PToco: Prefix-based Token-level Collaboration Enhances Reasoning for Multi-LLMs

Yuang Bian, Yupian Lin, Jingping Liu*, Tong Ruan*
School of Information Science and Engineering, East China University of Science and Technology, Shanghai, China
yuangbian1999@gmail.com, yupianlin@aliyun.com, {jingpingliu, ruantong}@ecust.edu.cn

Abstract

Collaboration between multiple Large Language Models (LLMs) has attracted significant attention for its potential to mitigate hallucinations and enhance reasoning capabilities. Previous approaches, such as multi-agent debate and decoding-time integration, either rely on highly capable models with strong self-reflection abilities or are limited to models sharing the same tokenizer. To address these limitations, we introduce PToco (Prefix-based Token-level Collaboration), a novel mechanism that enables effective collaboration among less capable LLMs, independent of tokenizer differences. PToco uses a prefix-grouping method to extract consensus among tokens with varying levels of granularity, ensuring coherent and robust token generation across multiple models. Experimental results on a series of reasoning tasks demonstrate that PToco significantly improves performance over individual models.[1] Furthermore, this approach generalizes well across different quantities and sizes of participating models, providing a more flexible and efficient solution for multi-LLM ensembles.

1 Introduction

The rapid development of large language models (LLMs) has led to remarkable improvements across a wide range of tasks (OpenAI, 2023; Huang et al., 2022; Zhao et al., 2023). Despite LLM's impressive capabilities, they are prone to hallucinations (Huang et al., 2023; Rawte et al., 2023), a phenomenon in which models generate statements that appear plausible but are factually incorrect or nonsensical. As a result, a noticeable disparity still remains between the performance of LLMs and human-level proficiency, especially in tasks requiring complex reasoning (Cobbe et al., 2021; Valmeekam et al., 2022).

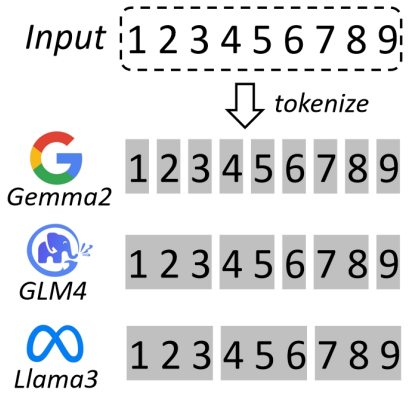

Figure 1: Illustration of different tokenization strategies for numerical data. Gemma2 tokenizes numbers into individual digits; Llama3 groups three digits together; GLM4 employs a hybrid approach.

To address this issue, previous researches have resorted to model ensemble techniques. One prominent approach involves multi-agent debate (Du et al., 2023; Liang et al., 2023), where different LLMs engage in iterative critique and response, refining their outputs over multiple turns. This method encourages deeper reasoning by allowing models to self-correct based on peer critiques. However, meaningful debate is typically limited to highly capable models like GPT4 (OpenAI, 2023), while smaller LLMs struggle to effectively critique or reflect on errors. Moreover, the debate approach necessitates that each LLM generates a complete answer as its position before debating, which disregards the token-level uncertainty inherent during decoding time. Another ensemble method involves token-level collaboration, where integration is performed at each decoding step. This relies on each LLM predicting the probability distribution over its vocabulary for the possible next token, and selecting the token with the highest overall probability across the models. While this method eliminates the need for natural language communication, it is limited to LLMs that share the same tokenizer.

*Corresponding authors.

[1]The code for the experiments is publicly available at https://github.com/BianYuang/PToco

Proceedings of the 31st International Conference on Computational Linguistics, pages 8326–8335
January 19–24, 2025.

As shown in Figure 1, different families of LLMs employ distinct tokenization strategies, especially when handling numerical data which is particularly susceptible to hallucinations. This discrepancy in tokenization can result in models generating tokens with varying levels of granularity from the same prompt, making it difficult to directly integrate their outputs.

Building on these insights, we propose PToco (**P**refix-based **To**ken-level **Co**llaboration), a novel mechanism for decoding-time collaboration among LLMs with different tokenization strategies. PToco specifically addresses the challenge of tokenization inconsistency, which arises when multiple models generate tokens of varying granularities, such as words, subwords, or characters.

In detail, PToco introduces a prefix-based approach to ensure effective collaboration. At each decoding step, PToco first classifies the tokens generated by different LLMs into groups based on their prefix relationships—whether one token is a prefix of another. PToco computes the cumulative probability for each group by aggregating the probabilities of all tokens within the group. The group with the highest cumulative probability is selected, and the longest common prefix within that group is chosen as the final output token for the current round. This method ensures a coherent and consistent generation process across models, regardless of the different granularities of predicted tokens in each round.

Our key contributions are outlined as follows:

- We introduce PToco, a framework that enables token-level collaboration between multiple LLMs during decoding steps, without relying on identical tokenizers.
- We conduct a comprehensive set of experiments to demonstrate the effectiveness of PToco when integrating a number of less capable LLMs with fewer than 10 billion parameters.
- We perform extensive ablation studies to assess PToco's generalizability across LLMs of varying sizes and quantities.

2 Related Work

2.1 Reasoning Ability in LLMs

Chain-of-Thought (Wei et al., 2022) prompting, which utilizes a sequence of intermediate reasoning steps to progressively reach the final outcome, has emerged as a pivotal technique for eliciting reasoning capabilities in LLMs (Luo et al., 2023; Zhou et al., 2023; Huang and Chang, 2022). This concept was later expanded by Yao et al. (2024) and Long (2023) into Tree-of-Thought, which investigates various intermediate steps to find the optimal reasoning path, with the option to backtrack if needed. Graph-of-Thought (Besta et al., 2024) further advances the reasoning capabilities of LLMs by modeling the reasoning process as a graph, where thoughts are represented as vertices and their relationships as edges. Across our experiments with PToco, we integrate both CoT and few-shot CoT strategies into prompt engineerings to enhance performance.

2.2 Collaboration between Multiple LLMs

Multi-agent Debate

Multi-agent debate has emerged as a widely adopted method for collaboration between multiple LLMs, aiming to leverage collective intelligence through interaction. This concept was first proposed by Du et al. (2023), where multiple agents generate initial responses individually and then iteratively refine them through rounds of critique and feedback from their peers. Liang et al. (2023) extended this framework by introducing a judge to evaluate the contributions of the debaters and determine the outcome. Other studies (Chen et al., 2023; Bakhtin et al., 2022; Wang et al., 2024) emphasize on the strategies of cooperation to enhance overall task-solving capabilities. However, multi-agent debate has primarily been explored using highly capable models such as GPT-4, which are known for their strong self-reflection abilities. In this paper, we investigate how less capable LLMs could effectively collaborate through a communication-free method.

Token-level Integration

Due to the auto-regressive nature of language models, token-level integration has been proposed as a method to achieve consensus during each decoding step for multiple LLMs. Hoang et al. (2023) enhanced translation accuracy through blending the probability distributions generated by a neural machine translation system and an LLM. Similarly, Li et al. (2024) proposed combining an untrusted LLM with a smaller, benign model, adjusting the weight between the two models to address concerns such as copyright infringement and data poisoning.

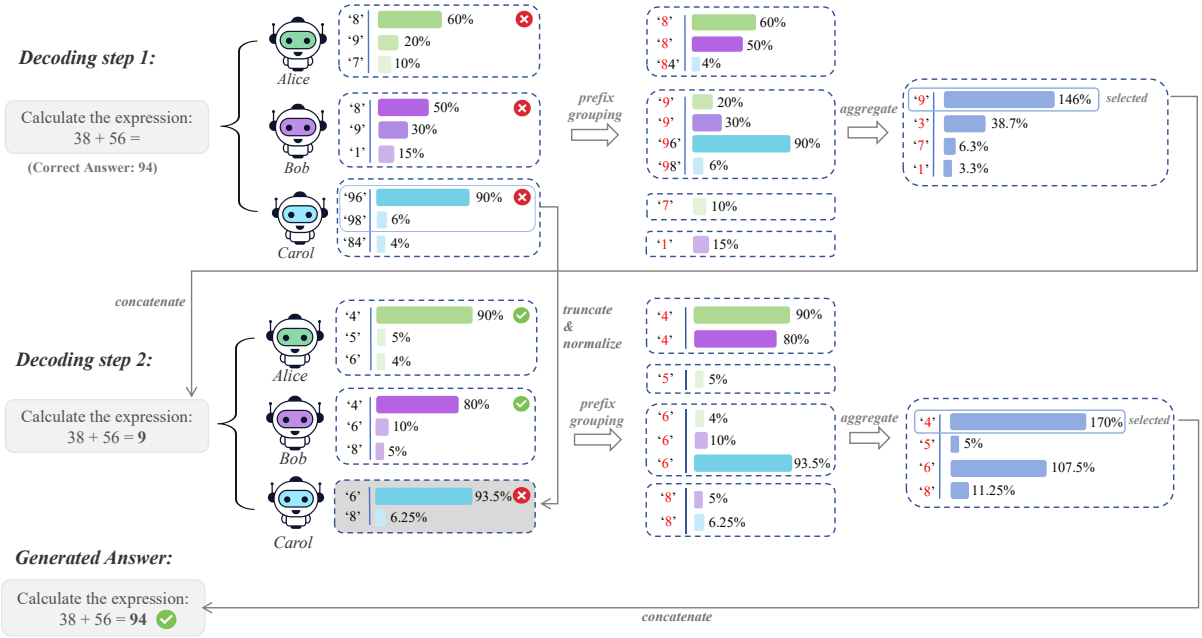

Figure 2: An example of PToco's collaborative generation process. In the first round, three LLMs (named as Alice, Bob and Carol) separately predict the possible next tokens. All tokens are further classified into groups according to prefix relationships (with the common prefix in a group highlighted in red). The longest common prefix('9') of the group with highest cumulative probability(146%) is then selected as the generated token. In the second round, '9' is concatenated into the prompt and the same process repeats. Due to a coarser tokenizer, Carol directly utilizes the predictions from round 1 without performing forward computation (presented in gray shading). The prefix '9' is truncated from '96' and '98', and the corresponding probabilities are normalized to sum to 1 to serve as Carol's predictions for round 2.

However, these methods require all participating models to share the same tokenizer, which limits their flexibility. Other approaches (Shen et al., 2024; Jin et al., 2024) employ dynamic mechanisms to select the model best suited for generating the next token based on their strengths, but this can overlook the potential contributions of other models in the ensemble. To overcome these limitations, we propose a framework that allows all participating LLMs to contribute to token generation at each decoding step, irrespective of differences in their tokenization strategies.

3 PToco: Prefix-based Token-level Collaboration Mechanism

In this section, we introduce PToco - a novel, iterative token-level collaboration mechanism that allows multiple LLMs with different tokenization strategies to work together seamlessly. The whole mechanism is illustrated in detail in Figure 2. By employing a prefix-based grouping method, PToco effectively identifies consensus among tokens of varying granularities generated by different models in each decoding step. Additionally, it incorporates a refined iterative process to prevent misalignment in subsequent decoding steps, ensuring robust and coherent token generation. The PToco approach operates through the following three key stages: synchronized token prediction, prefix grouping and consensus selection. We discuss each stage and the iterative process in detail in the following part.

3.1 Synchronized Token Prediction

Suppose there are n large language models (LLMs) participating in collaboration, each denoted as $M_1, M_2, \ldots, M_n$. Let p be the initial prompt of a query or instruction provided to these models. In the first round of PToco, each model M_i takes p as input and performs forward computation to generate a probability distribution over its own vocabulary V_i for the possible next token t. The probability of token t is given by the softmax function:

$$P_i(t) = \frac{\exp\left(\frac{z_i(t)}{\tau}\right)}{\sum_{v \in V_i} \exp\left(\frac{z_i(v)}{\tau}\right)},$$

where $z_i(t)$ is the logit score for token t from model M_i, and τ is the temperature parameter applied to

control the randomness of the distribution. We set $\tau = 1$ in our experiments, and its impact is discussed further in the ablation part.

For each model, we retain the top-k tokens with the highest probabilities in the distribution (with $k = 5$ in our experiments). Tokens with probabilities below a predefined threshold p_{thres} (set to 5% in our experiments) are further filtered out, leaving a set of j tokens, where $j \leq k$. This process prevents low-probability tokens from introducing noise in the subsequent prefix grouping step, ensuring that only high-probability tokens are kept for further analysis. The retained tokens for each model are presented as:

$$T_i = \{(t_{i,1}, p_{i,1}), \ldots, (t_{i,j}, p_{i,j}) \mid p_{i,j} \geq p_{\text{thres}}\},$$

where $t_{i,j}$ is the j-th token predicted by model M_i, and $p_{i,j}$ is its associated probability.

Finally, the selected tokens from all models are combined to form the set T_{all}, which includes all valid tokens and their corresponding probabilities:

$$T_{all} = \bigcup_i T_i.$$

3.2 Prefix Grouping

A key challenge in integrating multiple LLMs at the token level arises from differences in their tokenization strategies. Tokenizers across LLMs can vary significantly in granularity; some models generate tokens at the word level, while others produce tokens at subword or even character-level granularity. Consequently, the tokens in T_{all} often do not align precisely—tokens may represent different segments of the text even when referring to the same underlying content. This misalignment makes it difficult to directly aggregate the probabilities predicted by the models.

To address this issue, we propose a *prefix grouping* method that leverages the prefix relationships between tokens in T_{all}. Specifically, two tokens - t_a and t_b, are considered to share a prefix relationship if t_a is a prefix of t_b or vice versa. As outlined in Algorithm 1, prefix grouping iteratively organizes tokens from T_{all} into groups based on these relationships. Once the process is complete, the shortest token within each group will also serve as the longest common prefix of all the tokens within that group, representing the consensus across predicted tokens of varying granularities.

Algorithm 1 Prefix Grouping

Input: A list of tokens $T_{all} = [t_1, t_2, \ldots, t_n]$
Output: Groups $G_1, G_2, \ldots, G_m$, where each group contains all tokens linked by prefix relationships.
Initialize $Groups = []$
for each t_i in T_{all} **do**
    Initialize $placed \leftarrow$ False
    for each G_j in $Groups$ **do**
        if t_i has a prefix relation with any token in G_j **then**
            Add t_i to G_j
            $placed \leftarrow True$
            break
        end if
    end for
    if $placed = False$ **then**
        Create new group $G_{\text{new}} = [t_i]$
        add to $Groups$
    end if
end for
Return $Groups$

3.3 Consensus Selection

After the filtered tokens are grouped, the next step is to determine which token will be selected as the final output for this round. For each group G_k, the cumulative probability is calculated by summing up the probabilities over all tokens within that group. The group with the highest cumulative probability is denoted as G_{max}:

$$G_{\text{max}} = \arg\max_{G_k} \sum_{t_{i,j} \in G_k} p_{i,j}.$$

From G_{max}, the shortest token is chosen as the generated token for the current decoding step. This ensures that the generated token is both highly probable and representative of the group's consensus, allowing for more coherent and reliable outputs in multi-model collaboration.

3.4 Iterative Decoding

Following the typical iterative decoding process of LLMs, PToco also operates iteratively after the initial round of token generation. In round $i + 1$, the input prompt p_{i+1} is formed by concatenating the previous prompt p_i with the token generated in round i. However, forcefully providing the concatenated prompt to each model for generating the next token can lead to problems, particularly when

a model expects to generate a complete token in round i but only receives its prefix in round $i+1$. Figure 3 illustrates this issue, where Llama3 fails to predict the correct continuation '4' after being given only the prefix '9'. This inability possibly arises when LLMs are being trained on large-scale corpus. The tokenizer assigns different IDs to the prefix and the full token during preprocessing, causing the model to interpret them as distinct tokens rather than related parts of the same sequence.

To address this, before proceeding to round $i+1$, PToco will first examine the predictions from round i for each model, considering the following three scenarios:

1. If the generated prefix from round i appears as a standalone token in the filtered top-k tokens predicted in round i, PToco directly passes the concatenated input to the model for round $i+1$. In this case, the model continues its prediction smoothly, as it recognizes the prefix as a valid token.

2. If the standalone prefix is absent from the top-k list, but longer tokens starting with the prefix are present, this suggests the model might employ a coarser tokenization strategy. PToco extracts these longer tokens with their corresponding probabilities from round i, removes the prefix, and re-normalizes the probabilities to sum to 1. These adjusted tokens and probabilities then directly serve as the prediction of the model in round $i+1$. In this case, the model essentially "takes a break", relying on the predictions from the previous round without requiring forward computation.

3. If neither the prefix nor any longer tokens beginning with the prefix appear in the top-k list, it indicates a divergence between this model and others in the ensemble. In this case, PToco passes the concatenated input to the model for a fresh token prediction in round $i+1$, aiming to realign the model with the ensemble in subsequent steps.

This mechanism ensures that PToco accommodates models with different tokenization strategies, maintaining coherence and flexibility throughout generation.

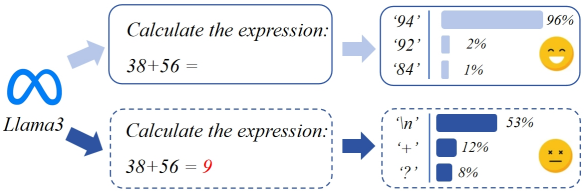

Figure 3: Llama3 loses track of the intended sequence when the input prompt is concatenated with a possible prefix '9'

4 Experiments

4.1 Experimental Setup

4.1.1 Tasks and Datasets

To evaluate the effectiveness of the PToco approach, we conducted experiments across five datasets spanning two categories of reasoning tasks: mathematical reasoning and symbolic reasoning. For each dataset, we report accuracy using exact match between the generated answer and the reference.

(1) Mathematical Reasoning

Add&Mul: A randomly generated dataset involving addition and multiplication tasks. It consists of six subsets, with tasks on 8-digit, 12-digit, 16-digit, and 20-digit addition as well as 2-digit and 3-digit multiplication between two numbers.

AQuA (Ling et al., 2017): A dataset on algebra requiring models to answer multiple-choice questions by reasoning through a set of given options.

GSM8K (Cobbe et al., 2021): A dataset of grade-school level math problems that typically require 2 to 8 reasoning steps to solve.

(2) Symbolic Reasoning

Last Letter Concat: A dataset where the task is to take the last letter of each word in a given four-word phrase and concatenate them into a new word.

WordSort (Srivastava et al., 2022): A dataset which requires sorting a list of 8-10 words alphabetically by comparing characters one by one.

4.1.2 Models and Baselines

To verify whether the proposed method could work efficiently when integrating less capable LLMs, we use four open-source models with parameters less than 10 billion for collaboration. The selected models are Qwen2-7B (Yang et al., 2024), Llama3-8B (Touvron et al., 2023), Gemma2-9B (Team et al., 2024), and GLM4-9B (Yang et al., 2024) (all chat

Models	Last Letter Concat	Word Sort	Add & Mul	AQuA	GSM8k	Avg.
Qwen2-7B	56.7	11.7	68.4	60.6	76.3	54.7
Llama3-8B	68.0	72.3	30.7	44.5	74.7	58.0
GLM4-9B	67.3	85.3	10.2	59.5	81.3	60.7
Gemma2-9B	68.0	73.3	64.5	47.2	80.6	66.7
Llama3-8B & GLM4-9B & Gemma2-9B	**85.3** (+17.3)	**92.7** (+7.4)	58.5 (-6.0)	59.1 (-0.4)	83.1 (+1.8)	75.7 (+9.0)
Qwen2-7B & GLM4-9B & Gemma2-9B	82.0 (+14.0)	89.3 (+4.0)	71.4 (+3.0)	**67.3** (+6.7)	**86.4** (+5.1)	**79.3** (+12.6)
Qwen2-7B & Llama3-8B & Gemma2-9B	84.7 (+16.7)	79.3 (+6.0)	**76.1** (+7.7)	59.1 (-1.5)	86.1 (+5.5)	77.1 (+10.4)
Qwen2-7B & Llama3-8B & GLM4-9B	82.0 (+14.0)	87.3 (+1.7)	59.4 (-9.0)	63.0 (+2.4)	83.3 (+2.0)	74.9 (+8.2)

Table 1: Accuracy comparison across various datasets when integrating 3 LLMs using PToco. The 'Avg.' column shows the average performance for all tests. Performance gains over the baseline are indicated in parentheses.

versions that have undergone instruction-tuning and reinforcement learning from human feedback). Each of these models demonstrates strong performance across various NLP benchmarks, making them ideal candidate for assessing whether PToco can yield improvements beyond their already impressive individual results. Additionally, these models exhibit diverse strengths across the five selected datasets, providing an excellent opportunity to observe the performance of PToco when integrating models with varying capabilities.

We conducted 16 experiments for each dataset, utilizing different combinations of the four models. This involves evaluating each model individually, in pairs, in groups of three, and finally, integrating all four models to explore their collective potential. We report the settings of three-model collaboration in our main results, and further discusses other combinations in section 5.1 as an ablation study on the quantities of participating LLMs.

For the single-model setups, outputs are generated using greedy decoding. For the multi-model collaborative setups, we set the temperature parameter to 1 in the softmax function to obtain the raw probability distribution over the vocabulary. For each combination, we set the baseline according to the highest score achieved by any individual participants within the ensemble. This allows us to assess clearly whether PToco's collaborative framework results in a performance improvement over the best-performing single model.

4.2 Main Results

Table 1 presents the performance of PToco across five datasets when integrating three models per trial. The results show that PToco demonstrates significant improvements over baselines when two or more of the three collaborating models have relatively stronger individual capabilities. We further discuss this conclusion from the following aspects:

(1) PToco shows the highest overall gains when three models with relatively stronger abilities are combined. In Last Letter Concat, the collaboration between the three best-performing models (Llama3, GLM4 and Gemma2) results in an improvement of 17.3% over the baseline, which is the largest increase across all datasets. And the same combination of models achieves the top score of 92.7 among all configurations in Word Sort. Likewise, the combination of Qwen2, GLM4, and Gemma2 delivers the best performance of 86.4 on GSM8k. These results affirm that when models with comparable capabilities collaborate, PToco maximizes their collective strengths, leading to superior overall outcomes across diverse tasks.

(2) When two stronger models are paired with a weaker one, PToco still achieves notable improvement, though the gains are less substantial compared to combinations of three strong ones. For instance, the combination of Qwen2, GLM4, and Gemma2 showed a 3% increase in Add&Mul and a 4% boost in Word Sort, despite GLM4's poor ability in Add&Mul and Qwen2's weak performance in Word Sort. While these results remain below the top-scoring combinations of three strong models, they demonstrate that as long as the stronger models dominate in the ensemble, PToco delivers robust improvements. However, the weaker model still limits the system's full potential, indicating PToco's effectiveness is maximized when models of similar strength collaborate.

(3) The performance of the overall system is constrained when two weaker models are paired with a stronger one, as seen in both the Add&Mul and AQuA datasets. In Add&Mul, when weaker models like Llama3 and GLM4 are combined with a stronger model, such as Qwen2 or Gemma2, the results fail to surpass the baseline set by the only stronger model. Similarly, in the AQuA dataset, the

inclusion of Llama3 and Gemma2 in the collaboration struggles to enhance overall performance. This indicates that the only stronger model is unable to compensate fully for the weaknesses of the other two, and the overall effectiveness diminishes when weaker models outnumber the stronger ones.

5 Ablations and Analysis

5.1 Impact of Model Quantity in Collaboration

In this section, we further evaluate the generalizability of the proposed PToco method when integrating different numbers of LLMs (specifically 2 and 4). The results, as shown in Table 2, along with the results of the 3-model integration (Table 1), demonstrate that PToco consistently provides performance gains under a reasonable selection of participating models. We discuss the cases of 2-model and 4-model integration in detail as below:

(1) The collaboration between two models achieves strong gains when the two models show similar initial strengths. Across the two datasets of Last Letter Concat and GSM8K, the four models—Qwen2, Llama3, GLM4, and Gemma2—all exhibit relatively close scores when evaluated individually. Given these similarities, any combination of two models within this set results in performance improvements on both datasets. In contrast, the overall performance tends to be compromised when two models with notably different abilities collaborate. In Word Sort, Qwen2 performs significantly worse than the other three models. When Qwen2 is combined with another stronger model, the overall score remains lower than the stronger model's individual performance. This shows that PToco's overall performance tends to be dragged down by the weaker model when there are only 2 participants.

(2) When combining all four models together, PToco consistently delivers strong performance across all tasks, achieving the highest average score of 79.7. Given that the individual average scores of the four models are relatively close (ranging from 54.7 to 66.7), this further demonstrates the robustness and effectiveness of PToco when integrating LLMs with similar capabilities.

5.2 Impact of Decoding Temperature

In the proposed PToco method, the temperature applied at every decoding step will affect the probability distribution over the vocabulary predicted by each LLM. To estimate the influence, we select the optimal combination of LLMs (integrating either 2, 3, or 4 models) for each dataset, and apply a range of temperatures between 0.5 and 1.5 during collaboration. As Figure 4 shows, PToco performs best when the temperature is set to 1.0. One possible explanation is that the logits predicted by LLMs remain unscaled under this setting, thus preserving their raw confidence levels for each token without over-amplifying or diminishing.

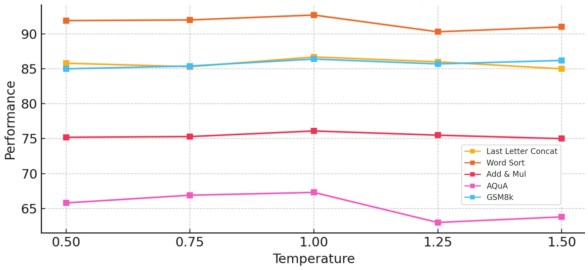

Figure 4: Performance across datasets under different decoding temperature

5.3 Collaboration across Scales

In this part, we aim to demonstrate the effectiveness of the PToco method when combining LLMs of varying scales. Experiments are carried out across six distinct tasks within the Add&Mul dataset. We select the best-performing individual model (Qwen2-7B) and ensemble (Qwen2-7B & Llama3-8B & Gemma2-9B), pairing them both with Llama3-70B for further evaluation.

Table 3 presents the results of these experiments. Notably, Llama3-70B achieves the highest average score when tested individually. When it is further paired with Qwen2-7B, the average score increases by an additional 9.7 points. Moreover, the collaboration between Llama3-70B and the previous best-performing combination yields the highest score of 80.7, marking the best score across all setups. These results confirm that the PToco method is effective when integrating LLMs of differing scales, demonstrating its capacity to enhance outcomes without relying on models of similar size.

6 Conclusion

In this paper, we present PToco, a novel token-level collaboration framework that enables multiple less capable large language models to collaborate effectively. PToco enhances token generation accuracy by leveraging prefix-based grouping and consensus selection mechanisms, allowing models to work to-

Models	Last Letter Concat	Word Sort	Add & Mul	AQuA	GSM8k	Avg.
Qwen2-7B	56.7	11.7	68.4	60.6	76.3	54.7
Llama3-8B	68.0	72.3	30.7	44.5	74.7	58.0
GLM4-9B	67.3	85.3	10.2	59.5	81.3	60.7
Gemma2-9B	68.0	73.3	64.5	47.2	80.6	66.7
Qwen2-7B & Llama3-8B	79.3 (+11.3)	56.7 (-15.6)	65.2 (-3.2)	55.1 (-5.5)	80.0 (+3.7)	67.3 (+0.6)
Qwen2-7B & GLM4-9B	77.3 (+10.0)	80.3 (-5.0)	43.3 (-25.1)	**63.8** (+3.2)	84.6 (+3.3)	69.9 (+9.2)
Qwen2-7B & Gemma2-9B	83.3 (+15.3)	62.3 (-11.0)	72.5 (+4.1)	60.2 (-0.4)	84.5 (+3.9)	72.6 (+5.9)
Llama3-8B & GLM4-9B	76.7 (+8.7)	89.0 (+3.7)	32.1 (+1.4)	55.1 (-4.4)	82.4 (+1.1)	67.1 (+6.4)
Llama3-8B & Gemma2-9B	80.7 (+12.7)	84.3 (+11.0)	64.2 (-0.3)	50.0 (+2.8)	80.8 (+0.2)	71.2 (+4.5)
GLM4-9B & Gemma2-9B	80.0 (+12)	**91.0** (+5.7)	43.9 (-20.6)	62.2 (+2.7)	84.9 (+3.6)	72.4 (+5.7)
All 4 Models	**86.7** (+18.7)	90.0 (+4.7)	**74.9** (+6.5)	61.8 (+1.2)	**85.1** (+3.8)	**79.7** (+13)

Table 2: Accuracy comparison across various datasets when integrating 2 and 4 LLMs using PToco

Models	Add-8	Add-12	Add-16	Add-20	Mul-2	Mul-3	Avg.
Qwen2-7B	92.0	89.1	73.7	68.0	72.9	14.5	68.4
Llama3-8B	81.8	47.4	16.8	3.7	24.7	10.0	30.7
Gemma2-9B	90.9	74.3	64.2	45.7	96.6	15.5	64.5
Llama3-70B	92.5	86.2	79.4	49.5	82.4	22.2	68.7
Qwen2-7B & Llama3-70B	96.2	94.1	**88.7**	80.0	87.5	24.1	78.4
Qwen2-7B & Llama3-8B & Gemma2-9B & Llama3-70B	**97.5**	**94.5**	87.5	**81.5**	**96.7**	**26.4**	**80.7**

Table 3: Accuracy comparison when integrating models of different scales on the Add&Mul dataset. Add-8 represents addition of two 8-digit numbers. Mul-2 represents multiplication of two 2-digit numbers.

gether without requiring identical tokenizers. Our comprehensive experiments across multiple reasoning tasks show that PToco consistently improves performance under a reasonable selection of participating LLMs. Additionally, PToco demonstrates flexibility across different model scales and quantities. Overall, PToco offers a powerful solution to enhance reasoning in multi-LLM ensembles, driving substantial improvements in task-solving capabilities.

7 Limitations

While collaborating through PToco significantly improves performance, there are several limitations.

Firstly, PToco's performance is influenced by the initial strengths of the participating models. When weaker models dominate the ensemble, the overall performance tends to decline. This necessitates preliminary testing to assess the individual performance of each model.

In addition, PToco requires real-time access to token-level probabilities at each decoding step. This restriction limits PToco's applicability to only open-source LLMs.

Lastly, collaborative reasoning increases both memory and time overhead. The need to process multiple models simultaneously, aggregate token-level predictions, and manage prefix-based grouping adds computational complexity, leading to higher GPU memory usage and longer inference times compared to using a single model.

Acknowledgments

We thank the anonymous reviewers for their insightful comments. This paper was supported by the Shanghai Sailing Program (No. 23YF1409400), National Natural Science Foundation of China (No. 62306112), and Shanghai Pilot Program for Basic Research (No. 22TQ1400100-20).

References

Anton Bakhtin, Noam Brown, Emily Dinan, Gabriele Farina, Colin Flaherty, Daniel Fried, Andrew Goff, Jonathan Gray, Hengyuan Hu, et al. 2022. Human-level play in the game of diplomacy by combining language models with strategic reasoning. *Science*, 378(6624):1067–1074.

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, et al. 2024. Graph of thoughts: Solving elaborate problems with large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 17682–17690.

Weize Chen, Yusheng Su, Jingwei Zuo, Cheng Yang, Chenfei Yuan, Chi-Min Chan, Heyang Yu, Yaxi Lu, Yi-Hsin Hung, Chen Qian, et al. 2023. Agentverse: Facilitating multi-agent collaboration and exploring emergent behaviors. In *The Twelfth International Conference on Learning Representations*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B Tenenbaum, and Igor Mordatch. 2023. Improving factuality and reasoning in language models through multiagent debate. *arXiv preprint arXiv:2305.14325*.

Hieu Hoang, Huda Khayrallah, and Marcin Junczys-Dowmunt. 2023. On-the-fly fusion of large language models and machine translation. *arXiv preprint arXiv:2311.08306*.

Jiaxin Huang, Shixiang Shane Gu, Le Hou, Yuexin Wu, Xuezhi Wang, Hongkun Yu, and Jiawei Han. 2022. Large language models can self-improve. *arXiv preprint arXiv:2210.11610*.

Jie Huang and Kevin Chen-Chuan Chang. 2022. Towards reasoning in large language models: A survey. *arXiv preprint arXiv:2212.10403*.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, et al. 2023. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *arXiv preprint arXiv:2311.05232*.

Lifeng Jin, Baolin Peng, Linfeng Song, Haitao Mi, Ye Tian, and Dong Yu. 2024. Collaborative decoding of critical tokens for boosting factuality of large language models. *arXiv preprint arXiv:2402.17982*.

Tianlin Li, Qian Liu, Tianyu Pang, Chao Du, Qing Guo, Yang Liu, and Min Lin. 2024. Purifying large language models by ensembling a small language model. *arXiv preprint arXiv:2402.14845*.

Tian Liang, Zhiwei He, Wenxiang Jiao, Xing Wang, Yan Wang, Rui Wang, Yujiu Yang, Zhaopeng Tu, and Shuming Shi. 2023. Encouraging divergent thinking in large language models through multi-agent debate. *arXiv preprint arXiv:2305.19118*.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. Program induction by rationale generation: Learning to solve and explain algebraic word problems. *arXiv preprint arXiv:1705.04146*.

Jieyi Long. 2023. Large language model guided tree-of-thought. *arXiv preprint arXiv:2305.08291*.

Zheheng Luo, Qianqian Xie, and Sophia Ananiadou. 2023. Chatgpt as a factual inconsistency evaluator for text summarization. *arXiv preprint arXiv:2303.15621*.

OpenAI. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Vipula Rawte, Amit Sheth, and Amitava Das. 2023. A survey of hallucination in large foundation models. *arXiv preprint arXiv:2309.05922*.

Shannon Zejiang Shen, Hunter Lang, Bailin Wang, Yoon Kim, and David Sontag. 2024. Learning to decode collaboratively with multiple language models. *arXiv preprint arXiv:2403.03870*.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, et al. 2022. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *arXiv preprint arXiv:2206.04615*.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, et al. 2024. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Karthik Valmeekam, Alberto Olmo, Sarath Sreedharan, and Subbarao Kambhampati. 2022. Large language models still can't plan (a benchmark for llms on planning and reasoning about change). In *NeurIPS 2022 Foundation Models for Decision Making Workshop*.

Qineng Wang, Zihao Wang, Ying Su, Hanghang Tong, and Yangqiu Song. 2024. Rethinking the bounds of llm reasoning: Are multi-agent discussions the key? *arXiv preprint arXiv:2402.18272*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, et al. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2024. Tree of thoughts: Deliberate problem solving with large language models. *Advances in Neural Information Processing Systems*, 36.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*.

Ce Zhou, Qian Li, Chen Li, Jun Yu, Yixin Liu, Guangjing Wang, Kai Zhang, Cheng Ji, Qiben Yan, Lifang He, et al. 2023. A comprehensive survey on pretrained foundation models: A history from bert to chatgpt. *arXiv preprint arXiv:2302.09419*.