Unifying Demonstration Selection and Compression for In-Context Learning

Anonymous ACL submission

Abstract

In-context learning (ICL) facilitates large language models (LLMs) exhibiting spectacular emergent capabilities in various scenarios. Unfortunately, introducing demonstrations easily makes the prompt length explode, bringing a significant burden to hardware. In addition, random demonstrations usually achieve limited improvements in ICL, necessitating demonstration selection among accessible candidates. Previous studies introduce extra modules to perform demonstration compression or selection independently. In this paper, we propose an ICL framework UniICL, which **Uni**fies demonstration selection and compression, and final response generation via a single frozen LLM. Specifically, UniICL first projects actual demonstrations and inference text inputs into short virtual tokens, respectively. Then, virtual tokens are applied to select suitable demonstrations by measuring semantic similarity within latent space among candidate demonstrations and inference input. Finally, inference text inputs together with selected virtual demonstrations are fed into the same frozen LLM for response generation. Notably, UniICL is a parameter-efficient framework that only contains 17M trainable parameters originating from the projection layer. We conduct experiments and analysis over in- and out-domain datasets of both generative and understanding tasks, encompassing ICL scenarios with plentiful and limited demonstration candidates. Results show that UniICL effectively unifies 12× compression, demonstration selection, and response generation, efficiently scaling up the baseline from 4-shot to 64-shot ICL in IMDb with 24 GB CUDA allocation[1].

1 Introduction

In-context learning (ICL) (Brown et al., 2020; Xie et al., 2021; Wang et al., 2023b) exhibits powerful performance in practical applications with the emergence of scaled-up Transformer-based (Vaswani et al., 2017) Large Language Models (LLMs) (Wang et al., 2023c; Yang et al., 2023; Wei et al., 2023; Wang et al., 2023a; Min et al., 2022). In ICL, the provided demonstrations activate the pre-training knowledge of LLMs to allow them to perform well on various downstream tasks such as text summarization (Wang et al., 2023c; Yang et al., 2023) and text classification (Min et al., 2022) without gradient updating of billion parameters. Despite its significant role in the era of LLMs, ICL also brings an enormous challenge to the input window. Specifically, inevitably introducing demonstrations directly causes length disaster (Wang et al., 2024), which is more serious in long generative tasks such as question answering (Ghosal et al., 2022) and text summarization (Narayan et al., 2018), bringing significant memory costs and decreasing inference throughput as described in Figure 1. Except for length disaster, (Liu et al., 2021) points out that the quality of selected demonstrations significantly influences the ICL performance. The mentioned two issues derive two research lines focused on selecting salient demonstrations and utilizing demonstrations effi-

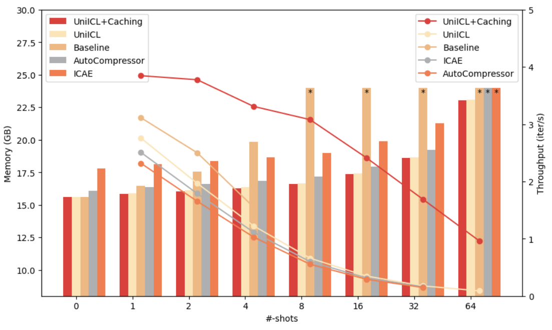

Figure 1: The efficiency comparison between UniICL and other compression methods in IMDb with the number of shots increasing from 0 to 64. * and the break of line chart indicates occurring memory explodes.

[1]The code and model will be released in the final version.

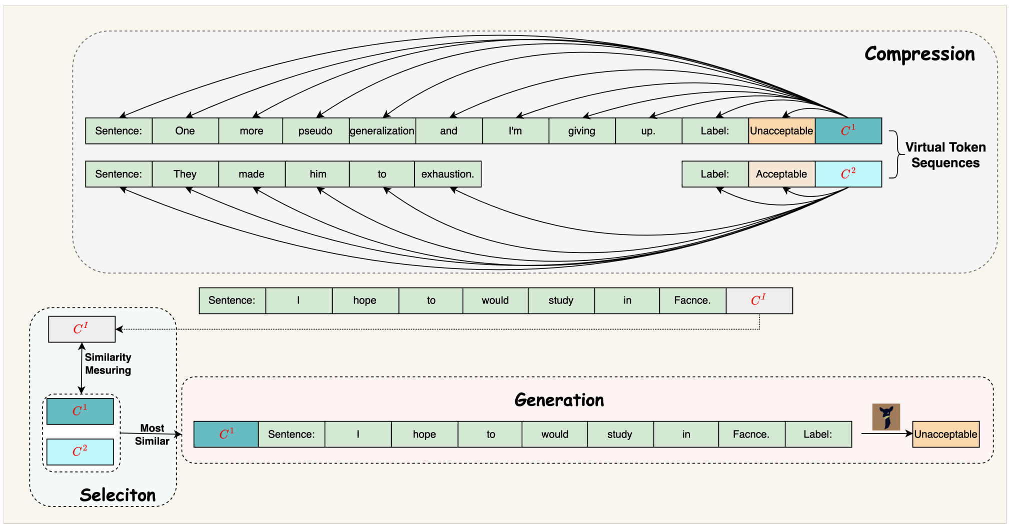
Compression
Sentence: One more pseudo generalization and I'm giving up. Label: Unacceptable C^1
Sentence: They made him to exhaustion. Label: Acceptable C^2
Virtual Token Sequences
Sentence: I hope to would study in Facnce. C^1
Similarity Mesuring
Most Similar
Generation
C^1 Sentence: I hope to would study in Facnce. Label:
Unacceptable
Seleciton

Figure 2: *Compressing* candidate demonstrations into **spans of compressed virtual tokens** and *selecting* one for the inference inputs. Compressed virtual tokens substitute the original demonstration together with the inference input, fed to the target LLM for prediction *generation*.

ciently.

Generally, researchers attempted to alleviate length explode (Wingate et al., 2022; Mu et al., 2023; Ge et al., 2023) via soft prompts. The main idea was to train an independent compressor to compress the input into compact virtual tokens. Similarly, (Liu et al., 2021; Lu et al., 2021) scores each candidate demonstration for the given inference input via an extra ranker, and (Ram et al., 2023; Wang et al., 2024) finds that a fine-tuned LLM is up to the task of ranking. However, the external compressor/ranker naturally leads to the Out-of-Distribution (OoD) problem between the compressor/ranker and the target LLM (Jiang et al., 2023), requiring extra large-scale data to perform alignment. In addition, the standalone compressor or ranker incurs additional memory costs as it necessitates simultaneous loading alongside the target LLM.

Motivated by the inherent semantic understanding ability of LLMs, we propose an ICL framework UniICL that unifies demonstration compression, demonstration selection, and response generation via the same frozen LLM. In UniICL, the same *frozen* LLM is applied thrice: it acts as the compressor to compress demonstrations into virtual tokens, as the selector to select suitable demonstrations based on the saliency measured by virtual tokens, and as the generator to produce final responses under the guidance of selected virtual demonstrations. Specifically, similar to the mask token in pretrained language understanding models (Devlin et al., 2018; Liu et al., 2019), UniICL introduces the compression slot [M] and attaches $k \times$ [M] behind demonstrations to be condensed. The LLM forwards each processed demonstration and outputs corresponding k hidden states on top of compression slots. Considering the latter compression slots naturally summarize the preceding demonstrations due to the transformer blocks, we train a simple projection layer to convert continuations to condensed virtual tokens. We briefly explain UniICL on the linguistic acceptability task concerning how it selects one demonstration among two compressed candidates and produces the prediction in Figure 2. On the one hand, virtual tokens effectively absorb semantic information from actual demonstrations, along with inference inputs, fed into the target LLM to perform generation as normal ICL. On the other hand, virtual tokens are again applied to measure saliency scores between demonstrations and given inference inputs within latent space, while demonstrations with higher scores are preferentially fed to the generator for response generation. Furthermore, UniICL compresses each demonstration independently of others, allowing caching virtual token sequences to configure a Demonstrations Bank (DB) for reusing, which will significantly bolster the inference throughput as demonstrated in Figure 1.

Generally, since the compressor, selector, and generator are the same frozen LLM, UniICL merely equips 17M trainable parameters originating from a projection layer and one special embedding [M], and we collect 30k public data for training. We extensively evaluate UniICL on in- and out-domain datasets of both generative and understanding tasks, covering the mainstream high- and low-resource ICL[2]. Results indicate that UniICL effectively substitutes 12× longer demonstrations with virtual tokens and scale up the inner LLM from 4-shot to 64-shot, significantly alleviating length disaster at an expense of 8% extra inference latency (without caching). Meanwhile, compared with the popular dense retriever, UniICL selects more suitable demonstrations in different tasks.

Our main contributions are as follows:

- To our knowledge, we are the first to propose an ICL framework that unifies compression, selection, and generation via a single frozen LLM, inherently bypassing the OoD problem.

- UniICL is a memory-friend framework that enables LLMs to perform large-shot ICL on consuming GPUs.

- Simple application of virtual tokens outperforms the popular dense retriever in selecting suitable demonstrations after jointly optimized by compression and selection augmentation loss.

2 Related Work

In this section, we simply rethink the methods focus on demonstration (prompt) compression and document retrieval.

2.1 Soft Prompt Compression

Recently, researchers attempted to utilize soft prompts to convert actual tokens to dense-information virtual tokens. Mostly from a distillation perspective, (Wingate et al., 2022) aligned the teacher model and the student model, where the teacher model accepted the actual task instruction while the student model fed the soft prompt. The main drawback of this approach was the lack of generalization that necessitated training for each lexically different instruction. To tackle the generalization problem, (Mu et al., 2023) proposed to learn a Llama-7b to compress instruction to virtual tokens, but only compress instruction was not powerful enough since the demonstrations were much longer in practical. To compress the demonstrations, (Chevalier et al., 2023) proposed AutoCompressor to generate compressed virtual tokens based on a fine-tuned LLM (Zhang et al., 2022). They first randomized segmented the texts with thousands of words into model-accepted range and then recursively generated soft prompts for each segment. Similarly, (Ge et al., 2023) proposed ICAE that employed a LoRA-adopted Llama-7b (Touvron et al., 2023) to compress the processed demonstrations to compact virtual tokens, and then fed to a frozen LLM to perform generation. Unlike previous works, the underlying LLM in UniICL was kept frozen during training and was trained on only 30k data.

2.2 Extractive Compression

Apart from employing soft prompts, researchers also endeavored to shorten prompts by extracting informative tokens from the original ones (Li, 2023; Jiang et al., 2023), namely token pruning (Kim et al., 2022) or token merging (Bolya et al., 2022). Recent works like LLMLingua (Jiang et al., 2023) and Selective Context (Li, 2023) shared similarities but diverged on whether to eliminate tokens with high or low Perplexity (PPL). LLMLingua emphasized tokens with high PPL, attributing them as more influential, resulting in achieving outstanding performance. As mentioned in their paper, extractive compression methods encountered Out-of-Distribution (OoD) issues between the extractor and the target LLM. To reconcile this, they fine-tuned Alpaca-7b (Taori et al., 2023) using the Alpaca dataset (Taori et al., 2023) to perform the alignment. However, UniICL naturally bypassed the distribution-aligned module, since the compressor and the target LLM were the same.

3 Methodology

We propose UniICL, a parameter-efficient ICL framework that unifies demonstration compression, demonstration selection, and response generation via a single LLM. Specifically, UniICL applies a general frozen LLM thrice, as a compressor to compress demonstrations into short virtual tokens, as a selector to choose suitable demonstrations for ICL, and as a generator to produce responses. As for the selection of the underlying LLM, previous work has proposed that the Decoder-only model performs better than the Encoder-Decoder model in prompt compression (Mu et al., 2023). We fol-

[2]High- and low-resource ICL are distinguished by the magnitude of candidates. The former usually selects demonstrations from plentiful candidates, and the latter merely selects from a handful of data.

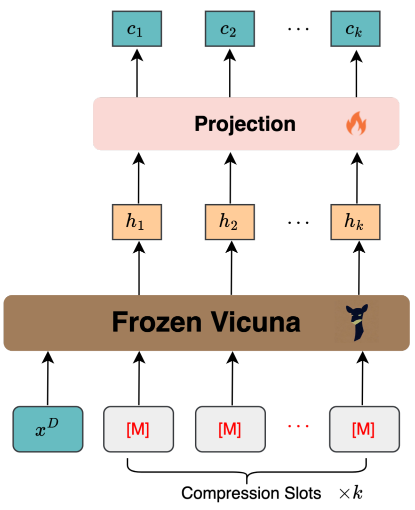

Figure 3: Demonstration compression.

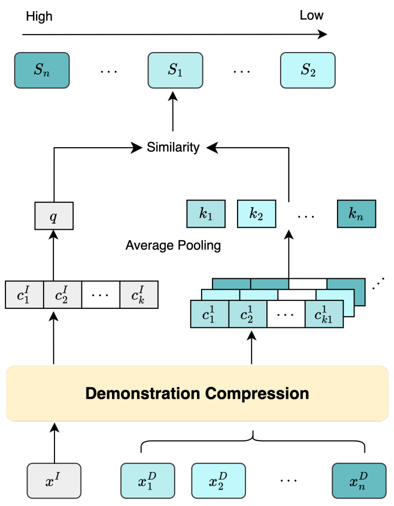

Figure 4: Demonstrations selection.

Methods	Additional Compressor	Compression Tool	# Trainable Parameters	Train Size
LLMLingua	YES	Pruning	7B	57k
AutoCompressor	NO	Soft Prompt	7B	UNKNOWN
ICAE	YES	Soft Prompt	70M	240k
UniICL	NO	Soft Prompt	17M	30k

Table 1: Comparsion among recent compression methods and UniICL. Compression Tool represents the involved compression technique of different methods.

low this conclusion and employ popular Vicuna-7b (Zheng et al., 2023) and BlueLM-7B[3] (Team, 2023) as the de facto backbone in UniICL.

We present a comparison between UniICL and other recent compression methods in Table 1. Without regard to the Lora-adopted compressor, ICAE (Ge et al., 2023) is most similar to UniICL in architecture and compression mode that parameter-efficiently compresses actual demonstrations into dense soft prompts.

3.1 Preliminary

Empirically, a prompt may encompass in-context demonstrations and an inference input in the ICL scenario. To avoid misleading and follow the existing definitions, we assume a prompt $P = (x_1^D, x_2^D, ..., x_n^D, x^I)$, where x_i^D is the ith demonstrations consist of an input-output pair (e.g. [DOC] -> [Sum]), and x^I is the inference input (e.g. documents in the summarization task). We first initialize the compression slots [M] $\in \mathrm{R}^d$ from the rarely used embedding of corresponding LLM, with gradient updating during training, d is the dimension of the input embeddings. The compression ratio r is a hyper-parameter, which is dynamically sampled from 2 to 16 during training, to calculate the specific number of compressed slots by $k = |x^D|/r$.

3.2 Demonstration Compression

For each demonstration in P, UniICL activates the compression slots to aggregate information from demonstrations in the forward propagation of frozen Vicuna respectively, as illustrated in Figure 3. Practically, we first attach k compression slots $M = k \times$ [M] to each demonstration x_i^D, formatting modified prompt fed to the Vicuna. Then, frozen Vicuna forwards the modified prompts and outputs the last hidden states $H_i = (h_1, h_2, ..., h_k)$ corresponding to k compression slot, dropping the others[4]:

$$\_, H_i = \mathrm{Encode}(x_i^D \oplus M) \tag{1}$$

Due to the transformer block in Vicuna, H_i is compelled to attend to the preceding actual tokens and UniICL inserts a linear layer to project H_i into compressed virtual tokens $C^i = (c_1, c_2, ..., c_k)$:

$$c_j = W_p \cdot h_j, \tag{2}$$

where W_p is the parameters of the projection layer.

3.3 Demonstration Selection

Except for substituting the origin demonstrations for generation, activated virtual tokens C^i are again applied for demonstration selection among candidates, as illustrated in Figure 4. Specifically, given

[3]We mainly utilize Vicuna-7B to verify the effectiveness, the experiments of BlueLM will be exhibited in Appendiex B for concise.

[4]$\oplus$ means token-level concatenation.

an inference input x^I and its candidate demonstrations $(x_1^D, x_2^D, ..., x_n^D)$, UniICL obtains their latent representation after average pooling:

$$\bar{C^i} = \frac{1}{k}\sum_{j=1}^{k} c_j, \tag{3}$$

denote as q for the input and $(k_1, k_2, ..., k_n)$ for candidate demonstrations for simlifying. We define the cosine similarity between q and k_i is the ith demonstration saliency score S_i:

$$S_i = cos(q, k_i). \tag{4}$$

3.4 In-context Generation

We employ the frozen Vicuna again to generate responses with the guiding of concatenated virtual demonstrations, as illustrated in Figure 5. For m-shot in-context learning, we obtain m virtual demonstrations after previously mentioned slot activating and demonstration selecting, denoted as C^1 to C^m. Then, we horizontally concatenate them, keeping their relative position unmodified. Finally, the concatenated virtual demonstrations together with actual inference inputs are fed into Vicuna, performing auto-regressive generation as normal:

$$y_i = \text{Decode}(C^1, ..., C^m; x^I; y_{<i}) \tag{5}$$

Except for the generative manner, virtual tokens are conveniently transferred to close-ended evaluation without modifications for understanding tasks through providing the candidate answers and measuring their perplexity (PPL) [5] respectively, e.g. (ppl^+, ppl^-) for sentiment classification:

$$\text{Prediction} = argmin(ppl^+, ppl^-), \tag{6}$$

where answers with PPL closest to 1 are judged to be the current prediction.

3.5 Training

The trainable parameters in UniICL are merely 17M originating from the projection layer W_p and the introduced compression slot [M]. We first use the common language modeling loss to optimize UniICL and get a base compression model. Then we add InfoNCE (He et al., 2020) to augment the demonstration selection ability:

$$\mathcal{L} = \mathcal{L}_{lm} + \mathcal{L}_{ctr}. \tag{7}$$

[5]https://huggingface.co/docs/transformers/perplexity

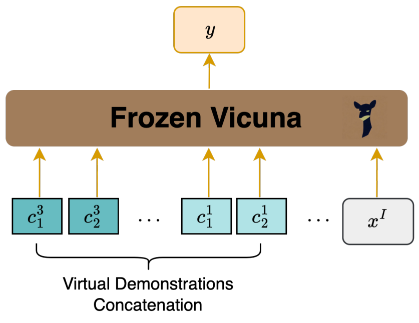

Figure 5: In-context generation.

Specifically, during the first training phase, we slice the source input of each training instance into two parts and randomly compress one, denoted the compression part as x_c and the unmodified as x_u, respectively. Afterward, we attach M to x_c and get virtual tokens C on top of the compression slot sequence, as described in Equ. 1 and Equ. 2. Therefore, the language modeling loss $\mathcal{L}_{lm}$ if obtained as:

$$\mathcal{L}_{lm} = -\frac{1}{|y|}\sum_{t=0} logP(y_t|x_2; C; y_{<t}), \tag{8}$$

where $y = (y_1, y_2, ...)$ is the reference label of the current training instasnce. Notably, distinguished from ICAE which always puts soft prompts in the front, UniICL keeps the relative position of C and x_u unchanged to make compressed tokens insensitive to their position in prompts. Namely, compressed tokens will be placed in the former (latter) if x_c is in front (behind) of x_u in the original sequence. Additionally, aiming to concatenate different virtual token sequences from different demonstrations for practical employment of ICL, we introduce concatenation compression that evenly slices the extra-long compression parts into two sub-segments during training, compresses them separately, and concatenates their virtual tokens horizontally.

Consequently, we utilize contrastive learning for selection augmentation and mine positives x_+^D and negatives x_-^D as illustrated in Figure 6. Specifically, given each training instance x^I and n candidate demonstrations $(x_1^D, x_2^D, ..., x_n^D)$ from two non-crossing training subsets, we employ Vicuna to calculate the PPL concerning the golden label of x^I, denoted as ppl^I. Then, we provide the ith demonstration and calculate its PPL, denoted as $(ppl_i^D, i \in [1, n])$. We count ppl^I as the baseline

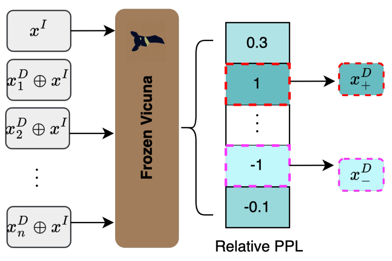

Figure 6: Negatives mining pipeline.

and calculate candidate relative PPL gains:

$$\widetilde{ppl_i^D} = ppl^I - ppl_i^D, i \in [1, n]. \tag{9}$$

Hence, k^+ (k^-) is the latent representation of demonstrations that furthest reduces (increases) ppl^I as processed in Equation 3, and the contrastive loss $\mathcal{L}_{ctr}$ can be formulated as:

$$\mathcal{L}_{ctr} = \frac{exp(cos(q, k^+))}{exp(cos(q, k^+)) + exp(cos(q, k^-))}. \tag{10}$$

In particular, if all relative PPL gains are less than 0, namely none of the candidate demonstrations help guide Vicuna to generate the golden label, we will apply the other set of candidates.

Notably, the inner compression ratio r for each training step is dynamic and randomly sampled from [2, 16] during training. After the two training phases, we blend 2,000 instances from the in-domain validation sets to evaluate the quality of virtual demonstrations provided to LLMs under various ratios. Ultimately, the preferred scheme involves compressing the demonstrations with $r = 12$, more details refer to Figure 7.

4 Experiment

In this section, we comprehensively evaluate UniICL over out-domain datasets to verify its effectiveness. Additionally, we detailedly introduce of involved datasets and evaluation metrics in Appendiex C.

4.1 Baselines

Naive Vicuna-7b serves as the fundamental baseline fed into actual demonstrations. AutoCompressor recurrently compresses demonstrations into virtual tokens. We employ their Llama2-7b version[6]. LLMLingua is a coarse-to-fine demonstration pruning method based on dropping uninformative words. We employ their released 7b version[7], of which compressor is a fine-tuned Llama-2. For a meaningful comparison, we replace target LLMs of LLMLingua (GPT-3.5-Turbo or Claude-v1.3) with the Vicuna-7b. ICAE[8] compresses demonstrations into soft prompts via a LoRA-adapted Llama2-7b, which is most similar to UniICL in model architecture and compression mode. Additionally, since selection augmentation is involved in the training of UniICL, we utilize the popular Sentence-BERT (S-BERT) (Reimers and Gurevych, 2019) as the dense retriever to construct an ICL pipeline for the above methods, serving as simple but effective selection-based baselines.

4.2 Settings

Considering the involved datasets and computation efficiency, we set the max allowed input length limit to 1,024 for both compression and generation. For a fair comparison, we set the allowed window of baselines to 1,024 and their compression ratio to 12. We fix the learning rate to 8e-5 and use Adam as the optimizer, and the effective batch size is 32 (8 GPUs data parallelism and 4 steps gradient accumulation). Additionally, we conducted all experiments on 8*NVIDIA A5000 24G GPUs based on BFloat 16 data type, and we set the evaluated shot to 8 for understanding tasks and 5 for generative tasks for illustration. Except for the MS MARCO preprocessed by Liang (Wang et al., 2022) [9], the other evaluation datasets are publicly achieved at hugging face.

We apply S-BERT to pre-rank and output the top 10 similar candidates from training sets according to each inference input and employ UniICL to perform selection among them in practice due to computation efficiency for high-resource ICL. On the contrary, the low-resource ICL setting fixes the candidate demonstrations to 20 for all inference inputs, performing pre-ranking and selecting as well.

4.3 Results

We comprehensively evaluate the ICL performance of UniICL on the out-domain dataset CoLA, SST-2, and IMDb by close-ended evaluation and ARXIV by open-ended evaluation, as demonstrated in Ta-

[6]https://github.com/princeton-nlp/AutoCompressors.

[7]https://github.com/microsoft/LLMLingua.

[8]https://github.com/getao/icae.

[9]https://github.com/microsoft/unilm/tree/master/simlm.

Model	#-shots	CoLA-dev	SST-2-dev	IMDb	ARXIV			XSum		
			Acc.		R-1	R-2	R-L	R-1	R-2	R-L
Vicuna	0-shot	56.2	91.7	92.6	34.3	9.1	27.4	19.9	5.0	13.4
	1-shot	58.2 (57.4)	90.7 (90.8)	91.9 (91.0)	34.4 (33.2)	9.1 (8.5)	27.5 (26.7)	21.2 (20.4)	5.8 (5.2)	14.5 (13.9)
	2-shot	62.1 (59.8)	92.1 (91.3)	91.7 (91.7)	-	-	-	-	-	-
	5-shot	62.3 (61.9)	93.0 (91.9)	94.1 (92.5)	-	-	-	-	-	-
AutoCompressor	1-shot	42.1 (40.9)	85.7 (84.2)	95.0 (95.1)	27.0 (26.4)	8.4 (8.2)	26.1 (25.8)	21.3 (20.3)	6.5 (6.3)	13.7 (13.7)
	2-shot	58.8 (56.3)	88.0 (86.4)	95.0 (94.6)	27.1 (26.2)	8.6 (7.9)	26.4 (25.4)	21.9 (21.4)	6.6 (6.4)	14.5 (14.1)
	5-shot	59.1 (58.8)	91.3 (89.1)	94.7 (94.8)	34.5 (33.7)	9.4 (9.1)	28.7 (27.9)	22.4 (21.7)	6.9 (6.7)	14.8 (14.3)
LLMLingua	1-shot	55.5 (55.0)	89.7 (89.6)	91.0 (89.9)	33.3 (33.1)	8.9 (8.7)	27.4 (27.1)	20.5 (19.7)	5.4 (5.2)	14.5 (14.4)
	2-shot	56.7 (55.7)	90.7 (90.2)	91.3 (91.0)	32.9 (32.0)	8.2 (8.1)	26.9 (25.9)	20.3 (20.0)	5.2 (5.1)	14.3 (14.1)
	5-shot	57.2 (56.9)	90.6 (90.2)	90.9 (91.2)	30.1 (29.7)	7.9 (7.4)	25.3 (24.6)	19.7 (18.6)	4.9 (4.9)	14.1 (14.3)
ICAE	1-shot	30.9 (30.9)	61.0 (60.1)	85.7 (83.3)	26.8 (24.6)	8.2 (7.1)	24.7 (22.9)	23.5 (21.9)	8.5 (7.8)	20.9 (20.3)
	2-shot	30.9 (30.9)	49.0 (52.8)	85.9 (85.9)	27.2 (25.5)	8.4 (7.6)	25.9 (24.3)	24.4 (23.2)	8.9 (8.4)	21.3 (20.8)
	5-shot	30.9 (30.9)	54.2 (51.0)	85.7 (85.9)	28.3 (26.9)	8.7 (7.7)	26.6 (25.8)	25.3 (24.9)	9.2 (8.8)	22.5 (21.6)
UniICL	1-shot	58.7 (58.0)	92.9 (91.7)	94.3 (92.3)	35.5 (34.7)	10.5 (10.2)	28.7 (27.9)	27.7 (25.5)	10.2 (9.1)	21.2 (20.0)
	2-shot	62.4 (61.0)	92.4 (91.6)	94.9 (93.3)	36.1 (35.2)	**10.8** (10.4)	29.4 (28.2)	29.4 (26.8)	11.0 (9.8)	22.3 (20.9)
	5-shot	62.6 (61.8)	93.1 (92.3)	94.5 (94.0)	35.8 (35.4)	10.6 (10.2)	29.5 (28.1)	30.7 (27.6)	11.3 (10.1)	22.8 (21.4)
UniICL$^{\spadesuit}$	1-shot	59.1 (58.7)	93.0 (91.9)	94.5 (91.6)	34.8 (34.7)	10.4 (10.3)	28.1 (27.8)	29.1 (26.2)	10.8 (9.4)	22.2 (20.7)
	2-shot	62.6 (61.2)	94.0 (93.0)	94.9 (92.3)	34.6 (34.3)	10.6 (10.4)	28.5 (28.3)	30.3 (28.9)	11.3 (10.5)	22.9 (21.7)
	5-shot	63.3 (61.5)	**94.7** (92.8)	95.0 (93.8)	35.6 (35.3)	11.0 (10.8)	29.1 (27.7)	31.1 (30.0)	11.7 (11.2)	23.5 (22.3)
	8-shot	63.8 (62.6)	**94.7** (93.1)	95.0 (94.2)	-	-	-	-	-	-
UniICL$^{\spadesuit}$ + L_{ctr}	1-shot	59.3 (58.9)	93.2 (92.4)	95.1 (92.8)	35.6 (35.1)	10.7 (10.5)	28.9 (28.3)	30.0 (27.9)	11.3 (10.1)	22.8 (21.5)
	2-shot	62.4 (62.0)	94.5 (92.8)	94.8 (93.4)	36.8 (35.3)	10.8 (10.6)	29.6 (28.9)	30.8 (29.2)	11.4 (10.7)	23.0 (21.9)
	5-shot	64.3 (61.8)	**94.7** (93.4)	**96.1** (94.2)	**37.1** (34.9)	**11.3** (11.2)	**30.0** (29.3)	**32.5** (30.6)	**12.3** (11.8)	**24.7** (23.8)
	8-shot	**64.7** (63.3)	**94.7** (94.1)	95.6 (95.0)	-	-	-	-	-	-

Table 2: The high- and low-ICL results on CoLA-dev, SST-2-dev, and IMDb. Results in () represent low-resource ICL. $^{\spadesuit}$ represents the demonstrations selected by UniICL, and the others are selected by S-BERT. +L_{ctr} indicates the selection augmented UniICL (optimized with Equation 7). **Bold** (underline) represents the best performance on high- and low-resource ICL.

ble 2. Specifically, UniICL outperforms naive Vicuna-7b fed with actual candidate demonstrations, which indicates that virtual demonstrations are more efficient and informative for guiding the target LLM, and UniICL outperforms all the baselines by compressing the same demonstrations preranked by S-BERT. Additionally, UniICL achieves further performance gains after selecting demonstrations via itself. Additionally, open-ended results indicate that virtual demonstrations still efficiently capture semantic information for ICL guiding, even though summarization demonstrations are much longer than understanding ones. Regarding ARXIV, the original ICL is not helpful enough due to its extremely over-length document, leaving little room for demonstrations. UniICL works as expected by compressing demonstrations and concatenating virtual demonstrations, and achieves +2.8 R-1 gains in the 5-shot setting that selects demonstrations via selection-augmented virtual tokens.

Furthermore, The ablation experiments of +$\mathcal{L}_{ctr}$ show that UniICL is faced with performance degradation without L_{ctr} and the performance gap becomes larger with the number of demonstrations increasing. The results of BlueLM are exhibited in Appendiex B.

Method	MS MARCO
BM25†	18.5
Vicuna	28.9
AutoCompressor	29.3
LLMLingua	27.8
ICAE	30.2
UniICL	**31.6**

Table 3: Results on MS MARCO. Vicuna applies the last hidden states of [EOS] to represent sentences in latent space. Following the previous study, we report MRR@10. † means citing from Liang (Wang et al., 2022).

Passage Ranking Since the virtual tokens naturally summarize semantic information of preceding sequences, we evaluate UniICL on the out-domain MS MARCO dataset in Table 3. UniICL significantly outperforms the sparse retrieval method BM25 algorithm and other compression methods. Notably, we don't compare UniICL with the popular retrieval models (Reimers and Gurevych, 2019; Wang et al., 2024) since most of them are fine-tuned on this dataset, unfair for comparison.

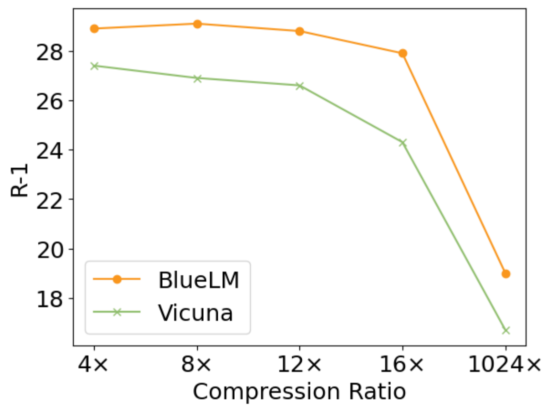

Figure 7: The overall sensitivity analysis of compression ratio.

5 Analysis

5.1 Compression Ratio

During training, the compression ratio is dynamically sampled from 2 to 16. We mix up 2,000 instances from the in-domain validation set, 1,000 for XSUM, and 1,000 for CICERO to select the compression ratio for UniICL in Figure 7, with the backbone of Vicuna and BlueLM respectively. Specifically, UniICL compresses the latter cut-off part while keeping the former ones uncompressed. Therefore, we can measure the dense information quality of the same content with different compression ratios by ROUGE-1 since it is more sensitive to token-level differences. The performance is relative smoothing when the compression ratio changes from $4\times$ to $12\times$. However, when it comes to $16\times$, an obvious drop occurs. Therefore, we set the compression ratio to 12 by default and apply this ratio to all experiments. The $1024\times$ compression ratio is equal to compressing anything to a single virtual token, due to the maximum allowed input length for compression being 1,024.

5.2 Efficiency Analysis

In UniICL, we incorporate an additional 17M trainable parameters into the 7b backbone, accounting for an approximate increase of 2.4%. We additionally evaluate the memory costs inference latency of UniICL and other compression methods in Figure 1. Notably, compressing each demonstration independently and allowing horizon concatenation among different virtual demonstrations convert the extra compression latency be a one-shot deal that the virtual tokens can be stored, formulating a Demonstration Bank (DB) for further reusing. In this case, UniICL will eliminate the extra latency if the selected demonstrations have been compressed and cached (UniICL+Caching). Despite this, parallel computation facilitates the compressing process, resulting in minimal throughput degradation (UniICL and Baseline). What's more, the naive 7B LLM occurs memory explosion for 8-shot settings, while UniICL successfully scales up to 64-shots within 24GB CUDA allocation.

5.3 Necessarity of UniICL

Intuitively, UniICL applies a "heavy" LLM as the compressor, but the generator and the compressor share all parameters without loading double parameters as LLMLingua, bypassing the OoD problem between the compressor and generator. Additionally, ICAE switches the LoRA parameters for compression and generations respectively, which appear to make the compressor and the generator share parameters. However, the compressor and the generator in ICAE are still two different LLMs inherently after assembling LoRA weight. Furthermore, the continual switch also breaks the latency-free merit of LoRA, resulting in more inference latency than the projection layer in UniICL, as depicted in Figure 1.

6 Conclusion

This paper proposes UniICL, a parameter-efficient ICL framework that unifies demonstration selection, demonstration compression, and final response generation via a frozen LLM. Experimental results show that the generated virtual tokens substitute the $12\times$ longer actual demonstrations with minimal time expenditure, scaling up the number of demonstrations from 4 to 64.

7 Limitations

Our study, while proposing an efficient unified ICL framework for demonstration compression and selection, still has limitations. Firstly, UniICL is limited to the realm of naive ICL leaving other advanced LLM prompting methods, e.g. Retrieval Augment Generation (RAG) and Chain-of-Thought (CoT) unexplored. Limited to the hardware, we employ the underlying LLM at a scale of 7 billion parameters. Larger-scale LLMs are welcome to enrich our findings in future studies.

References

Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. 2022. Token merging: Your vit but faster. *arXiv preprint arXiv:2210.09461*.

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

Alexis Chevalier, Alexander Wettig, Anirudh Ajith, and Danqi Chen. 2023. Adapting language models to compress contexts. *arXiv preprint arXiv:2305.14788*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Tao Ge, Jing Hu, Xun Wang, Si-Qing Chen, and Furu Wei. 2023. In-context autoencoder for context compression in a large language model. *arXiv preprint arXiv:2307.06945*.

Deepanway Ghosal, Siqi Shen, Navonil Majumder, Rada Mihalcea, and Soujanya Poria. 2022. Cicero: A dataset for contextualized commonsense inference in dialogues. *arXiv preprint arXiv:2203.13926*.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738.

Huiqiang Jiang, Qianhui Wu, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 2023. Llmlingua: Compressing prompts for accelerated inference of large language models. *arXiv preprint arXiv:2310.05736*.

Sehoon Kim, Sheng Shen, David Thorsley, Amir Gholami, Woosuk Kwon, Joseph Hassoun, and Kurt Keutzer. 2022. Learned token pruning for transformers. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 784–794.

Yucheng Li. 2023. Unlocking context constraints of llms: Enhancing context efficiency of llms with self-information-based content filtering. *arXiv preprint arXiv:2304.12102*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2021. What makes good in-context examples for gpt-3? *arXiv preprint arXiv:2101.06804*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2021. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. *arXiv preprint arXiv:2104.08786*.

Andrew Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pages 142–150.

Sewon Min, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. Noisy channel language model prompting for few-shot text classification. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5316–5330.

Jesse Mu, Xiang Lisa Li, and Noah Goodman. 2023. Learning to compress prompts with gist tokens. *arXiv preprint arXiv:2304.08467*.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. *ArXiv*, abs/1808.08745.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. Ms marco: A human generated machine reading comprehension dataset. *choice*, 2640:660.

Ori Ram, Yoav Levine, Itay Dalmedigos, Dor Muhlgay, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. 2023. In-context retrieval-augmented language models. *arXiv preprint arXiv:2302.00083*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. 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.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. 2023. Stanford alpaca: An instruction-following llama model.

BlueLM Team. 2023. Bluelm: An open multilingual 7b language model. https://github.com/vivo-ai-lab/BlueLM.

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

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Jiaan Wang, Yunlong Liang, Fandong Meng, Haoxiang Shi, Zhixu Li, Jinan Xu, Jianfeng Qu, and Jie Zhou. 2023a. Is chatgpt a good nlg evaluator? a preliminary study. *arXiv preprint arXiv:2303.04048*.

Lean Wang, Lei Li, Damai Dai, Deli Chen, Hao Zhou, Fandong Meng, Jie Zhou, and Xu Sun. 2023b. Label words are anchors: An information flow perspective for understanding in-context learning. *arXiv preprint arXiv:2305.14160*.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2022. Simlm: Pre-training with representation bottleneck for dense passage retrieval. *arXiv preprint arXiv:2207.02578*.

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2024. Large search model: Redefining search stack in the era of llms. In *ACM SIGIR Forum*, volume 57, pages 1–16. ACM New York, NY, USA.

Zengzhi Wang, Qiming Xie, Zixiang Ding, Yi Feng, and Rui Xia. 2023c. Is chatgpt a good sentiment analyzer? a preliminary study. *arXiv preprint arXiv:2304.04339*.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. 2018. Neural network acceptability judgments. *arXiv preprint 1805.12471*.

Xiang Wei, Xingyu Cui, Ning Cheng, Xiaobin Wang, Xin Zhang, Shen Huang, Pengjun Xie, Jinan Xu, Yufeng Chen, Meishan Zhang, et al. 2023. Zero-shot information extraction via chatting with chatgpt. *arXiv preprint arXiv:2302.10205*.

David Wingate, Mohammad Shoeybi, and Taylor Sorensen. 2022. Prompt compression and contrastive conditioning for controllability and toxicity reduction in language models. *arXiv preprint arXiv:2210.03162*.

Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. 2021. An explanation of in-context learning as implicit bayesian inference. *arXiv preprint arXiv:2111.02080*.

Xianjun Yang, Yan Li, Xinlu Zhang, Haifeng Chen, and Wei Cheng. 2023. Exploring the limits of chatgpt for query or aspect-based text summarization. *arXiv preprint arXiv:2302.08081*.

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

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *arXiv preprint arXiv:2306.05685*.

A In-Domain Evaluation

Method	XSUM			CICERO		
	R-1	R-2	R-L	R-1	R-2	R-L
Vicuna	19.9	5.0	13.5	17.3	3.3	14.3
+UniICL	27.2	9.2	22.5	24.6	5.1	21.7

Table 4: The in-domain results of XSUM and CICERO. Vicuna generates responses conditioned on the truncated inference input and UniICL first compresses the over-length input into virtual tokens.

We conduct the zero-shot in-domain generation evaluation on the entire test set of XSUM and CICERO in Table 4, by compressing the latter half to virtual tokens and keeping the former unmodified. UniICL significantly outperforms the baseline Vicuna-7b, indicating the compressed virtual tokens can provide the original truncated information by recovering the cut-off parts after supervised fine-tuning.

B Results on BlueLM

We extra conduct experiments on BlueLM (Team, 2023) to verify the generality of UniICL. We demonstrate the result of understanding tasks in Table 5, of the generative tasks in Table 6.

Model	#-shots	CoLA-dev	SST-2-dev	IMDb
			Acc.	
BlueLM	0-shot	71.6	81.2	48.8
	1-shot	69.6	82.6	64.8
	2-shot	70.0	87.0	65.6
	5-shot	70.5	88.6	68.7
UniICL	1-shot	69.6	81.2	65.4
	2-shot	68.7	82.6	67.0
	5-shot	71.7	87.0	70.4
UniICL♠	1-shot	69.8	80.0	62.0
	2-shot	70.1	80.8	67.0
	5-shot	71.8	85.6	69.6
	8-shot	72.3	87.4	69.4
UniICL♠ + L_{ctr}	1-shot	70.1	80	69.6
	2-shot	70.3	87.2	70.6
	5-shot	71.1	89.2	71.0
	8-shot	72.5	90.4	76.8

Table 5: The ICL results of understanding tasks with the backbone of BlueLM.

Method	#-shots	XSUM			ARXIV		
		R-1	R-2	R-L	R-1	R-2	R-L
BlueLM	0-shot	23.5	6.8	17.6	30.9	7.7	24.7
	1-shot	19.1	4.8	12.1	23.0	3.6	19.0
UniICL	1-shot	24.0	6.9	18.0	31.4	7.7	25.2
	2-shot	25.0	7.3	18.8	30.8	7.3	24.8
	5-shot	25.0	7.3	18.8	30.8	7.3	24.8
UniICL♠	1-shot	25.2	7.4	18.9	31.6	7.9	25.4
	2-shot	25.4	7.6	19.1	31.9	8.0	25.6
	5-shot	26.5	7.9	20.3	32.1	8.0	25.5

Table 6: The ICL results of generative tasks with the backbone of BlueLM.

Dataset	# words		
	(96,512]	(512,1024]	(1024,1536]
XSUM	-	10,000	4,697
CICERO	10,000	-	-
XSUM (Ctr)	5,000		

Table 7: The composition training set of UniICL. (m,n] represents the range of the number of words in each instance. XSUM (Ctr) is used for the second phase training in Equation 7.

C Datasets & Metrics

C.1 Datasets

We mix up two public datasets for compression and selection augmentation training, described in Table 7. Additionally, UniICL achieves outstanding performance on out-domain evaluation, involving text summarization (Narayan et al., 2018), passage ranking (Nguyen et al., 2016), sentiment classification (Maas et al., 2011; Socher et al., 2013), and linguistic acceptability (Warstadt et al., 2018), more details referring to Table 8. UniICL selects demonstrations from its training set in high-resource ICL, and we fixed the number of candidate demonstrations to 20 for low-resource ICL evaluation.

C.2 Evaluation Metrics

ROUGE (Lin, 2004) is a widely adopted metric in many generative tasks that evaluate how similar the generated hypothesis is to the golden label. Therefore, ROUGE is used in our experiments to evalu-

Dataset	Task	In-Domain	# Test	# Demonstrations
MS MARCO-dev	Passage Ranking	✗	6,980	-
XSUM	Text Summarization	✓	1,500	204,045/20
ARXIV	Text Summarization	✗	1,500	203,037/20
CoLA-dev	Lingustic Acceptability	✗	1,041	67,349/20
SST2-dev	Sentiment Classification	✗	872	8,551/20
IMDb	Sentiment Classification	✗	1,500	25,000/20

Table 8: The details of involved evaluation datasets. -dev represents employing development set due to their test sets are inaccessible. # Demonstrations represent the number of demonstrations to be selected in **high**/low-resource ICL settings.

ate the quality responses generated conditioned on compressed virtual tokens, and we report the F-1 scores of ROUGE-1, ROUGE-2, and ROUGE-L (abbreviated R-1, R-2, R-L in the following), and we employed the files2rouge [10] library in practice. Following the previous works, we report the accuracy of close-ended evaluation and MRR@10 for passage ranking.

[10]https://github.com/pltrdy/files2rouge.