Vision-centric Token Compression in Large Language Model

Ling Xing[1*], **Alex Jinpeng Wang**[2*], **Rui Yan**[1†], **Xiangbo Shu**[1], **Jinhui Tang**[3]
[1]Nanjing University of Science and Technology [2]Central South University
[3]Nanjing Forestry University

Abstract

Real-world applications are stretching context windows to hundreds of thousand of tokens while Large Language Models (LLMs) swell from billions to trillions of parameters. This dual expansion send compute and memory costs skyrocketing, making *token compression* indispensable. We introduce VISION CENTRIC TOKEN COMPRESSION (VIST), a *slow–fast* compression framework that mirrors human reading: the *fast* path renders distant tokens into images, letting a **frozen, lightweight vision encoder** skim the low-salience context; the *slow* path feeds the proximal window into the LLM for fine-grained reasoning. A Probability-informed Visual Enhancement (PVE) objective masks high-frequency tokens during training, steering the Resampler to concentrate on semantically rich regions—just as skilled reader gloss over function words. On eleven in-context learning benchmarks, VIST achieves the same accuracy with 2.3× fewer tokens, cutting FLOPs by 16% and memory by 50%. This method delivers remarkable results, outperforming the strongest text encoder-based compression method CEPE by **7.6**% on average over benchmarks like TriviaQA, NQ, PopQA, NLUI, and CLIN, setting a new standard for token efficiency in LLMs. The project is at https://github.com/CSU-JPG/VIST.

1 Introduction

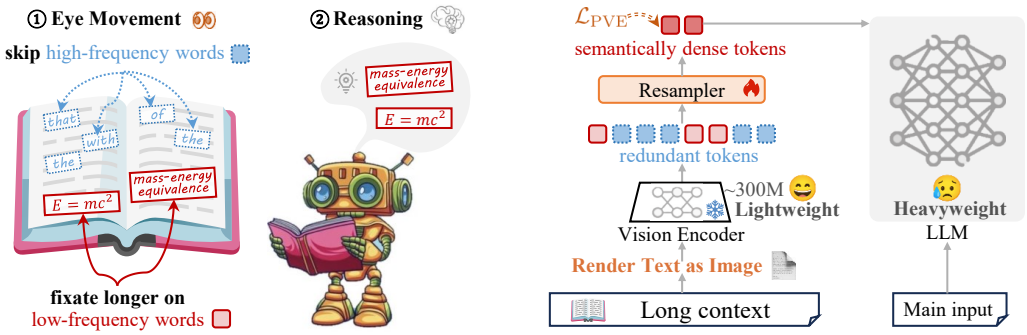
Large language models (LLMs) excel at short snippets, yet many real-world tasks, *e.g.*, long-document understanding [1, 2] and question answering [3, 4]—already require inputs far beyond the thousand-token regimes of early GPT-3 [1]. At the same time, parameter counts have leapt from billions to trillions [5, 6, 7]. In this dual squeeze of *longer context & larger models*, **compression shifts from a convenience to a necessity**: without shrinking the input, even the most powerful LLM cannot afford to reason over the information we want it to see.

Psycholinguistics shows that our eyes dance across text: we *fixate* on rare, content-rich words and *skip* almost one-third of high-frequency function words [8, 9, 10]. This *selective-reading* strategy forms a natural *slow-fast* circuit. A *fast visual pass* skims distant, low-salience context to maintain global context, while a *slow cognitive pass* focuses on nearby sentences that matter. (Figure 1 (a)).

Motivated by this circuit, we present VISION CENTRIC TOKEN COMPRESSION (**VIST**), a *slow-fast* token compression framework that mirrors human skimming. As illustrated in Figure 1 (b), VIST first converts loosely relevant long context into images, which are processed by a frozen vision encoder and a trainable Resampler to produce semantically compact visual tokens. These compressed tokens and the main input tokens are then consumed by the LLM. In this *slow-fast* setup, the *vision encoder*

* Equal contribution
† Corresponding author

39th Conference on Neural Information Processing Systems (NeurIPS 2025).



(a) Selective reading strategy based on word frequency in skilled readers.

(b) A visual pathway to unlock LLM context.

Figure 1: Our method **VIST** adopts a lightweight vision encoder to process loosely relevant long contexts, offering a **more cost-efficient alternative** to full LLM processing. However, the inherent redundancy in long text leads to redundant visual tokens. Motivated by **Selective Reading Strategy** where *low-frequency (content) words receive longer fixations while high-frequency function words are often skipped*, we design Probability-informed Visual Enhancement (*i.e.*, $\mathcal{L}_{\text{PVE}}$). This guides the Resampler to prioritize informative content over redundancy, **resulting in a 75% reduction in the number of visual tokens** and yielding semantically dense tokens.

acts like the human eye—selectively attending to salient information—while *the LLM functions as the brain*, concentrating on the most informative content for deeper reasoning.

Specifically, the frozen visual encoders (*e.g.*, CLIP [11]) trained on paired image-text data naturally *acquire OCR capabilities* [11, 12], making them a powerful tool for image-based text understanding. However, the inherent redundancy in long text leads to redundant visual tokens. To address the problem, we design **Probability-informed Visual Enhancement** (PVE), a contrastive scheme that enforces Resampler to prioritize informative content over redundancy. Concretely, PVE applies **frequency-based masking strategy** to text token embeddings from the LLM tokenizer, suppressing high-frequency (less informative) text tokens. This semantically rich text supervision guides the Resampler to focus on informative content, bridging the semantic gap between visual and text tokens, and enabling more effective token compression. Unlike previous work [13, 14, 15, 16, 17] that rely on LLMs to compute token-level information entropy for assessing importance, VIST adopts token frequency as a simple yet effective proxy, and further **reveals rare tokens are key contributors to overall semantic meaning** (*cf.* Figure 3 and §5).

VIST leverages a lightweight vision encoder to compress loosely relevant long contexts, offering a cost-efficient alternative to full-scale LLM computation. Furthermore, the vision encoder serves as a *visual text tokenizer*, offering several compelling advantages over traditional text tokenizers. ❶ **Simplified Tokenization.** Text tokenizers rely on complex tokenization rules and vocabulary constraints, typically involving nearly ten human-defined preprocessing steps (*e.g.*, lowercasing, punctuation and stop word removal, and tokenization) [18]. However, the vision encoder processes text more directly by treating rendered text images as visual inputs. ❷ **Vocabulary Bottleneck Mitigation.** Text tokenization, constrained by a finite vocabulary, becomes a bottleneck when scaling to many languages. A larger vocabulary increases memory and computational costs in the embedding matrix and output layer. However, vision encoder eliminates the need for text tokenizers and unifies various languages into a single image format that removes the need for a vocabulary [19, 20]. ❸ **Robustness to Character-Level Noise.** Vision encoders are more resilient to typos and low-level orthographic attacks, as they capture holistic visual patterns rather than relying on discrete token matching [20]. ❹ **Multilingual Efficiency.** While our work focuses on English, visual text tokenizer can reduce the number of tokens compared to traditional text tokenizer for languages (*e.g.*, 62% for Japanese, 78% for Korean, and 27% for Chinese). This reduction is particularly impactful in long-text scenarios. Taken together, *leveraging vision encoders for long-context compression is a promising and worthwhile direction to explore*.

To validate the effectiveness of VIST, we primarily compare with the text-encoder-based token compression counterpart CEPE [21]. VIST requires **2.3× fewer visual tokens** than text tokens for the same input, reducing **FLOPs by 16%** and **memory usage by 50%**. VIST also delivers consistent

gains over CEPE on both In-Context Learning and Open-domain Question Answering tasks, with **average gains of 3.6% across 11 datasets** and **5.7% across 3 datasets**, respectively—highlighting the effectiveness of visual representations for long-context modeling in LLMs.

2 Related Work

Token Compression. There has been a growing interest in expanding the context window for LLMs. A line of methods leverages LLM itself to compress raw long input. One may classify these works into two principal groups. **i)** *soft prompt-based* methods that adapt LLMs to compress context into fewer tokens [22, 23, 24, 25, 26]. **ii)** *selection-based* methods that remove redundant tokens based on information entropy computed by LLMs [13, 14, 15, 16, 17, 27]. All the long inputs typically need to be handled by the heavy LLMs, which incur high costs. Another line of work [28, 29] augments LLMs with the capacity to memorize previous long context information by external memory bank and retrieve relevant knowledge [30, 31, 32, 33]. Our method is orthogonal to these existing strategies and can be combined with them to achieve longer context length. The most related work is CEPE [21], which employs a lightweight text encoder to handle long contexts and integrates the information into LLM via cross-attention. While CEPE reduces workload on the LLM, it overlooks the redundancy in long text, making it harder for LLMs to effectively allocate attention to key content. In contrast, VIST compresses long text into compact visual tokens guided by high-density semantic text tokens.

Vision-centric Method. Text tokenization [34, 35, 36] breaks down text into tokens, serving as a fundamental step in natural language processing. However, tokenization-based methods lack robustness against spelling errors and face vocabulary bottlenecks. A new line of work tackles these issues in a tokenizer-free paradigm [37, 20, 38]. The representative method Pixel [20] renders text as images and learns to reconstruct masked image patches at the pixel level. It demonstrates strong cross-language translation capabilities and tolerance for text perturbation. Along this direction, recent work explores different pre-training objectives [39, 40], *e.g.*, contrastive learning [41], patch-and-text prediction [38]. Despite advancements, these methods overlook long-text scenarios and rely on complicated training pipelines, *e.g.*, OCR-based text understanding [19]. In contrast, VIST directly processes text images by leveraging a vision encoder pretrained on image-text pairs with strong OCR capabilities, and enhancing visual features using enriched text embeddings from LLM tokenizer. An emerging family of multimodal methods [42, 43, 44, 45, 46] leverage visual representations to process text and images together, enabling a wide range of applications involving visually-situated text, *e.g.*, webpage parsing [43], tables images analysis [47], and document understanding [42, 48]. In this work, we explore incorporating long-context information into LLMs from a visual perspective.

3 Methodology

In this section, we present our method VIST, which processes long in-context text by a lightweight visual encoder, effectively and efficiently extending the context length of LLMs.

3.1 Overall Pipeline

Our VIST, a *slow-fast* compression framework, is designed to efficiently process long texts by mimicking human reading. The *fast visual path* skims distant, low-salience long context via a lightweight vision encoder, while the *slow cognitive path* performs fine-grained reasoning on important content by LLM. As illustrated in Figure 2, the input long text (*i.e.*, T text tokens) is split into two parts: the first T_{e} text tokens processed in a visual view and the remaining T_{d} raw text tokens given to LLM, where $T = T_{\text{e}} + T_{\text{d}}$. Specifically, the T_{e} text tokens are evenly rendered into M images and fed into a frozen vision encoder. Then VIST employs a learnable Perceiver Resampler to compress text-rendered image features into a fixed count of tokens. Such compressed visual tokens are integrated into the LLM via cross-attention for the next-token prediction. The Perceiver Resampler is jointly trained with the LLM during tuning the cross-attentions in an end-to-end manner.

To empower the model with the ability to comprehend dense text in images, we devise **Probability-informed Visual Enhancement** (PVE, §3.4). PVE maximizes agreement between visual features obtained from the Perceiver Resampler and **text token embeddings extracted from LLM tokenizer**. This alignment bridges the global semantic gap between visual tokens and raw text tokens. Furthermore, to address token redundancy, VIST incorporates a **frequency-based masking** mechanism

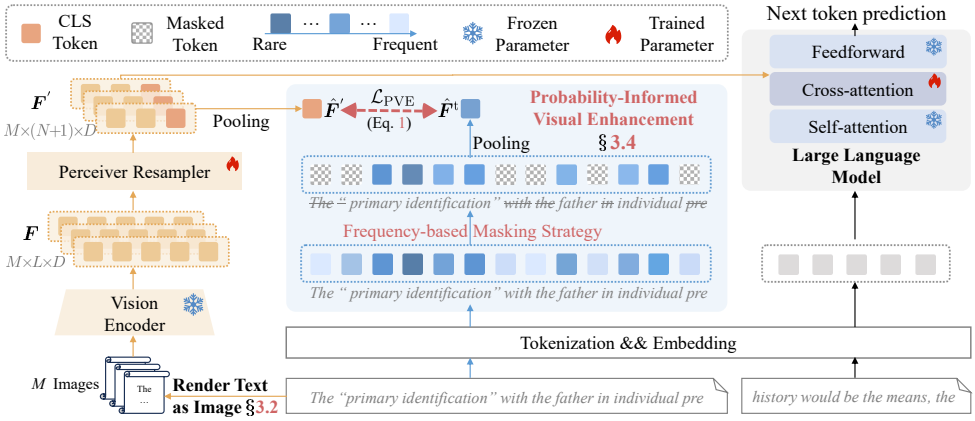


Figure 2: **Overview of VIST**. VIST, a *slow-fast* token compression framework, efficiently processes long texts by mimicking human skimming. First, the *fast visual path* converts long context into images and employs a lightweight vision encoder to capture semantically compact visual features. These features are then integrated into the LLM via cross-attention in the *slow cognitive path*, allowing LLM to focus on salient content for deeper reasoning. To prioritize informative content in text images, VIST employs **Frequency-based Masking** on text token embeddings from text tokenizer, suppressing high-frequency but low-information token (*e.g.*, "the" and "with"). Such refined embeddings guide the Resampler in extracting critical semantics from the images.

within PVE that selectively masks high-frequency, low-information text tokens, thereby improving the information density of the text embeddings. These refined embeddings serve as enriched supervision signals, encouraging visual features to be more compact and semantically meaningful.

3.2 Vision-centric Implementation

VIST transforms raw textual data into M uniformly distributed RGB images $\mathcal{X} = \{x_m \in \mathbb{R}^{H\times W\times C}\}_{m=1}^{M}$, where M can be dynamically adjusted based on the length of the input text. Concretely, each image is configured with height $H = 14$, width $W = 3,584$, and $C = 3$ RGB channels, which corresponds to a square color image with a resolution of 224×224. Text is rendered using a 10px font size and Google *Noto Sans* typeface. If text incompletely fill the image, white empty patches are masked to exclude them from attention score computation and loss calculation. Compared to text tokenizer-based methods, this rendering method does not lead to slower training speeds [44].

3.3 Token Reduction

The M text-rendered images are first processed by frozen vision encoder, specifically the ViT-L/14 [11] from OpenCLIP. The extracted features $\boldsymbol{F} \in \mathbb{R}^{M\times L\times D}$ are then fed into a trainable Perceiver Resampler [49], producing a fixed set of $N+1$ visual tokens per image (including a CLS token), denoted as $\boldsymbol{F}' \in \mathbb{R}^{M\times(N+1)\times D}$, where $N = 64$ and D is the feature dimension. During training, raw text data ($T_e = 4096$ text tokens) is rendered onto $M = 28$ images, resulting in $64 \times 28 = 1792$ visual tokens, passed to the cross-attention layer in LLM. This compression reduces the computational complexity of the cross-attention layer within the LLM. Moreover, the number of images M and tokens N can be dynamically adjusted during both training and inference, *allowing* VIST *to flexibly control the compression ratio*. VIST using a lightweight vision encoder, offers a more efficient approach than processing all text tokens directly within the LLM.

3.4 Probability-informed Visual Enhancement

In VIST, the frozen vision encoder is pre-trained primarily on general visual data (such as natural images) without exposure to rendered text images. Hence its ability to interpret dense textual information within images is constrained. To alleviate this problem, we develop a novel training objective, named Probability-informed Visual Enhancement (PVE). PVE enhances the understanding capabilities of Perceiver Resampler for rendered text images, enabling them to serve as robust substitutes for traditional text tokenizers.

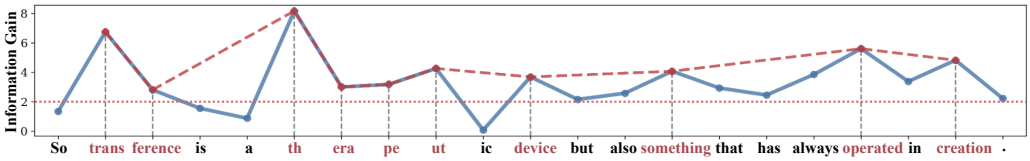


Figure 3: **Token-Level Information Gain** (IG) in sentence "*So **transference** is a **therapeut**ic **device** but also **something** that has always **operated** in **creation**.*". The red dashed line masks of 50% the most frequent tokens based on training set statistics. This strategy preserves tokens with higher information gain, while eliminating statistically prevalent but low-value tokens, enhancing semantic density.

Text-anchored Semantic Consistency. PVE encourages the Perceiver Resampler to learn a shared embedding space, aligning visual text features $\boldsymbol{F}^{'}$ with text token embeddings from text tokenizer. Concretely, PVE is formulated as a contrastive loss:

$$\mathcal{L}_{\text{PVE}}^{ij} = -\log \frac{\exp(\langle \hat{\boldsymbol{F}}_i^{'}, \hat{\boldsymbol{F}}_j^{\text{t}} \rangle / \tau)}{\sum_{k=1}^{B} \exp(\langle \hat{\boldsymbol{F}}_i^{'}, \hat{\boldsymbol{F}}_k^{\text{t}} \rangle / \tau)}, \tag{1}$$

where B is batch size and $\hat{\boldsymbol{F}}_i^{'}$ is obtained by applying average pooling to the CLS tokens from $\boldsymbol{F}_i^{'}$. $\hat{\boldsymbol{F}}_j^{\text{t}}$ is the averaged text token embedding after **frequency-based masking** and pooling. τ is the temperate parameter. Importantly, $\hat{\boldsymbol{F}}_i^{'}$ and $\hat{\boldsymbol{F}}_i^{\text{t}}$ are different representations derived from the same text.

Frequency-based Masking. PVE employs text token embeddings as supervision signals to guide the Resampler in extracting textual information from text images. However, long-form text is inherently redundant, where structural components and function words may dominate the token distribution. Such redundancy introduces noise that impedes Resampler from capturing **key semantic content**.

Our solution draws inspiration from Shannon information theory [50], which provides a formal way to quantify the information content of an event or a message. The formula is given by:

$$I(y) = -\log_2 P(y), \tag{2}$$

where $I(y)$ is the information content of event or messages y and $P(y)$ is the probability of y. It highlights the inverse relationship between the probability of an event and the information it carries. When applied to tokens in a corpus: **Rare tokens** (low-frequency) are treated as high-information tokens because they often carry domain-specific or contextually important information. **Frequent tokens** (high-frequency) have lower information content because they may serve more structural or grammatical purposes, contributing less to the unique meaning of the text. Figure 3 shows that masking 50% of the most frequent tokens based on corpus-level (*i.e.*, training set) frequency distribution still *preserves most high-information-gain (IG) tokens, ensuring minimal loss of critical information while reducing redundancy*. This aligns with the **selective reading strategies** [10, 51] observed in skilled readers. Based on this principle, we devise frequency-based masking strategy that uses token frequency as a proxy for semantic importance. This strategy masks frequent tokens but low-information tokens to improve the information density of text token embeddings. The importance score for each token is calculated as follows:

$$s_w = \log \frac{|\mathcal{S}|}{1 + \text{count}(w)}, \tag{3}$$

where $|\mathcal{S}|$ denotes the total number of samples, count(w) is the count of the token w (subword), and s_w is the importance score of token w. Token frequency statistics can be easily computed online with negligible overhead or precomputed. Based on the importance score for each token, we apply a 50% masking rate, where *tokens are randomly masked with tokens of lower importance score being more likely to be masked*. This ensures the Resampler prioritizes key content-bearing tokens, learning richer semantic representations and improving its ability to interpret dense text in rendered images.

4 Experiment

4.1 Experimental Setup

Pretraining. We validate VIST with TinyLlama [52]. The frozen vision encoder in our model is ViT-L/14 [11]. To reduce computational overhead, our model employs float16 precision and DeepSpeed Zero-2 with CPU off-loading[53]. Refer to Appendix A for details.

Table 1: Text perplexity on the last 256 tokens of long-context language modeling for ArXiv and Book datasets, PG19, Proof, and Code. T_e is token length for encoder, and T_d is for LLM. † denotes methods with the Resampler and PVE in our VIST. We report the Throughput of each model relative to TinyLlama. Δ is compression ratio.

Method	T_e	T_d	ArXiv	Book	PG19	Proof	Code	Throughput	Δ	TFLOPs	MEM(GB)
TinyLlama [52]	-	4096	$> 10^3$	$> 10^3$	$> 10^3$	$> 10^3$	$> 10^3$	1.0×	-	8.47	5.46
Replug [54]	-	4096	3.220	15.394	14.685	3.921	3.011	0.2×	-	9.15	6.12
Stream [55]	-	4096	3.116	15.188	14.372	**2.876**	2.764	1.6×	-	8.31	6.41
ToMe† [56]	2048	2048	3.536	15.607	16.213	4.128	3.234	3.8×	2.3	7.93	4.59
FastV† [57]	2048	2048	3.491	15.711	16.016	4.216	3.111	3.8×	2.3	7.88	4.59
CEPE* [21]	2048	2048	3.071	15.619	**11.737**	2.888	**2.151**	1.8×	-	8.26	4.80
VIST	2048	2048	**2.993**	**14.973**	13.205	3.057	2.247	3.8×	2.3	**7.72**(0.75↓)	**4.59**(0.87↓)
CEPE* [21]	6144	2048	3.005	14.919	**11.112**	**2.719**	**2.100**	2.1×	-	13.27	7.74
VIST	6144	2048	**2.989**	**14.894**	12.737	3.003	2.183	**5.3**×	2.3	**11.65**(1.62↓)	**4.94**(2.80↓)
CEPE* [21]	14,336	2048	3.003	14.921	**10.909**	**2.715**	2.162	3.3×	-	23.30	13.59
VIST	14,336	2048	**2.965**	**14.815**	11.933	2.971	**2.032**	**7.6**×	2.3	**19.52**(3.78↓)	**6.75** (6.84↓)

Competitors. i) Long-context models: Replug [54] and Stream [55] with TinyLlama [52]. **ii) Text-encoder-based compression method:** To compare the effectiveness of leveraging text tokens *v.s.*visual tokens for processing long contexts in LLM, we implement CEPE* by applying CEPE [21] to TinyLlama [52], replacing the vision encoder in VIST with a lightweight text encoder. All other architectural and training settings are kept identical. **iii) Vision-centric compression methods:** ToMe [56] merges, and FastV [57] prunes visual features from frozen vision encoders. For fairness, we match their compression rates to ours. Directly using these visual features in LLMs leads to high perplexity (>1k) due to the mismatch between visual and text tokens. Thus, we retain the Resampler and PVE in our VIST, denoting ToMe† and FastV†. See Appendix A.5 for more details.

Pretraining Dataset. Our pertaining dataset is an official sample of the RedPajama dataset [58], including 1B tokens from seven domains: ArXiv, Book, C4, Commoncrawl, GitHub, StackExchange, and Wikipedia. The training set of the corpus is preprocessed into 4608 text token sequences, where the first 4096 text token sequences are fed into the vision encoder (or text encoder for CEPE*) and the remaining 512 text tokens are provided to LLM.

Downstream Evaluation. We primarily evaluate tasks requiring long context processing, revealing vision tokens effectively handle extended context, outperforming previous text-encoder-based model. Comparisons across more methods and datasets are provided in the Appendix A.6.

4.2 Long-context Language Modeling

To assess the long-context language modeling (LCM) ability, we evaluate on ArXiv and Book from RedPajama [58] test split, alongside long-context datasets: PG19 [59], Proof [60], and Code [61]. The evaluation metric is perplexity (PPL) over the last 256 tokens of each input. Early tokens (typically farther from the current prediction point) are handled by the vision encoder, while the more recent tokens are passed to the LLM, under the assumption that proximity correlates with relevance.

Impact of Increased Text Length. Table 1 summarizes the results across different input lengths. Long-context language modeling can benefit from previous long contextual information. However, TinyLlama [52] supports only fixed-size inputs of 2048 tokens. Beyond this length, its performance drops sharply, with perplexity exceeding 10^3. In contrast, VIST demonstrates a consistent decrease in perplexity as the input text length increases. Vision-centric token compression models ToMe† and FastV† focus on natural images, where redundancy arises from local visual similarity. However, they are *ill-suited for text redundancy and struggle to preserve key semantic content in text image*, yielding higher perplexity than our VIST. Moreover, VIST obtains the lowest PPL (14.973) on Book datasets, when T_e and T_d are 2048. These results prove that **VIST effectively enhances the capability of modeling long-form language.**

Comparison on Inference Cost. In Table 1, VIST renders text into multiple images of size 224×224. 1024 text tokens need 7 images and VIST requires 56% fewer visual tokens than text tokens for the same input (*i.e.*, compression ratio Δ is 2.3, from 1024 text tokens to $448 = 7 \times 64$ visual tokens). We report the throughput of each model relative to TinyLlama. VIST achieves comparable performance with text-encoder-based compression model CEPE*, with 16% fewer FLOPs, 50% less memory usage, and higher throughput when processing 16k tokens.

Table 2: In-context learning accuracy averaged across 3 seeds (42, 43 and 44). Green highlights the gain from the additional demos. n_e is the number of demos for encoder and n_d for decoder (LLM).

Method	n_e	n_d	SST2	MR	AGN	SST5	NLUS	NLUI	TREC	TREF	DBP	BANK	CLIN	Avg.
TinyLlama [52]	-	2	76.0	67.7	63.4	27.6	5.2	4.4	28.8	9.6	38.0	23.0	22.4	33.3
TinyLlama [52]	-	20	**87.6**	71.7	**75.0**	30.1	**46.1**	32.6	**72.0**	**38.5**	**80.4**	**42.9**	**53.7**	**57.3**(24.0↑)
CEPE* [21]	18	2	76.9	**82.3**	66.9	29.1	9.6	30	39.2	12.7	71.1	27.2	39.8	44.1(10.8↑)
VIST	18	2	77.7	79.2	61.5	**42.7**	15.6	**40.6**	36.5	14.6	71.9	25.0	43.8	46.3(13.0↑)
TinyLlama [52]	-	50	**88.6**	64.8	21.4	42.5	**34.2**	30.4	**81.1**	**44.7**	3.4	**49.7**	39.7	45.5(12.2↑)
CEPE* [21]	48	2	82.9	79.4	63.9	42.3	28.1	31.1	32.6	14.7	71.5	29.0	39.1	46.8(13.5↑)
VIST	48	2	78.9	**85.2**	**71.9**	**44.4**	27.2	**43.1**	38.3	18.4	**73.1**	25.4	**48.1**	**50.4**(17.1↑)

4.3 In-context Learning

We evaluate VIST on In-Context Learning (ICL) tasks across 11 widely-used text-classification datasets: SST2 [62], MR [63], AGN [64], SST5 [62], TREC, TREF [65], DBP [64], NLUS, NLUI [66], BANK [67], and CLIN [68]. Following [21], we randomly sample 250 text examples per dataset. The ICL results in Table 2 are reported as the average accuracy over three random seeds. For VIST and CEPE*, we provide two demonstrations directly to the decoder, while the rest are processed by the encoder. More details in Appendix B.2.

Results. Table 2 examines the influence of increasing the number of demonstrations, where n_e is the number of demos for encoder and n_d for LLM. VIST shows a 13% accuracy improvement (from 33.3% to 46.3%) as more demonstrations (n_e is 18) are provided to the visual encoder, showcasing the capacity of LLM to comprehend text within visual signals when integrated with VIST. Furthermore, VIST outperforms CEPE* in average accuracy across all 11 datasets, which indicates **visual-based text understanding can effectively match or even surpass text encoder performance**. Though VIST and CEPE* ($n_e = 18, n_d = 2$) underperform TinyLlama ($n_d = 20$), they *achieve lower computational cost by processing most demonstrations (18) with lightweight encoder*. The performance gap on NLUS, TREC and TREF may stem from high category diversity, where the weak relevance between queries and demonstrations makes the lightweight encoding less effective than using the full LLM for all demos. Notably, the performance of TinyLlama declines with 50 demonstrations due to context window limit, while VIST remains efficient and stable.

4.4 Open-domain Question Answering

Open-domain Question Answering (QA) is a challenging task that requires model to generate accurate answers based on retrieved relevant information. Experiments are conducted on three open-domain QA datasets, including TriviaQA [69], NQ [70], and PopQA [71]. We use Contriever [72] to retrieve relevant k passages from Wikipedia, as in CEPE [21]. We prioritize passing the most relevant passages to the decoder to enhance performance. In Table 3, we report the exact match (EM) scores.

Table 3: **Open-domain QA results.** k_e represents the number of passages provided to the encoder, while k_d denotes the number of passages given to the LLM. We report the exact match score.

Method	k_e	k_d	TriviaQA	NQ	PopQA
TinyLlama [52]	-	10	21.45	8.45	10.79
TinyLlama [52]	-	15	< 1	< 1	< 1
VIST	10	0	21.27	8.51	10.67
CEPE* [21]	5	10	16.41	6.09	4.92
VIST	5	10	**25.20**(8.79↑)	**8.71**(2.62↑)	**11.44**(6.52↑)
CEPE* [21]	20	10	16.56	6.75	5.78
VIST	20	10	**25.67**(9.11↑)	**8.81**(2.06↑)	**11.84**(6.06↑)

Results. TinyLlama is limited by a maximum context window of 2048 tokens, restricting it to processing no more than 10 passages at a time. Beyond this limit, performance drops sharply, with EM score falling below 1. For passages $k_d = 10$, the input already approaches or even exceeds the 2048-token limit of TinyLlama, so we truncate the input to avoid performance degradation. When processing 5 extra passages (*i.e.*, $k_e = 5, k_d = 10$), VIST results in an EM score improvement of **3.75** compared to TinyLlama on TriviaQA [69] dataset. Moreover, it even surpasses text encoder-based approach CEPE* under the same input conditions, *e.g.*, delivering an EM score **9.11** points higher on the TriviaQA dataset when $k_e = 20, k_d = 10$. *This enhancement may be attributed to PVE in* VIST *which leverages enriched text embeddings to guide the Resampler in capturing key semantics.* By emphasizing critical details and filtering out noise from lengthy inputs, our VIST prioritizes relevant information—a crucial factor for success in open-domain QA tasks. In contrast, CEPE* degrades when more passages are provided to the encoder,

Table 4: The effect of visual token count for each image.

Tokens	ICL		Open-domain QA		
Per Image	TREC	MR	TriviaQA	NQ	PopQA
32	32.7	78.8	19.38	7.85	8.16
64	**36.5**	79.2	**25.20**	**8.71**	**11.44**
96	32.0	79.7	14.57	7.19	4.88
128	32.9	**87.0**	20.01	7.77	8.51

Table 5: The effect of different masking strategies in PVE. FM is frequency-based masking strategy, RM is random masking. The masking ratio is set to 50%.

RM	FM	ICL		Open-domain QA		
		NLUS	NLUI	TriviaQA	NQ	PopQA
		9.9	26.4	17.14	6.51	5.72
✓		8.3	30.2	24.88	8.35	10.19
	✓	**15.6**	**40.6**	**25.20**	**8.71**	**11.44**

Table 6: Ablation on the length of text inputs (in tokens).

Encoder	ICL		Open-domain QA		
Input Length	SST5	MR	TriviaQA	NQ	PopQA
1024	39.6	85.9	19.77	6.47	6.63
2048	39.3	73.8	22.85	8.08	9.77
4096	**42.7**	79.2	25.20	8.71	11.44
6144	37.8	**90.5**	**27.52**	**9.24**	**13.49**

Table 7: VIST effectively reduces PPL on LCM and improves accuracy on ICL by processing additional context. ‡ means our implementation on Mistral 7B [73].

Method	LCM		ICL	
	Arxiv	Book	SST2	DBP
Mistral	2.93	12.82	89.1	93.6
CEPE‡	2.83(0.10↓)	12.64(0.18↓)	90.8(1.7↑)	94.2(0.6↑)
VIST‡	**2.82(0.11↓)**	**12.61(0.21↓)**	**92.8(3.7↑)**	**95.3(1.7↑)**

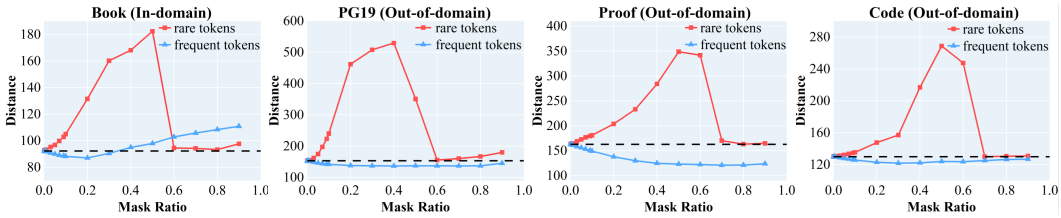


Figure 4: **Effect of frequent *vs.* rare tokens on semantic integrity.** Masking rare tokens (red) significantly disrupts semantic representation, increasing text-visual embedding distance, while masking frequent tokens (blue) has minimal impact, demonstrating that rare tokens are critical for preserving semantic meaning.

as additional passages introduce more noise and redundancy, making it harder to extract relevant answers. We also investigate the effectiveness of using only the fast path of VIST (*i.e.*, vision encoder) on the Open-domain QA task, which requires the model to generate accurate answers based on given relevant passages. Specifically, we feed only the top-10 relevant passages to the visual encoder (*i.e.*, $k_e = 10$), without providing any passages to the LLM directly (*i.e.*, setting $k_d = 0$). Surprisingly, this configuration yields performance on par with TinyLLaMA, despite the latter processing all 10 passages with a much heavier LLM. This demonstrates that the fast path of our method can distill and preserve the critical information from long contexts, providing a compact yet effective representation.

4.5 Ablation Study

We explore the effects of ❶ the masking strategy employed in PVE, ❷ the length of text provided to the encoder during training, ❸ the number of compressed tokens in each image (*i.e.*, N in §3.3), and ❹ extension to other LLM. In open-domain QA tasks, the model is fed 10 passages for LLM and 5 for encoder. ICL tasks use a fixed setup of 18 demonstrations for encoder and 2 for LLM. (More details in Appendix F.)

Number of Tokens in Each Image. The Resampler transforms image features into a fixed number of visual tokens. Table 4 analyzes the impact of visual token count for each image. Increasing the number of visual tokens reduces compression ratio, but as shown, ***a lower compression ratio does not always yield better results***. With 64 visual tokens, the model performs best on 4 out of 5 datasets, whereas 128 tokens only perform best on MR dataset. This discrepancy could be attributed to the trade-off between the amount of information preserved and the noise introduced during compression. Fewer tokens risk losing critical details, while more tokens may retain irrelevant information, which can hinder generalization. These findings emphasize that achieving a balance between compression and information retention is crucial for optimal performance across different datasets.

Masking Strategy in PVE (§3.4). Long texts often contain significant redundancy. To address this, we integrate a Frequency-based Masking (FM) strategy into PVE, improving the information density of text token embeddings. Table 5 compares the performance of VIST with FM, w/o FM, and with random masking. Excluding FM causes a notable decline in ICL and open-domain QA performance. This highlights the critical role of information-dense text token embeddings in guiding the visual encoder to capture more semantically meaningful and discriminative features.

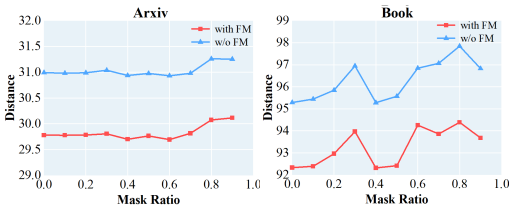


Figure 5: **Impact of Frequency-based Masking on Text-Visual Semantic Distance.** Compared to model without FM, VIST with FM consistently reduces the semantic distance between text token embeddings and visual features, across all masking ratios.

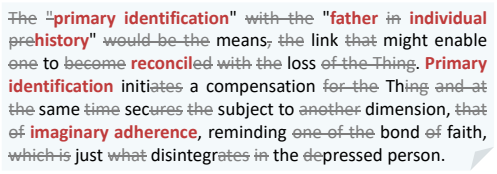


Figure 6: **Visualization of Masking Frequent Tokens.** Though we mask the top 50% most ~~frequent tokens~~ (based on training statistics), **key nouns** are completely preserved, proving that frequent tokens contribute less to semantic meaning.

In-context Text Length. Table 6 investigates the impact of in-context text length (measured in text tokens) provided to the visual encoder during training. Our model, trained with a longer encoder input length, generally yields higher EM scores on open-domain QA tasks. This may be because exposure to more lengthy texts during training helps our model better extract key information from extensive contexts, which is crucial for open-domain QA. Table 6 shows that longer training text inputs often boost ICL task accuracy. For instance, the best result on SST5 (42.7) was achieved with an encoder input length of 4096 text tokens, while the highest accuracy on MR (90.5) was obtained with the longest input length of 6144 tokens. Interestingly, we observed that training with an input length of 1024 tokens performed comparably to 2048 tokens, possibly because the total demo length for the encoder was close to 1024 tokens.

Extension to Other LLM. Mixture-of-expert models effectively scale model capacity while saving resources. To prove the generality of VIST, we also apply VIST to Mistral 7B [73]. In LCM task, VIST‡ and CEPE‡ process 4096 tokens, compared to the 2048 tokens processed by Mistral. For ICL, VIST‡ and CEPE‡ use 20 demonstrations, while Mistral uses only 2. As shown in Table 7, VIST‡ demonstrates superior performance over CEPE‡, by effectively leveraging additional context. To further validate the scalability of our method, results on larger-scale LLMs are included in Appendix F.

5 Discussion

Exploring Token Frequency as a Proxy for Semantic Importance. To assess the impact of rare versus frequent text tokens on global semantics across in-domain (Book [58]) and out-of-domain datasets (PG19 [59], Proof [60], and Code [61]), we first calculate importance score for each token using Eq. 3 based on training-set token frequency statistics. Two masking operations are then applied to the text token embeddings: ❶ Masking tokens with high importance scores (red line in Figure 4). ❷ Masking tokens with low importance scores (blue line in Figure 4). The distance between masked text embeddings and visual features is computed.

At low masking ratios (0.0 to 0.4), masking rare tokens causes a sharp increase in distance, while masking frequent tokens has minimal impact and even reduces distance. This suggests that rare tokens carry more critical semantic information, and their removal disrupts the alignment between text and visual tokens. In contrast, frequent tokens may contain more redundant or less informative content, so masking them has little impact or even improves the alignment by reducing noise. These findings support the hypothesis that *token frequency is a reasonable indicator of semantic importance, with rare tokens playing a more pivotal role in preserving semantic integrity*.

The Effect of Frequency-based Masking on Text-Visual Semantic Gap. VIST employs Frequency-based Masking (FM) within the Probability-informed Visual Enhancement to improve the semantic richness of text token embeddings. Figure 5 presents the impact of FM on the semantic alignment between text token embeddings and visual features extracted by the Perceiver Resampler. We experimented with random masking ratios (0.0 to 0.9) on text token embeddings and calculated the sum of cosine distances across all test samples in the Arxiv and Book datasets [58]. Across all ratios, VIST *with FM consistently exhibits smaller semantic distances than* VIST *without FM, highlighting FM effectively enhances semantic coherence*.

Token Redundancy Visualization. We study the semantic contribution of frequent tokens by selectively masking the top 50% most frequent tokens (based on training set statistics). Figure 6

illustrates key nouns are fully preserved. The masked tokens mainly include function words (*e.g.*, "the", "of"), primarily serving grammatical roles rather than conveying core semantic meaning. This proves FM strategy can **preserve critical semantic information while filtering out less relevant noise, thereby enhancing the ability of the model to focus on meaningful content**.

6 Conclusion

In-context learning with longer input sequences remains a prominent yet challenging topic in large language models (LLMs). In this work, we introduce a fully novel perspective to address this challenge by leveraging much lightweight visual encoder. To support longer input sequences in LLMs, we present VIST, a vision-centric token expansion method built upon a visual encoder framework. Our analysis further reveals there exists significant redundancy in text tokens, further validating the effectiveness and efficiency of our vision-encoder-based approach. With these advancements, VIST surpasses text-encoder-based token compression counterparts in both performance and efficiency. In future work, we plan to evaluate VIST across a broader range of downstream tasks and conduct a deeper investigation into text token redundancy.

Acknowledgement. This work is supported by the National Natural Science Foundation of China (Grant No. U25A20442, 62332010, 6250074347, 62472208, 62222207, 62427808).

References

[1] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. In *Advances in neural information processing systems*, volume 33, pages 1877–1901, 2020.

[2] Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

[3] Saku Sugawara, Xanh Ho, Anh-Khoa Duong Nguyen, and Akiko Aizawa. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*. International Committee on Computational Linguistics, 2020.

[4] Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. musique: Multihop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554, 2022.

[5] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[6] Yichun Yin, Wenyong Huang, Kaikai Song, Yehui Tang, Xueyu Wu, Wei Guo, Peng Guo, Yaoyuan Wang, Xiaojun Meng, Yasheng Wang, et al. Pangu ultra: Pushing the limits of dense large language models on ascend npus. *arXiv preprint arXiv:2504.07866*, 2025.

[7] Xiang Li, Yiqun Yao, Xin Jiang, Xuezhi Fang, Chao Wang, Xinzhang Liu, Zihan Wang, Yu Zhao, Xin Wang, Yuyao Huang, et al. 52b to 1t: Lessons learned via tele-flm series. *arXiv preprint arXiv:2407.02783*, 2024.

[8] Keith Rayner, Erik D Reichle, Michael J Stroud, Carrick C Williams, and Alexander Pollatsek. The effect of word frequency, word predictability, and font difficulty on the eye movements of young and older readers. *Psychology and aging*, 21(3):448, 2006.

[9] Alexander Strukelj and Diederick C Niehorster. One page of text: Eye movements during regular and thorough reading, skimming, and spell checking. *Journal of Eye Movement Research*, 11(1):10–16910, 2018.

[10] Ziwei Gu, Owen Raymond, Naser Al Madi, and Elena L Glassman. Why do skimmers perform better with grammar-preserving text saliency modulation (gp-tsm)? evidence from an eye tracking study. In *Extended Abstracts of the CHI Conference on Human Factors in Computing Systems*, pages 1–8, 2024.

[11] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763, 2021.

[12] Yiqi Lin, Conghui He, Alex Jinpeng Wang, Bin Wang, Weijia Li, and Mike Zheng Shou. Parrot captions teach clip to spot text. In *European Conference on Computer Vision*, pages 368–385. Springer, 2025.

[13] Yucheng Li, Bo Dong, Frank Guerin, and Chenghua Lin. Compressing context to enhance inference efficiency of large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 6342–6353, 2023.

[14] Huiqiang Jiang, Qianhui Wu, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 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, 2023.

[15] Kolby Nottingham, Yasaman Razeghi, Kyungmin Kim, JB Lanier, Pierre Baldi, Roy Fox, and Sameer Singh. Selective perception: Learning concise state descriptions for language model actors. In *Association for Computational Linguistics*, pages 327–341, 2024.

[16] Huiqiang Jiang, Qianhui Wu, Xufang Luo, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. Longllmlingua: Accelerating and enhancing llms in long context scenarios via prompt compression. *arXiv preprint arXiv:2310.06839*, 2023.

[17] Zhuoshi Pan, Qianhui Wu, Huiqiang Jiang, Menglin Xia, Xufang Luo, Jue Zhang, Qingwei Lin, Victor Rühle, Yuqing Yang, Chin-Yew Lin, et al. Llmlingua-2: Data distillation for efficient and faithful task-agnostic prompt compression. *arXiv preprint arXiv:2403.12968*, 2024.

[18] Philip Gage. A new algorithm for data compression. *The C Users Journal*, 12(2):23–38, 1994.

[19] Yintao Tai, Xiyang Liao, Alessandro Suglia, and Antonio Vergari. Pixar: Auto-regressive language modeling in pixel space. *arXiv preprint arXiv:2401.03321*, 2024.

[20] Phillip Rust, Jonas F Lotz, Emanuele Bugliarello, Elizabeth Salesky, Miryam de Lhoneux, and Desmond Elliott. Language modelling with pixels. In *International Conference on Learning Representations*, 2022.

[21] Howard Yen, Tianyu Gao, and Danqi Chen. Long-context language modeling with parallel context encoding. In *Association for Computational Linguistics*, 2024.

[22] David Wingate, Mohammad Shoeybi, and Taylor Sorensen. Prompt compression and contrastive conditioning for controllability and toxicity reduction in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 5621–5634, 2022.

[23] Peitian Zhang, Zheng Liu, Shitao Xiao, Ninglu Shao, Qiwei Ye, and Zhicheng Dou. Soaring from 4k to 400k: Extending llm's context with activation beacon. *arXiv preprint arXiv:2401.03462*, 2024.

[24] Jesse Mu, Xiang Li, and Noah Goodman. Learning to compress prompts with gist tokens. In *Advances in Neural Information Processing Systems*, volume 36, 2024.

[25] Alexis Chevalier, Alexander Wettig, Anirudh Ajith, and Danqi Chen. Adapting language models to compress contexts. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3829–3846, 2023.

[26] Tao Ge, Hu Jing, Lei Wang, Xun Wang, Si-Qing Chen, and Furu Wei. In-context autoencoder for context compression in a large language model. In *International Conference on Learning Representations*, 2024.

[27] Yuhong Li, Yingbing Huang, Bowen Yang, Bharat Venkitesh, Acyr Locatelli, Hanchen Ye, Tianle Cai, Patrick Lewis, and Deming Chen. Snapkv: Llm knows what you are looking for before generation. In *Advances in Neural Information Processing Systems*, volume 37, pages 22947–22970, 2024.

[28] Amirkeivan Mohtashami and Martin Jaggi. Random-access infinite context length for transformers. In *Advances in Neural Information Processing Systems*, volume 36, pages 54567–54585, 2023.

[29] Szymon Tworkowski, Konrad Staniszewski, Mikołaj Pacek, Yuhuai Wu, Henryk Michalewski, and Piotr Miłoś. Focused transformer: Contrastive training for context scaling. In *Advances in Neural Information Processing Systems*, volume 36, 2024.

[30] Peng Xu, Wei Ping, Xianchao Wu, Lawrence McAfee, Chen Zhu, Zihan Liu, Sandeep Subramanian, Evelina Bakhturina, Mohammad Shoeybi, and Bryan Catanzaro. Retrieval meets long context large language models. In *The Twelfth International Conference on Learning Representations*.

[31] Peitian Zhang, Shitao Xiao, Zheng Liu, Zhicheng Dou, and Jian-Yun Nie. Retrieve anything to augment large language models. *arXiv preprint arXiv:2310.07554*, 2023.

[32] Yuhuai Wu, Markus Norman Rabe, DeLesley Hutchins, and Christian Szegedy. Memorizing transformers. In *International Conference on Learning Representations*.

[33] Weizhi Wang, Li Dong, Hao Cheng, Xiaodong Liu, Xifeng Yan, Jianfeng Gao, and Furu Wei. Augmenting language models with long-term memory. In *Advances in Neural Information Processing Systems*, volume 36, 2024.

[34] Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, volume 1, page 2, 2019.

[35] Taku Kudo and John Richardson. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 66–71, 2018.

[36] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In *Association for Computational Linguistics*, pages 1715–1725, 2016.

[37] Elizabeth Salesky, David Etter, and Matt Post. Robust open-vocabulary translation from visual text representations. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 2021.

[38] Tianyu Gao, Zirui Wang, Adithya Bhaskar, and Danqi Chen. Improving language understanding from screenshots. *arXiv preprint arXiv:2402.14073*, 2024.

[39] Yekun Chai, Qingyi Liu, Jingwu Xiao, Shuohuan Wang, Yu Sun, and Hua Wu. Dual modalities of text: Visual and textual generative pre-training. *arXiv preprint arXiv:2404.10710*, 2024.

[40] Jonas Lotz, Elizabeth Salesky, Phillip Rust, and Desmond Elliott. Text rendering strategies for pixel language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 10155–10172, 2023.

[41] Chenghao Xiao, Zhuoxu Huang, Danlu Chen, G Thomas Hudson, Yizhi Li, Haoran Duan, Chenghua Lin, Jie Fu, Jungong Han, and Noura Al Moubayed. Pixel sentence representation learning. *arXiv preprint arXiv:2402.08183*, 2024.

[42] Geewook Kim, Teakgyu Hong, Moonbin Yim, JeongYeon Nam, Jinyoung Park, Jinyeong Yim, Wonseok Hwang, Sangdoo Yun, Dongyoon Han, and Seunghyun Park. Ocr-free document understanding transformer. In *European Conference on Computer Vision*, pages 498–517, 2022.

[43] Kenton Lee, Mandar Joshi, Iulia Raluca Turc, Hexiang Hu, Fangyu Liu, Julian Martin Eisenschlos, Urvashi Khandelwal, Peter Shaw, Ming-Wei Chang, and Kristina Toutanova. Pix2struct: Screenshot parsing as pretraining for visual language understanding. In *International Conference on Machine Learning*, pages 18893–18912, 2023.

[44] Michael Tschannen, Basil Mustafa, and Neil Houlsby. Clippo: Image-and-language understanding from pixels only. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11006–11017, 2023.

[45] Alex Jinpeng Wang, Linjie Li, Yiqi Lin, Min Li, Lijuan Wang, and Mike Zheng Shou. Leveraging visual tokens for extended text contexts in multi-modal learning. In *Advances in Neural Information Processing Systems*, 2024.

[46] Xiujun Li, Yujie Lu, Zhe Gan, Jianfeng Gao, William Yang Wang, and Yejin Choi. Text as images: Can multimodal large language models follow printed instructions in pixels? *arXiv preprint arXiv:2311.17647*, 2023.

[47] Liang Zhang, Anwen Hu, Jing Zhang, Shuo Hu, and Qin Jin. Mpmqa: multimodal question answering on product manuals. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 13958–13966, 2023.

[48] Anwen Hu, Haiyang Xu, Liang Zhang, Jiabo Ye, Ming Yan, Ji Zhang, Qin Jin, Fei Huang, and Jingren Zhou. mplug-docowl2: High-resolution compressing for ocr-free multi-page document understanding. *arXiv preprint arXiv:2409.03420*, 2024.

[49] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. In *Advances in neural information processing systems*, volume 35, pages 23716–23736, 2022.

[50] Claude Elwood Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948.

[51] Marc Brysbaert and Françoise Vitu. Word skipping: Implications for theories of eye movement control in reading. In *Eye guidance in reading and scene perception*, pages 125–147. Elsevier, 1998.

[52] Peiyuan Zhang, Guangtao Zeng, Tianduo Wang, and Wei Lu. Tinyllama: An open-source small language model. *arXiv preprint arXiv:2401.02385*, 2024.

[53] Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 3505–3506, 2020.

[54] Weijia Shi, Sewon Min, Michihiro Yasunaga, Minjoon Seo, Rich James, Mike Lewis, Luke Zettlemoyer, and Wen Tau Yih. Replug: Retrieval-augmented black-box language models. In *2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2024*, pages 8364–8377, 2024.

[55] Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. In *The Twelfth International Conference on Learning Representations*.

[56] Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. Token merging: Your vit but faster. In *International Conference on Learning Representations*.

[57] Liang Chen, Haozhe Zhao, Tianyu Liu, Shuai Bai, Junyang Lin, Chang Zhou, and Baobao Chang. An image is worth 1/2 tokens after layer 2: Plug-and-play inference acceleration for large vision-language models. In *European Conference on Computer Vision*, pages 19–35. Springer, 2024.

[58] Maurice Weber, Daniel Y. Fu, Quentin Anthony, Yonatan Oren, Shane Adams, Anton Alexandrov, Xiaozhong Lyu, Huu Nguyen, Xiaozhe Yao, Virginia Adams, Ben Athiwaratkun, Rahul Chalamala, Kezhen Chen, Max Ryabinin, Tri Dao, Percy Liang, Christopher Ré, Irina Rish, and Ce Zhang. Redpajama: an open dataset for training large language models. In *Advances in Neural Information Processing Systems*, 2024.

[59] Jack W Rae, Anna Potapenko, Siddhant M Jayakumar, Chloe Hillier, and Timothy P Lillicrap. Compressive transformers for long-range sequence modelling. In *International Conference on Learning Representations*, 2019.

[60] Zhangir Azerbayev, Edward Ayers, and Bartosz Piotrowski. Proofpile: A pre-training dataset of mathematical texts. 2023.

[61] Thomas Wolf, Loubna Ben Allal, Leandro von Werra, Li Jia, and Armel Zebaze. A dataset of python files from github, 2023.

[62] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642, 2013.

[63] Bo Pang and Lillian Lee. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Association for Computational Linguistics*, pages 115–124, 2005.

[64] Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In *Advances in neural information processing systems*, volume 28, 2015.

[65] Ellen M Voorhees and Dawn M Tice. Building a question answering test collection. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 200–207, 2000.

[66] Xingkun Liu, Arash Eshghi, Pawel Swietojanski, and Verena Rieser. Benchmarking natural language understanding services for building conversational agents. In *Increasing naturalness and flexibility in spoken dialogue interaction: 10th international workshop on spoken dialogue systems*, pages 165–183. Springer, 2021.

[67] Iñigo Casanueva, Tadas Temčinas, Daniela Gerz, Matthew Henderson, and Ivan Vulić. Efficient intent detection with dual sentence encoders. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 38–45, 2020.

[68] Stefan Larson, Anish Mahendran, Joseph J Peper, Christopher Clarke, Andrew Lee, Parker Hill, Jonathan K Kummerfeld, Kevin Leach, Michael A Laurenzano, Lingjia Tang, et al. An evaluation dataset for intent classification and out-of-scope prediction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, pages 1311–1316, 2019.

[69] Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Association for Computational Linguistics*, pages 1601–1611, 2017.

[70] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466, 2019.

[71] Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *Association for Computational Linguistics*, pages 9802–9822, 2023.

[72] Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. Unsupervised dense information retrieval with contrastive learning. *Transactions on Machine Learning Research*, 2022.

[73] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

[74] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

[75] I Loshchilov. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[76] Y Liu, M Ott, N Goyal, J Du, M Joshi, D Chen, O Levy, M Lewis, L Zettlemoyer, and V Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[77] Nir Ratner, Yoav Levine, Yonatan Belinkov, Ori Ram, Inbal Magar, Omri Abend, Ehud Karpas, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. Parallel context windows for large language models. *arXiv preprint arXiv:2212.10947*, 2022.

[78] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 11975–11986, 2023.

[79] Aydar Bulatov, Yury Kuratov, and Mikhail Burtsev. Recurrent memory transformer. *Advances in Neural Information Processing Systems*, 35:11079–11091, 2022.

[80] Pengpeng Li, Xiangbo Shu, Chun-Mei Feng, Yifei Feng, Wangmeng Zuo, and Jinhui Tang. Surgical video workflow analysis via visual-language learning. *npj Health Systems*, 2(1):5, 2025.

[81] Ling Xing, Hongyu Qu, Rui Yan, Xiangbo Shu, and Jinhui Tang. Locality-aware cross-modal correspondence learning for dense audio-visual events localization. *arXiv preprint arXiv:2409.07967*, 2024.

[82] Hongyu Qu, Rui Yan, Xiangbo Shu, Hailiang Gao, Peng Huang, and Guo-Sen Xie. Mvp-shot: Multi-velocity progressive-alignment framework for few-shot action recognition. *IEEE Transactions on Multimedia*, 2025.

[83] Hongyu Qu, Jianan Wei, Xiangbo Shu, and Wenguan Wang. Learning clustering-based prototypes for compositional zero-shot learning. In *International Conference on Learning Representations*, 2025.

[84] Hongyu Qu, Jianan Wei, Xiangbo Shu, Yazhou Yao, Wenguan Wang, and Jinhui Tang. Omnigaze: Reward-inspired generalizable gaze estimation in the wild. In *NeurIPS*, 2025.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The main claims made in the abstract and introduction accurately reflect our paper's contributions and scope.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The limitations of this work is discussed and the related details can be found in Appendix H.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: This paper does not include theoretical results. Instead, we provide comprehensive ablation study on our provided method.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We disclose all the information needed to reproduce the main experimental results of this paper and the data used in this paper is publicly available (see §4).

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: The code of this paper will be released later while the data used in this paper is publicly available.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: We specify all the training and test details in §4 and Appendix A to understand the experimental results.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: We report the average results over 3 different random seeds in In-context Learning Task in Table 2.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We conduct all experiments on NVIDIA V100 GPUs which is provided in Appendix A.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: The research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We discuss the broader impacts of this work in Appendix I

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.

- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: This paper poses no such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We have correctly and respectfully cited the original paper that produced the dataset used in this paper.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: About the code and model of this paper, they will be publicly available as soon as the paper is published.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: There is no research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: The core methodology and experimental pipeline of this study do not involve the use of any large language models (LLMs).

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (https://neurips.cc/Conferences/2025/LLM) for what should or should not be described.

The document is organized as follows:

- §A Experimental Details.
- §B Downstream Task Evaluation Details.
- §C Masking Empty Tokens.
- §D The Impact of Different Rendering Strategies.
- §E Comparison on Inference Cost.
- §F More Study Results.
- §G Visualization of Masking Frequent Tokens in Text.
- §H Limitations and Future Work.
- §I Broader Impacts.

A Experimental Details

A.1 Pretraining Details

We provide the data and optimization hyperparameters during the pre-training of VIST in Table 8. We conduct the experiments on NVIDIA V100 GPUs. The cross-attention layer is placed between the self-attention and feed-forward layers in each decoder (LLM) layer. When we first insert the cross-attention layers into the decoder, we initialize the weights of the key, value, and query projection matrices with the respective weights from the self-attention layer of the decoder in the same transformer block. Since the hidden dimension of the encoder $D = 1024$ is smaller than the hidden dimension of the decoder D', $D < D'$, we only copy the first D rows of the key and value projection matrices from the self-attention module to the cross-attention module. The output projection matrix is initialized with norms. The τ for PVE is 0.07. The weight of PVE is 1.

The rendered text images are shown in Figure 7. We also have several special design choices: (1) we do not attend to the white patches after the end-of-sequence text (following PIXEL [20]); (2) we normalize the input pixel values in each image.

We follow the Perceiver Resampler design used in Flamingo [49], as mentioned in Sec. 3.3. The Perceiver Resampler maps visual features from the frozen Vision Encoder to a fixed number of output tokens. Concretely, this transformer has a predefined number (*e.g.*, 64) of learnable latent vectors as queries, and the keys and values are a concatenation of the visual features with the learnable latent vectors. This mechanism allows the latent input queries to cross-attend to the visual features, producing compact visual representations. Note that the number of output tokens is equal to the number of latent vectors.

A.2 FLOPs Estimation

We estimate the FLOPs by calculating the number of operations involved in the forward pass for a single input instance. S pecifically, for each instance, we feed T_e tokens into the visual encoder and T_d tokens into the LLM. Only the forward pass is considered; decoding is excluded from FLOPs measurement. This provides a theoretical measure of computational complexity for a static input.

A.3 Throughput Measurement

Throughput is a practical measure reflecting the actual generation speed on specific hardware, defined as the number of output tokens generated per second during autoregressive decoding. For each input, our VIST processes T_e tokens via the visual encoder and T_d tokens by LLM, and then generates 256 tokens. We calculate Throughput as: $\frac{256}{\text{end-to-end latency}}$, where end-to-end latency refers to the total wall-clock time from receiving the input to the completion of generation (including text tokenization, image rendering, encoding, and decoding). For fair comparison, we ensure all models generate the same number of tokens from inputs of identical length. Note that we normalize the Throughput against TinyLLaMA under the same input-output setting to highlight relative speedups in Table 1.

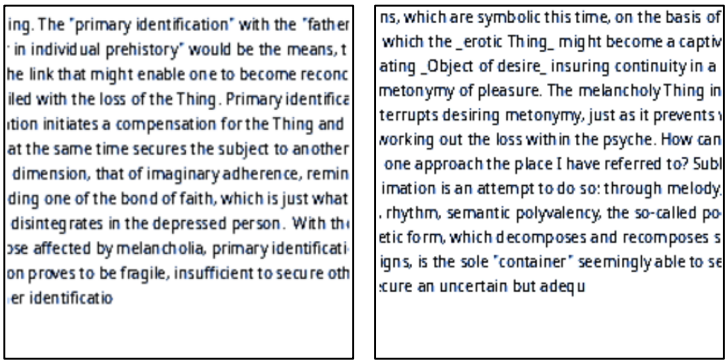


Figure 7: **Rendered text image.** We demonstrate the result of rendering 256 text tokens onto two 224x224 images.

A.4 Text Encoder-based Model

In this work, we additionally implement CEPE*. The text encoder in CEPE* follows the configuration of RoBERTa-large [76], as in CEPE [21]. The encoder contains approximately 400M parameters, while our visual encoder has around 300M parameters. Following CEPE [21], the encoder input tokens are segmented into 256-token chunks for parallel processing and then concatenated together for cross-attention in the LLM.

A.5 Vision-centric Compression Models

For fairness, we set the number of tokens to merge per layer to 8 for ToMe [56] and the reduction ratio to 75% for FastV [57] to match our compression rate. ToMe merges visual features from the frozen vision encoder, while FastV prunes them.

A.6 Comparison with Vision-centric and Long-context Models on More Datasets

Owing to space constraints, we present extended comparisons with vision-centric and long-context models on more datasets in the Appendix. We compare our method with vision-centric models (ToMe [56] and FastV [57]) and long-context models (Replug [54] and Stream [55]) on In-context Learning (SST5 and DBP) and open-domain QA (TriviaQA, NQ, and PopQA) tasks, as shown in Table 9. n_e is the number of demonstrations for encoder and n_d for decoder. k_e is the number of passages for encoder and k_d for decoder. To ensure a fair comparison, we provide a total of 20 demonstrations for the in-context learning tasks and 15 passages for the open-domain QA tasks.

Table 8: **Hyperparameters in VIST pretraining.**

Parameter	Value
Data	
Image size (height, width)	$(224, 224)$
Image mode	RGB
Font	Google Noto Sans
Font size	10
Optimization	
Peak learning rate	$3.0e^{-4}$
Warmup ratio	4%
Learning rate scheduler	Cosine decay [74]
Optimizer	AdamW [75]
β_1	0.9
β_2	0.999
ϵ	10^{-8}
Mixed precision training	`fp16`
Number of steps	2000 steps

Table 9: **Comparison with vision-centric and Long-context models.** on In-context Learning (SST5 and DBP) and Open-domain QA (TriviaQA, NQ, and PopQA) tasks.

Method	n_e	n_d	**SST5**	**DBP**	k_e	k_d	**TriviaQA**	**NQ**	**PopQA**
Replug	-	20	29.1	7.3	-	15	16.07	6.32	4.77
Stream	-	20	29.7	70.5	-	15	15.13	7.11	5.12
ToMe†	18	2	22.0	68.0	5	10	14.37	6.92	4.13
FastV†	18	2	24.8	65.4	5	10	14.69	6.08	4.96
VIST	18	2	**42.7**	**71.9**	5	10	**25.20**	**8.71**	**11.44**

Since RePlug and Stream do not incorporate an encoder module, all inputs are fed into the decoder. Table 9 shows our VIST achieves the best performance.

A.7 Extension to other LLM

When applying VIST or CEPE* to Mistral 7B, we leverage DeepSpeed Zero-2 with CPU offloading and use the SDPA attention mechanism. We further optimize the model by quantizing it to 4-bit precision.

B Downstream Task Evaluation Details

B.1 Long-context Language Modeling

Following CEPE [21], we evaluate VIST on three long context datasets: PG19 [59], Proof [60], and Code [61], sampling 5000 sequences for each dataset. During inference, VIST optimizes the efficiency and performance of long-text processing tasks by converting text input for vision encoders into images and masking out attention to the empty white patches in such images after the end of the text sequence.

B.2 In-Context Learning

For our in-context learning experiments, we follow the prompts used in [77, 21] for all datasets. We sample the in-context learning demonstrations from the training set such that each label has an equal number of demonstrations (except for possible remainders).

For CEPE*, the demonstrations for the text encoder are processed in parallel. In contrast, in Table 2, our approach renders all demonstrations into a suitable number of images based on text length, inserts blank lines at the end of each demonstration (similar to the role of "\n" in text processing), and then masks empty, text-free patches to eliminate distractions and enhance focus on meaningful content. More rendering strategies are discussed in Sec. D.

B.3 Open-domain Question Answering

For CEPE*, additional passages are encoded separately by the text encoder. For our approach, we first consolidate all passages and then render them into an appropriate number of images based on text length. The blank regions at the end of the images are excluded from attention computation.

B.4 Semantic Distance

In Figure 5 and 4, we calculate the global semantic distance between text token embeddings and visual features from Perceiver Resampler by computing the cosine distance for each sample and then summing the results. For the arXiv and Book datasets, we use the test set for this calculation, while for PG19, ProofPile, and CodeParrot, we randomly sample 100 samples from the test set to compute the distance.

In Figure 5, we fixed the random seed to ensure that the masked content is consistent for methods with and without FM. As the mask ratio increases from 0.1 to 0.5, the distance first rises and then falls.

Table 10: **The impact of masking empty tokens on In-context Learning** (ICL). ICL accuracy averaged across 3 seeds (42, 43 and 44). VIST* does not mask attention to the empty white patches after the end of the text sequence. VIST* and VIST use 2 demonstrations in the decoder, and the remaining demonstrations to the encoder.

Method	SST2	MR	AGN	SST5	NLUS	NLUI	TREC	TREF	DBP	BANK	CLIN	Avg.
Total Demonstrations = 20												
VIST*	88.7	88.9	65.2	36.9	8.4	24.8	38.8	11.4	61.2	18.0	42.0	44.0
VIST	77.7	79.2	61.5	**42.7**	15.6	**40.6**	36.5	14.6	71.9	25.0	43.8	46.3

Table 11: **The impact of masking empty tokens on Open-domain QA.** VIST* does not mask attention to the empty white patches after the end of the text sequence. k_e represents the number of passages provided to the encoder, while k_d denotes the number of passages given to the LLM. We report the exact match score.

Method	k_e	k_d	TriviaQA	NQ	PopQA
VIST*	5	10	21.82	7.53	8.71
VIST	5	10	**25.20**(3.38↑)	**8.71**(1.18↑)	**11.44**(2.73 ↑)

This is may be because random masking at ratios of 0.1 to 0.3 removes more critical information, while at 0.4 and 0.5, it may mask mostly redundant information.

C Masking Empty Tokens

We investigate the impact of masking out attention to the empty white patches after the end of the text sequence during both the training and inference stages. By comparing models that attend to these patches versus those that mask them, we analyze their effect on task performance. In Table 10, we present the results for In-context learning tasks, demonstrating that masking empty white patches leads to improved perplexity scores, as the model avoids unnecessary computations on non-informative regions. In Table 11, we show the results for open-domain QA tasks, where masking these patches enhances answer accuracy. These findings highlight the benefits of masking empty patches across different tasks, emphasizing its role in optimizing performance.

Table 12: **Different Rendering Strategies for 18 demonstrations.**

Method	Description
2-Image Rendering	All 18 demonstrations rendered into 2 images (high compression).
3-Image Rendering	All 18 demonstrations rendered into 3 images (balanced compression).
Dynamic Rendering	All demonstrations rendered into a suitable number of images based on text length.
Dynamic + Blank Lines	All demonstrations rendered into a suitable number of images with blank lines between demos.

D The Impact of Different Rendering Strategies

In this section, we investigate the impact of different rendering strategies on the in-context learning task. Specifically, we provide the vision encoder with 18 demonstrations and the LLM with 2 demonstrations, and experiment with the following rendering approaches: ❶ **Rendering all 18 demonstrations into 2 images**: This approach tests the ability of the model to handle highly condensed visual representations. ❷ **Rendering all 18 demonstrations into 3 images**: This strategy balances text density and image count, potentially improving readability by achieving a lower compression rate. ❸ **Rendering all demonstrations into a suitable number of images based on text length**: This dynamic approach ensures that each image contains an optimal amount of text, avoiding overcrowding or excessive fragmentation. ❹ **Rendering all demonstrations into a suitable number of images with blank lines inserted between demonstrations**: This method adds visual separation between demonstrations, mimicking the role of paragraph breaks in text processing.

Table 13: **Performance of Different Rendering Strategies on In-context learning tasks.** k_e represents the number of demonstrations provided to the encoder, while k_d denotes the number of demonstrations given to the LLM. We report the average accuracy across three random seeds.

Method	k_e	k_d	SST2	MR
2-Image Rendering	18	2	76.0	77.7
3-Image Rendering	18	2	76.6(0.6↑)	78.2(0.5↑)
Dynamic Rendering	18	2	77.0(1.0↑)	78.4(0.7↑)
Dynamic + Blank Lines	18	2	**77.7(1.7↑)**	**79.2(1.5↑)**

Table 14: **Comparing visual token counts to text token counts under different settings.**

Font Size	Visual Token Count	Tokens per Image	Text Token Count	Compression Ratio
10	256	64	218	3.41
9	256	64	240	3.75
8	256	64	270	4.22
7	256	64	310	4.84

The results, as shown in Table 13, reveal several key insights. The 3-Image Rendering strategy outperforms 2-Image Rendering primarily because it achieves a lower compression ratio, allowing for more detailed visual representations of the text. By distributing the 18 demos across 3 images instead of 2, the text is less densely packed, reducing the risk of information loss and improving the ability of the model to extract meaningful features. The Dynamic Rendering approach, which adapts the number of images based on text length, further improved performance. This indicates that dynamically adjusting the rendering process to match the content complexity is beneficial for maintaining contextual integrity. Finally, the Dynamic+Blank Lines method, which adds blank lines between demos, achieved the highest accuracy. These blank lines introduce clear visual separation, mimicking natural paragraph breaks and acting as visual cues to help the vision encoder distinguish between different text segments. This is particularly important as different demonstrations may belong to distinct categories or contexts. By visually separating them, the model can more easily process each segment independently, reducing the risk of confusion or misinterpretation and improving overall performance. For in-context learning, we adopt Dynamic+Blank Lines as the default rendering strategy.

Comparing Visual Token Counts to Text Token Counts under Different Settings. At font size 10, the number of visual tokens (256) from the frozen visual encoder slightly exceeds that of text tokens (218). However, with the Perceiver Resampler, our method reduces the final number of visual tokens per image to just 64. Despite this compression, VIST achieves comparable or even superior performance to the text-encoder-based baseline CEPE, as shown in the Table 14. These showcase that leveraging visual tokens for long-context compression is a promising and worthwhile direction to explore. At font sizes ≤ 8, the visual token count (256) is even lower than the text token count (310), demonstrating the intrinsic efficiency of the rendered text image. Even at font size 7, where compression is more aggressive, VIST outperforms the CEPE and performs on par with the font size 10 setting. These highlight the robustness of our method to font size variations and further support the effectiveness of using visual tokens for long-context compression.

E Comparison on Inference Cost

In Figure 8, we compared the memory usage and computational cost (FLOPs) of three methods—VIST, CEPE*, and TinyLlama—processing inputs of varying lengths. For VIST and CEPE*, when the total input length exceeds 2048 tokens, the excess portion is entirely handled by the encoder. For inputs shorter than or equal to 2048 tokens, the input is split equally between the encoder and the LLM. VIST demonstrates the lowest memory consumption across all input lengths, making it the most memory-efficient method. When processing 32k-length texts, CEPE*and TinyLlama encounter

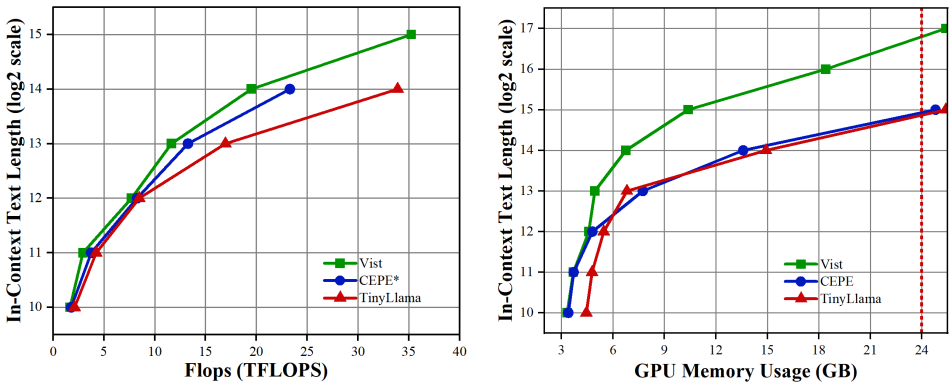


Figure 8: **VIST significantly increases the in-context text length from 16k to 64k at inference stage on a single 24GB RTX 4090 GPU, compared to CEPE*.** For CEPE* and VIST the input sequence is partitioned: the last 2048 tokens are processed by TinyLlama, while the remaining tokens are handled by a lightweight encoder.

Table 15: **The effect of different masking strategies in Probability-Informed Visual Enhancement on In-context learning**. We report the accuracy averaged across 3 seeds (42, 43 and 44). All methods use 2 demonstrations in the decoder and 18 demonstrations in the encoder. FM denotes Frequency-based Masking strategy, while RM is random masking strategy. The masking ratio is 50%.

FM	RM	SST2	MR	AGN	SST5	NLUS	NLUI	TREC	TREF	DBP	BANK	CLIN	Avg.
		70.1	77.3	55.3	34.5	9.9	26.4	36.3	11.2	61.3	15.2	42.8	40.0
	✓	72.8	76.2	60.1	40.5	8.3	30.2	32.6	13.3	66.0	20.1	38.9	41.7
✓		**77.7**	**79.2**	**61.5**	**42.7**	**15.6**	**40.6**	**36.5**	**14.6**	71.9	25.0	**43.8**	**46.3**

Out of Memory (OOM) on a 24GB GPU. Compared to CEPE*, VIST **extends the in-context text length from 16k to 64k tokens** during inference. VIST has a memory usage of 18.4 GB when processing a sequence of 64K, and only encounters an OOM error when processing a 128K sequence.

F More Ablation Study Results

In this section, we provide the complete results of our ablation study. Table 15 evaluates the impact of different masking strategies in Probability-informed Visual Enhancement on in-context learning performance. We compare two masking approaches: Frequency-based Masking (FM) and Random Masking (RM), with a fixed masking ratio of 50%. Our method with FM achieves a higher average accuracy compared to without FM or with RM. RM indiscriminately masks text tokens without considering their semantic relevance. This randomness can lead to the masking of critical tokens that are essential for understanding the context, resulting in a loss of important information and a degradation in model performance. While RM may occasionally perform well in certain datasets, its lack of focus on semantically rich tokens makes it less reliable overall. FM specifically masks high-frequency tokens, which are often low in semantic importance, allowing the visual features to focus more effectively on meaningful text tokens. By removing these less informative tokens, FM ensures that **the attention of the model is directed toward text segments that carry more significant contextual information**, thereby improving the quality of the learned representations.

The Impact of Different Visual Token Counts per Image. Table 16 investigates the impact of different visual token counts per image on in-context learning performance. The 64-token configuration achieves the highest average accuracy. This suggests that 64 tokens strike an optimal balance between capturing sufficient contextual information and avoiding excessive computational complexity. The 128-token configuration also performs well, particularly on SST2 (89.6) and MR (87.0), indicating that higher token counts can be beneficial for tasks requiring detailed context.

Table 16: **The effect of different visual token count for each image on In-context learning**. We report the accuracy averaged across 3 seeds (42, 43 and 44). All methods use 2 demonstrations in the decoder and 18 demonstrations in the encoder.

Tokens Per Image	SST2	MR	AGN	SST5	NLUS	NLUI	TREC	TREF	DBP	BANK	CLIN	Avg.
32	85.7	78.8	**63.5**	38.0	11.3	27.1	32.7	12.0	**73.4**	22.1	40.6	44.1
64	77.7	79.2	61.5	**42.7**	**15.6**	**40.6**	**36.5**	**14.6**	71.9	**25.0**	**43.8**	**46.3**
96	83.9	79.7	58.6	37.0	9.6	29.7	32.0	11.5	69.1	20.6	37.1	42.6
128	**89.6**	**87.0**	60.3	39.6	11.2	29.4	32.9	13.3	70.4	25.1	39.8	45.3

Table 17: **The effect of the length of text inputs (in tokens) provided to the visual encoder during training**. We report the accuracy averaged across 3 seeds (42, 43 and 44). All methods use 2 demonstrations in the decoder and 18 demonstrations in the encoder.

Encodr Input Length	SST2	MR	AGN	SST5	NLUS	NLUI	TREC	TREF	DBP	BANK	CLIN	Avg.
1024	77.6	85.9	**63.4**	39.6	12.2	32.9	35.3	13.9	66.1	**27.1**	42.4	45.1
2048	78.8	73.8	61.6	39.3	12.9	34.2	35.8	13.7	67.4	26.3	**44.8**	44.4
4096	77.7	79.2	61.5	**42.7**	**15.6**	**40.6**	**36.5**	**14.6**	**71.9**	25.0	43.8	**46.3**
6144	**81.7**	**90.5**	63.2	37.8	11.3	32.6	34.4	13.2	68.8	24.7	42.2	45.5

Interestingly, its average accuracy (45.3) is slightly lower than the 64-token setup, likely due to increased noise or redundancy in the visual representations.

The Effect of the Length of Text Tokens Provided to the Visual Encoder. In Table 17, we examine the effect of the length of text inputs (in tokens) provided to the visual encoder during training on in-context learning performance. The results indicate that the 4096-token input length achieves the highest average accuracy (46.3), outperforming other configurations across most datasets, which provides an optimal balance between capturing sufficient context and maintaining computational efficiency. The 6144-token configuration also performs well, particularly on MR (90.5), but its average accuracy (45.5) is slightly lower than the 4096-token setup, likely due to increased noise or redundancy in longer inputs. Similarly, the 1024-token and 2048-token configurations show lower performance, with average accuracies of 45.1 and 44.4, respectively, indicating that shorter input lengths may struggle to provide enough contextual information for the model to perform effectively.

The Impact of Different Frequency Masking Ratios. Table 18 examines the impact of different frequency masking ratios (30%, 50%, and 70%) on model performance. We observe that a moderate masking ratio (50%) leads to superior results, indicating the need for a trade-off between redundancy suppression and salient information preservation. Lower ratios (*e.g.*, 30%) may retain redundant or noisy frequency components, potentially distracting the model. In contrast, higher ratios (*e.g.*, 70%) risk discarding salient cues, leading to information loss. The 50% setting effectively suppresses irrelevant signals while preserving essential content, thus enhancing performance.

Swap Tokens Between the Visual Encoder and LLM. The relevance scores between passages and the question are available in open-domain QA. We conducted a study to examine how different passage allocation strategies affect performance. Specifically, in row 2 of the Table 19, we assign the top $-\ k_{\text{d}} = 10$ most relevant passages to the LLM and the less relevant $k_{\text{e}} = 5$ to the vision encoder. In row 3 of the Table 19, we swap the assignments (*i.e.*, less relevant passages go to the LLM while more relevant ones are compressed by the vision encoder). The performance remains comparable across both settings (*e.g.*, 8.71 vs. 8.56 on NQ), suggesting that even when highly relevant inputs are compressed, our method preserves essential information, highlighting the effectiveness and robustness of our slow-fast design.

The Impact of Different Vision Encoders. We conduct experiments using SigLIP-L [78] as the vision encoder. As shown in the Table 20, SigLIP-L achieves comparable performance to CLIP (ViT-L) within our VIST framework. This demonstrates the robustness and generality of our method across different vision encoders.

Extension to More LLMs with Different Scales. We apply VIST on LLaMa2-7B and LLaMa2-13B. Table 21 shows that our VIST with LLaMa2-7B/13B deliver consistent gains over CEPE with

Table 18: **The impact of masking ratio in Frequency-based Masking.**

Masking Ratio	TriviaQA	NQ	PopQA
30%	23.45	8.45	10.79
50%	25.20	8.71	11.44
70%	24.66	8.33	10.91

Table 19: **The impact of swapping tokens between the visual encoder and LLM.**

Method	k_e	k_d	TriviaQA	NQ	PopQA
VIST	5	top $-$ 10	25.20	8.71	11.44
VIST	top $-$ 10	5	24.68	8.56	11.35

LLaMa2-7B/13B (text-encoder-based baseline) and LLaMa2-7B/13B, highlighting the generalization ability of the proposed method. For the long-context modeling (LCM) task, the encoder is given extended 4096 tokens and the decoder 2048 tokens, with perplexity evaluated over the final 2048 tokens on Arxiv and Book datasets. For the in-context learning (ICL) task, we test on SST2 and DBP, providing 2 demonstrations to the decoder and 18 extra demonstrations to the encoder, and report accuracy.

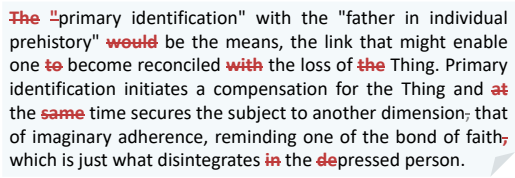


Figure 9: **Visualization of Masking Frequent Tokens with a 10% masking ratio.**

G Visualization of Masking Frequent Tokens

To demonstrate the effectiveness of our frequency-based masking strategy, we conduct a comparative analysis of token masking at different rates (10% in Figure 9 and 50% in Figure 10) based on the frequency statistics derived from the training corpus. The visualization results reveal distinct patterns in the masked token distribution. The most frequently occurring tokens are often common stopwords such as “the”, “of”, “and”, “with”, which typically carry less semantic weight. By masking these high-frequency tokens, the model is able to focus on more informative content, thereby improving the overall representation of key semantic features.

H Limitations and Future Work

One limitation of our approach lies in the static nature of the masking ratio. A dynamic masking ratio, adjusted based on token distribution, could improve performance. Due to resource constraints, we provide the LLM with a fixed input length of 512 tokens during training. Exploring longer inputs potentially enhances its ability to process more context and capture richer semantics, particularly in scenarios involving more complex or longer texts.

Multi-turn Dialog with VIST. To mitigate the inefficiency of re-encoding the entire context in streaming or multi-turn scenarios, a feasible solution is to adopt a lightweight memory caching strategy, inspired by summary accumulation techniques [79, 25], where summary vectors from all segments are concatenated to produce the summary of the entire content. Concretely, at the first dialogue turn, we encode the rendered text images using a frozen lightweight vision encoder followed by a Perceiver Resampler, which effectively compresses them into a small, fixed number of visual tokens (*e.g.*, 64 per image). These compressed visual tokens from Perceiver Resampler act

Table 20: **The impact of different vision encoders.**

Method	Vision Encoder	Arxiv	Book	SST5	NLUI
VIST	CLIP	2.82	**12.61**	42.7	**40.6**
VIST	SigLIP	**2.79**	12.71	**43.8**	40.1

Table 21: Results of extending our method to LLaMa models of different scales (7B and 13B). Our VIST consistently outperforms CEPE across model sizes, demonstrating strong scalability and generalization.

Method	LCM		ICL	
	Arxiv	Book	SST2	DBP
LLaMa2-7B	3.73	13.31	89.1	93.6
CEPE with LLaMa2-7B	3.11	12.78	90.2	94.0
VIST with LLaMa2-7B	**2.82**	**12.61**	**92.8**	**95.3**
LLaMa2-13B	3.30	11.42	92.0	94.8
CEPE with LLaMa2-13B	3.01	11.03	93.1	95.4
VIST with LLaMa2-13B	**2.57**	**10.87**	**94.3**	**96.2**

as summary vectors, capturing the essential semantics of each turn. Instead of discarding them, we cache these memory slots for future use. For each new dialogue turn, we concatenate the new input with all previous cached memory slots and jointly encode them using the same lightweight visual encoder and Perceiver Resampler. This allows the current input to interact with prior context without reprocessing the full raw history and incrementally accumulates dialogue history as a sequence of fixed-size memory slots. Because each turn contributes only a small number of tokens through compact visual representations, the total memory and compute overhead increases slowly—even across many turns. Meanwhile, the memory caching strategy enables each new input to be processed in the context of the entire conversation history. We believe this mechanism makes VIST readily extendable to streaming or interactive tasks, and plan to further explore its application to multi-turn multi-modal dialogue in future work.

Future work could investigate how different font sizes, types, or vision encoders impact performance, particularly in long-context scenarios. While our method currently uses RGB images, future work could explore techniques such as highlighting or bolding important tokens, or even transitioning from RGB to grayscale or binary images, to further reduce computational overhead and improve efficiency.

In this work, we demonstrate that employing a frequency-based masking strategy enhances the information density of text token embeddings. Though directly masking input tokens (*i.e.*, identifying and preserving more critical text tokens during the text-to-image rendering process) could further reduce computational overhead, it may risk degrading performance. Thus VIST employs masking strategy only to the text token embeddings used as supervision signals, while ensuring the integrity of the input data. Certainly, this requires further experimental validation, and it is a promising avenue for exploring a balanced method that optimizes efficiency without compromising performance.

I Broader Impacts

In practical applications, the demand for processing long texts is increasingly critical across various domains [2, 80], such as document summarization, legal analysis, medical record processing, and conversational AI. Efficiently handling long texts enables models to capture broader context, leading to more accurate and coherent outputs. In this work, we take a step to address this by leveraging visual tokens, which not only improve the efficiency and performance of current models but also open new avenues for further research in long-text processing. We further reveal high-frequency text tokens often contribute less to semantic meaning. This insight may inspire new algorithms to identify, filter, or simplify low-semantic-contribution tokens, reducing computational complexity, saving resources, and

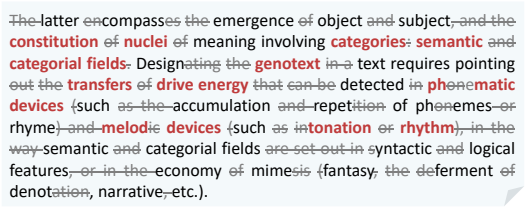


Figure 10: **Visualization of Masking Frequent Tokens with a 50% masking ratio.**

enhancing efficiency, especially for large-scale text data. Similar approaches can be extended to other domains, such as image or audio processing [81], to identify and eliminate redundant information, thereby enhancing overall efficiency. This perspective may also complements recent advances in few-shot and zero-shot generalization [82, 83], where models benefit from efficiently focusing on semantically rich cues rather than redundant patterns. Moreover, advances in gaze estimation [84] and visual saliency modeling further provide computational tools to simulate our frequency-mask based mechanism, bridging human perceptual studies and machine learning implementations.

Our work on long-context compression for LLMs improves efficiency, but it also carries potential societal risks. By enabling models to process longer contexts more effectively, it could inadvertently facilitate misuse, such as generating more convincing disinformation or harmful content that leverages extended context. Additionally, compressing long contexts may risk leaking sensitive information or amplifying biases if important contextual nuances are lost. While our method focuses on foundational algorithmic improvements, we acknowledge these risks and encourage future work on responsible deployment practices, including access controls and bias mitigation strategies, to minimize potential harms.