Position-guided Text Prompt for Vision-Language Pre-training

Jinpeng Wang[2] Pan Zhou[1*] Mike Zheng Shou[2*] Shuicheng Yan[1]
[1]Sea AI Lab [2]Show Lab, National University of Singapore

Code: https://github.com/sail-sg/ptp

Abstract

Vision-Language Pre-Training (VLP) has shown promising capabilities to align image and text pairs, facilitating a broad variety of cross-modal learning tasks. However, we observe that VLP models often lack the visual grounding/localization capability which is critical for many downstream tasks such as visual reasoning. In this work, we propose a novel Position-guided Text Prompt (PTP) paradigm to enhance the visual grounding ability of cross-modal models trained with VLP. Specifically, in the VLP phase, PTP divides the image into $N \times N$ blocks, and identifies the objects in each block through the widely used object detector in VLP. It then reformulates the visual grounding task into a fill-in-the-blank problem given a PTP by encouraging the model to predict the objects in the given blocks or regress the blocks of a given object, e.g. filling "[P]" or "[O]" in a PTP "The block [P] has a [O]". This mechanism improves the visual grounding capability of VLP models and thus helps them better handle various downstream tasks. By introducing PTP into several state-of-the-art VLP frameworks, we observe consistently significant improvements across representative cross-modal learning model architectures and several benchmarks, e.g. zero-shot Flickr30K Retrieval (+4.8 in average recall@1) for ViLT [16] baseline, and COCO Captioning (+5.3 in CIDEr) for SOTA BLIP [19] baseline. Moreover, PTP achieves comparable results with object-detector based methods [8, 23, 45], and much faster inference speed since PTP discards its object detector for inference while the later cannot.

1. Introduction

The vision-and-language pre-training (VLP) models like CLIP [31], ALIGN [14] and CoCa [42] have greatly advanced the state-of-the-art performance of many cross-modal learning tasks, e.g., visual question answering [4], reasoning [35], and image captioning [1, 7]. Typically, a generic cross-modal model is first pre-trained on large-scale image-caption data in a self-supervised fashion to see sufficient data for better generalization ability, and then fine-tuned on downstream tasks for adaptation. With remarkable effectiveness, this pre-training-then-fine-tuning paradigm of VLP models has dominated the multi-modality field.

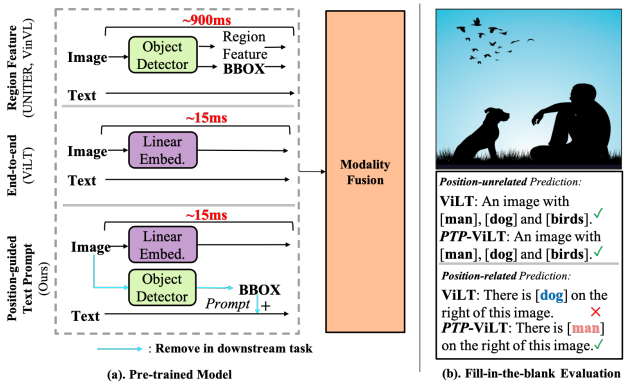

Figure 1. Comparison of three VLP learning frameworks and their performance. (a) compares region feature based VLP (RF-VLP), end-to-end VLP (E2E-VLP), and our position-guided text prompt based VLP (PTP-VLP). Our PTP-VLP only needs about 15ms for inference which is the same as E2E-VLP but is much faster than RF-VLP. (b) On position-aware questions widely occurred in many downstream tasks, with masked text and image input, RF-VLP and PTP-VLP can well predict objects, while E2E-VLP cannot pinpoint the position information of the object in the image.

In VLP, visual grounding is critical for many tasks as observed in previous research [3, 40]. To model the position information, traditional VLP models [3, 23, 45] (the top of Fig. 1 (a)) employ a faster-rcnn [33] pre-trained on the 1600 classes Visual Genome [17] to extract salient region features and bounding boxes. Then these models use both the bounding box and object feature as input. In this way, these models not only learn what objects are contained in the salient region and where are these objects. However, when using region features as input, the model pays attention to the items inside the bounding boxes and ignores the contextual data outside of them [13]. More seriously, on downstream task, these methods still need to use detectors to extract objects, giving very slow inference speed.

To get rid of region feature for higher efficiency, recent

*Corresponding authors.

works [13, 16] (the middle of Fig. 1 (a)) adopt raw-pixel image as input instead of region features, and train the model with Image Text Matching [8] and Masked Language Modeling [10] loss end-to-end. Despite their faster speed, these models cannot well learn the object positions and also their relations. As shown in Fig. 1 (b), we observe that a well-trained ViLT model [16] well know what objects are in an image. But this model does not learn the object positions accurately. For example, it wrongly predicts "*the dog is on the right of this image*". However, during fine-tuning, downstream tasks actually require the object position information to comprehensively understand the image. Such a gap largely impairs the performance on downstream tasks.

In this work, we aims to ease the position missing problem for these end-to-end models, and keep fast inference time for downstream tasks at the same time. Inspired by the recently prompt learning methods [15, 25, 32, 41], we propose a novel and effective **Position-guided Text Prompt (*PTP*)** paradigm (the bottom of Fig. 1 (a)) for cross-modality model pre-training. The key insight is that by adding position-based co-referential markers in both image and text, visual grounding can be reformulated into a fill-in-the-blank problem, maximally simplify the learning of object information. *PTP* grounds language expressions in images through two components: 1) block tag generation, dividing images into $N \times N$ blocks and identifying objects, and 2) text prompt generation, placing query text into a position-based template.

By bringing the position information into pre-training, our *PTP* enables strong visual grounding capabilities of VLP models. At the same time, as we do not used object detector for downstream tasks, we keep fast inference time. Experimental results show that our method outperforms their counterparts by a large margin especially for zero-shot setting. For example, our *PTP*-BLIP achieves 3.4% absolute accuracy gain over CoCa [42] in zero-shot retrieval Recall@1 on coco dataset with much less training data (4M vs. 3B) and a much smaller model (220M vs. 2.1B). In addition to the zero-shot task, we show that *PTP* can achieve strong performance for object position guided visual reasoning and the other common VLP tasks such as visual question answering, and image captioning.

2. Related Work

2.1. Vision-language Pre-training Models

Existing VLP models can be roughly grouped into three categories according to their architectures: one-stream models, dual-stream models and dual-stream + fusion encoder model. All three architectures are introduced below:

1) One-stream Model (e.g., UNITER [8], ViLT [16]) in Fig. 2 (a) operates on a concatenation of image and text inputs. *2) Dual-stream Model* (e.g., CLIP [31]) in Fig. 2 (b) uses separate but equally expensive transformer encoders

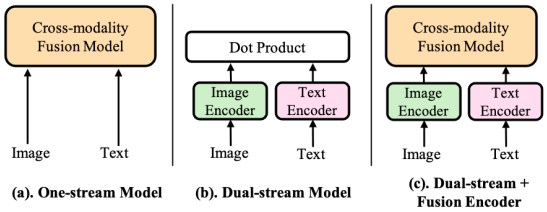

Figure 2. **Three widely-used categories of vision-and-language models.** The main difference is where to perform cross-modality information fusion. One-stream fuse at early stage and dual-stream fuse at late stage, while the last type fuse at middle stage.

for each modality. The two modalities are not concatenated at the input level and interaction between the pooled image vector and text vector at shallow layer. *3) Dual-stream with Fusion Model* (e.g., BLIP [19]) Fig. 2 (c) is a combination of one-stream and dual-stream model.

In this work, without loss of generality, we focus on prompting all these three kinds of VLP models due to their prevalence and adaptability to different downstream tasks.

2.2. Prompt Learning for Computer Vision

Prompt learning is originally designed for probing knowledge in pre-trained language models to specific downstream tasks [25, 32]. Recent years have seen a rise in the study of prompt tuning on vision taks, e.g. multi-modal learning and image understanding. The pioneer Color Prompt [41] adds color prompt on image and text color description for visual grounding. Most related to our work is Multi-modality Prompt [15] which presents multi-modality prompt tuning for VLPT models, achieving promising results on some vision-language tasks.

However, these efforts, like earlier NLP research, concentrate on prompt engineering in fine-tuning while leaving the pre-training phase unaffected. The goal of using the prompt design in this work, in contrast, is to provide the model the ability to understand semantic concepts at a finer level while it is still in the pre-training stage.

2.3. Learn Position Information in VLP

The grounding ability has shown to be essential for multiple cross-modality tasks [21, 26]. To introduce this ability into VLP models, bottom-up and top-down [3] and its follow-up works [8, 23] concatenate region feature and bounding box vector together. But object extraction is time-consuming in inference for downstream task. Recently, some works [21, 26, 44] propose train the VLP models with additional object localization loss or word patch alignment loss which, however, are hard to extend because they are specifically designed for particular frameworks. In contrast, we aim to propose a general framework for learning position information. To this end, we propose a simple text prompt that can be plug into existing frameworks easily.

3. Position-guided Text Prompt

In this section, we first elaborate on our proposed Position-guided Text Prompt paradigm (*PTP* for short). Then we introduce how to incorporate it with current vision-language pre-training (VLP) frameworks for boosting their visual grounding capabilities by taking the classical and popular VILT [16], CLIP [31] and BLIP [19] as examples.

3.1. *PTP* Paradigm

To enhance the visual grounding ability of cross-modal models trained by VLP, we propose a novel and effective Position-guided Text Prompt (*PTP*) that helps a cross-modal model perceive objects, and also align these objects with pertinent text. *PTP* differs from the conventional vision language alignment methods, e.g. [3, 8, 23, 45], that concatenate object feature and bounding box together as input to learn the alignment between objects and pertinent text, and thus paves an alternative way which indeed enjoys some advantages as shown and discussed in Sec. 3.2. As illustrated in Fig. 3, *PTP* has two steps: 1) block tag generation which divides an input image into several blocks and also identifies the objects in each block; and 2) text prompt generation that reformulates the visual grounding task into a fill-in-the-blank problem according to the object position information in step 1). Based on these steps, one can easily plug PTP into a VLP model by solving fill-in-the-blank problem in *PTP*. We will introduce these two steps below.

3.1.1 Block Tag Generation

As shown in Fig. 3, for each image-text pair in the training phase, we evenly divide the input image into $N \times N$ blocks. Then we identify the object in each block by one of the following two ways:

(1) Object Detector. We first adopt a strong Faster-rcnn [33] used in VinVL [45] to extract all objects for each image. This Faster-rcnn version is based on ResNeXt152 and is trained on 1600-classes Visual Genome [17]. Then we select top-K objects denoted by $\mathcal{O} = \{o_i\}_{i=1}^{K}$ with highest prediction confidence, where $o_i = (z_i, q_i)$ denotes an object with 4-dimensional region position vector z and object category q. For each block, we select the objects whose region center are in that block. At last, the final block tag for this block is q of these selected objects. In this work, we generate object tag with object detector as default.

(2) CLIP Model. Instead of heavy object detector, some recent works [46, 47] also try to generate region supervision based on CLIP [31] because of its efficiency and effectiveness. Inspired by these works, *PTP* can also generate block-wise object supervision via CLIP (ViT-B) model [1]. First, we extract M (3000 in default) key words/phrases that are most frequent on the whole text corpus [2]. These key words/phrases are regarded as our vocabulary V. Then we extract the text feature $e_i, i \in [1, \dots, M]$ of all these M key words/phrases embedding via CLIP text encoder.

[1]https://huggingface.co/openai/clip-vit-base-patch16

[2]Extract key word/phrase with NLTK (/https://github.com/nltk/nltk)

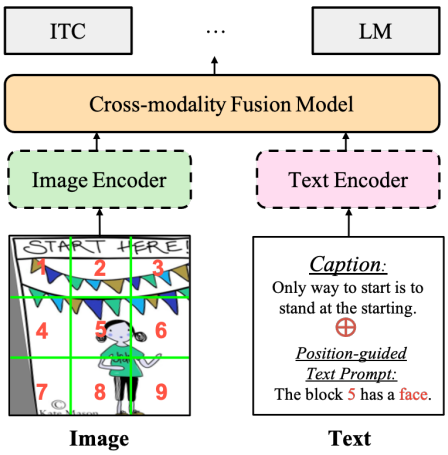

Figure 3. **Overall framework.** Any pre-training framework (one-stream, dual-stream, dual-stream+fusion encoder in Fig. 2) and most objectives can be integrated with our *PTP*. Dashed line indicates that the model may not exist. We remove the text prompt for the downstream task and evaluate the model as usual.

Additionally, we take the image embedding h from each block and compute the similarity across every text feature. The keyword/phrase with the highest similarity score is selected as the final object tag for this particular block. Formally, the index of object tag per block is computed as

$$I = \text{argmax}_{y \in [1,\dots,M]} \left(\frac{\exp(h^T e_y)}{\sum_{w \in V} \exp(h^T e_w)} \right), \quad (1)$$

where h is the visual feature embedding of selected block. Comparing with object detector, the CLIP model have two advantages. Firstly, as opposed to pre-defined object categories, more diverse object tags are produced. Secondly, the generation of block tag is much faster than object detector, e.g. 40× faster than Faster-RCNN (ResNeXt152) model. Please refer to Sec. 4.3 for comparison.

3.1.2 Text Prompt Generation

For the input image of each training pair, Sec. 3.1.1 already generate the object tags and positions which allows us to design a simple text prompt as follows:

"*The block [P] has a [O].*"

where $P \in \{1, \cdots, N^2\}$ denotes the index of selected block and is used to denote the object position; O denotes the object tag generated for the block P. Note, we explore more prompt design choices in Section 4.3. For a certain P, we may have various options for O because the block may contain multiple objects. For such situation, we select one O at random for each time. Thus, each sentence in *PTP* combines fine-grained object position and language, offering a novel method to align objects and relevant text.

3.2. Pre-training with *PTP*

In this work, we integrate our *PTP* into mainstream VLP frameworks, leading to *PTP*-ViLT [16], *PTP*-CLIP [31] and *PTP*-BLIP [19]. Following receipt of the *PTP*, we have two options for training these models:

Integrate into existing tasks. The simplest method for using text prompt is to change the text input. As shown in Fig. 3, the prompted text and original caption were simply padded together. Formally, the input caption x of our method is represented as:

$$x = [w, q], \tag{2}$$

where w is text and q is our generated text prompt. We train VLP models end-to-end using conventional objectives. Following [16, 19, 31], *PTP*-BLIP employs LM loss, ITM, and ITC loss; *PTP*-ViLT uses ITM and MLM loss; and *PTP*-CLIP solely applies ITC loss. This method is the default for all experiments due to its strong performance.

As a new pretext task. Alternatively, we explore the position prediction as an additional language modeling task. Formally, if D is the pretraining data and $y_1, \ldots, y_T$ is a training token sequence of our generated text prompt q, then at the timestep t, we devise our model to predict a probability distribution $p(t) = p(*|y_1, \ldots, y_{t-1})$. Then we regressively try to maximize the probability of being the correct token. The object prediction loss is computed as follow:

$$\mathcal{L}_{\mathrm{PTP}}(\theta) = -\mathbb{E}_{\mathbf{y}\sim D}\left[\sum_{t=1}^{T} \log P_\theta\left(\mathbf{y}_t \mid \mathbf{y}_{<t}\right)\right], \tag{3}$$

where θ is the trainable parameters of the model. In this way, the model is asked to predict *which block P has objects* and *what object O is in this block*.

Discussion. Notably, our method does not need to modify the base network and can be applied to any VLP models without bells and whistles. The model is designed to learn position information from raw-pixel image. Note that only during the pre-training stage, we would require the object's position information; yet on downstream tasks, we evaluate model in normal end-to-end ways without object information to get rid of the heavy object feature extraction.

4. Experiments

In this section, we empirically evaluate *PTP* on multiple downstream tasks and present a comprehensive study.

4.1. Experimental Settings

We first describe the pre-training experimental conditions, including the datasets, training configurations, evaluation procedures, and baseline models used in our studies.

Datasets. As in earlier studies [23, 45], we begin by using a 4M setup made up of four popular pre-training datasets (COCO [24], VG [17], SBU [28] and CC3M [34]). Following recent work [19], we also explore 14M setting, which includes additional CC12M [6] (actually only 10M image urls available) dataset besides 4M datasets. We follow OSCAR [23] to prepare the train corpus for *PTP*.

Training Settings. Our models are implemented in PyTorch [29] and pre-trained on 8 NVIDIA A100 GPUs. We adopt the optimizer and training setting from baseline works for fair comparison. We use RandAugment [9], excluding color inversion, as color information is crucial. Bounding box augmentation follows image affine transformations. During pre-training, random 224×224 image crops are used, increasing to 384×384 for finetuning.

Baselines. We evaluate three variants of pre-training frameworks, including one-stream ViLT [16], dual-encoder CLIP [31], and fusion-encoder BLIP [19], for their superior performance. For fair comparisons, we adopt the ViT-B/16 [11] as base vision encoder and use same dataset.

4.2. Main Results

In this section, we integrated our *PTP* into existing networks and compare to existing VLP methods on a wide range of vision-language downstream tasks. Then we introduce each task and finetuning strategy. More details can be found in the supplementary material.

4.2.1 Image-Text Retrieval

We evaluate *PTP* for both image-to-text retrieval (TR) and text-to-image retrieval (IR) on COCO and Flickr30K benchmarks. For *PTP*-BLIP, following original implementation, we adopt an additional re-ranking strategy.

We first report zero-shot retrieval result on both image-to-text and text-to-image setting in Tab. 1. We find *PTP* significantly improves baselines on all metrics. For example, for ViLT [16] baseline, *PTP* leads to 13.8 % absolute improvement (from 41.3 % to 55.1 %) over Recall@1 of image to text retrieval on MSCOCO. In addition, based on strong BLIP [19], our *PTP*-BLIP even outperforms CoCa [42] on most recalls of MSCOCO with much less data.

A summary comparison about fine-tuned setting between different models appears in Tab. 2, from which we observe that: (1) *PTP* outperforms the BLIP and ViLT baselines by a large margin in both datasets. For example, *PTP*-ViLT achieves an impressive 5.3% improvement on R@1 of TR in MSCOCO. (2) With strong BLIP as baseline, *PTP*-BLIP leads to state-of-the-art performance at same scale. Notice that the training cost remains the same BLIP baseline, because we train *PTP* with the same settings as the baseline and do not increase the maximum input text token. We can even reduce the gap between 4M setting and ALBEF [20] (14M data), with similar framework.

From all these results above, we point out UNITER [8],

Table 1. **Results of zero-shot image-text retrieval on Flickr30K and MSCOCO datasets.** We gray out the methods that train on much larger corpus or use much larger models. † means the model implemented by ourself and trained on same dataset since the original datasets is not accessible or not trained on these splits. The Avg is the mean of all image-to-text recalls and text-to-image recalls.

Method	#Images	Parameters	MSCOCO (5K test set)							Flickr30K (1K test set)						
			Image → Text			Text → Image				Image → Text			Text → Image			
			R@1	R@5	R@10	R@1	R@5	R@10	Avg	R@1	R@5	R@10	R@1	R@5	R@10	Avg
Unicoder-VL [18]	4M	170M	–	–	–	–	–	–		64.3	85.8	92.3	48.4	76.0	85.2	75.3
ImageBERT [30]	4M	170M	44.0	71.2	80.4	32.3	59.0	70.2	59.5	70.7	90.2	94.0	54.3	79.6	87.5	79.4
ViLT [16]	4M	87M	41.3	79.9	87.9	37.3	67.4	79.0	65.5	69.7	91.0	96.0	53.4	80.7	88.8	79.9
***PTP*-ViLT (ours)**	4M	87M	55.1	82.3	89.1	43.5	70.2	81.2	$70.2_{+4.7}$	74.5	93.7	96.5	60.3	85.5	90.4	$83.5_{+3.6}$
BLIP † [19]	4M	220M	57.4	81.1	88.7	41.4	66.0	75.3	68.3	76.0	92.8	96.1	58.4	80.0	86.7	81.7
***PTP*-BLIP (ours)**	4M	220M	**72.3**	**91.8**	**95.7**	**49.5**	**75.9**	**84.2**	$\mathbf{77.3}_{+9.0}$	**86.4**	**97.6**	**98.9**	**67.0**	**87.6**	**92.6**	$\mathbf{88.4}_{+6.7}$
***PTP*-BLIP (ours)**	14M	220M	73.2	92.4	96.1	53.6	79.2	87.1	78.6	87.1	98.4	99.3	73.1	91.0	94.8	90.3
CLIP [31]	300M	173M	58.4	81.5	88.1	37.8	62.4	72.2	66.7	88.0	98.7	99.4	68.7	90.6	95.2	90.1
ALIGN [14]	1.8B	820M	58.6	83.0	89.7	45.6	69.8	78.6	70.9	88.6	98.7	99.7	75.7	93.8	96.8	92.2
FILIP [40]	340M	787M	61.3	84.3	90.4	45.9	70.6	79.3	72.0	89.8	99.2	99.8	75.0	93.4	96.3	92.3
Flamingo [2]	2.1B	80B	65.9	87.3	92.9	48.0	73.3	82.1	74.9	89.3	98.8	99.7	79.5	95.3	97.9	93.4
CoCa [24]	3B	2.1B	66.3	86.2	91.8	51.2	74.2	82.0	75.3	92.5	99.5	99.9	80.4	95.7	97.7	94.3

Table 2. **Finetuning results of image-to-text retrieval and text-to-image retrieval on COCO and Flickr30K.** Notice that UNITER [8], OSCAR [23] and VinVL [45] all use bounding box and object feature.

Method	#Images	Parameters	MSCOCO (5K test set)							Flickr30K (1K test set)						
			Image → Text			Text → Image				Image → Text			Text → Image			
			R@1	R@5	R@10	R@1	R@5	R@10	Avg	R@1	R@5	R@10	R@1	R@5	R@10	Avg
UNITER [8]	4M	155M	65.7	88.6	93.8	52.9	79.9	88.0	78.2	87.3	98.0	99.2	75.6	94.1	96.8	91.8
OSCAR [23]	4M	155M	70.0	91.1	95.5	54.0	80.8	88.5	–	–	–	–	–	–	–	
VinVL [45]	4M	157M	74.6	92.6	96.3	58.1	83.2	90.1	82.5	–	–	–	–	–	–	–
ViLT [16]	4M	87M	61.8	86.2	92.6	41.3	72.0	82.5	72.7	81.4	95.6	97.6	61.9	86.8	92.8	86.0
***PTP*-ViLT (ours)**	4M	87M	67.1	90.5	94.3	45.3	79.1	88.4	$77.5_{+4.8}$	85.2	96.9	98.5	68.8	91.4	95.3	$89.4_{+3.4}$
BLIP † [19]	4M	220M	75.2	93.3	96.3	57.4	82.1	89.5	82.3	94.0	99.1	99.7	82.5	96.4	98.2	95.0
***PTP*-BLIP (ours)**	4M	220M	**83.7**	**97.0**	**98.7**	**68.1**	**89.4**	**94.2**	$\mathbf{88.5}_{+6.2}$	**96.1**	**99.8**	**100.0**	**84.2**	**96.6**	**98.6**	$\mathbf{95.9}_{+0.9}$
ALBEF [20]	14M	210M	77.6	94.3	97.2	60.7	84.3	90.5	84.1	95.9	99.8	100.0	85.6	97.5	98.9	96.3
BLIP [19]	14M	220M	80.6	95.2	97.6	63.1	85.3	91.1	85.5	96.6	99.8	100.0	87.2	97.5	98.8	96.7
***PTP*-BLIP (ours)**	14M	220M	**84.2**	**97.3**	**98.8**	**68.8**	**89.5**	**94.2**	**88.8**	**97.0**	**99.9**	**100.0**	**87.7**	**98.2**	**99.3**	$\mathbf{97.0}_{+0.3}$
ALIGN [14]	1.8B	820M	77.0	93.5	96.9	59.9	83.3	89.8	83.4	95.3	99.8	100.0	84.9	97.4	98.6	96.0
FILIP [40]	340M	787M	78.9	94.4	97.4	61.2	84.3	90.6	84.5	96.6	100.0	100.0	87.1	97.7	99.1	96.8
Florence [43]	900M	893M	81.8	95.2	–	63.2	85.7	–	–	97.2	99.9	–	87.9	98.1	–	–

OSCAR [23], VinVL [45], ImageBERT [30] all use faster-rcnn as we used. However, our *PTP* leads to much better results than these related works. Besides, we only use object detector in pre-training stage. This indicates *object detector is not the secret for success and how to leverage the position information is essential important for VLP models*.

4.2.2 Image Captioning

This task asks the model to describe the input image. We consider two datasets for image captioning: No-Caps [1] and COCO [24], both evaluated using the model finetuned on COCO with the LM loss. Like BLIP, captions start with "a picture of" for slightly better results. We avoid COCO dataset pre-training to prevent information leakage. For No-Caps, we adopt a zero-shot setting, as in BLIP, by evaluating with the model trained on the COCO dataset.

As shown in Tab. 3, related works utilizing a comparable quantity of pre-training data perform significantly worse than *PTP*-BLIP. The results of our method are closed to the VinVL [45] with fewer training samples and smaller image. Finally, with 14M setting, our method leads to close result with LEMON, which trained on billions data and requires two times higher resolution image.

4.2.3 Visual Question Answering

VQA [4] requires the model to predict an answer given an image and a question. For *PTP*-ViLT, we formulating VQA as a multi-answer classification task. For *PTP*-BLIP, we follow [19, 20] and consider it as an answer generation task that allows open-vocabulary VQA for better result.

The results are reported in Tab. 4. Compared to ViLT baseline, *PTP* brings 1.8% gains on both dev split. With 14M setting, *PTP*-BLIP achieves better performance than SimVLM [39], which uses 1.8B training samples and a ViT-Large based vision backbone.

Table 3. **Comparison with state-of-the-art image captioning methods on NoCaps and COCO Caption.** C: CIDEr, S: SPICE, B@4: BLEU@4. Notice that VinVL‡ and LEMON‡ require high resolution (800×1333) input images.

Method	#Images	Parameters	NoCaps validation								COCO Caption			
			in-domain		near-domain		out-domain		Overall		Karpathy test			
			CIDEr	SPICE	CIDEr	SPICE	CIDEr	SPICE	CIDEr	SPICE	B@4	METEOR	SPICE	CIDEr
OSCAR [23]	4M	155M	79.6	12.3	66.1	11.5	45.3	9.7	80.9	11.3	37.4	30.7	23.5	127.8
VinVL‡ [45]	5.7M	347M	103.1	14.2	96.1	13.8	88.3	12.1	95.5	13.5	38.5	30.4	23.4	130.8
BLIP † [19]	4M	220M	106.5	14.4	99.3	13.6	95.6	13.0	98.8	14.2	37.0	–	–	122.6
***PTP*-BLIP (ours)**	4M	220M	**108.3**	**14.9**	**105.0**	**14.2**	**105.6**	**14.2**	**106.0**	**14.7**	**42.5**	**32.3**	**25.4**	**145.2**
Enc-Dec [6]	15M	–	92.6	12.5	88.3	12.1	94.5	11.9	90.2	12.1	–	–	–	110.9
BLIP [19]	14M	220M	111.3	15.1	104.5	14.4	102.4	13.7	105.1	14.4	38.6	–	–	129.7
***PTP*-BLIP (ours)**	14M	220M	**112.8**	**15.2**	**107.3**	**14.9**	**108.1**	**14.3**	**106.3**	**14.7**	**42.7**	**32.4**	**25.4**	**145.3**
SimVLM$_{huge}$ [39]	1.8B	1.2B	113.7	–	110.9	–	115.2	–	112.2	–	40.6	33.7	25.4	143.3
LEMON$_{huge}$‡ [12]	200M	675M	118.0	15.4	116.3	15.1	120.2	14.5	117.3	15.0	42.6	–	–	145.5
Beit-3 [38]	35M+	1.9B	–	–	–	–	–	–	–	–	44.1	32.4	25.4	147.6

Table 4. **Comparison with state-of-the-art methods on VQA and NLVR^2.** Para. is short for parameters. Notice that VinVL [45] uses larger vision backbone and object feature from faster-rcnn. ALBEF [20] performs an extra pre-training step for NLVR^2 and BeIT-3 [38] uses additional 160GB text corpus.

Method	#Images	Para.	VQA		NLVR^2	
			test-dev	test-std	dev	test-P
UNITER [8]	4M	155M	72.70	72.91	77.18	77.85
OSCAR [23]	4M	155M	73.16	73.44	78.07	78.36
UNIMO [22]	5.6M	307M	75.06	75.27	-	-
VinVL$_L$ [45]	5.6M	347M	**76.52**	**76.60**	**82.67**	**83.98**
ViLT [16]	4M	87M	70.33	-	74.41	74.57
***PTP*-ViLT**	4M	87M	72.13$_{+1.8}$	74.36	76.52$_{+2.1}$	77.83$_{+3.3}$
BLIP † [19]	4M	220M	73.92	74.13	77.52	77.63
***PTP*-BLIP**	4M	220M	75.47$_{+1.6}$	75.88$_{+1.7}$	80.73$_{+3.2}$	81.24$_{+3.8}$
ALBEF [20]	14M	210M	75.84	76.04	82.55	83.14
BLIP [19]	14M	220M	77.54	77.62	82.67	82.30
***PTP*-BLIP**	14M	220M	**78.44**$_{+2.9}$	**78.33**$_{+1.7}$	**84.55**$_{+1.9}$	83.17$_{+0.9}$
SimVLM [39]	1.8B	1.2B	77.87	78.14	81.72	81.77
GIT [37]	0.8B	0.7B	-	78.81	-	-
Beit-3 [38]	35M+	1.9B	84.19	84.03	91.51	92.58

Table 5. **Comparisons with state-of-the-art methods for text-to-video retrieval on the 1k test split of the MSRVTT dataset.**

Method	R1↑	R5↑	R10↑	MdR↓
ActBERT [48]	8.6	23.4	33.1	36.0
MIL-NCE [27]	9.9	24.0	32.4	29.5
Frozen-in-time [5]	18.7	39.5	51.6	10.0
OA-Trans [36]	23.4	47.5	55.6	8.0
PTP-ViLT	**27.9**	**52.5**	**56.3**	**7.0**

4.2.4 Visual Reasoning

Natural Language Visual Reasoning (NLVR^2) [35] task is a binary classification task given triplets of two images and a question in natural language. This task relies on position information heavily. As shown in Tab. 4, SimVLM [39] is outperformed by *PTP*-BLIP, which has a reasonable model size and was pretrained on fewer instances. Meanwhile, our method is also closed to VinVL$_{large}$ model that adopt larger model and use object feature from strong object detector instead of raw-pixel image as input.

4.2.5 Video-Language Tasks

We analyze the generalization ability of our method to video-language tasks in this experiment. Specifically, we perform zero-shot transfer to text-to-video retrieval in Tab. 5, where we directly evaluate the models trained on COCO-retrieval. We just uniformly sample 8 frames each video in order to process video input, then concatenate the frame features into a single sequence. Our method leads to better result than OA-Trans [36] that focus on retrieval task, which showcase the generality capability of *PTP*.

4.3. Ablation & Design Choices

In this section, we first evaluate our method on retrieval task over three well-known baselines under 4M setting for comparison. Then we train a BLIP model on CC3M as baseline and perform various ablations.

4.3.1 The Variations of Architecture.

We experiment with three distinct kind baselines: ViLT, CLIP, and BLIP in order to explore the impact of *PTP*. Tab. 6 reports the performance on the COCO 5K test set. Comparing the outcomes of these baseline experiments, we find that *PTP* greatly improves the i2t and t2i performance. This suggests that *PTP* has good generality.

In addition, we also compare the running time. Since we do not use object detector or prompt in downstream task, the computation cost keep consistent with baseline models but 20 times faster than object feature based VinVL [45].

4.3.2 Text Prompt vs. Additional Pretext Task

We examine the effects of regarding *PTP* as a new pretext task. In this way, the pretext task does not influence the other pre-training objectives, such as ITM and ITC, but it does add to the cost of computation. Contrarily, the prompt design simply modifies the text input, therefore it will have an impact on all pre-training objectives.

Table 6. **The ablation on different architectures under 4M setting.** We report the i2t and t2i results on MSCOCO (5K test set). As we do not used object detector in downstream tasks, *PTP* is 20 times faster than object-feature based model.

Method	Time	MSCOCO (5K test set)						
		Image → Text			Text → Image			
		R@1	R@5	R@10	R@1	R@5	R@10	Avg
One-stream Models								
ViLT [16]	∼15	61.8	86.2	92.6	41.3	72.0	82.5	72.7
***PTP*-ViLT**	∼15	**67.1**	**90.5**	**94.3**	**45.3**	**79.1**	**88.4**	**77.5**$_{+4.8}$
Dual-stream Models								
CLIP† [31]	∼27	64.9	83.2	90.1	50.4	76.3	84.7	74.9
***PTP*-CLIP**	∼27	**68.3**	**86.4**	**92.7**	**54.1**	**80.1**	**86.8**	**78.1**$_{+3.2}$
Dual-stream + Fusion encoder Models								
BLIP † [19]	∼33	75.2	93.3	96.3	57.4	82.1	89.5	82.3
***PTP*-BLIP**	∼33	**83.7**	**97.0**	**98.7**	**68.1**	**89.4**	**94.2**	**88.5**$_{+6.2}$
Object-feature Based Models								
VinVL [45]	∼650	74.9	92.6	96.3	58.1	83.2	90.1	82.5

Table 7. **Text prompt vs. additional pretext head.** The last column is COCO captioning task.

Method	COCO TR@1	F30K TR@1	NLVR Acc(%)	Captioning CIDER
Baseline	70.6	53.4	76.1	121.2
Pretext	72.3 (1.7↑)	54.7 (2.3↑)	76.9 (0.8↑)	123.5 (2.3↑)
Prompt	**73.2 (2.6↑)**	**55.4 (2.0↑)**	**77.9 (1.8↑)**	**127.2 (6.0↑)**

We report the result in Tab. 7. We observe both Pretext and Prompt design improved the baseline over all four tasks. However, prompting is far preferable to pretext, particularly for COCO captioning CIDER (127.2 vs 123.5). In this work, we use prompt as default due to its efficiency.

4.3.3 Other Types of Text Prompt

In this experiment, we explore six different kind of prompts: *i*. The [O] is in block [P]. *ii*. The block [P] looks like [O]. *iii*. The [O] is in which block? In [P]. *iv*. The [O] is located in block [P]. *v*. (X_1, Y_1, W, H) has a [O]. (X_1, Y_1) is the top left point and W, H are the width and height for bounding box. *vi*. The block [P] has a [O]. *vii*. The block [NP] has a [O]. NP means we use nouns to represent the block position. e.g, from upper left to bottom right. More variations can be found in the supplementary.

We report the result in Tab. 8 and observe precise position does not produce superior results to block, the reason maybe precise position is hard to learn. In addition, we find use block ID (like 0) or nouns (like upper left) remain similar results. In the end, we discover that the hybrid version does not produce the best outcomes.

4.3.4 The Importance of Position in Text Prompt

In this experiment, we examine the efficacy of prompting our *PTP* for information at various granularities, such as without Positional. We simply use *[P] has [O]* when remove prompt. We list the results in Tab. 9. We observe: *i*.

Table 8. **Case study of text prompt on image-text retrieval.** A single-word change in prompt could yield a drastic difference. O is short for object and P is short for position.

Prompt	TR@1	IR@1
Baseline	70.6	53.4
The [O] is in the block [P].	72.7 (2.1↑)	54.1 (0.7↑)
The block [P] looks like [O].	73.3 (2.7↑)	53.9 (0.5↑)
The [O] is in which block? In [P].	72.3 (1.7↑)	54.9 (1.5↑)
The [O] is located in block [P].	72.3 (1.7↑)	54.2 (0.8↑)
(X1, Y1, W, H) has a [O].	72.5 (1.9↑)	54.3 (0.9↑)
The block in [NP] has a [O].	73.0 (2.4↑)	55.1 (1.7↑)
The block [P] has a [O].	**73.2 (2.6↑)**	**55.4 (2.0↑)**
Mixed	72.3 (1.7↑)	54.7 (1.2↑)

Table 9. **The position information is essential for prompt design.** Different variations of object prediction prompt design and evaluate on coco retrieval.

Object Tags	Prompt	Position	TR@1	IR@1
-	-	-	70.6	53.4
✓			70.2 (0.4↓)	52.7 (0.7↓)
✓	✓		70.3 (0.3↓)	52.9 (0.5↓)
✓		✓	70.8 (0.3↓)	52.4 (1.0↓)
✓	✓	✓	**73.3 (2.7↑)**	**55.4 (2.0↑)**

It's interesting to see that each component is crucial. Without any one component, the downstream performance to get progressively poorer. *ii*. Although OSCAR [23] discovered that using object tags as a supplementary input improved results when area features were used as input, we have shown that object tags are ineffective when raw pixel images are used. This serves as an illustration of the need to create a workable prompt for understanding the alignment between object tags and image region.

4.3.5 Number of Blocks

We explore if more fine-grained position information helps in our *PTP*. In Fig. 4, we varying the number of blocks from 1×1 (remove position information in *PTP*) to 4×4 and report the relative performance based on both BLIP and ViLT models. As can be seen, the results for both backbones are improved when the number of blocks is more than 1. However, once there are 16 blocks, all downstream activities experience a relative drop in performance. The reason may be that the predicted bounding box deviates from the localization of the real object, resulting in a mesh that is too small and may not contain the selected object. We hence recommend using 3×3 blocks, as it enjoys accurateness.

4.3.6 Is Object Detector Necessary?

In this work, a part of predicted bounding box information is coming from Faster-rcnn [33]. In order to verify the expressive power of object, we also consider two variations: *i*. Pure clip similarity. This design choice is adapted

Table 10. **The different ways to get grid pseudo label and its corresponding running time.** We report the image-to-text retrieval result on the COCO dataset for reference.

Method	Time	R1	R5	R10
baseline	-	70.6	91.3	95.4
Faster-RCNN (ResNet101)	10d	72.7	91.8	95.7
Faster-RCNN (ResNeXt152)	14d	**73.3**	**92.0**	96.1
CLIP Similarity	8h	72.9	**92.0**	**96.6**

mainly for efficiency reasons, where utilizing object detector is time consuming and not easy to access sometimes. *ii*. In addition to the powerful ResNext152-based object detector, we also use a smaller Faster-rcnn network that utilizes ResNet101 as backbone.

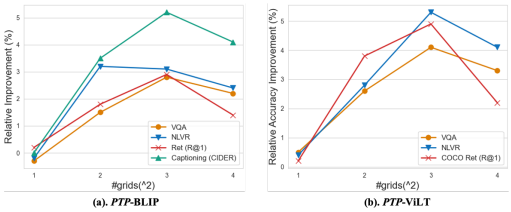

Figure 4. **The relation between the number of blocks and the relative accuracy improvement.** We explore two baselines and show the improvements over four different tasks.

The results are reported in Tab. 10. We also report the overall feature extracting time on 8 NVIDIA V100 GPUs. As can be seen from the table, we found that using stronger detector leads to better result, but bring huge computation cost at the same time. Moreover, we observe the result of CLIP embedding is very closed to Faster-rcnn (ResNeXt152). In addition, it takes only around 2.3% time of Faster-rcnn (ResNeXt152) version to extract pseudo label for each grid. We came to the conclusion that a clip model is a good alternative of object detector in *PTP*.

4.4. Visualization

To explore whether model training with the *PTP* framework does indeed learn position information, we design a fill-in-the-blank evaluation experiment in this section. Follow ViLT [16], we masked some key words and asked the model to predict the masked words and show its corresponding heatmap. We design two text prompts, given the noun to predict the localization and given the localization to predict the missing noun. We show top-3 predictions and more visualization results can be found in supplementary.

The results are shown in Fig. 5. On the one hand, we find that the *PTP*-ViLT can make correct object prediction based on the block position information and its visual concepts. On the other hand, when only masked the position information, we witness a high predicted probability value for corrected block. For example, in the bottom of Fig. 5, our model find all patches looks like *"man"* correctly. Based on these experiments and Fig. 1, we conclude that the *PTP* can

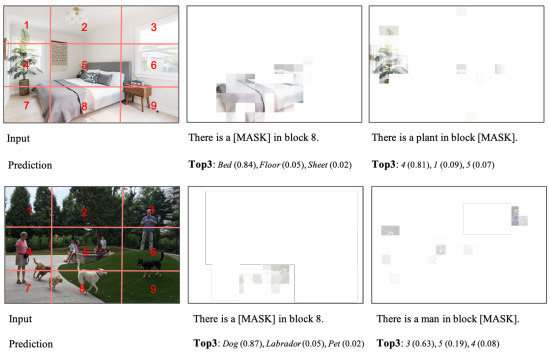

Figure 5. **The full-in-the-blank task evaluation.** We ask the model to predict *what objects are contained in given block* and *predict which blocks contain specific object*.

help the base VLP model learn position information very well based on our simple text prompt.

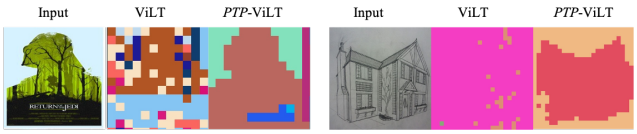

Figure 6. **Token cluster visualization.** We train ViLT and *PTP*-ViLT with ViT-B/32 model on CC3M train set. We show the token cluster result with KMeans algorithm from CC3M test set [34]. *PTP*-ViLT shows preferable clusters.

Furthermore, we cluster the token-level features with K-Means algorithm for ViLT and *PTP*-ViLT. Intuitively, the token with similar semantic should be clustered together. We show the visualization result in Fig. 6. Comparing with ViLT baseline, we observe that our method can cluster similar patches more accurate. This illustrate our *PTP* have fairly accurate learns semantic information.

5. Limitations and Conclusion

We first try to leverage the position information from existing object detector/trained model to VLP models with simple prompt. We provide a success practice cross-modal prompt settings to aid prompt engineering. Through rigorous experiments, we showed that *PTP* could serve as a general-purpose pipeline and improve the learning of position information without much extra computation cost. However, at this time, *PTP* does not take into account how to deal with the wrong object tag. Additionally, this work does not adequately explore more complicated prompts. Future research will also examine how well *PTP* performs on additional vision-language tasks.

Acknowledgement

This project is supported by the National Research Foundation, Singapore under its NRFF Award NRF-NRFF13-2021-0008.

References

[1] Harsh Agrawal, Karan Desai, Yufei Wang, Xinlei Chen, Rishabh Jain, Mark Johnson, Dhruv Batra, Devi Parikh, Stefan Lee, and Peter Anderson. Nocaps: Novel object captioning at scale. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8948–8957, 2019.

[2] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *arXiv preprint arXiv:2204.14198*, 2022.

[3] Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. Bottom-up and top-down attention for image captioning and visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6077–6086, 2018.

[4] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pages 2425–2433, 2015.

[5] Max Bain, Arsha Nagrani, Gül Varol, and Andrew Zisserman. Frozen in time: A joint video and image encoder for end-to-end retrieval. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1728–1738, 2021.

[6] Soravit Changpinyo, Piyush Sharma, Nan Ding, and Radu Soricut. Conceptual 12m: Pushing web-scale image-text pre-training to recognize long-tail visual concepts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3558–3568, 2021.

[7] Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco captions: Data collection and evaluation server. *arXiv preprint arXiv:1504.00325*, 2015.

[8] Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. Uniter: Learning universal image-text representations. 2019.

[9] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 702–703, 2020.

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

[11] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[12] Xiaowei Hu, Zhe Gan, Jianfeng Wang, Zhengyuan Yang, Zicheng Liu, Yumao Lu, and Lijuan Wang. Scaling up vision-language pre-training for image captioning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17980–17989, 2022.

[13] Zhicheng Huang, Zhaoyang Zeng, Yupan Huang, Bei Liu, Dongmei Fu, and Jianlong Fu. Seeing out of the box: End-to-end pre-training for vision-language representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12976–12985, 2021.

[14] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *International Conference on Machine Learning*, pages 4904–4916. PMLR, 2021.

[15] Muhammad Uzair Khattak, Hanoona Rasheed, Muhammad Maaz, Salman Khan, and Fahad Shahbaz Khan. Maple: Multi-modal prompt learning. *arXiv preprint arXiv:2210.03117*, 2022.

[16] Wonjae Kim, Bokyung Son, and Ildoo Kim. Vilt: Vision-and-language transformer without convolution or region supervision. In *International Conference on Machine Learning*, pages 5583–5594. PMLR, 2021.

[17] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International journal of computer vision*, 123(1):32–73, 2017.

[18] Gen Li, Nan Duan, Yuejian Fang, Ming Gong, and Daxin Jiang. Unicoder-vl: A universal encoder for vision and language by cross-modal pre-training. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 11336–11344, 2020.

[19] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *ICML*, 2022.

[20] Junnan Li, Ramprasaath Selvaraju, Akhilesh Gotmare, Shafiq Joty, Caiming Xiong, and Steven Chu Hong Hoi. Align before fuse: Vision and language representation learning with momentum distillation. *Advances in neural information processing systems*, 34:9694–9705, 2021.

[21] Liunian Harold Li, Pengchuan Zhang, Haotian Zhang, Jianwei Yang, Chunyuan Li, Yiwu Zhong, Lijuan Wang, Lu Yuan, Lei Zhang, Jenq-Neng Hwang, et al. Grounded language-image pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10965–10975, 2022.

[22] Wei Li, Can Gao, Guocheng Niu, Xinyan Xiao, Hao Liu, Jiachen Liu, Hua Wu, and Haifeng Wang. Unimo: Towards unified-modal understanding and generation via cross-modal contrastive learning. *arXiv preprint arXiv:2012.15409*, 2020.

[23] Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, et al. Oscar: Object-semantics aligned pre-training for vision-language tasks. In *European Conference on Computer Vision*, pages 121–137. Springer, 2020.

[24] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[25] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*, 2021.

[26] Zhijian Liu, Simon Stent, Jie Li, John Gideon, and Song Han. Loctex: Learning data-efficient visual representations from localized textual supervision. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2167–2176, 2021.

[27] Antoine Miech, Jean-Baptiste Alayrac, Lucas Smaira, Ivan Laptev, Josef Sivic, and Andrew Zisserman. End-to-end learning of visual representations from uncurated instructional videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9879–9889, 2020.

[28] Vicente Ordonez, Girish Kulkarni, and Tamara Berg. Im2text: Describing images using 1 million captioned photographs. *Advances in neural information processing systems*, 24, 2011.

[29] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[30] Di Qi, Lin Su, Jia Song, Edward Cui, Taroon Bharti, and Arun Sacheti. Imagebert: Cross-modal pre-training with large-scale weak-supervised image-text data. *arXiv preprint arXiv:2001.07966*, 2020.

[31] 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 *ICML*, pages 8748–8763. PMLR, 2021.

[32] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J Liu, et al. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(140):1–67, 2020.

[33] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015.

[34] Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2556–2565, 2018.

[35] Alane Suhr, Stephanie Zhou, Ally Zhang, Iris Zhang, Huajun Bai, and Yoav Artzi. A corpus for reasoning about natural language grounded in photographs. *arXiv preprint arXiv:1811.00491*, 2018.

[36] Jinpeng Wang, Yixiao Ge, Guanyu Cai, Rui Yan, Xudong Lin, Ying Shan, Xiaohu Qie, and Mike Zheng Shou. Object-aware video-language pre-training for retrieval. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3313–3322, 2022.

[37] Jianfeng Wang, Zhengyuan Yang, Xiaowei Hu, Linjie Li, Kevin Lin, Zhe Gan, Zicheng Liu, Ce Liu, and Lijuan Wang. Git: A generative image-to-text transformer for vision and language. *arXiv preprint arXiv:2205.14100*, 2022.

[38] Wenhui Wang, Hangbo Bao, Li Dong, Johan Bjorck, Zhiliang Peng, Qiang Liu, Kriti Aggarwal, Owais Khan Mohammed, Saksham Singhal, Subhojit Som, et al. Image as a foreign language: Beit pretraining for all vision and vision-language tasks. *arXiv preprint arXiv:2208.10442*, 2022.

[39] Zirui Wang, Jiahui Yu, Adams Wei Yu, Zihang Dai, Yulia Tsvetkov, and Yuan Cao. Simvlm: Simple visual language model pretraining with weak supervision. *arXiv preprint arXiv:2108.10904*, 2021.

[40] Lewei Yao, Runhui Huang, Lu Hou, Guansong Lu, Minzhe Niu, Hang Xu, Xiaodan Liang, Zhenguo Li, Xin Jiang, and Chunjing Xu. Filip: Fine-grained interactive language-image pre-training. *arXiv preprint arXiv:2111.07783*, 2021.

[41] Yuan Yao, Ao Zhang, Zhengyan Zhang, Zhiyuan Liu, Tat-Seng Chua, and Maosong Sun. Cpt: Colorful prompt tuning for pre-trained vision-language models. *arXiv preprint arXiv:2109.11797*, 2021.

[42] Jiahui Yu, Zirui Wang, Vijay Vasudevan, Legg Yeung, Mojtaba Seyedhosseini, and Yonghui Wu. Coca: Contrastive captioners are image-text foundation models. *arXiv preprint arXiv:2205.01917*, 2022.

[43] Lu Yuan, Dongdong Chen, Yi-Ling Chen, Noel Codella, Xiyang Dai, Jianfeng Gao, Houdong Hu, Xuedong Huang, Boxin Li, Chunyuan Li, et al. Florence: A new foundation model for computer vision. *arXiv preprint arXiv:2111.11432*, 2021.

[44] Yan Zeng, Xinsong Zhang, and Hang Li. Multi-grained vision language pre-training: Aligning texts with visual concepts. *arXiv preprint arXiv:2111.08276*, 2021.

[45] Pengchuan Zhang, Xiujun Li, Xiaowei Hu, Jianwei Yang, Lei Zhang, Lijuan Wang, Yejin Choi, and Jianfeng Gao. Vinvl: Making visual representations matter in vision-language models. 2021.

[46] Yiwu Zhong, Jianwei Yang, Pengchuan Zhang, Chunyuan Li, Noel Codella, Liunian Harold Li, Luowei Zhou, Xiyang Dai, Lu Yuan, Yin Li, et al. Regionclip: Region-based language-image pretraining. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16793–16803, 2022.

[47] Chong Zhou, Chen Change Loy, and Bo Dai. Denseclip: Extract free dense labels from clip. *arXiv preprint arXiv:2112.01071*, 2021.

[48] Linchao Zhu and Yi Yang. Actbert: Learning global-local video-text representations. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8746–8755, 2020.