INJECTING IMAGE DETAILS INTO CLIP'S FEATURE SPACE

Anonymous authors
Paper under double-blind review

ABSTRACT

Although CLIP-like Visual Language Models provide a functional joint feature space for image and text, due to the limitation of the CILP-like model's image input size (e.g., 224), subtle details are lost in the feature representation if we input high-resolution images (e.g., 2240). In this work, we introduce an efficient framework that can produce a single feature representation for a high-resolution image that injects image details and shares the same semantic space as the original CLIP. In the framework, we train a feature fusing model based on CLIP features extracted from a carefully designed image patch method (Complete Cover) that can cover objects of any scale, weakly supervised by image-agnostic class prompted queries. We validate our framework by retrieving images from class prompted queries on the existing real-world and synthetic datasets, showing significant performance improvement on these tasks. Furthermore, to fully demonstrate our framework's detail retrieval ability, we construct a CLEVR-like synthetic dataset called CLVER-DS, which is fully annotated and has a controllable object scale.

1 INTRODUCTION

Text-to-image retrieval task is to retrieve relevant images given a text query. The query can either be a sentence describing the whole image or an object name focusing on a small part of the image.

For instance, suppose we use CLIP to retrieve images containing a red helmet and use "red helmet" with a prompt as a text query; what appears first is always a sizeable red helmet right in the middle of the image. However, images showing people wearing red helmets on a football court are also what we want, and red helmets can be tiny in the image. Since CLIP was trained to match an image as a whole to a text description, it is hard for CLIP to retrieve the above image by just using "red helmet" with a prompt as a text query. This means images can be retrieved if their main parts match the text description. However, in many practical scenarios, we need to use a word to retrieve all related images in a database. As illustrated in Figure 1

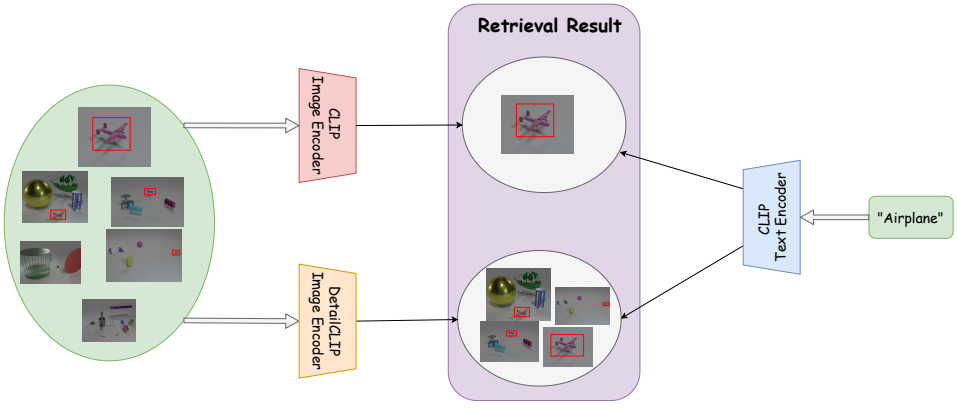


Figure 1: Retrieval results from CLIP model and our DetailCLIP model. DetailCLIP retrieves more images with small target objects.

The problem motioned above can be partially solved by dividing each image into many small patches and doing the retrieval task based on the feature of these small patches. However, this multi-feature method cannot obtain a useful feature directly for downstream tasks or end-to-end training. Models such as Crop-CLIP by vijishmadhavan (2022) completes the retrieval by first detecting the objects in the image with a detector YOLOV5. However, the capacity of the detector severely constrains the model performance. For example, the model behaves unpredictably when encountering objects from unseen classes, and the information about these objects will be lost before entering the retrieval stage. Models such as MDETR by Kamath et al., XDETR by Cai et al. (2022), FILIP by Yao et al. (2021) achieve many improvements in detail retrieval by retraining a model with multi-modality data and mining the fine-grained relationship during the training. However, they all require enormous training data and computational resources to complete.

In this paper, we propose a framework, "Detail Injected CLIP (**DetailCLIP**)," which can solve the above problem at a small cost. In this framework, we could inject detailed information in the image into a single feature that can be directly used for end-to-end training, with the cost of training a small transformer. Our main contributions are summarized as follows:

- We focus on the detailed class-with-prompt text-to-image retrieval task and propose an efficient framework **DetailCLIP** which can produce a single image feature with detailed information. We test the above text-to-image retrieval result on MSCOCO, LVIS, and synthetic datasets, and our framework outperforms current vision-language models.
- We design an image patching scheme "Complete Cover(**CC**)". **CC** patches can theoretically cover objects of any scale meanwhile reduce the redundant patches.
- We construct a retrieval benchmark based on the CLEVR Johnson et al. (2017) and ShapeNet Chang et al. (2015) 3D objects, called "CLEVR of Different Scales (**CLEVR-DS**)". With this full annotated and object scale controllable dataset, our framework's retrieval result outperforms the current method by a large margin.

We arrange subsequent chapters of this paper as follows. Section 2 gives an introduction to the related works. Section 3 gives a detailed description of the "Complete Cover (**CC**)" method proposed in this paper, and our **DetailCLIP** framework with corresponding loss. Section 4 introduces the benchmark proposed in this paper. In section 5, we go through many experiments to verify our framework's superiority.

2 Related Work

2.1 Vision-Language model overview

Current Vision-Language Model can be divided into different types through task objectives. CLIP Radford et al., ALIGN Jia et al., and Flamingo Alayrac et al. (2022) aligns the textual and visual information into a shared semantic space through contrastive learning task. There are other works for different tasks such as MDETR Kamath et al. (object detection), PhraseCut Wu et al. (segmentation), Florence Yuan et al. (Foundation Model), etc.

2.2 CLIP-like models

CLIP Radford et al. is a neural network trained by OpenAI on various image-text pairs. Given an image, CLIP predicts the most relevant text snippet or Vice Versa. However, the CLIP feature has limitations, such as being easy to fail on typographic attacks or fine-grained concepts. Besides, Zhou et al. (2021) argues that the quality of text feature is highly related to prompt methods (way to perform augmentation on the class label to generate text sentences). In SLIP by Mu et al. (2021), different views of each input image are used for text supervision and image self-supervision. It demonstrates that image self-supervision would benefit the performance of CLIP. At the same time, DeCLIP Li et al. (2021) adds several additional training objectives to CLIP to improve the performance of language supervision in the form of CLIP. Most recently, Li* et al. (2022) proposed GLIP, which unifies object detection and phrase grounding for pre-training and can learn object-level, language-aware, and semantic-rich visual representations.

2.3 VISION LANGUAGE MODEL APPLICATIONS

In order to capture the fine-grained alignment between image and text, RegionClip proposed by Zhong et al. (2021) creates a pool of object concepts from the text corpus and uses a pre-trained CLIP model to align a concept with an image region making pseudo labels. They use region-text pairs and ground-true image-text pairs to pretrain a vision-language model. RegionClip shows a better ability to recognize region objects and successfully transferred to the open-vocabulary object detection task, but without analysis on whether it can detect the small object in an image. Xu et al. (2022) used grouping mechanism into deep networks, which allows semantic segments to emerge automatically with only text supervision. GroupViT Xu et al. (2022) learns to group semantic regions together and successfully transfers to the semantic segmentation task.

3 METHODOLOGY

3.1 MOTIVATION AND EFFECTIVE SCALE SENSITIVITY

CLIP's capability to retrieve an object deteriorates as the object's size becomes small. We perform our experiment on LVIS dataset Gupta et al. (2019) and use

$$r_{max} = \frac{Maximum\ Area\ of\ the\ Objects\ in\ the\ image}{Area\ of\ the\ Image} \tag{1}$$

with different values as upper bound (threshold) to create subsets of LVIS. We choose LVIS because it has more categories than COCO Lin et al. (2014) and has more annotations for small objects. As shown in 3.1, CLIP performance monotonically decreases as the object size decreases.

Table 1: Performance of CLIP text-image retrieval task with different LVIS subsets.

r_{max}	† Recall@1	Recall@3	Recall@5
10^{0}	8.63%	15.19%	18.60%
$10^{-0.5}$	7.52%	13.87%	17.45 %
10^{-1}	5.75%	11.28%	14.66 %
$10^{-1.5}$	4.92%	9.61%	12.16 %
10^{-2}	3.98%	8.76%	11.48 %

† Refer to section 4.3 for the metric detail

We define the Effective Scale Sensitivity of CLIP-like models as the minimum object occupying percentage in an image that CLIP-like models can retrieve. We need to input images within the sensitivity of CLIP-like models. A natural thought in solving this problem is slicing an image into small patches and retrieving objects on those patches. In following sections, we propose our method to solve above problem with small cost.

3.2 PROBLEM DEFINITION

In this section, we will give a formal problem definition on Detail Injection with CLIP-like models, image, and a set of image patches. Suppose we generate p patches from a image $\mathbf{X}$, and we denote the set of image patches as $x_i \in \mathbf{X}$, where $i \in \{1, ..., p\}$. Then $\mathcal{F} : \mathbb{R}^{c\times w\times h} \rightarrow \mathbb{R}^d$ represents CLIP-like models image encoder, and the d dimensional feature u_i extracted from a single image patch x_i can be represented as:

$$u_i = \mathcal{F}(x_i), \tag{2}$$

We denote the set of image patch features for a given image as $\mathbf{U} = \{u_i\}$, where $i \in \{1, ..., p\}$. We define our fusing model as $\mathcal{D} : \mathbb{R}^{p\times d} \rightarrow \mathbb{R}^d$. The DetailCLIP feature v is obtained from:

$$v = \mathcal{D}(\mathbf{U}),\ \text{where}\ v \in \mathbb{R}^d \tag{3}$$

3.3 COMPLETE COVER

We will propose a patch generation scheme in this subsection. Consider the side length of an image to be n, and the number of possible patches to cover all possible objects is at $O(n^4)$ level, which is unbearable. We come up with the "Complete Cover (**CC**)" method to eliminate redundant patches while covering all possible objects. The schematic diagram of **Complete Cover** is Figure 2.

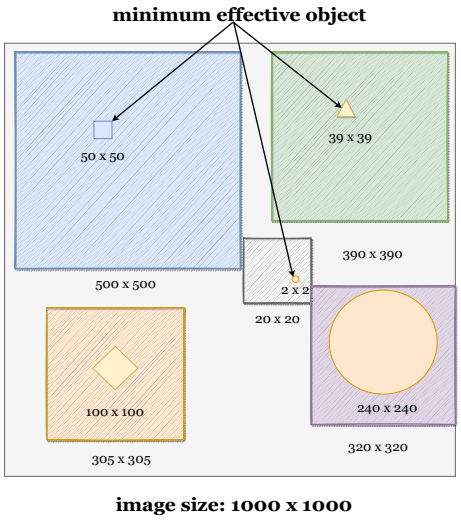


Figure 2: Illustration of Patch Selection of Complete Cover.Minimum effective object means the minimum object that can be retrieved by CLIP from the patch. Patches with different shapes will slide the whole image to cover objects equal to or bigger than the minimum effective size.

Firstly, we define the meaning of **cover**. Let Q be the set of pixels of a patch, P be the set of pixels in the bounding box of an object. In order for the patch to include objects while maintaining the capability to retrieve the patch, we define Q covers P as follows:

$$
\begin{aligned}
&Q \text{ covers } P \coloneqq C(Q,P) = \begin{cases} \text{True}, & \forall p=(x,y)\in P \rightarrow p \in Q \text{ and } |P| > c^2 \cdot |Q| \\ \text{False}, & \text{Otherwise} \end{cases} \\
&\text{where}: \ |\cdot| \text{ is the number of pixels in } \cdot \\
&\qquad\quad \text{c is effective scale sensitivity defined in our paper}
\end{aligned} \tag{4}
$$

Secondly, we call the set of all possible P in the image as S_{full}. In "Complete Cover (**CC**)" scheme, we design a greedy algorithm to generate a set of Q as S_{cc}, which satisfies:

$$
\forall\, P \in S_{\text{full}}, \exists\, Q \in S_{\text{cc}}, s.t.\, C(Q,P) = 1 \tag{5}
$$

The specific patch selection method is as follows: Given a effective scale sensitivity c, assume the full image pixel set is $Q_0 \in S_{\text{cc}}$, all the objects bounding box pixel set $P_0 \in S_{\text{full}}$ it can cover satisfies:

$$
C(Q_0, P_0) = 1, \forall P_0 \in S_{\text{full}}, \text{if } |P_0| > c^2 \cdot |Q_0|, \tag{6}
$$

Without loss of generality, consider $P \in S_{\text{full}}$ is square and P_0 with side length of a. In order to cover $P_1 \in S_{\text{full}}$ whose side length is $a/c-1$, we use a greedy way to get $Q_1 \in S_{\text{cc}}, s.t. C(Q_1, P_1) = 1$ by passing a global sliding window with a side length of $a-c$ and a step size of $a/c-2$. Repeat this procedure until we have the patches that can cover objects with side lengths ranging from a/c to $a/c-n$, where $n = a/c-1$.

Our **CC** method can better retain detailed information than simply slicing the image into non-overlapped, equal-sized patches by Dosovitskiy et al. (2020). **CC** face a trade-off of completeness and computational complexity with different c. A reasonable choice of c needs to generate a bearable number of patches while ensuring the retention of detailed information.

3.4 MODEL & LOSS

We will propose a fusion model and a proxy loss in this subsection. Assume the patch selection method is determined. We extract the features of each image patch, and then we use CLIP prompts to prompt all class labels into sentences and extract the features of each sentence. This section will discuss fusing multiple patch features into a single feature and using the image-agnostic text feature as a proxy to inject detailed information. The overview architecture of the framework is illustrated in Figure 3.

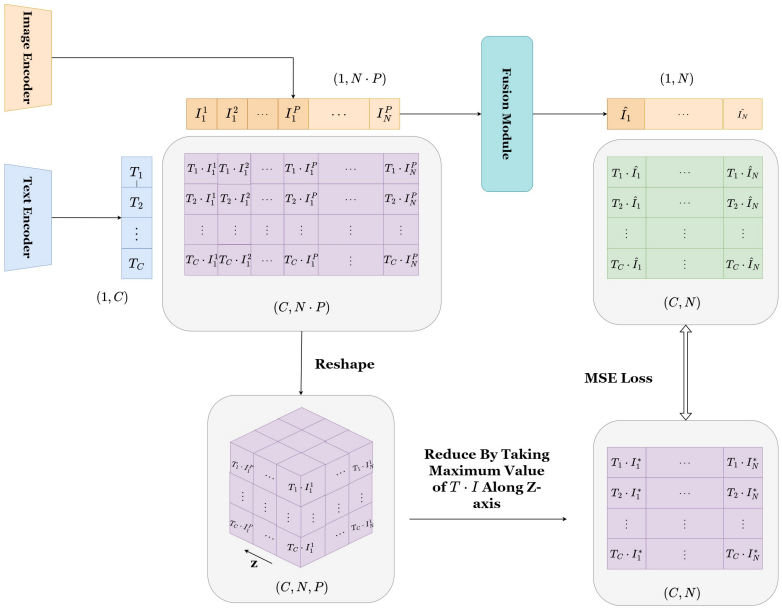


Figure 3: DetailCLIP framework with Query Proxy Loss.

3.4.1 FUSING MODEL

The essence of the DetailCLIP is injecting features from different patches into one new feature while keeping details as much as possible. We have many options to implement the fusing model, such as learning a weight to average the patch features, a Linear Projection, an MLP. We choose to implement the fusing model with a small transformer. Specifically, the input of this fusing model consists of two parts, the feature for all patches and the feature for the entire image. We use the former as the input source and the latter as the input target in the transformer.

3.4.2 QUERY PROXY LOSS

For a small object in an image, there will be one patch feature that contains the most information about the object, and we denote that feature as $u_{\text{max}} \in \mathbf{U}$. The loss function's purpose is to inject the patch feature information, which contains the most detailed information about the small object, into the fused feature. Specifically, we use a text feature w that describes the small object in the image as a proxy feature to draw the fused feature v close to the patch feature u_{max} since the text feature is in the same joint feature space as the image feature. We use a similarity function to measure the similarity between the proxy feature and the candidate patch feature and choose the most similar patch feature as u_{max}. Meanwhile, we get similarities between the fused feature and the proxy feature. Then, we minimize the distance function to draw these two similar distributions together.

$$\mathbf{L_{qp}} = \mathbf{D}[sim(v, w), sim(w, u_{\text{max}})] \tag{7}$$

where $w \in \mathbb{R}^d$ has the same dimension as v and u_{max}. The symbol *sim* represents the similarity measure function, and the symbol $\mathbf{D}$ represents the distance measure function. We try to learn v so that the distance between the similarity distribution of v and w and the similarity distribution of

$u_{\max}$ and w can be minimized. For every batch of images, we use all class prompted text features to calculate similarity, and no additional supervision is used besides the class name in the dataset. A Pytorch style code and the complete pipeline of our model can be seen at Appendix A.4.

4 BENCHMARK

In text-image retrieval, the traditional task "use a caption to retrieve single image" is widely used to evaluate the retrieval capability of the model. However, in practice, the need to "use a word to retrieve all related images in a database" is waiting to be fulfilled. Traditional text-image retrieval datasets such as Flickr30k and COCO-caption (MSCOCO dataset using caption to retrieve) are unsuitable for evaluating the latter task. We select some object detection datasets and use "prompting with class name" as a text query to retrieve all related images in a database. We construct the benchmark for this vital task by adopting existing datasets, making a new synthetic dataset, and designing the evaluation metrics.

4.1 EXISTING DATASETS

Theoretically, many existing datasets with class-wise object supervision and class names can be used to construct the benchmark. However, both have deficits not only for the task we proposed but also for analysis of the ability of methods in detail retrieval. Examples are shown in Figure 4.

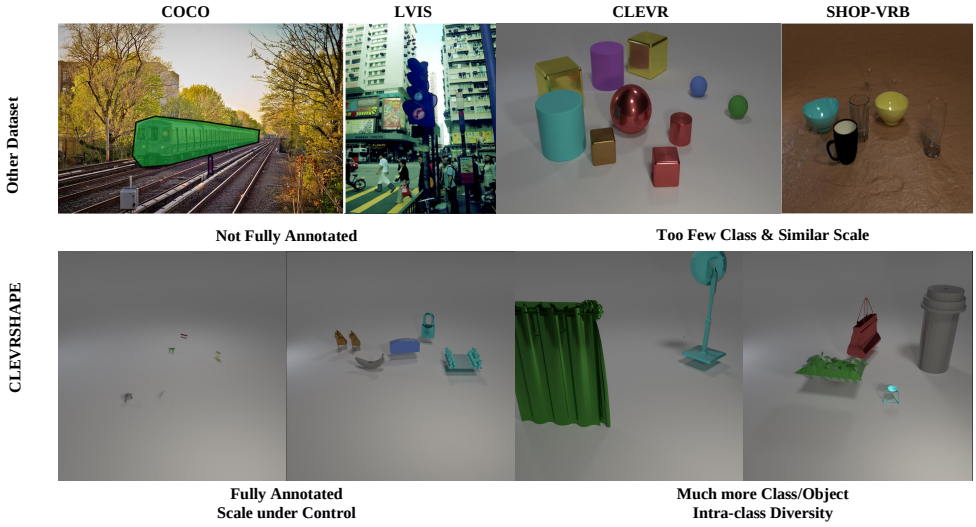


Figure 4: Multi Dataset Illustration

For real-world datasets, the label information for Visual Genome Krishna et al. (2017), ImageNet-1K Russakovsky et al. (2014), GPR1200 Schall et al. (2021) is more focused on the main object in the image. Since these datasets are collected from real-world scenes, full-annotation of the images is difficult to achieve. Therefore the annotation of fine detail is insufficient. Furthermore, the large dataset LVIS Gupta et al. (2019) has many missing labels, making it impossible to obtain accurate conclusions during the retrieval evaluation. Synthetic datasets such as CLEVR by Johnson et al. (2017), SHOP-VRB by Nazarczuk & Mikolajczyk (2020) have too few object categories. Besides, the datasets have no additional design and attention to the object size in the images.

4.2 CLEVR-DS

4.2.1 SUMMARY OF CLEVR-DS

In order to achieve accurate retrieval evaluation, we made a dataset, "CLEVR of Different Scales(**CLEVR-DS**)," which includes 138 categories in ShapeNet, and with mean instances number per image to almost 14 which is more than LVIS. We can realize full-annotation information (e.g., spatial position, bounding box, category, and attribute) in the image. Our CLEVR-DS covers a wide range of object scale (mix, large and small), and shows greater variability on scene complexity. More statistics and visualization about CLEVR-DS datasets is shown in Appendix A.2.

4.3 EVALUATION METRICS

Current evaluation metric for text-to-image retrieval is to calculate the top k recall accuracy of a text query. However, we focus on retrieving all related images in a database, and the top k retrieve result is not enough to evaluate the performance of our method. We propose a new evaluation metric Recall@k to evaluate the retrieval performance. Recall@k is calculated as follows,

$$\text{Recall@k} = \frac{t_k}{n} \tag{8}$$

where n is the number of images of the query (also a class) in the database. t_k represents the number of images that contain the query in the first $n \times k$ retrieved images. For the number k, we select 1, 3, and 5 as the anchor points of the evaluation indicators. All datasets are evaluated in this method.

5 EXPERIMENTS

5.1 IMPLEMENTATION DETAILS

Dataset setting: We focus primarily on two types of datasets: synthetic and real-world datasets. Since the synthetic dataset is fully annotated, we test our model's image fine-detail retrieval ability by setting the size of objects in the image. For CLEVR-DS, we define two sub-datasets as CLEVR-DS-S/CLEVR-DS-L, which contain only small/large objects of the query semantic information and the distractor data. For Unity-Retail datasets, we randomly split them in 7:1:2 for train, validation, and test set. We also evaluate our model on real-world datasets MSCOCO and LVIS. For MSCOCO, we randomly take 5000 images for validation and use the original validation set to test. Since LVIS has the same images as MSCOCO, we use the same split setting as MSCOCO.

Model architecture: In order to test the generalizability of DetailCLIP framework, we use several CLIP-like models as image encoders to extract image features. Besides, the image and text features are mapped into 512 and 768 dimensions for different image encoder architecture. We use a small transformer structure for the image feature fusing model with three encoders and three decoders. The whole framework is optimized through query proxy loss.

Patch selection mode: In the main result, we use **CC** to represent the patch selection method. To save computational cost, we set patch numbers as 166. We test the CILP-like model's retrieval ability for untrained retrieval on the **CC** patches and whole image features. Specifically, we choose the highest **CC** patch score as the image retrieval score. For **CC**@$\cdot$, the $\cdot$ represents the selection of hyper-parameter effective scale sensitivity. In the DetailCLIP scenario, the **CC** patch features are used to train the model. A study on choosing the value of k is in Appendix A.1.

5.2 MAIN RESULT ANALYSIS

5.2.1 RESULTS ON CLEVR-DS

This section uses CLIP, whose backbone is ViT-B/32, as the image and text feature extractor. We choose a tiny transformer with three encoders and three decoders to fuse the patch feature. In the non-training scenario, we use single-feature and multi-feature to retrieve. The former uses the similarity between the whole image feature and the text feature for decision-making, and the latter uses the largest similarity of all patch features with the text feature for decision-making. As shown in Table 2, DetailCLIP improves the retrieval performance from 10.61% to 22.54% on CLEVR-DS

dataset with mixed object size. At the same time, DetailCLIP has a significant improvement in the CLEVR-DS-S subset with 10.23% and a good improvement in the CLEVR-DS-L subset with 2.76%.

Dataset	Method	Single Feature	Input	Recall@1	Recall@3	Recall@5
CLEVR-DS	CLIP	✔	Full Image	10.61%	29.64%	49.50%
	CLIP	×	**CC**@10	**22.56**%	45.45%	64.91%
	DetailCLIP	✔	**CC**@10	22.54%	**46.16**%	**64.98**%
CLEVR-DS-S	CLIP	✔	Full Image	4.43%	16.25%	30.32%
	CLIP	×	**CC**@10	14.57%	32.55%	**37.12**%
	DetailCLIP	×	**CC**@10	**14.66**%	**32.77**%	32.78%
CLEVR-DS-L	CLIP	✔	Full Image	13.57%	23.19%	28.65%
	CLIP	×	**CC**@10	15.30%	**25.93**%	**31.97**%
	DetailCLIP	✔	**CC**@10	**16.33**%	25.47%	30.96%

Table 2: Retrieval Performance of DetailCLIP and CLIP on CLEVR-DS.

5.2.2 Results on Real-World and synthetic dataset

In this section, we use several feature extractors to test the performance of our DetailCLIP framework. To verify the effectiveness of our method on datasets with complicated scenes, we test on the MSCOCO, LVIS, and Unity-Retail datasets. The former two are real-world datasets but is not fully annotated. The latter is a synthetic dataset with more complicated scenes and is fully annotated. Table 3 demonstrates that the retrieval results of DetailCLIP on most datasets are better than the full image baseline and **CC**@10 baseline. e.g., on MSCOCO, DetailCLIP outperforms the full image baseline by ∼6% in recall@1, showing that the DetailCLIP feature is better than the original CLIP feature in retrieval task on the target domain. However, the margin between the full image baseline and the DetailCLIP of MSCOCO (∼6%) is smaller than Unity-Retail (∼17%) and CLEVR-DS-mix (∼20%). Although Unity-Retail is still a synthetic dataset, it has more complicated scenes than CLEVR-DS. We speculate that the full/not full-annotation problem causes the margin gap for different datasets.

DATASET			**LVIS**	**COCO**	**Unity**	**CLEVR-DS**
Method	Single Feature	Input	Recall@1	Recall@1	Recall@1	Recall@1
§ CLIP-ViT-B/16	✔	Full Image	7.49%	40.93%	24.63%	8.51%
§ CLIP-ViT-B/16	×	**CC**@10	**9.40**%	41.24%	23.11%	17.03%
DetailCLIP	✔	**CC**@10	7.66%	**44.19**%	**25.02**%	**18.09**%
† CLIP-ViT-B/14	✔	Full Image	15.12%	56.74%	35.74%	13.81%
† CLIP-ViT-B/14	×	**CC**@10	**22.00**%	59.40%	52.40%	33.21%
DetailCLIP	✔	**CC**@10	15.29%	**62.63**%	**55.21**%	**33.46**%
§ SLIP	✔	Full Image	9.65%	47.49%	24.42%	9.51%
‡ RegionCLIP	✔	Full Image	10.13%	46.06%	24.10%	10.98%

§ Trained on YFCC-15M
† Trained on 400M images-text pairs
‡ Trained on Conceptual Caption (CC3M)

Table 3: Retrieval Performance of DetailCLIP framework and other methods on four datasets.

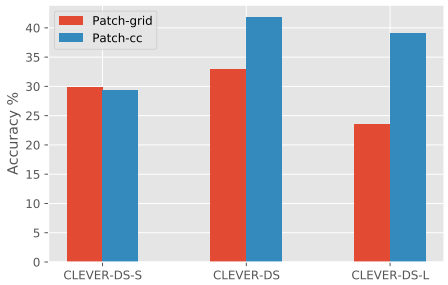
5.3 Ablation Study

5.3.1 Patch Generation And Upper Bound Ablation

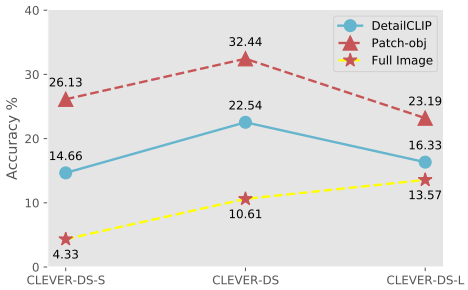
Firstly, we define patch generation schemes, "Patch-cc" and "Patch-grid." "Patch-grid" is simply slicing the image into non-overlapped, equal-sized patches. "Patch-cc" generates patches for an image following the Complete Cover (**CC**) scheme. In order to verify that **CC** can effectively generate the patches with different levels of details to inject and fuse, we compare the above two patch generation schemes on the CLEVR-DS dataset. According to Figure 5(a), patch-cc outperforms the patch-grid

scheme on all sizes of CLEVR-DS datasets except for CLEVR-DS-S. The above results demonstrate that our patch-cc method can generate better patches than patch-grid.

Secondly, "Patch-obj" generates patches by cropping objects from the image using the bounding box. Intuitively, directly retrieve from bounding box patches is the upper bond performance of CLIP. In Figure 5(b), "Full image" represents retrieve the original image CLIP feature, and "DetailCLIP" means to the retrieve the DetailCLIP feature, result of DetailCLIP with **CC** input 22.54% is closer to the result of "Patch-obj" 32.44%. More ablation study is in Appendix A.3



(a) Retrieval performance under different patch scheme.



(b) Retrieval result of different approach.

5.3.2 Fine-tune On Target Domain

Since our method has trained on the target dataset in an unsupervised manner, to address the method's effectiveness, we experiment with four different adaptation methods on the CLEVR-DS dataset. Each adds a trainable module on top of the CLIP model, as our method does. We tried two modules, MLP and Transformer, which use the original CLIP feature as input and the adapted feature on the target domain as output. Results in Table 4 show that the vanilla adaptation on the target dataset is not comparable to our method.

Table 4: Add the trainable modules on top of the CLIP model to fine-tune

Module		Patch-cc	Recall@1
Transformer	# of MLP-layers		
✗	0	✗	10.61%
✗	1	✗	15.91%
✗	2	✗	14.70%
✗	3	✗	13.96%
✔	✗	✗	15.90%
✔	✗	✔	**22.54**%

6 Conclusion & Future Work

Our paper presents a feature fusion model, DetailCLIP, for the text-to-image retrieval task. DetailCLIP shares the same semantic space with CLIP-like models and achieves an outstanding performance in detail retrieval. We proposed a CC patch selection scheme and a Transformer-based framework with query proxy loss to obtain a detail-friendly feature representation. To verify the retrieval performance of DetailCLIP, we constructed the CLEVR-DS dataset. Extensive experiments on this dataset and other popular datasets demonstrate that DetailCLIP can surpass the retrieval performance of CLIP-like models. However, selecting patches that contain the target object has a strong inductive bias on removing the redundant information in an image. It is computationally expensive and not available for end-to-end training. CLIP provides great semantic space, and directly removing redundant information in the space by modifying the CLIP feature will be an interesting work.

REFERENCES

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.

Zhaowei Cai, Gukyeong Kwon, Avinash Ravichandran, Erhan Bas, Zhuowen Tu, Rahul Bhotika, and Stefano Soatto. X-detr: A versatile architecture for instance-wise vision-language tasks. *arXiv preprint arXiv:2204.05626*, 2022.

Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. ShapeNet: An Information-Rich 3D Model Repository. Technical Report arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *CoRR*, abs/2010.11929, 2020. URL https://arxiv.org/abs/2010.11929.

Agrim Gupta, Piotr Dollar, and Ross Girshick. Lvis: A dataset for large vocabulary instance segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 5356–5364, 2019.

Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc V. Le, Yunhsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. URL http://arxiv.org/abs/2102.05918.

Justin Johnson, Bharath Hariharan, Laurens Van Der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2901–2910, 2017.

Aishwarya Kamath, Mannat Singh, Yann LeCun, Gabriel Synnaeve, Ishan Misra, and Nicolas Carion. MDETR – modulated detection for end-to-end multi-modal understanding. URL http://arxiv.org/abs/2104.12763.

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.

Liunian Harold Li*, Pengchuan Zhang*, Haotian Zhang*, Jianwei Yang, Chunyuan Li, Yiwu Zhong, Lijuan Wang, Lu Yuan, Lei Zhang, Jenq-Neng Hwang, Kai-Wei Chang, and Jianfeng Gao. Grounded language-image pre-training. In *CVPR*, 2022.

Yangguang Li, Feng Liang, Lichen Zhao, Yufeng Cui, Wanli Ouyang, Jing Shao, Fengwei Yu, and Junjie Yan. Supervision exists everywhere: A data efficient contrastive language-image pre-training paradigm. *arXiv preprint arXiv:2110.05208*, 2021.

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*, pp. 740–755. Springer, 2014.

Norman Mu, Alexander Kirillov, David A. Wagner, and Saining Xie. SLIP: self-supervision meets language-image pre-training. *CoRR*, abs/2112.12750, 2021. URL https://arxiv.org/abs/2112.12750.

Michal Nazarczuk and Krystian Mikolajczyk. Shop-vrb: A visual reasoning benchmark for object perception. *International Conference on Robotics and Automation (ICRA)*, 2020.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. URL http://arxiv.org/abs/2103.00020.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *CoRR*, abs/1409.0575, 2014. URL http://arxiv.org/abs/1409.0575.

Konstantin Schall, Kai Uwe Barthel, Nico Hezel, and Klaus Jung. GPR1200: A benchmark for general-purpose content-based image retrieval. *CoRR*, abs/2111.13122, 2021. URL https://arxiv.org/abs/2111.13122.

vijishmadhavan. Crop-clip. https://github.com/vijishmadhavan/Crop-CLIP#Simple-App, 2022.

Chenyun Wu, Zhe Lin, Scott Cohen, Trung Bui, and Subhransu Maji. PhraseCut: Language-based image segmentation in the wild. URL http://arxiv.org/abs/2008.01187.

Jiarui Xu, Shalini De Mello, Sifei Liu, Wonmin Byeon, Thomas Breuel, Jan Kautz, and Xiaolong Wang. Groupvit: Semantic segmentation emerges from text supervision. *arXiv preprint arXiv:2202.11094*, 2022.

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.

Lu Yuan, Dongdong Chen, Yi-Ling Chen, Noel Codella, Xiyang Dai, Jianfeng Gao, Houdong Hu, Xuedong Huang, Boxin Li, Chunyuan Li, Ce Liu, Mengchen Liu, Zicheng Liu, Yumao Lu, Yu Shi, Lijuan Wang, Jianfeng Wang, Bin Xiao, Zhen Xiao, Jianwei Yang, Michael Zeng, Luowei Zhou, and Pengchuan Zhang. Florence: A new foundation model for computer vision. URL http://arxiv.org/abs/2111.11432.

Yiwu Zhong, Jianwei Yang, Pengchuan Zhang, Chunyuan Li, Noel Codella, Liunian Harold Li, Luowei Zhou, Xiyang Dai, Lu Yuan, Yin Li, and Jianfeng Gao. Regionclip: Region-based language-image pretraining. *CoRR*, abs/2112.09106, 2021. URL https://arxiv.org/abs/2112.09106.

Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. *arXiv preprint arXiv:2109.01134*, 2021.

A Appendix

A.1 The Effectiveness of Complete Cover

Let us first take a look at the brute force algorithm. For each image with sidelength a, the number of all possible patches is $O(a^4)$, since a rectangular patch is defined by its top-left and bottom-right corner coordinates. Each coordinate comprises two numbers (x, y) that leads to $O(a^4)$ patches. Even if we confine the patches to be square, there are also $O(a^3)$ patches.

For simplicity, we take square patches as our example. Please note that the c we defined as the ratio of the perimeter in main paper is equivalent to the ratio of the sidelength. Moreover, if we adopt Complete Cover scheme, we can obtain patches (covers) at different levels with side lengths of $[c, 2c, 3c, \cdots, a]$. For each level, the number of patches to cover all targets for corresponding sidelength are $O(\left(\frac{a}{c}\right)^2), O(\left(\frac{a}{2c}\right)^2), O(\left(\frac{a}{3c}\right)^2), \cdots, O(\left(\frac{a}{a}\right)^2)$ respectively. The total number of patches introduces by Complete Cover is:

$$\left(\frac{a}{c}\right)^2 * (1 + \frac{1}{2^2} + \frac{1}{3^2} + \cdots) \tag{9}$$

$$= \left(\frac{a}{c}\right)^2 \times \frac{\pi^2}{6} \tag{10}$$

$$= O(a^2) \tag{11}$$

where c is a constant across the experiment.

We plot number of patches under different sidelengths with $c = 3$ in Figure 5and a quadratic function $y = 0.25x^2$. We can see that they fit well.

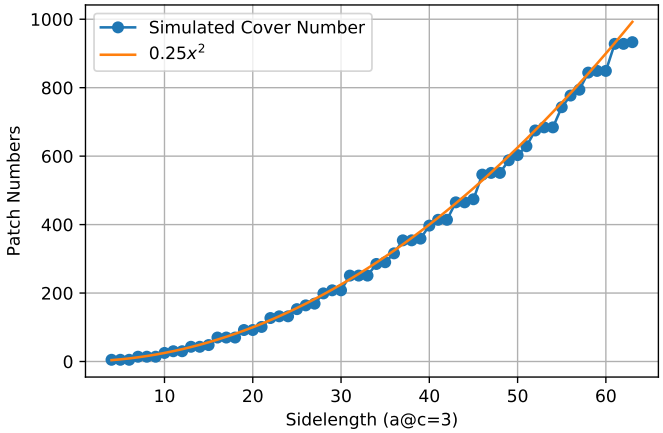


Figure 5: Patch numbers for different sidelengths when $c = 3$.

Please note that, before crop patches, we resize the image to ensure the sidelength a is divisible by c. The minimum effective range for Q is defined as the range of sidelength for P that makes $C(P, Q) = 1$. The formula for cover sidelength, minimum effective range, and the patch number for a given c are presented in Table 5.

Table 5: Relationship between the cover sidelength, the minimum effective range, and the number of patches at a given c

Level	Cover Sidelength	Minimum Effective Range	Patch Numbers
1	a	$a \geq x \geq \frac{a}{c}$	1
2	$a - c$	$\frac{a}{c} > x \geq \frac{a-c}{c}$	$(\frac{2c+ac-a}{-c^2+2c+ac-a})^2$
3	$a - 2c$	$\frac{a-c}{c} > x \geq \frac{a-2c}{c}$	$(\frac{3c+ac-a}{-2c^2+3c+ac-a})^2$
...	...	...	...
n	$(\frac{nc+ac-a}{-(n-1)c^2+nc+ac-a})^2$	$\frac{a-(n-2)c}{c} > x \geq \frac{a-(n-1)c}{c}$	$a - (n-1)c$
...	...	...	...

A.2 Data Statistics

We lists the datasets we choose to perform the class with prompt text-to-image retrieval task. The statistics about CLEVR-DS, Unity-Retail, MSCOCO and LVIS are shown from Table 6 to Table 10. These four datasets have relatively more instances per image than

other datasets, which are more likely to contain small objects in an image. Particularly for our CLEVR-DS, the mean number of instance per image of CLEVR-DS is more than LVIS.

Table 6: Statistics of CLEVR-DS

Per Image	Mean	Min	Max
Instance	13.78	1	50
Class	13.78	1	50

Table 7: Statistics of MSCOCO

Per Image	Mean	Min	Max
Instance	7.33	1	93
Class	2.92	1	18

Table 8: Statistics of Unity-Retail

Per Image	Mean	Min	Max
Instance	25.52	16	42
Class	13.52	9	19

Table 9: Statistics of LVIS

Per Image	Mean	Min	Max
Instance	11.2	1	294
Class	3.4	1	24

Table 10: Class number of datasets.

DATASET	COCO	LVIS	CLEVR-DS	Unity-Retail
Image Number	122,219	122,219	10,000	1,000
Class Number	80	1230	138	16

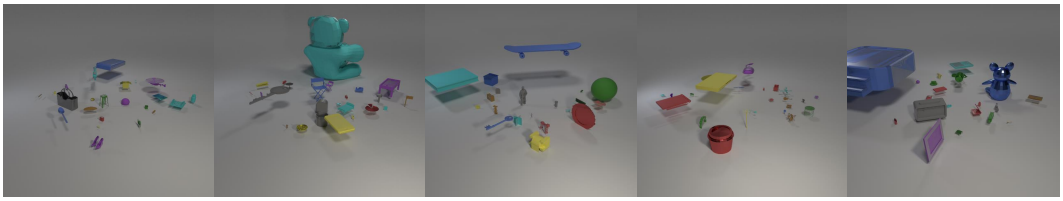


Figure 6: Above are samples of our synthetic CLEVR-DS dataset. This dataset is similar to the original CLEVR but far more challenging in scene complexity. We randomly scatter 1-50 instances from 138 classes of ShapeNet objects on the image. Each image contains 1-3 big objects and more than 10 small objects.

A.3 More Ablation Study

In order to fully demonstrate the retrieval ability of our method, we use a smaller CLEVR-DS dataset with an average of three objects per image, without any occlusion. Next ablation study use the smaller dataset.

A.3.1 Information Injection Ablation

In this section, we test the DetailCLIP framework's ability to inject detailed information in a different number of patches, and we use CLEVR-DS to perform the task. We use a different number of patches to train the DetailCLIP fusing model and test the retrieval performance of the fused feature. In Figure 7, "Grid" and "CC" represents the patch selection method which is the same as the above setting, and "Mix," "Small," and "Large" means the object scale in CLEVR-DS dataset. As is shown in Figure 7, with the increase in patch number, DetailCLIP's performance stays stable, which proves that our framework could inject detailed information from many patches. It also shows whether the patch generation method is patch-cc or patch-grid. The results of DetailCLIP are similar to the multi-feature CLIP results, which proves that DetailCLIP can effectively inject information from different sources.

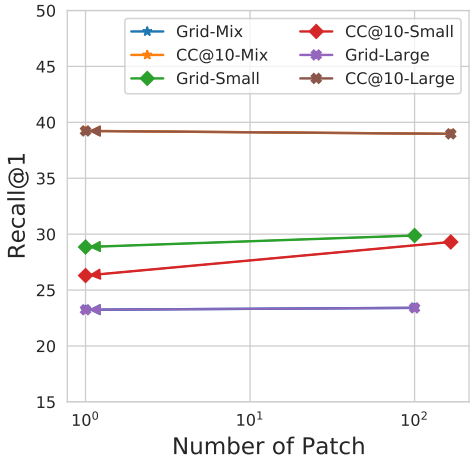


Figure 7: After applying DetailCLIP, we achieve 100x #Patch Reduction with no significant performance loss.

A.3.2 DetailCLIP Performance Under Different Complete Cover Schemes

Based on different c, we list the number of patches at different levels respectively in Table 11. We set $\frac{a}{c} = k$ for convenience of notation.

Table 11: Number of patches at different levels.

CC@k	Patch Numbers	Level 1	Level 2	Level 3	Level 4	Level 5	Level 6	Level 7
1	1	1						
2	5	1	4					
3	14	1	4	9				
4	30	1	4	9	16			
5	39	1	4	9	25			
6	66	1	4	9	16	36		
7	79	1	4	9	16	49		
8	103	1	4	9	25	64		
9	136	1	4	9	16	25	81	
10	166	1	4	9	16	36	100	
11	187	1	4	9	16	36	121	
12	248	1	4	9	16	25	49	144
13	273	1	4	9	16	25	49	169
14	315	1	4	9	16	25	64	196
15	355	1	4	9	16	36	64	225

To select a suitable k, we test the performance of the DetailCLIP model with 14 different k. The results are shown in Figure 8. Lines with star markers are retrieval results, and lines without star markers are DetailCLIP results. Dash lines are results for recall@1, and we use the same color for the same patch selection method under the same recall. The figure shows several facts:

- The recall@1 results for DetailCLIP (single feature) are comparable with retrieval baseline (multi-feature) for any k. The DetailCLIP has slightly better performance with large k.
- The performance for both DetailCLIP and retrieval baseline begin to increase since $k = 2$, and become saturated at $k = 9$. Patch-cc's performance exceeds patch-grid since $k = 4$ for recall@1.
- For $k \geq 8$, the number of patches increases dramatically, but the performance of the DetailCLIP model does not.

Based on the analysis above and the trade-off between DetailCLIP performance and computation complexity, we select $k = 10$ to finish experiments in the main body of our paper. Also, when $k = 10$, the last level of the patch number is the same as for the patch-grid method.

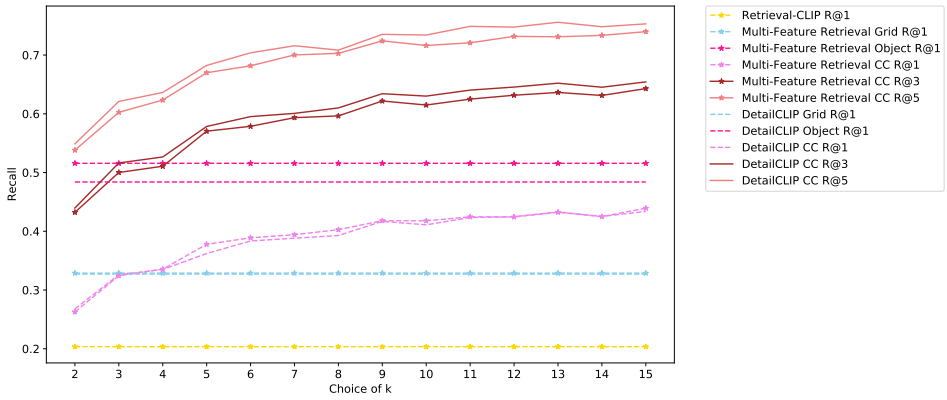


Figure 8: Recall for retrieval and DetailCLIP models with different k Values

A.4 PYTORCH-LIKE CODE

A Pytorch style code and the complete pipeline of our DetailCLIP model are listed in 1.

```
# b: batch size
# p: number of patch
# t: number of text feature
# f: feature dim
# vanilla_feature: clip feature for entire image, (b, f)
# patch_feature: clip feature for different patches, (b, p, f)
# text_feature: clip feature for text prompts, (t, f)
# fusing_model: DetailCLIP model
# All features are normalized.

DetailCLIP_feature=fusing_model(patch_feature, vanilla_feature)
patch_feature=rearrange(patch_feature, 'b p f -> (b p) f')
proxy_feature=text_feature
# (t,  f) @ (f, b * p) -> (t, b * p)
q_p_similarity=proxy_feature @ patch_feature.T
q_p_similarity=rearrange(q_p_similarity, 'k (b p) -> k b p')
# (t, b, p) -> (t, b)
q_p_similarity_max=q_p_similarity.max(-1)
# (t, f) @ (f, b) -> (t, b)
q_c_similarity=proxy_feature @ DetailCLIP_feature.T
query_proxy_loss=mse_loss(q_p_similarity_max, q_c_similarity
```

Listing 1: Pytorch Like Code

A.5 RETRIEVAL RESULT VISUALIZATION

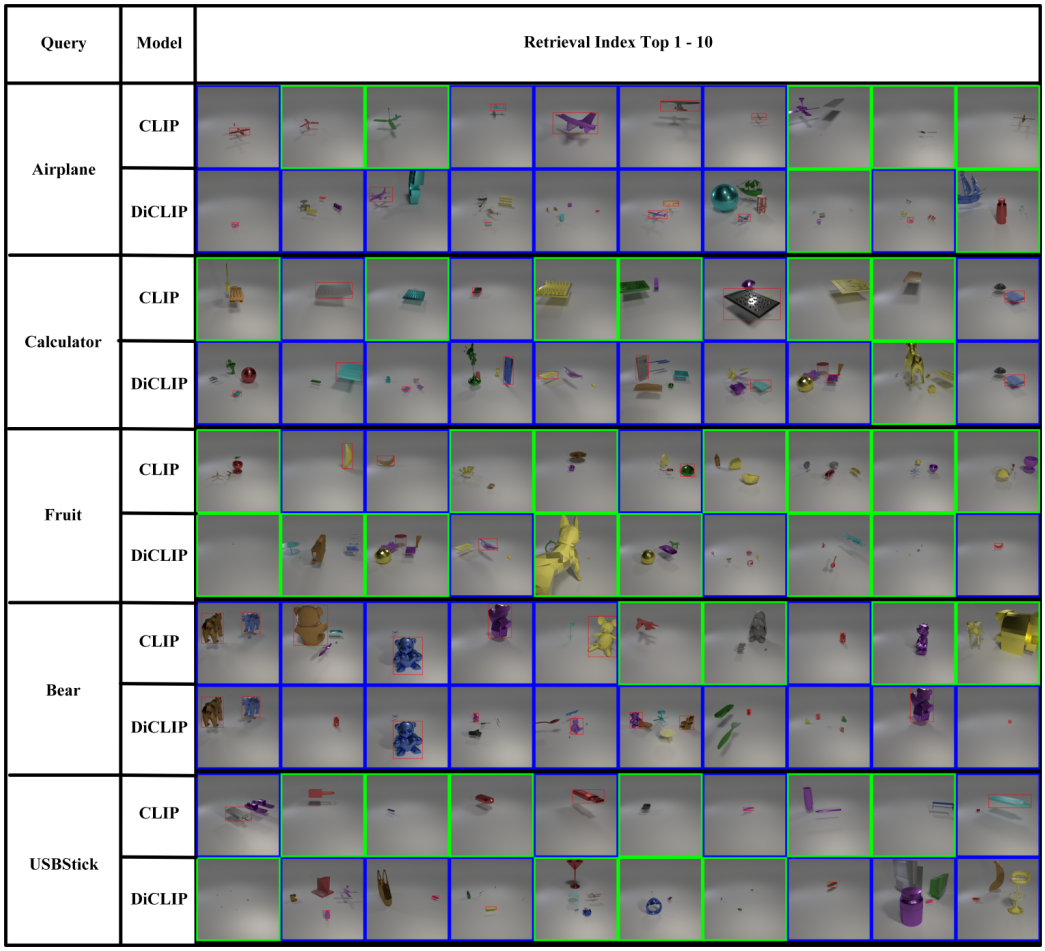


Figure 9: Retrieval result visualization. The ground truth images for the query are surrounded by blue frames, while green frames surround others. Note that two rows comprise a group. Each row is a ranked retrieval result, the larger portion of blue frames the better. The upper row in each group is the top 10 retrieval result for the CLIP, and the lower row is the result for the DetailCLIP. The ground truth objects are marked with a red bounding box for visualization. It should be aware that non of the methods here produces the bounding box.

A.6 HYPER-PARAMETER TUNING

We use AdamW optimizer with a linear learning rate scheduler and a linear warm-up training strategy for ten epochs. DetailCLIP is trained using a single GTX 2080ti. Throughout our DetailCLIP experiments, the batch size is set to 30. For different k, the hyper-parameters of DetailCLIP are independently grid searched over the table below. We select the hyper-parameters on a validation set and report the result on a held-out test set.

Table 12: Hyper-Parameter candidate in validation Set.

Name	Candidate
Learning Rate	[0.001, 0.003, 0.005, 0.007, 0.01]
Weight Decay	[0, 0.001]
Step Size for Learning Rate Decay	[60, 120]
Gamma Value for Learning Rate Decay	[0.5, 0.7, 0.9]
Gradient Clip Value	[0.00001, 0.0001]
Layer Normalization's Epsilon	[0.0001, 0.001, 0.01]

A.7 DATASHEET

Motivation	
For what purpose was the dataset created? Who created this dataset? Who funded the creation of the dataset?	The dataset was created for training and evaluating the text-image retrieval models.
Any other comments?	Compared with other datasets for text-image retrieval such as MSCOCO, LVIS, Conceptual Captions, etc., our dataset focus on detail retrieval (small objects), and every object has an annotation in our dataset if they appear in a image. Also, we can generate images as many as we need.
Composition	
What do the instances that comprise the dataset represent (e.g. documents, photos, people, countries)?	138 different types of common objects from ShapeNet rendered at different sizes on a clean background.
How many instances are there in total (of each type, if appropriate)?	The dataset contains 10k image-annotation pairs.
Does the dataset contain all possible instances or is it a sample (not necessarily random) of instances from a larger set?	Our dataset is procedurally generated. There can be as many instances as one need.
What data does each instance consist of?	Each instance contains an image with every object in the image annotated with its box and class.
Is there a label or target associated with each instance?	Yes, Each instance has full annotations, including all objects with their bounding box, texture, size, category, etc.
Is any information missing from individual instances?	No.

Are relationships between individual instances made explicit?	Instance are i.i.d. generated from the same program. Objects in the image share the same object classes, but with different sizes and view angles. All instances share the same visual appearance.
Are there recommended data splits?	We use random splits for the training and testing and validation sets.
Are there any errors, sources of noise, or redundancies in the dataset?	No.
Is the dataset self-contained, or does it link to or otherwise rely on external resources?	The dataset is self-contained.
Does the dataset contain data that might be considered confidential?	No.
Does the dataset contain data that, if viewed directly, might be offensive, insulting, threatening, or might otherwise cause anxiety?	No.
Any other comments?	None.
Collection Process	
How was the data associated with each instance acquired?	The objects we used to generate our dataset are taken from ShapeNet, which is available publicly on the web.
What mechanisms or procedures were used to collect the data?	The data was generated using modified CLEVR pipeline and Blender.
If the dataset is a sample from a larger set, what was the sampling strategy?	Only objects from ShapeNet are sampled. We first discard all broken objects that cannot be loaded by the modified CLEVR pipeline (259 classes remained). Then, we choose 138 classes. We sample 10 3D object models from the remaining 138 classes.
Who was involved in the data collection process?	Researchers at our institute.
Over what timeframe was the data collected?	The dataset was generated in September 2022. We didn't filter the sources based on the creation date.
Were any ethical review processes conducted?	Yes.
Preprocessing / Cleaning / Labeling	
Was any preprocessing/cleaning/labeling of the data done (e.g. discretization or bucketing, tokenization, part-of-speech tagging, SIFT feature extraction, removal of instances, processing of missing values)?	No.
Is the software used to preprocess/clean/label the instances available?	No (but we will release the code later).
Any other comments?	None.
Uses	
Has the dataset been used for any tasks already?	No.
Is there a repository that links to any or all papers or systems that use the dataset?	No, the dataset is only used to train and evaluate the models in this paper for now.

What (other) tasks could the dataset be used for?	The dataset be used for relation detection and localization tasks.
Is there anything about the composition of the dataset or the way it was collected and preprocessed/cleaned/labeled that might impact future uses?	The dataset is rendered by ShapeNet objects, one may need to change the objects to other 3D models if ShapeNet cannot satisfy the needs.
Are there tasks for which the dataset should not be used?	For tasks related to data which have enormous domain gap to ShapeNet objects, the dataset should not be used.
Any other comments?	None.
Distribution	
Will the dataset be distributed to third parties outside of the entity (e.g. company, institution, organization) on behalf of which the dataset was created?	Yes.
How will the dataset will be distributed (e.g., tarball on website, API, GitHub)?	Github.
When will the dataset be distributed?	TBD.
Any other comments?	None.

A.8 MODEL CARD

Model Details	
Person or organization developing model	DetailCLIP was developed by our institute.
Model date	DetailCLIP will be released on October, 2022.
Model version	DetailCLIP described in this paper is version 1.0.0.
Model type	DetailCLIP is a shallow transformer-based feature fusing model for text-image retrieval task
Information about training algorithms, parameters, fairness constraints or other applied approaches, and features	Please see the Data Card (Appendix A.7) for the information about training data and Section 6.1 for the information about the training process. We listed the choice of hyperparameters in Appendix A.6.
Paper or other resource for more information	Please see the paper for details on DetailCLIP. Our implementation will be available later at our Github repository.
License	TBD.
Where to send questions or comments about the model	Please contact the corresponding authors for any questions or comments.
Intended Use	
Primary intended uses	We release the DetailCLIP for text-based image retrieval tasks, especially for detail retrieval (small objects).
Primary intended users	We primarily target researchers and the related research community who is interested in detail retrieval task or CLIP feature fusing task.

Out-of-scope use cases	TBD.
Data, Limitations, and Recommendations	
Data selection for training	Training data for DetailCLIP is randomly sampled from CLEVR-DS. As for the dataset generation procedure, please see our Datasheet (Appendix A.7) for more information.
Data selection for evaluation	The validation set is randomly sampled from CLEVR-DS, with an emphasis on detail retrieval for small objects in an image from text.
Limitations	The feature input to DetailCLIP model is based CLIP, which inherit not only the capabilities of CLIP, but also the limitations of CLIP, e.g., If an object cannot be recognized in any patch by CLIP, DetailCLIP cannot improve the situation as well. Caution should be taken on the use of model trained on synthetic data on real world scenario. Furthermore, our synthetic dataset is background-free, which means the visual context is relatively simple. The text vocabulary we used is fairly small, which could lead to bias on out-of-vocabulary classes.
Recommendations for future work	Extend DetailCLIP to a larger real-world dataset with a large text vocabulary. Another direction is to use patches feature in place of text features as retrieval queries, since they share the same feature space. This could lead to a new text-free learning-by-retrieval feature learning paradigm.