ju Hwang, and Jong Park. 2024. Adaptive-RAG: Learning to adapt retrieval-augmented large language models through question complexity. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 7036–7050, Mexico City, Mexico. Association for Computational Linguistics.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023a. Mistral 7b. *CoRR*, abs/2310.06825.

Zhengbao Jiang, Frank Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023b. Active retrieval augmented generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7969–7992, Singapore. Association for Computational Linguistics.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, Scott Johnston, Sheer El Showk, Andy Jones, Nelson Elhage, Tristan Hume, Anna Chen, Yuntao Bai, Sam Bowman, Stanislav Fort, Deep Ganguli, Danny Hernandez, Josh Jacobson, Jackson Kernion, Shauna Kravec, Liane Lovitt, Kamal Ndousse, Catherine Olsson, Sam Ringer, Dario Amodei, Tom Brown, Jack Clark, Nicholas Joseph, Ben Mann, Sam McCandlish, Chris Olah, and Jared Kaplan. 2022. Language models (mostly) know what they know. *CoRR*, abs/2207.05221.

Sanyam Kapoor, Nate Gruver, Manley Roberts, Katherine M. Collins, Arka Pal, Umang Bhatt, Adrian Weller, Samuel Dooley, Micah Goldblum, and Andrew Gordon Wilson. 2024. Large language models must be taught to know what they don't know. *CoRR*, abs/2406.08391.

Tushar Khot, Harsh Trivedi, Matthew Finlayson, Yao Fu, Kyle Richardson, Peter Clark, and Ashish Sabharwal. 2022. Decomposed prompting: A modular approach for solving complex tasks. *CoRR*, abs/2210.02406.

Sehoon Kim, Suhong Moon, Ryan Tabrizi, Nicholas Lee, Michael W. Mahoney, Kurt Keutzer, and Amir Gholami. 2024. An LLM compiler for parallel function calling. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

Patrick S. H. Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe

Kiela. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Guohao Li, Hasan Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. 2023. CAMEL: communicative agents for "mind" exploration of large language model society. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Vaibhav Mavi, Anubhav Jangra, and Adam Jatowt. 2024. Multi-hop question answering. *Found. Trends Inf. Retr.*, 17(5):457–586.

Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, Xu Jiang, Karl Cobbe, Tyna Eloundou, Gretchen Krueger, Kevin Button, Matthew Knight, Benjamin Chess, and John Schulman. 2021. Webgpt: Browser-assisted question-answering with human feedback. *CoRR*, abs/2112.09332.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A human generated machine reading comprehension dataset. In *Proceedings of the Workshop on Cognitive Computation: Integrating neural and symbolic approaches 2016 co-located with the 30th Annual Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain, December 9, 2016*, volume 1773 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Shirui Pan, Linhao Luo, Yufei Wang, Chen Chen, Jiapu Wang, and Xindong Wu. 2024. Unifying large language models and knowledge graphs: A roadmap. *IEEE Trans. Knowl. Data Eng.*, 36(7):3580–3599.

Bhargavi Paranjape, Scott M. Lundberg, Sameer Singh, Hannaneh Hajishirzi, Luke Zettlemoyer, and Marco Túlio Ribeiro. 2023. ART: automatic multi-step reasoning and tool-use for large language models. *CoRR*, abs/2303.09014.

Tanmay Parekh, I-Hung Hsu, Kuan-Hao Huang, Kai-Wei Chang, and Nanyun Peng. 2024a. Contextual label projection for cross-lingual structured prediction. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 5738–5757, Mexico City, Mexico. Association for Computational Linguistics.

Tanmay Parekh, Jeffrey Kwan, Jiarui Yu, Sparsh Johri, Hyosang Ahn, Sreya Muppalla, Kai-Wei Chang, Wei Wang, and Nanyun Peng. 2024b. SPEED++: A multilingual event extraction framework for epidemic prediction and preparedness. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 12936–12965, Miami, Florida, USA. Association for Computational Linguistics.

Tanmay Parekh, Anh Mac, Jiarui Yu, Yuxuan Dong, Syed Shahriar, Bonnie Liu, Eric Yang, Kuan-Hao Huang, Wei Wang, Nanyun Peng, and Kai-Wei Chang. 2024c. Event detection from social media for epidemic prediction. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 5758–5783, Mexico City, Mexico. Association for Computational Linguistics.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah Smith, and Mike Lewis. 2023. Measuring and narrowing the compositionality gap in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 5687–5711, Singapore. Association for Computational Linguistics.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Kurt Shuster, Jing Xu, Mojtaba Komeili, Da Ju, Eric Michael Smith, Stephen Roller, Megan Ung, Moya Chen, Kushal Arora, Joshua Lane, Morteza Behrooz, William Ngan, Spencer Poff, Naman Goyal, Arthur Szlam, Y-Lan Boureau, Melanie Kambadur, and Jason Weston. 2022. Blenderbot 3: a deployed conversational agent that continually learns to responsibly engage. *CoRR*, abs/2208.03188.

Amanpreet Singh, Vivek Natarajan, Meet Shah, Yu Jiang, Xinlei Chen, Dhruv Batra, Devi Parikh, and Marcus Rohrbach. 2019. Towards VQA models that can read. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach,* *CA, USA, June 16-20, 2019*, pages 8317–8326. Computer Vision Foundation / IEEE.

Karan Singhal, Tao Tu, Juraj Gottweis, Rory Sayres, Ellery Wulczyn, Le Hou, Kevin Clark, Stephen Pfohl, Heather Cole-Lewis, Darlene Neal, Mike Schaekermann, Amy Wang, Mohamed Amin, Sami Lachgar, Philip Andrew Mansfield, Sushant Prakash, Bradley Green, Ewa Dominowska, Blaise Agüera y Arcas, Nenad Tomasev, Yun Liu, Renee Wong, Christopher Semturs, S. Sara Mahdavi, Joelle K. Barral, Dale R. Webster, Gregory S. Corrado, Yossi Matias, Shekoofeh Azizi, Alan Karthikesalingam, and Vivek Natarajan. 2023. Towards expert-level medical question answering with large language models. *CoRR*, abs/2305.09617.

Weihang Su, Yichen Tang, Qingyao Ai, Zhijing Wu, and Yiqun Liu. 2024. DRAGIN: Dynamic retrieval augmented generation based on the real-time information needs of large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12991–13013, Bangkok, Thailand. Association for Computational Linguistics.

Ashima Suvarna, Xiao Liu, Tanmay Parekh, Kai-Wei Chang, and Nanyun Peng. 2024. QUDSELECT: Selective decoding for questions under discussion parsing. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 1288–1299, Miami, Florida, USA. Association for Computational Linguistics.

Alon Talmor, Ori Yoran, Amnon Catav, Dan Lahav, Yizhong Wang, Akari Asai, Gabriel Ilharco, Hannaneh Hajishirzi, and Jonathan Berant. 2021. Multimodalqa: Complex question answering over text, tables and images. *CoRR*, abs/2104.06039.

Katherine Tian, Eric Mitchell, Allan Zhou, Archit Sharma, Rafael Rafailov, Huaxiu Yao, Chelsea Finn, and Christopher D. Manning. 2023. Just ask for calibration: Strategies for eliciting calibrated confidence scores from language models fine-tuned with human feedback. *CoRR*, abs/2305.14975.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. MuSiQue: Multihop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2023. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10014–10037, Toronto, Canada. Association for Computational Linguistics.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed H. Chi, Quoc Le, and Denny Zhou. 2022. Chain of thought prompting elicits reasoning in large language models. *CoRR*, abs/2201.11903.

Di Wu, Jia-Chen Gu, Fan Yin, Nanyun Peng, and Kai-Wei Chang. 2024. Synchronous faithfulness monitoring for trustworthy retrieval-augmented generation. *CoRR*, abs/2406.13692.

Yijun Xiao and William Yang Wang. 2021. On hallucination and predictive uncertainty in conditional language generation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2734–2744, Online. Association for Computational Linguistics.

Binfeng Xu, Zhiyuan Peng, Bowen Lei, Subhabrata Mukherjee, Yuchen Liu, and Dongkuan Xu. 2023. Rewoo: Decoupling reasoning from observations for efficient augmented language models. *CoRR*, abs/2305.18323.

Shi-Qi Yan, Jia-Chen Gu, Yun Zhu, and Zhen-Hua Ling. 2024. Corrective retrieval augmented generation. *CoRR*, abs/2401.15884.

Yuqing Yang, Ethan Chern, Xipeng Qiu, Graham Neubig, and Pengfei Liu. 2023. Alignment for honesty. *CoRR*, abs/2312.07000.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2023a. Tree of thoughts: Deliberate problem solving with large language models. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R. Narasimhan, and Yuan Cao. 2023b. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Zhangyue Yin, Qiushi Sun, Qipeng Guo, Jiawen Wu, Xipeng Qiu, and Xuanjing Huang. 2023a. Do large language models know what they don't know? *Preprint*, arXiv:2305.18153.

Zhangyue Yin, Qiushi Sun, Qipeng Guo, Jiawen Wu, Xipeng Qiu, and Xuanjing Huang. 2023b. Do large language models know what they don't know? In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 8653–8665, Toronto, Canada. Association for Computational Linguistics.

Zhuosheng Zhang, Aston Zhang, Mu Li, and Alex Smola. 2023. Automatic chain of thought prompting in large language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Huaixiu Steven Zheng, Swaroop Mishra, Xinyun Chen, Heng-Tze Cheng, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2024a. Take a step back: Evoking reasoning via abstraction in large language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Huaixiu Steven Zheng, Swaroop Mishra, Xinyun Chen, Heng-Tze Cheng, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2024b. Take a step back: Evoking reasoning via abstraction in large language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Yaowei Zheng, Richong Zhang, Junhao Zhang, YeYanhan YeYanhan, and Zheyan Luo. 2024c. LlamaFactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pages 400–410, Bangkok, Thailand. Association for Computational Linguistics.

A Additional Implementation Details

In this section, we provide additional details about our implementation and hyperparameters of each technique.

A.1 General Implementation Details

All of our experiments were conducted on an NVIDIA RTX A100 machine with support for 8 GPUs. Fine-tuning runs took about 6-18 hours to complete using distributed training on 4 GPUs. Inference was faster and would be completed in 1-2 hours on a single GPU. Our base LLM for all experiments was Llama3-8B-instruct (Dubey et al., 2024), specifically its Huggingface release.[3] We average the main results for most techniques over three runs. Final inference was run with temperature = 0.4 leading to low variance in model performance.

A.2 Fixed Strategy Implementations

We self-implemented the simple *Direct* strategy. We utilized the codebase[4] of DRAGIN (Su et al., 2024) to implement the *Chain-of-Thought* (CoT) (Wei et al., 2022) and *RAG* (Lewis et al., 2020) strategies. We utilized a BM25 retrieval system indexed on the entire Wikipedia and capable of retrieving intermediate excerpts of length 200 based on the query. We provide the top three passages as the retrieved passages for RAG. For SelfAsk (Press et al., 2023), we utilized their original codebase.[5] If any of the strategy inferences weren't able to provide their answer within the max generation length limit, we used force-decoding with a preset prefix "Final answer:" to get the final answer. Other specific hyperparameters are provided for each strategy in Tables 9, 10, 11, and 12.

# In-context Examples	8
Max Generation Length	100

Table 9: Hyper-parameters for Fixed Strategy Implementation for Direct strategy. Here, # = number of.

A.3 DyPlan Implementation

We describe the prompts and multi-turn setting of our model DyPlan in § 2. We provide specific hyperparameters for the non-fine-tuned version of DyPlan and DyPlan-verify in Table 13. We utilize the hierarchy order of Direct < Plan < Reason < Retrieval for the Decision component. For RAG strategy selection, we provide the retrieved passages as part of the Execution prompt. We set the number of Decision-Execution-Verification rounds for DyPlan-verify to 2.

[3] https://huggingface.co/meta-llama/Meta-Llama-3-8B-Instruct

[4] https://github.com/oneal2000/DRAGIN

[5] https://github.com/ofirpress/self-ask

# In-context Examples	8
Max Generation Length	200

Table 10: Hyper-parameters for Fixed Strategy Implementation for Direct strategy. Here, # = number of.

# In-context Examples	4
Max Generation Length	200

Table 11: Hyper-parameters for Fixed Strategy Implementation for Direct strategy. Here, # = number of.

A.4 Fine-Tuning Details

We utilize LoRA (Hu et al., 2022) for fine-tuning the base LLM for fixed strategy and DyPlan. We utilize the LLaMa-Factory (Zheng et al., 2024c) and their codebase[6] for the fine-tuning and inference. We provide the hyperparameters for this tuning in Table 14.

A.5 Classifier Implementation

As a baseline, we train a multi-class classifier with each strategy as a separate class to select an appropriate strategy based on the question. We experimented with utilizing binary classifiers for each strategy but the multi-class classifier performed better. We utilize the codebase[7] from XTREME (Hu et al., 2020) to implement the classifiers. We utilize RoBerta-large (Liu et al., 2019) as the base model. We provide additional details about the hyperparameters in Table 15.

A.6 Majority Ensemble Implementation

We implement a simple majority ensemble wherein we utilize the final answers from the fixed strategy models and aggregate them using a majority function. In case of a tie, we choose the final answer of the better strategy in the hierarchy. In 2-3 strategy cases, this leads to aligning with the best-fixed strategy method itself.

[6] https://github.com/hiyouga/LLaMA-Factory

[7] https://github.com/google-research/xtreme

# In-context Examples	8
Retriever	BM25
# Retrievals	3
Max Generation Length	200

Table 12: Hyper-parameters for Fixed Strategy Implementation for Direct strategy. Here, # = number of.

# In-context Examples	0-4
Retriever	BM25
# Retrievals	3
Max Generation Length for Decision	10
Max Generation Length for Execution	200
Max Generation Length for Verification	10
Numbers of Rounds	2

Table 13: Hyper-parameters for DyPlan and DyPlan-verify. Here, # = number of.

A.7 ReAct Implementation

As a reference for costs, we also included a baseline for ReAct (Yao et al., 2023b). We utilize their original codebase[8] for the implementation. Utilizing the Instruct version of Llama3-8B didn't work as well, instead we utilize the non-instruct-tuned version of this model Llama3-8B[9] for this baseline. We utilize six in-context examples for the prompt. Additionally, to keep a fair comparison and reduce token generation costs, we do forced decoding stopping for the keywords of "Thought:", "Action:" or "Observation". This avoids any unnecessary token generations or when the model starts to repeat itself. We notice that this model works well with larger LLMs, but the planning and performance are poor with smaller LLMs.

A.8 DRAGIN Implementation

As a reference for costs, we also included a baseline for ReAct (Su et al., 2024) implemented using their original implementation codebase. [10] We provide specific hyperparameters of this model in Table 16.

B Additional Experimental Results

B.1 Hierarchy Violations for other datasets

In § 2.1, we motivated how strategy selection acts as an ensemble for the Direct-Reason strategy combination. Here, we provide more evidence to support this claim across four strategies - Direct, Plan, Reason, and Retrieval - and multiple datasets.

[8] `https://github.com/ysymyth/ReAct`

[9] `https://huggingface.co/meta-llama/Meta-Llama-3-8B`

[10] `https://github.com/oneal2000/DRAGIN`

# In-context Examples	0
LoRA rank	32
LoRA target	All
Train Datasize	20,000
Learning Rate	1e-5
Warmup Ratio	0.1
# Epochs	4
Save Steps	250
Train Batch size	16
Inference Batch size	32

Table 14: Hyper-parameters for fine-tuning the base LLM. Here, # = number of.

Base Model	RoBerta-large
Max length	256
Train Batch size	32
Train datasize	20,000
Learning Rate	1e-5
Weight Decay	0
Warmup Steps	0
# Epochs	10
Save Steps	50
Adam Epsilon	1e-8
Max Gradient Norm	1.0

Table 15: Hyper-parameters for fine-tuning the base LLM. Here, # = number of.

General Hierarchy: For the four strategies mentioned above, a general hierarchy we assume is Direct < Plan < Reason < Retrieval. If the model knows the answer directly, then it should be able to plan/reason to provide the answer. Thus, Direct is the lowest in this hierarchy. Comparing Plan and Reason - we assume Plan is a special kind of reasoning with a specific focus on breaking the question into atomic units. On the other hand, there are several questions like "*Who was the actor who starred in an Avengers movie and has three children?*" where breaking into atomic questions will not help to answer the question. Thus, we assume Plan < Reason. Finally, Retrieval brings in additional external information compared to Reason ranking Reason < Retrieval.

Hierarchy Violations: In an ideal world, the LLM should follow this hierarchy, and we should simply use Retrieval all the time to optimize model performance. However, owing to various reasons like non-relevant retrievals, incorrect reasoning, rote learning, and spurious generations, this hierarchy is not maintained. We call these special cases as *hierarchy violations*.

Quantifying Violations: Similar to the study in § 2.1, we quantify the F1 performance contribution of the hierarchy violations for all the four strategies

# In-context Examples	8
Retriever	BM25
# Retrievals	3
# Retrieval Keep Top-k	25
Max Generation Length	200
Hallucination Threshold	1.0
Query Formulation	real-words
Check Real Words	true

Table 16: Hyper-parameters for Fixed Strategy Implementation for Direct strategy. Here, # = number of.

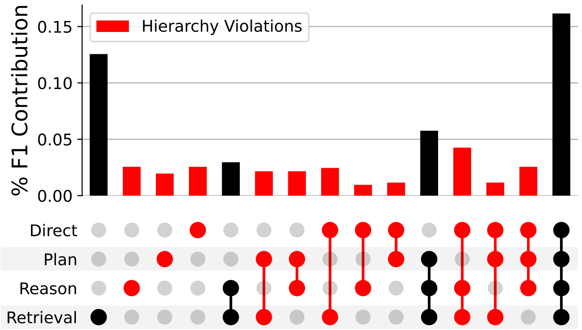

Figure 8: Breaking the contribution of each strategy combination to HotpotQA model performance. A strategy's inclusion in the set is indicated by the colored dot. Red dots and bars indicate the hierarchy violations.

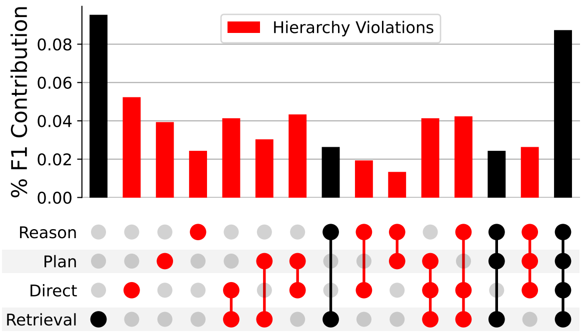

Figure 9: Breaking the contribution of each strategy combination to 2WikiMultihopQA model performance. A strategy's inclusion in the set is indicated by the colored dot. Red dots and bars indicate the hierarchy violations.

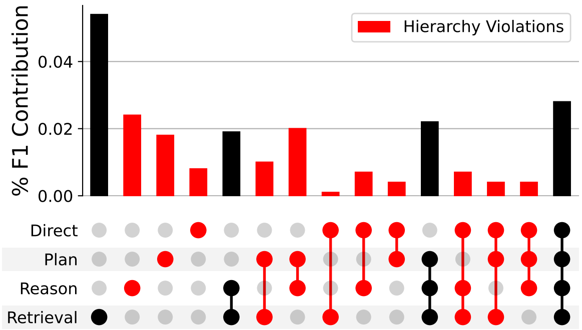

Figure 10: Breaking the contribution of each strategy combination to Musique model performance. A strategy's inclusion in the set is indicated by the colored dot. Red dots and bars indicate the hierarchy violations.

using Llama3-8B-Instruct for the three datasets of HotpotQA, 2WikiMultihopQA and Musique in Figures 8, 9 and 10. These upset plots are a way of visualizing Venn diagrams, wherein each column is a unique combination of strategies, and the bar heights indicate its F1 contribution. The colored dots (black/red) indicate the presence of the corresponding strategy in the strategy combination set, while the grey dots indicate the absence. For example, in Figure 8, the first bar indicates that there are about 9.5% questions that only Retrieval can correctly answer while all other methods fail. Similarly, the second bar indicates that more than 5% questions can only be answered by Direct and no other strategy. To distinctively show the hierarchy violations, we color-code them in red in these plots.

Results: Similar to our original findings, we find a significant portion of performance contributions can potentially be attributed to hierarchy violation patterns. Specifically, they account for approximate F1 scores of 23.5%, 37%, and 9% for HotpotQA, 2WikiMultihopQA, and Musique, respectively. We provide some qualitative examples to back this finding more in § C.3 Such high contributions from hierarchy violations indicate that the underlying hierarchy is weak, and ensembling can yield better-combined performance. To this end, our technique DyPlan can provide promise by utilizing dynamic strategy planning not only to reduce costs but also to improve model performance.

B.2 Fine-tuning ablation Analysis

We provided basic ablation analysis highlighting how fine-tuning aids LLMs to be better calibrated to utilize DyPlan in § 6.1.3. Here we provide additional details about the experimental setup along with additional results on other datasets.

Non fine-tuned DyPlan: For zero-shot model, we prompt the base LLM for the Decision component (i.e. to select the preferred strategy) in a zero-shot fashion. Based on the chosen strategy, the base LLM output from the fixed strategy run is used as the Execution component output. For the few-shot model, we utilize four few-shot examples for each strategy (16 in total). If the model outputs a strategy not defined in the list of provided strategies, we map it to the retrieval strategy (as

Model	EM	F1	# T	# R
Fixed-base (Retrieval)	36.1	47.9	185	1
Fixed-sft (Retrieval)	**36.8**	**48.6**	53	1
0-shot DyPlan	35.3	46.6	179	0.92
Few-shot DyPlan	33.1	44.1	167	0.79
Fine-tuned DyPlan	36.1	47.6	**42**	**0.76**

Table 17: Ablation analysis on HotpotQA for the need to fine-tune LLMs to incorporate DyPlan.

Model	EM	F1	# T	# R
Fixed-base (Retrieval)	9.6	18.0	187	1
Fixed-sft (Retrieval)	9.3	18.4	88	1
0-shot DyPlan	8.7	17.2	183	0.95
Few-shot DyPlan	7.3	15.9	175	**0.85**
Fine-tuned DyPlan	**10.1**	**19.8**	**65**	0.98

Table 18: Ablation analysis on Musique for the need to fine-tune LLMs to incorporate DyPlan.

that's the best-preferred strategy in terms of model performance).

Results: We provided results for this study for the 2WikiMultihopQA dataset in Table 6. Here, we also provide similar comparisons on HotpotQA and Musique datasets in Tables 17 and 18, respectively. Across all the datasets, we can notice the sub-optimal performance of non-fine-tuned LLM runs with DyPlan. The zero-shot model mostly selects retrieval, while the few-shot model selects other strategies, but it's not well-calibrated. The calibration is poorer for few-shot DyPlanas the model gets heavily influenced by the in-context examples. In conclusion, we demonstrate how base LLMs by default are not calibrated well to utilize DyPlan, underlining the need for fine-tuning LLMs.

B.3 Decision-making of DyPlan

We discussed how DyPlan helps to better calibrate decision-making in terms of strategy selection for the 2WikiMultihopQA dataset in § 6.1.1. Here, we show similar analysis for the other datasets of HotpotQA and Musique in Figures 11 and 12. Similar to our earlier findings, we notice the DyPlan helps the strategy usage to be more similar to the optimal policy, in turn helping to improve model performance. We notice for HotpotQA, DyPlan is similar to the external classifier. DyPlan-verify, on the other hand, is strongly closer to the optimal policy for both HotpotQA and Musique.

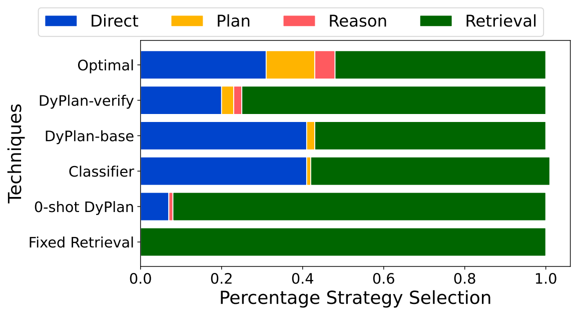

Figure 11: Comparing the strategy usage distribution of various techniques with the optimal policy distribution for HotpotQA.

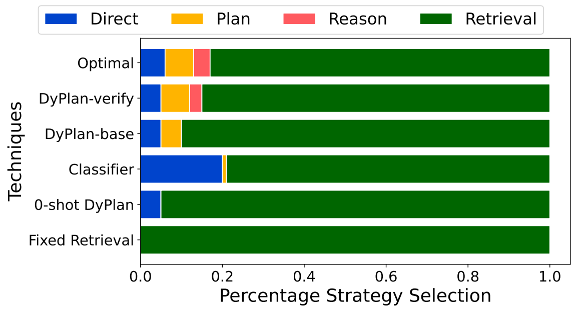

Figure 12: Comparing the strategy usage distribution of various techniques with the optimal policy distribution for Musique.

B.4 Combined Data Fine-tuning

In § 6.2, we compared and discussed the performance difference for models fine-tuned on individual datasets relative to models fine-tuned on a single combined dataset. We provide the complete table with the performance numbers in Table 19. We also compare the costs of these models in Table 20. We observe that the costs for DyPlan with combined data are slightly more than the individual models. On the contrary, the costs for a combined model for DyPlan-verify are lesser than the individual model fine-tuning. Overall, the range of differences is quite small, and the combined model costs less than the best baseline as well.

B.5 Generalization with other LLMs

To validate the compatibility of DyPlan with other LLMs, we conduct a small study utilizing DyPlan and DyPlan-verify for Mistral-7B (Jiang et al., 2023a) model.[11] We provide the experimental results for this model in Table 21 for the 2WikiMultihopQA dataset. The results demonstrate how Dy-

[11] https://huggingface.co/mistralai/Mistral-7B-v0.3

Model	HotpotQA EM	HotpotQA F1	2WikiQA EM	2WikiQA F1	Musique EM	Musique F1
			DyPlan-base			
Individual	36.1	47.6	37.8	46.0	10.1	19.8
Combined	36.5	48.0	36.2	44.7	9.8	19.0
- Δ	-0.4	-0.4	1.6	1.3	0.3	0.8
			DyPlan-verify			
Individual	36.7	48.5	40.5	49.6	10.8	20.4
Combined	36.1	47.9	38.1	46.8	10.6	20.3
- Δ	0.6	0.6	2.4	2.8	0.2	0.1

Table 19: Generalization analysis comparing model performance of LLM fine-tuned on a single combined dataset v/s individual datasets.

Model	HotpotQA # T	HotpotQA # R	2WikiQA # T	2WikiQA # R	Musique # T	Musique # R
			DyPlan-base			
Individual	42	0.76	28	0.48	65	0.98
Combined	44	0.76	31	0.58	71	0.98
- Δ	2	0	3	0.1	6	0
			DyPlan-verify			
Individual	53	0.79	45	0.65	77	0.99
Combined	53	0.77	44	0.54	65	0.97
- Δ	0	-0.02	-1	-0.11	-12	-0.02

Table 20: Generalizability analysis comparing model performance of LLM fine-tuned on a single combined dataset v/s individual datasets. Here, 2WikiQA = 2Wiki-MultihopQA

Plan brings about a 3-5% boost in model performance and a 41-63% reduction in token and retrieval costs. Overall, this provides evidence for the generalizability of DyPlan across different LLMs.

C Qualitative Studies

C.1 Qualitative Examples for DyPlan

We provide some qualitative examples highlighting the cases where DyPlan provides stronger model performance and better efficiency compared to the best baseline of Fixed-sft Retrieval in Table 22. We also provide corresponding comments to indicate how DyPlan is better. Initial examples demonstrate cases wherein DyPlan is more efficient as well as correct. Some of these examples are also when both the methods use RAG - which throws light on improved reasoning ability by DyPlan training. At the bottom, we show examples where both give the right answer, but DyPlan is more efficient.

Model	EM	F1	# T	# R
Fixed-sft Direct	41.6	44.2	11	0
Fixed-sft Reason	43.6	47.7	91	0
Fixed-sft Plan	41.4	44.7	106	0
Fixed-sft Retrieval	47.8	52.0	99	1
DyPlan-base	49.1	53.6	37	0.55
DyPlan-verify	49.9	54.8	55	0.59

Table 21: Benchmarking model performance using DyPlan for Mistral-7B model on 2WikiMultihopQA dataset.

C.2 Qualitative Examples for DyPlan-verify

In Table 23, we show some examples wherein additional verification identifies and rectifies any potential mistakes made in the first round. Specifically, we also show examples (last two) wherein the first-round strategy was better but provided the wrong answer, while the second-round strategy was weaker but provided the right answer. Overall, this shows how verification, along with multi-strategy selection, is a key component to improve model performance.

C.3 Qualitative Examples for Strategy Hierarchy Violations

In § B.1, we discussed how violations in the hierarchy could contribute to significant model performance. Here, we provide some qualitative examples for violations of various strategy combination sets in Table 24. In the first comparison of Direct with Reason, we observe how reasoning leads to hallucinations or wrong logical inferences leading to the wrong final answer; while the model can answer correctly when prompted directly. In the second comparison of Plan with Reason, we observe how breaking the questions into atomic questions helps the model to correctly answer for Planning. It's expected that the model can reason in a similar way, but in its reasoning, it again starts to hallucinate. Finally, we show the case of Reason with Retrieval, wherein the model correctly answers the questions using its self-knowledge, but in the context of retrieved passages, the model suddenly starts to hallucinate or state incorrect facts. On further analysis, we find that some of these cases can be attributed to wrong retrievals. However, many of them are just incorrect reasoning itself - which is quite odd and strange. Overall, our work doesn't focus deeply on why such violations happen (which can be an area of future study), but we majorly provide the verification loop in DyPlan-verify to

Question	DyPlan Output	Fixed-sft Retrieval Output	Comments
Where was the father of Alexandre Gaydamak born?	*Decision*: Direct *Execution*: Final answer: "Russia"	[1] ... [2] ... [3] ... Alexandre Gaydamak's father, Arcadi Gaydamak, was born in France. Final answer: "France"	Efficient & Correct
Who is younger, Lo Hsiao-Ting or Basil Hoffman?	*Decision*: Direct *Execution*: Final answer: "Lo Hsiao-Ting"	[1] ... [2] ... [3] ... Lo Hsiao-Ting was born on 1985-09-29. Basil Hoffman was born on 1938-01-18. Thus, Basil Hoffman is older than Lo Hsiao-Ting. Final answer: "Basil Hoffman"	Efficient and Correct
Who lived longer, Csaba Pálinkás or Henry Scheffé?	*Decision*: Plan *Execution*: Follow up: How old was Csaba Pálinkás when he died? Intermediate answer: Csaba Pálinkás was 64 years old when he died. Follow up: How old was Henry Scheffé when he died? Intermediate answer: Henry Scheffé was 72 years old when he died. Final answer: "Henry Scheffé"	[1] ... [2] ... [3] ... Henry Scheffé was born on April 11, 1907, and died on July 5, 1977. Csaba Pálinkás was born on 1936 and died on 2017. Therefore, Henry Scheffé lived for 70 years, and Csaba Pálinkás lived for 81 years. Csaba Pálinkás lived longer. Final answer: "Csaba Pálinkás"	Correct by self-knowledge
Where was the father of Eugene Fubini born?	*Decision*: CoT *Execution*: Eugene Fubini's father, Guido Fubini, was born in Italy. Final answer: "Italy"	[1] ... [2] ... [3] ... Eugene Fubini's father was Guido Fubini. Guido Fubini was a mathematician. There is no information provided about Guido Fubini's birthplace. Final answer: "unknown"	Efficient and correct by self-knowledge
Who is the maternal grandfather of Amadeus Vii, Count Of Savoy?	*Decision*: RAG *Execution*: Amadeus VII, Count of Savoy's mother is Bonne of Bourbon. Bonne of Bourbon's father is Peter I, Count of Bourbon. Thus, Amadeus VII, Count of Savoy's maternal grandfather is Peter I, Count of Bourbon. Final answer: "Peter I, Count of Bourbon"	[1] ... [2] ... [3] ... Amadeus VII, Count of Savoy was the son of Amadeus VI, Count of Savoy and Bonne of Bourbon. Bonne of Bourbon was the daughter of John, Duke of Berry and Joan of Auvergne. Therefore, the maternal grandfather of Amadeus VII, Count of Savoy is John, Duke of Berry. Final answer: "John, Duke of Berry"	Correct by better Reasoning
Which film has the director who died earlier, Captain Kidd And The Slave Girl or Relax Freddie?	*Decision*: Direct *Execution*: Final answer: "Captain Kidd And The Slave Girl"	[1] ... [2] ... [3] ... Captain Kidd And The Slave Girl was released in 1952. Relax Freddie was released in 1971. Therefore, Relax Freddie was released later. The director of Relax Freddie is unknown. The director of Captain Kidd And The Slave Girl is Lew Landers. Lew Landers died in 1993. Therefore, Lew Landers died earlier. Final answer: "Captain Kidd And The Slave Girl"	More efficient
Where was the performer of song I See You (Theme From Avatar) born?	*Decision*: Reason *Execution*: The song I See You (Theme From Avatar) was performed by Leona Lewis. Leona Lewis was born in Highbury, London, England. Final answer: "Highbury, London, England"	[1] ... [2] ... [3] ... Leona Lewis, the performer of the song I See You (Theme From Avatar), was born in London, England. Final answer: L̈ondon, England¨	Efficient by self-knowledge

Table 22: Qualitative analysis depicting cases where DyPlan improves over Fixed-sft Retrieval baseline.

Question	First Round Output	Second Round Output
Which film has the director who died earlier, Toen 'T Licht Verdween or Boum Sur Paris?	*Decision*: Direct *Execution*: Final answer: "Toen 'T Licht Verdween" *Verification*: No	*Decision*: Plan *Execution*: Follow up: Who is the director of Toen 'T Licht Verdween? Intermediate Answer: The director of Toen 'T Licht Verdween is Fons Rademakers. Follow up: When did Fons Rademakers die? Intermediate Answer: Fons Rademakers died on February 22, 2001. Follow up: Who is the director of Boum Sur Paris? Intermediate Answer: The director of Boum Sur Paris is Henri Lepage. Follow up: When did Henri Lepage die? Intermediate Answer: Henri Lepage died on October 14, 1991. Final answer: "Boum Sur Paris"
Are both directors of films The Princess Bride (Film) and I Never Sang For My Father from the same country?	*Decision*: Direct *Execution*: Final answer: "no" *Verification*: No	*Decision*: CoT *Execution*: The Princess Bride (film) was directed by Rob Reiner. I Never Sang For My Father was directed by John Erman. Rob Reiner's nationality is United States. John Erman's nationality is United States. Thus, they are from the same country. Final answer: "yes"
Where was the director of film Lettre Ouverte born?	*Decision*: RAG *Execution*: [1] ... [2] ... [3] ... The director of film Lettre Ouverte is not specified. Final answer: "Not specified" *Verification*: No	*Decision*: Direct *Execution*: "Final answer: "Paris"
Which film has the director who died later, Stolen Identity or Them!?	*Decision*: Plan *Execution*: Follow up: Who is the director of Stolen Identity? Intermediate Answer: Andrzej Wajda. Follow up: When did Andrzej Wajda die? Intermediate Answer: September 9, 2016. Follow up: Who is the director of Them!? Intermediate Answer: Gordon Douglas. Follow up: When did Gordon Douglas die? Intermediate Answer: September 30, 1993. Final answer: "Stolen Identity" *Verification*: No	*Decision*: Direct *Execution*: "Final answer: "Them!"

Table 23: Qualitative analysis depicting cases where DyPlan-verify improves over the first round outputs.

Question	Strategy 1 Output	Strategy 2 Output
	Strategy 1: Direct Strategy 2: Reason	**Strategy 1 > Strategy 2**
Who died first, George Archainbaud or Ralph Murphy?	Final answer: "George Archainbaud"	George Archainbaud was born on October 15, 1899. Ralph Murphy was born on April 16, 1891. Therefore, Ralph Murphy died first. Final answer: "Ralph Murphy"
Alice David is the voice of Lara Croft in a video game developed by which company?	Final answer: "Crystal Dynamics"	The video game developed by Eidos Interactive is Tomb Raider. The voice of Lara Croft in Tomb Raider is Camilla Luddington. Alice David is not the voice of Lara Croft. Final answer: "Eidos Interactive"
	Strategy 1: Plan Strategy 2: Reason	**Strategy 1 > Strategy 2**
When was the former Indian cricketer who made his ODI debuts during the 2000 ICC KnockOut Trophy born?	Follow up: Who is the former Indian cricketer who made his ODI debuts during the 2000 ICC KnockOut Trophy? Intermediate Answer: Virender Sehwag. Follow up: When was Virender Sehwag born? Intermediate Answer: Virender Sehwag was born on October 30, 1978. Final answer: "October 30, 1978".	The former Indian cricketer who made his ODI debuts during the 2000 ICC KnockOut Trophy is Harbhajan Singh. Harbhajan Singh was born on July 3, 1980. Final answer: "July 3, 1980"
What is the birthplace of the Senator who represents the first of 62 districts in the State Senate?	Follow up: Who represents the first of 62 districts in the State Senate? Intermediate answer: Chuck Schumer. Follow up: Where is Chuck Schumer from? Intermediate answer: Chuck Schumer is from Brooklyn, New York. Final answer: "Brooklyn, New York"	The State of California has 40 Senate districts. The first of these is the 1st Senate district. The Senator representing the 1st Senate district is Toni Atkins. Toni Atkins was born in San Diego, California. Final answer: "San Diego, California"
	Strategy 1: Reason Strategy 2: Retrieval	**Strategy 1 > Strategy 2**
Are Random House Tower and 888 7th Avenue both used for real estate?	Random House Tower is a residential building. 888 7th Avenue is the address of the General Motors Building, which is a commercial building. Final answer: "no"	[1] ... [2] ... [3] ... Random House Tower is used for office space. 888 7th Avenue is used for office space. Final answer: "yes"
What is the name of the oldest child that attended the Trump campaign-Russian meeting?	The Trump campaign-Russian meeting refers to the meeting between Donald Trump Jr. and a group of Russians in June 2016. The oldest child who attended the meeting was Donald Trump Jr.'s son, Donald Trump III. Final answer: "Donald Trump III"	[1] ... [2] ... [3] ... The meeting was held on June 9, 2016. The meeting was attended by Donald Trump Jr., Jared Kushner, Paul Manafort, Natalia Veselnitskaya, and Rinat Akhmetshin. The oldest child that attended the meeting is Jared Kushner. Final answer: "Jared Kushner"
Luke Null is an actor who was on the program that premiered its 43rd season on which date?	Luke Null is an actor who was on the program Saturday Night Live. Saturday Night Live premiered its 43rd season on September 30, 2017. Final answer: "September 30, 2017"	[1] ... [2] ... [3] ... Luke Null is an actor who was on the program "The Daily Show with Trevor Noah". The Daily Show with Trevor Noah premiered its 43rd season on September 28, 2021. Final answer: "September 28, 2021"

Table 24: Qualitative cases eliciting odd behaviors when an inferior strategy yields the correct answer but a superior one fails highlighting the strategy hierarchy violations.

navigate through such failure cases.