A Appendix

A.1 Alternative Positive Generation Methods

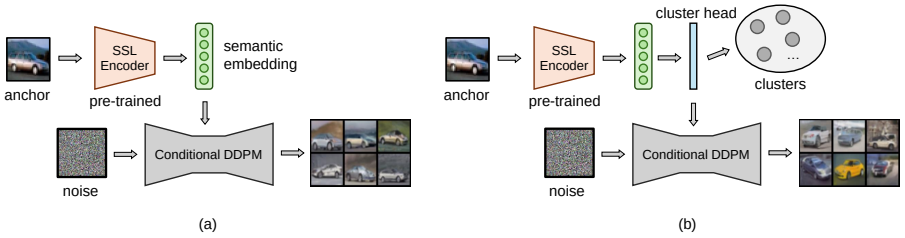


Fig. A.1: Alternative positive generation methods. (a) RCG, using a pre-trained SSL encoder to generate semantic embedding of the anchor image as the condition to guide diffusion sampling. (b) RCG-cluster, using an unsupervised cluster head to cluster the embeddings in RCG and then using the cluster output as the condition.

In this section, we discuss two alternative positive generation methods using the diffusion model and compare them with our proposed CLSP. (1) Similar to the idea in [33], we use an image encoder (pre-trained by any SSL method) to transfer the raw image distribution to a low-dimensional semantic embedding. Subsequently, we train a conditional diffusion model to map a noise distribution to the image distribution conditioned on the semantic embedding. This approach, termed RCG, is illustrated in Fig. A.1(a). Due to the influence of semantic embedding on the diffusion sampling process, the resulting synthetic images typically contain similar semantic content to that of the anchor image and can be used as additional positives. However, one of the key drawbacks of RCG is that the synthetic positives often closely resemble the anchor image, potentially diminishing the benefits of learning these "easy" positives compared to CLSP. As can be seen in Fig. A.2, RCG-generated images exhibit a higher visual similarity to the anchor images, with less variation in semantic content. (2) To mitigate the aforementioned limitation in RCG, we can use a cluster head (e.g., MLP) to cluster the semantic embeddings generated in RCG to distinct clusters and then use these clusters as the conditional to guide diffusion sampling. We name this method RCG-cluster (Fig. A.1(b)). The idea is inspired by the class conditional diffusion model [24], where a pre-trained model can generate diverse images belonging to a specific class given its label. In RCG-cluster, the cluster head is trained using unsupervised learning techniques such as k-means, and the resulting clusters serve as conditions similar to those in the class conditional diffusion model. RCG-cluster has the potential to generate even more diverse positives than CLSP because the generated images are not explicitly correlated with the semantic embedding of the anchor image. However, it may introduce an increased risk of false positives when the anchor image is not mapped into the right cluster (Fig. A.2).

Fig. A.2: Visualization of the synthetic positives generated by different methods. Compared to CLSP, the positives generated by RCG are more similar to the anchor image and thus "easier" for contrastive learning. RCG-cluster brings more variance to the synthetic positives. However, it also introduces more false positives when the anchor image is not mapped to the right cluster.

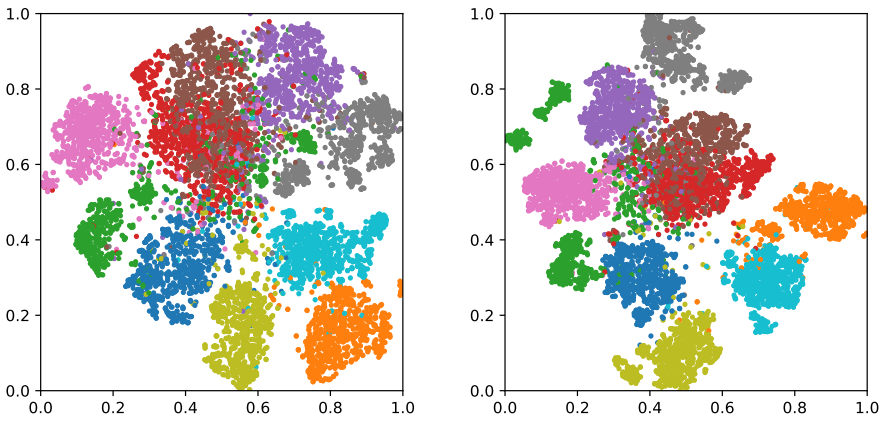
Table A.1: Linear evaluation results of different positive generation methods on CIFAR datasets. All methods are based on the SimCLR framework.

Method	CIFAR10	CIFAR100
RCG	92.41	67.90
RCG-cluster	92.65	66.20
CLSP-SimCLR	94.37	72.01

Table A.1 demonstrates the linear evaluation results of different positive generation methods on CIFAR datasets. For fast evaluation, we set the cluster size in the RCG-cluster to 10 and 100 for CIFAR10 and CIFAR100, respectively. We can see that our CLSP-SimCLR performs the best among these three methods on both CIFAR10 and CIFAR100. RCG-cluster performs better than RCG on CIFAR10 but worse on CIFAR100, the reason could be that CIFAR100 has more classes, which makes the cluster head less accurate and consequently yields more false positives to contrastive learning.

A.2 Implementation Details

Diffusion Model. We pre-train our unconditional diffusion models on CIFAR10, CIFAR100, and STL10 datasets. Following the setting in [23], we use U-Net with an initial filter size of 128 as the backbone. We use a linear β scheduler from $\beta_1 = 1e-4$ to $\beta_T = 0.02$, $T = 1000$. Self-attention is used at the $16\times$ 16 feature map resolution. Dropout ratio is set to 0.1. We train the dif-



(a) SimCLR **(b)** CLSP-SimCLR

Fig. A.3: t-SNE visualization of representations learned by SimCLR and CLSP-SimCLR. Each color represents the representation of a specific class.

fusion model for 2000 epochs with Adam optimizer and a fixed learning rate 0.0002. Only random horizontal flipping is used as the data augmentation. For the STL10 dataset, we resize the image to 64×64 and only use the *unlabeled* split for training. We use DDIM to speed up the diffusion sampling process and the sampling timestamp is set to 200. For CIFAR10, CIFAR100, and STL10 datasets, we use the features from the last layer of the U-Net encoder for feature interpolation. For ImageNet100, we use the features from the last 4 layers of the encoder for feature interpolation.

Self-supervised Pre-training. We use ResNet-18 as the default backbone for all experiments. On CIFAR10 and CIFAR100 datasets, we replace the first 7×7 Conv of stride 2 with 3×3 Conv of stride 1 and remove the maxpooling layer. For STL10, we just replace the first convolutional layer as in CIFAR datasets and keep the maxpooling layer. The batch size is 1024 for SimCLR-based approaches and 512 for MoCo-based approaches on CIFAR10, CIFAR100, and STL10. The batch size is 256 for all methods on ImageNet100. We use SGD as the optimizer with a weight decay of 0.0001 and momentum of 0.9. The initial learning rate is 0.3 with a cosine decay schedule, and linear warmup is used for the first 10 epochs. The temperature τ is 0.2 for all methods. We use standard data augmentations for both anchor images and synthetic images, including random cropping and resizing, random flipping, color distortion, grayscale, and polarization.

Linear Evaluation. Following the similar offline linear evaluation procedure in [7], we freeze the encoder and train a linear classifier using an SGD optimizer without weight decay for 100 epochs. Only random cropping and random flipping are used as data augmentation. The initial learning rate is 10 and reduced to 1.0 and 0.1 at 60 and 80 epochs.

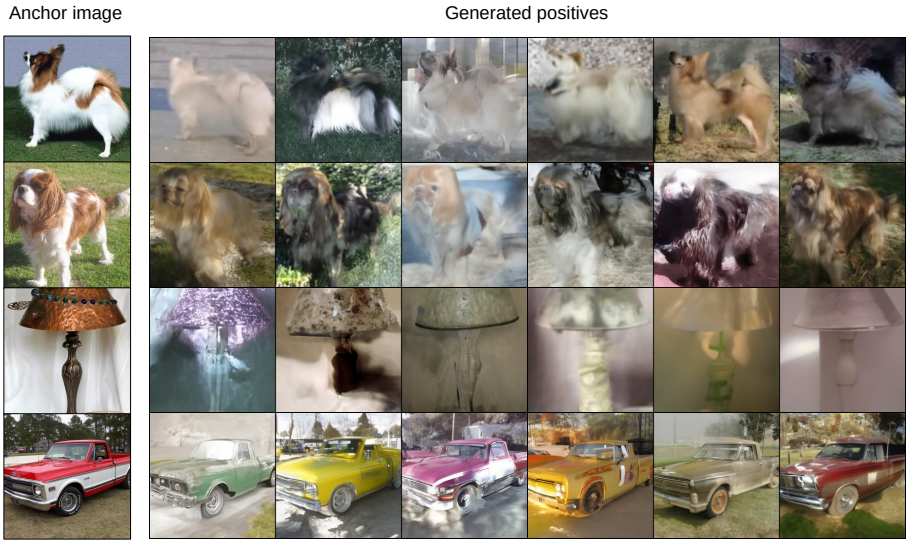


Fig. A.4: Visualization of generated positives using CLSP on ImageNet100 dataset.

Table A.2: Linear evaluation results of different models trained with varied batch sizes on CIFAR10.

Batch size	64	128	256	512	1024
SimCLR	87.61	90.34	91.01	91.32	91.47
MoCoV2	91.89	92.93	92.90	92.94	92.80
CLSP-SimCLR	91.18	93.36	94.04	94.26	94.37
CLSP-MoCoV2	92.55	93.43	94.35	94.41	94.10

A.3 Visualization of Representations

We compare the representations learned by our proposed CLSP-SimCLR and standard SimCLR using t-SNE visualization in Fig. A.3. It can be seen that SimCLR has more overlapped representations among different classes. Such overlapped representations are non-discriminative and offer less information for the downstream tasks. However, the clusters learned by the proposed CLSP-SimCLR are more dense and separable, meaning that the SSL model learns better discriminative features of different classes and provides more information for downstream tasks.

A.4 Impact of Batch Size

To evaluate the influence of batch size on the proposed CLSP, we compared the linear evaluation results of SimCLR, MoCoV2, CLSP-SimCLR, and CLSP-MoCoV2 on CIFAR10 under different batch sizes. The results are shown in Table

A.2. We can see that even though small batch size degrades the linear evaluation results, our CLSP-SimCLR and CLSP-MoCoV2 consistently outperform the baseline SimCLR and MoCoV2 methods under different batch sizes.