Multi-LLM Text Summarization

Jiangnan Fang[1], Cheng-Tse Liu[2], Jieun Kim[3], Yash Bhedaru[4], Ethan Liu[5], Nikhil Singh[6], Nedim Lipka[7], Puneet Mathur[7], Nesreen K. Ahmed[8], Franck Dernoncourt[7], Ryan A. Rossi[7], Hanieh Deilamsalehy[7]

[1,2,3,4,5,6]University of California, Santa Cruz, [7]Adobe Research, [8]Cisco Research

{[1]jfang53, [2]cliu282, [3]jkim716, [5]eliu25}@ucsc.edu

Abstract

In this work, we propose a Multi-LLM summarization framework, and investigate two different multi-LLM strategies including centralized and decentralized. Our multi-LLM summarization framework has two fundamentally important steps at each round of conversation: generation and evaluation. These steps are different depending on whether our multi-LLM decentralized summarization is used or centralized. In both our multi-LLM decentralized and centralized strategies, we have k different LLMs that generate diverse summaries of the text. However, during evaluation, our multi-LLM centralized summarization approach leverages a single LLM to evaluate the summaries and select the best one whereas k LLMs are used for decentralized multi-LLM summarization. Overall, we find that our multi-LLM summarization approaches significantly outperform the baselines that leverage only a single LLM by up to 3x. These results indicate the effectiveness of multi-LLM approaches for summarization.

1 Introduction

Large language models (LLMs) have been shown to have the potential to produce high-quality summaries (Chowdhery et al., 2022; Zhang et al., 2023; Goyal et al., 2023; Pu et al., 2023b). However, despite the remarkable progress in LLM-based summarization, limitations still exist for documents where useful information may be sparsely distributed throughout the text. Research by (Liu et al., 2023) highlights that a naive application of LLMs may overlook critical details or fail to grasp the holistic meaning of a document, indicating the need for more refined methods.

To address this, recent efforts have explored prompt-engineering techniques to guide LLMs towards producing better summaries (Adams et al., 2023). These techniques, while promising, still face limitations in consistently delivering high-quality summaries across different document types and structures. We show that by combining the capabilities of multiple models with a diverse set of knowledge bases it's possible to achieve more robust summaries across domains.

Summary of Main Contributions.

- We propose the first framework for multi-LLM text summarization and investigate two topologies: centralized and decentralized.
- We find that multi-LLM text summarization often performs better than using a single LLM for summarization, and we show that the best performing method in the framework aligns with human judgments.
- We conduct experiments on how prompting, number of LLMs, and various combinations of generating and evaluating LLMs can affect quality of summaries in the multi-LLM setup.

2 Related Work

2.1 Summarization

Recent advancements in summarization have increasingly leveraged large language models (LLMs), moving beyond fine-tuned transformer models like Pegasus, BART, and T5. Studies consistently show that LLMs can generate summaries with higher coherence, relevance, and factual accuracy, often rivaling or surpassing human-written summaries (Goyal et al., 2023; Zhang et al., 2023; Pu et al., 2023b).

For example, Goyal et al. (2023) demonstrated that GPT-3 (text-davinci-002) produced summaries preferred by human evaluators over fine-tuned models like Pegasus and BRIO on structured datasets such as CNN/DM (Nallapati et al., 2016) and XSUM (Narayan et al., 2018). Similarly, Zhang et al. (2023) emphasized the importance of instruction tuning in achieving superior zero-shot perfor-

Proceedings of Recent Advances in Natural Language Processing,pages 352–362
Varna, Sep 8–10, 2025
https://doi.org/10.26615/978-954-452-098-4-043

mance for summarization tasks. Pu et al. (2023b) further highlighted improved factual consistency and reduced hallucinations when using LLMs.

While these studies validate the potential of LLMs in summarizing well-structured texts, they may falter for inputs lacking clear structural cues and exhibiting greater complexity. In Keswani et al. (2024), semantic clustering and multi-stage summarization with LLaMA2 are used to manage lengthy inputs, and in Chhibbar and Kalita (2024), middle truncation, "skimming", and redundancy removal show better performance for texts longer than 70,000 words. However, some past approaches rely on predefined hierarchical processing strategies that oversimplify the nuanced relationships within the text. Moreover, as Liu et al. (2023) noted, LLMs tend to neglect content from the middle sections of longer documents, resulting in incomplete or unbalanced summaries.

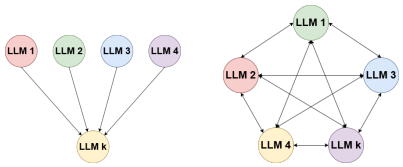
Our work aims to improve performance for both long and short text summarization, and it builds upon aforementioned foundations by proposing a multi-LLM framework designed to overcome these shortcomings through information exchange and collaborative synthesis.

2.2 Multi-LLM

The concept of leveraging multiple LLMs collaboratively has gained traction in recent research, particularly for tasks requiring complex reasoning and factual accuracy. For instance, Liang et al. (2024) introduced the Multi-Agent-Debate (MAD) framework, where LLMs engage in iterative debates to refine their reasoning. This framework demonstrated that a multi-agent GPT-3.5-Turbo setup outperformed GPT-4 on reasoning datasets. Similarly, Chen et al. (2024) proposed RECONCILE, a framework where LLMs collaboratively refine answers and explanations, achieving significant improvements over single-agent systems. Li et al. (2024) extended this line of research by optimizing agent connections, showing that sparse networks can maintain performance while reducing computational overhead.

Although these studies reveal the potential of multi-LLM approaches, their focus remains on structured reasoning tasks, such as question answering and fact-checking. They have not been adequately explored in the context of synthesizing distributed information, addressing content imbalances, and preserving the coherence of summaries across extended texts.

We hope to bridge this gap by adapting multi-LLM frameworks to the domain of document summarization, addressing limitations of both single LLM and traditional hierarchical techniques, and positioning multi-LLM summarization as a promising solution.



(a) Centralized (b) Decentralized

Figure 1: Centralized and Decentralized approaches using a 5-LLM example. Similar topologies can be applied to any ("k") number of LLMs. In centralized interactions, all models communicate with a central model; in decentralized interactions, each model communicate with every other model and also itself.

3 Multi-LLM Summarization Framework

In this work, we propose a novel multi-LLM summarization framework that leverages multiple large language models to enhance summarization quality of long document input. Through the distribution of generation and evaluation of candidate summaries across multiple models, our framework aims to provide better summaries than single LLM methods, leveraging expertise from different models. We present two interaction topologies, **centralized** and **decentralized**, to guide the collaboration, evaluation, and refinement of summaries between LLMs. Visually these two methods can be represented at a high level in Figure 1. In the datasets we test, articles are typically tens of thousands of words long and exceed the context window of most standard LLMs. To handle this, we establish a two stage process that involves chunking the source document, independently summarizing each chunk of the source document, and then applying a second round of chunking and summarization on the concatenated intermediate results. Throughout both these stages, both frameworks allow multiple LLMs to collaborate and converge on a single final high quality summary of the entire original reference document. Table 1 provides an overview of our framework's four main variations.

Multi-LLM Summarization Framework	General Mechanism	Stage
CENTRALIZED (Sec. 4)	Single-Round (Sec. 4.1)	Generation (§ 4.1) Evaluation (§ 4.1)
	Conversational (Sec. 4.2)	Generation (§ 4.2) Evaluation (§ 4.2)
DECENTRALIZED (Sec. 5)	Single-Round (Sec. 5.1)	Generation (§ 5.1) Evaluation (§ 5.1)
	Conversational (Sec. 5.2)	Generation (§ 5.2) Evaluation (§ 5.2)

Table 1: Overview of Multi-LLM Framework (Sections 4-5).

Algorithm 1 Centralized Multi-LLM Summary

Require: ordered set $\mathcal{S} = \{S_1, \ldots, S_m\}$ of summaries, set $\mathcal{M} = \{M_1, \ldots, M_k\}$ of k LLMs, a central agent $C \in \mathcal{M}$, max number of conversational rounds $t_{\max}$, initial summarization prompt P (*e.g.*, Figure 2), evaluation prompt P_ec (*e.g.*, Figure 5) for centralized version
Ensure: summary S^* of the text
$S = \text{CREATESUMMARY}(\mathcal{S})$
for $i = 1$ to $t_{\max}$ **do** ▷ conversation rounds
  for each model $M_j \in \mathcal{M}$ **do**
    $S_j^{(i)} = M_j(P, S)$
  Let $\mathcal{S}_i = \{S_1^{(i)}, S_2^{(i)}, \ldots, S_k^{(i)}\}$
  $E^{(i)} = C(P_{ec}, \mathcal{S}_i)$
  $\mathbf{r} = \text{AGGRRESULTS}(E^{(i)})$
  $j \leftarrow \text{argmax}_{M_j \in \mathcal{M}} r_j$
  Set $S^* \leftarrow S_j^{(i)}$
  if $\text{CONVERGED}(\mathbf{r})$ **then return** S^*
  Set P to prompt in Figure 3.

Algorithm 2 Decentralized Multi-LLM Summary

Require: ordered set $\mathcal{S} = \{S_1, \ldots, S_m\}$ of summaries, set $\mathcal{M} = \{M_1, \ldots, M_k\}$ of k LLMs, max number of conversational rounds $t_{\max}$, initial summarization prompt P (*e.g.*, Figure 2), evaluation prompt P_e (*e.g.*, Figure 4)
Ensure: summary S^* of the text
$S = \text{CREATESUMMARY}(\mathcal{S})$
for $i = 1$ to $t_{\max}$ **do** ▷ conversation rounds
  for each model $M_j \in \mathcal{M}$ **do**
    $S_j^{(i)} = M_j(P, S)$
  Let $\mathcal{S}_i = \{S_1^{(i)}, S_2^{(i)}, \ldots, S_k^{(i)}\}$
  for each model $M_j \in \mathcal{M}$ **do**
    $E_j^{(i)} = M_j(P_e, S_1^{(i)}, \ldots, S_k^{(i)})$
  Set $\mathcal{E}_i = \{E_1^{(i)}, E_2^{(i)}, \ldots, E_k^{(i)}\}$
  $\mathbf{r} = \text{AGGRRESULTS}(E_1^{(i)}, \ldots, E_k^{(i)})$
  $j \leftarrow \text{argmax}_{M_j \in \mathcal{M}} r_j$
  Set $S^* \leftarrow S_j^{(i)}$
  if $\text{CONVERGED}(\mathbf{r})$ **then return** S^*
  Set P to prompt in Figure 3.

4 Centralized Multi-LLM Summarization

The steps for centralized summarization can be found in Algorithm 1. This method leverages multiple LLMs to generate candidate summaries and uses a central LLM to evaluate their quality and guide iterative refinements.

```
Provide a concise summary of the text in
around 160 words. Output the summary text
only and nothing else.

                [text]
```

Figure 2: Prompt for generating the initial summary in the first round.

```
Given the original text below, along with
the summaries of that text by [k] LLMs,
please generate a better summary of the
original text in about 160 words.

ORIGINAL:

                [text]

Summary by M_1:

            [LLM 1's summary]

                  ⋮

Summary by M_k:

            [LLM k's summary]
```

Figure 3: Generation prompt that is used after the initial round of conversation among the multiple LLMs. Note that the above prompt is for generating the final summary, however, for the chunk-level generation, it would just be the actual chunk.

4.1 Single Round

In the simplest case, we prompt each LLM once, gather their summaries, and then perform a single evaluation step to select the best final summary. This is the initial process before we extend it to multiple rounds.

```
Given the original text below, along with
the summaries of that text by [k] agents,
please evaluate the summaries and output the
name of the agent that has the best summary.
Output the exact name only and nothing else.

ORIGINAL:

 [chunk or concatenated chunk summaries S]

Summary by agent_1:

             [LLM 1's summary]
                    ⋮

Summary by agent_k:

             [LLM k's summary]
```

Figure 4: Evaluation prompt for evaluating the summaries generated by different LLMs using our conversational (decentralized) multi-LLM framework. "k" is a parameter reflecting the number of LLMs that generate summaries.

```
Given the initial text below, along with the
summaries of that text by [k] LLMs, please
evaluate the generated summaries and output
the name of the LLM has the best summary. On
a separate line indicate a confidence level
between 0 and 10.
ORIGINAL:

                  [text]

Summary by M_1:

             [LLM 1's summary]
                    ⋮

Summary by M_k:

             [LLM k's summary]

Remember, on a separate line indicate a
confidence level between 0 and 10
```

Figure 5: Evaluation prompt for evaluating the summaries generated using our conversational (centralized) multi-LLM framework. More specifically, we have added an instruction for centralized multi-LLM summarization approach that in addition to providing the best summary, it also outputs the confidence level between 0 and 10. "k" is a parameter reflecting the number of summary-generating LLMs.

Generation Phase: In the single-round setting, each LLM from the list of participating models $\mathcal{M} = \{M_1, \ldots, M_k\}$ independently generates a summary of the same input text using a common prompt P. The prompt P is illustrated in Figure 2.

```
Provide a concise summary of the text in
around 160 words. Output the summary text
only and nothing else.

     [concatenated chunk summaries S]
```

Figure 6: Generation prompt for generating the final summary from the summarized chunks using our conversational (decentralized) multi-LLM framework. This prompt is the same as the one for the initial summary.

Formally, for each LLM $M_j \in \mathcal{M}$, the output is $S_j = M_j(P, S)$ where S represents the input text. Running this step for all M_j yields a set of summaries $\mathcal{S} = \{S_1, \ldots, S_k\}$.

This initial generation stage corresponds to line 4 of Algorithm 1. Conceptually, each model contributes its unique perspective, leading to a diverse pool of candidate summaries, which is important for robust summary selection in the following evaluation phase.

Evaluation Phase: After collecting the set of candidate summaries $\mathcal{S}$, we select a central agent $C \in \mathcal{M}$ to evaluate these summaries. The central LLM C uses an evaluation prompt P_{ec}, as shown in Figure 5, to assess the quality of each summary. To reduce potential bias arising from authorship attribution, we use anonymized identifiers for summaries like `agent_1`, `agent_2`, etc. during evaluation.

Formally, we obtain $E = C(P_{ec}, \mathcal{S})$, where E is the central LLM's evaluation of all candidate summaries. This includes the choice for the best summary (expressed as its anonymized identifier) and a confidence score for that evaluation (expressed as an integer from 0 to 10), denoted together as $\mathbf{r} = \textsc{AggrResults}(E)$ in Algorithm 1. We deanonymize the identifier to recover the text of the selected summary S_j and set this as our final output S^*. In the single-round regime, this terminates the process as no further iterations are performed.

In the evaluation prompt, we include the prompt to output a confidence score so there is a variable on which to impose a stopping condition. This allows us to extend the centralized process to multiple rounds of generation and evaluation using that condition. This process is explained in subsequent sections.

4.2 Conversational

In the conversational approach, we repeat the generation and evaluation phases multiple times. We

define each generation-evaluation process as one round and define conditions under which the process ends or a new round should begin, up to a maximum number of rounds.

Generation Phase: The first round of the conversational approach mirrors the single-round procedure (Section 4.1). Each LLM M_j generates an initial summary $S_j^{(1)} = M_j(P, S)$ from the original input text S using the prompt P. If the evaluation result from the previous round has a confidence score less than the threshold or, if the LLM fails to output a readable confidence score, the pipeline proceeds to the next round. For the second and subsequent rounds, we use the prompt $P^{(i)}$, shown in Figure 3. LLMs in the second and subsequent rounds have access to both the text to be summarized and summaries from the previous round. Concretely, in round $i > 1$ we have $S_j^{(i)} = M_j(P^{(i)}, S)$. The hope is that LLM is able to iteratively improve summarization based upon previous outputs from itself and other models.

Evaluation Phase: The evaluation phase in round $i > 1$ is conceptually similar to the single-round setting (Section 4.1), but now operates on candidate summaries generated immediately before in the generation phase $\mathcal{S}_i = \{S_1^{(i)}, \ldots, S_k^{(i)}\}$. The central LLM C evaluates these candidates like so: $E^{(i)} = C(P_{ec}, \mathcal{S}_i)$, where P_{ec} is the prompt. If the confidence level meets the threshold, the process terminates, and the summary chosen by the central LLM is accepted as S^*. Otherwise, we proceed to the next round of summary generation and evaluation. For the confidence scores we have chosen the range 0-10 as it is fine-grained but also is one of the most common rating scales.

5 Decentralized Multi-LLM Summarization

In Section 4 we introduced the summarization procedure for centralized approach. We now extend the paradigm for the evaluator as well. In the decentralized approach, multiple LLMs also participate in the evaluation process with the hope that a best summary decided on consensus is more robust compared to a single model's decision.

5.1 Single Round

Generation Phase: Generation procedure is the same as that in the centralized approach described in Section 4.1. As before, multiple LLMs independently generate summaries for the input text, obtaining the list of summaries $\mathcal{S} = \{S_1, \ldots, S_k\}$.

Evaluation Phase: For evaluation, each model that authored a summary is prompted with a new evaluation prompt (Figure 4) which does not include a confidence level and receives the text to be summarized along with summaries authored by all agents including itself. More formally, model preferences $E_1^{(i)}, \ldots, E_k^{(i)}$ are collected, where each $E_j^{(i)}$ represents model M_j's choice of the best summary among $S_1^{(i)}, \ldots, S_k^{(i)}$. These preferences are aggregated into a result vector $\mathbf{r} \in 1, \ldots, k^k$, where each element r_j indicates which model's summary was chosen by model M_j. Convergence is achieved when a majority of models select the same summary, formally expressed as $\exists m \in 1, \ldots, k : |j : r_j = m| > \frac{k}{2}$. [1] When no majority choice emerges, the single-round approach ($t_{\max} = 1$) the algorithm selects the summary from a designated tie-breaker model M_t, where $t \in 1, \ldots, k$. Since the tie-breaker model can be any model in the multi-LLM setup, we run experiments with different choices of evaluator and tie-breaking models. Formally, the final summary S^* is determined as:

$$S^* = \begin{cases} S_m^{(1)} & \text{if } \exists m : |\{j : E_j^{(1)} = m\}| > \frac{k}{2} \\ S_t^{(1)} & \text{if } \max_l |\{j : E_j^{(1)} = l\}| \leq \frac{k}{2} \end{cases}$$

where $m \in 1, \ldots, k : |j : r_j = m| > \frac{k}{2}$.

5.2 Conversational

The conversational approach extends the decentralized framework by introducing multiple rounds of generation and evaluation phases. Each generation-evaluation cycle constitutes a round, with iterations continuing until either consensus is achieved or a maximum number of rounds ($t_{\max}$) is reached.

Generation Phase: Generation follows the methodology in Section 4.1, producing the set of summaries $\mathcal{S} = S_1, \ldots, S_k$. A key distinction from the single-round approach lies in the conditional regeneration mechanism: when consensus fails in the first round, subsequent rounds use a new prompt (Figure 3) which includes generated summaries from previous evaluations.

Evaluation Phase: The first round of evaluation is identical to that in the single-round approach, but enters additional rounds with new generation

[1] Here our implementation requires votes exceeding absolute majority for a summary to be immediately selected. In the case of 2 LLMs, this is equivalent to a unanimous decision because one vote does not satisfy absolute majority.

prompts. Formally, let $E_j^{(i)}$ represent model M_j's choice in round i. In the single-round case, non-consensus (when $\max_m |\{j : E_j^{(i)} = m\}| \leq \frac{k}{2}$) triggers an immediate fallback to a tie-breaker model. In contrast, the conversational approach initiates a new generation-evaluation round with an updated prompt (Figure 3). This process continues until either a majority consensus emerges or $t_{\max}$ rounds are exhausted. After $t_{\max}$ rounds without a consensus, the algorithm defaults to the tie-breaker mechanism described in Section 5.1.

6 Experiments

6.1 Experimental Setup

We use test sets of ArXiv (Cohan et al., 2018) and GovReport (Huang et al., 2021) to evaluate our summarization methods. Due to limited resources, we select only the first 20% or the first 1,288 documents from ArXiv. These documents range from 241 to 44,489 words long, averaging 5,950 words. Their summaries range from 46 to 290 words long, averaging 164 words. The GovReport dataset contains 973 articles ranging from 396 to 31,371 words long, averaging 7,379 words long, while their summaries range from 67 to 1,363 words long, averaging 571 words. We assess the quality of LLM-generated summaries using ROUGE-1, ROUGE-L, BLEU-1, and BLEU-4 metrics. For comparison with our multi-LLM approach, unless otherwise mentioned, we leverage GPT-3.5, GPT-4o, GPT-4o mini, and LLaMA3-8B as baselines. For these models, we perform the same chunking across all models, and the summarization prompt is identical to that in the first round of the multi-LLM process (Figure 6). Unless otherwise mentioned, all models use 4K-character chunk-size, and the final summary represents a concatenation of the generated summaries. Finally, unless otherwise mentioned, we set $W = 160$ for all the models.

6.2 Main Results

Our multi-LLM framework outperforms single-LLM baselines by up to 3×, as seen in Table 2. The fact that both precision- and recall-focused metrics improved means the multi-LLM approach is robust. On average the centralized method improves the scores by 73%, and the decentralized method outperforms baselines by 70%.

We see that additional rounds of generation and evaluation do not further improve scores. This shows that even with just 2 LLMs and a single round of generation and evaluation we observe performance gains, meaning that the least costly version of the multi-LLM system is still able to deliver better summaries compared to single-LLM approaches. We suspect a reason multiple rounds of summaries do not outperform single round summaries is that models tested perform relatively consistently for an input text while a multi-round approach absent conversation history (and therefore a model's previous summaries) relies on fluctuations in model performance to produce better summaries that is selected by the judge LLM.

6.3 Ablation Studies

We also assess the performance of the multi-LLM framework with alternative setups, which again produce competitive results compared to the first decentralized and centralized setup and higher scores than single-LLM baselines. It shows that our proposed framework applies to these setups as well.

Varying Model Combinations: In Table 2 we use GPT-3.5 and GPT-4o mini as the participating models in the multi-LLM framework. We further experiment with alternative combinations of models in the framework. As shown in Table 3 we again observe improvements across the board compared to the single-LLM baselines in Table 2, regardless of default model and number of rounds and type of interaction (decentralized vs. centralized).

Varying the Number of LLMs: In this experiment we use 3 LLMs in the setup instead of 2. We observe a 54% improvement for the decentralized method and 59% for the centralized method on average over single-LLM summaries, and for individual scores we see improvements of up to 2.9×. More detailed results are presented in Table 4.

While the 3-LLM system still outperform the single-LLM baseline, increasing the number of LLMs from 2 to 3 does not improve performance upon the 2-LLM system, contrary to the trend observed in the previous sections where 2-LLM system outperform single-LLM baselines.

We offer two possible explanations for this finding. First, adding an additional LLM increases the complexity of the pipeline, which may lead to propagation of noise or redundancy in intermediate summaries. This added complexity could dilute the strengths of individual LLMs and reduce overall coherence and relevance in the final output. Second, the integration of a third LLM introduces a greater risk of inconsistencies in summarization

		ArXiv				GovReport			
		ROUGE-1 ↑	ROUGE-L ↑	BLEU-1 ↑	BLEU-4 ↑	ROUGE-1 ↑	ROUGE-L ↑	BLEU-1 ↑	BLEU-4 ↑
	LLaMA3-8B	0.180	0.106	0.084	0.021	0.403	0.177	0.242	0.079
	GPT-3.5	0.193	0.114	0.093	0.026	0.390	0.178	0.226	0.084
	GPT-4o mini	0.217	0.118	0.108	0.020	0.384	0.156	0.224	0.058
	GPT-4o	0.165	0.095	0.073	0.015	0.372	0.155	0.211	0.059
Decentralized	Multi-LLM 3 round max	0.313	0.163	0.200	0.029	0.447	0.180	0.458	0.098
	Multi-LLM 1 round max	**0.339**	**0.180**	**0.224**	**0.043**	0.468	0.190	0.477	0.112
Centralized	Multi-LLM 3 round max	0.329	0.168	0.217	0.031	0.468	0.189	0.470	0.109
	Multi-LLM 1 round max	0.333	0.173	0.219	0.036	**0.479**	**0.197**	**0.485**	**0.121**

Table 2: Results for the **decentralized** and **centralized** Multi-LLM approaches. For the multi-LLM pipelines participating models are GPT-3.5 and GPT-4o mini. The results use GPT-3.5 for the evaluator in the centralized approach, and summaries from GPT-3.5 are chosen in tie-breaking for both centralized and de-centralized approaches.

	Max Rounds	Multi-LLM Model Combination	ROUGE-1 ↑	ROUGE-L ↑	BLEU-1 ↑	BLEU-4 ↑
Decentralized	**3 Rounds**	**GPT-3.5** & GPT-4o mini	**0.313**	**0.163**	**0.200**	**0.029**
		GPT-4o & **GPT-3.5**	**0.313**	0.159	0.197	0.025
		GPT-4o & **GPT-4o mini**	0.302	0.152	0.185	0.022
	1 Rounds	**GPT-3.5** & GPT-4o mini	**0.339**	**0.180**	**0.224**	**0.043**
		GPT-4o & **GPT-3.5**	0.328	0.170	0.212	0.033
		GPT-4o & **GPT-4o mini**	0.305	0.153	0.189	0.023
Centralized	**3 Rounds**	**GPT-3.5** & GPT-4o mini	**0.329**	**0.168**	**0.217**	**0.031**
		GPT-4o & **GPT-3.5**	0.325	0.166	0.214	0.029
		GPT-4o & **GPT-4o mini**	0.304	0.153	0.188	0.022
	1 Rounds	**GPT-3.5** & GPT-4o mini	0.333	**0.173**	0.219	**0.036**
		GPT-4o & **GPT-3.5**	**0.339**	0.177	**0.228**	0.039
		GPT-4o & **GPT-4o mini**	0.306	0.155	0.190	0.022

Table 3: Varying the combination of models in our Multi-LLM approaches. Note rounds is the max number of rounds allowed and all results are for ArXiv. Bolded numbers are best scores for each round-model combination. Underlined numbers are overall best scores for each metric in this table. Furthermore, the central LLM is highlighted in blue and for the decentralized multi-LLM approaches, we highlight the LLM used for tie-breaking in green.

			ArXiv				GovReport			
			ROUGE-1 ↑	ROUGE-L ↑	BLEU-1 ↑	BLEU-4 ↑	ROUGE-1 ↑	ROUGE-L ↑	BLEU-1 ↑	BLEU-4 ↑
2-LLMs GPT-3.5 Evaluator	**Decentralized**	3 rounds	0.313	0.163	0.200	0.029	0.447	0.180	0.458	0.098
		1 rounds	**0.339**	**0.180**	**0.224**	**0.043**	**0.468**	**0.190**	**0.477**	**0.112**
	Centralized	3 rounds	0.329	0.168	0.217	0.031	0.468	0.189	0.470	0.109
		1 rounds	**0.333**	**0.173**	**0.219**	**0.036**	**0.479**	**0.197**	**0.485**	**0.121**
3-LLMs GPT-4o mini Evaluator	**Decentralized**	3 rounds	**0.301**	**0.154**	0.184	**0.024**	**0.445**	0.178	**0.449**	**0.095**
		1 rounds	0.299	0.152	0.184	0.023	0.442	0.178	0.447	0.094
	Centralized	3 rounds	0.300	**0.153**	0.185	0.023	**0.443**	0.178	0.447	**0.094**
		1 rounds	0.300	0.152	**0.186**	0.023	0.442	0.178	**0.449**	0.093
3-LLMs GPT-3.5 Evaluator	**Decentralized**	3 rounds	0.300	0.154	0.184	0.024	0.446	0.179	0.443	0.094
		1 rounds	**0.309**	**0.159**	**0.193**	**0.027**	**0.451**	**0.182**	**0.459**	**0.099**
	Centralized	3 rounds	0.294	0.151	0.177	0.023	0.451	0.181	0.440	0.095
		1 rounds	**0.329**	**0.172**	**0.214**	**0.036**	**0.460**	**0.189**	**0.451**	**0.104**

Table 4: Multi-LLM framework with three models. We bold the best results for each combination of the experimental variables, and we underline the best results overall. For ease of comparison, we reproduce the best-performing 2-LLM results obtained in Table 2

		Input Tokens	Output Tokens	Average Tokens	Total Tokens
Decentralized	Multi-LLM 3 round max	383.73M	25.63M	14.62M	409.37M
	Multi-LLM 1 round max	129.36M	11.89M	11.77M	141.25M
Centralized	Multi-LLM 3 round max	216.65M	19.55M	14.76M	236.2M
	Multi-LLM 1 round max	77.69M	6.77M	10.56M	84.46M

Table 5: Cost Analysis of our Multi-LLM Decentralized and Centralized Summarization Methods. Note M= millions of tokens.

			ArXiv				GovReport			
			ROUGE-1 ↑	ROUGE-L ↑	BLEU-1 ↑	BLEU-4 ↑	ROUGE-1 ↑	ROUGE-L ↑	BLEU-1 ↑	BLEU-4 ↑
GPT-4o mini Evaluator	**Decentralized**	Multi-LLM 3 round max	0.317	0.160	0.206	0.026	**0.445**	**0.178**	**0.452**	**0.094**
		Multi-LLM 1 round max	**0.326**	**0.163**	**0.221**	**0.027**	0.438	0.175	0.446	0.089
	Centralized	Multi-LLM 3 round max	0.315	0.158	0.201	0.027	**0.441**	**0.176**	**0.447**	**0.092**
		Multi-LLM 1 round max	**0.330**	**0.165**	**0.222**	**0.028**	0.439	0.175	0.446	0.090
GPT-3.5 Evaluator	**Decentralized**	Multi-LLM 3 round max	0.313	0.163	0.200	0.029	0.447	0.180	0.458	0.098
		Multi-LLM 1 round max	**0.339**	**0.180**	**0.224**	**0.043**	**0.468**	**0.190**	**0.477**	**0.112**
	Centralized	Multi-LLM 3 round max	0.329	0.168	0.217	0.031	0.468	0.189	0.470	0.109
		Multi-LLM 1 round max	**0.333**	**0.173**	**0.219**	**0.036**	**0.479**	**0.197**	**0.485**	**0.121**
GPT-4o Evaluator	**Decentralized**	Multi-LLM 3 round max	**0.326**	**0.166**	**0.214**	0.030	0.446	0.179	0.456	0.098
		Multi-LLM 1 round max	0.325	0.165	0.211	0.030	**0.456**	**0.183**	**0.461**	**0.100**
	Centralized	Multi-LLM 3 round max	0.318	0.162	0.206	0.027	0.449	0.181	0.452	0.096
		Multi-LLM 1 round max	**0.327**	**0.167**	**0.215**	**0.031**	**0.461**	**0.186**	**0.467**	**0.105**

Table 6: Results for different evaluating and tie-breaking models for Multi-LLM approaches. The choice of the tie-breaker models is the same as the choice of evaluator model. We bold the best results for each combination of the experimental variables, and we underline the best results overall. For ease of comparison, we reproduce the best-performing 2-LLM results obtained in Table 2

styles, which may negatively affect evaluation metrics like ROUGE that rely on lexical overlap.

6.4 Cost Analysis

Table 5 presents the cost analysis for both decentralized and centralized methods based on the results in Table 2. The input and output token counts for evaluation for the decentralized method are twice those for the centralized method, which reflect the number of LLMs in the setup.

On a more theoretical note, let I denote the number of input tokens in the original text and $O_{\max}$ represent an upper bound on summary length. In each conversation round i (up to $t_{\max}$ rounds), we prompt k LLMs with I input tokens (we ignore instruction texts here since they are often short and constant in length). Each LLM then produces up to $O_{\max}$ output tokens. We consider input and output tokens separately since they often incur different costs.

For the generation phase in the centralized method, the input token cost per round is $\mathcal{O}(k \cdot I)$, and the output token cost is $\mathcal{O}(k \cdot O_{\max})$. For evaluation, the central LLM processes k candidate summaries, i.e. the **input** token cost is $\mathcal{O}(k \cdot O_{\max})$. The output cost for evaluation is $\mathcal{O}(1)$: since we instruct the central LLM to output only an anonymous identifier for the chosen summary, we reduce output token length in evaluation, thereby reducing the chance of hallucination and enabling more straightforward cost accounting.

Over $t_{\max}$ rounds, the total input token usage is in the order of $\mathcal{O}(t_{\max} \cdot k \cdot (I + O_{\max}))$. Output tokens is $\mathcal{O}(t_{\max} \cdot k \cdot O_{\max})$. Although this complexity may appear large, $t_{\max}$ is typically small (e.g., 2 or 3), and $O_{\max}$ is usually constrained (e.g., a brief 160-word summary).

For the decentralized approach, the worst-case generation token cost from generation remains the same as the centralized method. However, evaluation cost scales to $\mathcal{O}(t_{\max} \cdot k^2 \cdot O_{\max})$ because all k models now receive $\mathcal{O}(k \cdot O_{\max})$ input tokens. This results in the new total input token usage in the order of $\mathcal{O}(t_{\max} \cdot k \cdot (I + k \cdot O_{\max}))$, or, $\mathcal{O}(t_{\max} \cdot k \cdot I + k^2 \cdot O_{\max})$. Since $k^2 \cdot O_{\max}$ may dominate for large k, this term can become the bottleneck. However, in practical scenarios, k (the number of LLMs) is often small (e.g., 2–5), making the decentralized evaluation overhead manageable. The output token for generation remains the same, but for evaluation, it scales to $\mathcal{O}(t_{\max} \cdot k \cdot 1)$, since now all k models are evaluating. The total output token cost, therefore, is still dominated by the generation phase, giving $\mathcal{O}(t_{\max} \cdot k \cdot O_{\max})$. In other words, for both centralized and decentralized methods, output tokens scale in the same way. This highlights the advantage of only outputting identifiers in the evaluation phase as API costs per token are often higher for output than for input.

7 Conclusion

We presented a multi-LLM framework for text summarization, and proposed two strategies, decentralized and centralized summarization. We demonstrated that the proposed multi-LLM summarization techniques lead to better generated summaries. Our results indicate that multi-LLM approaches are useful for improving text summarization. Although the scope of this study is limited to few and mostly proprietary models due to resource constraints, we hope future works expand upon the diversity of models for more robust results.

References

Griffin Adams, Alexander Fabbri, Faisal Ladhak, Eric Lehman, and Noémie Elhadad. 2023. From sparse to dense: Gpt-4 summarization with chain of density prompting.

Joshua Ainslie, Santiago Ontanon, Chris Alberti, Vaclav Cvicek, Zachary Fisher, Philip Pham, Anirudh Ravula, Sumit Sanghai, Qifan Wang, and Li Yang. 2020. Etc: Encoding long and structured inputs in transformers.

Lochan Basyal and Mihir Sanghvi. 2023. Text summarization using large language models: A comparative study of mpt-7b-instruct, falcon-7b-instruct, and openai chat-gpt models.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer.

Yapei Chang, Kyle Lo, Tanya Goyal, and Mohit Iyyer. 2024. Booookscore: A systematic exploration of book-length summarization in the era of llms.

Justin Chih-Yao Chen, Swarnadeep Saha, and Mohit Bansal. 2024. Reconcile: Round-table conference improves reasoning via consensus among diverse llms.

Naman Chhibbar and Jugal Kalita. 2024. Automatic summarization of long documents. In *Proceedings of the 21st International Conference on Natural Language Processing (ICON)*, pages 607–615, AU-KBC Research Centre, Chennai, India. NLP Association of India (NLPAI).

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. 2022. Palm: Scaling language modeling with pathways.

Janara Christensen, Stephen Soderland, Gagan Bansal, and Mausam. 2014. Hierarchical summarization: Scaling up multi-document summarization. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 902–912. Association for Computational Linguistics.

Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. 2018. A discourse-aware attention model for abstractive summarization of long documents.

John M. Conroy and Hoa Trang Dang. 2008. Mind the gap: Dangers of divorcing evaluations of summary content from linguistic quality. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 145–152, Manchester, UK. Coling 2008 Organizing Committee.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-xl: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988, Florence, Italy. Association for Computational Linguistics.

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

Alexios Gidiotis and Grigorios Tsoumakas. 2020. A divide-and-conquer approach to the summarization of long documents. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:3029–3040.

Yihong Gong and Xin Liu. 2001. Generic text summarization using relevance measure and latent semantic analysis. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 19–25. ACM.

Tanya Goyal, Junyi Jessy Li, and Greg Durrett. 2023. News summarization and evaluation in the era of gpt-3.

Mandy Guo, Joshua Ainslie, David Uthus, Santiago Ontanon, Jianmo Ni, Yun-Hsuan Sung, and Yinfei Yang. 2021. Longt5: Efficient text-to-text transformer for long sequences.

Karl Moritz Hermann, Tomáš Kočiský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend.

Luyang Huang, Shuyang Cao, Nikolaus Parulian, Heng Ji, and Lu Wang. 2021. Efficient attentions for long document summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1419–1436, Online. Association for Computational Linguistics.

Emma Järvinen. 2024. Long-input summarization using large language models.

Gunjan Keswani, Wani Bisen, Hirkani Padwad, Yash Wankhedkar, Sudhanshu Pandey, and Ayushi Soni. 2024. Abstractive long text summarization using large language models. *International Journal of Intelligent Systems and Applications in Engineering*, 12(12s):160–168.

Anastassia Kornilova and Vladimir Eidelman. 2019. BillSum: A corpus for automatic summarization of US legislation. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 48–56, Hong Kong, China. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Irene Li, Aosong Feng, Dragomir Radev, and Rex Ying. 2023. Hipool: Modeling long documents using graph neural networks. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 161–171, Toronto, Canada. Association for Computational Linguistics.

Yunxuan Li, Yibing Du, Jiageng Zhang, Le Hou, Peter Grabowski, Yeqing Li, and Eugene Ie. 2024. Improving multi-agent debate with sparse communication topology.

Tian Liang, Zhiwei He, Wenxiang Jiao, Xing Wang, Rui Wang, Yujiu Yang, Zhaopeng Tu, and Shuming Shi. 2024. Encouraging divergent thinking in large language models through multi-agent debate.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2023. Lost in the middle: How language models use long contexts.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3730–3740.

Yixin Liu, Pengfei Liu, Dragomir Radev, and Graham Neubig. 2022. Brio: Bringing order to abstractive summarization.

S. Mallick, A. Ghosh, et al. 2019. A survey on extractive text summarization. *Journal of Artificial Intelligence Research*, 65:123–143.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into texts. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 404–411.

Ramesh Nallapati, Bowen Zhou, Cicero Nogueira dos santos, Caglar Gulcehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization.

Bo Pang, Erik Nijkamp, Wojciech Kryściński, Silvio Savarese, Yingbo Zhou, and Caiming Xiong. 2022. Long document summarization with top-down and bottom-up inference.

Dongqi Pu, Yifan Wang, and Vera Demberg. 2023a. Incorporating distributions of discourse structure for long document abstractive summarization.

Xiao Pu, Mingqi Gao, and Xiaojun Wan. 2023b. Summarization is (almost) dead.

Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 379–389.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H. Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Fevry, Jason Alan Fries, Ryan Teehan, Tali Bers, Stella Biderman, Leo Gao, Thomas Wolf, and Alexander M. Rush. 2022. Multitask prompted training enables zero-shot task generalization.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083.

Sam Shleifer. 2020. Distilbart-cnn-12-6. `https://huggingface.co/sshleifer/distilbart-cnn-12-6`. Accessed: 2024-05-29.

Sainbayar Sukhbaatar, Edouard Grave, Piotr Bojanowski, and Armand Joulin. 2019. Adaptive attention span in transformers. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 331–335, Florence, Italy. Association for Computational Linguistics.

Wen Xiao and Giuseppe Carenini. 2019. Extractive summarization of long documents by combining global and local context. In *Proceedings of the*

2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 3011–3021, Hong Kong, China. Association for Computational Linguistics.

Manzil Zaheer, Guru Guruganesh, Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. 2021. Big bird: Transformers for longer sequences.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J. Liu. 2020. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization.

Tianyi Zhang, Faisal Ladhak, Esin Durmus, Percy Liang, Kathleen McKeown, and Tatsunori B. Hashimoto. 2023. Benchmarking large language models for news summarization.