# Dipper: <u>Di</u>versity in <u>P</u>rompts for <u>P</u>roducing Large Language Model <u>E</u>nsembles in <u>R</u>easoning Tasks

**Wenyang Hu**[*1,2], **Gregory Kang Ruey Lau**[* 1,3], **Diwen Liu**[1], **Jizhuo Chen**[1],
**See-Kiong Ng**[1], **Bryan Kian Hsiang Low**[1]
[1]National University of Singapore, [2]SAP,
[3]CNRS@CREATE, 1 Create Way, #08-01 Create Tower, Singapore 138602
{wenyang,greglau,lowkh}@comp.nus.edu.sg, seekiong@nus.edu.sg

## Abstract

Large Language Models (LLMs), particularly smaller variants, still struggle with complex reasoning tasks. While inference-time prompting can guide reasoning, existing methods often rely on sequential queries. Ensemble approaches offer a promising path to performance gains, especially given recent batch inference speed-ups. This work introduces DIPPER, a novel, training-free framework that transforms a single LLM into an effective inference-time ensemble. By feeding the model an optimized and diverse set of prompts in parallel, DIPPER elicits varied reasoning paths, leading to performance gains. We empirically demonstrate significant improvements on mathematical reasoning benchmarks, such as MATH, where a DIPPER ensemble of three Qwen2-MATH-1.5B instances (via parallel prompting of a single model) outperforms a larger 7B model.

## 1 Introduction

Despite remarkable advancements, Large Language Models (LLMs), particularly smaller models that are often constrained by resource limitations (e.g., GPU memory), continue to struggle with complex reasoning tasks (Huang and Chang, 2023). While inference-time methods offers a promising approach for enhancing LLM performance, especially for these smaller models (Snell et al., 2024), existing methods like Chain-of-Thought (CoT) and Reflexion often rely on *sequential* LLM queries, thereby incurring additional latency costs (Qiao et al., 2023; Zheng et al., 2023; Yao et al., 2023).

Ensemble methods, which involve the use of multiple constituent models in *parallel*, have been shown to improve models' performance and robustness in classical machine-learning settings (Ganaie et al., 2022) and are promising approaches to achieve better inference-time performance, although less well-studied in the LLM setting. The

prospects of applying such methods to LLMs are increasingly attractive, given recent developments that have enabled significant speed-ups in parallel, LLM batch inference. These include methods to efficiently handle key-value cache memory (Kwon et al., 2023) and prompt caching to efficiently reuse common prompts for multiple queries (Zhu et al., 2024; Gim et al., 2024), enabling sub-linear (in the number of queries) costs for batch inference. However, a key challenge for successful ensembles is the **diversity** among their constituents (Krogh and Vedelsby, 1994; Zaidi et al., 2020). This principle extends to LLM ensembles, where achieving meaningful diversity from a single base model remains a central challenge.

Current approaches injecting such diversity, such as using heterogeneous model types (i.e., different LLMs) (Jiang et al., 2023; Huang et al., 2024), are often impractical due to memory constraints or use preferences for a single model type. Alternatively, methods like self-consistency, which rely on stochastic sampling from the same prompt (Wang et al., 2023b), typically yield limited diversity, thereby capping potential performance gains. We identify an influential yet overlooked source of diversity: **system prompts**. LLMs can generate varied reasoning pathways and outputs for the same task when guided by different instructional prompts (Kojima et al., 2023). This observation motivates our central research question: *How can we systematically leverage prompt diversity to construct high-performing LLM ensembles from a single base model efficiently, without model retraining?*

To address this, we introduce DIPPER, a novel, training-free LLM ensemble framework that constructs an LLM ensemble by feeding a single base LLM an optimized, diverse set of reasoning prompts in parallel. This approach harnesses the parallel processing capabilities of modern LLM inference systems to achieve significant performance improvements in reasoning tasks, particularly for

---

[*]Equal contribution.

resource-constrained models. DIPPER is notably simple, resource-efficient, and readily applicable to any black-box LLM via API access. Our key contributions are summarized as follows:

- We propose DIPPER, a novel framework for constructing inference-time ensembles from a single LLM using diverse reasoning prompts, adaptable to any (including black-box) LLM, and detail its core design principles (Sec. 4).

- We develop a training-free, theory-inspired prompt diversity measure that, when used with our framework, can be efficiently optimized to maximize ensemble performance (Sec. 4.3).

- We empirically demonstrate that our framework produces significant performance gains on math reasoning tasks (MATH, GSM8K, and MMLU-STEM), where our ensemble consisting of just a few small models (e.g., three Qwen2-MATH-1.5B) can outperform a larger model (e.g., Qwen2-MATH-7B) (Sec. 5).

## 2 Background and related works

**LLMs and prompts.** Consider an LLM $M$ which can be viewed as a black box that encodes conditional probability distribution of text responses $y$ over any text input $q$ and prompt $w$, from which we can sample response $\hat{y}$, i.e.

$$\hat{y} \sim M(q, w) = p_M(y|q, w). \tag{1}$$

In practice, $w$ can be reasoning prompts that instruct LLMs to reason about $q$, e.g., "Let's think step by step" in CoT (Wei et al., 2023). These prompts aim to influence the final LLM response, potentially by inducing additional LLM output (e.g., reasoning steps), and have been shown to yield performance boosts.

**Prompt optimization.** To alleviate manual effort of prompt engineering, prompt optimization (Zhou et al., 2023; Lin et al., 2024; Yang et al., 2024; Hu et al., 2024) works aim to automatically search for optimal prompts to maximize an LLM's performance on specific tasks. However, such works have mainly focused on finding the best prompt for a single LLM, leaving the potential of optimizing prompts for LLM ensembles underexplored. In contrast, our work proposes a broader, novel framework for designing inference-time LLM ensembles with diverse prompts, and

can incorporate existing prompt optimization methods. For example, we showed that an optimized prompt (e.g., "self-reflection" (Shinn et al., 2024)) can be combined with our framework.

**LLM ensembles.** Ensemble methods, which combines multiple models to achieve superior performance and robustness (Ganaie et al., 2022), have seen limited application in the LLM domain. Prior LLM ensemble works have focused on heterogeneous ensembles that combines outputs from different LLM architectures or API providers (Jiang et al., 2023), multi-agent LLM settings that focuses on interactions among agents (Du et al., 2023; Liu et al., 2023; Chen et al., 2023), or homogeneous self-ensembles that generate multiple responses from a single LLM using stochastic sampling (Wang et al., 2023b).

However, to the best of our knowledge, we are not aware of any work that has proposed forming and optimizing for homogeneous LLM ensembles where their *diversity is injected through varying reasoning prompts* to constituents with the same underlying LLM model. Our work's focus on such an approach exploits LLMs' unique capabilities of generating diverse output given only changes to its prompts, allowing for a simple but effective method to achieve significant training-free boosts to LLM reasoning performance using inference-time compute, especially given recent developments in LLM batch inference methods (Kwon et al., 2023; Zhu et al., 2024; Gim et al., 2024).

## 3 Problem formulation

Consider a task $\mathcal{T}$ with instances described as tuples $t := (q_t, c_t^*)$, where $q_t$ is a text query and $c_t^*$ is the corresponding solution. We denote the response from a single LLM $M$ as $\hat{y} := \{\hat{r}, \hat{c}\}$ which consists of its reasoning $\hat{r}$ and final answer $\hat{c}$. We evaluate the performance of the model with a specific prompt $w$, denoted as $M(\cdot, w)$, on the task by computing its expected accuracy over the set of task instances $\mathcal{T}$, i.e., $F(M(\cdot, w); \mathcal{T}) := \mathbf{E}_{t \sim \mathcal{T}}[\mathbb{I}\{\hat{c}_t = c_t^*\}]$, which in practice is computed over a representative test set.

We denote a homogeneous LLM ensemble as $\mathcal{E}(\cdot; M, n, \phi)$, consisting of $n$ instances of the same model $M$ and in general has an adjustable inference-time design parameter $\phi$. The ensemble produces a final answer when provided a task query, i.e., $\mathcal{E}(q_t; M, n, \phi) \rightarrow \hat{c}_t$, and we can evaluate its

performance based on its expected accuracy:

$$F(\mathcal{E}, \mathcal{T}) = \mathbf{E}_{t \sim \mathcal{T}}[\mathbb{I}\{\mathcal{E}(q_t; M, n, \phi) = c_t^*\}]. \quad (2)$$

Our objective is to design an ensemble framework with an appropriate design parameter $\phi$ such that given fixed $M, n$ and a small labeled development set, we can efficiently maximize Eq. (2) by optimizing for $\phi$ to produce the best performing ensemble without additional training.

## 4 Method

A key driver of the ensemble's performance is the diversity present among the constituents in the ensemble. Intuitively, a group where every member thinks the same way as each other is likely to result in less robust reasoning and decision-making (e.g. "groupthink") compared to a group consisting of members with diverse thinking styles. Similarly, having an ensemble of LLM instances where *all constituents are identical* may be expected to yield less performance advantage compared to a diversified ensemble. In our setting where only a single LLM model is available, self-ensembles (Wang et al., 2023b) are examples of the former, as the constituents rely on LLM sampling stochasticity to generate potentially diverse responses, but will nonetheless still be sampling from the same distribution in Eq. (1) and hence face limited diversity. Our framework for LLM ensembles, DIPPER, efficiently introduces such diversity at inference time even when only one LLM model is available, through the use of high fidelity, diverse prompts.

In this section, we first provide an overview of our framework DIPPER, before elaborating on the various components.

### 4.1 Overview of the DIPPER framework

Drawing inspiration from how using different prompts $w$ would result in varying response distributions in Eq. (1) given the same model $M$, our DIPPER framework has the set of prompts $\{w_i\}_{i=1}^n$ fed into the ensemble of $n$ LLM instances as the key ensemble design parameter $\phi$. DIPPER consists of the following three components:

1. **Prompt Generator.** First, an LLM generates a large candidate pool of prompts (denoted as $\mathcal{W}$), which can be based on some description of the task and in-context prompt examples that we think may be effective, if such prior knowledge is available. The goal is for the prompts

to invoke various types of reasoning pathways when addressing queries, hence injecting diversity into the ensemble.

2. **Prompt Selector.** Drawing parallel to data/prompt selection (Wu et al., 2024; Wang et al., 2025; Chen et al., 2025; Lau et al., 2024; Hemachandra et al., 2025), we select a subset of $n$ prompts $\{w_i \in \mathcal{W}\}_{i=1}^n$ from the candidate pool of prompts $\mathcal{W}$, where the selection is optimized based on a diversity metric that acts as an approximation of the relative performance of each subset.

3. **Response Aggregator.** Finally, the responses from the $n$ constituent LLMs are aggregated through a response aggregator operation $\mathcal{A}$ to produce a single final response for the ensemble.

Putting everything together, our DIPPER framework characterizes an ensemble of size $n$ via $\mathcal{E}(q_t; M, n, \{w_i\}_{i=1}^n) \coloneqq \mathcal{A}(\{M(q_t, w_i)\}_{i=1}^n) \rightarrow \hat{c}_t$, where the subset of prompts $\{w_i\}_{i=1}^n$ is chosen from a candidate pool $\mathcal{W}$ to optimize the expected ensemble performance $F(\mathcal{E}, \mathcal{T})$ for a target task $\mathcal{T}$. We now describe each component in detail.

### 4.2 Prompt Generator

The first component plays the important role of generating a large pool of candidate prompts with the following desiderata:

1. **Fidelity.** Each prompt should be able to influence the LLM into applying a certain type of reasoning approach to the task without significantly degrading the task performance.

2. **Diversity.** The prompts should differ sufficiently to elicit various reasoning pathways and provide a diverse selection pool for the subsequent component.

We first show that LLMs are capable of generating prompts that meet these desiderata, via the most direct way of prompting it to generate a pool of candidate prompts while providing it with exemplars illustrating different reasoning prompts. To do so, we considered a list of 7 reasoning prompts inspired by existing works (Wang et al., 2023a; Deng et al., 2023; Yao et al., 2022) on prompting methods to boost reasoning capabilities. Given these prompts as exemplars, we used GPT-4o to further generate a set of 200 different candidate
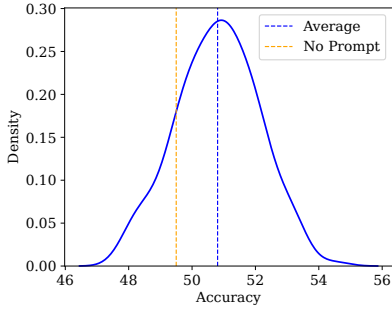
Figure 1: The accuracy distribution of 200 candidate prompts on MATH with Qwen2-MATH-1.5B.

prompts that each represent a different reasoning approach (details in Appx. A.1).

Fig. 1 shows the distribution of average accuracy over a sampled test set of MATH (Hendrycks et al., 2021) questions for each prompt, when used with the Qwen2-MATH-1.5B model (i.e., $F(M(\cdot, w); \mathcal{T})$ for $w_i \in \mathcal{W}$). Note that the distribution of accuracy is largely higher than that of the base model without prompts, and similar to the accuracies achieved by the reasoning prompt exemplars, demonstrating the fidelity requirement. Qualitatively, we see that the prompts are also relatively diverse – they generally specify certain reasoning approaches inspired by various subject domains (see Appx. C.5). We will quantify this diversity in Sec. 4.3 with our proposed metric.

Note that when generating the prompts, we did not pass any task description to the LLM prompt generator. We did so as the reasoning prompts can be task-agnostic. In practice, the candidate pool of reasoning prompts need not be generated on-the-fly, but can be drawn from a shared pool prepared beforehand by a more powerful LLM, to be used by ensembles consisting of much smaller LLMs, as we demonstrated. The actual selection of relevant prompts from this larger pool can then be done by the prompt selector component, which we will describe next in Sec. 4.3.

### 4.3 Prompt Selector

With our framework, the optimization problem in Eq. (2) reduces to an optimization to choose the best subset of prompts $\{w_i\}_{i=1}^n$ from the set of candidate prompts $\mathcal{W}$:

$$\underset{\{w_i \in \mathcal{W}\}_{i=1}^n}{\arg\max} \; F(\mathcal{E}(q_t; M, n, \{w_i\}_{i=1}^n), \mathcal{T}). \quad (3)$$

Unfortunately, directly optimizing Eq. (2) is a combinatorial problem that is very challenging, even

if a development/validation set is available for the task of interest. For example, selecting 5 prompts from a candidate pool of 200 prompts involves searching over $\binom{200}{5} \approx 2.5 \times 10^9$ candidates. Instead, we note that the best ensemble composition requires a balance of the two desiderata: fidelity and diversity. Hence, we propose optimizing Eq. (2) by considering how to prioritize the prompts that have the best predicted performance on the task $\mathcal{T}$, while maximizing the diversity of the selected set of prompts. Our method draws inspiration from past works on determinantal point processes (DPP) (Kulesza, 2012; Lau et al., 2025a), which consider similarity kernels comprising separate quality and diversity terms that match our requirements.

**Prompt fidelity.** First, we can approximate the predicted performance of each prompt by its average performance on a task development set $\mathcal{T}_d$[1]. Note that as inference using these various prompts on a small development set can be done in parallel, this process can in practice be significantly sped up by existing batch inference techniques such as those employed by vLLM (Kwon et al., 2023). Specifically, for a candidate pool of prompts $\mathcal{W}$ and development set $\mathcal{T}_d$, we can define a prompt fidelity mapping $u : \mathcal{W} \to [0, 1]$,

$$u(w) := F(M(\cdot, w), \mathcal{T}_d), \quad (4)$$

where $M(\cdot, w)$ is the LLM model conditioned by prompt $w \in \mathcal{W}$, and $F$ the expected accuracy defined in Section 3. In practice, for a candidate pool of size $n$, $u(w)$ can be represented as an $n \times 1$ column vector, with the elements representing each prompt's expected accuracy.

**Semantic entropy.** Then, we measure prompt diversity by considering how different the semantic meanings of the $n$ role prompts are from each other. We represent each prompt's semantic meaning with a mapping $R$ from its text representation $w$ into a normalized continuous vector $s \in \mathbb{R}^p$ in a $p$-dimensional semantic embedding space $\mathcal{S}$ through a sentence embedding model $M_s$ (Reimers and Gurevych, 2019), i.e., $R(w) := M_s(w)$. This mapping can be represented as an $n \times p$ prompt embedding matrix $R = [s_1, \cdots, s_n]$ where $s$ is a $1 \times p$ row vector representing each prompt.

---

[1]Without such a development set, an uninformed prior on the performance (e.g. uniform distribution across roles), or an informed-prior based on domain knowledge, could also be used.

To quantify prompt diversity of a given set of prompts, we propose to compute the volume enclosed by the selected prompts in semantic space. Intuitively, for $n$ fixed prompts, more diverse prompts point to more varied directions in semantic space, and enclose larger volume. Specifically, we note that from basic geometry, the determinant of a Gram matrix is the squared volume of the parallelepiped spanned by the embedding vectors. Hence, we define the semantic volume metric $V$ as

$$V := \log \det(RR^T), \qquad (5)$$

where we take the logarithm (for numerical stability) of the Gram matrix determinant[2]. Sec. C.4 shows how sets of prompts that are qualitatively observed to be more diverse have larger quantitative semantic volume.

**Fidelity-adjusted semantic volume (FASV).** To incorporate the prompts' expected accuracy information, we can compute the performance-adjusted prompt embedding matrix,

$$\tilde{R} := \exp(\frac{\alpha}{2} \operatorname{diag}(u))R, \qquad (6)$$

where $\operatorname{diag}(u)$ is the diagonal matrix with its $i^{\text{th}}$ diagonal element being the corresponding element $u_i$. This essentially scales each row $s_i$ in $R$ by an exponential factor based on its corresponding predicted accuracy, $\exp(\frac{\alpha}{2}u_i)$, where $\alpha$ is a scalar hyperparameter influencing the balance between diversity and expected performance. Intuitively, prompts with higher expected accuracy would then be able to support larger semantic volume and hence be prioritized for inclusion into the ensemble. The adjusted embedding matrix can then be used to compute the semantic volume in Eq. (5), which simplifies to

$$\tilde{V} = \log \det(\tilde{R}\tilde{R}^T) = V + \alpha\|u\|_1, \qquad (7)$$

providing an interpretable expression illustrating the balance between the diversity (i.e., the semantic volume metric in Eq. (5)) and fidelity desiderata (i.e., the L1 norm of the prompt fidelity metric in (Eq. (4)) that needs to be optimized for the ensemble. Derivation details are in Appendix B, and we provide empirical analysis of the effectiveness of this combined metric in Section 5.3.

---

[2]We omit a factor of 2 which does not affect the optimization process. For our setting, we also have $n < p$ as the semantic embedding space is usually high dimensional.

---

**Algorithm 1** DIPPER FASV algorithm

1: **Input:** LLM model $M$, Initial candidate prompt set $\bar{\mathcal{W}}$, Semantic embedding model $M_s$, Development set $\mathcal{T}_d$, Ensemble size $n$, Fidelity-diversity hyperparam $\alpha$
2: **Output**: Ensemble prompt set $\mathcal{Z}$
3: $\mathcal{Z} \leftarrow \{\ \}$
4: $\bar{u}(w) \leftarrow [F(M(\cdot, w_i), \mathcal{T}_d) \text{ for } w_i \in \bar{\mathcal{W}}]$
5: $\mathcal{Z} \leftarrow \mathcal{Z} \cup \arg\max_w \bar{u}(w)$
6: $\mathcal{W} \leftarrow \bar{\mathcal{W}} \setminus \arg\max_w \bar{u}(w)$
7: **for** $j = 1, \ldots, n$ **do**
8: $\quad \tilde{\mathcal{V}} \leftarrow [\ ]$
9: $\quad$ **for** $w_k \in \mathcal{W}$ **do**
10: $\quad\quad \mathcal{P} \leftarrow \mathcal{Z} \cup w_k$
11: $\quad\quad u(w) \leftarrow [F(M(\cdot, w_i), \mathcal{T}_d) \text{ for } w_i \in \mathcal{P}]$
12: $\quad\quad R(w) \leftarrow [M_s(w_i) \text{ for } w_i \in \mathcal{P}]$
13: $\quad\quad \tilde{V}_{w_k} \leftarrow \log\det(RR^T) + \alpha\|u\|_1$
14: $\quad\quad \tilde{\mathcal{V}}(w) \leftarrow [\tilde{\mathcal{V}}(w), \tilde{V}_{w_k}]$
15: $\quad$ **end for**
16: $\quad \mathcal{Z} \leftarrow \mathcal{Z} \cup \arg\max_w \tilde{\mathcal{V}}(w)$
17: $\quad \mathcal{W} \leftarrow \mathcal{W} \setminus \arg\max_w \tilde{\mathcal{V}}(w)$
18: **end for**
19: **return** $\mathcal{Z}$

---

**Optimization of semantic entropy.** We can now recast Eq. (2) as an optimization of the fidelity-adjusted semantic volume metric $\tilde{V}$ evaluated over the set of candidate prompts. Note that instead of the expected ensemble performance $F(\mathcal{E})$, which is an objective that can only be optimized by blackbox optimization methods like Bayesian Optimization (Garnett, 2023; Dai et al., 2023; Lau et al., 2025b), our metric $\tilde{V}$ can be efficiently approximated by well-established heuristics.

Specifically, as the semantic volume metric is submodular, we can optimize for the best subset of roles by incrementally building the subset with a greedy approach up to the desired size $n$ and still be guaranteed a good approximation (Nemhauser et al., 1978). This is an important advantage that allows us an efficient and theoretically-inspired approach to obtain the best ensemble prompts. Our proposed algorithm is outlined in Algorithm 1.

## 4.4 Response Aggregator

Given the responses from the various LLMs of constituents, the aggregation method determines how much information from the constituents is used to derive the final output of the ensemble. We consider the two most popular approaches:

**Majority voting (MV)**   It involves extracting the final answer $\hat{c}$ from each LLM response $\hat{y} = \{\hat{r}, \hat{c}\}$, and selecting the answer that has been proposed the most number of times. This approach does not evaluate the quality of reasoning $\hat{r}$ output produced by each LLM, but is easily implementable.

**Best-of-N**   An external reward model is implemented to evaluate the response $\hat{y}$ of each agent, and the response with the highest score is selected as the final response. This approach does not leverage consensus among constituents but could be effective in identifying the correct responses that would be only covered by a few agents.

## 5   Experiments

**Experimental set-up.** We empirically evaluate our framework on mathematically reasoning tasks with the MATH (Hendrycks et al., 2021), GSM8K, and MMLU-STEM datasets. We implement our framework using the GPT-4o as our prompt generator and Qwen2-MATH-1.5B as the constituent model in the ensemble, where the ensemble constituents are run in parallel using vLLM (Kwon et al., 2023) for fast batch inference. Further details (Appx. A) and additional results (Appx. C) are in the Appendix.

**Baselines.**   We evaluate our DIPPER framework by comparing it against the "Self-ensemble" method, which lacks prompt diversity but incorporates diversity through repeated response sampling (Wang et al., 2023b), along with the single model performance as a reference. We also include two other implementation variants of DIPPER in our analysis, beyond the implementation based on semantic volume, "Dipper (FASV)":

1. **Random+.**   Here we randomly sample prompts from the candidate pool based on a probability distribution proportional to their predicted accuracy as defined in Eq. (4), i.e., $p(w) \propto u(w)$. This aims to achieve diversity through the sampling process while prioritizing prompts with higher predicted accuracy.

2. **Top-n.**   Here we greedily select the top $n$ prompts which are ranked based on their predicted accuracy $u(w)$. It assumes that the diversity of prompts introduced by our prompt generation process is sufficient and hence does not explicitly optimize for ensemble diversity during the prompt selection phase.
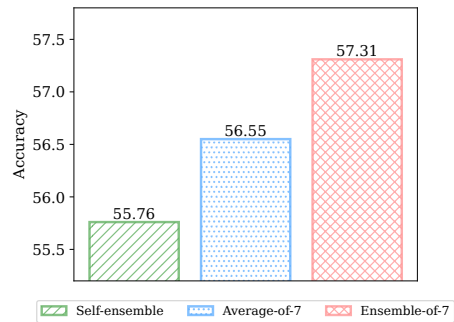


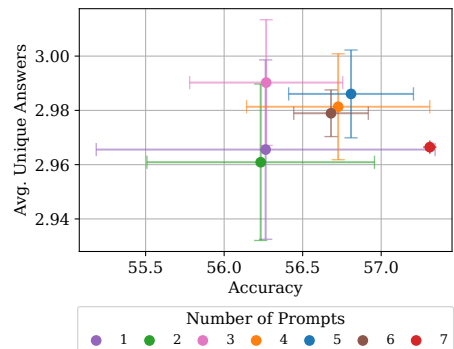Figure 2: Comparison of different ensembles of 7 reasoning prompts on MATH.



Figure 3: Accuracy vs. average number of unique answers using different numbers of prompts in ensembles.

### 5.1   Ensembles with fixed prompt methods

To motivate our DIPPER framework and demonstrate the importance of prompt diversity, we first consider a fixed set of seven distinct reasoning prompts inspired by existing works (Wang et al., 2023a; Deng et al., 2023; Yao et al., 2022) (details in Appx. A.1). With a fixed ensemble size of seven, Fig. 2 shows that an ensemble using these seven different prompts (57.31%) outperforms both a baseline self-ensemble without prompt variation (55.76%) and the average performance (56.55%) of seven self-ensembles, each using only one of the distinct prompts.

In addition, we evaluated the impact of prompt diversity by constructing ensembles with varying numbers of unique prompts (from one to seven) drawn from this set, while maintaining an ensemble size of seven. When fewer than seven unique prompts were used, responses were randomly sampled to meet the ensemble size. The result in Fig. 3 indicates that increasing the number of unique prompts generally leads to higher accuracy and reduced variance. This suggests that prompt diversity within an ensemble can enhance performance and consistency, particularly when the performances of

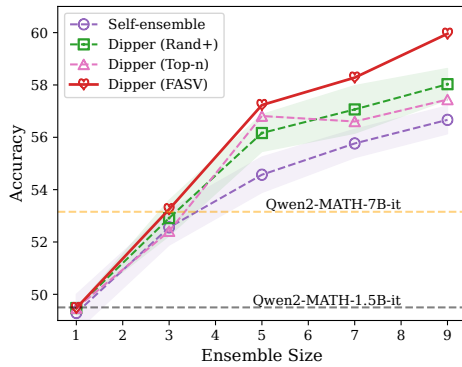prompts are unknown before the final evaluation.



Figure 4: Comparison of different ensemble methods on MATH for Qwen2-MATH-1.5B.
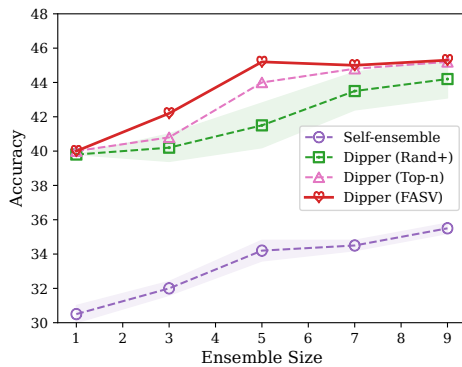


Figure 5: Comparison of different ensemble methods on MATH for the LLaMA3.2B model.

## 5.2 Ensembles with optimized prompt diversity

Next, we consider our full DIPPER framework. We first generate a pool of prompt candidates ($|\mathcal{W}| = 200$) using the 7 reasoning prompts in the previous section as in-context exemplars (details in Appx. A.1) and then perform prompt fidelity-diversity optimization (Sec. 4.3) to select the best ensemble prompts. As shown in Fig. 4, our full DIPPER implementation with FASV achieves the highest accuracy compared to the self-ensemble baseline and all other DIPPER variants across various ensemble sizes. DIPPER also significantly outperforms the single LLM. For example, DIPPER with $n = 9$ has close to a 10%-pt increase (~20% accuracy gain) compared to the single LLM baseline. In fact, our ensemble that consists of just 3 Qwen2-MATH-1.5B models already (slightly) outperforms the next model size class, the Qwen2-MATH-7B model. Note also that the performance gain of DIPPER over the self-consistency baseline is about as large as the gain from moving up one model class (from the 1.5B to 7B model). We see similar results on MATH with the general model
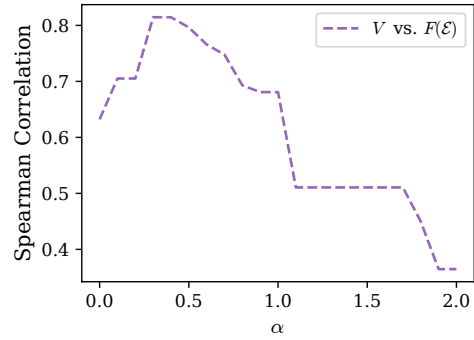


Figure 6: Spearman correlation between $V$ and test performance $F(\mathcal{E})$ on the MATH under different fidelity-diversity hyperparameter $\alpha$.
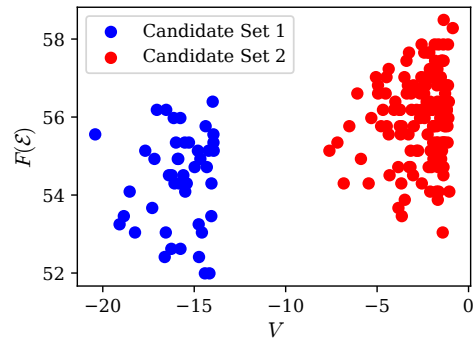


Figure 7: The scatter plot showing correlation between $V$ and $F(\mathcal{E})$ for different prompt candidates.

LLaMA3.2B in Fig. 5, where DIPPER (FASV) is shown consistently effective. Refer to Appendix C for more results for other datasets (e.g., GSM8K and MMLU-STEM).

## 5.3 Fidelity-diversity optimization

To further understand the mechanisms behind DIPPER's performance gains, we analyze the predictive power of our fidelity-adjusted semantic volume metric in Eq. (7) (which we denote as $V$ in this section for notational simplicity) on the final ensemble performance on the test set $F(\mathcal{E})$. We quantify this by computing the Spearman correlation between $V$ and $F(\mathcal{E})$: the higher the Spearman correlation, the better our optimization of the ensemble prompts via $V$ will lead to higher ensemble test performance. Fig. 6 shows the Spearman correlation of $V$ and $F(\mathcal{E})$ for the MATH dataset experiment, with different fidelity-diversity hyperparameter $\alpha$ values. We can observe two key insights.

First, there is a relatively strong positive correlation between $V$ and $F(\mathcal{E})$, going as high as 0.8 for some values of $\alpha$. This corroborates our main results where our DIPPER method that explicitly optimizes for $V$ outperforms other baselines and achieves higher $F(\mathcal{E})$.

Second, there is a U-shape trend between the Spearman correlation and hyperparameter value $\alpha$, where the correlation increases as $\alpha$ increases from 0, but decreases after a certain point. This trend demonstrates the need of taking into account both fidelity and diversity when optimizing for the set of ensemble prompts, as we discussed in Section 4.3. On the one hand, $\alpha = 0$ corresponds to the case where we focus solely on diversity and ignore the fidelity or individual predicted performance of prompts ($u(w)$) – this may select a set of diverse prompts, but potentially some irrelevant or poor performing prompts. However, if we emphasize fidelity too much and disregard diversity, we may end up selecting very similar prompts resulting in less ensemble performance gains. At the extreme, choosing large $\alpha$ reduces to the Top-n baseline implementation, which has poorer performance than DIPPER that optimizes for semantic volume. In practice, just like other machine learning hyperparameters, we could inform the choice of $\alpha$ with the development set, if available.

### 5.4 Prompt candidate matters

We then analyze how the candidate pool diversity introduced by Prompt Generator contributes to our framework. Out of the original candidate set $\mathcal{W}$ (Candidate set 2), we obtained another set by selecting one cluster after performing k-means clustering over $\mathcal{W}$ with $k = 4$ ($\mathcal{W}'$, Candidate set 1). We then randomly select ensembles of size $n = 5$, and plot their respective $V$ and $F(\mathcal{E})$ in Fig. 7. We can see that ensembles from $\mathcal{W}'$ have much lower accuracy and semantic volume compared to those from $\mathcal{W}$, illustrating the importance of the candidate pool diversity from the Prompt Generator.

### 5.5 DIPPER combined with other prompting methods like Reflexion

In addition, we also show that our ensemble framework DIPPER is **orthogonal** to other established prompting techniques (e.g. CoT and Reflexion (Shinn et al., 2024)), allowing it to stack and bring greater performance. To demonstrate this, we first use DIPPER to select 5 agents and query each agent with questions from the MATH dataset. Their initial responses will then be self-reflected according to the method proposed in Reflexion (Shinn et al., 2024), before being aggregated into the final answer with MV. We found that combining self-reflection with DIPPER achieves a performance gain of $8\%$ (from an accuracy of $57\%$ to $65\%$),
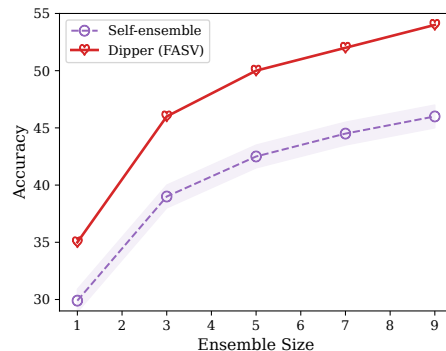


Figure 8: Comparison of DIPPER and self-ensemble baseline on MATH using LLaMA-3B and Best-of-N aggregation.

demonstrating that DIPPER has the potential to be extended further or combined with other methods.

### 5.6 Generalization to Best-of-N aggregation

Finally, we study the effects of using the Best-of-N aggregation for our response aggregator component, showing that DIPPER can work well with external reward models. We use an existing reward model, "Qwen2.5-Math-RM-72B" to assess the quality of the generated responses and select the final response, using the LLaMA-3B model and Best-of-N aggregation in our DIPPER framework. As seen in Fig. 8, the DIPPER variants significantly beats the self-ensemble baseline when Best-of-N is used. In this scenario, the performance among the DIPPER variants are closer since Best-of-N considers each constituent individually rather than jointly (like in MV) to produce the final answer, though our full DIPPER implementation still consistently performs the best. We also see that DIPPER can stack with benefits from a strong verifier model, given its performance gains compared to the result in Fig. 9 where no verifier model is available.

## 6 Conclusion

In this work, we have proposed a novel framework, DIPPER, where a single LLM model type is fed an optimized, diverse set of reasoning prompts in parallel, effectively producing an ensemble at inference time to achieve performance improvement in reasoning tasks. Our empirical findings have demonstrated the effectiveness of various DIPPER implementations in improving inference performance for a variety of reasoning tasks, which may inspire future works to investigate additional optimization methods for prompt-based inference-time ensembles to further improve performance gains.

## Limitations

Our framework DIPPER focuses on developing inference-time ensembles where each constituent is based on the same base model – this caters to the most common and straightforward scenario where users are using a single LLM model and can apply DIPPER to further boost its reasoning performance at inference time without additional training. However, when users may wish to use heterogeneous models, DIPPER currently does not take into account such model diversity, which we believe may enable further performance boosts if properly optimized. We leave it to future works to potentially build on DIPPER to extend it beyond its current limitations in this regard.

## Acknowledgment

## References

Weize Chen, Yusheng Su, Jingwei Zuo, Cheng Yang, Chenfei Yuan, Chi-Min Chan, Heyang Yu, Yaxi Lu, Yi-Hsin Hung, Chen Qian, Yujia Qin, Xin Cong, Ruobing Xie, Zhiyuan Liu, Maosong Sun, and Jie Zhou. 2023. AgentVerse: Facilitating Multi-Agent Collaboration and Exploring Emergent Behaviors. *Preprint*, arXiv:2308.10848.

Zhiliang Chen, Gregory Kang Ruey Lau, Chuan-Sheng Foo, and Bryan Kian Hsiang Low. 2025. Duet: Optimizing training data mixtures via feedback from unseen evaluation tasks. In *ICLR 2025 Workshop on Navigating and Addressing Data Problems for Foundation Models*.

Zhongxiang Dai, Gregory Kang Ruey Lau, Arun Verma, Yao Shu, Bryan Kian Hsiang Low, and Patrick Jaillet. 2023. Quantum bayesian optimization. *Advances in Neural Information Processing Systems*, 36:20179–20207.

Yihe Deng, Weitong Zhang, Zixiang Chen, and Quanquan Gu. 2023. Rephrase and respond: Let large language models ask better questions for themselves. *arXiv preprint arXiv:2311.04205*.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B. Tenenbaum, and Igor Mordatch. 2023. Improving Factuality and Reasoning in Language Models through Multiagent Debate. *Preprint*, arxiv:2305.14325.

M. A. Ganaie, Minghui Hu, A. K. Malik, M. Tanveer, and P. N. Suganthan. 2022. Ensemble deep learning: A review. *Engineering Applications of Artificial Intelligence*, 115:105151.

Roman Garnett. 2023. *Bayesian Optimization*. Cambridge University Press.

In Gim, Guojun Chen, Seung-seob Lee, Nikhil Sarda, Anurag Khandelwal, and Lin Zhong. 2024. Prompt Cache: Modular Attention Reuse for Low-Latency Inference. *Preprint*, arXiv:2311.04934.

Apivich Hemachandra, Gregory Kang Ruey Lau, See-Kiong Ng, and Bryan Kian Hsiang Low. 2025. Pied: Physics-informed experimental design. In *Proc. ICLR*.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the MATH dataset. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Wenyang Hu, Yao Shu, Zongmin Yu, Zhaoxuan Wu, Xiaoqiang Lin, Zhongxiang Dai, See-Kiong Ng, and Bryan Kian Hsiang Low. 2024. Localized zeroth-order prompt optimization. In *Proc. NeurIPS*.

Jie Huang and Kevin Chen-Chuan Chang. 2023. Towards Reasoning in Large Language Models: A Survey. *Preprint*, arxiv:2212.10403.

Yichong Huang, Xiaocheng Feng, Baohang Li, Yang Xiang, Hui Wang, Bing Qin, and Ting Liu. 2024. Ensemble Learning for Heterogeneous Large Language Models with Deep Parallel Collaboration. *Preprint*, arXiv:2404.12715.

Dongfu Jiang, Xiang Ren, and Bill Yuchen Lin. 2023. LLM-Blender: Ensembling Large Language Models with Pairwise Ranking and Generative Fusion. *Preprint*, arXiv:2306.02561.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2023. Large Language Models are Zero-Shot Reasoners. *Preprint*, arXiv:2205.11916.

Anders Krogh and Jesper Vedelsby. 1994. Neural Network Ensembles, Cross Validation, and Active Learning. In *Advances in Neural Information Processing Systems*, volume 7. MIT Press.

Alex Kulesza. 2012. Determinantal Point Processes for Machine Learning. *Foundations and Trends® in Machine Learning*, 5(2-3):123–286.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Gregory Kang Ruey Lau, Hieu Dao, Nicole Kan Hui Lin, and Bryan Kian Hsiang Low. 2025a. Uncertainty quantification for mllms. In *ICML 2025 Workshop on Reliable and Responsible Foundation Models*.

Gregory Kang Ruey Lau, Apivich Hemachandra, See-Kiong Ng, and Bryan Kian Hsiang Low. 2024. PINNACLE: PINN Adaptive ColLocation and Experimental points selection. In *Proc. ICLR*.

Gregory Kang Ruey Lau, Yue Ran Kang, Zi-Yu Khoo, Apivich Hemachandra, Ruth Wan Theng Chew, and Bryan Kian Hsiang Low. 2025b. Readme: Rapid equation discovery with multimodal encoders. In *2nd AI for Math Workshop@ ICML 2025*.

Xiaoqiang Lin, Zhaoxuan Wu, Zhongxiang Dai, Wenyang Hu, Yao Shu, See-Kiong Ng, Patrick Jaillet, and Bryan Kian Hsiang Low. 2024. Use Your INSTINCT: INSTruction optimization usIng Neural bandits Coupled with Transformers. In *Proc. ICML*.

Zijun Liu, Yanzhe Zhang, Peng Li, Yang Liu, and Diyi Yang. 2023. Dynamic LLM-Agent Network: An LLM-agent Collaboration Framework with Agent Team Optimization.

George Nemhauser, Laurence Wolsey, and M. Fisher. 1978. An analysis of approximations for maximizing submodular set functions—i. *Mathematical Programming*, 14:265–294.

Shuofei Qiao, Yixin Ou, Ningyu Zhang, Xiang Chen, Yunzhi Yao, Shumin Deng, Chuanqi Tan, Fei Huang, and Huajun Chen. 2023. Reasoning with Language Model Prompting: A Survey. *Preprint*, arxiv:2212.09597.

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.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2024. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*, 36.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. 2024. Scaling LLM Test-Time Compute Optimally can be More Effective than Scaling Model Parameters. *Preprint*, arXiv:2408.03314.

Qwen Team. 2024. Introducing Qwen2-Math. https://qwenlm.github.io/blog/qwen2-math/.

Jingtan Wang, Xiaoqiang Lin, Rui Qiao, Pang Wei Koh, Chuan-Sheng Foo, and Bryan Kian Hsiang Low. 2025. NICE data selection for instruction tuning in LLMs with non-differentiable evaluation metric. In *Proc. ICML*.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023a. Self-Consistency Improves Chain of Thought Reasoning in Language Models. *Preprint*, arxiv:2203.11171.

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

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. *Preprint*, arXiv:2201.11903.

Zhaoxuan Wu, Xiaoqiang Lin, Zhongxiang Dai, Wenyang Hu, Yao Shu, See-Kiong Ng, Patrick Jaillet, and Bryan Kian Hsiang Low. 2024. Prompt optimization with ease? efficient ordering-aware automated selection of exemplars. In *Proc. NeurIPS*.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V Le, Denny Zhou, and Xinyun Chen. 2024. Large language models as optimizers. In *Proc. ICLR*.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of Thoughts: Deliberate Problem Solving with Large Language Models. *Preprint*, arxiv:2305.10601.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2022. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*.

Sheheryar Zaidi, Arber Zela, Thomas Elsken, Chris Holmes, Frank Hutter, and Yee Whye Teh. 2020. Neural Ensemble Search for Uncertainty Estimation and Dataset Shift. https://arxiv.org/abs/2006.08573v3.

Chuanyang Zheng, Zhengying Liu, Enze Xie, Zhenguo Li, and Yu Li. 2023. Progressive-Hint Prompting Improves Reasoning in Large Language Models. *Preprint*, arxiv:2304.09797.

Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. 2023. Large language models are human-level prompt engineers. In *Proc. ICLR*.

Hanlin Zhu, Banghua Zhu, and Jiantao Jiao. 2024. Efficient Prompt Caching via Embedding Similarity. *Preprint*, arXiv:2402.01173.

# A  Detailed Experimental Setting

The huggingface model path for the primary model we used is "Qwen/Qwen2-Math-1.5B-Instruct" and the sentence transformer's model path is 'all-MiniLM-L6-v2'. We use the default generation parameters for Qwen2-Math-1.5B-Instrct: the temperature is set to 0.7, the top probability used for filtering is set to 0.8, and the repetition penalty is set to 1.05. We also set the max tokens to be generated to 512.

## A.1  Fixed 7 prompts and Prompt Generation

We consider 7 prompts inspired by existing works and list them in Tab. 1 below.

Table 1: The table of 7 basic reasoning prompts inspired by existing works.

| Prompt |
| --- |
| Let's think step-by-step to find the answer. |
| Reflect on the question carefully before answering. |
| Rephrase the question in your own words before responding. |
| Actively reason through the question and answer each part systematically. |
| Answer this question as a scientist would. |
| Eliminate the obviously incorrect answers first and then choose the most likely correct answer. |
| Analyze the context of the question and use relevant information to derive the answer. |

We use the prompt template in Tab. 2 to generate 200 diverse prompts.

Table 2: The prompt template for generating more reasoning prompts based on the 7 prompts.

| Prompt Generation Template |
| --- |
| Here are some instruction examples:<br><br>{7 reasoning prompts}<br><br>Study the above examples and brainstorm 200 similar instructions with detailed descriptions of different reasoning behaviors that are helpful for reasoning. Those 200 proposed instructions should be diverse enough. |

## A.2  Evaluation

We primarily consider three datasets in our paper. For MATH, we randomly sample 10% test samples from each category in its official test split and form a fixed subset of size 500. We then uniformly randomly sample 20 samples from this subset to create a validation dataset and use the rest 480 samples as the hold-out test dataset. For GSM8K and MMLU-STEM, we use their official split of test data and uniformly randomly sample 20 samples to form a validation dataset for each task, and use the rest samples as the hold-out test data.

In the inference evaluation, we use 4-shot exemplars for MATH, 8-shot for GSM8K, and 5-shot for MMLU-STEM. Those exemplars are adopted from the evaluation setting in Qwen2-MATH (Team, 2024) and fixed for all questions and all methods.

# B Fidelity-adjusted semantic volume metric

In this section, we provide the explicit derivation of how our fidelity-adjusted semantic volume metric can be simplified to a weighted sum of two terms representing the diversity and fidelity desiderata in Eq. (7), which clearly illustrates the balance between the two desiderata during the optimization process.

$$\tilde{V} = \log \det(\tilde{R}\tilde{R}^T) \tag{8}$$

$$= \log \det \left( \exp(\frac{\alpha}{2}\operatorname{diag}(u))R \right) \left( \exp(\frac{\alpha}{2}\operatorname{diag}(u))R \right)^T \tag{9}$$

$$= \log \left[ \det \left( \exp(\frac{\alpha}{2}\operatorname{diag}(u)) \right) \det \left( RR^T \right) \det \left( \exp(\frac{\alpha}{2}\operatorname{diag}(u))^T \right) \right] \tag{10}$$

$$= \log \det(RR^T) + 2\log \det \left( \exp(\frac{\alpha}{2}\operatorname{diag}(u)) \right) \tag{11}$$

$$= V + 2\log \prod_i \exp(\frac{\alpha}{2}u_i) \tag{12}$$

$$= V + \alpha\|u\|_1 \tag{13}$$

where Eq. (9) follows from the definition of $\tilde{R}$ in Eq. (6), Eq. (10) uses the identity $\det(AB) = \det(A)\det(B)$, Eq. (11) the identity $\log(AB) = \log(A) + \log(B)$, Eq. (12) the definition of semantic volume in Eq. (5), and Eq. (13) noting that $\sum_i u_i = \|u\|_1$ since $u \geq 0$.

# C Additional Results

## C.1 Results on General-Purpose Model

To show our method DIPPER also generalizes to a general-purpose model (e.g., LLaMA), we evaluate its performance using LLaMA3.2-3B-it model on MATH and MMLU-STEM. The results are presented in Fig. 9 and Fig. 10, which demonstrate that our full DIPPER implementation consistently outperforms the self-ensemble baseline and other DIPPER variants across datasets.
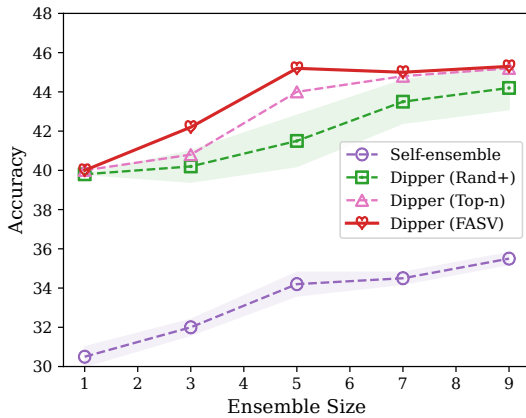


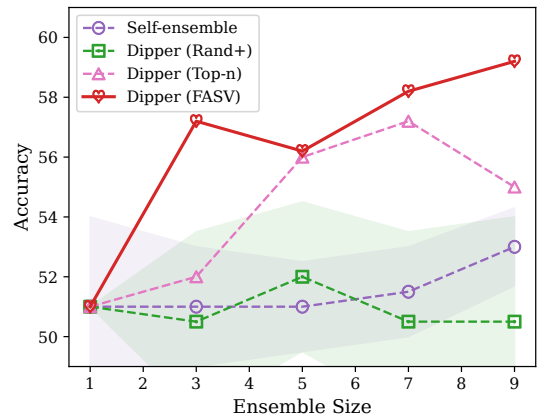Figure 9: Comparison of different ensemble methods on MATH for the LLaMA3.2-3B-it model.

Figure 10: Comparison of different ensemble methods on MMLU-STEM for the LLaMA3.2-3B-it model.

## C.2 Results on more datasets for the Qwen2-MATH-1.5B model

Apart from the MATH dataset, we also evaluate the performance of DIPPER using the Qwen2-MATH-1.5B model on MMLU-STEM and GSM8K. The results in Fig. 11 and Fig. 12 again demonstrate that our full DIPPER implementation can consistently outperform the self-ensemble baseline and achieve superior or comparable results against the other DIPPER variants. The performance gains in GSM8K is more limited compared to the gains in experiments for other datasets as it is an easier dataset where the base model

can already achieve high accuracy. As can be seen across all our experimental results, our full DIPPER implementation comprising the theoretically-inspired semantic volume diversity optimization component achieves the most consistent performance, unlike some of the other variants.
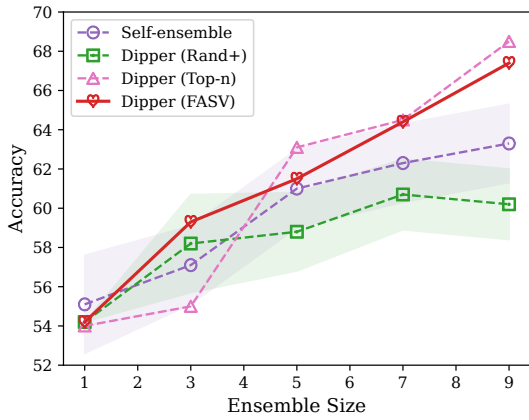


Figure 11: Comparison of different ensemble methods on MMLU-STEM. DIPPER variants outperform the self-ensemble baseline, consistent with the other experiments.
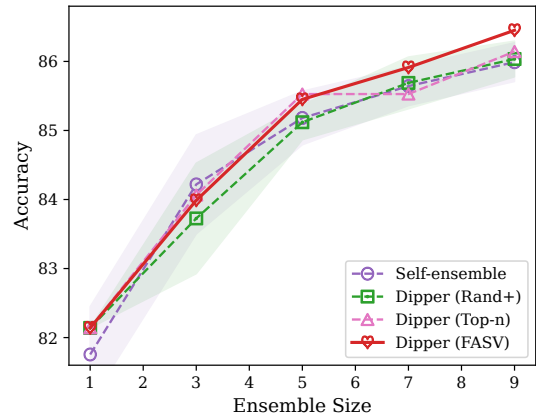
Figure 12: Comparison of different ensemble methods on GSM8K. DIPPER still outperforms the self-ensemble baseline, although gains are not as obvious as in other benchmarks as it is an easier task where the base model can already perform well.

## C.3    More Results on Prompt Diversity

We also show that a strong Spearman correlation between $V$ and $F(\mathcal{E})$ exists for different datasets (e.g., GSM8K). The results presented in Fig. 13 and Fig. 14 demonstrate a consistent Spearman correlation between the semantic diversity $V$ and accuracy $F(\mathcal{E})$ exists. Besides, choosing different fidelity-diversity hyperparameters $\alpha$ may give different results when optimizing for the diversity.
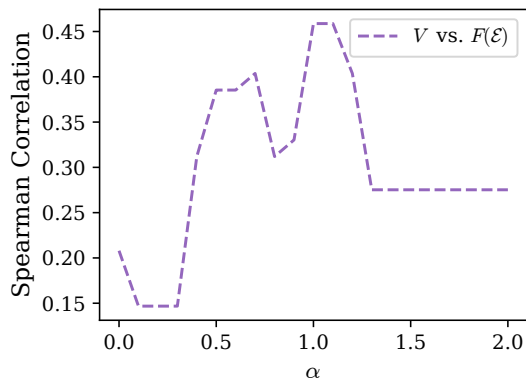


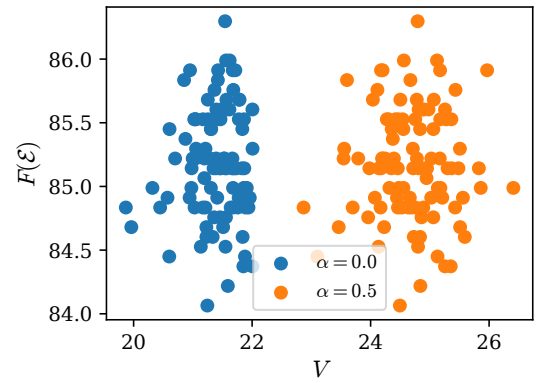Figure 13: Line plot showing the Spearman correlation between $V$ and $F(\mathcal{E})$ on GSM8K with different $\alpha$ values.

Figure 14: Scatter plot showing the Spearman correlation between $V$ and $F(\mathcal{E})$ on GSM8K with 2 different $\alpha$ values.

## C.4    Illustration of Semantic Volume

We illustrate our semantic column metric $V$ here by comparing the semantic volume of (1) a set of 5 prompts generated by just paraphrasing a single original prompt, and (2) a set of 5 prompts randomly selected from the diverse candidate pool $\mathcal{W}$. Table 3 shows how sets of prompts that are qualitatively observed to be more diverse have larger quantitative semantic volume.

| Set of Prompts | $V$ |
|---|---|
| **Use a Scenario Analysis Approach**: Analyze different scenarios to determine their feasibility and impact. <br> **Consider Cause and Effect**: Identify potential causes and their effects to understand the question better. <br> **Use a Benchmarking Approach**: Compare the question to best practices or standards to find the best answer. <br> **Break Down the Problem**: Divide the question into smaller, manageable parts and tackle each part individually before synthesizing the overall answer. <br> **Apply Mathematical Logic**: Use mathematical principles and logic to solve the problem, even if it's not a math question. | -1.24 |
| Let's analyze this one step at a time. <br> Let's break this down step by step. <br> Let's tackle each part individually. <br> Let's approach this incrementally. <br> Let's examine this in a methodical manner. | -1.80 |

Table 3: Example of two sets of prompts with the corresponding diversity score $V$.

### C.5 Generated prompts based on 7 prompts

Below in Table 4 we provide some examples of the generated prompts from GPT-4o based on the 7 prompts.

Table 4: Examples of reasoning prompts generated based on 7 basic prompts.

| Prompt |
|---|
| **Break Down the Problem**: Divide the question into smaller, manageable parts and tackle each part individually before synthesizing the overall answer. |
| **Apply Mathematical Logic**: Use mathematical principles and logic to solve the problem, even if it's not a math question. |
| **Use Analogies**: Relate the question to a familiar concept or situation to better understand and solve it. |
| **Consider the Opposite**: Think about what the answer would be if the opposite were true, to gain a different perspective. |
| **Consider Cause and Effect**: Identify potential causes and their effects to understand the question better. |