CLIP Models are Few-shot Learners: Empirical Studies on VQA and Visual Entailment

Haoyu Song[†][*], **Li Dong**[‡], **Wei-Nan Zhang**[†], **Ting Liu**[†], **Furu Wei**[‡]
[†] Harbin Institute of Technology
[‡] Microsoft Research
{hysong,wnzhang,tliu}@ir.hit.edu.cn
{lidong1,fuwei}@microsoft.com

Abstract

CLIP has shown a remarkable zero-shot capability on a wide range of vision tasks. Previously, CLIP is only regarded as a powerful visual encoder. However, after being pretrained by language supervision from a large amount of image-caption pairs, CLIP itself should also have acquired some few-shot abilities for vision-language tasks. In this work, we empirically show that CLIP can be a strong vision-language few-shot learner by leveraging the power of language. We first evaluate CLIP's zero-shot performance on a typical visual question answering task and demonstrate a zero-shot cross-modality transfer capability of CLIP on the visual entailment task. Then we propose a parameter-efficient fine-tuning strategy to boost the few-shot performance on the vqa task. We achieve competitive zero/few-shot results on the visual question answering and visual entailment tasks without introducing any additional pre-training procedure.

1 Introduction

Vision-language understanding (VLU) tasks, such as visual question answering (Antol et al., 2015) and visual entailment (Xie et al., 2019), test a system's ability to comprehensively understand the semantics of both visual world and natural language. To capture the alignment between vision and language, various efforts have been made to build the vision-language pre-trained models (Lu et al., 2019; Chen et al., 2020; Su et al., 2020; Zhang et al., 2021; Wang et al., 2021). Despite their superior performances, these methods have extensively utilized human-annotated training data that are expensive or require expert knowledge, such as object detection datasets (Lin et al., 2014; Kuznetsova et al., 2020) and aligned image-text pairs (Deng et al., 2009; Sharma et al., 2018). Collecting such datasets requires heavy work on data gathering and human annotation, and thus their scales are only in the realm of tens of millions, which are much smaller than the Internet text corpora for NLP pretraining (Devlin et al., 2019; Brown et al., 2020).

[*]Contribution during internship at Microsoft Research.

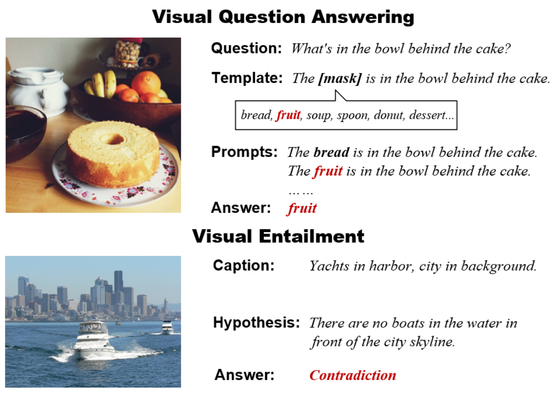

Visual Question Answering
Question: What's in the bowl behind the cake?
Template: The [mask] is in the bowl behind the cake.
bread, fruit, soup, spoon, donut, dessert...
Prompts: The bread is in the bowl behind the cake.
The fruit is in the bowl behind the cake.
……
Answer: fruit
Visual Entailment
Caption: Yachts in harbor, city in background.
Hypothesis: There are no boats in the water in front of the city skyline.
Answer: Contradiction

Figure 1: Examples of the two vision-language understanding tasks. For VQA, language prompts are used. For visual entailment, caption and hypothesis, i.e., text-text pairs, are used in training, while image and hypothesis, i.e., image-text pairs, are used at inference.

Recently, CLIP (Radford et al., 2021) has been proposed to learn visual concepts with natural language supervision, where its 400 million image-text pairs are crawled from the Internet. CLIP consists of a visual encoder and a text encoder, and it learns visual representations by aligning images and texts through contrastive loss. In this way, CLIP achieves strong zero-shot performances on vision benchmarks such as ImageNet. Besides, Shen et al. (2022) prove that CLIP could be leveraged as a strong visual encoder to benefit downstream vision-language tasks. However, there are two major differences between CLIP and previous visual encoders: 1) it is trained on much larger yet noisy web data, and 2) it has a shallow interaction between vision and language. The first feature promises the generalization ability of CLIP, and the second one equips alignment ability across modalities. Could the strong zero-shot ability of CLIP be transferred to vision-language understanding tasks?

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 6088 - 6100
May 22-27, 2022 ©2022 Association for Computational Linguistics

To answer the above question, in this work, we empirically study how to transfer CLIP's zero-shot ability into VLU tasks and further turn CLIP into a few-shot learner. We carried out experiments on two VLU tasks: 1) visual question answering, where the model needs to give an answer according to the details of an image and a natural sentence question, and 2) visual entailment, where the model needs to determine the entailment relation between an image and a natural sentence. Figure 1 demonstrates the basic forms of the two studied tasks.

For the zero-shot visual question answering task, the key to a successful zero-shot capability transfer is to mitigate the gap between the pre-training task of CLIP and the task form of question answering. Inspired by the recent advancements of few-shot learning in NLP (Schick and Schütze, 2021b; Gao et al., 2021), we address this issue by introducing a two-step prompt generation strategy, including automatic conversions from question to statement to get masked templates, and a span-infilling with generative pre-trained T5 model (Raffel et al., 2020) to get candidate answers.

We explore a zero-shot cross-modality (language and vision) transfer capability through the visual entailment task. Specifically, we replace the image with its captions during training and only update a small classification layer. Then at inference, as usual, we still use image-text pairs for testing. This allows us to investigate how well the language and vision representations are aligned in CLIP models.

We further leverage few-shot learning to improve CLIP's visual question answering performance based on the zero-shot transferring methods. We find that optimizing only bias and normalization (BiNor) parameters would make better use of limited examples and yield better results than the latest few-shot model *Frozen* (Tsimpoukelli et al., 2021). Experiments confirm that CLIP models can be good vision-language few-shot learners.

Our contributions are summarized as follows:

- To the best of our knowledge, this is the first work that studies how to transfer CLIP's zero-shot capabilities into VLU tasks and confirms CLIP models can be good few-shot learners.

- A zero-shot cross-modality transfer capability in CLIP is demonstrated.

- A parameter-efficient fine-tuning strategy, BiNor, is proposed to boost CLIP's few-shot visual question answering performance.

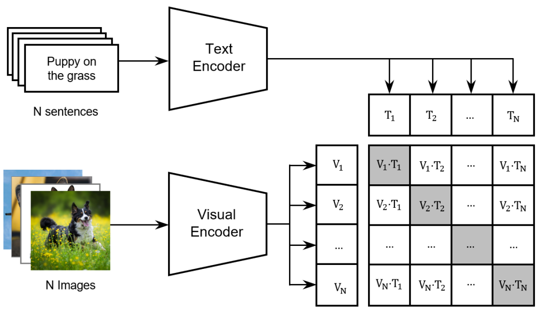


Figure 2: CLIP consists of a visual encoder $\mathbb{V}$, a text encoder $\mathbb{T}$, and a dot product between their outputs. It is trained to align images and texts with a contrastive loss. The dot product is used as an alignment score.

2 Preliminaries

2.1 CLIP

CLIP, short for Contrastive Language-Image Pre-training (Radford et al., 2021), learns visual representations from natural language supervision. Figure 2 shows its key components and the way it works. It consists of a visual encoder $\mathbb{V}$, e.g. ResNet (He et al., 2016) and ViT (Dosovitskiy et al., 2020), and a text encoder $\mathbb{T}$, e.g. transformer (Vaswani et al., 2017), where they encode images and texts independently. Followed up is a dot-product between the two encoders' outputs, i.e. $\mathbb{T}(\text{text}) \cdot \mathbb{V}(\text{image})$, which is used as an alignment score between the input image and text. It is pre-trained to distinguish aligned image-text pairs from randomly combined ones by a contrastive loss. Instead of training on vision benchmarks, CLIP leverages abundant language supervisions from 400 million web-crawled image-text pairs and can conduct a variety of image classification tasks without specific optimizing. However, directly applying CLIP as a vision-language understanding model is still difficult (Kim et al., 2021; Shen et al., 2022).

2.2 Vision-Language Understanding Tasks

Visual question answering. The task of VQA requires the model to answer questions about the details of input images. Following previous work, we experiment on the VQAv2 (Goyal et al., 2017) dataset and formulate the task as a classification problem over 3,129 pre-defined most frequent answers. The images in VQAv2 come from Microsoft COCO (Lin et al., 2014), and there are 65 types of questions in the dataset, such as *how many* and *what color is*. For answers, there are three types, including *yes/no*, *number*, and *other*.

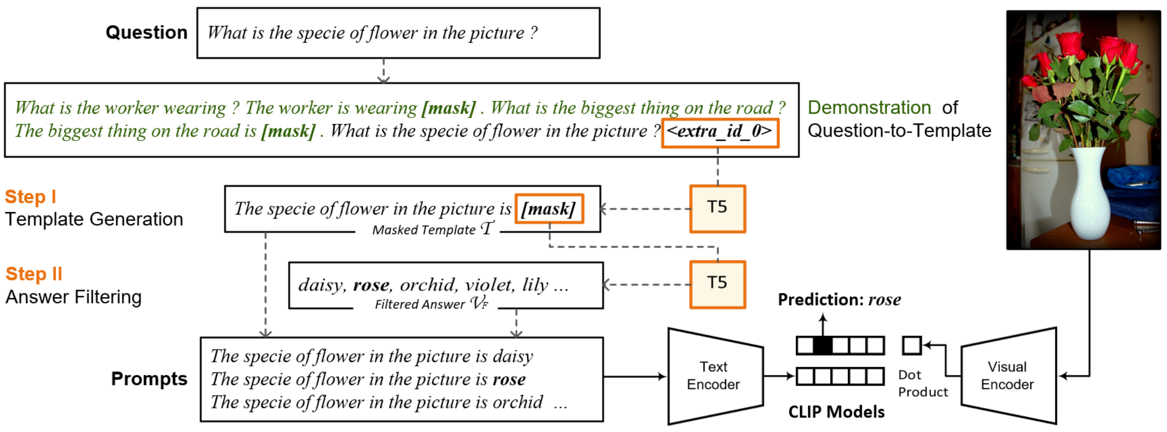

Question
What is the specie of flower in the picture ?
What is the worker wearing ? The worker is wearing [mask] . What is the biggest thing on the road ? The biggest thing on the road is [mask] . What is the specie of flower in the picture ? <extra_id_0>
Demonstration of Question-to-Template
Step I
Template Generation
The specie of flower in the picture is [mask]
Masked Template $\mathcal{T}$
T5
Step II
Answer Filtering
daisy, rose, orchid, violet, lily ...
Filtered Answer $\mathcal{V}_F$
T5
Prediction: rose
Prompts
The specie of flower in the picture is daisy
The specie of flower in the picture is rose
The specie of flower in the picture is orchid ...
Text Encoder
CLIP Models
Dot Product
Visual Encoder

Figure 3: The overall framework of the proposed TAP-C method for zero-shot VQA. TAP-C first generates a masked template from the question by demonstrating examples to T5 and then filters out impossible answers according to the language model. Infilling the masked template with selected answers results in prompts, which could be paired with images to calculate image-text alignment scores by the CLIP. The dashed line denotes the process of prompts generation (§ 3.1), and the solid line denotes prompting CLIP to conduct zero-shot VQA (§ 3.2).

Visual entailment. Similar to the natural language inference (NLI), the task of visual entailment predicts the entailment relations, including *entailment*, *neutral*, and *contradiction*, between a premise and a hypothesis. Under the VL setting, the premise in visual entailment is based on the details of an image rather than textual descriptions in NLI. The SNLI-VE dataset (Xie et al., 2019) is adapted from SNLI (Bowman et al., 2015) and replaces SNLI's premises with the images in the Flickr30k dataset (Young et al., 2014). Considering the above characteristics, here we leverage the SNLI-VE dataset to verify the zero-shot cross-modality (language and vision) transfer capabilities of the CLIP models. This zero-shot setting investigates how well the vision and language representations are aligned in CLIP models.

3 Zero-shot VQA

3.1 A Two-Step Prompt Generation Method

Previous works (Kim et al., 2021; Shen et al., 2022) have found that directly applying CLIP models for zero-shot VL tasks are infeasible. For example, nearly random-chance level zero-shot performances are observed on the VQAv2 dataset by directly applying a "question: [*question text*] answer: [*answer text*]" prompt template (Shen et al., 2022). After rethinking the essence of prompt engineering in CLIP, we can find that the key to a successful zero-shot capability transfer for the VQA task is to mitigate the gap between natural language description and the form of question answering.

Motivated by the above observations, we propose a two-step automatic prompt generation method to enable the zero-shot VQA capabilities in CLIP models, with the assistant of a pre-trained generative T5 model (Raffel et al., 2020). The key ideas of the two-step prompt generation method is illustrated in Figure 3: the first step is to convert the question into a masked template $\mathcal{T}$, and the second step is to filter out impossible answers by language model and get a candidate answer set $\mathcal{V}_F$. The infilled template connects both the question and answers in a natural description way and thus could be an ideal form of prompt for the VQA task.

Step I: Automatic Template Generation

This step is designed to convert the question into a template, which is a statement with a mask token. To tackle the conversion challenge, we explore two ways, including an in-context demonstration method and a dependency parsing based method.

Demonstration to T5. The idea of this conversion method is relatively simple: by demonstrating question-to-template (with [mask] token) examples to the language model, the model could implicitly capture the conversion pattern. We define a few examples for each question type and convert the questions according to their types. Figure 3 shows a conversion example. More cases could be found at appendix D. Specifically, we use T5 (Raffel et al., 2020), a large pre-trained text-to-text Transformer, for the question to template conversion. T5 is pre-trained to infill the missing spans (replaced by T5 special tokens, e.g. <*extra_id_0*>) of a sentence.

We present a concatenation of examples, question, and the <*extra_id_0*> token to T5 for conditional generation to restore it, and the generated span is our masked template, named as $\mathcal{T}_{\text{demo}}$.

Dependency parsing. Although the T5 conversion method works well in most situations, it still faces some out-of-coverage problems. To compensate for this shortcoming, we turn to a traditional dependency parsing based way. This method converts a question to a statement by its part-of-speech tagging and parsing results, where the wh-word, root word, auxiliary, or copula, as well as prepositions and particles that are dependents of the wh-word or the root, are identified, and transformations are performed according to grammar rules. We use the Stanza (Qi et al., 2020) to POS tag and parse the question and leave the answer as a mask token. Then the rules[1] in Demszky et al. (2018) are leveraged to perform the conversion. We name the template obtained in this way as $\mathcal{T}_{\text{parsing}}$.

Step II: Answer Filtering

As common sense, "*the specie of a flower*" can never be a vase. Therefore, leveraging pre-trained language models, which have well learned such concepts during pre-training, to filter out less likely answers would have a positive influence on the final question answering performance. Given a masked template $\mathcal{T}$, a language model $\mathcal{L}$, and the answer vocabulary $\mathcal{V}$, we get the filtered answers $\mathcal{V}_F$ as:

$$\underset{v \in \mathcal{V}}{\text{Top-}k} \left\{ \log P_{\mathcal{L}} \left([\text{mask}] = v | \mathcal{T} \right) \right\}, \tag{1}$$

where the [mask] is the answer span in template $\mathcal{T}$, and $P_{\mathcal{L}}$ is the output distribution of the language model. Here we also apply the T5 to infill answers because it makes no assumption about the length and position of the span. Once we get the template $\mathcal{T}$ and the filtered answers $\mathcal{V}_F$, we replace the [mask] token in template $\mathcal{T}$ with every selected answer in $\mathcal{V}_F$ to get the prompts $\mathcal{P}$.

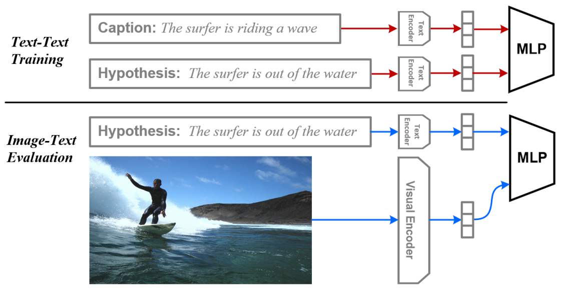
3.2 TAP-C Method for VQA

The proposed method follows a Template-Answer-Prompt then CLIP discrimination pipeline, and thus we name it as **TAP-C**. To make better use of template $\mathcal{T}_{\text{parsing}}$ and $\mathcal{T}_{\text{demo}}$, we use an ensemble of both templates by simply setting a threshold for the T5's generation confidence. We prefer to use $\mathcal{T}_{\text{demo}}$ but use $\mathcal{T}_{\text{parsing}}$ if the generation confidence is low. Finally, given an image i and the generated prompts $\mathcal{P}$, the TAP-C method can get a zero-shot VQA prediction by:

$$\max_{v \in \mathcal{V}_F,\ p_v \in \mathcal{P}} \left\{ \mathbb{V}(i) \cdot \mathbb{T}(p_v) \right\}, \tag{2}$$

where $\mathbb{V}$ and $\mathbb{T}$ are the visual and text encoders in CLIP models. The p_v is a prompt generated by the TAP-C method, where the masked template is infilled with answer v from the filtered answer vocabulary $\mathcal{V}_F$.

[1]https://github.com/kelvinguu/qanli



Text-Text Training
Caption: The surfer is riding a wave
Hypothesis: The surfer is out of the water
Text Encoder
Text Encoder
MLP
Image-Text Evaluation
Hypothesis: The surfer is out of the water
Text Encoder
Visual Encoder
MLP

Figure 4: Zero-shot cross-modality transfer on visual entailment task. The red line denotes the text-only training process, and the blue line denotes the image-text, i.e., cross-modality, evaluation process. The MLP is shared between training and evaluation, while both encoders in CLIP models are not updated.

4 Zero-shot Cross-modality Transfer

Recent pre-trained multilingual language models (Wu and Dredze, 2019; Liu et al., 2020; Xue et al., 2021) have been shown to be successful in transferring representations across different languages. For example, they can be only fine-tuned on a source language and evaluated on various target languages without specific training, yet still achieving good performance. On the other hand, the CLIP models achieve strong zero-shot performances on both image-to-text and text-to-image retrieval tasks (Radford et al., 2021) only through a dot product between vision and language representations, which gives us an intuition that the two modalities are well aligned in the CLIP models. Is there a cross-modality capability between language and vision in the CLIP models, just like the multilingual ones across languages?

To answer the above question, we utilize the visual entailment task (§ 2.2) to explore the zero-shot cross-modality performance. Figure 4 briefs the key idea. Specifically, we train an MLP classifier over the fused representations of premise and hypothesis, and the fusion function is:

$$\text{fuse}(v_1, v_2) = [v_1, v_2, v_1 + v_2, v_1 - v_2, v_1 \cdot v_2],$$

where v_1 and v_2 are two input vectors. During training, text-only premise and hypothesis are used as the input of CLIP text encoder:

$$\text{MLP}\left\{\text{fuse}\left(\mathbb{T}(\text{pre}_t),\ \mathbb{T}(\text{hyp}_t)\right)\right\}, \tag{3}$$

where $\mathbb{T}$ is the CLIP text encoder and is not updated during training. And pre_t and hyp_t are the text premise and hypothesis. Then at inference, the premise is in image and is fed into the CLIP visual encoder. The trained MLP is leveraged for prediction:

$$\max\left\{\text{MLP}\left\{\text{fuse}\left(\mathbb{V}(\text{pre}_i),\ \mathbb{T}(\text{hyp}_t)\right)\right\}\right\}, \tag{4}$$

where the pre_i is the image premise and $\mathbb{V}$ is the CLIP visual encoder.

5 Few-shot Learning for VQA

In this section, We aim to investigate whether the CLIP models could benefit from *few-shot* learning, where we work on the visual question answering task to study it.

5.1 Setup of Few-shot VQA

Here we briefly define the terminology used in our few-shot visual question answering settings:

- **Number of ways.** Originally, it is defined as the distinct classes in a task. However, rather than defining a 3,129-way task according to the answer vocabulary, we define the number of ways as question type times answer type (§ 2.2), i.e., 65×3=195 ways, to ensure the model's generalization ability where it can answer a type of questions.
- **Number of shots.** The number of distinct examples in each way. Here a shot is an image along with the question and the answer.
- **Support set and query set.** Before training, we will sample a 195-way K-shot subset from the VQAv2 training set, and thus there are 195×K distinct examples available during few-shot learning. In each training epoch, we select C ways out of 195 ways for parameter optimizing and divide *k* shots in each way into support set and query set with a fixed proportion. The support set is used for model training, and the query set is used for performance evaluation (similar to a typical dev set).

5.2 Parameter-efficient Fine-tuning

Under the few-shot setting, our goal is to make the CLIP models learn from N-way K-shot examples and improve the zero-shot VQA performance. Specifically, we identify only a very small set of parameters in CLIP models (about 0.3 million out of over 100 million, details in appendix B.3), including the *bias* term and *normalization* term, to be optimized. For either the BatchNorm in ResNet or the LayerNorm in Transformer, the normalization could be uniformly denoted as:

$$y = \frac{x - \text{E}(x)}{\sqrt{\text{Var}(x) + \epsilon}} \cdot \gamma + \beta, \tag{5}$$

where x and y are the mini-batched input and output, and the γ and β are learned parameters. And for all the linear layers and projection layers in CLIP models, they could be denoted as:

$$o = w \cdot h + bias, \tag{6}$$

where h and o are the input and output vectors. We define the learnable parameter set as:

$$\text{P}_{learn} = \{bias, \gamma, \beta\}. \tag{7}$$

We optimize the **Bi**as and **Nor**malization (BiNor) parameters on the few-shot examples with a standard cross-entropy loss over the dot products from each image-prompt pair (Eq.2).

Besides, when there are a few examples available, we could also leverage an in-context demonstration manner to improve the performance of the *answer filtering* process in TAP-C (§ 3.1) by:

$$\underset{v \in \mathcal{V}}{\text{Top-}k}\left\{\log P_{\mathcal{L}}\left([\text{mask}] = v \mid [\mathcal{D}, \mathcal{T}]\right)\right\}, \tag{8}$$

where the $\mathcal{D}$ denotes the demonstrations. $\mathcal{D}$ is similar to template $\mathcal{T}$ but has been infilled with the answers, and it is sampled from the same type of question in the available few-shot examples. The resulting filtered vocabulary is noted as $\mathcal{V}_{\text{demo}}$. We report the few-shot training procedure in appendix C.

6 Experiments

6.1 Experimental Settings

Datasets. For visual question answering and visual entailment, we carry out experiments on the VQAv2 (Goyal et al., 2017) and the SNLI-VE (Xie et al., 2019) datasets, respectively. We report the statistics of the two datasets in appendix A. For the VQA task's evaluation, we follow the *Frozen* model (Tsimpoukelli et al., 2021) to calculate the *vqa scores* on the VQAv2 validation set. For visual entailment, we calculate the *accuracy* on both validation and test sets through the *sklearn* toolkit.

CLIP models. According to the types of visual encoders, e.g. ResNet or ViT, CLIP models have different variants, resulting in a significant difference in the number of learnable *bias* and *normalization* parameters. We report the number of learnable parameters of CLIP variants in appendix B.3. We select two best performing (and publicly available) variants from two kinds of visual encoders, including the CLIP Res50x16 and the CLIP ViT-B/16, to empirically study their zero-shot and few-shot vision-language understanding performances by applying our transferring methods (§§ 3–5).

6.2 Results of Zero-shot VQA

As previous VL models heavily rely on object detection sub-modules, it is not feasible to directly apply them under the zero-shot setting. Here we setup zero-shot VL baselines from two latest works:

- **Frozen.** *Frozen* (Tsimpoukelli et al., 2021) prompts a seven-billion-parameter 32-layer language model with image representations. It is trained on aligned image-caption data and is also the first model that shows promising zero-shot and few-shot VQA performances.

- **Question irrelevant prompt.** Shen et al. (2022) explored directly prompting the CLIP models for the VQA task. They used a "question: [*question text*] answer: [*answer text*]" template, together with the prompt engineering of image classification, to prepare prompts. The resulting prompts are irrelevant to questions, and thus we note this method as QIP.

We report the zero-shot VQA results in Table 1. The experimental results verify our hypothesis (§ 3.1) that the prompts of CLIP should be used to describe the labels rather than the tasks. As we can see, the question irrelevant prompting methods simply present the task description and answers to the CLIP models and only get barely better than random guess results. In contrast, by converting questions into templates and filtering answers with pre-trained language models, our TAP-C method enables CLIP models a strong zero-shot capability on the VQA task, even compared with the seven-billion-parameter *Frozen* zero-shot model.

Zero-shot Methods	Yes/No	Number	Other	All
Frozen (Tsimpoukelli et al., 2021)	-	-	-	29.50
QIP (Shen et al., 2022)				
w/ CLIP$_{\text{Res101}}$	53.01	6.67	0.96	21.26
w/ CLIP$_{\text{Res50x16}}$	56.16	9.76	1.39	23.07
w/ CLIP$_{\text{ViT-B/16}}$	53.89	7.67	0.70	21.40
TAP-C (Ours)				
w/ CLIP$_{\text{Res50x16}}$	**71.65**	18.74	18.22	38.36
w/ CLIP$_{\text{ViT-B/16}}$	71.38	**20.95**	**18.55**	**38.72**

Table 1: Zero-shot results on the VQAv2 validation set.

Training	Text + Text		Image + Text
Evaluation	Image + Text	Image Masked	Text + Text
Majority	33.37 / 33.37	33.37 / 33.37	33.37 / 33.37
CLIP$_{\text{ViT-B/16}}$	64.11 / 64.66	35.05 / 35.69	65.97 / 66.23
CLIP$_{\text{Res101}}$	64.29 / 64.86	36.27 / 35.34	65.67 / 66.28
CLIP$_{\text{Res50x16}}$	67.24 / 66.63	36.36 / 36.05	67.64 / 68.18

Table 2: Zero-shot cross-modality transfer results on the SNLI-VE valid and test set (*valid acc* / *test acc*).

6.3 Zero-shot Cross-modality Transfer

We report the zero-shot cross-modality transfer results in Table 2. We first investigate the language to vision transfer capability. As introduced in § 4, we train a classifier on the text-only SNLI-VE dataset where the image is replaced by its caption. At inference, the trained classifier is evaluated by taking the image and text as inputs. As shown in the first group of results, after solely trained on text-text (caption as the premise) entailment data, different CLIP variants could successfully gain a similar discriminative ability under the image-text setting. To ensure that the above results are indeed transferring from language to vision, we made a double check by masking out the images at inference time, and the results are reported at *Image Masked*. As we can see, the results are similar to a random guess of three relations, indicating the images are of importance in the cross-modality evaluation.

Now that we have observed the language to vision transferring capability in CLIP models, we further investigate whether there is also a vision to language transfer capability. We conduct a similar experiment but train the classifier on the original SNLI-VE dataset, i.e., image premise and text hypothesis. At inference, we evaluate the classifier with the text-only valid and test data. The results are reported in Table 2, which confirms the vision to language capability. Since text data are usually much cheaper than visual data, the first kind of transferring tends to be more promising in practice.

Fully Supervised Results on Test-Dev			Y/N_{full}	NUM_{full}	$\text{OTHER}_{\text{full}}$	ALL_{full}		
Mcan (Yu et al., 2019)			86.82	53.26	60.72	70.63		
CLIP-ViL$_{\text{p}}$ (Shen et al., 2022)			-	-	-	76.48		
Few-shot Results on Validation Set	$\text{Y/N}_{K=1}$	$\text{NUM}_{K=1}$	$\text{OTHER}_{K=1}$	$\text{ALL}_{K=1}$	$\text{Y/N}_{K=4}$	$\text{NUM}_{K=4}$	$\text{OTHER}_{K=4}$	$\text{ALL}_{K=4}$
Frozen$_{\text{blind}}$ (Tsimpoukelli et al., 2021)	-	-	-	33.50	-	-	-	33.30
Frozen (Tsimpoukelli et al., 2021)	-	-	-	35.70	-	-	-	38.20
TAP-C$_{\text{ViT-B/16}}$ (Ours)	71.03	29.72	25.73	**43.27**	71.53	31.40	28.36	44.98
w/o $\mathcal{V}_{\text{demo}}$	71.03	**29.74**	19.01	39.96	71.53	**31.45**	21.78	41.74
TAP-C$_{\text{Res50x16}}$ (Ours)	**71.77**	26.75	**25.88**	43.24	**71.86**	27.86	**30.86**	**45.87**
w/o $\mathcal{V}_{\text{demo}}$	**71.77**	26.73	19.97	40.32	**71.86**	27.92	22.43	41.72
	$\text{Y/N}_{K=16}$	$\text{NUM}_{K=16}$	$\text{OTHER}_{K=16}$	$\text{ALL}_{K=16}$	$\text{Y/N}_{K=32}$	$\text{NUM}_{K=32}$	$\text{OTHER}_{K=32}$	$\text{ALL}_{K=32}$
TAP-C$_{\text{ViT-B/16}}$ (Ours)	**73.05**	**31.46**	32.13	47.42	**73.60**	**32.55**	35.02	49.19
w/o $\mathcal{V}_{\text{demo}}$	**73.05**	31.44	25.08	43.94	**73.60**	32.52	26.95	45.21
TAP-C$_{\text{Res50x16}}$ (Ours)	72.98	29.96	**35.58**	**48.89**	73.51	31.56	**37.35**	**50.18**
w/o $\mathcal{V}_{\text{demo}}$	72.98	29.87	26.53	44.42	73.51	31.70	28.26	45.71

Table 3: Few-shot VQA results under different k values, along with two fully supervised models' performance as references. The $\mathcal{V}_{\text{demo}}$ enhances answer filtering by few-shot demonstration to T5, details in Eq.8. Our few-shot method not only outperform *Frozen*, but also achieves stable improvements under different k values.

Zero-shot Methods	Yes/No	Number	Other	All
TAP-C$_{\text{ViT-B/16}}$	71.38	20.95	18.55	38.72
w/o $\mathcal{T}_{\text{demo}}$	71.36	20.86	17.96†	38.41†
w/o $\mathcal{T}_{\text{parsing}}$	70.82†	19.86†	18.40	38.29†

Table 4: Ablation results of template generation methods. $\mathcal{T}_{\text{demo}}$ and $\mathcal{T}_{\text{parsing}}$ denote the T5 demonstration template and dependency parsing template. "†" means statistically significant difference (2-tailed t-test, p<0.01).

Methods	K=0	K=4	K=32
TAP-C$_{\text{ViT-B/16}}$	38.72	44.98	49.19
w/o a.filt.	32.57 (16%)	35.07 (22%)	40.21 (18%)
w/o t.gen. + a.filt.	21.40 (45%)	22.59 (50%)	23.76 (52%)
TAP-C$_{\text{Res50x16}}$	38.36	45.87	50.18
w/o a.filt.	32.43 (16%)	34.56 (25%)	40.97 (18%)
w/o t.gen. + a.filt.	23.07 (40%)	23.98 (48%)	24.86 (51%)

Table 5: Ablation results of TAP-C. In brackets is the percentage of VQA performance degradation. When both steps are all removed, the few-shot learning is performed on CLIP with question irrelevant prompts.

6.4 Results of Few-shot VQA

We report the few-shot VQA results in Table 3. We take the *Frozen* model and the image blacked out *Frozen*$_{\text{blind}}$ as baselines. Under different k, our methods could always learn from limited training examples and improve over the zero-shot results, which confirms that CLIP models could be VL few-shot learners. With the increase of the number of shots, significant performance gains are observed in *other* category, which concurs with our intuition: as we sample examples from each question type, most answers in *other* category are not revealed to the model. As a result, the model could always learn to improve. Similarly, presenting examples to the T5 could also improve the answer filtering process, leading to significant performance gains over the *other* category. In contrast, the score of *number* category improves significantly when the model just begins to see some training examples while slowing down as k continues to increase.

6.5 Analyses and Discussion

The effects of template generation methods. Our TAP-C method uses an ensemble of dependency parsing template $\mathcal{T}_{\text{parsing}}$ and T5 demonstration template $\mathcal{T}_{\text{demo}}$. Here we investigate whether it is necessary to use such an ensemble. We report the ablation results of two templates in Table 4. The results show that the two templates have different effects over different questions, and the ensemble could make the best use of their advantages.

The effects of two steps in TAP-C. The TAP-C method generates prompts through template generation (**t.gen.**) and answer filtering (**a.filt.**). Here we quantify how much each step contributes to the final zero/few-shot VQA performances. We report the ablation results in Table 5. When we remove the *answer filtering* step (w/o a.filt.), both the zero-shot and few-shot performances generally fall by about 20%, but the models still retain some few-shot learning capabilities. We further remove the *template generation* step and only use question irrelevant templates: all results are nearly cut in half, indicating the importance of considering questions in both zero-shot and few-shot scenarios.

Few-shot Mehtods	K=1			K=4			K=16			K=32		
	Full-FT	BitFit	BiNor	Full-FT	BitFit	BiNor	Full-FT	BitFit	BiNor	Full-FT	BitFit	BiNor
TAP-C$_{\text{ViT-B/16}}$	37.78*	42.96	**43.27**	38.30*	44.77	**44.98**	39.99	46.80	**47.42**	40.35	47.78	**49.19**
TAP-C$_{\text{Res101}}$	36.63*	42.21	**42.98**	37.92*	43.02	**44.72**	39.42	44.83	**47.43**	39.58	45.59	**48.86**
TAP-C$_{\text{Res50x16}}$	35.96*	**43.38**	43.24	38.03*	44.30	**45.87**	39.84	46.57	**48.89**	40.42	47.74	**50.18**

Table 6: The performance comparisons of different fine-tuning strategies on the few-shot VQA task. Results marked with "$*$" are lower than the zero-shot performance. Full-FT is short for full fine-tuning. Besides BiNor's good performance, it improves ResNet CLIPs more significantly due to the number of normalization parameters.

Comparisons of fine-tuning methods. We only update the bias and normalization parameters during few-shot learning (§ 5.2). To investigate whether our BiNor fine-tuning strategy works well, we compare BiNor with two fine-tuning methods: 1) **Full-FT** (Full fine-tuning), which updates all parameters in the model. 2) **BitFit** (Ben Zaken et al., 2021), which only updates the bias-terms in all model layers. We report the comparison results in Table 6. Both BiNor and BitFit significantly outperform the full fine-tuning way: millions of parameters are very easy to overfit to a few training examples. When *k* is small, the performance differences between BiNor and BitFit are very small. When *k* becomes larger, BiNor begins to outperform BitFit with a noticeable margin. Our BiNor fine-tuning strategy is similar to the BitFit but differs in that it also updates the normalization parameters, which would grant the ResNet CLIP models better flexibility to adapt to new examples due to their larger number of batch normalization parameters. For the specific number of different parameters in each CLIP variant, please refer to the appendix B.3.

Limitations of TAP-C. The proposed TAP-C method explores CLIP models' potential to conduct zero/few-shot VQA tasks. However, we also found several limitations that hinder further improving the few-shot performance, which could be rooted in the CLIP models. First, CLIP models struggle with counting the number of fine-grained objects in an image, especially counting from a small area of the image. This shortcoming can hardly be improved by any kind of language knowledge. Besides, the CLIP models perform poorly in distinguishing subtle semantic differences. For example, when asked "what is the man in the background doing?", all the experimented CLIP models give predictions of the man "in the foreground". Under such cases, even if the TAP-C method perfectly converts the question into a prompt, the final results would still be wrong. Nevertheless, We believe this issue could be well addressed by enhancing CLIP models with a stronger text encoder, and we will make explorations in future work.

7 Related Work

Vision-language few-shot learning. Leveraging aligned caption data, vision-language models pre-trained by an image-text discriminative loss have recently enabled strong zero-shot generalization on image classification and cross-modality retrieval tasks (Jia et al., 2021; Radford et al., 2021). Different from the discriminative manner, Tsimpoukelli et al. (2021) prompt a large frozen language model with vision prefix in a generative way, which is the first vision-language few-shot model.

Language model prompting. This work is also inspired by the line of research in language model prompting (Liu et al., 2021). Initialized by the GPT series (Radford et al., 2018, 2019; Brown et al., 2020), prompting has become a popular manner to mining knowledge from pre-trained language models (Petroni et al., 2019) in a zero-shot or few-shot way (Shin et al., 2020; Gao et al., 2021; Qin and Eisner, 2021). Besides mining knowledge from the language model, PET work (Schick and Schütze, 2021a,b) presents a semi-supervised prompting method for improving few-shot language understanding performance.

8 Conclusions

In this work, we empirically studied how to transfer CLIP models into vision-language understanding tasks. We first explored the CLIP models' zero-shot VQA capability by leveraging language prompts and further proposed a parameter-efficient fine-tuning method to boost the few-shot performance. We also demonstrate a zero-shot cross-modality transfer capability of CLIP models on the visual entailment task. Experiments and analyses on VQAv2 and SNLI-VE confirm that the CLIP models can be good VL few-shot learners.

Acknowledgements

Haoyu Song, Wei-Nan Zhang, and Ting Liu are supported by the Science and Technology Innovation 2030 Major Project of China (No.2020AAA0108605), National Natural Science Foundation of China (No.62076081, No.61772153, and No.61936010), and Natural Science Foundation of Heilongjiang (No.YQ2021F006).

References

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

Elad Ben Zaken, Shauli Ravfogel, and Yoav Goldberg. 2021. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. *arXiv preprint: 2106.10199*.

Samuel Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901.

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020. Uniter: Universal image-text representation learning. In *European conference on computer vision*, pages 104–120. Springer.

Dorottya Demszky, Kelvin Guu, and Percy Liang. 2018. Transforming question answering datasets into natural language inference datasets. *arXiv preprint: 1809.02922*.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of the ACL 2021 (Volume 1: Long Papers)*, pages 3816–3830, Online. Association for Computational Linguistics.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. 2017. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6904–6913.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc V Le, Yunhsuan Sung, Zhen Li, and Tom Duerig. 2021. Scaling up visual and vision-language representation learning with noisy text supervision. In *International Conference on Machine Learning*.

Wonjae Kim, Bokyung Son, and Ildoo Kim. 2021. Vilt: Vision-and-language transformer without convolution or region supervision. In *International Conference on Machine Learning*.

Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Alexander Kolesnikov, et al. 2020. The open images dataset v4. *International Journal of Computer Vision*, 128(7):1956–1981.

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

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

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *Advances in Neural Information Processing Systems*, 32:13–23.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of EMNLP-IJCNLP 2019*, pages 2463–2473.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A Python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*.

Guanghui Qin and Jason Eisner. 2021. Learning how to ask: Querying lms with mixtures of soft prompts. In *Proceedings of NAACL 2021*, pages 5203–5212.

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

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67.

Timo Schick and Hinrich Schütze. 2021a. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the EACL*, pages 255–269.

Timo Schick and Hinrich Schütze. 2021b. It's not just size that matters: Small language models are also few-shot learners. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2339–2352.

Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. 2018. 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*, pages 2556–2565.

Sheng Shen, Liunian Harold Li, Hao Tan, Mohit Bansal, Anna Rohrbach, Kai-Wei Chang, Zhewei Yao, and Kurt Keutzer. 2022. How much can clip benefit vision-and-language tasks? In *International Conference on Learning Representations*.

Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of EMNLP 2020*, pages 4222–4235.

Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. 2020. Vl-bert: Pre-training of generic visual-linguistic representations. In *International Conference on Learning Representations*.

Maria Tsimpoukelli, Jacob Menick, Serkan Cabi, SM Eslami, Oriol Vinyals, and Felix Hill. 2021. Multimodal few-shot learning with frozen language models. In *NeurIPS*, volume 34.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Wenhui Wang, Hangbo Bao, Li Dong, and Furu Wei. 2021. Vlmo: Unified vision-language pre-training with mixture-of-modality-experts. *arXiv preprint: 2111.02358*.

Shijie Wu and Mark Dredze. 2019. Beto, bentz, becas: The surprising cross-lingual effectiveness of bert. In *Proceedings of EMNLP 2019*, pages 833–844.

Ning Xie, Farley Lai, Derek Doran, and Asim Kadav. 2019. Visual entailment: A novel task for fine-grained image understanding. *arXiv preprint: 1901.06706*.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mt5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of NAACL 2021*, pages 483–498.

Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. 2014. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *Transactions of the Association for Computational Linguistics*, 2:67–78.

Zhou Yu, Jun Yu, Yuhao Cui, Dacheng Tao, and Qi Tian. 2019. Deep modular co-attention networks for visual question answering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6281–6290.

Pengchuan Zhang, Xiujun Li, Xiaowei Hu, Jianwei Yang, Lei Zhang, Lijuan Wang, Yejin Choi, and Jianfeng Gao. 2021. Vinvl: Revisiting visual representations in vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5579–5588.

Appendix

A Datasets Statistics

Datasets	*# Train*	*# Valid*	*# Test*	*# Vocab*
VQAv2	443,757	214,354	-	19,174
	82,783	40,504	-	
SNLI-VE	529,527	17,858	17,901	32,191
	29,783	1,000	1,000	

Table 7: Basic statistics of the two datasets. The upper is the number of examples, and the lower is the number of distinct images. And *# Vocab* is the vocabulary size.

B Details of Implementation

B.1 Zero-shot Model Briefs

In our experiments, we leverage two kinds of pre-trained models: the CLIP variants and the T5. We brief these models as follows.

For the CLIP models, the text encoder is always a transformer, but its hidden size varies according to the size of visual encoders. And there are two architectures of visual encoders, including the vision transformer (ViT) and ResNet.

- **CLIP ViT-B/16**: both the text and visual encoders are 12-layer, 512-hidden transformers.
- **CLIP RN101**: the text encoder is a 12-layer transformer, and the visual encoder is ResNet101, both with a hidden size of 512.
- **CLIP RN50x16**: the text encoder is a 12-layer transformer, and the visual encoder is ResNet50x16, both with a hidden size of 768.

All CLIP models we used are from the official CLIP repository[2]. For the language model T5, we use a publicly available T5_{large} checkpoint from the Huggingface repository[3]. The T5_{large} has 24 hidden layers, 16 self-attention heads, 1024 hidden size, and a total of 770M parameters. It is trained on Colossal Clean Crawled Corpus (C4). Note that the T5 model had not been trained or finetuned under both few-shot and zero-shot settings.

B.2 Hyperparameters

We report the hyperparameter settings of few-shot CLIP training in Table 8. We apply the same set of hyperparameters to fine-tune both ResNet CLIP and ViT CLIP.

[2]https://github.com/openai/CLIP

[3]https://huggingface.co/models

Hyperparameters	Value
Training epochs	30
Batch size	8
Initial temperature	0.07
Maximum temperature	100.0
Adam ϵ	1e-8
Adam β	(0.9, 0.999)
Learning rate	2e-5
Gradient clipping	2.0
Weight decay	0.001
Number of filtered answers	200

Table 8: Hyperparameters used for CLIP models in few-shot learning.

Hyperparameters	Value
Layers	3
Layer Dimension	1024-128-3
Training epochs	20
Adam ϵ	1e-8
Adam β	(0.9, 0.999)
Gradient clipping	2.0
Learning rate	{1e-6, 3e-6, 5e-6}
Batch size	{32, 64, 128}
Dropout	{0, 0.1, 0.4}

Table 9: Hyperparameters of the MLP classifier in zero-shot language to vision transfer.

The hyperparameters used for the MLP classifier in the visual entailment task are reported in Table 9. We performed grid searches on the combination of the learning rate, batch size, and dropout. The CLIP variants reached the best performances under different parameter combinations.

Template Generation	
Hyperparameters	Value
Number of beams	20
Number of returned sequences	10
Max returned span length	30
Answer Filtering	
Hyperparameters	Values
Batch size	128
Number of beams	200
Number of returned sequences	200
Max returned span length	6
Max number of demonstration	16

Table 10: Hyperparameters used for T5 in template generation and answer filtering.

Table 10 shows the hyperparameter configurations for T5's conditional generation, which is leveraged to generate the masked template and filter answers.

CLIP	*# Bias*	*# Normalize*	*# BiNor*	*# All*
CLIP$_{\text{RN101}}$	127,488	123,392	189,184	100M
CLIP$_{\text{RN50x16}}$	209,088	132,160	319,488	229M
CLIP$_{\text{ViT-B/16}}$	171,008	65,536	203,776	149M

Table 11: Statistics of different type of parameters in CLIP models.

B.3 The Number of Learnable Parameters

Table 11 shows the number of different type of learnable parameter in CLIP models. The counting of Bias and Normalization share the β in Eq.7. The numbers of BiNor parameters are about 0.2M to 0.3M, accounting for less than 0.3% of all parameters.

C Few-shot Training Procedure

Algorithm 1 CLIP models few-shot training.

```
Input: $\mathbb{V}$: visual encoder, ResNet or ViT; $\mathbb{T}$: text encoder,
    Transformer; $I$: few-shot images; $P$: few-shot prompts;
    $A$: few-shot answers; $\tau$: learned temperature parameter;
    $N$: max iterations; Adam: optimizer;
Output: Few-shot CLIP model.
 1: initial $epoch = 0$, freeze parameters in $\mathbb{V}$ and $\mathbb{T}$ except
    bias and normalization;
 2: repeat
 3:     Sample C-way K-shot $E$ from ($I$,$P$,$A$);
 4:     Split $E$ into support set and query set;
 5:     for all minibatch (i,p,a) in support set do
 6:         I$_f$ = $\mathbb{V}$(i), T$_f$ = $\mathbb{T}$(p);
 7:         I$_e$ = norm(I$_f$), T$_e$ = norm(T$_f$);
 8:         logits = $\tau$ * dot($I_e$, $T_e$);
 9:         labels = map(a, p);
10:         loss = cross_entropy(logits, labels);
11:         Adam.step();
12:     end for;
13:     $epoch = epoch + 1$;
14:     Evaluate on query set;
15: until ($epoch \geq N$).
```

D Examples of Template Generation

In this section, we showcase several template generation examples to illustrate how the proposed method works. Since we have introduced how to convert a question into a masked template by demonstrating examples to the T5 (§ 3.1), here we directly present several examples in Table 12. These examples are sampled from five different question types and also cover the three answer types. As shown in Table 12, a single demo in the demonstration consists of a question and an answer with the [mask] token. Notice that the [mask] token is only a placeholder rather than a real mask in the pre-trained language models. Different from the <*extra_id_0*> in T5 that represents a corrupted span, the [mask] is used to inform the T5 where the answer words should be placed. After seeing several examples in the demonstration, the powerful T5$_{\text{large}}$ model could capture the conversion pattern in each type of question and perfectly complete most conversions without ignoring the subtle grammars. Once the masked template is generated, we could infill the [mask] place with answer words and then carry out further processing. The processing for the *yes/no* type is a little different: as it is a binary task, we directly generate a positive prompt and a negative prompt, rather than masked templates, for the *yes* and *no*, respectively.

Question Type	what color is
Demonstration	What color is the floor of this area? The color of floor of this area is [mask]. What color is the pillow the cat is on? The color of the pillow the cat on is [mask]. What color is the child's shorts? The color of the child's shorts is [mask]. What color is the lettering on the business sign? The color of the lettering on the business sign is [mask].
Question **Generated Template**	What color is the fence behind the man? The color of the fence behind the man is [mask].
Question **Generated Template**	What color is the statue near the building? The color of the statue near the building is [mask].

Question Type	why is the
Demonstration	Why is the ground surface near the train a different color? The ground surface near the train is in a different color because of [mask]. Why is the cat under an umbrella? The cat under an umbrella is because of [mask]. Why is the laptop sitting above a larger keyboard? The laptop is sitting above a larger keyboard because of [mask]. Why is the car being towed? The car is being towed because of [mask].
Question **Generated Template**	Why is the little boy having fun? The little boy is having fun because of [mask].
Question **Generated Template**	Why is the elephant's trunk two color's? The elephant's trunk is two colors because of [mask].

Question Type	which
Demonstration	Which utensil is on the table in the foreground? The [mask] utensil is on the table in the foreground. Which way is the train going? The [mask] way is the train going. Which hand holds the racket? The [mask] hand holds the racket. Which foot is lifted in the air? The [mask] foot is lifted in the air.
Question **Generated Template**	Which hot dog has a larger variety of toppings? The [mask] hot dog has a larger variety of toppings.
Question **Generated Template**	Which operating system is being used on this computer? The [mask] operating system is being used on this computer.
Question **Generated Template**	Which side of the room is the television probably on? The [mask] side of the room is the television probably on.

Question Type	how many
Demonstration	How many unopened rolls of paper are in the picture? There are [mask] unopened rolls of paper in the picture. How many engines does the closest airplane have? The closest airplane has [mask] engines. How many different types of doors are visible? There are [mask] different types of doors visible. How many people are wearing plaid shirts? There are [mask] people wearing plaid shirts.
Question **Generated Template**	How many people are participating in the eating contest? There are [mask] people participating in the eating contest.
Question **Generated Template**	How many cabinets have been installed? There are [mask] cabinets installed.
Question **Generated Template**	How many people in this picture are wearing glasses? There are [mask] people wearing glasses.

Question Type	does this
Demonstration	Positive: Does this food look burnt? This food looks burnt. Does this appear to be a noisy environment? This appears to be a noisy environment. Negative: Does this boat have an engine? This boat has no engine. Does this type of fruit change color? This type of fruit does not change color. Does this animal produce dairy products? This animal does not produce dairy products. Does this pizza look hot? This pizza does not look hot.
Question **Generated Prompts**	Does this look like a happy occasion? Yes$\rightarrow$ This looks like a happy occasion No $\rightarrow$ This does not look like a happy occasion
Question **Generated Prompts**	Does this man have both of his skis on? Yes$\rightarrow$ This man has both of his skis on No $\rightarrow$ This man does not have both of his skis on
Question **Generated Prompts**	Does this transportation run on gasoline? Yes$\rightarrow$ This transportation runs on gasoline No $\rightarrow$ This transportation does not run on gasoline

Table 12: Examples of generating masked templates. The demonstrations are defined for each type of question and are demonstrated to the T5. For the binary *yes/no* type, we directly generate positive prompt for *yes* and negative prompt for *no*. Full question types are available at https://github.com/GT-Vision-Lab/VQA/tree/master/QuestionTypes.