

Supplementary Material for SaCo Loss: Sample-wise Affinity Consistency for Vision-Language Pre-training

Sitong Wu^{1,*} Haoru Tan^{2,*} Zhuotao Tian³ Yukang Chen¹
Xiaojuan Qi² Jiaya Jia^{1,3,†}

¹CUHK ²HKU ³SmartMore

This material includes the following parts:

- Sec. 1 provides the pseudo code for illustrating how to incorporate our SaCo loss with the baseline vision-language model.
- Sec. 2 provides the detailed dataset descriptions and experimental settings for pre-training.
- Sec. 3 introduces the evaluation tasks and settings in detail, including the datasets, experimental setups and implementation details for each task.
- Sec. 4 presents more experiments and analysis, including the comparison on different pre-training strategies (Sec. 4.1), evaluation on linear-prob image classification (Sec. 4.2), and analysis on pre-training cost (Sec. 4.3).
- Sec. 5 discusses more about the solutions to instability problems when pre-training from scratch.

1. Pseudo Code

Our proposed SaCo loss can be incorporated with the existing vision-language models. For better understanding, we take the popular CLIP [12] model as an example, and present the pseudo-code of integrating our SaCo loss into it. Algorithm 1 and Algorithm 2 correspond to the process of “pre-training from scratch” and “continue pre-training” settings, respectively.

2. Pre-training Details

2.1. Pre-train Datasets

We utilize two commonly used open-source image-text datasets at varying scales, with around 3M and 15M pairs, respectively. It is worth noting that, due to expired image links or non-English captions, we were unable to obtain complete data for these datasets. Consequently, there might be slight performance discrepancies when comparing models trained on the complete image versions. We compare the data volume of the original version and our collected

*Equal contribution.

†Corresponding author.

Algorithm 1: Pre-train CLIP from scratch with our SaCo loss

Input: image-text pair dataset $\mathcal{D} = \{(x_j^I, x_j^T)\}_{j=1}^D$, total iterations T , batch size n , loss weight α and β , image pseudo-affinity $\tilde{\mathbf{S}}^I$, a randomly initialized CLIP model (including projection weight W^I and W^T , image_encoder, text_encoder, temperature τ)

```

1 for  $i$  from 1 to  $T$  do
2   # sample a mini-batch of images and texts
3   sample  $(\mathbf{x}_i^I, \mathbf{x}_i^T) \sim \mathcal{D}$ 
4   # extract image and text embedding
5    $\tilde{\mathbf{I}}_i = \text{image\_encoder}(\mathbf{x}_i^I)$  #  $[n, d^I]$ 
6    $\tilde{\mathbf{T}}_i = \text{text\_encoder}(\mathbf{x}_i^T)$  #  $[n, d^T]$ 
7   # projection & L2 normalization
8    $\mathbf{I}_i = \text{L2\_norm}(\tilde{\mathbf{I}}_i \cdot W^I, \text{axis}=1)$  #  $[n, d]$ 
9    $\mathbf{T}_i = \text{L2\_norm}(\tilde{\mathbf{T}}_i \cdot W^T, \text{axis}=1)$  #  $[n, d]$ 
10  # contrastive loss
11  logits = np.dot( $\mathbf{I}_i, \mathbf{T}_i.T$ ) /  $\tau$  #  $[n, n]$ 
12  labels = np.arange(n)
13   $\mathcal{L}_{\text{cont}}^I = \text{cross\_entropy}(\text{logits}, \text{labels}, \text{axis}=0)$ 
14   $\mathcal{L}_{\text{cont}}^T = \text{cross\_entropy}(\text{logits}, \text{labels}, \text{axis}=1)$ 
15   $\mathcal{L}_{\text{cont}} = (\mathcal{L}_{\text{cont}}^I + \mathcal{L}_{\text{cont}}^T) / 2$ 
16  # SaCo loss
17  image_affinity  $\mathbf{S}_i^I = \text{np.dot}(\mathbf{I}_i, \mathbf{I}_i.T)$  #  $[n, n]$ 
18  text_affinity  $\mathbf{S}_i^T = \text{np.dot}(\mathbf{T}_i, \mathbf{T}_i.T)$  #  $[n, n]$ 
19   $\mathcal{L}_{\text{SaCo}} = \text{np.sum}(\text{np.abs}(\mathbf{S}_i^I - \mathbf{S}_i^T))$ 
20  # pseudo-affinity mimic loss
21   $\mathcal{L}_{\text{mimic}} = \text{np.sum}(\text{np.abs}(\mathbf{S}_i^I - \tilde{\mathbf{S}}_i^I))$ 
22  # total loss
23   $\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{cont}} + \alpha \mathcal{L}_{\text{SaCo}} + \beta \mathcal{L}_{\text{mimic}}$ 
24 end
```

version in Table 1. Below are detailed descriptions of these two datasets:

CC3M [13]. This dataset was collected from 5 billion web pages, and public by Google in 2018. It contains 3,318,333 image-text pairs, where the image descriptions are obtained from the HTML alt-text attribute. Unfortunately, around

Dataset	Original	Our Collected
CC3M	3.3M	2.8M
YFCC15M	15.4 M	15.0M

Table 1. Comparison of the size of the pre-train dataset under the original version and our collected version. Due to the expired image link, our collected version has less data compared to its original counterpart.

0.5M images are inaccessible due to the broken image links, so we finally collected around 2.8M image-text pairs for pre-training.

YFCC15M [7]. This dataset is a subset of YFCC100M [14]. It has two primary versions: YFCC15M-V1 was created by applying the same filtering rule used in CLIP [12], while YFCC15M-V2 was obtained through DeCLIP [7] using a distinct filtering strategy. In addition, the YFCC15M-V2 dataset also encompasses additional data collected from the Internet. Since YFCC15M-V2 is of superior quality than YFCC15M-V1, we use YFCC15M-V2 in our experiments and omit it to “YFCC15M”.

LLaVA-595K [9]. This dataset was constructed by LLaVA [9]. It contains 595k image-text pairs filtered from CC3M to strike a balance between concept coverage and training efficiency.

2.2. Pre-train Settings

In our paper, we explore two pre-training strategies, namely pre-training from scratch and continue pre-training, respectively. The training settings vary from the pre-training strategies, which are detailed as follows.

Pre-training from Scratch. The models are trained for 32 epochs with a batch size of 2048. We adopt AdamW [10] optimizer with a weight decay of 0.2. The learning rate first increases linearly from 0.0001 to 0.001 within one epoch, and then gradually decays until zero following the cosine anneal strategy.

Continue Pre-training. Given a well-trained converged vision-language model, we proceed to train it with our SaCo loss as an additional objective. The training setting mainly follows the baseline, except for fewer epochs and a smaller learning rate. Specifically, when continuously pre-training on OpenAI’s CLIP [12] model, we employ the AdamW [10] optimizer with a training duration of one epoch. The learning rate is set to 1e-6 and remains constant throughout the training.

3. Downstream Evaluation Tasks

We perform evaluations across a wide range of tasks, including both cross-modality tasks and single-modality

Algorithm 2: Continue pre-train a well-trained CLIP with our SaCo loss

Input: image-text pair dataset $\mathcal{D} = \{(x_j^I, x_j^T)\}_{j=1}^D$, total iterations T , batch size n , loss weight α , well-trained CLIP model (including projection weight W^I and W^T , temperature τ , image_encoder, text_encoder)

```

1 for  $i$  from 1 to  $T$  do
2   # sample a mini-batch of images and texts
3   sample  $(\mathbf{x}_i^I, \mathbf{x}_i^T) \sim \mathcal{D}$ 
4   # extract image and text embedding
5    $\tilde{\mathbf{I}}_i = \text{image\_encoder}(\mathbf{x}_i^I)$  #  $[n, d^I]$ 
6    $\tilde{\mathbf{T}}_i = \text{text\_encoder}(\mathbf{x}_i^T)$  #  $[n, d^T]$ 
7   # projection & L2 normalization
8    $\mathbf{I}_i = \text{L2\_norm}(\tilde{\mathbf{I}}_i \cdot W^I, \text{axis}=1)$  #  $[n, d]$ 
9    $\mathbf{T}_i = \text{L2\_norm}(\tilde{\mathbf{T}}_i \cdot W^T, \text{axis}=1)$  #  $[n, d]$ 
10  # contrastive loss
11  logits = np.dot( $\mathbf{I}_i, \mathbf{T}_i.T$ ) /  $\tau$  #  $[n, n]$ 
12  labels = np.arange(n)
13   $\mathcal{L}_{\text{cont}}^I = \text{cross\_entropy}(\text{logits}, \text{labels}, \text{axis}=0)$ 
14   $\mathcal{L}_{\text{cont}}^T = \text{cross\_entropy}(\text{logits}, \text{labels}, \text{axis}=1)$ 
15   $\mathcal{L}_{\text{cont}} = (\mathcal{L}_{\text{cont}}^I + \mathcal{L}_{\text{cont}}^T) / 2$ 
16  # SaCo loss
17  image_affinity  $\mathbf{S}_i^I = \text{np.dot}(\mathbf{I}_i, \mathbf{I}_i.T)$  #  $[n, n]$ 
18  text_affinity  $\mathbf{S}_i^T = \text{np.dot}(\mathbf{T}_i, \mathbf{T}_i.T)$  #  $[n, n]$ 
19   $\mathcal{L}_{\text{SaCo}} = \text{np.sum}(\text{np.abs}(\mathbf{S}_i^I - \mathbf{S}_i^T))$ 
20  # total loss
21   $\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{cont}} + \alpha \mathcal{L}_{\text{SaCo}}$ 
22 end
```

tasks. The datasets and implementation details for each task are described as follows.

3.1. Zero-shot Classification

Zero-shot classification is the most commonly used downstream task for transferability evaluation. The text embeddings of the candidate class names are used as the classifier to determine the probability that each image belongs to each class. The performance of this task is measured by top- k accuracy ($k=1,5$) in our paper.

Prompt Engineering. Since the image descriptions in pre-training dataset are usually a long sentence rather than a single-word class name, we employ prompts like ‘a photo of a class name’ to extend the class name through some pre-defined templates. For a fair comparison, we follow the prompt engineering strategy in CLIP [12] during zero-shot classification evaluation. Specifically, each category has 80 text prompts, *i.e.*, and class names with 80 different templates. We average the embedding of all the text prompts of each category as its final classifier. The prompt engineering can effectively narrow the domain gap between pre-training data and test data and take into account the diverse circum-

stances of the images.

Datasets. We evaluate the zero-shot classification on the widely-used *ImageNet-1K* dataset, which is designed to be representative of real-world visual data. It is a large-scale image classification dataset consisting of 1.28 million images belonging to 1,000 different classes. The classes cover a wide range of concepts, such as animals, plants, everyday objects, vehicles, and so on.

3.2. Zero-shot Image-Text Retrieval

Image-text retrieval task consists of two sub-tasks, namely image-to-text (I2T) retrieval and text-to-image (T2I) retrieval, respectively. Taking text-to-image retrieval as an example, it aims to retrieve the most images and texts given a textual description. We report the Recall@ k ($k=1,5,10$) for performance comparison. In order to truly reflect the property of pre-trained embedding space, we also evaluate this task under zero-shot settings without fine-tuning. Specifically, we extract the image and text embeddings from the corresponding encoders and perform retrieval based on the cosine similarity between candidate image-text pairs.

Datasets. We report the zero-shot image-text retrieval results on two benchmarks, MS-COCO [8] and Flickr30K [11]. *MS-COCO* [8] is a large-scale dataset designed for a variety of computer vision tasks, including image recognition, retrieval, segmentation, detection, and captioning. It contains 330,000 images of 91 common object categories, with more than 2.5 million object instances labeled and annotated with rich captions. For the image-text retrieval task, the dataset provides a set of high-quality image-caption pairs, where each image is accompanied by 5 captions describing the content of the image in detail. The captions are written by human annotators, which ensures that they are accurate and relevant to the image. The captions are designed to be descriptive, not just literal, and they capture the overall meaning and context of the image. For a fair comparison, we only evaluate its test set containing 5,000 images following previous works [1, 3, 6, 15]. *Flickr30K* [11] consists of 31,783 images of everyday activities and scenes, collected from the Flickr website. Each image has 5 textual descriptions, obtained from Amazon’s Mechanical Turk service. We evaluate its test set with 1,000 images.

3.3. Image Classification

Image Classification is a typical task to evaluate the quality of learned vision embedding space of a pre-trained vision-language model. In our experiments, we evaluate this task under a linear probe setting without disturbing the pre-trained image embedding space. Specifically, the image encoder is frozen and regarded as a feature extractor. For each training image, we first extract its image feature and then learn an extra linear classifier on top of it to predict

the probability of each class. We report top-1 accuracy to measure the performance.

Linear Prob Settings. We train a linear classifier on top of the frozen features extracted from the image encoder. The linear classifier is optimized using AdamW [10] for 90 epochs with a batch size of 1024. The learning rate is initially set to $5e-4$ and then decreases until zero following the cosine strategy. The image is resized to 224×224 during both training and evaluation.

Datasets. Our experiments are conducted on four image classification datasets. *ImageNet-1K* [2] is one of the most widely-used image recognition datasets with 1,000 categories. More information is provided in Sec. 3 above. *Tiny-ImageNet* [5] is a subset of the ImageNet dataset, including 100,000 images with 200 categories. Each class has 500 training images, 50 validation images, and 50 test images. It provides a more condensed dataset compared to the full-scale ImageNet, while still maintaining a diverse set of object classes and image variations. Each image has a size of 64×64 by default. *CIFAR-10* [4] is a classical dataset in the computer vision field, which was created by the Canadian Institute for Advanced Research (CIFAR). It consists of 60,000 images, 50,000 for training, and 10,000 for testing, each of size 32×32 pixels. All the images are uniformly divided into 10 categories, with 6,000 images per class. *CIFAR-100* [4] is an extension of the CIFAR-10 dataset, designing to be more challenging and comprehensive. It consists of 60,000 images, each of which is assigned a category label within 100 classes and has a resolution of 32×32 pixels. Similarly, these images are split into training and testing sets with 50,000 and 10,000 images, respectively. The categories cover a wide range of fine-grained concepts, including objects, animals, vehicles, and natural scenes.

4. More Experiments

4.1. Pre-train from Scratch vs. Continue Pre-train

As mentioned in the main paper, we present two pre-training strategies when incorporating our SaCo loss with the baseline model, namely pre-training from scratch and continue pre-training. Pre-training from scratch is the most commonly used strategy during vision-language pre-training, while continue pre-training is a more efficient manner to improve the embedding space of a well-trained model. In order to investigate the effect of these two pre-training strategies, we train the CLIP-R50 model with our SaCo loss from scratch and continuously, respectively.

As shown in Table 2, continue pre-training achieves consistent improvements on both tasks, especially the retrieval task. Pre-training from scratch can bring further improvements, among which classification tasks benefit more significantly. This demonstrates that pre-training from scratch

Method	Zero-shot Classification				Zero-shot Image-text Retrieval				
	ImageNet-1K [2] Top-1 Accuracy	Flickr30K [11]		MS-COCO [8]		Image → Text		Text → Image	
			Image → Text Recall@1	Recall@5	Text → Image Recall@1	Recall@5	Recall@1	Recall@5	Recall@1
CLIP	17.9	29.2	58.5	24.8	51.7	16.4	38.2	13.3	32.4
CLIP + Ours [◇]	20.2 (+2.3)	40.9 (+11.7)	70.6 (+12.1)	30.5 (+5.7)	56.5 (+4.8)	21.1 (+4.7)	45.0 (+6.8)	16.0 (+2.7)	36.2 (+3.8)
CLIP + Ours [♣]	22.5 (+4.6)	42.0 (+12.8)	72.1 (+13.5)	31.8 (+7.0)	57.8 (+6.1)	21.1 (+4.7)	45.3 (+7.1)	16.0 (+2.7)	36.5 (+4.1)

Table 2. Comparison of different pre-training strategies. All the models are pre-trained on the CC3M dataset and evaluated on zero-shot classification and zero-shot image-text retrieval tasks. We employ CLIP with an R50 image encoder as the baseline model. The superscript [♣] represents the corresponding model pre-trained from scratch. The superscript [◇] means that the model is obtained by continuously pre-training the baseline model equipped with our SaCo loss.

Method	Image Encoder	ImageNet-1K [2]	Tiny-ImageNet [5]	CIFAR-10 [4]	CIFAR-100 [4]
CLIP	R50	46.4	30.5	62.5	41.7
CLIP + Ours	R50	60.7 (+14.3)	37.5 (+7.0)	75.4 (+12.9)	55.4 (+13.7)
CLIP	ViT-B/32	36.0	48.4	77.0	54.6
CLIP + Ours	ViT-B/32	48.7 (+12.7)	55.7 (+7.3)	88.7 (+11.7)	69.3 (+14.7)
CLIP	ViT-B/16	45.4	42.5	77.9	56.7
CLIP + Ours	ViT-B/16	56.1 (+10.7)	49.7 (+7.2)	89.2 (+11.3)	70.0 (+13.3)

Table 3. Linear prob image classification results (top-1 accuracy). All the models are pre-trained from scratch on the CC3M dataset.

performs best and maximizes the benefits of our approach. This can be attributed to the fact that continue pre-training is more about promoting the cross-modal affinity consistency by slightly adjusting the embedding space, while during pre-training from scratch, our loss can consistently play a role in the learning process of the embedding space from chaos to alignment, giving it more opportunities to converge to a better state.

4.2. Image Classification

Table 3 compares the linear probe image classification performance on four datasets. The performance is evaluated on various datasets, including ImageNet-1K, Tiny-ImageNet, CIFAR-10, and CIFAR-100. Comparing the results, it is evident that the combination of CLIP with the additional method consistently improves the classification accuracy across all encoders and datasets. Notably, our approach achieves significant performance gains in all scenarios. For example, when using the R50 encoder, the CLIP + Ours approach achieves an accuracy of 60.7% on ImageNet-1K, showing a remarkable improvement of 14.3 percentage points. Similarly, for Tiny-ImageNet, CIFAR-10, and CIFAR-100, our approach demonstrates substantial accuracy gains of 7.0%, 12.9%, and 13.7%, respectively. These results highlight the effectiveness of our approach in enhancing the performance of linear prob image classification.

We also present a feature visualization analysis of CLIP and the model trained using our proposed approach on the CIFAR-10 dataset in Figure 1. It is evident that our approach significantly improves the feature distribution in the

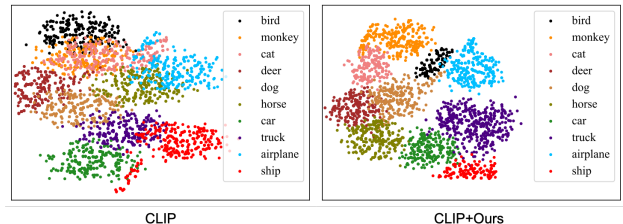


Figure 1. Feature visualization on the CIFAR-10 dataset. The visualization technique employed is T-SNE, which enables the representation of high-dimensional feature vectors in a lower-dimensional space.

feature space of CLIP. The problem of intertwined, overlapping, and mixed features between different classes is notably alleviated. For instance, the distribution of features for objects such as *airplane* and *bird* becomes closer in the feature space, indicating a more reasonable semantic relationship learned by our approach.

4.3. Computational Cost Analysis

Since our loss is only used during training, it has no impact on the inference efficiency and only affects the training computational cost. Table 4 shows the quantitative comparison in terms of the GPU memory and time consumption per iteration. We report the average results for 10 runs. It can be found that our loss only brings minor additional GPU memory cost (14.73 vs. 14.85 GB) and time consumption (0.66 vs. 0.69 seconds). The cost of pre-training from scratch is slightly higher than continue pre-training due to the pseudo-


		Retrieved Texts					
 <p>Query Image</p>	<p>An officer in a black uniform and hat stands to the left of a large structure with other officers in the background.</p>	<p>A female police officer, wearing an officer's hat and sunglasses, stands in uniform in front of a window-lined street block.</p>	<p>A female police officer in a cap and navy uniform smiles while wearing sunglasses outside of a shop.</p>	<p>A large, green, navigating wheel is held by an older man in a captain's hat and uniform.</p>	<p>The chef is trying to debone a chicken.</p>	<p>Baseline</p>	
	<p>A man in uniform is standing in front of a large building.</p>	<p>Outside a building, a uniformed security guard looks at the camera from behind a fence.</p>	<p>A security officer with a tiny face and big glasses leans on a metal gate looking into the camera.</p>	<p>An officer squints directly at the camera as he leans on a metal bar.</p>	<p>A guard is on the look out while on duty.</p>		<p>Baseline + Ours</p>

Figure 2. Visualization of zero-shot image-text retrieval. The baseline model is CLIP with R50 image encoder. All the models are pre-trained from scratch on CC3M dataset. The example are selected from Flickr30K dataset.

Method	GPU memory	Time (per iteration)
CLIP	14.73 GB	0.66 s
CLIP + Ours (continue pre-train)	14.81 GB	0.68 s
CLIP + Ours (pre-train from scratch)	14.85 GB	0.69 s

Table 4. Comparison of pre-training cost in terms of GPU memory and time consumption per iteration. The baseline model is CLIP with ViT-B/32 image encoder.

affinity mimicking objective. Note that the pseudo-affinity can be pre-extracted before training, so it will not lead to much computational burden.

4.4. More Visualization

Figure 2 presents visualizations for image-text retrieval results. It is evident that our approach significantly improves the relevance of retrieved text to the image content compared to the baseline. For instance, the last two retrieved texts by the baseline method are almost irrelevant to the image content, such as "chef", "chicken" and "wheel". In contrast, our approach promotes the retrieval of more relevant text that better aligns with the image content. This improvement is critical in enhancing the overall performance of the CLIP model in image-text retrieval tasks.

5. Discussion on Training Instability Problem

As mentioned in Sec.4.2.1 in the main paper, when pre-training a vision-language model with our loss from scratch, a problem of training instability will be encountered. In particular, our SaCo loss first drops sharply, then begins to rise for a while, and finally decreases again in the later training stage (refer to the yellow curve of Figure 3b in the main paper). Through experimental analysis, we find that the underlying reason is related to the trade-off between contrastive learning and affinity consistency constraint. In the early training stage, the affinity in both modalities is extremely noisy due to the chaotic embedding space. At this time, it is ineffective to overemphasize the affinity consistency, and the excessive consistency constraint will hinder

the optimization of embedding space. Accordingly, we designed several strategies to solve this problem. Here, we give more discussion on the motivation, implementation details, experimental results, and analysis for these strategies.

Strategy-1: Incremental loss weight. Inspired by the above analysis, we relax the affinity consistency constraint when the affinity is not accurate enough. Considering that inaccurate affinity is more likely to appear in the early training stage, we gradually increase the weight α of the SaCo loss in Eq.(6) of the main paper as the training progresses. We compare several common weight-increasing schemes, namely linear, exponential, and polynomial in Table 5. The linear scheme performs best among them. Unfortunately, although such instability problem can be alleviated via this strategy to a certain extent (yellow curve vs. light-green curve of Figure 3b in the main paper), it still cannot be eradicated.

Strategy-2: Sample-wise masking. Intuitively, if the image and text representation is less precise or unaligned, it will also have a negative impact on affinity accuracy. In this case, our affinity consistency constraint should be relaxed and the optimization of representation requires sufficient tolerance. Motivated by this, for each image-text pair, we utilize the similarity between its image and text embeddings as a criterion to assess the quality of its representations. Larger similarity means more reliable representations. Given the similarity of all the image-text pairs, we can simply obtain a binary mask by thresholding the similarity, where the element "0" in the binary mask denotes the corresponding pair is ignored when computing the SaCo loss, and element "1" means the corresponding pair will be subjected to the SaCo loss. We call this strategy "Sample-wise hard masking". In addition, as the similarity ranges from 0 to 1, we can also treat the similarity as a sample-wise soft weight coefficient for SaCo loss, which is termed "Sample-wise soft masking". Table 5 compares the performance of these two masking strategies. It can be seen that their performance is similar, and hard masking is slightly better. So, the results of the sample-wise masking strategy shown in Figure 3b and Figure 5b of the main paper corre-

Method	Strategy	ImageNet-1K	Flickr30K		MS-COCO	
		Top-1 Acc.	I2T(R@1)	T2I(R@1)	I2T(R@1)	T2I(R@1)
CLIP	-	11.9	16.0	12.2	8.2	6.5
CLIP + Ours	-	16.2	21.3	15.8	11.0	7.9
CLIP + Ours	Incremental loss weight (<i>exponential</i>)	16.6	23.2	17.1	12.3	9.4
CLIP + Ours	Incremental loss weight (<i>polynomial p=2</i>)	16.3	22.9	17.0	12.1	8.9
CLIP + Ours	Incremental loss weight (<i>linear</i>)	17.7	23.4	17.3	12.5	9.4
CLIP + Ours	Sample-wise <i>hard</i> masking	17.2	22.8	16.2	12.1	8.6
CLIP + Ours	Sample-wise <i>soft</i> masking	17.1	23.5	16.3	11.9	8.5
CLIP + Ours	Pseudo-affinity mimicking	18.3	25.3	18.3	13.5	10.2

Table 5. Comparison of different strategies for training instability problem. We use CLIP with ViT-B/32 as the baseline model. All the models are pre-trained from scratch on the CC3M dataset. We evaluate zero-shot classification on ImageNet-1K [2] dataset, and zero-shot image-text retrieval on Flickr30K [11] and MS-COCO [8] datasets. “R@1” and “Acc.” are short for “Recall@1” and accuracy, respectively. “I2T” and “T2I” represent image-to-text retrieval and text-to-image retrieval, respectively.

spond to the hard masking for its slight superiority. However, this masking-based strategy is also unable to essentially solve this problem. Although the performance has improved, the SaCo loss still has a recovery stage (Figure 3b in the main paper).

Strategy-3: Pseudo-affinity mimicking. The above two ineffective strategies aim to avoid the excessive consistency constraints on inaccurate affinity. Different from them, we find that directly promoting the accuracy of affinity is a more effective solution. As described in the main paper, we introduce a pseudo-affinity mimicking objective to mimic the affinity of the training model with that of a well-trained model. Experimental results show that this strategy is the most effective one while with minor additional computational costs.

References

- [1] DeLong Chen, Zhao Wu, Fan Liu, Zaiquan Yang, Yixiang Huang, Yiping Bao, and Erjin Zhou. Prototypical contrastive language image pretraining. *arXiv preprint arXiv:2206.10996*, 2022. 3
- [2] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 3, 4, 6
- [3] Xiaoyi Dong, Jianmin Bao, Yinglin Zheng, Ting Zhang, Dongdong Chen, Hao Yang, Ming Zeng, Weiming Zhang, Lu Yuan, Dong Chen, et al. Maskclip: Masked self-distillation advances contrastive language-image pretraining. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10995–11005, 2023. 3
- [4] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 3, 4
- [5] Ya Le and Xuan Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015. 3, 4
- [6] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *International Conference on Machine Learning*, pages 12888–12900. PMLR, 2022. 3
- [7] 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. 2
- [8] 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 *Computer Vision—ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6–12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014. 3, 4, 6
- [9] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36, 2024. 2
- [10] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 2, 3
- [11] Bryan A Plummer, Liwei Wang, Chris M Cervantes, Juan C Caicedo, Julia Hockenmaier, and Svetlana Lazebnik. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. In *Proceedings of the IEEE international conference on computer vision*, pages 2641–2649, 2015. 3, 4, 6
- [12] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 1, 2
- [13] Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2556–2565, 2018. 1
- [14] Bart Thomee, David A Shamma, Gerald Friedland, Benjamin Elizalde, Karl Ni, Douglas Poland, Damian Borth, and

Li-Jia Li. Yfcc100m: The new data in multimedia research. *Communications of the ACM*, 59(2):64–73, 2016. [2](#)

- [15] Yuting Gao, Jinfeng Liu, Zihan Xu, Tong Wu, Wei Liu, Jie Yang, Ke Li, and Xing Sun. Softclip: Softer cross-modal alignment makes clip stronger. *arXiv preprint arXiv:2303.17561*, 2023. [3](#)