Long-CLIP: Unlocking the Long-Text Capability of CLIP

Beichen Zhang[§1,2], Pan Zhang[1], Xiaoyi Dong[1,3⋆], Yuhang Zang[1], and Jiaqi Wang[1⋆]

[1]Shanghai AI Laboratory [2]Shanghai Jiao Tong University
[3]The Chinese University of Hong Kong
zhangbeichen@sjtu.edu.cn,
{zhangpan, dongxiaoyi, zangyuhang, wangjiaqi}@pjlab.org.cn
https://github.com/beichenzbc/Long-CLIP

Abstract. Contrastive Language-Image Pre-training (CLIP) has been the cornerstone for zero-shot classification, text-image retrieval, and text-image generation by aligning image and text modalities. Despite its widespread adoption, a significant limitation of CLIP lies in the inadequate length of text input. The length of the text token is restricted to 77, and an empirical study shows the actual effective length is even less than 20. This prevents CLIP from handling detailed descriptions, limiting its applications for image retrieval and text-to-image generation with extensive prerequisites. To this end, we propose Long-CLIP as a plug-and-play alternative to CLIP that supports long-text input, retains or even surpasses its zero-shot generalizability, and aligns the CLIP latent space, making it readily replace CLIP without any further adaptation in downstream frameworks. Nevertheless, achieving this goal is far from straightforward, as simplistic fine-tuning can result in a significant degradation of CLIP's performance. Moreover, substituting the text encoder with a language model supporting longer contexts necessitates pretraining with vast amounts of data, incurring significant expenses. Accordingly, Long-CLIP introduces an efficient fine-tuning solution on CLIP with two novel strategies designed to maintain the original capabilities, including (1) a **knowledge-preserved stretching** of positional embedding and (2) a **primary component matching** of CLIP features. With leveraging just one million extra long text-image pairs, Long-CLIP has shown the superiority to CLIP for about 20% in long caption text-image retrieval and 6% in traditional text-image retrieval tasks, *e.g.*, COCO and Flickr30k. Furthermore, Long-CLIP offers enhanced capabilities for generating images from detailed text descriptions by replacing CLIP in a plug-and-play manner. Codes and models are released at https://github.com/beichenzbc/Long-CLIP.

Keywords: Multimodality · Zero-shot Image Classification · Text-Image Retrieval · Text-to-Image Generation

⋆ Corresponding author. § Work done during an internship in Shanghai AI Laboratory.

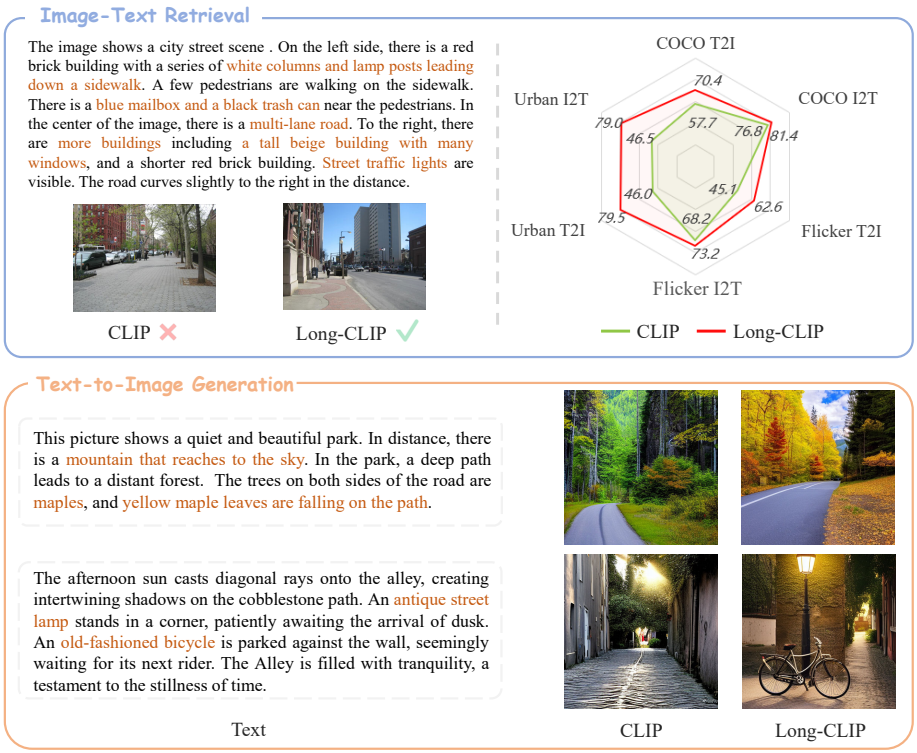


Fig. 1: Our Long-CLIP model can capture more detailed attributes and break through the limit of 77 tokens, thus benefiting both **image-text retrieval** tasks and **text-to-image generation** tasks. For retrieval, our Long-CLIP model can better capture and represent fine-grained attributes in both modalities, and therefore significantly improves the performance. For image generation, our Long-CLIP model offers enhanced capabilities for generating from detailed text descriptions by replacing the text encoder of CLIP in a plug-and-play manner.

1 Introduction

Contrastive Language-Image Pre-training (CLIP) [20] is a vision-language foundation model comprising a text encoder and an image encoder. It aligns the vision and language modality based on contrastive learning, widely adopted in downstream tasks, *e.g.*, zero-shot classification [26], text-image retrieval [16], and text-to-image generation [6, 23].

Nevertheless, the input text length of CLIP is greatly restricted, making it unsuitable for processing detailed descriptions. Specifically, CLIP's text encoder employs an absolute positional embedding limited to 77 tokens, establishing a strict ceiling on input token numbers. Moreover, since the training datasets for CLIP predominantly consist of brief texts, the positional embedding for higher token positions in the CLIP's text encoder are inadequately trained, resulting

in an even shorter effective token length. An empirical investigation, detailed in Sec. 3.1, reveals that the actual effective length for CLIP is merely 20 tokens.

Furthermore, the absence of a long-text capability not only restricts the potential use of CLIP's text encoder, but also limits the image encoder's ability to capture details and relationships within images. During training, the input text is usually summary text that only contains primary attributes, causing the image encoder to focus on most crucial elements of the image and disregard the other details. Secondly, CLIP demonstrates limitations in accurately modeling the relationships between diverse attributes. Prior research [27] has highlighted that CLIP utilizes a 'bag of concepts' approach for representing various attributes, which can lead to glaring mistakes. For instance, CLIP might assert with high confidence that a lemon is purple when presented with an image containing both a lemon and an eggplant. This significantly undermines its capability to address complex scenarios involving multiple interacting attributes.

Conversely, long texts possess numerous crucial characteristics, evolving fine-grained attributes and indicating the interrelationship between them. Therefore, unlocking the long-text capability of CLIP is of great importance.

A straightforward strategy to achieve this goal is to release the hard restriction on the input text length via interpolating positional embedding, and then fine-tune CLIP with pairs of images and text data involving long descriptions. However, this strategy involves three crucial limitations: (1) the naive interpolation of positional embedding breaks the well-established representation of short text positions; (2) the image features extracted by the image encoder try to cover all the details to align the long text during fine-tuning, regardless of their varying importance. The overwhelming detailed information in image features after fine-tuning disturbs its alignment with short text; (3) the fine-tuning step will shift the embedding space of CLIP features, leading to extra adaptation cost in downstream frameworks, *e.g.*, Stable Diffusion [23]. Experimental results show this simplistic strategy will significantly affect CLIP's short-text capability. The zero-shot classification accuracy on ImageNet will decrease by 13.1%, and T2I R@1 on COCO will also decrease by 14.4%.

To this end, we propose Long-CLIP as a plug-and-play alternative to CLIP. After efficiently fine-tuning CLIP using just an extra 1 million long text-image pairs with only **0.25** hours on **8** GPUs, Long-CLIP seamlessly supports long-text input and retains or even surpasses the zero-shot generalizability of CLIP on various benchmarks. Notably, Long-CLIP aligns the CLIP latent space, readily replacing CLIP without any further adaptation in downstream frameworks. To be specific, Long-CLIP introduces two novel designs to achieve these targets, including (1) a **knowledge-preserved stretching** of positional embedding and (2) a **primary component matching** of CLIP features.

In the **knowledge-preserved stretching** of positional embedding, we first performs empirically study, detailed in Sec. 3.1, revealing that the actual effective text length for CLIP is merely 20 tokens. Inspired by this insightful observation, we first keep the first 20 well-trained positional embedding and interpolate the rest 57 insufficiently trained positional embedding by a larger value. This strat-

egy not only improves the overall length but also minimizes the disruption to the well-established position representation. In **primary component matching** of CLIP features, apart from aligning the fine-grained image feature with a long detailed caption, we also extract the coarse-grained information from the fine-grained image feature and align it with a short summary caption. This requires the model to not only capture different details in an image, but also identify the most important components among them.

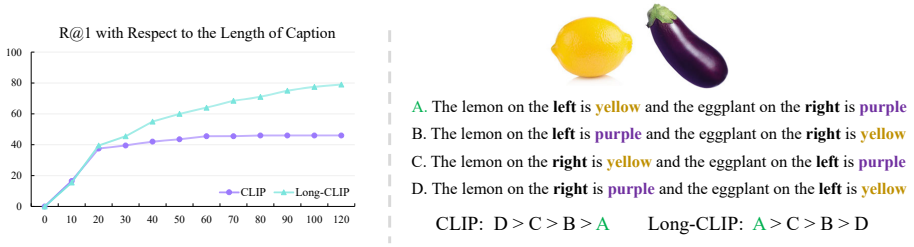
Experiments in Sec. 4 demonstrate that our performance significantly surpasses the original CLIP. We improve the recall rate by 25% on long-text image retrieval tasks, and by 6% on short-text image retrieval tasks. There's no decay on zero-shot classification task. Moreover, as we mostly maintain CLIP's latent space, our model can replace the original pre-trained text-encoder in image generating models like Stable Diffusion [23] to unlock its long-text capability in a **plug-and-play** manner. No additional training is needed.

2 Related Works

Contrastive Language-Image Pre-training (CLIP). CLIP [20] is a multi-modal model based on contrastive learning. It's training data consists of massive text-image pairs: an image and its corresponding textual description. Through contrastive learning, the model aims to learn the matching relationship between open-world text-image pairs. As CLIP has strong zero-shot generalization abilities, it has been successfully utilized in detection [7, 13], segmentation [12, 30], video understanding [15, 29] and most commonly, image generating [4, 6, 21, 28]. Many subsequent works have chosen to use the pre-trained vision or text encoder of CLIP. Thanks to its powerful zero-shot generalization ability, these methods can achieve the ability to process open-world information. DALLE-2 [21], for example, learns a model for converting CLIP text features to image features, and another model to reconstruct CLIP image features to images, thus achieving open-world text-to-image generation. However, some subsequent works [9,32] have recognized that CLIP lacks the capability of extracting fine-grained information. Therefore, they adopt a similar contrastive method to align the input tokens in a complete sentence with a region of the whole image. However, these works still struggle to capture fine-grained information in a long-caption setting.

Positional Embedding Interpolation. To increase the context length, many works [3] leverage positional embedding interpolation. Specifically, it linearly down-scales the input position indices to match the original context window size. This is usually a better strategy than extrapolation. However, this is usually applied to relative positional embedding like RoPE [25].

Vision-Language Dataset. With the development of multimodal capabilities, people are no longer satisfied with fixed-category image datasets like ImageNet [5] and CIFAR10 [11]. Instead, the datasets that contain both images and their corresponding natural language descriptions are required to support open-world applications. The common open-world vision-language datasets include Visual Genome [10], Conceptual-12M [1], SBU [18], COCO [14], LAION-



a) Low effective token length b) Fail in modeling interrelationship

Fig. 2: Two major shortcomings of CLIP. The R@1 increases very slowly when the input length exceeds 20 tokens, indicating that the true effective length of CLIP is even no longer than 20 tokens. Moreover, CLIP fails to extract the interrelationship between different attributes, as it assigns the highest similarity score to a representation that mismatches both color and relative position.

5B [24] and so on. However, these datasets typically only contain short captions. ShareGPT4V [2] dataset, on the other hand, is a large-scale dataset with 1.2 million long captions. It covers rich information including object properties and spatial relationships. The average character number of the caption reaches 826, which is dozens of times longer than previous datasets.

3 Method

3.1 Exploring the Effective Length of CLIP

In theory, an exceptionally efficient model has the capability to extract all the information from a lengthy caption, regardless of its placement at the beginning, middle, or end of the text. If a model fails to extract information beyond a certain length, we can infer that regardless of the maximum input length it can handle, that specific length represents its true effective limit.

Based on this assumption, we conduct an experiment using urban-200 dataset, a long-caption image-text evaluation dataset constructed by us, which will be elaborated in Sec. 4.2. In theory, the R@1 should also gradually increase as we incrementally increase the input caption length if the model can leverage the additional information. However, our observations reveal that once the input length surpasses 20 tokens, the R@1 of CLIP exhibits slow growth. This suggests that the true effective length of CLIP is no more than 20 tokens, as it struggles to leverage the additional information beyond this point. In contrast, our model's performance continues to improve as the input length increases, reaching its peak at the maximum value within the dataset. This indicates that our model is capable of consistently utilizing the newly added information in the captions.

3.2 Knowledge Preserving Stretching

Due to the adoption of a learned absolute positional embedding in the text encoder of CLIP, the length of the input text token imposes a rigid constraint.

To address the difficulty of training a new positional embedding, an interpolation strategy is frequently employed. However, this approach often involves a trade-off between supporting longer input lengths and maintaining the integrity of the well-established position representation. Typically, a common strategy is linearly interpolating the positional embedding using a fixed ratio, denoted as λ_1. The calculation for obtaining the new positional embedding PE^* is as follows:

$$PE^*(pos) = (1-\alpha) \times PE(\lfloor \frac{pos}{\lambda_1} \rfloor) + \alpha \times PE(\lceil \frac{pos}{\lambda_1} \rceil), \quad \alpha = \frac{pos \ \% \ \lambda_1}{\lambda_1} \tag{1}$$

where $PE(pos)$ represents the positional embedding for the pos_{th} position and α is a ratio between 0 and 1, determining whether the interpolated positional embedding for the pos_{th} position is closer to its preceding or following position.

In this specific task, the straightforward strategy of linear interpolation may not be the most suitable approach. This is because the majority of training texts are likely to be considerably shorter than 77 in the original CLIP model. Therefore the lower positions have been well-trained and can effectively indicate absolute positions, while the higher positions have not received sufficient training and can only provide a rough estimation of relative positions. Consequently, the cost of interpolating the lower positions is much higher compared to interpolating the higher ones, as doing so is more likely to disrupt the well-established representation of absolute position.

Therefore, instead of performing full interpolation with a fixed value, we choose to retain the embedding of the top 20 positions, which aligns with the effective length identified in our experiment. As for the remaining 57 positions, we apply interpolation using a larger ratio denoted as λ_2. This process can be mathematically formulated as follows:

$$PE^*(pos) = \begin{cases} PE(pos), \ pos \leq 20 \\ (1-\alpha) \times PE(\lfloor \frac{pos}{\lambda_2} \rfloor) + \alpha \times PE(\lceil \frac{pos}{\lambda_2} \rceil), \quad \alpha = \frac{pos \ \% \ \lambda_2}{\lambda_2}, \ otherwise \end{cases} \tag{2}$$

Experiment in Sec. 4.4 shows that our strategy can improve R@1 on short-text image retrieval by 20% and even support a longer input length.

3.3 Fine-tuning with Primary Component matching

Merely relaxing the strict length constraint on input tokens through positional embedding interpolation is insufficient to fully unlock the long-text capability of the model, as the effective length remains unchanged. Therefore, it is necessary to perform fine-tuning on the CLIP model using long captions.

However, simply fine-tuning the model with long captions will lead to a degradation on the short-text capability. The underlying reason is that a capable

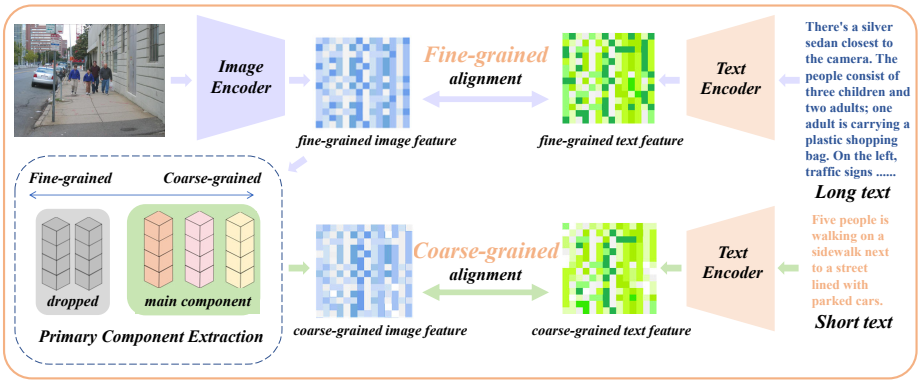


Fig. 3: The pipeline of our training process. We align the fine-grained image feature with the a long detailed caption. Moreover, we also apply primary component extraction to keep the main component and extract the coarse-grained image feature. Then, it is aligned with a short summary caption.

model should not only capture various attributes but also understand their relative relationships and differing importance. Merely fine-tuning the model with long captions may push it to another extreme, where it attempts to encompass all attributes in a single image without distinguishing their respective importance.

To this end, we propose a Primary Component matching strategy in long-text fine-tuning to both unlock the long-text capability and maintain the short-text capability. Apart from aligning the fine-grained feature of an image with its long caption, we extract a coarse-grained image feature that focuses on capturing key attributes. This coarse-grained feature is then aligned with a short summary caption. By doing so, we require the model not only to capture detailed attributes but also to discern and prioritize the importance of different attributes.

This strategy enables the model to learn to cover the necessary detailed attributes while also understanding which attributes hold greater significance. By incorporating this approach into the fine-tuning process, we aim to enhance the model's ability to handle both long and short captions effectively.

Therefore, the key to our task is to find a method for extracting coarse-grained image features from fine-grained image features. This method should be capable of extracting different attributes from the fine-grained image feature and analyzing their importance. To achieve this, we have designed three core modules. The first module is a component-decomposition function $\mathcal{F}$. This function decomposes the feature into several vectors that represent different attributes and also analyzes the importance of each attribute. The second module is a component-filtration function $\mathcal{E}$ which filters out less important attributes based on their analyzed importance. The final module is a component-reconstruction function $\mathcal{F}^{-1}$ which can reconstruct the image feature with different attribute vectors and their corresponding importance.

```
#image_encoder
#text_encoder
#I       - minibatch of input image
#T_long  - minibatch of input long caption
#T_short - minibatch of input short caption
#t       - learned temperature parameter
#alpha   - hyperparameter for balancing coarse-grained
#          and fine-grained alignment
#PCE     - Our primary component extraction algorithm

# extract and align the fine-grained feature of each modality
T_fine = text_encoder(T_long)
I_fine = image_encoder(I)

#compute the logits and the loss for fine-grained alignment
logits_fine = np.dot(I_fine, T_fine.T) * np.exp(t)
labels      = np.arange(n)
loss_fine   = cross_entropy_loss(logits_fine, labels)

#reconstruct and align the coarse-grained feature of each modality
T_coarse = text_encoder(T_short)
I_coarse = PCE(I_fine)

#compute the logits and the loss for fine-grained alignment
logits_coarse = np.dot(I_coarse, T_coarse.T) * np.exp(t)
labels        = np.arange(n)
loss_coarse   = cross_entropy_loss(logits_coarse, labels)

#compute the total loss
loss = loss_fine + alpha * loss_coarse
```

Fig. 4: The Numpy-like pseudo-code of our fine-tuning. We separately align the fine-grained and coarse-grained information in both modalities.

Given a fine-grained image feature I_{fine}, we first extract its different component vectors v_t and the corresponding importance i_t as Eq. 3

$$(v_1, i_1),\ (v_2, i_2),\ ...,\ (v_n, i_n) = \mathcal{F}(I_{fine}) \tag{3}$$

Then, we apply our component-filtration function $\mathcal{E}$ to select the key components and wipe out the others as Eq. 4.

$$(v_{k_1}, i_{k_1}),\ (v_{k_2}, i_{k_2}),\ ...,\ (v_{k_m}, i_{k_m}) = \mathcal{E}[(v_1, i_1),\ (v_2, i_2),\ ...,\ (v_n, i_n)], \quad m \ll n \tag{4}$$

Finally, we apply our component-reconstruction function $\mathcal{F}^{-1}$ to reconstruct the image feature with only the key components and their importance as Eq. 5.

$$I_{coarse} = \mathcal{F}^{-1}[(v_{k_1}, i_{k_1}),\ (v_{k_2}, i_{k_2}),\ ...,\ (v_{k_m}, i_{k_m})] \tag{5}$$

Our total Primary Component Extraction method which extracted the coarse-grained image feature from its fine-grianed one can be represented as Eq. 6

$$I_{coarse} = \mathcal{F}^{-1}(\mathcal{E}(\mathcal{F}(I_{fine})) \tag{6}$$

Thus, the reconstructed image feature only contains the most important coarse-grained information of a given image, and can be aligned with the short summary text properly if the fine-grained image feature can also indicate the true importance of different attributes correctly. Otherwise, if the model can't indicate the true importance, the above process may wipe out the most important attributes while keeping the less important ones, and thus the image feature may fail to match the corresponding short texts.

The implementation of our three core modules can be varied. We utilize a widely-used dimensionality reduction algorithm, Principal Component Analysis method, to achieve the above process. Specifically, the Eigen Value Decomposition (EVD) of the covariance matrix serves as our component-decomposition function $\mathcal{F}$. Then, we select the eigenvectors corresponding to the top 32 largest eigenvalues, which serves as our component-filtration function $\mathcal{E}$. Finally, we reconstruct the image by the linear combination of the selected eigenvectors as component-reconstruction function $\mathcal{F}^{-1}$.

We demonstrate our training pipeline in Fig. 3 and the pseudo-code in Fig. 4.

4 Experiments

4.1 Experiment Setting

Evaluation Dataset. We evaluate our model in three downstream tasks.

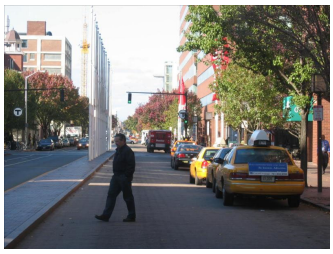
1) zero-shot image classification. The main dataset is *ImageNet-1K* [5]. Moreover, *ImageNet-V2* [22], *ImageNet-O* [8], *CIFAR-10* [11], *CIFAR-100* [11] are also used to analyze classification ability.

2) short-caption image-text retrieval. For traditional short-caption image-text retrieval, we use *COCO2017* [14] and *Flickr30k* [31]. For *COCO2017*, we use the 5k validation set. For *Flickr30k*, to increase difficulty and avoid benchmark saturation, we use the whole 30k dataset instead of the 1k test set.

3) long-caption image-text retrieval. For long-caption image-text retrieval, we use random 1k (image, long text) pairs separated from ShareGPT4V [2]. Moreover, we also collect 200 similar images describing urban scenes and use GPT-4V [17] to generate a long caption, which will be elaborated in Sec. 4.2.

Evaluation Setting. All the experiments in this section share same settings, including the class template in zero-shot classification, all directly truncating the input token if it's longer than restriction, and so on.

Training setting. We use the ShareGPT4V [2] dataset as training dataset, which contains about 1M (long caption, image) pairs. The random 1k data is separated as an evaluation dataset. We fine-tune for 1 epoch with batch size 2048. Detailed hyper-parameter settings will be listed in supplementary material.



A man in a black jacket is crossing a busy city street. The street is filled with cars of various colors, including yellow taxis and red trucks. A traffic light hangs overhead, currently displaying a green signal. The perspective of the photo is from the sidewalk, giving a sense of being part of the city's hustle and bustle. The sky above is clear, suggesting good weather. The street is lined with tall buildings and trees, creating a vibrant cityscape.

A person with dark hair is crossing a street in an urban environment. He's wearing a black jacket, dark-colored pants, and dark shoes, and is carrying a backpack. To his right is a pedestrian signal with "WALK" illuminated above them. On the left, there is a parked red car, and in the background, there are tall buildings, trees, and a sidewalk that extends along a glass building. The sky is clear with few clouds.

Fig. 5: Some (image, long text) pairs of our urban dataset. The images are collected from Visual Genome [10] while the long captions are generated by GPT-4V [17] and are checked and corrected manually. For these two similar image both showcasing a person crossing the road, the key attributes to distinguish them is marked in brown.

4.2 New Evaluation Dataset on Long Caption Text-Image Retrieval

Most current image-text retrieval datasets like COCO [14] and Flickr30k [31] only contain short captions and can only evaluate the model on coarse-grained retrieval ability. To better evaluate long-text fine-grained ability, we collect an urban dataset which contains 200 (image, long caption) pairs.

Specifically, we manually select 200 similar images from Visual Genome dataset [10], all showcasing a busy urban view. Then, we use GPT-4V [17] to generate a long and complete sentence to describe the image, including types, colors and relative locations of the attributes. The model can successfully match the images with the correct captions only if it successfully understands and models the detailed attributes in both modalities.

The average length of our caption reaches about 101 words, which is far longer than the current existing retrieval datasets.

Fig. 5 showcases some examples of our dataset, which is much harder to distinguish than traditional retrieval datasets, posing a higher demand on the capabilities of the model.

After the first submission, we further scaled up **Urban-200** into **Urban-1k**. The dataset has been released at `https://huggingface.co/datasets/BeichenZhang/Urban1k`. Urban-200 is used in the main paper. Detailed results about Urban-1k is shown in supplementary materials.

Table 1: The R@1 of long-caption text-image retrieval on 1k ShareGPT4V [2] validation set and Urban-200 dataset. Best result is in **bold**.

		ShareGPT4V		Urban-200	
		Image-to-Text	Text-to-Image	Image-to-Text	Text-to-Image
B/16	CLIP	78.2	79.6	46.5	46.0
	Direct Fine-tuning	94.1	**93.6**	78.5	78.0
	Long-CLIP(Ours)	**94.6**	93.3	**79.5**	**79.0**
L/14	CLIP	81.8	84.0	47.0	47.0
	Direct Fine-tuning	95.3	95.4	78.0	76.5
	Long-CLIP(Ours)	**95.8**	**95.6**	**81.5**	**81.5**

Table 2: Results of short-caption text-image retrieval on the 5k COCO2017 validation set and the whole **30k** Flickr30K dataset. Best result is in **bold**.

		COCO						Flickr30k					
		Image-to-Text			Text-to-Image			Image-to-Text			Text-to-Image		
		R@1	R@5	R@10	R@1	R@5	R@10	R@1	R@5	R@10	R@1	R@5	R@10
B/16	CLIP	51.8	76.8	84.3	32.7	57.7	68.2	44.1	68.2	77.0	24.7	45.1	54.6
	Direct Fine-tuning	37.4	62.3	72.1	21.8	43.4	54.5	25.7	45.8	55.4	17.9	34.5	43.1
	Long-CLIP(Ours)	**57.6**	**81.1**	**87.8**	**40.4**	**65.8**	**75.2**	**46.8**	**71.4**	**79.8**	**34.1**	**56.3**	**65.7**
L/14	CLIP	56.1	79.5	86.8	35.4	60.1	70.2	48.5	72.6	80.8	28.0	49.3	58.7
	Direct Fine-tuning	37.9	63.1	72.2	23.1	45.1	55.9	26.0	46.3	55.6	17.9	34.9	43.5
	Long-CLIP(Ours)	**62.8**	**85.1**	**91.2**	**46.3**	**70.8**	**79.8**	**53.4**	**77.5**	**85.3**	**41.2**	**64.1**	**72.6**

Table 3: Results of zero-shot image classification in the above five validation sets. Best result is in **bold**.

		ImageNet	ImageNet-O	ImageNet-V2	Cifar10	Cifar100	Average
B/16	CLIP	**68.4**	42.2	**61.9**	**90.8**	67.3	66.12
	Direct Fine-tuning	55.1	31.7	44.8	83.9	59.2	54.94
	Long-CLIP(Ours)	66.8	**42.7**	61.2	90.7	**69.3**	**66.14**
L/14	CLIP	**75.5**	31.9	**69.9**	**95.5**	76.8	**69.92**
	Direct Fine-tuning	58.4	29.2	52.7	92.7	68.7	60.3
	Long-CLIP(Ours)	73.5	**33.7**	67.9	95.3	**78.5**	69.78

4.3 Comparing with CLIP Model

We compare our model with CLIP in the above three different downstream tasks. The interpolation ratio λ_2 in knowledge-stretching is set 4, extending the maximum input length to 248. To evaluate the scalability of our method, we also fine-tune CLIP ViT-L/14 model using our knowledge-preserved stretching of positional embedding and primary component matching strategy as well.

Moreover, we also compare with a direct fine-tuning method discussed in Sec. 1, which simply interpolates each position with a fixed ratio $\lambda_1 = 3$ and only apply (long text, image) pairs in training.

The detailed result is shown in Tab. 1, Tab. 2, and Tab. 3, which shows that our method outperforms our baseline in most aspects. Specifically, for fine-grained task, *i.e.* short or long text-image retrieval, both image and text feature

Table 4: A comparison of whether to use our two strategies. Best result is in **bold**.

KPS	PCM	ImageNet	Cifar100	COCO T2I R@5	Flickr I2T R@5	urban T2I R@1
✗	✗	55.1	59.2	43.4	45.8	78.0
✗	✓	58.8	63.5	46.1	46.0	76.5
✓	✗	65.6	65.9	64.3	70.4	78.0
✓	✓	**66.8**	**69.3**	**65.8**	**71.4**	**79.0**

Table 5: A comparison of different strategies aiming to keep the short-text capability. Best result is in **bold**.

Strategy	ImageNet	Cifar100	COCO T2I R@5	Flickr I2T R@5	Urban-200 T2I R@1
Undistinguished	65.5	67.5	64.6	66.8	71.0
Mixed-length text	66.4	67.8	64.2	68.2	67.5
Bounded encoding	66.8	67.9	65.7	69.3	**80.0**
Ours	**66.8**	**69.3**	**65.8**	**71.4**	79.0

extracted can represent more detailed information of multiple attributes to distinguish from other similar candidates. Therefore, our model can significantly improve performance. For coarse-grained classification tasks, our method doesn't suffer from significant performance degradation like the simple approach thanks to our interpolation and training strategies. This will be discussed in Sec. 4.4

4.4 Ablation Study

Effectiveness of our core component. There are two core component strategy in our method. The first is the knowledge-preserved stretching of positional embedding, which prevents a distortion on well-established positional representation. The second is the primary component matching, aligning both coarse-grained and fine-grained information for a text-image pair. To demonstrate that both strategies can improve the performance of the model, we tested the model's accuracy on different tasks both with and without using these two strategies. For the knowledge-preserved stretching (KPS), we compare it with interpolating each position with a fixed ratio 3. For the primary component matching (PCM), we compare it with simply fine-tuning with (long caption, image) pairs. The result is demonstrated in Tab. 4. As can be seen, removing any of the strategies we proposed will result in a significant loss of short text capabilities.

Different strategy keeping short-text capability. We leverage primary component matching during long-text fine-tuning. And obviously, there exist multiple strategies to help maintain the short-text capability. To further validate the superiority of our method, we design and implement three different strategies.

1) Undistinguished Image Feature. No distinguishing between coarse-grained and fine-grained image features. Align the same image feature I with its corresponding long text T_{long} and short text T_{short} at the same time.

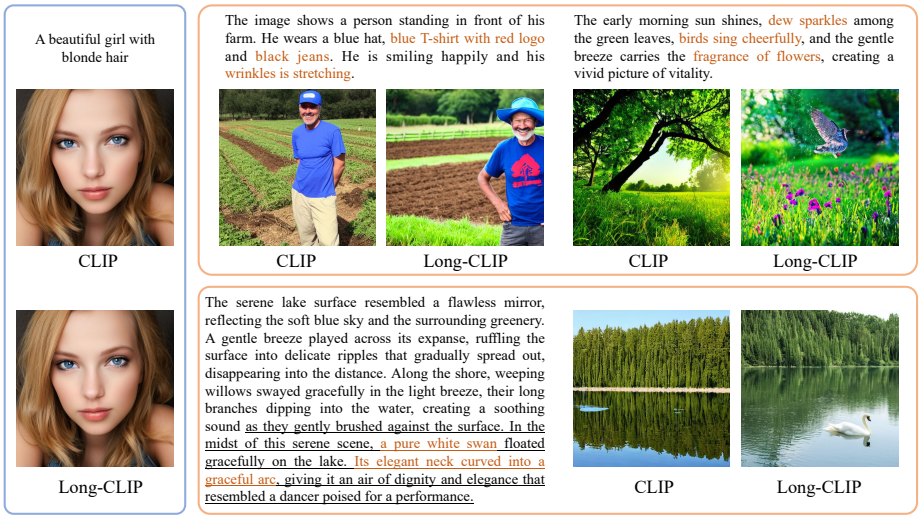


Fig. 6: Our Long-CLIP model can benefit the **text-to-image generation** in a plug-and-play manner in these three aspects. For short simple captions (left), the image generated is nearly the same. For a detailed caption (right top), our model can take in more detailed attributes in it. The caption marked in brown are the detailed attributes missed by the original CLIP, but successfully captured by us. For a long caption (right bottom) exceeding 77 tokens, our model can take in the complete sentence. The underlined caption will be truncated in original CLIP, but can be kept by us.

2) Mixed-length Text. Randomly replace 10% long text with the corresponding short text and align the mixed-length text with the image feature.

3) Bounded Text Encoder. While calculating the long text-image contrastive loss, we also compute the SmoothL1 Loss of the short text features under the current encoder and the frozen original CLIP encoder to ensure that the features of short texts do not have significant deviations.

The detailed result is shown in Tab. 5. Ours turns out to outperform others.

Theoretically, the other three strategies also introduce short text in training and can help maintain the short-text capability. The reason why they are less effective is that they still haven't required the model to identify the importance of different attributes. For example, we can observe a significant performance drop on long-text image retrieval tasks for those methods who try to align the same image feature with both long text and short text, *i.e.* Undistinguished and Mixed-length text. The image encoder may be confused as the short text only contains a very limited number of components, which conflicts with our requirements to include all the components in long-text fine-tuning. Ours, instead, requires the model to include all components and identify the most important component. This is consistent with what a picture really implies.

4.5 Plug-and-Play in Image Generation using CLIP

CLIP is widely used in many downstream task, including segmentation [12, 30], detection [7, 13] and text-to-image generation models like Stable Diffusion [23]. The pre-trained text encoder is often used to extract the text feature of an input to serve as a guidance during image generating. However, just like retrieval tasks, the image generation models using the original CLIP text encoder are still limited by the restriction on 77 tokens and the lack of long-text capability.

Indeed, thanks to the interpolation and matching strategy, we are able to maintain the latent space of the original CLIP model while enhancing its ability to represent fine-grained details in long texts. As a result, our Long-CLIP model exhibits a plug-and-play effect when integrated into existing models.

For example, we can replace the original CLIP text encoder with our Long-CLIP text encoder in Stable-Diffusion-V1-5 [23] with no additional training and keeping other components unchanged. The provided figure (Fig. 6) offers a visual demonstration of the examples, showcasing the enhanced long-text capability of our model. For short and simple text prompts, the images generated by our model closely resemble those generated by the original model. This demonstrates that the short-text capability of the original model is preserved in our approach. However, when presented with longer and more detailed text prompts, our model surpasses the original model by capturing previously overlooked details and including information that exceeds the original 77-token limit.

This plug-and-play nature of our Long-CLIP model allows for seamless integration into various applications and models, provides enhanced capabilities in handling both short and long texts, and supports paragraph-level generation without sacrificing the efficiency and effectiveness of the original CLIP model.

After the first submission, we further use our Long-CLIP model in SDXL [19], which will be discussed in detail in supplementary materials.

5 Conclusion

We have propose Long-CLIP, a strong and flexible CLIP model with long-text capability. Our model can support text inputs of up to 248 tokens, and can better capture the detailed attributes, obtaining a large improvement on retrieval task. Moreover, our model keeps the performance on zero-shot classification and can replace the CLIP encoder in a plug-and-play manner in image generation task.

Potential Limitations. Our Long-CLIP model still has an upper bound of input token length, though largely improved. While some relative positional embedding, like RoPE [25], which doesn't pose a hard limit on input token length although the performance may largely decrease for long text input.

Scaling-up Potential. Due to the scarcity of long text-image pairs, we only leverage 1M (image, long-text) pairs in ShareGPT4V [2]. This gives us imagination on the scaling-up potential when a large amount of data used. As the sufficient long texts can provide complex information, covering world knowledge, object properties, spatial relationships, and even aesthetic evaluations, the model's ability may be significantly improved.

Acknowledgments

This work is partially supported by the National Key R&D Program of China (2022ZD0160201), and Shanghai Artificial Intelligence Laboratory.

References

1. Changpinyo, S., Sharma, P., Ding, N., Soricut, R.: Conceptual 12M: Pushing web-scale image-text pre-training to recognize long-tail visual concepts. In: CVPR (2021)
2. Chen, L., Li, J., Dong, X., Zhang, P., He, C., Wang, J., Zhao, F., Lin, D.: Sharegpt4v: Improving large multi-modal models with better captions. arXiv preprint arXiv:2311.12793 (2023)
3. Chen, S., Wong, S., Chen, L., Tian, Y.: Extending context window of large language models via positional interpolation. CoRR **abs/2306.15595** (2023)
4. Crowson, K., Biderman, S., Kornis, D., Stander, D., Hallahan, E., Castricato, L., Raff, E.: VQGAN-CLIP: open domain image generation and editing with natural language guidance. In: Avidan, S., Brostow, G.J., Cissé, M., Farinella, G.M., Hassner, T. (eds.) ECCV. Lecture Notes in Computer Science, vol. 13697, pp. 88–105. Springer (2022)
5. Deng, J., Dong, W., Socher, R., Li, L., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: CVPR. pp. 248–255 (2009)
6. Frans, K., Soros, L.B., Witkowski, O.: Clipdraw: Exploring text-to-drawing synthesis through language-image encoders. In: Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., Oh, A. (eds.) NeurIPS (2022)
7. Gu, X., Lin, T., Kuo, W., Cui, Y.: Open-vocabulary object detection via vision and language knowledge distillation. In: ICLR. OpenReview.net (2022)
8. Hendrycks, D., Zhao, K., Basart, S., Steinhardt, J., Song, D.: Natural adversarial examples. In: CVPR. pp. 15262–15271. Computer Vision Foundation / IEEE (2021)
9. Kim, D., Angelova, A., Kuo, W.: Region-aware pretraining for open-vocabulary object detection with vision transformers. In: CVPR. pp. 11144–11154. IEEE (2023)
10. Krishna, R., Zhu, Y., Groth, O., Johnson, J., Hata, K., Kravitz, J., Chen, S., Kalantidis, Y., Li, L., Shamma, D.A., Bernstein, M.S., Fei-Fei, L.: Visual genome: Connecting language and vision using crowdsourced dense image annotations. Int. J. Comput. Vis. **123**(1), 32–73 (2017)
11. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)
12. Li, B., Weinberger, K.Q., Belongie, S.J., Koltun, V., Ranftl, R.: Language-driven semantic segmentation. In: ICLR. OpenReview.net (2022)
13. Li, L.H., Zhang, P., Zhang, H., Yang, J., Li, C., Zhong, Y., Wang, L., Yuan, L., Zhang, L., Hwang, J., Chang, K., Gao, J.: Grounded language-image pre-training. CoRR **abs/2112.03857** (2021)
14. Lin, T., Maire, M., Belongie, S.J., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft COCO: common objects in context. In: Fleet, D.J., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV. Lecture Notes in Computer Science, vol. 8693, pp. 740–755. Springer (2014)
15. Luo, H., Ji, L., Zhong, M., Chen, Y., Lei, W., Duan, N., Li, T.: Clip4clip: An empirical study of CLIP for end to end video clip retrieval and captioning. Neurocomputing **508**, 293–304 (2022)

16. Luo, Z., Zhao, P., Xu, C., Geng, X., Shen, T., Tao, C., Ma, J., Lin, Q., Jiang, D.: Lexlip: Lexicon-bottlenecked language-image pre-training for large-scale image-text sparse retrieval. In: IEEE/CVF International Conference on Computer Vision, ICCV 2023, Paris, France, October 1-6, 2023. pp. 11172–11183. IEEE (2023)
17. OpenAI: GPT-4 technical report. CoRR **abs/2303.08774** (2023)
18. Ordonez, V., Kulkarni, G., Berg, T.L.: Im2text: Describing images using 1 million captioned photographs. In: Shawe-Taylor, J., Zemel, R.S., Bartlett, P.L., Pereira, F.C.N., Weinberger, K.Q. (eds.) NeruIPS. pp. 1143–1151 (2011)
19. Podell, D., English, Z., Lacey, K., Blattmann, A., Dockhorn, T., Müller, J., Penna, J., Rombach, R.: SDXL: improving latent diffusion models for high-resolution image synthesis. CoRR **abs/2307.01952** (2023)
20. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., Sutskever, I.: Learning transferable visual models from natural language supervision. In: Meila, M., Zhang, T. (eds.) ICML. Proceedings of Machine Learning Research, vol. 139, pp. 8748–8763. PMLR (2021)
21. Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., Chen, M.: Hierarchical text-conditional image generation with CLIP latents. CoRR **abs/2204.06125** (2022)
22. Recht, B., Roelofs, R., Schmidt, L., Shankar, V.: Do imagenet classifiers generalize to imagenet? In: Chaudhuri, K., Salakhutdinov, R. (eds.) ICML. Proceedings of Machine Learning Research, vol. 97, pp. 5389–5400. PMLR (2019)
23. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models (2021)
24. Schuhmann, C., Beaumont, R., Vencu, R., Gordon, C., Wightman, R., Cherti, M., Coombes, T., Katta, A., Mullis, C., Wortsman, M., Schramowski, P., Kundurthy, S., Crowson, K., Schmidt, L., Kaczmarczyk, R., Jitsev, J.: LAION-5B: an open large-scale dataset for training next generation image-text models. In: Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., Oh, A. (eds.) NeruIPS (2022)
25. Su, J., Ahmed, M.H.M., Lu, Y., Pan, S., Bo, W., Liu, Y.: Roformer: Enhanced transformer with rotary position embedding. Neurocomputing **568**, 127063 (2024)
26. Sun, Z., Fang, Y., Wu, T., Zhang, P., Zang, Y., Kong, S., Xiong, Y., Lin, D., Wang, J.: Alpha-clip: A CLIP model focusing on wherever you want. CoRR **abs/2312.03818** (2023)
27. Tang, Y., Yamada, Y., Zhang, Y., Yildirim, I.: When are lemons purple? the concept association bias of vision-language models. In: EMNLP. pp. 14333–14348. Association for Computational Linguistics (2023)
28. Vinker, Y., Pajouheshgar, E., Bo, J.Y., Bachmann, R.C., Bermano, A.H., Cohen-Or, D., Zamir, A., Shamir, A.: Clipasso: semantically-aware object sketching. ACM Trans. Graph. **41**(4), 86:1–86:11 (2022)
29. Xu, H., Ghosh, G., Huang, P., Okhonko, D., Aghajanyan, A., Metze, F., Zettlemoyer, L., Feichtenhofer, C.: Videoclip: Contrastive pre-training for zero-shot video-text understanding. In: Moens, M., Huang, X., Specia, L., Yih, S.W. (eds.) EMNLP. pp. 6787–6800. Association for Computational Linguistics (2021)
30. Xu, J., Mello, S.D., Liu, S., Byeon, W., Breuel, T.M., Kautz, J., Wang, X.: Groupvit: Semantic segmentation emerges from text supervision. In: CVPR. pp. 18113–18123. IEEE (2022)
31. Young, P., Lai, A., Hodosh, M., Hockenmaier, J.: From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. Trans. Assoc. Comput. Linguistics **2**, 67–78 (2014)
32. Zeng, Y., Zhang, X., Li, H.: Multi-grained vision language pre-training: Aligning texts with visual concepts. arXiv preprint arXiv:2111.08276 (2021)