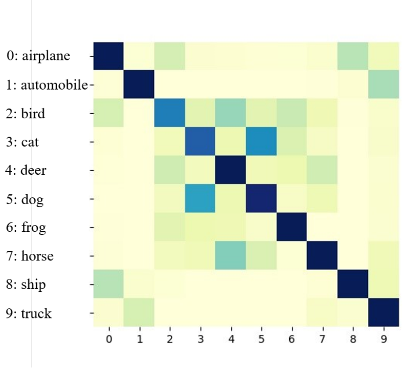


Figure A.3: Cosine similarity heat map of learned features from SimCLR on CIFAR-10 dataset. The darker the color, the larger the similarity.

A TECHNICAL DETAILS

A.1 IMPLICIT BIAS OF SIMCLR ON CIFAR-10.

Figure A.3 plots the cosine similarity heat map of learned features from SimCLR on CIFAR-10 dataset. To calculate the similarity of class A (figures denoted by a_i) to class B (figures denoted by b_i), we first calculate the mean of b_i as $\bar{b}$. Then, we sum up $\sum_i \text{sim}(a_i, \bar{b})$ and plot is with colors. Hence, the similarity matrix shown in Figure A.3 is not symmetric.

A.2 INVESTIGATIONS ON $C(f)$.

Figures A.4 and A.5 illustrate the evolution of different complexity measurements during the training process under the Gaussian mixture setting and the CIFAR-10 respectively.

In the Gaussian mixture setting, the feature extractor is a fully connected ReLU network. Besides $C(f)$, we also evaluate the popular sum of squared weights. The observations on SimCLR are listed as below:

- The expected Lipschitz constant $C(f)$ is small in initialization. It first increases (till around 100 iterations) and then consistently decreases. This empirically supports the implicit bias towards minimizing $C(f)$.
- $C(f)$ and the sum of squared weights share very similar patterns.
- The SNE loss is non-increasing, as if we are doing stochastic neighbor embedding using l_2-distance.

In the CIFAR-10 case, the feature extractor is ResNet-18 plus a fully-connected projection layer. The output from ResNet-18 is usually called representation (512 dimensional) and is utilized for downstream tasks while the projection (128 dimension) is used for training. Besides $C(f)$, we also evaluate the l_2-norm of the representation. The observations for SimCLR and t-SimCLR on CIFAR-10 are summarized as below:

- $C(f)$ for the projection layer shares similar patterns as in the Gaussian mixture case, first increase and then decreases. However, $C(f)$ for the representation layer monotonically decreases.
- $C(f)$ for the projection layer and the l_2-norm in the representation layer share almost identical patterns.
- Comparing SimCLR, both the the calculated $C(f)$ and l_2-norm are much smaller for t-SimCLR.

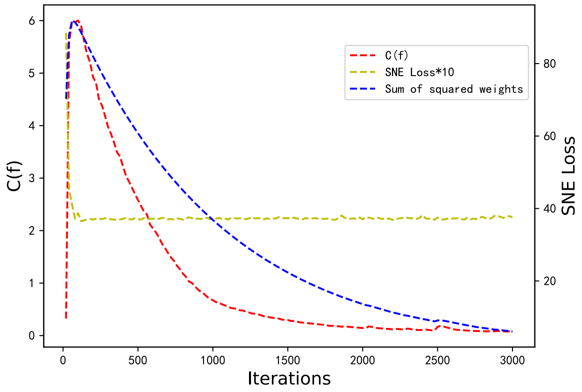


Figure A.4: Empirical evaluation on the complexity of the learned feature mapping during training under the Gaussian mixture setting. Two complexity measurements are considered, i.e., $C(f)$ as in (3.2) and the SNE loss as in (2.2). The SNE loss here only serves as in indicator for how well the pairwise distances are preserved. The training objective is the standard InfoNCE loss. The SNE loss decreases quickly until in the first 100 iterations and then stays flat.

In conclusion, on one hand, our empirical results demonstrate that the complexity of the feature extractor $C(f)$ does decrease during training and seem to be implicitly minimized. On the other hand, its trend is shared with other more popularly used complexity measurements.

A.3 PROOF OF COROLLARY 3.6

In this section, we illustrate with rigor how the hypothesized implicit bias can give rise to structure-preserving property of SSCL. Corollary 3.6 states that minimizing the (Lipschitz) complexity of the feature mapping will also result in the best match between P and Q (under permutation). To provide more theoretical insight, we present the following lemma in the simpler vector-matching case.

Lemma A.1. Let $0<x_1<\cdots<x_m$ and $0<y_1<\cdots<y_m$ be two real-valued sequences, normalized such that $\sum_{i=1}^m x_i^2 = \sum_{i=1}^m y_i^2 = 1$. Consider a permutation π of $\{1,\cdots,m\}$ and denote all such permutations as T. Then

$$\operatorname*{argmin}_{\pi\in T}\sum_{i=1}^{m}\frac{y_{\pi(i)}}{x_i} = \operatorname*{argmin}_{\pi\in T}\sum_{i=1}^{m}(x_i-y_{\pi(i)})^2 := \pi^*,$$

where $\pi^*(i)=i$ for all $i=1,\cdots,m$.

Proof. By the rearrangement inequality, we have

$$\sum_{i=1}^{m}\frac{y_{\pi(i)}}{x_i}\geq\sum_{i=1}^{m}\frac{y_i}{x_i}.$$

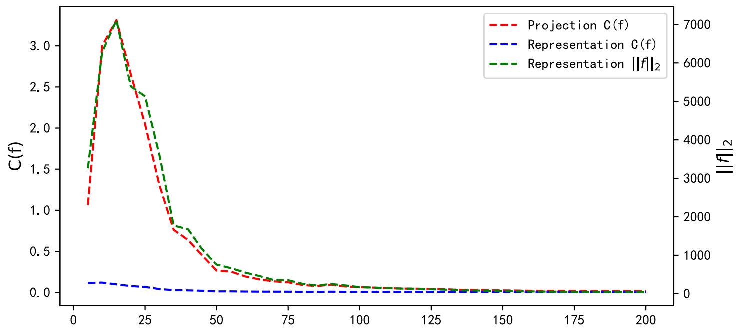
Similarly,

$$\sum_{i=1}^{m}(x_i-y_{\pi(i)})^2=\sum_{i=1}^{m}x_i^2+\sum_{i=1}^{m}y_i^2-2\sum_{i=1}^{m}x_i\cdot y_{\pi(i)}\geq 2-2\sum_{i=1}^{m}x_i\cdot y_i.$$

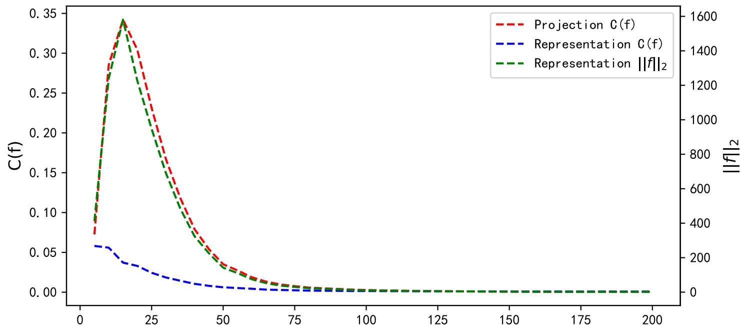
□

Lemma A.1 gives a vector-version illustration of our Corollary 3.6, stating that minimizing the expected derivative (to zero) of the mapping function f, i.e., $\sum_i f(x_i)/x_1$ leads to preserving the norm difference of the input vector and output vector.

Next, we provide the proof of Theorem 3.5.



(a) SimCLR on CIFAR-10.



(b) t-SimCLR on CIFAR-10.

Figure A.5: Empirical evaluation on the complexity of the learned feature mapping during training on CIFAR-10. Two complexity measurements are considered, i.e., $C(f)$ as in (3.2) and l_2-norm. Specifically, we calculate the expected Lipschitz constant on both the representation layer (512-dimensional) and the projection layer (128-dimensional). Figure (a) and (b) show the trends (along the 200 training epochs) for SimCLR and t-SimCLR respectively.

Proof of Theorem 3.5. Straightforwardly, we can write

$$\begin{aligned}\|\bar{P}-Q^{\pi}\|_F &= \sum_{i\neq j}\left(\frac{1}{p_{ij}}+q_{\pi(i)\pi(j)}\right)^2\\ &= \sum_{i\neq j}\frac{1}{p_{ij}^2}+\sum_{i\neq j}{q_{\pi(i)\pi(j)}}^2+2\sum_{i\neq j}\frac{q_{\pi(i)\pi(j)}}{p_{ij}}\\ &= 2C_1(P,Q^{\pi})+\sum_{i\neq j}\frac{1}{p_{ij}^2}+\sum_{i\neq j}{q_{ij}}^2\end{aligned}$$

Thus, minimizing $C_1(P,Q^{\pi})$ also minimizes $\|\bar{P}-Q^{\pi}\|_F$. □

Theorem 3.5 is a straightforward generalization of Lemma A.1. Next, we provide proof for Corollary 3.6, restated below.

Proof of Corollary 3.6. Recall the SimCLR loss $L_{\text{InfoNCE}}=\frac{1}{2n}\sum_{i=1}^{n}(l(\boldsymbol{x}_i,\boldsymbol{x}_i')+l(\boldsymbol{x}_i',\boldsymbol{x}_i))$, where

$$l(\boldsymbol{x}_i,\boldsymbol{x}_i')=-\log\frac{\exp(\text{sim}(f(\boldsymbol{x}_i),f(\boldsymbol{x}_i'))/\tau)}{\sum_{x\in\mathcal{D}_n\cup\mathcal{D}_n'\setminus\{\boldsymbol{x}_i\}}\exp(\text{sim}(f(\boldsymbol{x}_i),f(\boldsymbol{x}))/\tau)}.$$

Without loss of generality, let $\tau = 1$. Notice that $l(\boldsymbol{x}_i, \boldsymbol{x}_i')$ is monotonically decreasing as $\mathrm{sim}(f(\boldsymbol{x}_i), f(\boldsymbol{x}_i'))$ increases, due to the monotonicity of function $\frac{x}{x+c}$ with respect to $x > 0$ for any $c > 0$. Hence, in order for L_{InfoNCE} to be minimized, perfect alignment is required, i.e., $f(\boldsymbol{x}_i) = f(\boldsymbol{x}_i')$ for any $i = 1, \dots, n$.

With perfect alignment achieved, L_{InfoNCE} only concerns the pairwise similarity between negative samples $f(\boldsymbol{x}_i)$'s, which can be simplified as $L_{\text{InfoNCE}} \geq L_{\text{uniform}}$ where

$$\begin{aligned} L_{\text{uniform}} &= \frac{1}{n}\sum_{i=1}^{n} -\log \frac{e}{e + \sum_{j\neq i}\exp(\mathrm{sim}(f(\boldsymbol{x}_i), f(\boldsymbol{x}_j)))} \\ &\geq \log\left(\frac{1}{n}\sum_{i=1}^{n}\left(1 + \frac{1}{e}\sum_{j\neq i}\exp(\mathrm{sim}(f(\boldsymbol{x}_i), f(\boldsymbol{x}_j)))\right)\right) \\ &\geq \log\left(1 + \frac{1}{n\cdot e}\sum_{1\leq i\neq j\leq n}\exp(\mathrm{sim}(f(\boldsymbol{x}_i), f(\boldsymbol{x}_j)))\right). \end{aligned}$$

L_{uniform} can be minimized by mapping $\boldsymbol{x}_i$'s as distant as possible, hence the connection to Tammas problem and the uniformity principle.

With sufficient capacity of the feature mapping f, the SimCLR loss can be minimized to its (empirical) global minima. However, such f is not unique since L_{InfoNCE} is invariant to permutations of mapping relationships from $\boldsymbol{x}_i$ to $f(\boldsymbol{x}_i)$. If f_n^* further minimizes $C(f)$ on the sample level, i.e.,

$$f_n^* := \underset{f}{\mathrm{argmin}} C_n(f) = \underset{f}{\mathrm{argmin}} \sum_{1\leq i\neq j\leq n} \frac{\|f(\boldsymbol{x}_i) - f(\boldsymbol{x}_j)\|_2}{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2},$$

Then, f_n^* also solves a type of SNE problem with uniformity constraint (3.4) as stated in Theorem 3.5. To see this, if we define $q_{ij} = -\|f(\boldsymbol{x}_i) - f(\boldsymbol{x}_j)\|_2$ and $p_{ij} = -\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2$, which is reasonable since the larger the distance, the smaller the similarity, we can directly apply the results in Theorem 3.5.

$\square$

Remark A.2. As can be seen from Theorem 3.5 and the proof of Corollary 3.6, we showcase the relationship between minimizing $C(f)$ and structure preserving property by considering a special SNE problem, where the pairwise similarity is not modeled by Gaussian as standard, hence the word “resembling" in Corollary 3.6. Although $q_{ij} = -\|f(\boldsymbol{x}_i) - f(\boldsymbol{x}_j)\|_2$ is unorthodox, it is reasonable since the larger the distance, the smaller the similarity. If we consider the SNE method as in Hinton et al. (2006), our proof does not go through directly and demands more complicated analysis. However, our results are still valid in connecting the complexity of the feature map to the pairwise similarity matching.

Our statement in Corollary 3.6 requires perfect alignment or perfect uniformity. When the assumptions are not perfectly met, we can still obtain insights for the resulting feature mapping. Alignment and uniformity (Wang & Isola, 2020) is not the whole story of contrastive learning, and our identified structure-preserving property implicitly induced by complexity minimization provides an other angle of the learning process. From this perspective, contrastive learning can be thought of as a combination of alignment and SNE with uniformity constraint. In Figure A.3, while obtaining approximate alignment and uniformity, the feature mapping also preserves the relative relationships of the clusters (labels).

A.4 ALIGNMENT AND UNIFORMITY OF T-SIMCLR

Due to the change of training objective, we may want to reevaluate the properties of the learned feature from t-SimCLR. We will show that alignment still hold while uniformity is changed (to infinity).

Let us consider a compact region $\Omega \subset \mathbb{R}^d$ and $\boldsymbol{x}_i \in \Omega$. Let t be the transformation such that the augmented data point $\boldsymbol{x}_i' = t(\boldsymbol{x}_i)$ is still in Ω. Wang & Isola (2020) showed that the contrastive loss can be decomposed into the alignment loss and the uniformity loss. Zimmermann et al. (2021) further showed that the contrastive loss converges to the cross-entropy between latent distributions, where the underlying latent space is assumed to be uniform, and the positive pairs are specified to be an exponential distribution. In this section, we show a parallel result, which states that in the population level, the t-SNE loss is the cross-entropy between two distributions of generating positive pairs.

Theorem A.3. Let $H(\cdot,\cdot)$ be the cross entropy between distributions. Let $p(\boldsymbol{x})$ be the density of $\boldsymbol{x}$, $p(\cdot|\boldsymbol{x})$ be the conditional density of generating a positive pair, and define

$$q_f(\boldsymbol{x}'|\boldsymbol{x}) = C_f(\boldsymbol{x})^{-1}\frac{p(\boldsymbol{x}')}{1+\|f(\boldsymbol{x})-f(\boldsymbol{x}')\|_2^2}, \text{ with } C_f(\boldsymbol{x}) = \int_\Omega \frac{p(\boldsymbol{x}')}{1+\|f(\boldsymbol{x})-f(\boldsymbol{x}')\|_2^2}\mathrm{d}\boldsymbol{x}'.$$

Then, we have

$$\mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x})}(H(p(\cdot|\boldsymbol{x}),q_f(\cdot|\boldsymbol{x})) = L_a(f)+L_u(f), \tag{A.1}$$

which corresponds to the population-level t-SimCLR loss where

$$\begin{aligned} L_a &= \mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x})}\mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x}'|\boldsymbol{x})}\log(1+\|f(\boldsymbol{x})-f(\boldsymbol{x}')\|_2^2),\\ L_u &= \mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x})}\log\mathbb{E}_{\widetilde{\boldsymbol{x}}\sim p(\widetilde{\boldsymbol{x}})}(1+\|f(\boldsymbol{x})-f(\widetilde{\boldsymbol{x}})\|_2^2)^{-1}. \end{aligned}$$

Proof. Note that

$$\begin{aligned} &H(p(\cdot|\boldsymbol{x}),q_f(\cdot|\boldsymbol{x}))\\ =&-\int_\Omega p(\boldsymbol{x}'|\boldsymbol{x})\log\left(\frac{p(\boldsymbol{x}')}{1+\|f(\boldsymbol{x})-f(\boldsymbol{x}')\|_2^2}\right)\mathrm{d}\boldsymbol{x}'+\log C_f(\boldsymbol{x})\\ =&\int_\Omega p(\boldsymbol{x}'|\boldsymbol{x})\log(1+\|f(\boldsymbol{x})-f(\boldsymbol{x}')\|_2^2)\mathrm{d}\boldsymbol{x}'-\int_\Omega p(\boldsymbol{x}'|\boldsymbol{x})\log(p(\boldsymbol{x}'))\mathrm{d}\boldsymbol{x}'+\log\int_\Omega\frac{p(\boldsymbol{x}')}{1+\|f(\boldsymbol{x})-f(\boldsymbol{x}')\|_2^2}\mathrm{d}\boldsymbol{x}'\\ =&\int_\Omega p(\boldsymbol{x}'|\boldsymbol{x})\log(1+\|f(\boldsymbol{x})-f(\boldsymbol{x}')\|_2^2)\mathrm{d}\boldsymbol{x}'-\int_\Omega p(\boldsymbol{x}'|\boldsymbol{x})\log(p(\boldsymbol{x}'))\mathrm{d}\boldsymbol{x}'+\log\mathbb{E}_{\boldsymbol{x}'\sim p(\boldsymbol{x}')}(1+\|f(\boldsymbol{x})-f(\boldsymbol{x}')\|_2^2)^{-1}. \end{aligned}$$

Taking expectation with respect to $\boldsymbol{x}$ leads to

$$\begin{aligned} &\mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x})}H(p(\cdot|\boldsymbol{x}),q_f(\cdot|\boldsymbol{x}))\\ =&\mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x})}\mathbb{E}_{\boldsymbol{x}'\sim p(\boldsymbol{x}'|\boldsymbol{x})}\log(1+\|f(\boldsymbol{x})-f(\boldsymbol{x}')\|_2^2)+\mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x})}\log\mathbb{E}_{\widetilde{\boldsymbol{x}}\sim p(\widetilde{\boldsymbol{x}})}(1+\|f(\boldsymbol{x})-f(\widetilde{\boldsymbol{x}})\|_2^2)^{-1}\\ &-\int_\Omega\int_\Omega p(\boldsymbol{x})p(\boldsymbol{x}'|\boldsymbol{x})\log(p(\boldsymbol{x}'))\mathrm{d}\boldsymbol{x}'\mathrm{d}\boldsymbol{x}\\ =&L_a(f)+L_u(f)-C_p, \end{aligned}$$

where

$$C_p = \int_\Omega\int_\Omega p(\boldsymbol{x})p(\boldsymbol{x}'|\boldsymbol{x})\log(p(\boldsymbol{x}'))\mathrm{d}\boldsymbol{x}'\mathrm{d}\boldsymbol{x} = \int_\Omega\int_\Omega p(\boldsymbol{x},\boldsymbol{x}')\log(p(\boldsymbol{x}'))\mathrm{d}\boldsymbol{x}'\mathrm{d}\boldsymbol{x}$$

does not depend on f.

$$\begin{aligned} &\mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x})}H(p(\cdot|\boldsymbol{x}),q_f(\cdot|\boldsymbol{x}))\\ =&\int_\Omega p(\boldsymbol{x})\frac{1}{p(\boldsymbol{x})}\int_\Omega p(\boldsymbol{x},\boldsymbol{x}')\log\left(\frac{p(\boldsymbol{x}')}{1+\|f(\boldsymbol{x})-f(\boldsymbol{x}')\|_2^2}\right)\mathrm{d}\boldsymbol{x}'\mathrm{d}\boldsymbol{x}\\ &-\int_\Omega\int_\Omega\frac{p(\boldsymbol{x})p(\boldsymbol{x}')}{1+\|f(\boldsymbol{x})-f(\boldsymbol{x}')\|_2^2}\mathrm{d}\boldsymbol{x}\mathrm{d}\boldsymbol{x}'\\ =&\int_\Omega\int_\Omega p(\boldsymbol{x},\boldsymbol{x}')\log\left(\frac{p(\boldsymbol{x}')}{1+\|f(\boldsymbol{x})-f(\boldsymbol{x}')\|_2^2}\right)\mathrm{d}\boldsymbol{x}'\mathrm{d}\boldsymbol{x}\\ &-\int_\Omega\int_\Omega\frac{p(\boldsymbol{x})p(\boldsymbol{x}')}{1+\|f(\boldsymbol{x})-f(\boldsymbol{x}')\|_2^2}\mathrm{d}\boldsymbol{x}\mathrm{d}\boldsymbol{x}'. \end{aligned}$$

□

In Theorem A.3, L_a is the alignment loss and L_u is the uniformity loss. The decomposition is much more natural for t-SimCLR as opposed to that in L_{InfoNCE}, mainly due to the change from conditional to joint distribution when modeling the pairwise similarity. Furthermore, if the t-SimCLR loss is minimized, we must have $p(\cdot|\boldsymbol{x}) = q_f(\cdot|\boldsymbol{x})$, provided f has sufficient capacity. Note that if $p(\cdot|\boldsymbol{x}) = q_f(\cdot|\boldsymbol{x})$, then $P_{j|i}$ and $Q_{j|i}$ are perfectly matched, which indicates that we obtain a perfect neighbor embedding.

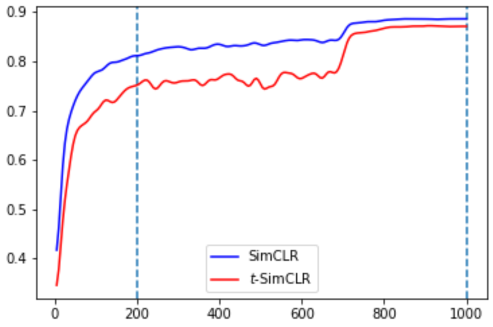


Figure B.6: Nearest neighbor test accuracy vs. training epochs. SimCLR and t-SimCLR share similar trends and convergence speed.

Theorem A.3 implies that the optimal feature mapping f^* satisfies

$$p(\cdot|\boldsymbol{x}) = q_{f^*}(\cdot|\boldsymbol{x}),$$

which further implies that for any $\boldsymbol{x} \in \Omega$,

$$\begin{aligned} & C_{f^*}(\boldsymbol{x})^{-1} \frac{p(\boldsymbol{x}')}{1+\|f^*(\boldsymbol{x}) - f^*(\boldsymbol{x}')\|_2^2} \propto C(\boldsymbol{x})^{-1} p(\boldsymbol{x}'|\boldsymbol{x}) \\ \Leftrightarrow\ & C_{f^*}(\boldsymbol{x})^{-1} \frac{1}{1+\|f^*(\boldsymbol{x}) - f^*(\boldsymbol{x}')\|_2^2} \propto C(\boldsymbol{x})^{-1} \frac{p(\boldsymbol{x},\boldsymbol{x}')}{p(\boldsymbol{x})p(\boldsymbol{x}')}, \end{aligned} \tag{A.2}$$

where $C(\boldsymbol{x}) = \int p(\boldsymbol{x}'|\boldsymbol{x}) \mathrm{d}\boldsymbol{x}'$. Unlike the usual normalized SimCLR, t-SNE does not assume any special structure on f (e.g., $\|f\|_2 = 1$), thus f can go to infinity. Comparing to the finite sample t-SimCLR loss, the population version is trickier to analyze. This is because for a given point $\boldsymbol{x}'$, it can be an augmented sample of some $\boldsymbol{x}$ (with probability $p(\boldsymbol{x}'|\boldsymbol{x})$), or a negative sample of $\boldsymbol{x}$ (when we treat $\boldsymbol{x}'$ as another sample point). This reflects the essential difficulty between population and finite samples in contrastive learning, not only for t-SimCLR.

For clustered data, (A.2) provides two important messages, provided that the augmentation is not too extreme and the augmented sample $\boldsymbol{x}'$ stays in the same cluster as the original $\boldsymbol{x}$. On one hand, when $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ belongs to different clusters, the joint density $p(\boldsymbol{x} = \boldsymbol{x}_1, \boldsymbol{x}' = \boldsymbol{x}_2)$ will be very small, close to zero, which indicates that $\|f^*(\boldsymbol{x}_1) - f^*(\boldsymbol{x}_2)\|_2$ is very large, tending to infinity. On the other hand, for $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ belonging to the same cluster, $p(\boldsymbol{x} = \boldsymbol{x}_1, \boldsymbol{x}' = \boldsymbol{x}_2)$ will be relatively large. Hence, the features of the same cluster will stay close. Overall, we will observe similar clustered structure in the feature space. This is confirmed in the Gaussian mixture setting in Figure 1(c), in which case, the problem can be oversimplified as mapping 5 points in $\mathbb{R}^2$ to the unit-circle.

B EXPERIMENT DETAILS

B.1 CIFAR-10 SETTINGS

CIFAR-10 (Krizhevsky, 2009) is a colorful image dataset with 50000 training samples and 10000 test samples from 10 categories. We use ResNet-18 (He et al., 2016) as the feature extractor, and the other settings such as projection head all follow the original settings of SimCLR (Chen et al., 2020a). To evaluate the quality of the features, we follow the KNN evaluation protocol (Wu et al., 2018). which computes the cosine similarities in the embedding space between the test image and its nearest neighbors, and make the prediction via weighted voting. We train each model with batch size of 256 and 200 epochs for quicker evaluation. For t-SimCLR, without specifying otherwise, we grid search the t_{df} and τ with range $\{1, 2, 5, 10\}$ and $\{1, 2, 5, 10\}$ respectively.

Ablation of training epochs We also run the SimCLR and t-SimCLR experiments in the more standard 1000 epochs setting. For SimCLR, we use batch size of 512, learning rate of 0.3, temperature

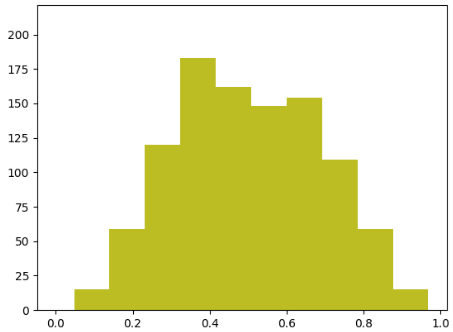


Figure B.7: The histogram of IoUs for 1000 constructed positive pairs in CIFAR-10. The empirical distribution is almost symmetric around 0.5.

of 0.7, and weight dacay of 0.0001. For t-SimCLR, we use batch size of 512, learning rate of 0.8, temperature of 10, weight dacay of 0.0002, and $t_{df} = 5$. The nearest neighbor accuracy for SimCLR is 87.2% vs. that for t-SimCLR is 88.8%.

B.2 IMAGE AUGMENTATION

When processing images, several popular augmentations are usually adopted (following the setting in SimCLR Chen et al. (2020a)), e.g., random resized crop (crops a random portion of image and resize it to the original size), horizontal flip, color jitter (randomly change the brightness, contrast, saturation and hue of an image). To illustrate the natural weighting scheme in Section 4.1, we considered random resized crop and specifies the weights by the IoU (intersection over union) of the positive pair. In particular, two augmented images are created from an anchor image. Each augmentation crops a rectangular region of the image, denoted by r_1,r_2 respectively, and their IoU is defined by the area of intersection $r_1 \cap r_2$ divided by the area of the union $r_1 \cup r_2$. The IoU is always between 0 and 1. In our experiment, we chose the default settings and Figure B.7 illustrates the IoU histogram of 1000 constructed positive pairs.

B.3 DEGREE OF FREEDOM IN t-SIMCLR

Feature dimension efficiency in OOD case. To further investigate the generalization ability of SSCL methods, we devise a challenging setting where the model is trained on CIFAR-10 and tested on CIFAR-100 classification. In this case, we evaluate the effect of increasing feature dimensions in the projection layer, as an extension on the CIFAR-10 in-distribution case. The results are shown in Figure B.8, where there are two things to note:

- The gain of extra dimensions in the OOD case does vanish later than that in the in-distribution case.
- The advantage of SimCLR vs. t-SimCLR is very significant with around 10% improvement when $d = 128$ using nearest neighbor[5] classification, indicating that t-SimCLR produces better separated clusters.

Relationship between t_{df} and d_z. The larger the degree of freedom t_{df}, the less heavy-tail the t-distribution. As d_z decreases, the crowding problem becomes more severe and as recommended by (Van der Maaten & Hinton, 2008), a smaller t_{df} tends to work better. We evaluate the sensitivity of t_{df} (1, 5, 10) under different choices of d_z (1, 2, 4, 8, 16, 32, 64, 128) in CIFAR-10 and the results are reported in Figure B.9. As can be seen, when d_z is small (1,2,4,8), $t_{df} = 1$ outperforms. Comparing

---

[5] When evaluating by training linear classifiers for 100 epochs, the accuracy for SimCLR is 46.4% and that for t-SimCLR is 48.14% (averaged over 3 replications).

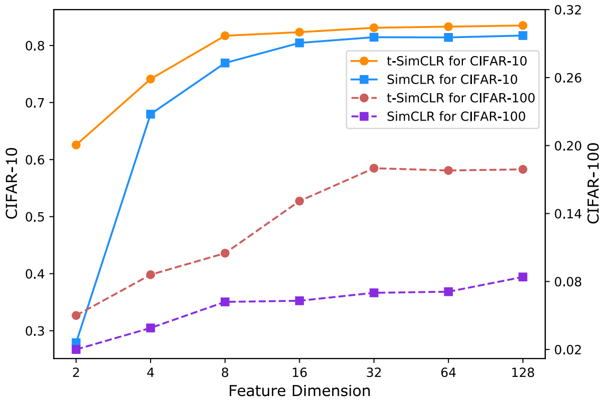


Figure B.8: Extension on Figure 2(b). Nearest neighbor classification accuracy for SimCLR vs. t-SimCLR on both CIFAR-10 (in-distribution) and CIFAR-100 (out-of-distribution) using different feature dimensions.

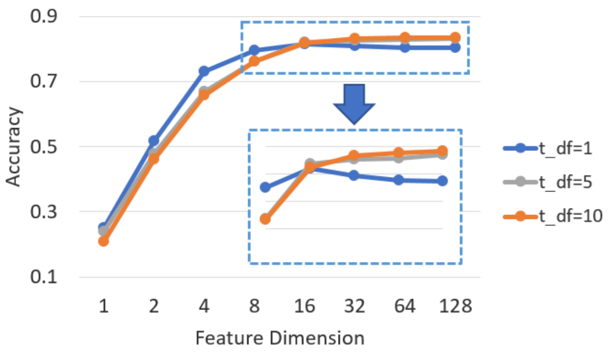


Figure B.9: Nearest neighbor classification accuracy on CIFAR-10 for t-SimCLR using different feature dimensions and different degrees of freedom (t_df).

$t_{df} = 5$ and $t_{df} = 10$, the two perform similarly when d_z is large (16,32,64,128) but the smaller $t_{df} = 5$ yields better accuracy when $d_z = 1,2,4$.

Tuning temperature vs. tuning t_{df}. As illustrated in Section 4.2, when the feature space dimension is low, the heavy-tailed t-distribution is a better choice than Gaussian to alleviate the crowding problem. Even though tuning the temperature of L_{InfoNCE}, i.e., making τ larger, can also have the effect of making the distribution less concentrated (τ can be seen as the standard deviation), tuning temperature and tuning t_{df} are fundamentally different. The former is controlling how fast does the similarity $Q_{i,j}$ decays as the distance between $\boldsymbol{z}_i$ and $\boldsymbol{z}_j$ increases, while the latter serves as a scaling factor, offering constant level modification of the scheme. In our experiments with SimCLR vs t-SimCLR on CIFAR-10, temperature is tuned as a hyperparameter. The difference in τ can never make up to the difference between the baseline SimCLR and t-SimCLR. We found $\tau = 0.5$ to work better for the base SimCLR while larger τ works better with our t-SimCLR. We recommend $\tau = 5$ as the default choice.

B.4 IMAGENET PRE-TRAINING

To show the ability for large scale domain transfer and OOD generalization, we conduct experiments on ImageNet pre-training based on MoCo-v2 with its official implementation[6]. We follow most of their settings, e.g, data augmentation, 200 epochs pre-training, and optimization strategy, etc. The loss

---

[6] https://github.com/facebookresearch/moco

Table B.3: Domain transfer results of vanilla MoCo-v2 and t-MoCo-v2.

Method	Aircraft	Birdsnap	Caltech101	Cars	CIFAR10	CIFAR100	DTD	Pets	SUN397	Avg.
MoCo-v2 (800 epochs)	**83.80**	45.51	83.01	**86.18**	**96.42**	71.69	**71.70**	**89.11**	55.61	75.89
MoCo-v2 (200 epochs)	82.75	44.53	83.31	85.24	95.81	72.75	71.22	86.70	56.05	75.37
t-MoCo-v2 (200 epochs)	82.78	**53.46**	**86.81**	86.17	96.04	**78.32**	69.20	87.95	**59.30**	**77.78**

Table B.4: OOD accuracies of vanilla MoCo-v2 and t-MoCo-v2 on domain generalization benchmarks.

Method	PACS	VLCS	Office-Home	Avg.
MoCo-v2 (800 epochs)	58.9	69.8	41.6	56.8
MoCo-v2 (200 epochs)	58.5	70.4	36.6	55.2
t-MoCo-v2 (200 epochs)	**61.3**	**75.1**	**42.1**	**59.5**

is modified according to Section 4.2 and batch normalization is applied along every dimension. We grid search the t_{df} and τ with range $\{2, 5, 10, 15\}$ and $\{0.2, 2, 5, 10\}$ respectively. Finally we choose $t_{df} = 10$ and $\tau = 5$ to be the optimal hyperparameters. We use this pre-train model as initialization for domain transfer and OOD experiments.

B.5 DOMAIN TRANSFER

We compare MoCo-v2 pre-trained with 800 / 200 epochs and t-MoCo-v2 on Aircraft, Birdsnap, Caltech101, Cars, CIFAR10, CIFAR100, DTD, Pets, and SUN397 in Table B.3. We follow the transfer settings in Ericsson et al. (2021) to finetune the pre-trained models. For datasets Birdsnap, Cars, CIFAR10, CIFAR100, DTD, and SUN397, we report the top-1 accuracy metric, while for Aircraft, Caltech101, and Pets, we report the mean per-class accuracy metric. We also follow Ericsson et al. (2021) to split each dataset into training, validation, and test sets. On each dataset, we perform a hyperparameter search as follows. (1) We choose the initial learning rate according to a grid of 4 logarithmically spaced values between 1×10^{-4} and 1×10^{-1}; (2) We choose the weight decay parameter according to a grid of 4 logarithmically spaced values between 1×10^{-6} and 1×10^{-3}, plus no weight decay; (3) The weight decay values are divided by the learning rate; (4) For each pair of learning rate and weight decay, we finetune the pre-trained model for 5000 steps by SGD with Nesterov momentum 0.9, batch size of 64, and cosine annealing learning rate schedule without restarts. As can be seen in Table B.3, our t-MoCo-v2 with 200 epochs even outperform the baseline with 800 epochs on average.

B.6 OOD GENERALIZATION

To demonstrate the advantage of our modification, we also compare MoCo-v2 pre-trained with 800 / 200 epochs and t-MoCo-v2 on OOD generalization benchmarks: PACS Li et al. (2017), VLCS Fang et al. (2013), Office-Home Venkateswara et al. (2017). We follow the standard way to conduct the experiments, i.e., choosing one domain as the test domain and using the remaining domains as training domains, which is named the leave-one-domain-out protocol. The top linear classifier is trained on the training domains and tested on the test domain. Each domain rotates as the test domain and the average accuracy is reported for each dataset in Table B.4. On each dataset, we perform a hyperparameter search following DomainBed Gulrajani & Lopez-Paz (2021). We adopt the leave-one-domain-out cross-validation setup in DomainBed with 10 experiments for hyperparameter selection and run 3 trials. As can be seen in Table B.4, our t-MoCo-v2 with 200 epochs even significantly outperform the baseline with 800 epochs for all of the three datasets.

B.7 SSCL INSPIRED DATA VISUALIZATION

t-SNE (Van der Maaten & Hinton, 2008) and its variants are designed for data visualization. However, for more complicated data, such as colored images, the results are not satisfactory. Using standard t-SNE, the 2D visualization of the 50K training images of CIFAR-10 (labels denoted as 0, 1,...,9) can be seen in Figure B.10, where different labels are hardly separated. The poor performance of t-SNE on CIFAR-10 can be traced back to the poor distance choice on images, i.e., l_2-norm. Inspired by

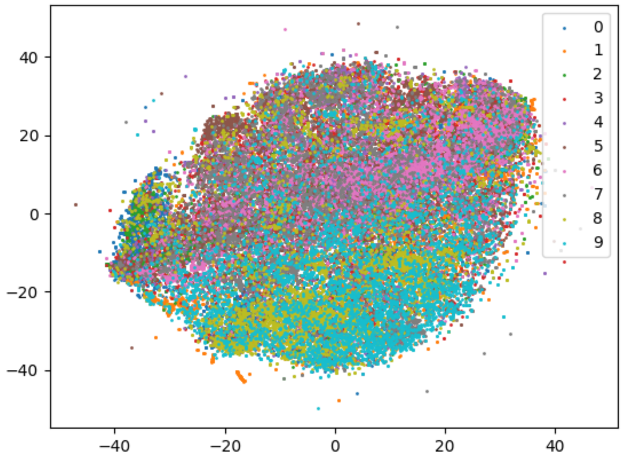


Figure B.10: 50K CIFAR-10 training images visualization in 2D with t-**SNE**.

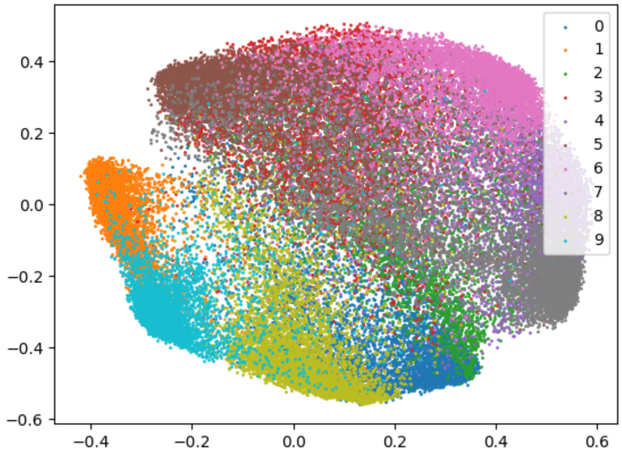


Figure B.11: 50K CIFAR-10 training images visualization in 2D with the default t-**SimCLR**.

the success of SSCL for natural images, t-SNE can potentially be improved by incorporating data augmentations.

In light of our perspective (S1), t-SNE can take advantage of the distance specified with (3.1) and the resulting model is essentially our t-SimCLR with feature dimension 2. The visualization from t-SimCLR is shown in Figure B.11, which is much more separated (the nearest neighbor classification accuracy on CIFAR-10 test data is 56.6%). By choosing the feature dimension to be 2, various SSCL methods can also be made into data visualizing tools. In Figure B.12, we visualize the outcome from SimCLR (the nearest neighbor classification accuracy on CIFAR-10 test data is 24.8%).

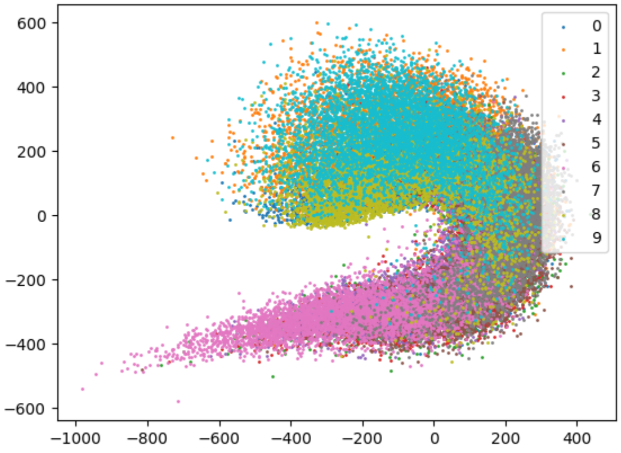


Figure B.12: 50K CIFAR-10 training images visualization in 2D with the **SimCLR**.