EECS 442-Effect of Contrastive Loss on Unsupervised Representation Learning

Yixuan Wang, Haomeng Zhang, Qingyi Chen, Shuangrui Ding
{yixuanwa,haomeng,chenqy,markding}@umich.edu

1. Introduction

The unsupervised representation learning methods can learn the high-level representations that perform well on the downstream tasks without labels. As we learnt in the course EECS 442, the models based on the contrastive loss significantly outperform regular generative models. The contrastive loss helps the model pull together the similar instances and push apart the dissimilar instances in the embedding space. Then the question arises: what should a good contrastive loss look like?

In this project, we are proposing to survey several designs of contrastive loss and analyze the effects of different contrastive losses on the criteria of *Alignment* and *Uniformity* [16]. We empirically demonstrated that the choice of contrastive loss would significantly influence the learning performance. [1]

2. Related work

Various papers [7, 12, 15, 17, 9, 2, 1, 5] have arisen on unsupervised contrastive representation learning. Instance discrimination [17] is formulated as a non-parametric classification problem and tackles the computational challenge by memory bank and noise-contrastive estimation. However, the trick of memory bank also results in inconsistency of the samples. To better improve the mechanism of memory bank, MoCo [7] proposes a momentum contrast mechanism for constructing a dynamic dictionaries for contrastive learning. SimCLR [15] simplifies the contrastive learning framework and introduces nonlinear transformation (MLP) between the representation and the contrastive loss. Besides, SwAV [1] proposes an online clustering-based method without the memory bank and momentum encoder. In the paper, we focus on the CMC framework [15], which considers an image in Lab color space to be a paired example of the co-occurrence of two views, the L view and the ab view.

The aforementioned frameworks all depend on the contrastive loss [6, 13, 12, 17, 14] to train the model. Basically, the design of the contrastive loss is inspired by the field of metric learning. In the paper, we survey several designs of contrastive loss, which will be elaborated in the following section.

[1]The code is available at here.

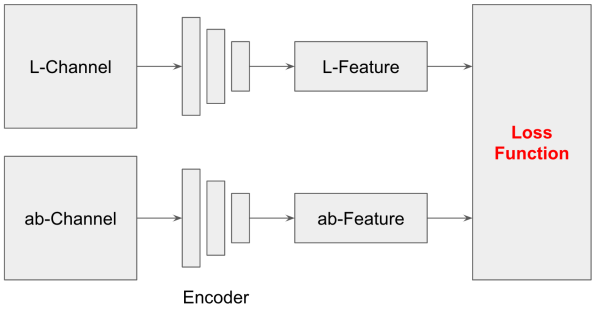


Figure 1: CMC framework used in our paper. We survey the effect of the loss function by compare and contrast the downstream performance, and the metric of alignment and uniformity.

3. Method

In the section, we first introduce our Contrastive Multiview Coding (CMC) model[2] in detail and then formulate several types of the contrastive loss. Finally, we use the criteria of *Alignment* and *Uniformity* [16] to evaluate how well a certain type of contrastive loss performs.

3.1. Contrastive Multiview Coding [15]

We learn a vector representation for images: in this representation, two corrupted versions of the same image should share great similarity, while the similarity of the corrupted versions from two different images should be fairly small. Shown in Figure 1, CMC applies the luminance (i.e. grayscale intensity) and chromaticity (i.e. color) in the *Lab* color space as *views* (*L*-channel and *ab*-channel) of an image. A good representation should be able to create similar vectors for these two views and should contain the information shared between two views. Take softmax loss as example,

[2]The elaboration references EECS442 ps8 by Prof. Andrew Owens and course staff.

$$\mathcal{L}_{\text{contrast}}^{V_1,V_2} = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{h_\theta\left(v_1^i, v_2^1\right)}{\sum_{j=1}^{k+1} h_\theta\left(v_1^i, v_2^j\right)}$$

where v_1 and v_2 are two different views of the data, and k is the number of negative samples: the function h_θ measures the similarity between the representations of the two views, and is implemented using a neural network encoder:

$$h_\theta\left(v_1, v_2\right) = \exp\left(\frac{f_{\theta_1}\left(v_1\right)\cdot f_{\theta_2}\left(v_2\right)}{\left\|f_{\theta_1}\left(v_1\right)\right\|\cdot\left\|f_{\theta_2}\left(v_2\right)\right\|}\cdot\frac{1}{\tau}\right)$$

and f_{θ_1} and f_{θ_2} are encoders for extracting representations from view 1 and view 2, respectively. The constant τ is the temperature hyperparameter for controlling the range of the numbers that are exponentiated.

We will minimize a symmetric objective function that sums $\mathcal{L}_{\text{contrast}}^{V_1,V_2}$ and $\mathcal{L}_{\text{contrast}}^{V_2,V_1}$, i.e.,

$$\mathcal{L}\left(V_1, V_2\right) = \mathcal{L}_{\text{contrast}}^{V_1,V_2} + \mathcal{L}_{\text{contrast}}^{V_2,V_1}$$

3.2. Types of Contrastive Loss

In this paper, we inspect four types of contrastive loss in the condition of CMC model. For simplification, we unify the notation as follow. We assume the representation set $X = \{x_i \in \mathbb{R}^D\}_{i=0}^N$ where N is the number of data in set X and D is the representation dimension. All x_i is normalized. $x \cdot y$ means the inner product operation of vectors $x, y \in \mathbb{R}^D$. We denote the distance function d as the euclidean distance between vectors, $d\left(x_i, x_j\right) = \left\|x_i - x_j\right\|_2$.

Spring-like Loss [6] We formulate $Y(x_i, x_j) = 1$ if (x_i, x_j) is a positive pair, otherwise $Y(x_i, x_j) = 0$. The spring-like loss is

$$\mathcal{L}(x_i, x_j) = \begin{cases} \frac{1}{2}d\left(x_i, x_j\right), & \text{if } Y(x_i, x_j) = 1, \\ \frac{1}{2}\left[m - d(x_i, x_j)\right]_+, & \text{if } Y(x_i, x_j) = 0, \end{cases}$$

where $[\cdot]_+ = \max(0, \cdot)$ and m is a hyperparameter.

Triplet Loss [13] For an anchor x, we want to ensure that it is closer to the positive data x^p than it is to the negative data x^n. The triplet loss is

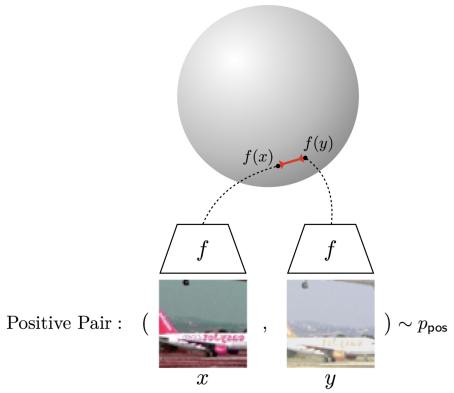
$$\mathcal{L}(x, x_p, x_n) = \left[d\left(x^p, x\right) - d\left(x^n, x\right) + \alpha\right]_+,$$

where α is a margin constant and $[\cdot]_+ = \max(0, \cdot)$.

$(N+1)$-way Softmax [12] The $(N+1)$-way softmax samples a set X which has one positive sample x_1 and $N-1$ negative samples $\{x_i\}_{i=2}^N$. The $(N+1)$-way softmax loss is

$$\mathcal{L}_N(x) = -\log\frac{\exp(x_1 \cdot x/\tau)}{\sum_{i=1}^{N}\exp(x_i \cdot x/\tau)},$$

where τ is a hyper-parameter.



(a) **Alignment**: Similar samples have similar features.

(b) **Uniformity**: Preserve maximal information

Figure 2: Illustration of alignment and uniformity of feature distributions on the output unit hypershpere. [16]

InfoNCE [17] We can adapt noise-contrastive estimation (NCE) to approximate the full softmax. The probability that representation x corresponds to the i-th example is:

$$P(i \mid x) = \frac{\exp(x_i \cdot x/\tau)}{Z}, \quad Z = \sum_{i=1}^{N}\exp(x_i \cdot x/\tau)$$

where Z is the normalizing constant. We formalize the noise distribution as a uniform distribution $P_n = 1/n$ and assume that noise samples are m times more frequent than data samples. The posterior probability of sample i with feature x being from the data distribution (denoted by $D = 1$) is:

$$h(i, x) = P(D = 1 \mid i, x) = \frac{P(i \mid x)}{P(i \mid x) + mP_n(i)}$$

where m is a hyperparamter. Our approximated training objective is to minimize the negative log-posterior distribution of data and noise samples,

$$\mathcal{L}_{NCE} = -\underset{x\sim P_d}{E}[\log h(i, x)] - m\underset{x'\sim P_n}{E}\left[\log\left(1 - h\left(i, x'\right)\right)\right]$$

where P_d denotes the actual data distribution.

Layer (type)	Output Shape	Param #
Conv2d-1	[-1, 12, 32, 32]	588
ReLU-2	[-1, 12, 32, 32]	0
Conv2d-3	[-1, 24, 16, 16]	4,632
ReLU-4	[-1, 24, 16, 16]	0
Conv2d-5	[-1, 48, 8, 8]	18,480
ReLU-6	[-1, 48, 8, 8]	0
Conv2d-7	[-1, 24, 4, 4]	18,456
ReLU-8	[-1, 24, 4, 4]	0

Figure 3: Encoder architecture

3.3. Alignment and Uniformity [16]

Besides the model downstream performance, we are going to investigate the effect of aforementioned loss through Alignment and Uniformity [16]. We quantify two properties alignment and uniformity (see in Figure 2) as below.

Alignment The alignment loss is defined with the expected distance with positive pairs,

$$\mathcal{L}_{\text{align}}(\alpha) \triangleq \underset{(x,y)\sim p_{\text{pos}}}{\mathbb{E}} \left[\|x-y\|_2^{\alpha}\right], \quad \alpha > 0.$$

Minimizing alignment loss helps the model pull the positive pairs closer.

Uniformity We want the uniformity metric to be asymptotically correct. Thus, we consider Gaussian potential kernel,

$$G_t(x,y) \triangleq e^{-t\|x-y\|_2^2}, \quad t > 0.$$

We define the uniformity loss as the logarithm of the average pairwise Gaussian potential:

$$\mathcal{L}_{\text{uniform}}(t) \triangleq \log \underset{\substack{\text{i.i.d} \\ x,y\sim p_{\text{data}}}}{\mathbb{E}} \left[G_t(x,y)\right]$$

Minimizing uniformity loss helps the model preserve maximal information.

4. Experiment

In this section, we empirically verify the effect of the contrastive on unsupervised representation learning.

4.1. Setup

Model We simplify the encoder in CMC as five 4×4 convolution layers followed by ReLU layer shown in Figure 3. We apply Adam optimizer [10] setting learning rate $1e$-3, betas $(0.5, 0.999)$. We train the model for 100 epochs with different contrastive losses.

To measure the downstream classification task, we fix the parameter of CMC model we have trained in the manner of unsupervised constrastive learning and concatenate two channels' features. We feed the concatenation into the linear readout head and train it with label information.

STL-10 Dataset [3] STL-10 is an image recognition dataset designed for developing unsupervised learning algorithms. It contains 500 training images (10 pre-defined folds), 800 test images per class and 100000 unlabeled images for unsupervised learning. The format of image is 96×96 RGB image. Noted that these unlabeled examples are extracted from a similar but broader distribution of images. In our experiment, due to GPU resource limit, we set batch size 128 in training/test set.

4.2. Experimental Result

We evaluate the loss in two aspects. Firstly, we compare the linear classifier accuracy among models trained by four kinds of contrastive loss. We find the softmax loss makes the best accuracy in the table 1. Secondly, we calculate the metrics of uniform and alignment among four representations outputted by CMC model. Surprisingly, the uniform and alignment from softmax do not show priority. Instead, InfoNCE leads in the uniformity metric and Triplet leads in the alignment metric.

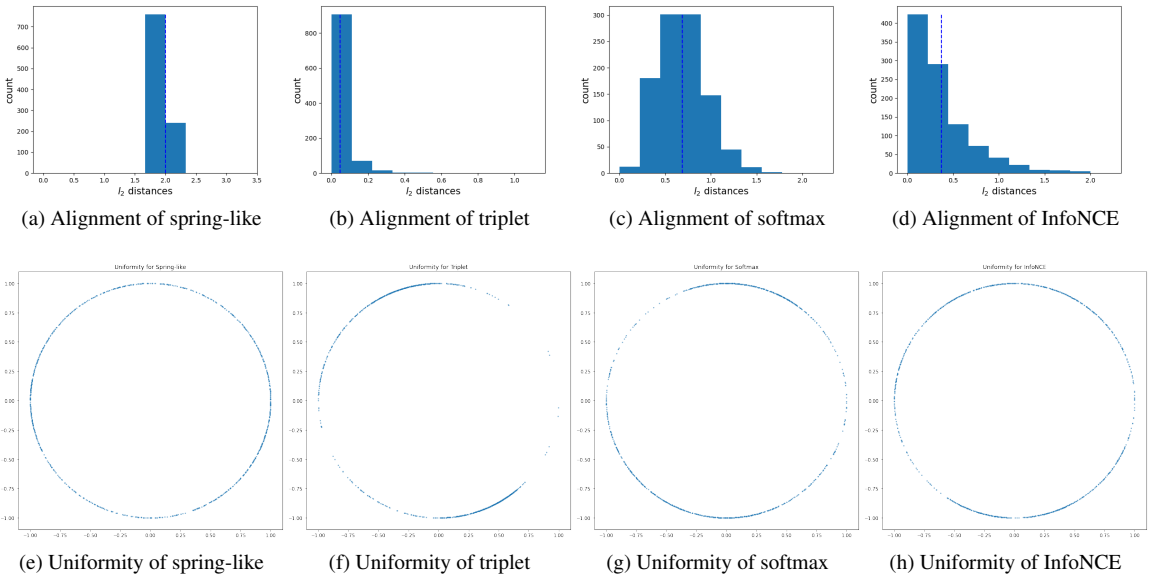
Loss type	Accuracy	Uniformity	Alignment
Spring-like	22.9%	-0.012	2
Triplet	34.3%	-0.516	**0.0473**
Softmax	**48.1%**	-2.214	0.686
InfoNCE	41.8%	**-2.636**	0.374

Table 1: Linear classifier accuracy and uniformity/alignment metric. The **bold** marker denotes the best performance. We evaluate uniformity/alignment on 1000 test samples.

To give a straightforward insight about those effect of contrastive loss, we visualize these two metrics in Figure 4.

To interpret these visualizations, first, for alignment, we would like to see in the plot a pattern that the bars in the histogram gather to the left side of the plot, and the more left, the better. This is because we want the the distances between positive pairs to be as small as possible. In the plots of alignment derived from different types of contrastive losses, the mean value represented by the dash line coincides with each alignment metric in all the plots. Also, the plot demonstrates that triplet loss can make positive pairs most similar in the representation. As the conclusion, in terms of alignment, all types of losses generally do well, and triplet loss performs best.

For uniformity, we apply t-SNE [11] to visualize the high-dimension feature onto the 2-D plane. To interpret the plots, we would like the points evenly distributed around the circle to ensure that various features are learnt. We observe that data samples in the plot of spring-like distribute more evenly, which actually contradicts to our former numerical experiment result. We speculate that the t-SNE algorithm



(a) Alignment of spring-like (b) Alignment of triplet (c) Alignment of softmax (d) Alignment of InfoNCE

(e) Uniformity of spring-like (f) Uniformity of triplet (g) Uniformity of softmax (h) Uniformity of InfoNCE

Figure 4: Visualization of alignment and uniformity. The dash line in upper row denotes the mean of alignment metric. We visualize it on 1000 test samples. In lower row, we visualizes high-dimensional data by t-SNE.

accounts for this phenomenon. Despite the contradiction from spring-like loss, we would like to conclude that InfoNCE and Softmax loss makes a uniform distribution of data, while triplet does fail.

4.3. Discussion

In general, the experimental result is consistent with our expectations by inspecting the construction of different types of loss and their performance. We see that softmax has the best performance in accuracy. Meanwhile, the two metrics of uniformity and alignment on softmax are also relatively good (though not the best). Besides, triplet loss is good at minimizing the alignment, since triplet loss's formula considers positive and negative sample simultaneously. It attaches vital importance to pulling positive sample closer than negative sample by a large margin. Thus, the encoder trained by triplet loss are more likely to assign similar features to similar samples. InfoNCE strongly leads in optimizing uniformity property. Compared with other choices of loss, InfoNCE is the only one loss to take global samples and noise approximation into account. Therefore, InfoNCE prefer the encoder to assign feature more globally so that the good uniformity is guaranteed.

5. Conclusion

In the project, we surveyed several designs of contrastive loss and analyzed the effect of different types of contrastive losses on two criteria: alignment and uniformity. We implemented these losses and did the experiments on the downstream classification task. Then, we compared each contrastive loss via calculating and visualizing alignment and uniformity. As the conclusion, we empirically demonstrated that the choice and construction of contrastive loss would significantly affect the learning of the models.

As the future work, we can do a more comprehensive survey. First, the encoder architecture is too shallow for nowadays downstream task. If a deeper encoder (e.g. resnet [8]) is applied, we may get more useful and promising insights. Besides, we can replace the linear readout head with nonlinear model MLP to better utilize and weight the representation that the encoders assign to sample. On top of model, the STL-10 dataset we chose is a small dataset in term of scale. Conducting our survey on a larger dataset such as ImageNet [4] might be helpful.

References

[1] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in Neural Information Processing Systems*, 33, 2020.

[2] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020.

[3] Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 215–223, 2011.

[4] 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.

[5] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in Neural Information Processing Systems*, 33, 2020.

[6] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742. IEEE, 2006.

[7] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738, 2020.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[9] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. *arXiv preprint arXiv:1808.06670*, 2018.

[10] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[11] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

[12] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[13] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[14] Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. In *Advances in neural information processing systems*, pages 1857–1865, 2016.

[15] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. *arXiv preprint arXiv:1906.05849*, 2019.

[16] Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. *arXiv preprint arXiv:2005.10242*, 2020.

[17] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3733–3742, 2018.