Hybrid-RACA: Hybrid Retrieval-Augmented Composition Assistance for Real-time Text Prediction

Menglin Xia* **Xuchao Zhang*** **Camille Couturier**
Guoqing Zheng **Saravan Rajmohan** **Victor Rühle**
Microsoft
{mollyxia, xuchaozhang, cacoutur, zheng, saravar, viruh}@microsoft.com

Abstract

Large language models (LLMs) enhanced with retrieval augmentation has shown great performance in many applications. However, the computational demands for these models pose a challenge when applying them to real-time tasks, such as composition assistance. To address this, we propose Hybrid Retrieval-Augmented Composition Assistance (Hybrid-RACA), a novel system for real-time text prediction that efficiently combines a cloud-based LLM with a smaller client-side model through retrieval augmented memory. This integration enables the client model to generate better responses, benefiting from the LLM's capabilities and cloud-based data. Meanwhile, via a novel *asynchronous* memory update mechanism, the client model can deliver real-time completions to user inputs without the need to wait for responses from the cloud. Our experiments on five datasets demonstrate that Hybrid-RACA offers strong performance while maintaining low latency.

1 Introduction

Large language models have become powerful tools in language processing and they are widely adopted across applications. When augmented with retrieved documents (Lewis et al., 2020; Liu et al., 2022), these models can generate more relevant and useful responses. However, the large size of these models and the additional retrieval step introduce significant computational overhead. This leads to increased latency and higher operational costs, limiting their effectiveness in real-time applications, such as composition assistance.

Real-time *composition assistance* tools are designed to swiftly suggest next words or sentences to help users write faster. These systems must operate within tight latency budgets, and they are frequently triggered as the user types. To minimize latency (including model inference latency and communication to the cloud) and to reduce costs, these models are usually deployed on users' edge devices. This imposes strict constraints on the model's size and capabilities, limiting the effectiveness of composition assistance. While recent advancements have enabled models such as Llama (Touvron et al., 2023) to run on smaller devices[1], they still fall short in terms of achieving real-time responses.

For real-time tasks, we encounter a dilemma: LLMs offer superior performance but they are slow and expensive to run, whereas client models are agile and efficient but limited in performance. Hybrid computing between client and cloud models is a promising approach to bridge the gap between the challenges of latency and model performance. However, in existing hybrid computing patterns, such as model routing and split computing (Kudugunta et al., 2021; Matsubara et al., 2022), client and cloud models usually function with synchronized communication. This means that whenever the cloud model is utilized, the system must wait for the cloud model to complete its processing before producing the output. Therefore, simply applying existing hybrid patterns to cloud-based LLMs will not resolve the issue of latency and cost. Besides, existing hybrid patterns usually overlook cloud-based data, which could be essential for effective composition assistance, such as accessing relevant documents in companies' cloud storage.

To address these challenges, we propose a novel Hybrid Retrieval-Augmented Composition Assistance (Hybrid-RACA) system (see Figure 1). This system leverages a cloud LLM and cloud data to boost the performance of small language models on client devices through retrieval augmentation, while operating *in an asynchronous manner*. Hybrid-RACA consists of an augmentation coordinator and a small model for text prediction de-

*These authors contributed equally to this work.

[1] https://github.com/ggerganov/llama.cpp

Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track, pages 120–131
November 12-16, 2024 ©2024 Association for Computational Linguistics

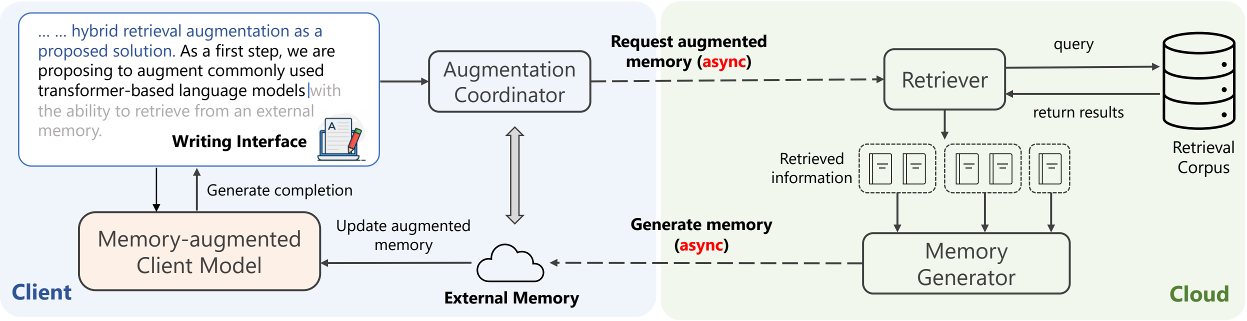

... ... hybrid retrieval augmentation as a proposed solution. As a first step, we are proposing to augment commonly used transformer-based language models|with the ability to retrieve from an external memory.
Writing Interface
Augmentation Coordinator
Request augmented memory (async)
Retriever
query
return results
Retrieval Corpus
Retrieved information
Generate completion
Memory-augmented Client Model
Update augmented memory
Generate memory (async)
Memory Generator
External Memory
Client
Cloud

Figure 1: Overview of the Hybrid-RACA system, which is a hybrid system for composition assistance. The top left box represents the writing interface. The framework has four main components: augmentation coordinator and client model on the client side (left), and retriever and LLM-based memory generator on the cloud (right).

ployed on client devices, as well as a retriever and an LLM located on the cloud server. The client augmentation coordinator sends asynchronous request to the cloud. The cloud retrieves relevant documents and employs an LLM to compress the retrieved documents into shorter snippets of information, which we refer to as *memory*, and sends it asynchronously to the client. On the client side, an instruction-tuned client model leverages available memory to suggest the next words.

The Hybrid-RACA system offers several benefits. (1) *Enhanced utility*: Hybrid retrieval augmentation enables the client model to make better suggestions by leveraging cloud-based resources. (2) *Low latency*: Asynchronous augmentation allows the client to make predictions without waiting for the cloud. This mitigates the effects of network latency and avoids slow inference inherent to cloud-based retrieval-augmented LLMs. (3) *Reduced client-to-cloud communication*: the augmentation coordinator minimizes the client-to-cloud communication by requesting augmented memory only when existing memory becomes stale, reducing the frequency of calling the cloud models and thus saving cost. Furthermore, using LLM-compressed memory further reduces data transfer volume.

To evaluate our system, we conduct experiments on the text prediction task on five datasets from diverse domains. We compare our model to several baselines and show that our model exhibits substantial utility improvement in text prediction while maintaining low latency. The code for our system will be made available at: `https://github.com/microsoft/hybrid-raca`.

2 Related Work

Hybrid Computing Hybrid computing divides processing tasks between the edge and the cloud, effectively addressing the limited computation capabilities of edge devices and enabling real-time responses of critical services (Loghin et al., 2019; Wang et al., 2020). For example, split computing partitions machine learning modules between edge and cloud devices to balance overall computation cost and efficiency (Matsubara et al., 2022; Osia et al., 2020). Communication between edge and cloud in split computing is inherently synchronized, as both devices contribute to completing one inference run. More recently, task-specific model routing (Kudugunta et al., 2021) has also emerged as a promising approach for hybrid computing via routing between client and cloud models. Nonetheless, the overall system still needs to wait for the cloud model whenever it is used, thus limiting the overall latency. Another notable paradigm for hybrid computing in machine learning is federated learning, which leverages multiple computing devices for training machine learning models for safety or efficiency purposes (Bonawitz et al., 2019). However, this technique is less commonly used for inference. In addition to hybrid computing, there is also literature on improving efficiency of models deployed on edge devices (Tambe et al., 2021) as well as methods on reducing the size of large models for deployment on smaller devices (Hoefler et al., 2021). These methods are orthogonal to our work.

Retrieval Augmented Models Retrieval augmentation enhances a language model with retrieved information from external databases. Various methods have been proposed to integrate the retrieved data into the language model, including the use of prompts (Lewis et al., 2020; Guu et al., 2020; Shi et al., 2023), cross-attention modules (Borgeaud et al., 2021), vector concatenation (Izacard and Grave, 2021; Fan et al., 2021), and output distribution adjustment at decoding (Khandelwal et al., 2020; Liu et al., 2022). In this work, we adopt the prompting method, which incorporates retrieved data into the input. However, the Hybrid-RACA system can be extended to other retrieval augmentation approaches.

3 Hybrid-RACA

We present our Hybrid-RACA system that leverages cloud-generated memory to enhance the utility of client-based language model while maintaining low latency for composition assistance.

In Hybrid-RACA, the augmentation coordinator (client) monitors the writing context and sends an asynchronous request for an augmented memory from the cloud. The retriever on the cloud searches for relevant data upon request. Subsequently, The memory generator (cloud) leverages an LLM to construct a memory that includes all essential information from the retrieved data, optimizing its usefulness. Finally, the memory is transmitted to the client and seamlessly integrated into the client model for offering real-time suggestions. Algorithm 1 describes the inference workflow of Hybrid-RACA.

In the following subsections, we discuss the details of the four main components.

3.1 Augmentation Coordinator

The augmentation coordinator manages the augmented memory $\mathcal{M}$ by monitoring changes to the *writing context*, which we define as the text the user has already typed (see Fig.2). To determine whether a memory update is necessary, the coordinator takes into account the current context $\boldsymbol{x}_t$ and the context $\boldsymbol{x}_{t-1}$ from the previous step and calculates the edit distance $\mathrm{ED}(\boldsymbol{x}_t, \boldsymbol{x}_{t-1})$. Once the distance exceeds a pre-determined threshold τ, the coordinator initiates a request to the cloud server asking for a new memory. We employ the Levenshtein distance (Yujian and Bo, 2007) to measure token-level difference. To avoid redundant memory requests, we adopt an incremental memory update approach, where only the newly updated context is used as the query input to generate the new memory m_t. When the augmented memory reaches its maximum capacity of $\mathcal{M}$, the oldest memory m_0 is deprecated and replaced by the new memory m_t.

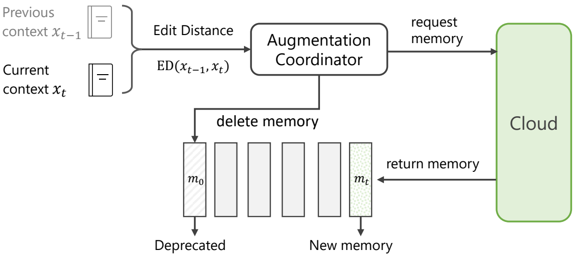


Figure 2: Process of the augmentation coordinator

3.2 Retrieval-Augmented Memory Generator

Upon receiving a request from the augmentation coordinator, the memory generator on the cloud initiates the preparation of the augmented memory, which will be returned to the client. The memory preparation process consists of two steps: document retrieval and memory generation.

Document Retrieval Given an input query $\boldsymbol{x}$, the goal of the retriever is to retrieve the most relevant documents $\mathcal{D}_r = \{d_1, \ldots, d_k\}$ from a large corpus $\mathcal{D}$, where $\mathcal{D}_r \subseteq \mathcal{D}$. We use the Dense Passage Retrieval (DPR) (Karpukhin et al., 2020) model in our implementation.

Memory Generation After retrieving the relevant documents $\mathcal{D}_r$, we employ a LLM to generate concise key takeaways that capture essential information from the documents. We use the key takeaways instead of the original documents because the client model is a small language model that usually struggles with processing long context and has a strict limit on input context length. Additionally, extracting key takeaways significantly reduces the memory size, resulting in lower communication and inference cost for the client.

To generate key takeaways from retrieved documents $\mathcal{D}_r$, we split the documents into chunks and utilize an LLM to extract key takeaways from each chunk. To minimize the frequency of LLM calls, we consolidate multiple chunks within one document. Subsequently, all generated key takeaways from retrieval documents are merged to form the memory m_t for the current t-th memory request.

3.3 Memory-Augmented Client Model

The goal of the client model is to generate useful completions to the user input. Enhanced by cloud-generated memory, our client model learns to make more relevant predictions. We further adopt instruction-tuning to bolster the client model's ability to effectively leverage cloud-generated memory.

Algorithm 1: Inference workflow of Hybrid-RACA

```
Data: current user input $x_t$, input history $x_{t-1}$, retrieval corpus $\mathcal{D}$, retrieval model $\mathcal{M}_{\text{retrieval}}$, cloud-based LLM
      $\mathcal{M}_{\text{cloud}}$, client model $\mathcal{M}_{\text{client}}$, memory $\mathcal{M}$
while $x_t$ do
    $\text{ED}_\text{t} = \text{EditDistance}(x_t, x_{t-1})$ ;                  ▷ Compute changes in context
    if $\text{ED}_\text{t} > \tau$ ;                                         ▷ Send async request to the cloud
    then
        async $\mathcal{D}_r = \{d_1, ...d_k\}$: $\mathcal{D}_r \sim \mathcal{M}_{\text{retrieval}}(x_t, \mathcal{D})$ ;   ▷ Retrieve relevant documents
        async $m_t \sim \mathcal{M}_{\text{cloud}}(\mathcal{D}_r)$ ;                           ▷ Generate memory
        $\mathcal{M} = Update(\mathcal{M}, m_t)$ ;                                   ▷ Update $\mathcal{M}$ with $m_t$
        Sample $y_t \sim \mathcal{M}_{\text{client}}(x_t, \mathcal{M})$ ;            ▷ Text prediction with the client model
        if $Accept(y_t)$ then
            $x_{t-1} \leftarrow \{x_{t-1}, x_t\}, x_t \leftarrow \{x_t, y_t\}$ ;     ▷ User accepts suggestion
        else
            $x_t \leftarrow \{x_t, Input()\}$ ;                   ▷ User rejects suggestion and enters new input
        end
    end
end
```

Instruction enhanced prompt

Reference: In 2020, Generative Pre-trained Transformer 3 (GPT-3) was unveiled, a deep learning-based autoregressive language model that can produce human-like text ... ... This process has eliminated the need for laborious manual labeling and human supervision.
Complete the following text based on the reference:
Generative Pre-trained Transformer 3 (GPT-3) is an autoregressive language model released in 2020 that

Output

is capable of producing human-like text when prompted with an initial text.

Figure 3: Example of constructing instruction-tuning data

To instruction-tune the client model, we leverage an LLM to generate the instruction tuning data. Given a document d, we use the beginning part of the document as the input prompt $\boldsymbol{x} = \mathcal{I}(d)$ and use $\boldsymbol{x}$ to generate the augmented memory $\mathcal{M}$. We formulate an instruction-enhanced prompt to instruct the model to make predictions based on the memory (see Fig.3). As for the ground truth labels $\hat{\boldsymbol{y}}$, a straightforward approach is to directly use the remaining part of the document d. However, this is not ideal as there is usually a discrepancy between the original text and the memory, which can negatively impact the performance of the client model. To address this, we employ an LLM to generate the labels $\hat{\boldsymbol{y}} = \mathcal{M}_{\text{cloud}}(\mathcal{I}(d), \mathcal{M})$.

Then we finetune the client model on the instruction-enhanced prompt and the LLM-generated labels. The model is finetuned on the task to predict $\hat{\boldsymbol{y}}$ given $\boldsymbol{x}$ and $\mathcal{M}$. To minimize the discrepancy between our model's predictions $\boldsymbol{y}$ and the LLM-generated labels $\hat{\boldsymbol{y}}$, we employ the cross-entropy loss on the generated tokens in finetuning:

$$\mathcal{L}_d = -\sum_{i=1}^{l} \hat{y}_i \log\left(p_\theta(y_i|\boldsymbol{x}, \mathcal{M}, \hat{y}_{<i})\right) \qquad (1)$$

where l is the length of the label and $p_\theta(\cdot)$ refers to the probability of tokens generated by the client model.

4 Experiments

In this section, we introduce the experimental setup (Section 4.1) and present the evaluation results of Hybrid-RACA system on utility (Section 4.2.1), inference latency (Section 4.2.2) and effects of asynchronous memory update (Section 4.2.3).

4.1 Experimental Setup

Datasets and Labels We train our models on WikiText-103 (Merity et al., 2016) and evaluate them on the text prediction task on five datasets, including in-domain evaluation on WikiText-103, and out-of-domain evaluation on Enron Emails (Klimt and Yang, 2004), HackerNews[2], NIH Ex-

[2] https://news.ycombinator.com/

		PPL	GLEU	BLEU4	ROUGE1	ROUGEL	METEOR	BERTScore
OPT-125M	Vanilla OPT	9.3	11.4	6.9	27.5	22.1	20.2	84.0
	HybridRAG	4.3	12.8	9.6	28.4	23.4	22.4	84.5
	Hybrid-RACA w/o FT	3.8	14.7	12.2	29.9	25.1	24.3	84.8
	Hybrid-RACA FT	3.4	23.0	21.4	39.6	32.8	34.4	87.0
	Hybrid-RACA IT	**2.6**	**30.2**	**28.8**	**48.3**	**40.2**	**44.1**	**89.0**
OPT-350M	Vanilla OPT	7.4	13.2	8.8	30.1	24.3	22.8	84.8
	HybridRAG	3.6	15.4	12.5	31.6	26.0	25.6	85.4
	Hybrid-RACA w/o FT	3.3	17.6	15.4	33.5	27.9	28.0	85.7
	Hybrid-RACA FT	3.2	23.9	22.3	40.7	33.8	35.5	87.4
	Hybrid-RACA IT	**2.4**	**32.6**	**31.4**	**50.8**	**42.9**	**46.6**	**89.5**

Table 1: In-domain evaluation of Hybrid-RACA performance

		Enron Emails		NIH ExPorter		Hacker News		Youtube Subtitles	
		PPL	GLEU	PPL	GLEU	PPL	GLEU	PPL	GLEU
OPT-125M	Vanilla OPT	8.5	5.8	7.4	9.3	7.5	8.0	9.2	5.7
	HybridRAG	6.3	8.0	4.4	10.7	7.2	7.5	7.0	7.2
	Hybrid-RACA w/o FT	4.6	9.0	4.1	10.9	5.6	8.9	5.9	7.1
	Hybrid-RACA FT	4.4	13.8	3.7	16.8	5.3	14.8	5.5	12.5
	Hybrid-RACA IT	**3.3**	**22.9**	**2.9**	**24.2**	**3.8**	**20.2**	**4.4**	**20.4**
OPT-350M	Vanilla OPT	7.4	5.9	6.2	10.3	6.4	8.5	7.7	6.3
	HybridRAG	5.5	9.1	3.7	12.4	6.1	8.4	5.8	8.5
	Hybrid-RACA w/o FT	4.1	12.5	3.5	12.6	4.8	11.6	5.0	9.9
	Hybrid-RACA FT	4.2	13.3	3.5	17.9	5.1	13.3	5.2	13.4
	Hybrid-RACA IT	**3.1**	**24.7**	**2.7**	**25.5**	**3.7**	**20.7**	**4.2**	**20.8**

Table 2: Out-of-domain evaluation of Hybrid-RACA performance

Model	GPT Score
GPT3.5	**7.73**
Vanilla OPT-125M	2.20
Vanilla OPT-350M	2.60
Hybrid-RACA IT OPT-125M	5.27
Hybrid-RACA IT OPT-350M	**5.49**

Table 3: LLM evaluation of text completion quality

Porter[3], and Youtube Subtitles (Gao et al., 2020), covering a diverse range of domains. We use an LLM to generate ground truth labels for evaluation.

Evaluation Metrics To evaluate utility, we use standard metrics including perplexity (PPL) (Jelinek et al., 1977), GLEU (Wu et al., 2016), BLEU (Papineni et al., 2002), ROUGE (Lin, 2004), METEOR (Banerjee and Lavie, 2005), and BERTScore (Zhang et al., 2020). We calculate perplexity by measuring how well the model predicts the labels given the prompts. We use other metrics to measure the degree of similarity between model predictions and the labels. In addition, we evaluated the completion quality of 100 sampled data points using GPT-4-turbo, rating on a scale of 1-10. To evaluate the inference latency of our system, we measure the average running time.

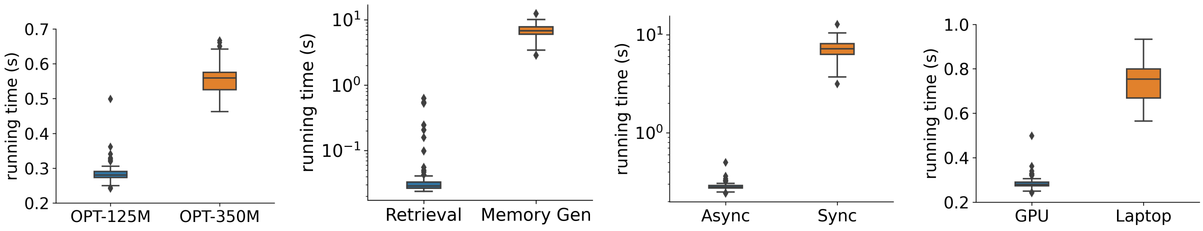
Implementation Details For the client model, we compare two small OPT models (Zhang et al., 2022): OPT-125M and OPT-350M. Both models are decoder-only transformers that are small enough to run with limited latency budget. We employ greedy search for decoding. For the LLM, we use the GPT-3.5 text-davinci-003 model[4]. We set max new tokens to 44 for both label generation and text prediction. For document retrieval, we use the Faiss library (Johnson et al., 2019) and set $k = 3$ after a hyperparameter search on WikiText data.

For latency evaluation, we deploy the client models on two different machines: a GPU machine with an 11GB Nvidia Tesla K80 GPU, and a laptop without a GPU. We set max new tokens to 15 for latency evaluation.

Baseline Methods We compare our approach against four baselines. We ensure a fair comparison by regenerating labels for each baseline, based on the memory used by that baseline.

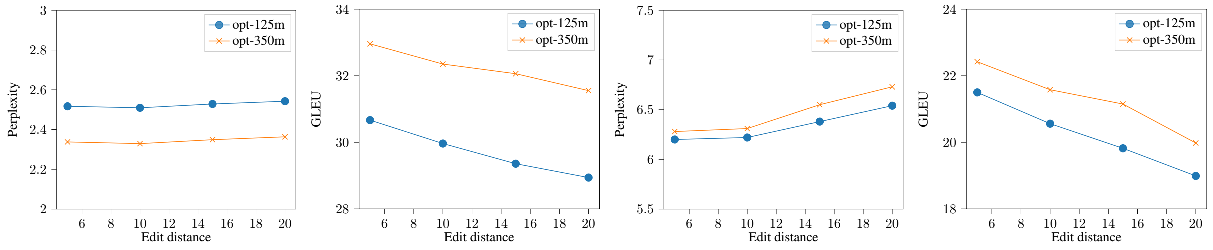
[3] `https://exporter.nih.gov/`

[4] With OpenAI API `https://platform.openai.com/docs/models/gpt-3.5`, temperature $= 0$, top_$p = 1$



(a) 125M vs 350M (b) retrieval vs memory (c) async vs sync (d) GPU vs laptop

Figure 4: Inference latency for client inference, retrieval and memory generation on multiple devices



(a) PPL (async labels) (b) GLEU (async labels) (c) PPL (sync labels) (d) GLEU (sync labels)

Figure 5: Hybrid-RACA performance with asynchronous memory update.

Vanilla OPT - A vanilla client model for text prediction without additional memory from the cloud.

Hybrid-RAG - The RAG approach (Lewis et al., 2020) can be turned into a hybrid setup with our system. In this setting, we retrieve and feed the full retrieved text to the client model.[5]

Hybrid-RACA w/o FT and *Hybrid-RACA FT* - To assess the efficacy of our instruction-tuned client model, we examine two variants of the client model, one without finetuning (Hybrid-RACA w/o FT) and one finetuned to use the memory to predict the original remaining text (Hybrid-RACA FT).

4.2 Experimental Results

4.2.1 Utility

Table 1 presents the performance of the models on WikiText-103. Table 2 presents the perplexity and GLEU scores on the other four datasets. The results show that our approach outperforms all baselines and generalizes well to out-of-domain data. The HybridRAG approach outperforms a vanilla OPT baseline with retrieval augmentation, and the Hybrid-RACA w/o FT model improves upon it by using the LLM to extract key takeaways from retrieved data. This indicates that the representation of the context is vital to client model performance. Furthermore, our final model, *Hybrid-RACA IT* (Instruction-tuned Hybrid-RACA), shows the best performance, suggesting that instruction-tuning helps the model better leverage context. Further, OPT-350M based models consistently outperform OPT-125M ones, showing that model size is critical to its overall performance. Table 3 shows the evaluation results from GPT-4-turbo, demonstrating that Hybrid-RACA significantly enhances text completion quality.

4.2.2 Inference Latency

We performed a latency evaluation for Hybrid-RACA. Fig.4a shows the run times for the client models on a GPU machine. Unsurprisingly, OPT-125M is 49.3% faster compared to OPT-350M. Fig.4b presents the run times for retrieval and memory generation steps, showing that memory generation with LLM consumes the majority time for memory preparation. Fig.4c compares our asynchronous Hybrid-RACA (OPT-125M) to a synchronous approach by directly calling GPT-3.5 and a retriever for composition assistance. Notably, our approach showcases an impressive speed improvement, achieving a remarkable 138x faster performance compared to the synchronous approach. Fig.4d compares the run times of Hybrid-RACA OPT-125M on a GPU machine and a laptop without GPU. It shows that our approach can be deployed on edge devices without GPUs, although slower.

Notably, we didn't use caching or quantization. These methods are orthogonal to our work and can

[5] This only works if the documents are sufficiently short to fit in the limited input context of the client model.

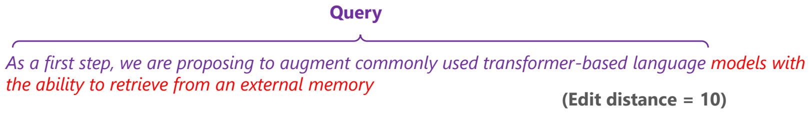

Query
As a first step, we are proposing to augment commonly used transformer-based language models with the ability to retrieve from an external memory
(Edit distance = 10)

Figure 6: An example of setting edit distance threshold = 10 in asynchronous memory update. In this setting, text prediction is generated from the entire prompt, but only the beginning part is used for memory generation.

be used in conjunction to further improve the speed.

4.2.3 Asynchronous Memory Update

Fig.5 illustrates the impact of asynchronous memory update on model utility. To measure this effect, we conducted an experiment in which we gradually increased the edit distance threshold that determines how often the client model requests for memory updates. For each prompt, we use the beginning part of the prompt as the query for memory generation and the entire prompt for text prediction, mimicking the case where the memory lags behind the current input context due to asynchronous communication between client and cloud. Figure 6 demonstrates how we set the edit distance threshold in async memory update.

Fig.5a and Fig.5b show the trend in perplexity and GLEU scores with increased edit distance threshold, evaluated against GPT3.5 generated labels with the same asynchronous setup. Model utility remains relatively stable in perplexity with a deceasing trend in GLEU compared to LLMs. Fig.5c and Fig.5d show the scores of the client model under the asynchronous setup, evaluated against labels generated in an ideal synchronous memory update setup, where the memory is created using the entire prompt without lag. Due to the difference in the freshness of the memory, there is a larger gap between the asynchronous predictions and the synchronous labels. As the edit distance threshold increases, the memory becomes less up-to-date, resulting in a decline in model utility. Nevertheless, it still significantly outperformed the baselines.

5 Conclusion

In this paper, we propose Hybrid-RACA, a novel hybrid retrieval-augmented generation system for real-time composition assistance. By integrating LLM-enhanced memory into our instruction-tuned client model with asynchronous update, we show with experiment results on multiple datasets that our hybrid approach enables substantial utility improvements over smaller language models while maintaining inference efficiency, making it a valuable solution for real-time tasks.

Broader Impact

In our research, we present a pioneering approach to the future landscape of AI applications, envisioning a hybrid system that brings the best of client and cloud worlds. Our unique design allows client and cloud models to function seamlessly in a composition assistance scenario, achieving better performance by levering cloud models and data, and ensuring low-latency and cost-effectiveness by utilizing on-device client models. We believe that our hybrid solution with asynchronous communication is also a valuable solution to make advanced AI more accessible to a wider range of users, including those in resource-constrained environments or with limited access to high-speed internet connections. We believe that our vision can be extended to more applications not limited to composition assistance. Furthermore, our efficient solution, which combines edge and cloud computing, offers great potential to energy conservation. By minimizing the necessity to access resource-intensive large language models (LLMs), notorious for their high energy consumption, our approach mitigates potential harm to the environment. This not only underscores our commitment to sustainability but also highlights the practical benefits of our technology in addressing energy challenges.

Ethical Considerations

Hybrid-RACA is a composition assistance tool that integrates client and cloud models and data. In our implementation, data is transmitted between the client and cloud as plain text. However, this transmission process poses potential privacy and confidentiality risks. To mitigate these risks, security measures such as cryptography and access controls can be implemented. When instruction-tuning the client model, we used LLMs to generate completions, which can be considered as a form of synthetic data generation. Like other work that leverages LLMs, this might raise privacy and copyright concerns. We are committed to follow the best practices currently available to minimize privacy and copyright risks by conducting experiments on public datasets and adopting security guardrails.

References

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.

Keith Bonawitz, Hubert Eichner, Wolfgang Grieskamp, Dzmitry Huba, Alex Ingerman, Vladimir Ivanov, Chloe Kiddon, Jakub Konečnỳ, Stefano Mazzocchi, Brendan McMahan, et al. 2019. Towards federated learning at scale: System design. *Proceedings of machine learning and systems*, 1:374–388.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego de Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones, Albin Cassirer, Andy Brock, Michela Paganini, Geoffrey Irving, Oriol Vinyals, Simon Osindero, Karen Simonyan, Jack W. Rae, Erich Elsen, and Laurent Sifre. 2021. Improving language models by retrieving from trillions of tokens. *CoRR*, abs/2112.04426.

Angela Fan, Claire Gardent, Chloé Braud, and Antoine Bordes. 2021. Augmenting transformers with KNN-based composite memory for dialog. *Transactions of the Association for Computational Linguistics*, 9:82–99.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. 2020. The Pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Realm: Retrieval-augmented language model pre-training. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org.

Torsten Hoefler, Dan Alistarh, Tal Ben-Nun, Nikoli Dryden, and Alexandra Peste. 2021. Sparsity in deep learning: Pruning and growth for efficient inference and training in neural networks. *J. Mach. Learn. Res.*, 22(1).

Gautier Izacard and Edouard Grave. 2021. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 874–880, Online. Association for Computational Linguistics.

Fred Jelinek, Robert L Mercer, Lalit R Bahl, and James K Baker. 1977. Perplexity—a measure of the difficulty of speech recognition tasks. *The Journal of the Acoustical Society of America*, 62(S1):S63–S63.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, 7(3):535–547.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906*.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2020. Generalization through memorization: Nearest neighbor language models. In *International Conference on Learning Representations*.

Bryan Klimt and Yiming Yang. 2004. The enron corpus: A new dataset for email classification research. In *European conference on machine learning*, pages 217–226. Springer.

Sneha Kudugunta, Yanping Huang, Ankur Bapna, Maxim Krikun, Dmitry Lepikhin, Minh-Thang Luong, and Orhan Firat. 2021. Beyond distillation: Task-level mixture-of-experts for efficient inference. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3577–3599, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Ruibo Liu, Guoqing Zheng, Shashank Gupta, Radhika Gaonkar, Chongyang Gao, Soroush Vosoughi, Milad Shokouhi, and Ahmed Hassan Awadallah. 2022. Knowledge infused decoding. *arXiv preprint arXiv:2204.03084*.

Dumitrel Loghin, Lavanya Ramapantulu, and Yong Meng Teo. 2019. Towards analyzing the performance of hybrid edge-cloud processing. In *2019 IEEE International Conference on Edge Computing (EDGE)*, pages 87–94.

Yoshitomo Matsubara, Marco Levorato, and Francesco Restuccia. 2022. Split computing and early exiting for deep learning applications: Survey and research challenges. *ACM Comput. Surv.*, 55(5).

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer sentinel mixture models.

Seyed Ali Osia, Ali Shahin Shamsabadi, Sina Sajadmanesh, Ali Taheri, Kleomenis Katevas, Hamid R. Rabiee, Nicholas D. Lane, and Hamed Haddadi. 2020. A hybrid deep learning architecture for privacy-preserving mobile analytics. *IEEE Internet of Things Journal*, 7(5):4505–4518.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. pages 311–318.

Weijia Shi, Sewon Min, Michihiro Yasunaga, Minjoon Seo, Rich James, Mike Lewis, Luke Zettlemoyer, and Wen tau Yih. 2023. Replug: Retrieval-augmented black-box language models.

Thierry Tambe, Coleman Hooper, Lillian Pentecost, Tianyu Jia, En-Yu Yang, Marco Donato, Victor Sanh, Paul Whatmough, Alexander M. Rush, David Brooks, and Gu-Yeon Wei. 2021. Edgebert: Sentence-level energy optimizations for latency-aware multi-task nlp inference. In *MICRO-54: 54th Annual IEEE/ACM International Symposium on Microarchitecture*, MICRO '21, page 830–844, New York, NY, USA. Association for Computing Machinery.

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*.

Bo Wang, Changhai Wang, Wanwei Huang, Ying Song, and Xiaoyun Qin. 2020. A survey and taxonomy on task offloading for edge-cloud computing. *IEEE Access*, 8:186080–186101.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation.

Li Yujian and Liu Bo. 2007. A normalized levenshtein distance metric. *IEEE transactions on pattern analysis and machine intelligence*, 29(6):1091–1095.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

A More results on utility evaluation

The results of the model utility on Enron Emails, NIH ExPorter, HackerNews, and YouTubeSubtitles datasets evaluated in all seven metrics are presented in Tables 4 and 5. We can observe that our model consistently outperforms all the other baselines.

B Template used to calculate GPT-score

We use the following template to instruct GPT-4-turbo to evaluate the performance of of the models:

```
    Please act as an impartial judge and evaluate the quality of the text completion provided by an AI assistant to the text prompt displayed below. For this evaluation, you should primarily consider the following criteria:
relevance: Is the completion relevant to the prompt? Is the completion a fluent continuation from the prompt?
correctness: Is the completion correct and factual?
fluency: Is the completion fluent, free of grammatical errors, and devoid of redundant repetitions? Please note that it is acceptable for the completion to stop abruptly before the end of a sentence.
Begin your evaluation by providing a short explanation. Be as objective as possible. After providing your explanation, you must rate the response on a scale of 1 to 10 by strictly following this format: "[[rating]]", for example: "Rating: [[5]]"

[Text Prompt]
{prompt}

[The Start of Assistant's Completion]
{completion}
[The End of Assistant's Completion]
```

		PPL	GLEU	BLEU-4	ROUGE-1	ROUGE-L	METEOR	BERTScore
Enron Emails	Vanilla OPT	7.4	5.9	2.7	17.3	14.3	13.2	80.2
	HybridRAG	5.5	9.1	6.6	21.7	18.1	17.0	80.6
	Hybrid-RACA w/o FT	4.1	12.5	10.8	25.3	21.6	21.1	81.8
	Hybrid-RACA FT	4.2	13.3	11.6	26.5	22.1	22.8	83.1
	Hybrid-RACA IT	**3.1**	**24.7**	**22.7**	**43.9**	**35.4**	**39.6**	**87.9**
NIH ExPorter	Vanilla OPT	6.2	10.3	5.4	27.7	22.3	19.6	85.3
	HybridRAG	3.7	12.4	8.9	30.2	24.5	23.3	85.8
	Hybrid-RACA w/o FT	3.5	12.6	9.3	30.0	24.6	23.7	85.7
	Hybrid-RACA FT	3.5	17.9	15.4	36.5	29.4	30.6	87.2
	Hybrid-RACA IT	**2.7**	**25.5**	**23.2**	**45.9**	**37.2**	**41.2**	**89.2**
Hacker News	Vanilla OPT	6.4	8.5	5.0	24.7	20.5	16.3	84.9
	HybridRAG	6.1	8.4	5.6	22.4	18.9	14.7	83.9
	Hybrid-RACA w/o FT	4.8	11.6	9.2	27.0	22.6	19.4	84.9
	Hybrid-RACA FT	5.1	13.3	11.4	28.2	23.0	21.6	84.8
	Hybrid-RACA IT	**3.7**	**20.7**	**18.2**	**40.3**	**31.6**	**35.3**	**87.8**
Youtube Subtitles	Vanilla OPT	7.7	6.3	2.7	17.8	15.1	13.8	82.2
	HybridRAG	5.8	8.5	5.2	22.3	18.1	17.4	83.5
	Hybrid-RACA w/o FT	5.0	9.9	7.4	22.1	18.4	18.1	83.2
	Hybrid-RACA FT	5.2	13.4	11.0	27.1	22.0	23.0	84.5
	Hybrid-RACA IT	**4.2**	**20.8**	**18.3**	**39.2**	**30.7**	**34.7**	**87.4**

Table 4: Comparison of the utility performance of the OPT-350M-based Hybrid-RACA models and baselines on four datasets

		PPL	GLEU	BLEU-4	ROUGE-1	ROUGE-L	METEOR	BERTScore
Enron Emails	Vanilla OPT	8.5	5.8	2.6	17.4	14.7	13.5	80.1
	Hybrid-RACA	6.3	8.0	5.9	20.0	17.1	15.4	79.6
	Hybrid-RACA w/o FT	4.6	9.0	6.9	20.8	17.9	16.9	80.9
	Hybrid-RACA FT	4.4	13.8	12.1	26.9	22.6	23.3	83.3
	Hybrid-RACA IT	**3.3**	**22.9**	**20.9**	**41.6**	**33.3**	**37.1**	**86.9**
NIH ExPorter	Vanilla OPT	7.4	9.3	4.5	25.9	21.1	18.3	84.8
	HybridRAG	4.4	10.7	7.1	27.4	22.5	20.8	84.9
	Hybrid-RACA w/o FT	4.1	10.9	7.7	26.9	22.5	21.0	84.9
	Hybrid-RACA FT	3.7	16.8	14.4	34.9	28.3	29.3	86.7
	Hybrid-RACA IT	**2.9**	**24.2**	**21.9**	**44.3**	**35.6**	**39.4**	**88.8**
Hacker News	Vanilla OPT	7.5	8.0	4.6	23.1	19.4	15.3	84.1
	Hybrid-RACA	7.2	7.5	4.8	20.9	18.0	13.4	83.4
	Hybrid-RACA w/o FT	5.6	8.9	6.4	22.6	19.4	15.3	83.8
	Hybrid-RACA FT	5.3	14.8	12.8	30.3	24.8	23.6	85.4
	Hybrid-RACA IT	**3.8**	**20.2**	**18.0**	**39.3**	**30.8**	**33.3**	**87.5**
Youtube Subtitles	Vanilla OPT	9.2	5.7	2.2	16.7	14.2	13.1	82.6
	Hybrid-RACA	7.0	7.2	4.1	19.4	16.7	15.5	82.9
	Hybrid-RACA w/o FT	5.9	7.1	4.0	18.2	15.7	14.8	82.1
	Hybrid-RACA FT	5.5	12.5	9.9	26.1	21.4	22.5	84.6
	Hybrid-RACA IT	**4.4**	**20.4**	**17.8**	**38.7**	**30.5**	**34.8**	**87.3**

Table 5: Comparison of the utility performance of the OPT-125M-based Hybrid-RACA models and baselines on four datasets

C Examples of the model completions

Table 6 shows a working example for Hybrid-RACA models and Table 7 and 8 show examples of failing cases.

Table 7 is a failing case for both OPT-125M and OPT-350M Hybrid-RACA models. In this case, the memory doesn't contain the information needed to complete the text. As a large language model, GPT3.5 is capable of ignoring the memory and using its parametric memory to generate the completion. However, the smaller client models tend to pick entities present in the memory for text generation despite that the resulting completion is not factually accurate. Table 8 shows an example of working case for Hybrid-RACA OPT-350M IT model, but a failing case for the OPT-125M based model. In this case, the memories are bullet points generated from several document chunks; the client model with limited reasoning abilities does not allow them to process the memories extensively and reorganize them. We've noticed that when the small OPT client models combine information from different bits of the memories, they usually end up fabricating wrong facts.

Improving the memory generator by reducing duplicate information, and enhancing the reasoning abilities of the client model or encouraging it to select more useful memories and ignore irrelevant content would be some of the ways to address these failing cases and limitations.

prompt	Air Mata Iboe (Perfected spelling: Air
GPT-3.5 generated reference	Mata Ibu) was a musical extravaganza released in December 1941. It was distributed by Columbia Pictures and premiered at Sampoerna Theatre in Surabaya. It was advertised under the Dutch title Tran
Hybrid-RACA OPT-125M IT	Mata Iboe) was released in December 1941 and was distributed by Columbia Pictures. It was advertised under the Dutch title Tranen Eener Moeder and was promoted as a "musical extravaganza".
Hybrid-RACA OPT-350M IT	Mata Iboe) was released in December 1941 and was distributed by Columbia Pictures. It was advertised under the Dutch title Tranen Eener Moeder and was promoted as a "musical extravaganza".

Table 6: A working case for Hybrid-RACA models

prompt	Viva el Príncipe (English: Long Live the Prince) is the thirteenth studio album by Mexican recording artist
GPT-3.5 generated reference	Cristian Castro. It was released on November 30, 2010 and was met with critical acclaim. The album peaked at No. 5 on the Argentine Chamber of Phonograms and Videograms Producers pop charts and received gold
Hybrid-RACA OPT-125M IT	David Jeffries. It was released on November 30, 2010 worldwide and peaked at No. 5 on the Argentine Chamber of Phonograms and Videograms Producers pop charts. The album was certified diamond by AMPROF
Hybrid-RACA OPT-350M IT	David Jeffries. It was released on November 30, 2010 worldwide and peaked at No. 5 on the Argentine Chamber of Phonograms and Videograms Producers pop charts. It was certified diamond by AMPROFON

Table 7: A failing case for Hybrid-RACA models.

prompt	"Make Me Like You" is a song by American singer Gwen Stefani
GPT-3.5 generated reference	, released as the second single from her third studio album This Is What the Truth Feels Like. The track was written by Stefani, Justin Tranter, Julia Michaels, Mattias Larsson and Robin Fredriks
Hybrid-RACA OPT-125M IT	that has become a classic. Released digitally on February 12, 2016 as the album's second single, it was serviced to mainstream radio on February 16, 2016 in the United States. The track was written by Stefani
Hybrid-RACA OPT-350M IT	that was released digitally on February 12, 2016. It was written by Stefani, Justin Tranter, Julia Michaels, Mattias Larsson and Robin Fredriksson under their stage name Mattman & Robin. The

Table 8: A working case for Hybrid-RACA OPT-350M IT but failed for other variants.