An empirical study of CLIP fine-tuning with similarity clusters

Shixuan Liu
University of Michigan
Ann Arbor, MI
shixuanl@umich.edu

Yiwei Lyu
University of Michigan
Ann Arbor, MI
yiweilyu@umich.edu

Honglak Lee
University of Michigan
LG AI Research
Ann Arbor, MI
honglak@eecs.umich.edu

Todd C. Hollon
University of Michigan
Ann Arbor, MI
tocho@med.umich.edu

Abstract

With the success of CLIP training for learning transferable visual representations, fine-tuning CLIP models on smaller datasets for better downstream performance is an important area of research. A method for improving CLIP models is to increase the difficulty of negative examples. While the majority of research has focused on manually crafting hard negative captions, this strategy requires additional engineering labor, fails to generalize to different domains, and causes additional overfitting. Here, we conduct an empirical study to systematically explore an alternative approach: construct minibatches that include similarity clusters to increase the difficulty of negative examples. We propose a generalized framework, called *SimCLIP*, for similarity-based CLIP fine-tuning. By enforcing that each minibatch contains clusters of similar examples, SimCLIP fine-tuning can improve model performance compared to standard CLIP fine-tuning. We extensively study which SimCLIP configurations and factors contribute most to downstream performance. We also analyze SimCLIP's performance on rare special sets, compositionality of attributes, and generalization across dataset sizes. Our observations provides better understanding of similarity-based minibatch construction methods as well as new insights into CLIP fine-tuning. The code for our experiments is available at https://github.com/sx-liu/SimCLIP/.

1 Introduction

In recent years, language-guided visual representation learning, such as CLIP [19], BLIP [14] and FLAVA [23], has become an effective method for learning visual representations. Contrastive objectives [1] are commonly used to align image-text pairs in a joint representation space. One example is the infoNCE loss [25, 19], which is effective in learning joint vision-language representations and extendable with other objectives [23, 30, 31]. The most well-known joint vision-language representation model is CLIP [19], which is trained via infoNCE loss over a large corpus of image-text pairs and has demonstrated strong performance in various vision-only and vision-language downstream tasks.

Pre-trained CLIP models are often fine-tuned for improved performance within specific domains or downstream tasks, where the fine-tuning process involves continued training with the same CLIP objective but over a specific fine-tuning dataset of image-text pairs. However, a problem with CLIP

38th Workshop on Fine-Tuning in Machine Learning (NeurIPS 2024).

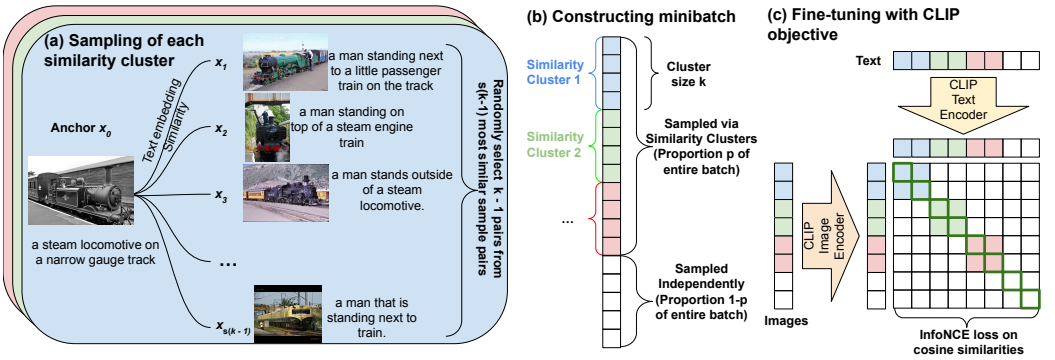

Figure 1: An overview of SimCLIP, a generalized framework for CLIP fine-tuning via similarity clusters. **(a)** We sample each similarity cluster via text embedding similarity; **(b)** We construct each minibatch with a proportion of it consisting of similarity clusters and the remaining sampled independently, and **(c)** We perform fine-tuning with CLIP objective.

fine-tuning is that CLIP training alone is often too "easy" and allows the model to shortcut without improving visual representations [29]. Intuitively, contrastive objectives will only force the model to gain enough representation power to distinguish between different images presented within the same minibatch, and images within a minibatch are distinct enough that the model can easily distinguish them without needing to learn to represent image details. Therefore, one prominent research direction for improving CLIP fine-tuning has been how to create effective hard-negatives within minibatches during fine-tuning.

Previous work focuses on manually constructing hard-negative captions [29, 31]. These approaches rely on heavily-engineered rule-based construction methods, which can be labor-intensive and hard to automate/generalize, and have the risk of overfitting the model on certain language pattern [10]. Another possible approach for making the negatives in a minibatch "hard" is through inclusion of similarity clusters, such that we sample similar image-text pairs from the fine-tuning dataset instead of manual construction of new hard-negative data. The main advantage of this approach is that it does not require manual engineering of specific rules for hard negative construction, while still being able to train with high-quality, hard negative examples. While there is some relevant research on the optimal sampling strategy for single-mode contrastive learning objectives [20], there has been little existing exploration in the context of multi-modal contrastive learning. The existing attempt at this approach [29] only tried one particular setting: adding one additional similar image-text pair to every instance in the minibatch.

In this paper, we aim to systematically explore this similarity cluster approach. We propose *SimCLIP*, a generalized framework of CLIP fine-tuning with similarity clusters that allow construction of minibatches with varying similarity cluster sizes, proportions, and other configurations. We then conduct extensive experiments over these various configurations and analyze their effects on various downstream task performances. The key contribution of this empirical study is as follows:

- We propose SimCLIP, a generalized framework of CLIP fine-tuning with similarity clusters.
- We conduct extensive experiments over various configurations within the SimCLIP framework, and identify factors that are important (e.g. cluster size and proportion) and unimportant (e.g. neighborhood size) for downstream performance.
- We identified downstream tasks where SimCLIP will help. We empirically demonstrate that SimCLIP learns better representations of rare specialized classes. We observe that SimCLIP may not perform well on tasks that require understanding order and compositionally of attributes.
- We demonstrate that SimCLIP performs well when fine-tuning on both small datasets such as COCO [16] and larger datasets such as CC3M [22].

2 Related Work

Fine-tuning in Multi-modal Contrastive Learning After the original CLIP model [19] was proposed, a series of methods have been introduced to improve the fine-tuning quality of the multi-modal contrastive learning, which can be broadly categorized into two research directions. The first direction focuses on how to further enhance model performance by augmenting the data and fine-graining the alignment. FILIP [27] proposes token-level alignment by maximum similarity between visual and textual tokens. Other works, such as OSCAR [15], VinVL [32], align multi-level semantic elements inside the texts and images. While OSCAR and VinVL only extracts these semantics within the visual modality, some follow-up studies propose more fine-grained semantic space alignment for both modalities, such as Pyramidclip [5] and Softclip [4]. These methods usually rely on some off-the-shelf ROI feature extractors or the rich annotation of some specific datasets. Another direction is to better supervise or regularize the original CLIP objective. Cyclip [7] proposes a regularization on representation space, which enforces the cycle consistency of the learned features. Notably, regularizations can also be used along with aforementioned augmentations. For example, SGVL [9] leverages scene graphs to incorporate structured representations into fine-tuning.

Hard Negative Mining Hard negatives are leveraged to improve representation learning [8, 26, 6], contrastive learning [20, 11], as well as vision-language representation learning [29, 31]. Suitably constructed hard negatives can also be used for in-depth evaluations of these representations [24, 29, 18, 33, 10]. Specifically, NegCLIP [29] opens the boundary of hard negative construction by suggesting swapping the position of certain components inside a positive caption. Following this direction, a few other works, such as SVLC [3], Rösch et al. [21], investigate how to better extend or fine-grain these hard negatives. Other works focuses on how to better incorporate these negatives into training, such as through intra-modal and ranking cross-modal regularizations [31].

While all these previous research on vision-language representation learning concentrates on some data manipulations, this paper will try to uncover the possibility of mining hard negative without those artifacts, such as extracted semantic elements or manually synthesized hard negatives. We will explore a fine-tuning strategy that only uses existing training data and enforce hard contrastive objectives through special minibatch construction methods.

3 Method

The strong alternative sampling purposed in [29] suggests the following: When constructing a training minibatch, for each randomly sampled training instance, we will additionally sample one out of its three nearest neighbors in the embedding space to include in the minibatch. While not discussed in detail in the original paper, this approach intends to increase the training minibatch similarity and potentially fix the lack of hard contrast from the image modality.

Algorithm 1 Similarity Sampling

Input: Training data $X = (x_1, x_2, \ldots, x_n)$, Embeddings $E = (e_1, e_2, \ldots, e_n)$, Batch size N, Cluster size k, Similarity proportion p, Neighborhood size s
Output: Sampled minibatch B
Randomly sample a minibatch $B \subset X$ of size $N(1-p)$
Calculate the number of clusters: $n_{cluster} = \frac{pN}{k}$
for $i = 1$ to $n_{cluster}$ **do**
    Randomly sample an anchor point $x_m \in X$
    Compute cosine similarities between embeddings e_m and embeddings of other points $(e_1, e_2, ...)$
    Construct Neighborhood S by selecting top $s(k-1)$ points by largest embedding cosine similarities to x_m
    Randomly select $k-1$ points from S to form S'
    Add the anchor x_m and points in S' to B
end for
return B

We follow the trace of this work and propose a more generalized version of strong alternative sampling, as illustrated in Figure 1: for total batch size N, instead of sampling 1 close neighbor per instance in the minibatch, we construct a proportion p of our minibatch with $\frac{pN}{k}$ similarity clusters where each similarity cluster has size k, while the remaining $1-p$ proportion is randomly sample individual instances. At the beginning of each training epoch, we will calculate the text embeddings of all the training samples using the text encoder of the CLIP model. To construct each similarity cluster within each minibatch, we first randomly sample one training image-text pair as the anchor point, then

we retrieve $s(k-1)$ nearest neighbors of the anchor by text embedding to form a "neighborhood" (where s is the "neighborhood size"), and lastly, we randomly select $k-1$ instances from the neighborhood to form a similarity cluster of size k together with the starting instance. We call the CLIP models fine-tuned with minibatches with similarity clusters SimCLIP for short. Existing works can often be viewed as a special case of this generalized scheme: we have regular CLIP fine-tuning when $p=0$ or $k=1$; the strong alternative sampling in NegCLIP [29] (i.e. NegCLIP without manually constructed additional hard negative text samples) can be viewed as SimCLIP with $k=2$, $p=1.0$ and $s=3$.

SimCLIP also supports **warmup similarity proportion**: instead of setting the similarity proportion p to be a constant, we can also gradually increase this factor exponentially throughout the fine-tuning epochs until it reaches a final proportion of p. We divide the total number of epochs into I intervals of similar length. We start with a similarity proportion of $0.5^{I-1}p$ for the first interval, and then we double the similarity proportion for each subsequent interval (so the proportion for the final interval is p).

4 Experiments

4.1 Research Questions

Since SimCLIP is a general framework with configurable settings, we need to find out how each factor affects fine-tuning performance, and which configurations are optimal. Our first research question is **RQ1: How do various configurable factors of SimCLIP affect fine-tuning performance?** Specifically, we are interested in the effects of the following factors: cluster size k, similarity proportion p, warmup vs fixed similarity proportions, neighborhood size s, similarity embedding type (i.e. constructing similarity clusters via text embeddings vs image embeddings), and online vs offline similarity embeddings (i.e. whether we use current model's embeddings to construct similarity clusters or use the pre-fine-tune model's embeddings).

In addition to the configurations, we also conduct analysis over SimCLIP's performance under various scenarios to determine the strengths/weaknesses of SimCLIP. We pick 2 specific scenarios that were known to be challenging for regular CLIP fine-tuning: **rare special sets (RQ2)** and **compositionality of attributes (RQ3)**. We explain the intuition behind each scenario and detail the experiment setups within sections 4.5 and 4.6. Through **RQ2** and **RQ3**, we aim to gain a better understanding of which situations are ideal for applying SimCLIP.

Finally, due to computational constraints, most of our experiments will be conducted on a relatively small dataset, and we would like to verify **RQ4: does SimCLIP, especially with the optimal configuration from RQ1, works well when fine-tuning on a much larger dataset?**

4.2 Dataset and Fine-tuning Details

Dataset We use Microsoft COCO dataset [16] as the primary data source in most of the experiments. COCO dataset consists of 118K images in the training split and 5k images in the validation split. Each image in this dataset includes 5 corresponding captions. We randomly sample one out of five captions each time during CLIP fine-tuning.

Fine-tuning We use *clip-vit-base-patch32*[1] variant of OpenAI CLIP model as the baseline pre-trained model, and follow the regular CLIP objective [19] during the fine-tuning process. On COCO dataset, under each configuration, we fine-tune the model for 20 epochs on 4 A40 GPUs with batch size $N=1024$, and select the checkpoint with top average Recall@1 accuracy on COCO validation set for downstream evaluation. All the other hyperparameters are listed in table 3 in appendix.

4.3 Evaluation Tasks and Protocol

In order to evaluate the overall performance as well as generalization capabilities of the fine-tuned CLIP models, we evaluate the models on COCO validation set and several downstream datasets. COCO and Flickr30k [28] validation sets for retrieval evaluation, and CIFAR-100 [12] and ImageNet [2] for classification evaluation. Flickr30k dataset is a large corpus of image-text pairs with well annotated data samples. For fairness of comparison, we use `CLIP_benchmark`[2] as the platform

[1] https://huggingface.co/openai/clip-vit-base-patch32

[2] https://github.com/LAION-AI/CLIP_benchmark

for evaluation. The CIFAR-100 dataset includes 10,000 tiny images of resolution 32×32 for testing, which consists of 100 classes evenly distributed. ImageNet validation set contributes another 50,000 images evenly sampled from 1,000 categories, containing a wide variety of objects. For these two classification tasks, we follow the evaluation instructions proposed along with the original CLIP paper[3]. These datasets supports two different evaluation strategy and contains single or multiple object images of varied resolutions, which provides a comprehensive evaluation for our CLIP models.

4.4 RQ1: How various configuration factors affect fine-tuning performance

Cluster Size k, similarity proportion p, and warm-up vs fixed proportion These three factors determine the "difficulty" of the minibatches. Larger k and p make the contrastive objective more challenging for the model by including larger similarity clusters (so more "hard" contrastive pairs) and more similarity clusters in each minibatch, but they also make overall data distribution of the minibatch farther away from the original fine-tuning dataset, which may destabilize training. For this evaluation, we repeat each configuration with 3 different random seeds to obtain more accurate results.

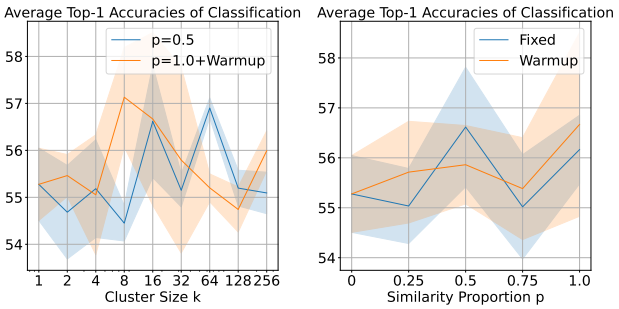

Figure 2: Downstream Classification performance of SimCLIP with varying cluster size k and similarity proportion p, with and without warmup. Note that when $k = 1$ or $p = 0$, we perform regular CLIP fine-tuning. The error bars are estimated with 3 random seeds. For this experiment, we set $s = 1$, and we use $k = 16$ for the proportion plot on the right. We found that optimal k usually lies within the medium values (between 8 and 64); with fixed similarity proportion, the optimal p is 0.5, while with warmup the optimal p goes to 1.0.

We show the effects of cluster size on downstream classification task accuracies in Figure 2. Among the classification task results, we found that the best similarity proportion is $p = 0.5$ with fixed similarity proportion and $p = 1.0$ with warm-up. This indicates that, without warm-up, filling half of each minibatch with similarity clusters seems to achieve the best balance between training stability and difficulty of the contrastive objective, while with warmup we can go up to $p = 1.0$ since in the earlier epochs the effective proportion is lower. We also found that for warmup and fixed proportions with optimal p, the best cluster size k seems to be 8 and 64 respectively, with second best both at $k = 16$, all of which are medium values.

We also show results on the retrieval tasks in Figure 3. We found that the optimal k on retrieval tasks (64-128) is higher than that on classification tasks, but the performance still falls off when k reaches 256.

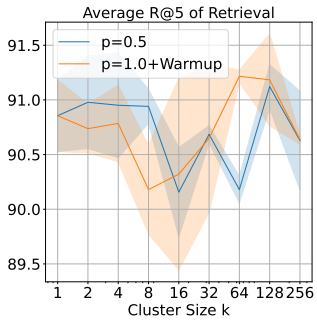

Figure 3: Retrieval accuracies of SimCLIP with varying cluster size k, similarity proportion p, with and without warmup. The best k on retrieval tasks are slightly larger (between 64 and 128 across configurations)

Overall, the results indicate that cluster sizes larger than those used in prior works ($k = 2$) can improve performance on downstream tasks. The optimal k are medium-sized and cannot be too large. A medium-sized similarity proportion works well without warmup, but with warmup we can use a large similarity proportion. We also found that fine-tuning with warmup yields slightly better top performance averaged across classification and retrieval tasks. We include full results on individual tasks in Sections B.1 and B.2 in Appendix.

Neighborhood size We demonstrate how neighborhood size s affects the downstream task performance in Figure 4. We do not see a clear trend on whether larger s helps

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

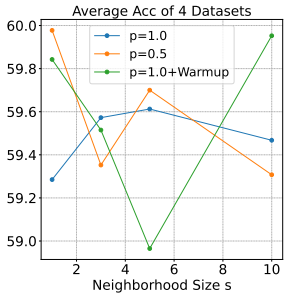

Figure 4: Downstream task performance of SimCLIP with different neighborhood sizes, with $k = 16$. This shows that choosing larger neighborhoods ($s > 1$) is unnecessary.

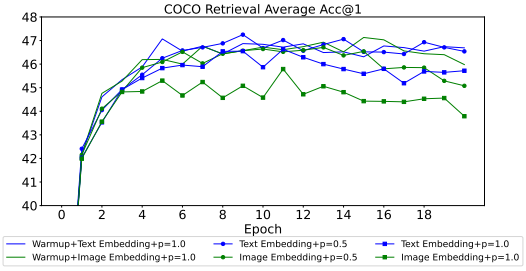

Figure 5: Retrieval accuracies of SimCLIP with various configurations on MSCOCO validation set over the epochs fine-tuned ($k = 16, s = 1$). Text embeddings yield more stable performance across various configurations.

with downstream performance. In fact, picking the simplest setting $s = 1$ yields good performance, especially on the classification tasks. This indicates that selecting a neighborhood size larger than 1 is unnecessary. We include full results of this analysis in Figure 10 in Appendix.

Online vs Offline We evaluate the performance difference between online and offline text embeddings in Table 1, where we compare all the accuracies on two top-ranking variants of SimCLIP. We found that the SimCLIP with online embeddings consistently outperforms offline embeddings among the classification tasks, but online embeddings only win half of the retrieval metrics. Therefore, using online text embeddings may be worthwhile despite computational overhead if we want optimal downstream classification performance.

		p=0.5		p=1.0 warmup	
Task Type	Dataset	Online	Offline	Online	Offline
Retrieval	COCO R@1	**62.77**	62.25	60.50	**64.38**
	COCO R@5	**86.39**	86.19	83.93	**87.38**
	Flickr30k R@1	78.36	**78.70**	**78.82**	78.63
	Flickr30k R@5	93.86	**94.79**	**94.28**	93.81
Classification	CIFAR-100 Top1	**61.77**	60.29	**63.31**	60.31
	CIFAR-100 Top5	**87.02**	85.23	**87.84**	85.94
	ImageNet Top1	**53.72**	52.87	**55.07**	52.25
	ImageNet Top5	**80.59**	80.34	**81.46**	79.41

Table 1: Downstream Task Performance of SimCLIP with Online vs. Offline Embeddings, $k = 16, s = 1$. Online Embedding achieves better performance on the majority of metrics.

Text Embeddings vs Image Embeddings We generally found that using text embedding to construct similarity clusters yields more consistent results in SimCLIP fine-tuning. We compare the validation retrieval performance over the fine-tuning epochs in Figure 5 under three different settings. While using both embeddings yielded similar performance under some configurations, the performance is significantly worse with image embedding under other configurations (such as p=1.0 without warmup). Therefore, using text embeddings to compute similarity clusters yields more stable performance across configurations. Full results of this analysis are shown in Figure 11 in Appendix.

4.5 RQ2: Does SimCLIP improve performance on images from rare special sets?

One major intuition behind the similarity cluster approach is the following scenario: with a fine-tuning dataset D and batch size N, there is a small subset $B \in D$ of a certain concept with $|B| << \frac{|D|}{N}$, then in most of the minibatches during regular CLIP fine-tuning, there will be at most one instance from B in the minibatch. The model only needs to learn to distinguish whether an instance belongs to B or not. Moreover, the model is not forced to learn differences within B. SimCLIP mitigates this problem by forcing similar instances within the same minibatch, thus forcing the model to learn details that can distinguish between different B instances when similarity clusters of multiple B instances appear in the minibatches.

To verify this intuition, we conduct the following experiment: we mix 50 MNIST [13] digits (5 from each class, with caption "A MNIST digit of {}") into the training split of COCO dataset, and then we fine-tune CLIP model with the mixed data. Our evaluation metric is zero-shot digit classification on MNIST test set using the same text templates as the captions. The 50 MNIST digits represent the "rare special set" situation: they belong to a separate class from regular COCO images, they are

very rare in number (50 in 118K), and there is a well-defined "intra-B" task (digit classification) that the pre-trained CLIP model performs poorly on (only 20.61% accuracy). In order to perform well on the digit classification task, the model must learn to distinguish between MNIST digits during fine-tuning, rather than simply being able to distinguish an MNIST digit from regular COCO images.

We fine-tune CLIP on our mixed dataset with both regular CLIP and SimCLIP (k=16, p=1.0, no warmup) for 10 epochs, and we show the zero-shot digit classification accuracy in Figure 6. We see that SimCLIP is clearly able to better distinguish the different digits in MNIST with the same amounts of fine-tuning. Therefore, SimCLIP indeed improves performance on rare special sets.

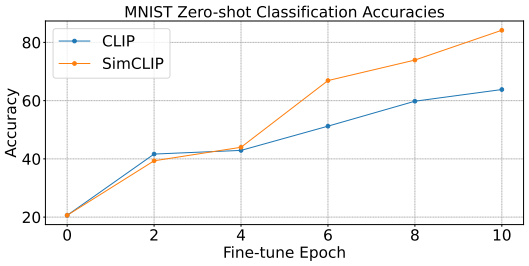

Figure 6: The MNIST zero-shot classification accuracies at the end of each epoch for the MNIST Mix-in experiment. SimCLIP significantly improves the model's representations of digits from just 50 MNIST digits mixed into 118K COCO image-text pairs.

4.6 RQ3: Does SimCLIP help CLIP with understanding compositionality of attributes?

Prior works found that one significant weakness of CLIP models are their inability to understand and process compositionality, yielding image/text representations that are invariant to different compositions of attributes [29]. Several benchmarks have been created to thoroughly evaluate this weakness, such as ARO [29] and SugarCrepe [10]. ARO contains four subtasks built on three separate datasets, the `VG-Attribution`, `VG-Relation`, `COCO Order` and `Flickr30k Order`, where the evaluated model must distinguish between similar captions with flipped word ordering or relationships. SugarCrepe is another benchmark designed for compositionality evaluation. that is designed to be less biased and hackable compared to previous benchmarks. There are 3 different types of tasks included in this benchmark, `REPLCAE`, `SWAP` and `NEGATE/ADD`, corresponding to 3 different methods of producing the confusion options.

These tasks have typically been tackled via manually designing hard-negative captions that targets compositionality of attributes. We evaluate various SimCLIP settings on these benchmarks to see if SimCLIP can improve CLIP's performance on these tasks as well. The results are shown in Figure 7. Unfortunately, SimCLIP did not seem to be able to improve the model's performance on most tasks, with only small gains on SugarCrepe REPLACE and NEGATE/ADD tasks and no improvement on any other tasks compared to regular CLIP fine-tuning (i.e. $k = 1$). Full results of this analysis is shown in Figure 12 in Appendix.

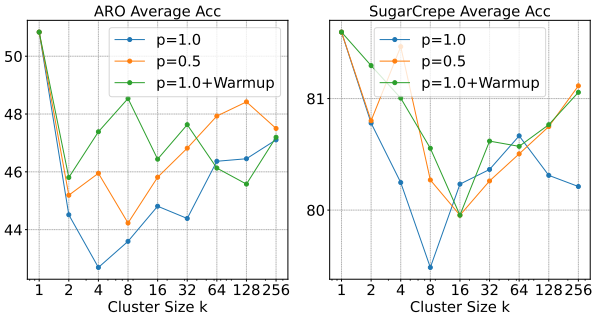

Figure 7: Accuracies of SimCLIP on ARO and SugarCrepe benchmarks, with $s = 1$. Note that when $k = 1$, we perform regular CLIP fine-tuning, and larger k (i.e. SimCLIP) did not improve performance in most tasks.

4.7 RQ4: Does SimCLIP generalize to larger fine-tuning datasets than COCO?

To verify whether SimCLIP works on larger fine-tuning datasets than COCO, we fine-tuned the same CLIP model on CC3M [22], which has a 22.6X larger training set compared to COCO. We pick one of the better SimCLIP configurations determined in section 4.4 ($p = 1.0$, $k = 16$, $s = 1$, online with warmup). We follow the same fine-tuning protocol for CC3M, except with fewer total epochs (5 instead of 20) due to computational limits. We show the results in Table 2. We found that SimCLIP outperforms regular fine-tuning on all 4 evaluation tasks, which indicates that SimCLIP also works well on larger fine-tuning datasets.

5 Conclusions

In this empirical study, we systematically explored similarity-cluster-based CLIP fine-tuning. We proposed a generalized framework of CLIP fine-tuning with similarity clusters called SimCLIP, and we conducted extensive experiments and analysis to determine the best configurations and the effects of SimCLIP on downstream task performances. We also discovered situations where SimCLIP is good at (e.g. rare special sets) and not good at (e.g. compositionality of attributes) dealing with. Our findings can be informative for future researchers when deciding how to fine-tune a CLIP model and could bring new insights into future research in CLIP fine-tuning.

Dataset	CLIP	SimCLIP
COCO R@1	39.09	**39.94**
COCO R@5	70.72	**71.86**
Flickr30k R@1	66.19	**66.97**
Flickr30k R@5	87.08	**88.43**
CIFAR-100 Top1	62.17	**64.28**
CIFAR-100 Top5	87.65	**88.11**
ImageNet Top1	50.27	**52.40**
ImageNet Top5	78.01	**79.91**

Table 2: Downstream Task Performance of regular CLIP and SimCLIP after fine-tuning on CC3M.

Acknowledgements

This work was supported by the Chan Zuckerberg Foundation (CZI) Advancing Imaging Through Collaborative Project grant, UM Precision Health Investigators Awards grant program (T.H.), Translational AI Award from the UM Department of Neurosurgery (T.H.), UM Frankel IHBH Innovative Multidisciplinary Research Pilot Award (T.H.), and the UM Research Scouts program (T.H.). This research was also supported, in part, through computational resources and services provided by Advanced Research Computing, a division of Information and Technology Services at the University of Michigan.

References

[1] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations, 2020.

[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, 2009.

[3] Sivan Doveh, Assaf Arbelle, Sivan Harary, Rameswar Panda, Roei Herzig, Eli Schwartz, Donghyun Kim, Raja Giryes, Rogerio Feris, Shimon Ullman, and Leonid Karlinsky. Teaching structured visionlanguage concepts to visionlanguage models, 2023.

[4] Yuting Gao, Jinfeng Liu, Zihan Xu, Tong Wu, Enwei Zhang, Ke Li, Jie Yang, Wei Liu, and Xing Sun. Softclip: Softer cross-modal alignment makes clip stronger. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 1860–1868, 2024.

[5] Yuting Gao, Jinfeng Liu, Zihan Xu, Jun Zhang, Ke Li, Rongrong Ji, and Chunhua Shen. Pyramidclip: Hierarchical feature alignment for vision-language model pretraining. *Advances in neural information processing systems*, 35:35959–35970, 2022.

[6] Weifeng Ge, Weilin Huang, Dengke Dong, and Matthew R. Scott. Deep metric learning with hierarchical triplet loss, 2018.

[7] Shashank Goel, Hritik Bansal, Sumit Bhatia, Ryan Rossi, Vishwa Vinay, and Aditya Grover. Cyclip: Cyclic contrastive language-image pretraining. *Advances in Neural Information Processing Systems*, 35:6704–6719, 2022.

[8] Ben Harwood, Vijay Kumar B G, Gustavo Carneiro, Ian Reid, and Tom Drummond. Smart mining for deep metric learning, 2017.

[9] Roei Herzig, Alon Mendelson, Leonid Karlinsky, Assaf Arbelle, Rogerio Feris, Trevor Darrell, and Amir Globerson. Incorporating structured representations into pretrained vision language models using scene graphs, 2023.

[10] Cheng-Yu Hsieh, Jieyu Zhang, Zixian Ma, Aniruddha Kembhavi, and Ranjay Krishna. Sugarcrepe: Fixing hackable benchmarks for vision-language compositionality, 2023.

[11] Yannis Kalantidis, Mert Bulent Sariyildiz, Noe Pion, Philippe Weinzaepfel, and Diane Larlus. Hard negative mixing for contrastive learning, 2020.

[12] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[13] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

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

[15] Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, et al. Oscar: Object-semantics aligned pre-training for vision-language tasks. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXX 16*, pages 121–137. Springer, 2020.

[16] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context, 2015.

[17] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2019.

[18] Zixian Ma, Jerry Hong, Mustafa Omer Gul, Mona Gandhi, Irena Gao, and Ranjay Krishna. Crepe: Can vision-language foundation models reason compositionally?, 2023.

[19] 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, 2021.

[20] Joshua Robinson, Ching-Yao Chuang, Suvrit Sra, and Stefanie Jegelka. Contrastive learning with hard negative samples, 2021.

[21] Philipp J. Rösch, Norbert Oswald, Michaela Geierhos, and Jindřich Libovický. Enhancing conceptual understanding in multimodal contrastive learning through hard negative samples, 2024.

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

[23] Amanpreet Singh, Ronghang Hu, Vedanuj Goswami, Guillaume Couairon, Wojciech Galuba, Marcus Rohrbach, and Douwe Kiela. Flava: A foundational language and vision alignment model, 2022.

[24] Tristan Thrush, Ryan Jiang, Max Bartolo, Amanpreet Singh, Adina Williams, Douwe Kiela, and Candace Ross. Winoground: Probing vision and language models for visio-linguistic compositionality, 2022.

[25] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding, 2019.

[26] Chao-Yuan Wu, R. Manmatha, Alexander J. Smola, and Philipp Krähenbühl. Sampling matters in deep embedding learning, 2018.

[27] 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, 2021.

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

[29] Mert Yuksekgonul, Federico Bianchi, Pratyusha Kalluri, Dan Jurafsky, and James Zou. When and why vision-language models behave like bags-of-words, and what to do about it?, 2023.

[30] Yan Zeng, Xinsong Zhang, and Hang Li. Multi-grained vision language pre-training: Aligning texts with visual concepts, 2022.

[31] Le Zhang, Rabiul Awal, and Aishwarya Agrawal. Contrasting intra-modal and ranking cross-modal hard negatives to enhance visio-linguistic compositional understanding, 2024.

[32] Pengchuan Zhang, Xiujun Li, Xiaowei Hu, Jianwei Yang, Lei Zhang, Lijuan Wang, Yejin Choi, and Jianfeng Gao. Vinvl: Revisiting visual representations in vision-language models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5579–5588, 2021.

[33] Tiancheng Zhao, Tianqi Zhang, Mingwei Zhu, Haozhan Shen, Kyusong Lee, Xiaopeng Lu, and Jianwei Yin. Vl-checklist: Evaluating pre-trained vision-language models with objects, attributes and relations, 2023.

A Hyperparameters details

We provide additional details about hyperparameters used in our COCO experiments in Table 3.

Hyperparameter	Value
Batch Size	1024
Learning Rate	1.0e-5
Epochs	20
Warmup Steps	200
Image Mean	[0.48145466, 0.4578275, 0.40821073]
Image std	[0.26862954, 0.26130258, 0.27577711]
Image Augmentation	Resize; RandomCrop (0.8, 1.0)
Optimizer Momentum	$\beta_1 = 0.9, \beta_2 = 0.98$
Weight Decay	0.1
Eps	1.0e-6
Optimizer	AdamW [17]

Table 3: Hyperparameters of Fine-tuning Process

B Detailed Experiment results

B.1 Classification Details with various cluster size and similarity proportion

We present the detailed accuracies for classification tasks on CIFAR-100 and ImageNet in Figure 8. The top-1 and top-5 per-class accuracies are plotted against cluster size k and similarity proportion p respectively. Despite the high variance, we can still observe peaks at some medium values of k and p.

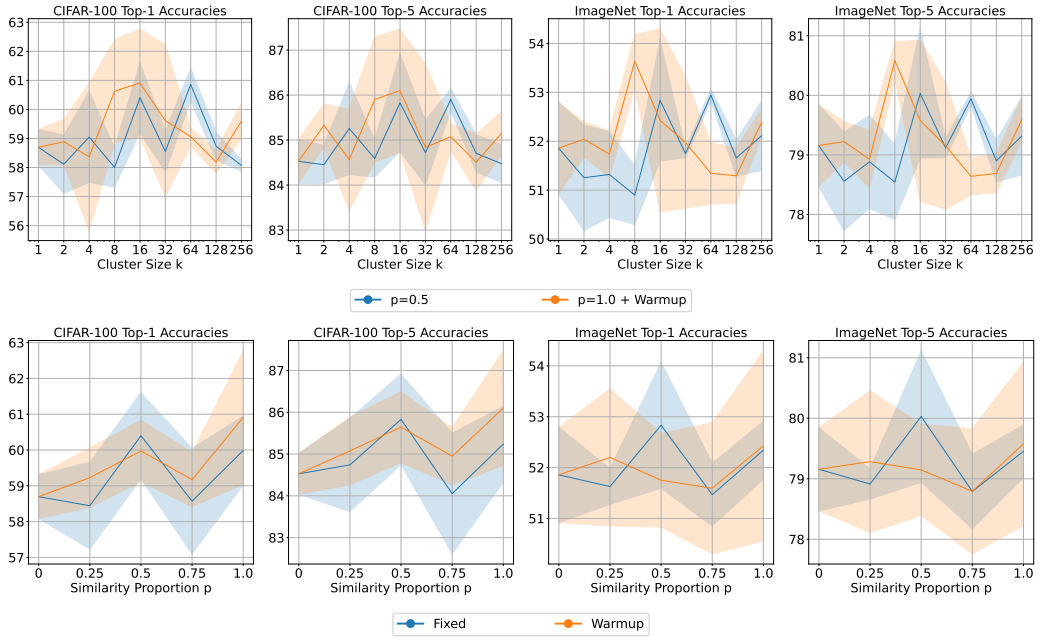

Figure 8: Downstream Classification performance of SimCLIP with varying cluster size k and similarity proportion p, with and without warmup. Note that when $k = 1$ or $p = 0$, we perform regular CLIP fine-tuning.

B.2 Retreival Details with various cluster size and similarity proportion

We present the detailed accuracies for retrieval tasks on COCO and Flickr30k in Figure 9. We separately plot the image-to-text (i2t) and text-to-image (t2i) recall accuracies against the cluster size k for two datasets.

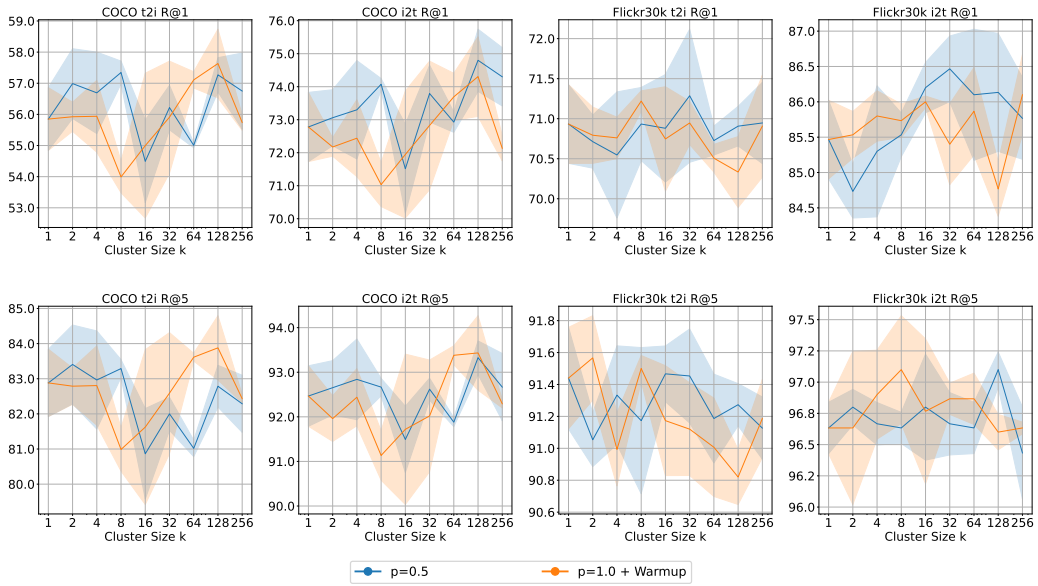

Figure 9: Retrieval accuracies of SimCLIP with varying cluster size k, similarity proportion p, with and without warmup. Note that when $k = 1$, we perform regular CLIP fine-tuning.

B.3 Neighborhood Size Experiment Result Details

We present the details of neighborhood analysis in Figure 10. We evaluate the classification and retrieval accuracies for the given SimCLIP configuration at $s = 1, 3, 5, 10$. The results shown in Figure 4 indicates that choosing a neighborhood size of $s = 1$ is sufficient as there is no clear trend of improvement with $s > 1$.

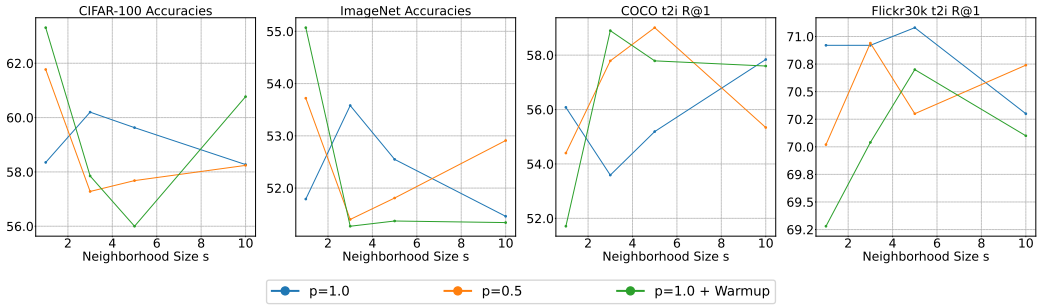

Figure 10: Downstream task performance of SimCLIP with different neighborhood size, with $k = 16$.

B.4 Text Embeddings vs. Image Embeddings Experiment Result Details

We present the details of the effects of different embeddings in Figure 11. We show the t2i and i2t accuracies on the validation set at the end of each epoch. We found that using text embeddings generally yields more stable fine-tuning performance.

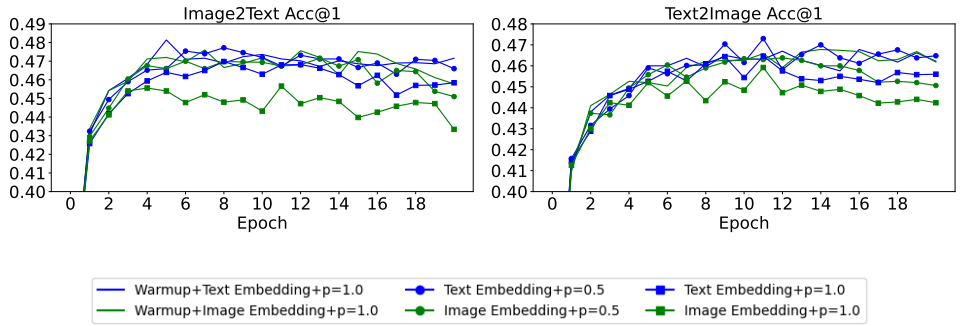

Figure 11: Retrieval accuracies of SimCLIP with various configurations on MSCOCO validation set over the epochs fine-tuned ($k = 16, s = 1$).

B.5 Compositionality Experiment Result Details

We present the details of evaluation on compositionality datasets ARO and SugarCrepe in Figure 12 and 13. We found that SimCLIP generally do not improve the model's ability to understand compositionally of attributes.

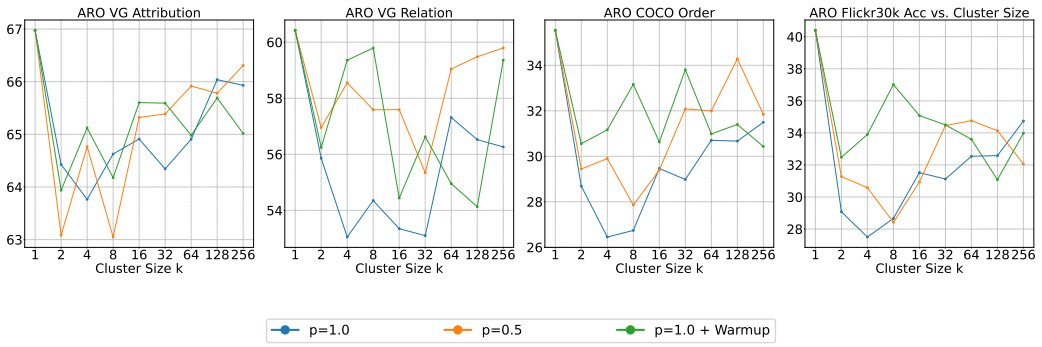

Figure 12: `VG-Attribution`, `VG-Relation`, `COCO Order` and `Flickr30k Order` accuracies of three variants of CLIP on ARO benchmark.

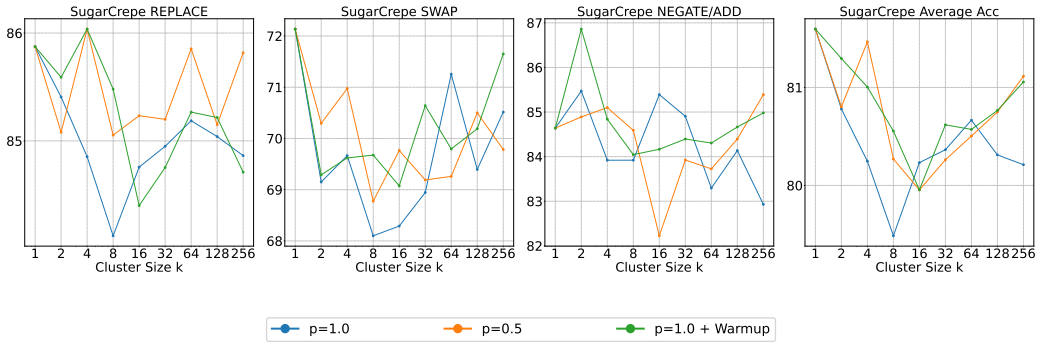

Figure 13: Accuracies of three variants of CLIP on ARO and SugarCrepe benchmark.