Supplementary Materials

A. Implementation Details

After obtaining entities $\{e_1, e_2, e_3, ...\}$, we can construct the entity-aware hard prompt. To this end, we randomly drop certain entities (*e.g.*, e_2) and insert the remaining ones into a prompt template, resulting in a sentence like "There are $e_1, e_3, ...$ in the image." Subsequently, we employ the tokenizer and word embeddings from GPT-2 to convert this sentence into dense vectors $\mathbf{h} \in \mathbb{R}^{n\times d}$. Here, n represents the length of vector $\mathbf{h}$, and $d = 768$ indicates the dimension of GPT-2's latent space. The soft prompt, generated by the Transformer-based projector, is denoted as $\mathbf{s} \in \mathbb{R}^{m\times d}$, where m corresponds to the length of the soft prompt $\mathbf{s}$. Consequently, the prompt fed into GPT-2 can be represented as $\mathbf{p} = \{\mathbf{s}; \mathbf{h}\}, \mathbf{p} \in \mathbb{R}^{(m+n)\times d}$, where $\{;\}$ denotes concatenation.

The auto-regressive objective is employed to train parameters θ of the decoder. It is defined as follows:

$$\mathcal{L}_{obj} = -\frac{1}{|\mathbf{w}|}\sum_{i=1}^{|\mathbf{w}|} \log p(w_i|\mathbf{s}; \mathbf{h}; \mathbf{w}_{\leq i} : \theta) \tag{2}$$

We train ViECap on various source domains with the hyperparameters shown in Tab. 9. During inference across different target domains, we retrieve visual entities using the frozen CLIP, which can be formulated as:

$$p_i = \frac{\exp(\mathrm{sim}(I, T_i)/\tau)}{\sum_{j=1}^{N}\exp(\mathrm{sim}(I, T_j)/\tau)} \tag{3}$$

where $\mathrm{sim}(I, T_i)$ denotes the cosine similarity between image I and class name T_i, τ and N refer to the temperature and the size of vocabulary, respectively. We choose the top k class names with p_i greater than threshold p_{thres} as retrieved entities. For all evaluations on cross-domain captioning, we leverage the same values of k, p_{thres}, and τ (*i.e.,* 3, 0.2, and 0.01, respectively). For evaluations on in-domain captioning, we set k, p_{thres}, and τ to 3, 0.4, and 0.01 for COCO; 3, 0.3, and 0.01 for Flickr30k; 2, 0.1, and 0.007 for Flickrstyle10K.

Hyperparameters	**COCO**	**Flickr30k**	**FlickrStyle10K**
Epochs	15	30	25
Batch size	80	80	128
Learning rate	$2e^{-5}$	$2e^{-5}$	$3e^{-4}$
Masking rate	0.4	0.4	0.4

Table 9. Training hyperparameter.

B. Unsupervised Metric

Furthermore, we report the captioning performance using the unsupervised metric, *i.e.*, CLIP score (CLIP-S), to further validate the effectiveness of ViECap. We compare with other text-only methods (*i.e.*, MAGIC, CapDec) in cross-domain captioning to assess the transferability of our model. As presented in Tab. 10, ViECap outperforms all other methods in cross-domain captioning by a large margin, indicating its robustness in handling domain shifts within diverse images.

Methods	**COCO ⇒ NoCaps val**				**COCO ⇒**	**Flickr30k**
	In	Near	Out	Overall	**Flickr30k**	**⇒ COCO**
MAGIC	0.665	0.664	0.658	0.662	0.686	0.661
CapDec	0.711	0.701	0.671	0.692	0.737	0.694
ViECap	**0.738**	**0.751**	**0.764**	**0.754**	**0.761**	**0.744**

Table 10. Quantitative results in the cross-domain captioning using the unsupervised metric CLIP-S.

C. Hard Prompt Variants

We explore the influence of different prompt templates on ViECap's captioning performance. As shown in Tab. 11, ViECap shows minor sensitivity to changes in prompt templates, even when using a step-by-step hard prompt variant (variant 3). We speculate that the model is more effective for the altered parts in the template (*i.e.,* visual entities) due to fine-tuning GPT-2.

D. Soft Prompt Length

We investigate the impact of different lengths of soft prompts on the captioning performance of ViECap. As shown in Tab. 12, we arrive at the same conclusion as the experiment on hard prompt variants, *i.e.*, increasing the length of soft prompts does not significantly improve the performance of ViECap while fine-tuning GPT-2.

E. Time Cost

Tab. 13 compares the time cost of CLIP-based retrieval and detector-based retrieval (*i.e.*, Faster R-CNN). We calculate the average time cost of processing 100 images from the COCO testing set on a single NVIDIA TITAN V GPU. For detector-based retrieval, we use Faster R-CNN with the backbone of ResNet-101[2]. The results indicate that our model is four times faster than Faster R-CNN, from processing a single image to obtaining the detected entities. Note that the integrating of additional entity-aware hard prompts only incurs a minor time increase of 0.57 ms compared to CapDec while significantly outperforming CapDec by a large margin across various benchmarks.

F. Vocabularies

The quality of the vocabulary impacts the retrieval performance of CLIP and the transferability of ViECap. For

[2]We utilize the model and pre-trained weights from https://github.com/open-mmlab/mmdetection

Hard prompt variants	COCO Test	NoCaps val In	Near	Out	Overall
Default: "There are ... in the image."	92.9	61.1	64.3	65.0	66.2
Variant 1: "There are ... in the scene. The image shows"	92.6	59.2	63.5	64.2	65.2
Variant 2: "A photo of ..., a caption to describe this image is"	92.3	60.3	63.4	64.6	65.3
Variant 3: "To describe this image, let us think step by step. In this image, we can see ..., so a sentence to describe this picture is"	91.5	59.2	62.7	64.9	64.9

Table 11. Results on variants of different hard prompt templates. "..." denotes the parts to be filled by visual entities.

Soft prompt length	COCO Test	NoCaps val In	Near	Out	Overall
Length: 10	92.9	61.1	64.3	65.0	66.2
Length: 20	92.3	60.3	63.8	64.5	65.6
Length: 30	92.3	60.8	63.9	65.3	66.0
Length: 40	92.3	60.2	64.1	65.0	65.9

Table 12. Results on different lengths of soft prompts.

Models	**Encoding + Retrieval** (ms)	**Decoding** (ms)
ViECap	20.39 + 0.57	127.99
Faster R-CNN	86.76	-

Table 13. The average time cost of captioning 100 COCO images using ViECap and Faster R-CNN during inference. Encoding denotes the time cost of encoding a single image to features. Retrieval refers to the average speed of detecting entities from features. Decoding refers to the average time cost of generating a sentence by the decoder.

results reported in this paper, we leverage the COCO vocabulary for the COCO testing set and the VGOI vocabulary for all other datasets. Visual Genome contains various class name annotations, but they suffer from noise and incorrect annotations. We select class names consisting of a single word to construct Visual Genome vocabulary (17069). Zhang *et al.* [54] clean Visual Genome to build a clean corpus (*i.e.*, VGOI vocabulary), which comprises 1848 class names. We also construct the COCO (80) vocabulary and the Open Image (601) vocabulary using class names from the corresponding class annotations.

The NoCaps dataset contains three domains: 1) *in-domain* only contains COCO classes, 2) *near-domain* contains both COCO and Open Image classes, and 3) *out-of-domain* only contains Open Image classes. Tab. 14 shows the results of NoCaps on different vocabularies. A specific domain of the captioning dataset benefits from a specific vocabulary (*e.g.*, COCO vocabulary achieves the best performance in the *in-domain* of NoCaps, and Open Image vocabulary achieves the best performance in the *out-of-domain* of NoCaps). However, when aiming for transferability to a novel domain where a specific vocabulary is not attainable, a large, diverse, and clean vocabulary describing various classes becomes crucial. As shown in Tab. 14, the VGOI vocabulary achieves a great trade-off between the *in-domain* and *out-of-domain* captioning performance. Notably, a large but noisy vocabulary, as seen in the Visual Genome vocabulary in Tab. 14, does not significantly improve ViECap's performance.

Vocabulary	Size	NoCaps val In	Near	Out	Overall
COCO vocabulary	80	63.6	51.0	22.7	44.9
Open Image vocabulary	601	59.5	66.8	69.4	69.2
VGOI vocabulary	1848	61.1	64.3	65.0	66.2
Visual Genome vocabulary	17069	56.8	50.5	41.9	50.0

Table 14. Results on NoCaps using different vocabularies.

G. Datasets

COCO and Flickr30k are commonly used benchmarks for evaluating image captioning models. We divide these datasets into three parts (*i.e.*, training, validation, and testing set) following the Karpathy *et al.* split [23]. This results in 113,000, 5,000, and 5,000 samples for COCO and 10,300, 1,000, and 1,000 samples for Flickr30k, respectively.

NoCaps is divided into three domains, evaluating the capability of models to describe novel objects in images - *in-domain* consists solely of COCO classes, *near-domain* includes both COCO and novel classes, and *out-of-domain* comprises only novel classes. As suggested by OSCAR [29], we assess the models using only the validation set.

Additionally, FlickrStyle10K assesses the task of generating captions with new styles, *i.e.*, "romantic" and "humorous". Since only 7,000 training samples are publicly available, following the approach used in MemCap [56], we randomly sample 6,000 captions as our training set, while the remaining image-text pairs constitute our testing set.

H. Visualizations

Additional visualization results are presented in Fig. 5, showcasing the remarkable transferability of ViECap. Our model excels not only in describing novel objects but also in generating captions for images with various styles.

Here, we leverage weights trained on the COCO training set for captioning. The first row displays the captioning results on the COCO testing set, demonstrating the successful description of in-domain objects by both CapDec and ViECap. The second row presents results on the *out-of-domain* of NoCaps, showcasing ViECap's ability to generate high-quality texts related to unseen objects.

Rows 3 to 7 illustrate captioning results for Office-Home [39], a benchmark dataset for image domain adaptation, which comprises four different styles of image domains: 1) Art, artistic images in the form of sketches, paintings, ornamentation, *etc*., 2) Clipart, collection of clip art images, 3) Product, images of objects without a background, and 4) Real-World, images of objects captured with a regular camera. We evaluate the captioning performance of ViECap across these diverse image styles, using the first image from different categories in Office-Home (*i.e.*, we do not choose a specific image but simply use the first image of each class in the dataset). Despite a few incorrect captions, ViECap is capable of describing different styles of images with reasonable descriptions in most cases, highlighting that our captioning model can effectively transfer to various styles of images and generate appropriate captions related to images.

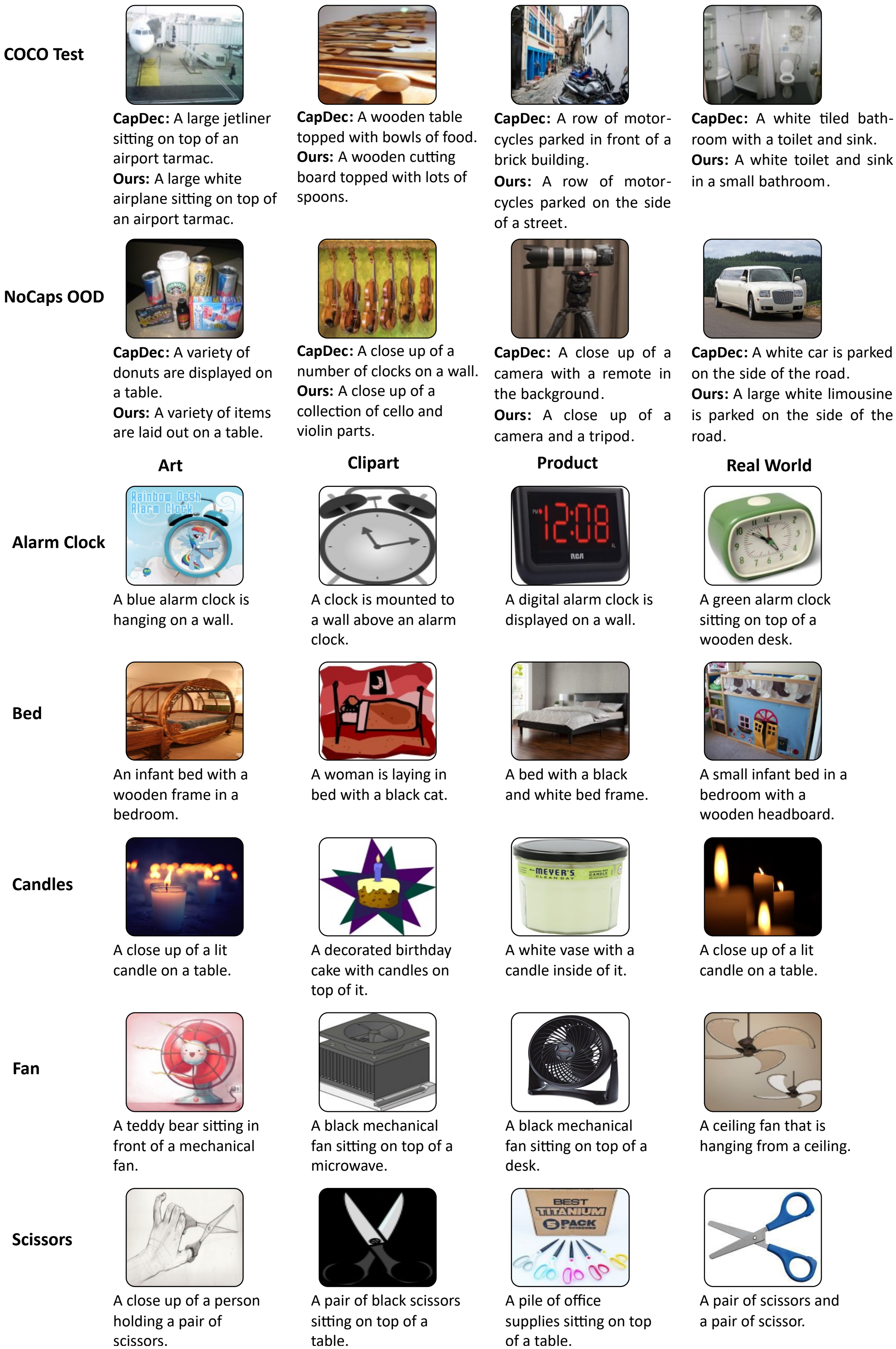

Figure 5. More visualization results of ViECap on in-domain captioning (row 1), cross-domain captioning (row 2), and image-domain-adaptation captioning (row 3-7).