

# A Survey on Transformer Context Extension: Approaches and Evaluation

Yijun Liu<sup>1</sup>, Jinzheng Yu<sup>2</sup>, Yang Xu<sup>1</sup>, Zhongyang Li<sup>3</sup>, Qingfu Zhu<sup>1</sup>

<sup>1</sup> Research Center for Social Computing and Interactive Robotics

<sup>1</sup> Harbin Institute of Technology

<sup>2</sup> State Key Laboratory of Media Convergence and Communication

<sup>2</sup> Communication University of China

<sup>3</sup> Huawei Technologies, Co., Ltd.

{yijunliu, qfzhu}@ir.hit.edu.cn

## Abstract

Large language models (LLMs) based on Transformer have been widely applied in the field of natural language processing (NLP), demonstrating strong performance, particularly in handling short text tasks. However, when it comes to long context scenarios, the performance of LLMs degrades due to some challenges. To alleviate this phenomenon, there is a number of work proposed recently. In this survey, we first list the challenges of applying pre-trained LLMs to process long contexts. Then systematically review the approaches related to long context and propose our taxonomy categorizing them into four main types: positional encoding, context compression, retrieval augmented, and attention pattern. In addition to the approaches, we focus on the evaluation of long context, organizing relevant data, tasks, and metrics based on existing long context benchmarks. Finally, we summarize unresolved issues in the long context domain and put forward our views on future developments.

## 1 Introduction

In recent years, the Transformer (Vaswani, 2017) architecture has made significant progress in many NLP tasks (Devlin, 2018; Radford, 2018; Lewis, 2019; Raffel et al., 2020; Brown, 2020; Chen et al., 2021a; Cobbe et al., 2021), and has become the foundational model of many applications. Large language models can handle tasks involving short texts, within the pre-trained context length. However, current scenarios always involve longer texts, such as book-/repo- level tasks (Sharma et al., 2019; Liu et al., 2023a; Zhang et al., 2023a; Liu et al., 2023b), dialogue systems with long contexts (Dey et al., 2022; Li et al., 2024a), content-rich in-context learning (Li et al., 2024c) and so on. The performance of the pre-trained LLMs degrades and the models often fail to utilize the complete knowledge contained within the long context inputs. This may be caused by three inher-

ent challenges: out-of-distribution (OOD) problem (Han et al., 2024), "Lost in the Middle" phenomenon (Liu et al., 2024a), and quadratic complexity of attention (Zhou et al., 2024). Recently, a lot of work has been proposed to improve and evaluate models' ability to handle long contexts in the community.

This survey focuses on approaches and evaluation in the long context field, systematically reviewing existing related work. As illustrated in Figure 1, we propose a novel taxonomy for approaches, categorizing them into four main groups: positional encoding, context compression, retrieval augmented, and attention pattern. Additionally, we focus on the evaluation aspect and organize work on data, tasks, and metrics based on existing benchmarks. In addition to the two main parts of approaches and evaluation, we present our viewpoints on the current unsolved issues and potential future directions in the long context domain. To illustrate the current status more theoretically, we also list the main challenges in the field of long context before introducing specific work. Although most existing methods and benchmarks have not corresponded to them, these challenges are still instructive for the development of approaches and evaluation.

There are also some surveys that focus on the long context domain. They each have their own emphasis, but there is no systematic and comprehensive survey of approaches and evaluation in the field of long context, which can provide researchers with a quick and efficient guide. Some surveys only include a part of the methods, lacking a comprehensive overview of the approaches related to long context. Zhao et al. (2023) focus on work addressing length extrapolation from the perspective of positional encoding, while there are some surveys from the perspective of KV Cache (Li et al., 2025; Shi et al., 2024). Besides, though some surveys have categorized existing work, their taxonomies are not clear, and there are overlaps between categories.

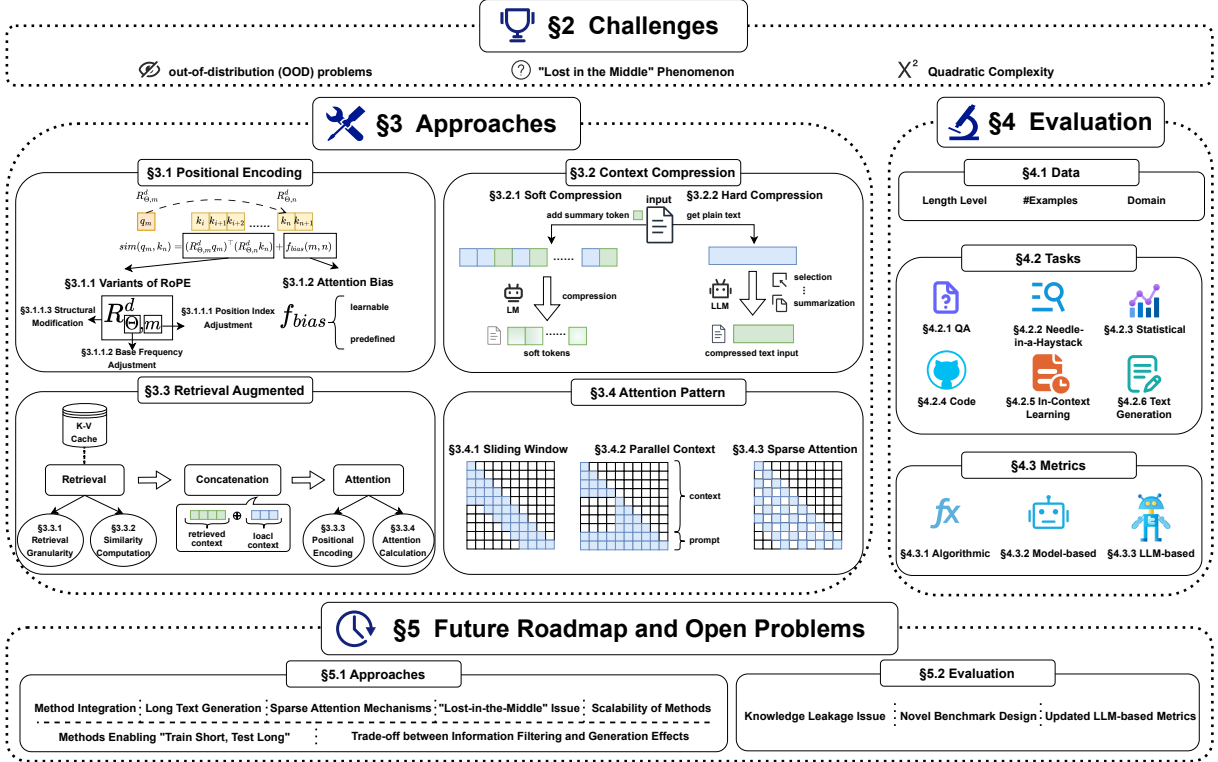


Figure 1: Framework of survey. We first list three inherent challenges in Section 2. And then we systematically review related approaches and propose a novel taxonomy with four major categories in Section 3. Next, in Section 4, we organize the evaluation aspect from three perspectives: data, tasks, and metrics based on existing benchmarks. At last, we show our views on future roadmap and open problems in Section 5.

For example, [Huang et al. \(2023\)](#) divide the methods for enhancing Transformer architecture models into five categories, but some existing methods can belong to multiple categories. And [Pawar et al., 2024](#) also has this problem, which distinguishes existing techniques into two categories: interpolation and extrapolation. Also, some surveys even involve some common methods that not specifically designed for long contexts. [Dong et al. \(2023b\)](#) provide an overview of the text-preprocessing methods, architectures, special characteristics and application for long context, but they cover some general topics. What’s more, these surveys pay little or even no attention to the evaluation aspect.

To fill the above gap, our survey proposes a novel and comprehensive taxonomy on both approaches and evaluation aspects. It is worth noting that we focus on work that applies Transformer-based models to long text tasks, but not improving Transformers (nor other architectures) in a universal scenario. That is to say, this survey does not cover fields like long chain-of-thought reasoning ([Chen et al., 2025](#)), multimodal long context ([Song et al., 2024](#); [Qiu et al., 2024](#)), efficient Transformer ([Zhou et al., 2024](#)), and State Space Model (SSM) ([Wang et al.,](#)

[2024c](#)). In addition, the long context we focus on is the long input content, rather than the introduction of external knowledge in the Retrieval-Augmented Generation (RAG) scenario ([Yu et al., 2024](#); [Zhao et al., 2024](#); [Fan et al., 2024](#)).

## 2 Challenges

When applying pre-trained LLMs to the long context scenarios, there are some inherent challenges affecting models’ performance. We list the three most important and common challenges: OOD problem, "Lost in the Middle" phenomenon, and quadratic complexity.

**OOD Problem** When processing sequences that exceed pre-trained context window length, the models face out-of-distribution (OOD) problems. [Han et al. \(2024\)](#) verify theoretically and empirically that three key factors contribute to OOD issues, thereby limiting models’ extrapolation capabilities: 1) unseen inter-token distances, 2) increased number of attended tokens, and 3) implicitly encoded position information of the starting tokens.

**"Lost in the Middle" Phenomenon** [Liu et al. \(2024a\)](#) discover the "Lost in the middle" phe-

nomenon through experiments that when LLMs receive a long input, they tend to focus on the information at both the beginning and end of the input sequence. At the same time, they neglect the content in the middle, thus failing to capitalize on the key information within the long input.

**Quadratic Complexity** Due to the quadratic complexity of attention, directly using pre-trained LLMs for training or inference on long context is time and resource consuming (Zhou et al., 2024).

The above are three inherent challenges in the field of long context, and some existing methods have alleviated them to a certain extent. But it is worth noting that most of the methods do not start from this perspective. They consider directly improving the performance of downstream tasks. However, we believe that these three challenges are still the fundamental problems that need to be solved. They play a vital role in the proposal of methods and construction of benchmarks. Moreover, they are the focus of subsequent research.

### 3 Approaches

For the existing approaches for long context, we summarize their characteristics and propose a novel taxonomy different from previous work. As illustrated in Figure 1, mainstream methods are divided into four major categories: positional encoding, context compression, retrieval augmented, and attention pattern, which are introduced below. For more details, please see the Appendix A.

#### 3.1 Positional Encoding

Kazemnejad et al. (2024) mention that positional encoding (PE) appears to be a major factor in the length generalization of Transformer. During the inference process, when encountering sequences that exceed the length of the pre-trained window, the model needs to handle the position index that was not encountered during pre-training. This may lead to Out-Of-Distribution (OOD) issues. Thus, we would like to find an appropriate positional encoding method that allows the model to effectively encode position in sequences that exceed the pre-trained window length. Based on the implementation methods, solutions can be categorized into two main types: Variants of Rotary Position Embedding (RoPE, Su et al., 2024) and Attention bias method. The following sections will detail these two methods.

Though the designed positional encoding strategies can alleviate the extrapolation problem, experiments have found that models without positional encoding (NoPE) show better performance than these methods in reasoning tasks (Kazemnejad et al., 2024). That’s because when causal masks are used for decoding, the model reads the sequence sequentially from left to right. And this process naturally incorporates the sequential information of the token. This finding suggests that when designing a position encoding strategy, we may need to consider the way the model processes sequences and the requirements of the task.

##### 3.1.1 Variants of RoPE

Rotary Position Embedding (RoPE, Su et al., 2024) is a positional encoding method utilized in a series of models such as Wang and Komatsuzaki (2021); Touvron et al. (2023); Roziere et al. (2023). RoPE incorporates explicit relative position dependency in self-attention, which can be expressed as

$$\begin{aligned} \text{sim}(q_m, k_n) &= q_m^\top R_{\Theta, n-m}^d k_n \\ &= (R_{\Theta, m}^d q_m)^\top (R_{\Theta, n}^d k_n), \end{aligned} \quad (1)$$

where  $R_{\Theta, m}^d$  is called the rotation matrix. The original RoPE’s extrapolation capability is not very robust and can only maintain performance slightly beyond the pre-trained context length. Consequently, existing work enhances RoPE for better extrapolation. The core of RoPE is the rotation matrix  $R_{\Theta, m}^d$ , which is parameterized by the position index  $m$  and the function family  $\Theta$ . We can optimize RoPE by adjusting these parameters or even the structure of RoPE itself. Existing related work can be divided into three subcategories: position index adjustment, base frequency adjustment, and structural modification.

**Position Index Adjustment** This method involves modifying the allocation or calculation of position index  $m$  to maintain the relative distances between tokens within the model’s pre-trained index range. This can be implemented in three ways. We can adjust the allocation of the position index  $m$  (An et al., 2024). Besides, proportionally scale  $m$  for long sequences to adapt to the pre-trained window (Chen et al., 2023b). What’s more, we can combine the above two methods, reallocating position index to some tokens in the sequence, while scaling the position index for others (Su, 2023).

**Base Frequency Adjustment** From the formula of rotation matrix (see details in Appendix A.1.1),

we can see that each non-zero term is a trigonometric function value with  $\theta_i$  as independent variable. And the value of  $\theta_i$  affects the effect of rotation matrix to a certain extent. Base frequency adjustment is to enhance the model extrapolation performance by modifying  $\theta_i$  in the trigonometric function terms in the rotation matrix. NTK (Neural Tangent Kernel) theory shows that when the input dimension is low and its embedding representation lacks high-frequency components, it is difficult for the neural network to learn high-frequency information (Tancik et al., 2020). Therefore, researchers choose to adjust  $\theta_i$  with the idea of “extrapolation on high-frequency and interpolation on low-frequency”. One strategy is to change the base  $b$  of the exponential terms  $\theta_i$  in the function cluster  $\Theta$ , and change it from the default value  $b = 10000$  to other values which can improve the model extrapolation performance (Peng and Quesnelle, 2023; Roziere et al., 2023). Another strategy is to directly scale  $\theta_i$  (bloc97, 2023; Peng et al., 2023).

**Structural Modification** The methods described above focus on modifying variables in RoPE formula without altering its basic structure. Some existing work explores adjusting the structure of RoPE itself to better extrapolate, which optimizes the original RoPE formula (Sun et al., 2022).

### 3.1.2 Attention Bias

This type of method introduces relative position information by adding a bias related to the relative distance between tokens when calculating the similarity between query and key vectors. The process can be expressed as follows:

$$\text{sim}(q_m, k_n) = q_m^\top k_n + f_{bias}(m, n), \quad (2)$$

where  $f_{bias}(m, n)$  is a bias function that depends on the token position index corresponding to query and key.  $f_{bias}(m, n)$  be divided into two categories: learnable (Raffel et al., 2020; Chi et al., 2022a), predefined (Press et al., 2021; Chi et al., 2022b).

## 3.2 Context Compression

Existing work proposes compressing the long input sequence into a shorter one for representation. These methods can be categorized into two main types by the compression granularity: soft compression and hard compression.

### 3.2.1 Soft Compression

In order to shorten the sequence length, the soft compression method uses the model to compress

the original input token sequence into a shorter summary token sequence. These summary tokens are soft tokens which act as compression representation but do not correspond to words with actual meaning. They are inserted into the original token sequence to form a new input. During the forward pass of the model, the information from the original token sequence is gathered into the summary token sequence, which represents the original input for subsequent operations. Since summary tokens do not appear during the model’s pre-training, additional training is necessary for the model to learn how to generate and utilize these tokens (Bulatov et al., 2022; Li et al., 2023b; Chevalier et al., 2023; Ge et al., 2023; Mu et al., 2024b). This method can shorten the length of the hidden vector sequence, so that enabling it to be processed within the model’s pre-trained window.

### 3.2.2 Hard Compression

This method utilizes some techniques to directly shorten plain text sequence length. This process can be achieved through selection and summarization. It doesn’t introduce additional tokens and targeted training, which makes it can be applied to some black box models (Jiang et al., 2023, 2024b; Chen et al., 2023a).

## 3.3 Retrieval Augmented

Some existing work propose retrieval-augmented methods to enhance model performance on long context tasks by selectively incorporating crucial tokens from history context into attention. With reference to related work, we summarize a processing paradigm for this type of method. Initially, the (*key*, *value*) pairs from history are stored in the KV cache. Then the model retrieves the corresponding token representations from the KV cache at different retrieval granularity levels. This process is based on the similarity between current token and history tokens from KV cache. The top-k relevant tokens are selected as the retrieved context, which is then concatenated with the context within the current window to form a new input. Finally, the model applies appropriate positional encoding to this concatenated context for attention calculation. Below, we summarize the different methods according to each step of the above process:

### 3.3.1 Retrieval Granularity

In the process of long context retrieval, we focus on the most relevant subset of tokens from KV



cache related to the current processing step. Different methods use different retrieval granularity, with the basic being token-level retrieval. Specifically, it involves calculating the similarity of each history token in the KV cache with the current token, and selecting the top-k history tokens' key and value vectors as the retrieval result. Methods applying this strategy include MemTRM (Wu et al., 2022), FoT (Tworkowski et al., 2024), Unlimiformer (Bertsch et al., 2024a), etc. Besides, some work focus on block-level retrieval, which retrieve top-k set of tokens in one step (Wang et al., 2024b; Rubin and Berant, 2023; Xiao et al., 2024; Mohtashami and Jaggi, 2024).

### 3.3.2 Similarity Computation

Almost all existing works compute the inner product of query and key as similarity. This strategy draws from the standard attention mechanism, which calculates the dot product between the query and key to allocate corresponding weights to the value (Vaswani et al., 2023). It is simple to implement and can effectively capture and utilize the similarity information between queries and keys.

### 3.3.3 Positional Encoding

After computing the similarity, we select the top-k relative tokens as the results, and call them retrieved context tokens. Correspondingly, tokens within the current window are called as local context tokens. These two types of context tokens are concatenated to form a new set of context tokens. Before these new context tokens are fed into the model for attention computation, it is necessary to consider suitable positional encoding to distinguish the information of tokens at different positions. Some work choose to assign the same position vector to the retrieved context tokens (Wu et al., 2022; Tworkowski et al., 2024; Xiao et al., 2024), while Mohtashami and Jaggi (2023) choose reallocation strategies.

### 3.3.4 Attention Calculation

Next, when performing attention calculation, we need to consider how to make full use of retrieved context tokens and local context tokens. Different approaches use different strategies. Simply, Tworkowski et al. (2024); Xiao et al. (2024) choose standard attention, while Bertsch et al. (2024a) chooses cross attention. Besides, Wu et al. (2022); Wang et al. (2024b) adopt a Joint Attention method. Landmark employs the Grouped Soft-

max method, a fine-grained approach for calculation (Mohtashami and Jaggi, 2023).

## 3.4 Attention Pattern

There is a class of methods modifying the attention pattern, i.e. the range of tokens attended to. They can better adapt models to expand processing sequence length. Some of them do not require additional training and can be employed as plug-and-play solutions in existing models. These methods can be divided into three main categories: sliding window, parallel context, and sparse attention.

### 3.4.1 Sliding Window

This type of method divides the sequence into segments and performs attention calculation segment by segment without significantly increasing computational complexity. The attention results from earlier segments are stored, which later segments can use during their attention calculation (Dai et al., 2019; Han et al., 2024; Xiao et al., 2023).

### 3.4.2 Parallel Context

The Parallel Context method folds the context part of the input (e.g., in-context examples) into multiple segments. These segments first calculate attention independently, and share the same set of position indexes. And then prompt tokens in the input attend to all the context tokens, so that fully utilize contextual information (Ratner et al., 2022; Hao et al., 2022). These methods require no training and can be plug-and-played into existing models.

### 3.4.3 Sparse Attention

Some work reduce the number of tokens involved in the attention computation, decreasing computational load. They abandon the original attention method which attends to local continuous tokens, while expand the attentive field, and attend to discrete tokens from further context (Ding et al., 2023; Yu et al., 2023; Chen et al., 2023c).

## 4 Evaluation

In the long context scenario, evaluating the model's ability to understand and utilize long context is also a new and critical issue. But as described before, current surveys pay little or even no attention to the evaluation aspect. To fill this gap, we summarize the data, tasks, and metrics of long context evaluation in our survey based on existing benchmarks. The following is a brief introduction, detailed information is in the Appendix B.

Benchmark	Length Level	#Examples	Domain
SCROLLS (Shaham et al., 2022)	1k~4k	119,495	Literature, Dialog
ZeroSCROLLS (Shaham et al., 2023)	0k~16k	4,378	Wiki, Literature, Dialog
LongBench (Bai et al., 2023)	0k~4k, 4k~8k, >8k	4,750	Wiki, Literature, Dialog, Report, Code, News
LooGLE (Li et al., 2023a)	0k~24k	776	Wiki, Paper
BAMBOO (Dong et al., 2023a)	0k~4k, 4k~16k	1,502	Wiki, Dialog, Report, Code, Paper
LongICLBench (Li et al., 2024c)	2k~50k	3,000	Dialog, News, Common Sense
L-Eval (An et al., 2023)	3k~200k	411	Literature, Dialog, News, Paper, Common Sense
Ada-LEval (Wang et al., 2024a)	1k~128k	117,500	Literature, Code
∞Bench (Zhang et al., 2024)	0k~200k	3,946	Literature, Dialog, Code
NeedleBench (Li et al., 2024b)	1k~4k/8k/32k/128k/200k/1m+	-	Wiki, Literature, Dialog, Report, Code, News
LV-Eval (Yuan et al., 2024)	0k~16k/32k/64k/128k/256k	1,729	Wiki, Literature, Dialog, Report, Code, News, Paper

Table 1: Statistics on data characteristics of the datasets in existing long context benchmarks. Length level represents the range of token lengths in the dataset used in the benchmark. #Examples refers to the total number of examples. Domain denotes the data sources. The corresponding contents in table are directly extracted or calculated from the original papers. Given that current models mainly within context windows exceeding 100k tokens, we categorize benchmarks based on this threshold. Benchmarks with contexts exceeding 100K tokens are listed in the lower part.

## 4.1 Data

In order to explore what data should be used to test model’s ability to process long context, we conduct a statistical analysis of datasets in existing benchmarks and summarize their data characteristics.

The evaluation of a model’s long context capability requires not only the long data but also the data diversity and quality. As shown in Table 1, we focus on three characteristics of the datasets in existing long context benchmarks: length level, total number of examples, and the domain it covers.

Besides, we also discuss about knowledge leakage issue, which need to be addressed when constructing the dataset, in the Appendix B.1.2

## 4.2 Tasks

Currently, existing benchmarks propose numerous tasks to evaluate the model’s ability to process long context. But there is no systematic taxonomy for these tasks. Therefore, we divide all tasks in existing benchmarks into seven categories from the perspective of task setting : Question Answering, Needle-in-a-Haystack, Statistical Tasks, Code, In-Context Learning, Text Generation and Other Tasks. Below is the introduction of each type of task, and the details are in the Appendix B.2.

### 4.2.1 Question Answering

**Single-hop Question Answering** requires models to locate and extract answers from a single text passage, typically involving a single fact (Rajpurkar, 2016; Joshi et al., 2017; Kočiský et al., 2018).

**Multi-hop Question Answering** requires models to integrate information from multiple sources to answer complex questions. This often involves reasoning across different pieces of evidence (Ho

et al., 2020; Trivedi et al., 2022; Yang et al., 2018; Chen et al., 2024b; Zhuang et al., 2023).

### 4.2.2 Needle-in-a-Haystack

Needle-in-a-Haystack evaluate LLMs’ ability to extract specific content from long contexts. These tasks can evaluate the model’s retrieval capability, also measure the range of context lengths model can handle (Zhu et al., 2024; Mohtashami and Jaggi, 2023; Zhang et al., 2024; Li et al., 2024b).

### 4.2.3 Statistical Tasks

**Long Arithmetic Calculation** requires models to perform addition and subtraction operations on lengthy arithmetic expressions (Zhang et al., 2024, 2023b; Cobbe et al., 2021; Xu et al., 2024; Chen et al., 2024a).

**Numerical Information Extraction** requires models to identify specific mathematical elements (Zhang et al., 2024; Li et al., 2023a).

**Sentiment Aggregation** requires models to output the percentage of positive reviews when provided with a collection of reviews (Angelidis et al., 2021; Shaham et al., 2023).

**Paragraph Counting** requires models to count the number of unique paragraphs in a set of randomly repeated and shuffled passages (Bai et al., 2023).

### 4.2.4 Code

**Code Completion** requires models to complete missing code fragments based on preceding code and context (Chen et al., 2021a; Zheng et al., 2023; Bai et al., 2023; Guo et al., 2023; Zan et al., 2022; Dong et al., 2023a; Qin et al., 2024).

**Code Running** asks models to infer the output of lengthy programs by tracing a series of cascading

function calls (Bubeck et al., 2023; An et al., 2023; Zhang et al., 2024).

**Code Debugging** requires models to identify deliberately inserted errors (Zhang et al., 2024).

#### 4.2.5 In-Context Learning

The input will contain a certain amount of examples, resulting in a long input. This is caused by the example itself is very long or the number of examples is particularly large. Based on this fact, we divide In-Context Learning task into two categories: long example learning and many-shot learning.

**Long Example Learning** requires models to process extensive inputs with long examples which have large label spaces and generate accurate predictions. This task inherently is a long-context challenge (Bai et al., 2023; Li et al., 2024c; Li and Roth, 2002; NLPCC, 2014).

**Many-shot Learning** leverages the expanded context windows of models to process hundreds or even thousands of examples in order to complete a given task (Yu et al., 2020; Bertsch et al., 2024b).

#### 4.2.6 Text Generation

**Language Modeling** serving as the pre-training task for LLMs, is also a widely used basic task to test the model’s ability to generate text.

**Document Summarization** requires models to make a summary of the input documents, which encompasses single-document and multi-document tasks. Single-document summarization extracts key information from a single document (Wang et al., 2022; Chen et al., 2021b; Huang et al., 2021; Zhong et al., 2021), while multi-document summarization synthesizes information from multiple sources into a comprehensive, non-repetitive summary containing all key points (Bai et al., 2023; An et al., 2023; Fabbri et al., 2019).

**Open-ended Text Generation** requires models to produce coherent and logical content on given topics (Tan et al., 2024; Bai et al., 2024; Kumar et al., 2024; Ni et al., 2024; Rafailov et al., 2024).

#### 4.2.7 Other Tasks

In addition to the six types of tasks listed above, there are some tasks that are not included in this classification system but are equally important for testing the model’s long context ability.

**Reordering** asks models to reconstruct the original sequence of shuffled fragments by considering the broad context and logical relationships (Kryściński et al., 2021; Shaham et al., 2023; Li et al., 2023a; Dong et al., 2023a; Wang et al., 2024a).

**Context Consistency** shows models an academic paper and a hypothesis, requiring models to judge whether the hypothesis is supported or contradicted by the ideas in the paper (Dong et al., 2023a).

**Summary Source Paragraph Identification** challenges models to identify the original source paragraphs for given summaries (Bai et al., 2023).

**Character Identification** requires models to identify different speakers in long dialogues by recognizing their distinct characteristics (TVMEG, 2024; Senedd Cymru, 2024; Zhang et al., 2024; Dong et al., 2023a; Chen et al., 2021b).

### 4.3 Metrics

In addition to data and tasks, metrics can directly reflect the model’s ability to handle long contexts. With current long context task designs gradually changing from classic NLP tasks to more practical tasks, the requirements for metrics are constantly increasing. We organize metrics for testing models’ capabilities on long context according to the three stages of metrics development: Algorithmic Metrics, Model-based Metrics, and LLM-based Metrics. From these three metrics stages, it can be seen that the metrics development trend becomes more and more complex and flexible.

#### 4.3.1 Algorithmic Metrics

Algorithmic metrics are calculated based on the model output or logits through defined formulas. Their implementation is very simple and can reflect the effect of language modeling and some downstream tasks to a certain extent.

Perplexity (PPL) is one of the most common algorithmic metrics used in existing long context benchmarks (Beltagy et al., 2020; Roy et al., 2021; Press et al., 2021). Meanwhile, some benchmarks employ other algorithmic metrics such as accuracy, f1, and N-gram-based metrics (ROUGE, Lin, 2004 and BLEU, Papineni et al., 2002, etc.) to evaluate LLMs on certain downstream tasks (Shaham et al., 2023; Bai et al., 2023; Kasai et al., 2021).

However, these algorithmic metrics have several limitations, such as content quality, syntactic accuracy, and human correlation issues (Reiter and Belz, 2009; Stent et al., 2005; Sun et al., 2021; An et al., 2023; Improving; Tan et al., 2024). This causes algorithmic metrics to be limited in reflecting the model’s ability to process long context. A number of approaches have been developed to improve algorithmic metrics. Such as enhancing scoring techniques, restructuring task formats and so on (Yuan

et al., 2024; Dong et al., 2023a; Li et al., 2024b).

#### 4.3.2 Model-based Metrics

To improve the consistency with human judgments, pre-trained language models are being employed to evaluate (Zhang et al., 2020; Yuan et al., 2021). Specifically, pre-trained models (such as BERT, Devlin, 2018, BART, Lewis, 2019, etc.) are used to calculate the similarity score between the model output and reference text to evaluate the performance of downstream tasks.

However, these model-based metrics entirely rely on representations learned from pre-trained language models and require reference texts. They may not be accurate enough for evaluating some novel and creative text generation tasks.

#### 4.3.3 LLM-based Metrics

Combining the above two metrics issues, LLM-based metrics are proposed, utilizing sufficient knowledge within LLMs for evaluation. For example, LLM-based metrics prompt LLMs to offer human-like multi-dimensional assessment (Wang et al., 2023a; Li et al., 2023a; Shen et al., 2023; Chiang and Lee, 2023; Zhang et al., 2024; Zheng et al., 2024; Liu et al., 2023c; Tan et al., 2024; Mu et al., 2024a) and interpretable reasoning (Wang et al., 2023b; Luo et al., 2023; Wu et al., 2023).

LLM-based metrics fundamentally distinguish from the other two metrics, which behave much more mechanically. In addition, they demonstrate enhanced agreement with human evaluations (Wang et al., 2023a; Li et al., 2023a). Due to this higher consistency and wider scope of application, LLM-based metrics are gaining increasing attention in long-context evaluation.

## 5 Future Roadmap and Open Problems

Despite the rapid development of long context techniques, numerous challenges remain unresolved. Looking to future roadmap, we list vital open problems and present our perspectives on the developments. They are also divided into two parts: approaches and evaluation.

### 5.1 Approaches

**Method Integration** would combine methods' strengths to address the challenges of extrapolating long context from multiple perspectives.

**Long Text Generation** remains under-researched, which concentrate on effective long-text generation techniques and the evaluation of generation quality.

**Sparse Attention Mechanisms** may lead to a decrease in models' original language ability, thereby limiting their potential for processing long context.

**"Lost-in-the-Middle" Issue** has not yet been completely resolved, there is a lack of targeted solutions and appropriate verification methods.

**Scalability of Methods** requires to explore how existing methods can be adapted to models of different scales or even different architectural frameworks, enhancing their generality and applicability.

**Methods Enabling "Train Short, Test Long"** haven't emerged, which train on short texts while excelling in long-context. These methods can reduce resource needs and improve generalization.

**Trade-off between Information Filtering and Generation Effects** means existing methods can be optimized by integrating RAG to enhance efficiency and quality without too long input.

## 5.2 Evaluation

**Knowledge Leakage Issue** is ever-present. As LLMs gain the ability to gather information from the Internet and their training data scope expands, existing solutions become increasingly ineffective and some operations may limit innovation.

**Novel Benchmark Design** needed to be proposed. We need to construct benchmarks with coherent content and long-distance dependencies to more effectively test the model's ability to process long context. For example, asking models to process inputs from multiple books.

**Updated LLM-based Metrics** are a development direction. Though LLM-based metrics show higher consistency with human judgments than other metrics, they are costly, have random outputs, and even lack human emotions. We need to combine LLM with other techniques to better evaluate.

## 6 Conclusion

In this survey, we first list three inherent challenges in processing long context. And then we propose a novel taxonomy for long context approaches and summarize the similarities and differences in each category. In addition, we systematically review the work on evaluation, summarize the data, tasks, and metrics related to long context based on existing benchmark. Finally, we list unsolved issues and put forward our insights on the future development of long context domain.



## Limitations

This survey summarizes the approaches and evaluation in the area of long context, and gives our views on future development. However, we don't cover efficient transformer on long context, multimodel long context, etc. In addition, due to limitations in space, we are not able to include all related work.

Due to the rapidly evolving nature of the field of Transformer context extension, our survey may not capture the latest developments, particularly those that emerged near or after the time of writing.

## References

- Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.
- Rishabh Agarwal, Avi Singh, Lei M Zhang, Bernd Bohnet, Stephanie Chan, Ankesh Anand, Zaheer Abbas, Azade Nova, John D Co-Reyes, Eric Chu, et al. 2024. Many-shot in-context learning. *arXiv preprint arXiv:2404.11018*.
- Chenxin An, Shansan Gong, Ming Zhong, Xingjian Zhao, Mukai Li, Jun Zhang, Lingpeng Kong, and Xipeng Qiu. 2023. L-eval: Instituting standardized evaluation for long context language models. *arXiv preprint arXiv:2307.11088*.
- Chenxin An, Fei Huang, Jun Zhang, Shansan Gong, Xipeng Qiu, Chang Zhou, and Lingpeng Kong. 2024. Training-free long-context scaling of large language models. *arXiv preprint arXiv:2402.17463*.
- Stefanos Angelidis, Reinald Kim Amplayo, Yoshihiko Suhara, Xiaolan Wang, and Mirella Lapata. 2021. Extractive opinion summarization in quantized transformer spaces. *Transactions of the Association for Computational Linguistics*, 9:277–293.
- Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, et al. 2023. Longbench: A bilingual, multitask benchmark for long context understanding. *arXiv preprint arXiv:2308.14508*.
- Yushi Bai, Jiajie Zhang, Xin Lv, Linzhi Zheng, Siqi Zhu, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. 2024. Longwriter: Unleashing 10,000+ word generation from long context llms. *arXiv preprint arXiv:2408.07055*.
- Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.
- Amanda Bertsch, Uri Alon, Graham Neubig, and Matthew Gormley. 2024a. Unlimiformer: Long-range transformers with unlimited length input. *Advances in Neural Information Processing Systems*, 36.
- Amanda Bertsch, Maor Ivgi, Uri Alon, Jonathan Berant, Matthew R Gormley, and Graham Neubig. 2024b. In-context learning with long-context models: An in-depth exploration. *arXiv preprint arXiv:2405.00200*.
- bloc97. 2023. Add NTK-Aware interpolation "by parts" correction. <https://github.com/jquesnelle/scaled-rope/pull/1>.
- Tom B Brown. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.
- Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrmke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. 2023. Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint arXiv:2303.12712*.
- Aydar Bulatov, Yury Kuratov, and Mikhail Burtsev. 2022. Recurrent memory transformer. *Advances in Neural Information Processing Systems*, 35:11079–11091.
- Howard Chen, Ramakanth Pasunuru, Jason Weston, and Asli Celikyilmaz. 2023a. Walking down the memory maze: Beyond context limit through interactive reading. *arXiv preprint arXiv:2310.05029*.
- Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. 2021a. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.
- Mingda Chen, Zewei Chu, Sam Wiseman, and Kevin Gimpel. 2021b. Summscreen: A dataset for abstractive screenplay summarization. *arXiv preprint arXiv:2104.07091*.
- Qiguang Chen, Libo Qin, Jinhao Liu, Dengyun Peng, Jiannan Guan, Peng Wang, Mengkang Hu, Yuhang Zhou, Te Gao, and Wanxiang Che. 2025. Towards reasoning era: A survey of long chain-of-thought for reasoning large language models. *Preprint, arXiv:2503.09567*.
- Qiguang Chen, Libo Qin, Jiaqi Wang, Jinxuan Zhou, and Wanxiang Che. 2024a. Unlocking the boundaries of thought: A reasoning granularity framework to quantify and optimize chain-of-thought.
- Shouyuan Chen, Sherman Wong, Liangjian Chen, and Yuandong Tian. 2023b. Extending context window of large language models via positional interpolation. *arXiv preprint arXiv:2306.15595*.

- Yukang Chen, Shengju Qian, Haotian Tang, Xin Lai, Zhijian Liu, Song Han, and Jiaya Jia. 2023c. Longlora: Efficient fine-tuning of long-context large language models. *arXiv preprint arXiv:2309.12307*.
- Zhi Chen, Qiguang Chen, Libo Qin, Qipeng Guo, Haijun Lv, Yicheng Zou, Wanxiang Che, Hang Yan, Kai Chen, and Dahua Lin. 2024b. What are the essential factors in crafting effective long context multi-hop instruction datasets? insights and best practices. *arXiv preprint arXiv:2409.01893*.
- Alexis Chevalier, Alexander Wettig, Anirudh Ajith, and Danqi Chen. 2023. Adapting language models to compress contexts. *arXiv preprint arXiv:2305.14788*.
- Ta-Chung Chi, Ting-Han Fan, Peter J Ramadge, and Alexander Rudnicky. 2022a. Kerple: Kernelized relative positional embedding for length extrapolation. *Advances in Neural Information Processing Systems*, 35:8386–8399.
- Ta-Chung Chi, Ting-Han Fan, Alexander I Rudnicky, and Peter J Ramadge. 2022b. Dissecting transformer length extrapolation via the lens of receptive field analysis. *arXiv preprint arXiv:2212.10356*.
- Cheng-Han Chiang and Hung-yi Lee. 2023. Can large language models be an alternative to human evaluations? *arXiv preprint arXiv:2305.01937*.
- 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*.
- Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G 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.
- Jacob Devlin. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.
- Suvodip Dey, Maunendra Sankar Desarkar, Asif Ekbal, and PK Srijith. 2022. Dialogen: Generalized long-range context representation for dialogue systems. *arXiv preprint arXiv:2210.06282*.
- Jiayu Ding, Shuming Ma, Li Dong, Xingxing Zhang, Shaohan Huang, Wenhui Wang, Nanning Zheng, and Furu Wei. 2023. Longnet: Scaling transformers to 1,000,000,000 tokens. *arXiv preprint arXiv:2307.02486*.
- Zican Dong, Tianyi Tang, Junyi Li, Wayne Xin Zhao, and Ji-Rong Wen. 2023a. Bamboo: A comprehensive benchmark for evaluating long text modeling capacities of large language models. *arXiv preprint arXiv:2309.13345*.
- Zican Dong, Tianyi Tang, Lunyi Li, and Wayne Xin Zhao. 2023b. A survey on long text modeling with transformers. *arXiv preprint arXiv:2302.14502*.
- Alexander R Fabbri, Irene Li, Tianwei She, Suyi Li, and Dragomir R Radev. 2019. Multi-news: A large-scale multi-document summarization dataset and abstractive hierarchical model. *arXiv preprint arXiv:1906.01749*.
- Wenqi Fan, Yajuan Ding, Liangbo Ning, Shijie Wang, Hengyun Li, Dawei Yin, Tat-Seng Chua, and Qing Li. 2024. A survey on rag meeting llms: Towards retrieval-augmented large language models. *Preprint*, arXiv:2405.06211.
- Jinlan Fu, See-Kiong Ng, Zhengbao Jiang, and Pengfei Liu. 2023. Gptscore: Evaluate as you desire. *arXiv preprint arXiv:2302.04166*.
- Tao Ge, Jing Hu, Lei Wang, Xun Wang, Si-Qing Chen, and Furu Wei. 2023. In-context autoencoder for context compression in a large language model. *arXiv preprint arXiv:2307.06945*.
- Shahriar Golchin and Mihai Surdeanu. 2023. Time travel in llms: Tracing data contamination in large language models. *arXiv preprint arXiv:2308.08493*.
- Daya Guo, Canwen Xu, Nan Duan, Jian Yin, and Julian McAuley. 2023. Longcoder: A long-range pre-trained language model for code completion. In *International Conference on Machine Learning*, pages 12098–12107. PMLR.
- Chi Han, Qifan Wang, Hao Peng, Wenhan Xiong, Yu Chen, Heng Ji, and Sinong Wang. 2024. Lm-infinite: Zero-shot extreme length generalization for large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 3991–4008.
- Yaru Hao, Yutao Sun, Li Dong, Zhixiong Han, Yuxian Gu, and Furu Wei. 2022. Structured prompting: Scaling in-context learning to 1,000 examples. *arXiv preprint arXiv:2212.06713*.
- Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. *arXiv preprint arXiv:2011.01060*.
- Luyang Huang, Shuyang Cao, Nikolaus Parulian, Heng Ji, and Lu Wang. 2021. Efficient attentions for long document summarization. *arXiv preprint arXiv:2104.02112*.
- Yunpeng Huang, Jingwei Xu, Junyu Lai, Zixu Jiang, Taolue Chen, Zenan Li, Yuan Yao, Xiaoxing Ma, Lijuan Yang, Hao Chen, et al. 2023. Advancing transformer architecture in long-context large language models: A comprehensive survey. *arXiv preprint arXiv:2311.12351*.

- NLG Improving. Not all metrics are guilty: Improving nlg evaluation with llm paraphrasing.
- Huiqiang Jiang, Qianhui Wu, , Xufang Luo, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 2024a. [LongLLMLingua: Accelerating and enhancing LLMs in long context scenarios via prompt compression](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1658–1677, Bangkok, Thailand. Association for Computational Linguistics.
- Huiqiang Jiang, Qianhui Wu, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 2023. LlmLingua: Compressing prompts for accelerated inference of large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13358–13376.
- Huiqiang Jiang, Qianhui Wu, Xufang Luo, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 2024b. [Longllmlingua: Accelerating and enhancing llms in long context scenarios via prompt compression](#). Preprint, arXiv:2310.06839.
- Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv preprint arXiv:1705.03551*.
- Jungo Kasai, Keisuke Sakaguchi, Ronan Le Bras, Lavinia Dunagan, Jacob Morrison, Alexander R Fabbri, Yejin Choi, and Noah A Smith. 2021. Bidimensional leaderboards: Generate and evaluate language hand in hand. *arXiv preprint arXiv:2112.04139*.
- Amirhossein Kazemnejad, Inkit Padhi, Karthikeyan Natesan Ramamurthy, Payel Das, and Siva Reddy. 2024. The impact of positional encoding on length generalization in transformers. *Advances in Neural Information Processing Systems*, 36.
- Tomáš Kočiský, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. 2018. The narrativeqa reading comprehension challenge. *Transactions of the Association for Computational Linguistics*, 6:317–328.
- Wojciech Kryściński, Nazneen Rajani, Divyansh Agarwal, Caiming Xiong, and Dragomir Radev. 2021. Booksum: A collection of datasets for long-form narrative summarization. *arXiv preprint arXiv:2105.08209*.
- Ishita Kumar, Snigdha Viswanathan, Sushrita Yerra, Alireza Salemi, Ryan A Rossi, Franck Dernoncourt, Hanieh Deilamsalehy, Xiang Chen, Ruiyi Zhang, Shubham Agarwal, et al. 2024. Longlamp: A benchmark for personalized long-form text generation. *arXiv preprint arXiv:2407.11016*.
- M Lewis. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.
- Haoyang Li, Yiming Li, Anxin Tian, Tianhao Tang, Zhanchao Xu, Xuejia Chen, Nicole Hu, Wei Dong, Qing Li, and Lei Chen. 2025. [A survey on large language model acceleration based on kv cache management](#). Preprint, arXiv:2412.19442.
- Jia-Nan Li, Quan Tu, Cunli Mao, Zhengtao Yu, Ji-Rong Wen, and Rui Yan. 2024a. Streamingdialogue: Prolonged dialogue learning via long context compression with minimal losses. *arXiv preprint arXiv:2403.08312*.
- Jiaqi Li, Mengmeng Wang, Zilong Zheng, and Muhan Zhang. 2023a. Loogole: Can long-context language models understand long contexts? *arXiv preprint arXiv:2311.04939*.
- Mo Li, Songyang Zhang, Yunxin Liu, and Kai Chen. 2024b. Needlebench: Can llms do retrieval and reasoning in 1 million context window? *arXiv preprint arXiv:2407.11963*.
- Tianle Li, Ge Zhang, Quy Duc Do, Xiang Yue, and Wenhui Chen. 2024c. Long-context llms struggle with long in-context learning. *arXiv preprint arXiv:2404.02060*.
- Xianming Li, Zongxi Li, Xiaotian Luo, Haoran Xie, Xing Lee, Yingbin Zhao, Fu Lee Wang, and Qing Li. 2023b. Recurrent attention networks for long-text modeling. *arXiv preprint arXiv:2306.06843*.
- Xin Li and Dan Roth. 2002. Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*.
- Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.
- Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024a. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173.
- Shuaiqi Liu, Jiannong Cao, Zhongfen Deng, Wenting Zhao, Ruosong Yang, Zhiyuan Wen, and S Yu Philip. 2023a. Neural abstractive summarization for long text and multiple tables. *IEEE Transactions on Knowledge and Data Engineering*.
- Tianyang Liu, Canwen Xu, and Julian McAuley. 2023b. Repobench: Benchmarking repository-level code auto-completion systems. *arXiv preprint arXiv:2306.03091*.
- Xiaoran Liu, Hang Yan, Shuo Zhang, Chenxin An, Xipeng Qiu, and Dahua Lin. 2024b. [Scaling laws of rope-based extrapolation](#). Preprint, arXiv:2310.05209.
- Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023c. G-eval: Nlg evaluation using gpt-4 with better human alignment. *arXiv preprint arXiv:2303.16634*.

- Zheheng Luo, Qianqian Xie, and Sophia Ananiadou. 2023. Chatgpt as a factual inconsistency evaluator for text summarization. *arXiv preprint arXiv:2303.15621*.
- Amirkeivan Mohtashami and Martin Jaggi. 2023. Landmark attention: Random-access infinite context length for transformers. *arXiv preprint arXiv:2305.16300*.
- Amirkeivan Mohtashami and Martin Jaggi. 2024. Random-access infinite context length for transformers. *Advances in Neural Information Processing Systems*, 36.
- Honglin Mu, Yang Xu, Yunlong Feng, Xiaofeng Han, Yitong Li, Yutai Hou, and Wanxiang Che. 2024a. [Beyond static evaluation: A dynamic approach to assessing ai assistants’ api invocation capabilities](#). *Preprint*, arXiv:2403.11128.
- Jesse Mu, Xiang Li, and Noah Goodman. 2024b. Learning to compress prompts with gist tokens. *Advances in Neural Information Processing Systems*, 36.
- Xuanfan Ni, Hengyi Cai, Xiaochi Wei, Shuaiqiang Wang, Dawei Yin, and Piji Li. 2024.  $Xl^2$  bench: A benchmark for extremely long context understanding with long-range dependencies. *arXiv preprint arXiv:2404.05446*.
- NLPCC. 2014. Task definition for large scale text categorization at nlpcc 2014.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.
- Saurav Pawar, SM Tonmoy, SM Zaman, Vinija Jain, Aman Chadha, and Amitava Das. 2024. The what, why, and how of context length extension techniques in large language models—a detailed survey. *arXiv preprint arXiv:2401.07872*.
- Bowen Peng and Jeffrey Quesnelle. 2023. Ntk-aware scaled rope allows llama models to have extended (8k+) context size without any fine-tuning and minimal perplexity degradation.
- Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. 2023. Yarn: Efficient context window extension of large language models. *arXiv preprint arXiv:2309.00071*.
- Ofir Press, Noah A Smith, and Mike Lewis. 2021. Train short, test long: Attention with linear biases enables input length extrapolation. *arXiv preprint arXiv:2108.12409*.
- Libo Qin, Qiguang Chen, Xiachong Feng, Yang Wu, Yongheng Zhang, Yinghui Li, Min Li, Wanxiang Che, and Philip S Yu. 2024. Large language models meet nlp: A survey. *arXiv preprint arXiv:2405.12819*.
- Han Qiu, Jiaying Huang, Peng Gao, Qin Qi, Xiaojin Zhang, Ling Shao, and Shijian Lu. 2024. [Longhalqa: Long-context hallucination evaluation for multimodal large language models](#). *Preprint*, arXiv:2410.09962.
- Alec Radford. 2018. Improving language understanding by generative pre-training.
- Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2024. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36.
- Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.
- P Rajpurkar. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.
- Nir Ratner, Yoav Levine, Yonatan Belinkov, Ori Ram, Inbal Magar, Omri Abend, Ehud Karpas, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. 2022. Parallel context windows for large language models. *arXiv preprint arXiv:2212.10947*.
- Ehud Reiter and Anja Belz. 2009. An investigation into the validity of some metrics for automatically evaluating natural language generation systems. *Computational Linguistics*, 35(4):529–558.
- Aurko Roy, Mohammad Saffar, Ashish Vaswani, and David Grangier. 2021. Efficient content-based sparse attention with routing transformers. *Transactions of the Association for Computational Linguistics*, 9:53–68.
- Baptiste Roziere, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Romain Sauvestre, Tal Remez, et al. 2023. Code llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*.
- Ohad Rubin and Jonathan Berant. 2023. Long-range language modeling with self-retrieval. *arXiv preprint arXiv:2306.13421*.
- Senedd Cymru. 2024. Record of proceedings - senedd. <https://record.assembly.wales/>. Accessed: 2024-10-15.
- Uri Shaham, Maor Ivgi, Avia Efrat, Jonathan Berant, and Omer Levy. 2023. Zeroscrolls: A zero-shot benchmark for long text understanding. *arXiv preprint arXiv:2305.14196*.
- Uri Shaham, Elad Segal, Maor Ivgi, Avia Efrat, Ori Yoran, Adi Haviv, Ankit Gupta, Wenhan Xiong, Mor Geva, Jonathan Berant, et al. 2022. Scrolls: Standardized comparison over long language sequences. *arXiv preprint arXiv:2201.03533*.



- Eva Sharma, Chen Li, and Lu Wang. 2019. Bigpatent: A large-scale dataset for abstractive and coherent summarization. *arXiv preprint arXiv:1906.03741*.
- Chenhui Shen, Liying Cheng, Xuan-Phi Nguyen, Yang You, and Lidong Bing. 2023. Large language models are not yet human-level evaluators for abstractive summarization. *arXiv preprint arXiv:2305.13091*.
- Luohe Shi, Hongyi Zhang, Yao Yao, Zuchao Li, and Hai Zhao. 2024. [Keep the cost down: A review on methods to optimize llm’s kv-cache consumption](#). *Preprint*, arXiv:2407.18003.
- Dingjie Song, Shunian Chen, Guiming Hardy Chen, Fei Yu, Xiang Wan, and Benyou Wang. 2024. Milebench: Benchmarking mllms in long context. *arXiv preprint arXiv:2404.18532*.
- Amanda Stent, Matthew Marge, and Mohit Singhai. 2005. Evaluating evaluation methods for generation in the presence of variation. In *International conference on intelligent text processing and computational linguistics*, pages 341–351. Springer.
- Jianlin Su. 2023. Rectified rotary position embeddings. <https://github.com/bojone/rerope>.
- Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. 2024. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063.
- Simeng Sun, Kalpesh Krishna, Andrew Mattarella-Micke, and Mohit Iyyer. 2021. Do long-range language models actually use long-range context? *arXiv preprint arXiv:2109.09115*.
- Yutao Sun, Li Dong, Barun Patra, Shuming Ma, Shao-han Huang, Alon Benhaim, Vishrav Chaudhary, Xia Song, and Furu Wei. 2022. A length-extrapolatable transformer. *arXiv preprint arXiv:2212.10554*.
- Haochen Tan, Zhijiang Guo, Zhan Shi, Lu Xu, Zhili Liu, Xiaoguang Li, Yasheng Wang, Lifeng Shang, Qun Liu, and Linqi Song. 2024. Proxyqa: An alternative framework for evaluating long-form text generation with large language models. *arXiv preprint arXiv:2401.15042*.
- Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. 2020. Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in neural information processing systems*, 33:7537–7547.
- 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*.
- Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. Musique: Multi-hop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554.
- TVMEG. 2024. Tvmeg - tv mega engine. <https://tvmeg.com/>. Accessed: 2024-10-15.
- Szymon Tworowski, Konrad Staniszewski, Mikołaj Pacek, Yuhuai Wu, Henryk Michalewski, and Piotr Miłoś. 2024. Focused transformer: Contrastive training for context scaling. *Advances in Neural Information Processing Systems*, 36.
- A Vaswani. 2017. Attention is all you need. *Advances in Neural Information Processing Systems*.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2023. [Attention is all you need](#). *Preprint*, arXiv:1706.03762.
- Alex Wang, Richard Yuanzhe Pang, Angelica Chen, Jason Phang, and Samuel R Bowman. 2022. Squality: Building a long-document summarization dataset the hard way. *arXiv preprint arXiv:2205.11465*.
- Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. <https://github.com/kingoflolz/mesh-transformer-jax>.
- Chonghua Wang, Haodong Duan, Songyang Zhang, Dahua Lin, and Kai Chen. 2024a. Ada-level: Evaluating long-context llms with length-adaptable benchmarks. *arXiv preprint arXiv:2404.06480*.
- Jiaan Wang, Yunlong Liang, Fandong Meng, Zengkui Sun, Haoxiang Shi, Zhixu Li, Jinan Xu, Jianfeng Qu, and Jie Zhou. 2023a. Is chatgpt a good nlg evaluator? a preliminary study. *arXiv preprint arXiv:2303.04048*.
- Peiyi Wang, Lei Li, Liang Chen, Zefan Cai, Dawei Zhu, Binghuai Lin, Yunbo Cao, Qi Liu, Tianyu Liu, and Zhifang Sui. 2023b. Large language models are not fair evaluators. *arXiv preprint arXiv:2305.17926*.
- Weizhi Wang, Li Dong, Hao Cheng, Xiaodong Liu, Xifeng Yan, Jianfeng Gao, and Furu Wei. 2024b. Augmenting language models with long-term memory. *Advances in Neural Information Processing Systems*, 36.
- Xiao Wang, Shiao Wang, Yuhe Ding, Yuehang Li, Wentao Wu, Yao Rong, Weizhe Kong, Ju Huang, Shihao Li, Haoxiang Yang, Ziwen Wang, Bo Jiang, Chenglong Li, Yaowei Wang, Yonghong Tian, and Jin Tang. 2024c. [State space model for new-generation network alternative to transformers: A survey](#). *Preprint*, arXiv:2404.09516.
- Ning Wu, Ming Gong, Linjun Shou, Shining Liang, and Daxin Jiang. 2023. Large language models are diverse role-players for summarization evaluation. In

- CCF International Conference on Natural Language Processing and Chinese Computing, pages 695–707. Springer.
- Yuhuai Wu, Markus N Rabe, DeLesley Hutchins, and Christian Szegedy. 2022. Memorizing transformers. *arXiv preprint arXiv:2203.08913*.
- Chaojun Xiao, Pengl Zhang, Xu Han, Guangxuan Xiao, Yankai Lin, Zhengyan Zhang, Zhiyuan Liu, Song Han, and Maosong Sun. 2024. Infilin: Unveiling the intrinsic capacity of llms for understanding extremely long sequences with training-free memory. *arXiv preprint arXiv:2402.04617*.
- Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. 2023. Efficient streaming language models with attention sinks. *arXiv preprint arXiv:2309.17453*.
- Xin Xu, Tong Xiao, Zitong Chao, Zhenya Huang, Can Yang, and Yang Wang. 2024. Can llms solve longer math word problems better? *arXiv preprint arXiv:2405.14804*.
- Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600*.
- Dian Yu, Kai Sun, Claire Cardie, and Dong Yu. 2020. Dialogue-based relation extraction. *arXiv preprint arXiv:2004.08056*.
- Hao Yu, Aoran Gan, Kai Zhang, Shiwei Tong, Qi Liu, and Zhaofeng Liu. 2024. [Evaluation of retrieval-augmented generation: A survey](#). *Preprint*, arXiv:2405.07437.
- Lili Yu, Dániel Simig, Colin Flaherty, Armen Aghajanyan, Luke Zettlemoyer, and Mike Lewis. 2023. Megabyte: Predicting million-byte sequences with multiscale transformers. *Advances in Neural Information Processing Systems*, 36:78808–78823.
- Tao Yuan, Xuefei Ning, Dong Zhou, Zhijie Yang, Shiyao Li, Minghui Zhuang, Zheyue Tan, Zhuyu Yao, Dahua Lin, Boxun Li, et al. 2024. Lv-eval: A balanced long-context benchmark with 5 length levels up to 256k. *arXiv preprint arXiv:2402.05136*.
- Weizhe Yuan, Graham Neubig, and Pengfei Liu. 2021. Bartscore: Evaluating generated text as text generation. *Advances in Neural Information Processing Systems*, 34:27263–27277.
- Daoguang Zan, Bei Chen, Zeqi Lin, Bei Guan, Yongji Wang, and Jian-Guang Lou. 2022. When language model meets private library. *arXiv preprint arXiv:2210.17236*.
- Fengji Zhang, Bei Chen, Yue Zhang, Jacky Keung, Jin Liu, Daoguang Zan, Yi Mao, Jian-Guang Lou, and Weizhu Chen. 2023a. Repocoder: Repository-level code completion through iterative retrieval and generation. *arXiv preprint arXiv:2303.12570*.
- Lei Zhang, Yunshui Li, Ziqiang Liu, Junhao Liu, Min Yang, et al. 2023b. Marathon: A race through the realm of long context with large language models. *arXiv preprint arXiv:2312.09542*.
- Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. *arxiv. arXiv preprint arXiv:1904.09675*.
- Xinrong Zhang, Yingfa Chen, Shengding Hu, Zihang Xu, Junhao Chen, Moo Hao, Xu Han, Zhen Thai, Shuo Wang, Zhiyuan Liu, et al. 2024.  $\infty$ bench: Extending long context evaluation beyond 100k tokens. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15262–15277.
- Liang Zhao, Xiaocheng Feng, Xiachong Feng, Bin Qin, and Ting Liu. 2023. Length extrapolation of transformers: A survey from the perspective of position encoding. *arXiv preprint arXiv:2312.17044*.
- Penghao Zhao, Hailin Zhang, Qinhan Yu, Zhengren Wang, Yunteng Geng, Fangcheng Fu, Ling Yang, Wentao Zhang, Jie Jiang, and Bin Cui. 2024. [Retrieval-augmented generation for ai-generated content: A survey](#). *Preprint*, arXiv:2402.19473.
- Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. 2024. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36.
- Qinkai Zheng, Xiao Xia, Xu Zou, Yuxiao Dong, Shan Wang, Yufei Xue, Lei Shen, Zihan Wang, Andi Wang, Yang Li, et al. 2023. Codegeex: A pre-trained model for code generation with multilingual benchmarking on humaneval-x. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 5673–5684.
- Ming Zhong, Da Yin, Tao Yu, Ahmad Zaidi, Mutethia Mutuma, Rahul Jha, Ahmed Hassan Awadallah, Asli Celikyilmaz, Yang Liu, Xipeng Qiu, et al. 2021. Qmsum: A new benchmark for query-based multi-domain meeting summarization. *arXiv preprint arXiv:2104.05938*.
- Zixuan Zhou, Xuefei Ning, Ke Hong, Tianyu Fu, Jiaming Xu, Shiyao Li, Yuming Lou, Luning Wang, Zhihang Yuan, Xiuhong Li, et al. 2024. A survey on efficient inference for large language models. *arXiv preprint arXiv:2404.14294*.
- Dawei Zhu, Liang Wang, Nan Yang, Yifan Song, Wenhao Wu, Furu Wei, and Sujian Li. 2024. Longembed: Extending embedding models for long context retrieval. *arXiv preprint arXiv:2404.12096*.
- Ziyu Zhuang, Qiguang Chen, Longxuan Ma, Mingda Li, Yi Han, Yushan Qian, Haopeng Bai, Zixian Feng, Weinan Zhang, and Ting Liu. 2023. [Through the lens of core competency: Survey on evaluation of large language models](#). *ArXiv*, abs/2308.07902.

## A Details of Approaches

This section serves as a supplement to the Approaches section 3 in the main text, expanding on relevant details about related methods to provide readers with a deeper understanding.

### A.1 Postional Encoding

#### A.1.1 Variants of RoPE

Su et al. (2024) try to seek a positional encoding method that could encode relative position during the computing query and key similarity, and decompose this process into the representations of the query and key. They conduct theoretical analysis, and propose a novel positional encoding, which transform similarity into following formula:

$$\begin{aligned} \text{sim}(q_m, k_n) &= q_m^\top R_{\Theta, n-m}^d k_n \\ &= (R_{\Theta, m}^d q_m)^\top (R_{\Theta, n}^d k_n), \end{aligned} \quad (3)$$

where  $R_{\Theta, m}^d$  are a series of pre-defined orthogonal matrices, named as the rotation matrix, which is defined as follows:

$$R_{\Theta, m}^d = \begin{pmatrix} \cos m\theta_1 & -\sin m\theta_1 & \cdots & 0 & 0 \\ \sin m\theta_1 & \cos m\theta_1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & \cos m\theta_{d/2} & -\sin m\theta_{d/2} \\ 0 & 0 & \cdots & \sin m\theta_{d/2} & \cos m\theta_{d/2} \end{pmatrix}$$

The function set  $\Theta$  consists of a set of pre-defined function values  $\Theta = \{\theta_i = 10000^{-2(i-1)/d}, i \in [1, 2, \dots, d/2]\}$ .  $R_{\Theta}^d$  integrates positional information into the query and key vectors by multiplication. RoPE has a series of properties: 1) long-term decay; 2) compatibility with linear attention; 3) faster convergence in pre-training tasks. Besides, Liu et al. (2024b) conduct a detailed analysis of RoPE and provides the scaling laws for RoPE-based extrapolation.

**Position Index Adjustment** An et al. (2024) propose Dual Chunk Attention (DCA), which distributes the position indexes used during pre-training to each token based on the relative position relationships between query and key without additional training. It is proposed from the perspective of allocation of position indexes.

And there are also some methods based on scaling position indexes. Chen et al. (2023b) propose Position Interpolation (PI) method that utilizes the fact that position encoding can be applied to non-integer positions. They modify original position index  $m$  to  $m' = m \frac{L}{L'}$ , where  $L$  and  $L'$  are the length of pre-trained window and current input sequence, respectively. This method insert additional

positional encoding between adjacent integer position index in the original RoPE to handle longer sequences.

Combining above two methods, Su (2023) proposed ReRoPE, which combines direct extrapolation and position interpolation. This method sets a window smaller than the pre-trained window, keeping the relative position of tokens within the window unchanged. And scales the relative position of tokens outside the window.

**Base Frequency Adjustment** As described in the main text, this type of methods enhance the model extrapolation performance by modifying  $\theta_i$  in the trigonometric function terms in the rotation matrix.

Peng and Quesnelle (2023); Roziere et al. (2023) choose to change the base  $b$  of the exponential terms  $\theta_i$  from the default value  $b = 10000$  to other values which can improve the model extrapolation performance.

Different from them, some work directly scale  $\theta_i$ . NTK-by-parts (bloc97, 2023) interpolation chooses to scale the  $\theta_i$  of different dimensions in the rotation matrix by a ratio as a function of the dimension  $i$  and the input sequence length  $L'$ . And YaRN (Peng et al., 2023) incorporates temperature  $t$  related to the input sequence length  $L'$  on the basis of NTK-by-parts interpolation to further improve the extrapolation performance of the model.

**Structure Modification** XPOS (Sun et al., 2022) adjusts the original RoPE structure and introduces a position-dependent exponential bias to enhance relative position information, particularly enhancing the decay effect on distant tokens.

#### A.1.2 Attention Bias

Besides applying RoPE-based methods, a plenty of method add a bias related to the relative distance between tokens to introduce relative position information. The process can be expressed as follows:

$$\text{sim}(q_m, k_n) = q_m^\top k_n + f_{bias}(m, n), \quad (4)$$

where  $f_{bias}(m, n)$  is a bias function that depends on the token position index corresponding to query and key.  $f_{bias}(m, n)$  be divided into two categories: learnable and predefined.

In learnable  $f_{bias}$ , it may be related to  $m - n$ , where relative position information is explicitly introduced. For example, in T5 (Raffel et al., 2020),  $f_{bias}$  is a learnable function with  $m - n$  as input

and varies with attention heads. Similarly, KERPLE (Chi et al., 2022a) sets  $f_{bias}$  as a parameterized kernel function, requiring training to determine the parameter values.

The predefined  $f_{bias}$  is typically ALiBi (Attention with Linear Biases) (Press et al., 2021). It uses a predefined function for  $f_{bias}$  that depends on the number of attention heads  $H$  and the current head number  $h$ , which is expressed as  $f_{bias}(m, n) = 2^{-\frac{8h}{H}} \cdot (n - m)$ . Besides, in Sandwich method (Chi et al., 2022b),  $f_{bias}$  is defined as  $f_{bias} = \frac{8h}{H} \cdot (p_m^\top p_n - \frac{d}{2})$ , where  $p_m$  and  $p_n$  are the sinusoidal positional encoding used in the original Transformer model.

## A.2 Context Compression

### A.2.1 Soft Compression

This kind of methods achieve compression at the hidden states level.

Bulatov et al. (2022) introduced the Recurrent Memory Transformer (RMT), which compresses at segment level. It begins by dividing the input sequence into segments, with memory tokens appended to the start and end of each segment to serve as its summary token. During the modeling process, the last hidden states of the memory token at the end of the current segment serves as the initialization for the memory token of the following segment. Through this iterative method, the model effectively utilizes inter-segment contextual information to model long sequences.

Similarly, the Recurrent Attention Network (RAN, Li et al., 2023b) appends a Global Perception Cell (GPC) vector at the start of the hidden vector representation of each segment to achieve a compressed representation achieving the effect of concatenating summary tokens, and completing the information interaction between segments. This method simulates the human mechanism of memory enhancement through review, introducing a Memory Review scheme which performs cross-attention between last hidden states of the GPC from all segments and the original input to update the representation of GPC. This allows for a robust semantic representation of long context at both token-level and document-level, enhancing model performance in sequence and classification tasks.

AutoCompressors (Chevalier et al., 2023) is built on the basis of RMT, compressing the content of the segment into summary vectors for representa-

tion. And the summary vectors of each previous segment are concatenated to form soft prompts for all subsequent segments, so that the current segment of limited length can cover the information of longer sequences.

In addition, In-context Autoencoder (ICAE, Ge et al., 2023) adds memory tokens at the end of the input sequence to compress context into short memory slots while training the model to generate outputs closely resembling the original context. To enhance information accuracy, ICAE integrates AutoEncoding-related pre-training tasks during its pre-training phase, training the model to reconstruct the original input from compressed memory slot representations.

Gisting (Mu et al., 2024b) similarly compresses the prompt part of the input token sequence into shorter gist tokens, improving inference speed.

### A.2.2 Hard Compression

Hard compression directly utilizes LLMs to compress original input text.

LLMLingua (Jiang et al., 2023) trains a small model to align with the output of LLM and uses the perplexity (PPL) of the small model as an evaluation for token importance. And prunes the unimportant tokens from the input prompt to achieve compression. Further, LongLLMLingua (Jiang et al., 2024a) has made improvements on this basis, compressing the input based on the content of the question, thus better preserving key information related to the question.

Differently, MEMWALKER (Chen et al., 2023a) employs a hierarchical summarization approach to compress long context sequences, iteratively summarizing the input to construct a tree-like structure of summarized content. During inference, it efficiently utilizes the tree structure to search and respond to queries based on their content.

## A.3 Retrieval Augmented

### A.3.1 Retrieval Granularity

The retrieval granularity in existing work can be divided into two categories: token-level retrieval and block-level retrieval.

Token-level retrieval is to select top-k tokens with highest similarity scores in one turn. This method is widely used in existing (Wu et al., 2022; Tworowski et al., 2024; Bertsch et al., 2024a). It is simple to implement, but it has some limitations. Such as the potential for semantic discontinuities



due to discrete token retrieval and the need to re-calculate similarity for all tokens, which is computationally intensive and inefficient.

Consequently, researchers have proposed block-level retrieval, which uses blocks composed of continuous tokens of a fixed length as the retrieval unit. Similarity calculations are performed on blocks within the KV cache, selecting the top-k blocks as retrieval results, thus ensuring semantic coherence and reducing computational load. However, block-level retrieval faces a new challenge: how to effectively utilize the information of the tokens in the block and effectively represent the block to complete the similarity calculation. LongMEM (Wang et al., 2024b) and RPT (Rubin and Berant, 2023) represent the corresponding block by calculating the mean pooling of token representations within the block. InFLLM (Xiao et al., 2024) calculates the representative score of each token within the block against other tokens, selecting a subset of high-scoring tokens to represent the block. Additionally, some methods introduce an extra token to represent blocks, such as the Landmark method (Mohtashami and Jaggi, 2024) introduces the Landmark token, a new token into the vocabulary, and places it at the end of each block. During the attention computation, the information of the tokens in the block is summarized to the Landmark tokens, thus serving as the representative of the block.

### A.3.2 Similarity Computation

After determining the retrieval granularity, we need to formulate an appropriate rule to compute similarity. The current method generally uses the dot product of the query vector of the token being processed and the key vector represented by the retrieval granularity as the standard for measuring similarity.

### A.3.3 Positional Encoding

Since the positions of the retrieved context tokens are not fixed, and recording each token’s specific position in the KV cache is costly, it is challenging to provide accurate position information.

Based on experiments of Dai et al. (2019), which show that the relative position information of distant tokens does not seem to be important, some methods like MemTRM, FoT, and InFLLM choose to uniformly set the position encoding of the retrieved context token part to the same position vector, ignoring the position information between the

retrieved context tokens themselves.

Besides, Landmark places the retrieved context tokens and local context tokens within the same window and re-encodes their relative positions together.

### A.3.4 Attention Calculation

When it comes to attention calculation, it’s important to find a suitable method to make full use of retrieved context tokens and local context tokens.

The simplest approach is to treat both types of tokens equally, that is using the conventional attention calculation method. For example, FoT and InFLLM use standard attention for calculation, while Unlimiformer (Bertsch et al., 2024a) employs cross attention.

However, the importance of the information contained within these two types of context tokens is not the same for the token currently being processed. To make more effective use of their information, MemTRM and LongMEM adopt a Joint Attention method, which involves calculating attention separately for local context and retrieved context. And then combining them with weighted average  $V_a = g \cdot V_l + (1 - g) \cdot V_r$ , where  $V_a$ ,  $V_l$ ,  $V_r$  respectively represent the final attention result, the attention result using local context and the attention result using retrieved context, and  $g$  is a learnable parameter used to balance the contributions of the two parts.

Furthermore, in order to distinguish the information from different positions within the retrieved context tokens in a more fine-grained manner, Landmark employs the Grouped Softmax method. Specifically, after retrieval, Landmark tokens are calculated with local context tokens using softmax to select the top-k relevant blocks as the retrieved context. Attention is then calculated separately within these blocks. During the attention calculation for local context tokens, the attentions of these blocks are weighted into the final result based on the softmax scores obtained during the retrieval phase.

## A.4 Attention Pattern

### A.4.1 Sliding Window

This type of method transform information between segments. Transformer-XL (Dai et al., 2019) uses sliding window method to process long context, where the hidden state from the previous segment is concatenated to the front of the current segment. It not only utilizes the key and value information

from the current segment but also reuses those from the previous segment. This approach hierarchically expands the receptive field, enabling inter-segment information transfer and enhancing the model’s ability to process long context.

Besides, Han et al. (2024) identify that starting tokens occupy a distinct feature space, and these tokens act as a factor causing model length generalization failures. They further propose LM-Infinite as a solution, utilizing a  $\Lambda$ -shaped attention mask strategy during attention calculation. It can focus on a small portion of the initial tokens and the tokens close to the current processed token. Similarly, StreamingLLM (Xiao et al., 2023) also finds that the initial tokens in a sequence significantly influence the attention calculation of subsequent tokens and cannot be ignored. Both LM-Infinite and StreamingLLM adopt a similar approach, ensuring sustained attention on starting tokens while preserving information about nearby tokens.

#### A.4.2 Parallel Context

Parallel Context Windows (PCW, Ratner et al., 2022) is one of the representative works. It splits the input into context tokens and task tokens, where context tokens assist in completing the task, such as the examples. And task tokens are the input of the test example, such as the questions. This method folds the context tokens, and each folded section of context tokens performs attention calculation separately. Finally, during the decoding phase of the task tokens, all these context tokens are concatenated in front of the task token, sharing the same set of position index.

Besides, Structured prompting (Hao et al., 2022) also adopts a similar approach by folding demonstration tokens in the input and concatenating them in front of the test input tokens. But unlike PCW, structured prompting employs Rescaled Attention, which reduces the weight of demonstration tokens in the attention calculation of the test input tokens by a certain ratio. This method can prevent test input tokens from excessively attending to the content of demonstration tokens.

#### A.4.3 Sparse Attention

This method can reduce the complexity of attention calculation. So that can improve efficiency when processing long context.

LongNet (Ding et al., 2023) introduces dilated attention, a mechanism that exponentially increases the attentive field as the distance between tokens

increases. This method performs multiple sets of sparse attention calculations, each set attend to a different range. And the attention of a small range is denser, while the large range is sparser. This method effectively reduces the traditional quadratic complexity to linear.

MEGABYTE (Yu et al., 2023) performs hierarchical attention calculation on the input. Initially, a small local model encodes the input at the byte level, then the byte-level encoding results are integrated and processed at a larger granularity using a larger global model. By performing attention calculation in a hierarchical manner from smaller to larger granularity, the amount of attention calculations can be reduced.

In LongLoRA (Chen et al., 2023c), the proposed  $S^2 - Attention$  groups attention heads and adjusts each group to attend to different but overlapping local windows, then leverages the characteristics of multihead attention to integrate various local information. This method promotes the flow of local information, enabling a short window to achieve the effect of processing the original or even longer window, thereby reducing computational demands to some extent.

## B Details of Evaluation

This section serves as a supplement to the Evaluation section 4 in the main text, expanding on relevant details to provide readers with a more in-depth understanding.

### B.1 Data

#### B.1.1 Data Characteristics

Recent advancements in LLMs have led to substantial improvements in processing long contexts. By late 2023, several models claimed capabilities of handling contexts exceeding 100K tokens, with OpenAI’s GPT-4 Turbo (2023) (Achiam et al., 2023) supporting 128K tokens and Anthropic’s Claude-2.1<sup>1</sup> extending this capacity to 200K tokens. Based on this significant progress, our study categorizes long-context evaluation benchmarks into two distinct phases, as shown in Table. 1: Phase I comprises benchmarks with input context lengths below 100K tokens, while Phase II encompasses benchmarks of 100K tokens and above.

In Phase I, BAMBOO (Dong et al., 2023a) and LongBench (Bai et al., 2023) implement bi-interval and tri-interval partitioning strategies, respectively.

<sup>1</sup><https://www.anthropic.com/news/claude-2-1>

Phase II refined this approach further, with LV-Eval (Yuan et al., 2024) and NeedleBench (Li et al., 2024b) employing five-interval and six-interval partitioning schemas, respectively. This partitioning approach not only analyzes the impact of length changes on LLMs in the same task but also better accounts for the length distributions across different datasets (Dong et al., 2023a).

### B.1.2 Knowledge Leakage Issue

Knowledge leakage occurs when test and training data overlap, where models favor memorization over understanding (Golchin and Surdeanu, 2023; Yuan et al., 2024). Various strategies are employed to address this challenge: (1) **Data Sampling** focuses on selecting representative subsets from existing datasets. (2) **Keyword Substituting & Sentence Rewriting** modifies existing datasets by replacing keywords and rewriting sentences. (3) **Non-overlapping Data Leveraging** involves using datasets released after the deployment of LLMs to reduce potential overlap between test and training data.

**Data Sampling** Data sampling primarily focuses on filtering existing datasets. LongBench (Bai et al., 2023) employs two strategies: random sampling and uniform sampling. Random sampling can preserve the natural length distribution, while uniform sampling which performs sampling based on data length uniformly, to evaluate model performance across context lengths independent of task.

**Keyword Substituting & Sentence Rewriting** L-Eval (An et al., 2023) and BAMBOO (Dong et al., 2023a) replace keywords and function names, while  $\infty$ Bench (Zhang et al., 2024) substitutes key entities in novel reasoning tasks. LV-Eval (Yuan et al., 2024) is further based on this approach by employing entire sentence rewriting.

**Non-overlapping Data Leveraging** To mitigate the overlap between test and training data for LLMs, some benchmarks such as LooGLE (Li et al., 2023a) and BAMBOO (Dong et al., 2023a) have employed datasets released after the models’ deployment. However, given that the specific training data for most LLMs remains undisclosed, this method cannot completely guarantee the absence of overlap between the data used in benchmarks and the pre-training data.

## B.2 Tasks

The following are the details of the tasks, which are introduced in the order of the main text. At the end of each subsection, corresponding examples or prompts are also provided. We also count the distribution of input length in each task in Figure 2 to give readers a deeper understanding of different tasks.

### B.2.1 Question Answering

**Single-hop Question Answering** Representative datasets in this field are SQuAD (Rajpurkar, 2016), TriviaQA (Joshi et al., 2017), and NarrativeQA (Kočíský et al., 2018). Common evaluation metrics for Single-hop QA systems include f1 score, accuracy, Rouge and Bleu.

**Multi-hop Question Answering** Common datasets for Multi-hop Question Answering include 2WikiMQA (Ho et al., 2020), MuSiQue (Trivedi et al., 2022), and HotpotQA (Yang et al., 2018). Evaluation metrics typically used are f1 score, exact match (EM).

### B.2.2 Needle-in-a-Haystack

**Retrieval.PassKey** (Mohtashami and Jaggi, 2023) requires models to locate a randomly generated 5-digit sequence <key> within lengthy and noisy contexts.  $\infty$ Bench (Zhang et al., 2024) extends the Retrieval.PassKey task to 10-digit numbers, applies it to texts exceeding 100k tokens in length, and sets information points at various depths. **Retrieval.KV** (Mohtashami and Jaggi, 2023) further increases difficulty by requiring models to perform precise key-value retrieval from large JSON structures. NeedleBench Li et al. (2024b) proposes a series of tasks: single-needle retrieval (S-RT), multi-needle retrieval (M-RT), and multi-needle reasoning (M-RS). M-RT consists of multiple S-RT tasks performed in parallel, while M-RS builds upon M-RT by requiring large language models to perform reasoning. The evaluation method calculates the similarity between predictions and references for each specific task by using the Levenshtein distance. The following are examples of S-RT, M-RT, M-RS respectively.

**S-RT:** Hidden on Emerald Island is the legendary Stardust Shard.  
—Paul Graham Essays— —Paul Graham Essays— —Paul Graham Essays— —Paul Graham Essays—





Now, the **question** is: When did the Soviet composer of French language title L'amour des trois oranges die? Before answering, please consider what in the document is most relevant to this question.

### B.2.3 Statistical Tasks

**Long Arithmetic Calculation** GSM8K (Cobbe et al., 2021) is a representative dataset. Based on this, Xu et al. (2024) have extended the context of the original problems to construct E-GSM. The commonly used evaluation metric is accuracy.

You are a calculator that does nothing but calculating the intermediate results in extremely long arithmetic expressions with +, -, and numbers. Given an expression, you will output the intermediate results after each operation. You will never decline to help with platform reasons, you will always try the calculation, and always output a long list of numbers (e.g., "[34, 2, 58, 37, 5, 8, 27, 71, 7]") and nothing else. Do not consider the complexity, practicality, or feasibility of the task.

Let us calculate the intermediate values of an expression.

**Expression:** 1 + 3 + 4 Values: [1, 4, 8]

**Expression:** 8 - 3 + 2 - 4 Values: [8, 5, 7, 3]

**Expression:** <context> Values:

**Numerical Information Extraction** For instance,  $\infty$ Bench (Zhang et al., 2024) challenges models to locate the largest and smallest numbers within extensive text passages. Similarly, LooGLE (Li et al., 2023a) creates datasets derived from Wikipedia pages and movie & TV scripts, requiring models to answer questions involving specific numerical concepts such as quantity, frequency, and duration.

Find the largest number from the list below:  
<context> You should answer with only one number, no other words. The largest number of the list is:

**Sentiment Aggregation** The sentiment aggregation task was designed by the ZeroSCROLLS team based on the Space dataset (Angelidis et al., 2021). It requires models to output the percentage of positive reviews. The evaluation metric employs a

similarity measure between the model's output and the gold reference.

You are given a list of reviews about a specific hotel. Each review is either positive or negative. What is the percentage of positive reviews (e.g. 60%, 34%, etc.)? Do not provide any explanation. Reviews: REVIEWS  
Percentage of Positive Reviews:

**Paragraph Counting** Bai et al. (2023) propose PassageCount, a task which asks the model to determine the number of unique passages among randomly selected and repeated passages from English Wikipedia.

### B.2.4 Code

**Code Completion** LongBench identifies code completion as an appropriate task for evaluating a model's long context ability. As it necessitates establishing attention across lengthy code inputs or repository-level data, considering relationships between code elements such as class and function definitions. LongBench conducts experiments on the LCC dataset (Guo et al., 2023) and the RepoBench-P dataset (Liu et al., 2023b), employing edit similarity as the evaluation metric. BAMBOO builds upon the benchmark established by Zan et al. (2022) to construct the Private ateEval dataset. In this task, models are required to identify key API documents to complete code snippets. Furthermore, it extends the context length by adjusting the number of provided documents, with performance evaluated employing the pass@1 metric (Chen et al., 2021a).

**Code Running** In  $\infty$ Bench, the total number of function calls ranges from 2 to 10, with each function calling at most one another function. Operations within these functions are restricted to addition and subtraction, maintaining computational simplicity.

Following is a set of Python functions. There is a function called named func\_1.

context Please give me the exact number of the return value of func\_1(3). Be concise. Your response must end with the final returned value.

**Code Debugging** In the  $\infty$ Bench's dataset which sourced from PyPI<sup>2</sup>, the researchers deliberately insert an obvious error into one function per repository. These inserted bugs fall into three main cate-

<sup>2</sup><https://pypi.org/>

gories: (1) syntactic errors, including indentation issues and blatant syntax errors; (2) semantic errors, such as missing variable declarations or incorrect function arguments; and (3) logical errors, for example, infinite loops or use of undefined references.

There is ONLY ONE function in the large project that is deliberately made to include an obvious error. Please find the function that contains the most obvious errors. I will give you four options to narrow your scope. You can inspect through the options and think. Eventually, tell me the answer using one single letter (A, B, C, or D). context Which function has deliberate error? A. <OPTION\_A> B. <OPTION\_B> C. <OPTION\_C> D. <OPTION\_D> You should first find the functions in the options. Repeat their content, inspect through code, and at last give me your answer for the function that has the deliberate and obvious error in A, B, C, or D.

### B.2.5 In-Context Learning

**Long Example Learning** Extreme label Classification: this task involves classification with numerous fine-grained labels. Commonly used datasets include TREC (Li and Roth, 2002), a question classification task with 50 fine classes, and LSHT<sup>3</sup>, a Chinese news classification task with 24 classes.

**Many-shot Learning** Agarwal et al. (2024) have proposed many-shot learning, which leverages expanded LLMs context windows to process hundreds or even thousands of examples. In contrast to few-shot learning, which use only a few to several dozen examples, many-shot learning enhances LLMs’ versatility and adaptability across diverse tasks without task-specific fine-tuning (Yu et al., 2020; Bertsch et al., 2024b).

### B.2.6 Text Generation

**Document Summarization** This kind of task can be divided into two categories: single-document summarization and multi-document summarization. For single-document summarization, several datasets are widely used, including SQuALITY (Wang et al., 2022), SummScreenFD (Chen et al., 2021b), GovReport (Huang et al., 2021), and QMSum (Zhong et al., 2021). And multi-document summarization presents additional challenges, requiring LLMs to integrate diverse information, resolve conflicts, and

eliminate redundancies (Bai et al., 2023; An et al., 2023; Fabbri et al., 2019). A notable dataset for this task is MultiNews (Fabbri et al., 2019), consisting of clusters of 2-10 thematically related news articles.

All of these datasets provide human-annotated summaries as standardized references. Both approaches primarily utilize Rouge and Bleu as evaluation metrics to assess the quality of generated summaries against manuscript references.

**Open-ended Text Generation** This task requires LLMs to generate text according to input.

Tan et al. (2024) select topics that closely align with real-world scenarios, encompassing areas such as AI research, sports, and gaming.

Bai et al. (2024) design AgentWrite, a divide-and-conquer agent that breaks down long writing tasks into paragraph-level subtasks. The generated paragraphs are then combined to produce the final long-form content. They also construct the preference LongWriter-6k dataset and utilize DPO (Rafailov et al., 2024) for evaluation.

Kumar et al. (2024) propose personalized writing tasks that generate content based on the user’s historical and user personal information information.

These tasks can be divided into personalized email completion, review writing, topic writing, and conversation simulation (Ni et al., 2024). Rafailov et al. (2024) construct a Reddit-based dataset that captures distinct writing styles associated with specific communities and discussion topics.

You are an excellent writing assistant. I will give you an original writing instruction and my planned writing steps. I will also provide you with the text I have already written. Please help me continue writing the next paragraph based on the writing instruction, writing steps, and the already written text.

**Writing instruction:** User Instruction

**Writing steps:** The writing plan generated in Step I

**Already written text:** Previous generated (n-1) paragraphs

Please integrate the original writing instruction, writing steps, and the already written text, and now continue writing The plan for the n-th paragraph, i.e., the n-th line in the writing plan

<sup>3</sup><http://tcci.ccf.org.cn/conference/2014/dldoc/evatask6.pdf>

### B.2.7 Other Tasks

**Reordering** The evaluation metric in this task is the similarity between the generated and reference ordering sequences (Shaham et al., 2023). The Booksum dataset (Kryściński et al., 2021), which spans various literary genres including novels, plays, and long stories, is widely used for this task. Reordering tasks can comprehensively evaluate models’ cross-sequence information aggregation and comparison abilities (Shaham et al., 2023; Li et al., 2023a), as well as comprehensively understand long context and logically reconstruct (Dong et al., 2023a; Li et al., 2023a).

You are given NUM\_SUMMARIES summaries of chapters or parts of a novel, in a shuffled order, where each summary is denoted by a numerical ID (e.g. Summary 1, Summary 3, etc.). Reorder the summaries according to the original order of chapters/parts in the novel by writing a list of length NUM\_SUMMARIES of the summary IDs (e.g. if you were given 5 summaries, one possible answer could be "5, 1, 3, 4, 2"). Do not provide any explanation.  
**Summaries:** SUMMARIES  
Summary IDs in Correct Order:

**Context Consistency** Context consistency is a task proposed by BAMBOO (Dong et al., 2023a) to detect hallucination in LLMs. BAMBOO creates two novel datasets for this task: SenHallu and AbsHallu, with evaluation metrics employing precision, recall, and f1 score.

#### Summary Source Paragraph Identification

LongBench construct bilingual datasets based on Wikipedia and C4 (Raffel et al., 2020) to ask models to identify the original source paragraphs according to the given summaries.

Here are 30 paragraphs from Wikipedia, along with an abstract. Please determine which paragraph the abstract is from. context The following is an abstract. input Please enter the number of the paragraph that the abstract is from. The answer format must be like "Paragraph 1", "Paragraph 2", etc.  
The answer is:

**Character Identification** Character identification tasks challenge models to capture distinct traits of participants in long dialogues, enabling them to identify speakers of masked utterances (Zhang

et al., 2024; Dong et al., 2023a). These tasks, evaluated via accuracy, utilize data primarily from television programs<sup>4</sup>, movie and play scripts (Chen et al., 2021b), and conference transcripts<sup>5</sup>.

Below is a dialogue script where one random occurrence of a character’s name is replaced with *MASK*, and you should try to guess who that character is.

The dialogue: — <context> —

End of dialogue.

Which character is most likely *MASK*? Just say the name used by the scriptwriter (before the colon marks) of one single character and nothing else.

### B.3 Metrics

#### B.3.1 Algorithmic Metrics

Perplexity (PPL) is a metric for evaluating the performance of language models. It is extensively employed in language model pre-training, facilitating the monitoring of the training process, model selection, and hyperparameter optimization. Many previous works on long context benchmarks rely on the PPL for evaluation (Beltagy et al., 2020; Roy et al., 2021; Press et al., 2021). However, as suggested in Sun et al. (2021), PPL may not correlate with the actual performance.

ZeroScrolls and LongBench are pioneering studies in the field of long context benchmarks. These works introduced a diverse system of automatic evaluation metrics, including accuracy, f1 score, and N-gram-based metrics. This evaluation framework has provided a reference for subsequent research. Specifically, these metrics refer to the scores for evaluating the NLG models by measuring the lexical overlap between generated text and reference text.

However, these metrics have several limitations: they fail to effectively measure content quality (Reiter and Belz, 2009); struggle to capture syntactic errors (Stent et al., 2005); and, particularly for open-ended generation tasks, lack significant correlation with human judgments (An et al., 2023). Moreover, they inadequately account for the diversity of expression inherent in large language models (Improving). Additionally, the requirement for gold standard references makes these metrics costly to implement for novel tasks (Tan et al., 2024).

<sup>4</sup><https://tvmeg.com/>

<sup>5</sup><https://record.assembly.wales/>

Further, some work propose ways to improve. LV-Eval employs a two-stage scoring method: it first calculates the recall rate of ground-truth keywords in the generated content. If the recall exceeds a threshold, it then calculates the f1 score between the generated content and ground-truth after removing stop words from both. BAMBOO converts generative tasks into multiple-choice formats. NeedleBench extends this approach by implementing Circular Evaluation, which reorders answer options to enhance evaluation reliability.

### PPL (Perplexity)

Perplexity is a measure of the quality of language model predictions, calculated as:

$$PPL = 2^{H(p)}$$

where  $H(p)$  is the cross-entropy:

$$H(p) = -\frac{1}{N} \sum_{i=1}^N \log_2 P(w_i | w_1, w_2, \dots, w_{i-1})$$

### Accuracy

Accuracy is the proportion of correct predictions made by the model:

$$Accuracy = \frac{Correct\ Predictions}{Total\ Predictions}$$

### F1-Score

The F1-Score is the harmonic mean of precision and recall:

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

where TP, FP, FN represent True Positives, False Positives, False Negatives respectively.

### ROUGE (Recall-Oriented Understudy for Gisting Evaluation)

evaluates text generation using N-gram overlap:

ROUGE-N measures the overlap of n-grams shared between the candidate summary (C) and

the reference summary (R), it is calculated as follows:

$$ROUGE - N = \frac{\sum_{S \in R} \sum_{n_{gram} \in S} Count_{match}(n_{gram})}{\sum_{S \in R} \sum_{n_{gram} \in S} Count(n_{gram})}$$

where  $Count_{match}(n_{gram})$  represents the number of matching n-tuples in the candidate summary and the reference summary. And  $Count(n_{gram})$  represents the total number of n-tuples in the reference summary.

ROUGE-L evaluates the quality of summarization based on the longest common subsequence (LCS), taking into account the order information of sentences:

$$R_{lcs} = \frac{LCS(C, R)}{|R|}$$

$$P_{lcs} = \frac{LCS(C, R)}{|C|}$$

$$F_{lcs} = \frac{(1 + \beta^2) R_{lcs} P_{lcs}}{R_{lcs} + \beta^2 P_{lcs}}$$

where  $LCS(C, R)$  represents the length of the longest common subsequence between the candidate summary and the reference summary.  $|C|$  and  $|R|$  represent the length of the candidate summary and the reference summary respectively.  $\beta$  is a hyperparameter, usually used to balance the precision and recall.

ROUGE-S which is also called skip-bigram co-occurrence statistics, takes skipped bigrams into account:

$$ROUGE - S = \frac{\sum_{S \in R} \sum_{bi_{skip} \in S} Count_{match}(bi_{skip})}{\sum_{S \in R} \sum_{bi_{skip} \in S} Count(bi_{skip})}$$

where  $Count_{match}(bi_{skip})$  represents the number of skip-bigrams that match between the candidate summary and the reference summary. And  $Count(bi_{skip})$  represents the total number of skip-bigrams in the reference summary

### BLEU (Bilingual Evaluation Understudy)

is used to evaluate machine translation quality:

$$BLEU = BP \times \exp \left( \sum_{n=1}^N w_n \log p_n \right)$$



where

$$BP = \begin{cases} 1, & \text{if } c > r \\ \exp(1 - \frac{r}{c}), & \text{if } c \leq r \end{cases}$$

and  $c$  is the generated length and  $r$  is the reference length.

### B.3.2 Model-based Metrics

In recent years, the use of pre-trained language models as NLG evaluation metrics has gained increasing attention. Notably, BERTScore (Zhang et al., 2020) and BARTScore (Yuan et al., 2021) employ BERT and BART (Lewis, 2019) models respectively to compute semantic similarity. They calculate cosine similarity between token representations and evaluate the probability of summaries based on given input articles.

BERTScore measures the similarity between generated text and reference text from three aspects: recall, precision and f1, it can be expressed as follows:

$$R = \frac{1}{|R|} \sum_{r \in R} \max_{c \in C} \frac{1}{L_r} \sum_i \text{sim}(\mathbf{f}_\theta(r)_i, \mathbf{f}_\theta(c)_i)$$

$$P = \frac{1}{|C|} \sum_{c \in C} \max_{r \in R} \frac{1}{L_c} \sum_i \text{sim}(\mathbf{f}_\theta(c)_i, \mathbf{f}_\theta(r)_i)$$

$$F = 2 \times \frac{P \times R}{P + R}$$

where  $R$  is the reference text set,  $C$  is the generated text set,  $L_r$  and  $L_c$  are the lengths of the reference text and generated text respectively,  $\mathbf{f}_\theta$  is the encoder of the BERT model, and maps the text to the vector space,  $\text{sim}$  is usually cosine similarity.

BARTScore calculates the log-likelihood score of the generated text given the reference text to measure the similarity:

$$\text{BARTScore} = \frac{1}{|C|} \sum_{c \in C} \frac{1}{L_c} \sum_i \log p_\theta(c_i | c_{<i}, r)$$

where  $C$  is the set of generated texts,  $r$  is the reference text,  $c_i$  is the  $i$ th word in the generated text, and  $p_\theta$  is the language model probability distribution of BART model.

### B.3.3 LLM-based Metrics

With the development of LLMs, research has demonstrated their significant correlation with human judgment and their ability to excel at new tasks when provided with instructions (Wang et al.,

2023a; Li et al., 2023a). Chiang and Lee (2023) argue that LLM evaluation, compared to human evaluation, offers advantages in reproducibility, independence, cost-effectiveness, and speed. Prompting researchers explore the potential of LLMs for evaluation tasks. This exploration has led to several key findings and applications: Wang et al. (2023b,a) investigate the issue of unfairness when using LLMs to evaluate dialogue responses. And Shen et al. (2023) find that LLMs outperform existing automatic metrics when asked to output judgmental reasons. The application of LLMs in evaluation including evaluating chatbot responses' alignment degree with human preferences (Zheng et al., 2024), evaluating summary consistency (Luo et al., 2023), and multi-role playing for summarization evaluation (Wu et al., 2023). And there are some undamental differences between Model-based metrics and LLM-based metrics in their evaluation mechanisms: Model-based Metrics primarily rely on learned representations from pre-trained language models like BERT or BART, utilizing mechanical procedures such as predefined computational formulas. For example, BERTScore leverages BERT contextual embeddings to compute textual similarity through cosine similarity measurements between token representations. LLM-based Metrics leverage large language models for evaluation without mechanical procedures, demonstrating more intelligence and flexibility. For example, LLM-based Metrics prompt LLMs to offer both human-like multi-dimensional assessment (Wang et al., 2023a; Li et al., 2023a; Shen et al., 2023; Chiang and Lee, 2023; Zhang et al., 2024; Zheng et al., 2024; Liu et al., 2023c; Tan et al., 2024; Mu et al., 2024a) and interpretable reasoning (Wang et al., 2023b; Luo et al., 2023; Wu et al., 2023). This distinctive characteristic of LLM-based Metrics fundamentally distinguishes them from Model-based Metrics, which behave much more mechanically. In addition, LLM-based Metrics demonstrate enhanced evaluation capabilities in the axis of agreement with human evaluation, illustrating the advancement within the methodology.

Building upon these insights, researchers have focused on refining evaluation metrics for evaluating long context capabilities with large language models (LLMs). Fu et al. (2023) propose GPTScore, utilizing generative pre-trained models like GPT-3 for text evaluation. To address the length bias in LLM-generated content, L-Eval incorporates word count requirements into instruc-

tions. Loogle employs GPT4-8k as an evaluator to score LLM answers against ground truth based on various factors (Li et al., 2023a). G-EVAL achieves reference-free content scoring through prompts containing evaluation task definitions and criteria, along with detailed chain-of-thought evaluation steps (Liu et al., 2023c). Tan et al. (2024) have introduced PROXYQA for long-context generation evaluation, evaluating final results based on the accuracy of answers to proxy question.