Exploring SimCLR for Self-Supervised Learning: CSC 422 Final Project

Logan Blake
Department of Computer Science
North Carolina State University
ldblake@ncsu.edu

Rob Clayton
Department of Computer Science
North Carolina State University
rtclayt2@ncsu.edu

Bennett Zug
Department of Computer Science
North Carolina State University
cbzug@ncsu.edu

Abstract

Self-supervised contrastive learning has emerged as a powerful alternative to large-scale labeled pre-training. We evaluate the SimCLR framework on the STL-10 dataset, focusing on how its representations mature over time. We train a ResNet-18 backbone for 300 epochs and use linear probing and t-SNE visualizations to evaluate the learned representations. Top-1 accuracy steadily improves to 89.6% after 200 epochs, with minimal further gains, reflecting the proportion of augmented pairs for which an element is closer to its counterpart than to any other negative sample. t-SNE visualizations show that our visualizations learned to separate vehicle classes, but still retained significant overlap among animal classes. Linear probing results further confirm this trend, showing high recall for vehicle classes (e.g., "Car" at 60.75%) but lower performance for animals (e.g., "Dog" at 20.5%). These findings demonstrate that contrastive pretraining can effectively learn useful features even from limited data. However, the difficulty in distinguishing visually similar animal classes suggests opportunities for improvement, such as through domain-specific augmentations or balanced sampling. Overall, our findings reinforce the growing relevance of self-supervised contrastive methods in data-scarce domains such as medical imaging, where robust feature learning without exhaustive annotation is critical.

1 Background

Labeling large vision datasets is time-consuming and costly. For example, the 14-million member ImageNet image dataset required tens of years of annotation effort, translating to hundred of thousands of dollars in expenses. In specialized fields like radiology, where every scan must be reviewed by trained experts, the process is even slower and more costly, which limits how much data can be collected for deep learning models.

Self-supervised learning (SSL) provides a solution to this challenge by utilizing unlabeled data. Among SSL techniques, contrastive learning stands out as a powerful approach, training an encoder to distinguish between similar and dissimilar examples without requiring explicit labels. SimCLR is one of the simplest yet most influential SSL frameworks, demonstrating that high-quality, transferable representations can be learned from unlabeled data through contrastive methods.

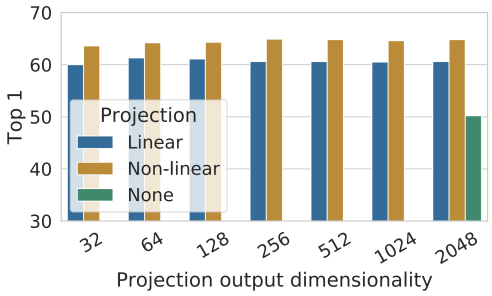

Figure 1: Figure from the SimCLR paper (Chen et al., 2020), showing that using a non-linear projection head significantly improves model accuracy by over 10% compared to no projection, and also outperforms linear projection by an additional 3%.

The key idea of SimCLR (Chen et al., 2020) is to create "data augmentations," versions of the unlabeled images with random crops and color jitters, and treat those augmentations as belonging to the same class, attempting to maximize the similarity between their embeddings.

SimCLR uses three key elements for effective SSL pre-training:

- **Complex augmentations**: By creating multiple augmented versions of each image through random crop and resize, color jitter, Gaussian blur, and horizontal flip, the model learns more generalized representations rather than trivial image similarities.
- **Projection head**: A learnable, non-linear projection head is introduced before the loss function, which significantly improves the quality of the learned representations. The projection head consists of a two-layer multi-layer perceptron (MLP), where the hidden layer uses a ReLU activation function. Figure 1 shows that the model accuracy improves through introducing this nonlinearity.
- **Large batch size**: Including many labels per batch allows the model to see more a diverse set of negative samples in each batch, with each batch more closely representing the data.

2 Methodology

The SimCLR (Chen et al., 2020) algorithm is quite a simple one. First, we generate augmentations for each example $\boldsymbol{x}$. The data augmentations used are a random crop and rescale, color jitter, and random Gaussian blur. We apply these augmentations twice, to create two augmented data examples generated from the same original image, $\tilde{\boldsymbol{x}}_i$ and $\tilde{\boldsymbol{x}}_j$. In the code by Silva (2020), these augmentations are generated by the `get_simclr_pipeline_transform()` function in the `ContrastiveLearningDataset` class. The code also applies a horizontal flip and a grayscale filter with probability $p = 0.5$ and $p = 0.2$ respectively.

Next, these augmentations are fed through an encoder to generate representation vectors. We use ResNet-18 (He et al., 2015), though the algorithm allows for a variety of choices. This generates representations $\boldsymbol{h}_i = f(\tilde{\boldsymbol{x}}_i) = \text{ResNet}(\tilde{\boldsymbol{x}}_i)$. In the code, this is defined through the `backbone` variable of the `ResNetSimCLR` class.

We then use a neural network to map these representations to a smaller space and introduce some nonlinearity. We use an MLP to generate $\boldsymbol{z}_i = g(\boldsymbol{h}_i) = W^{(2)}\sigma(W^{(1)}\boldsymbol{h}_i)$, with $W^{(n)}$ the weights arrays and $\sigma(\cdot)$ a ReLU. In code, this projection head is defined through the line

```
nn.Sequential(nn.Linear(dim_mlp, dim_mlp), nn.ReLU(), self.backbone.fc)
```

in the `ResNetSimCLR` class.

We then apply a contrastive loss function to the $\boldsymbol{z}_i$ representations. The paper terms the loss used *NT-Xent*, normalized temperature-scaled cross entropy loss. For a pair of two sampled augmentations

i, j, the loss is defined

$$\ell_{i,j} = -\log \frac{\exp(\text{sim}(\boldsymbol{z}_i, \boldsymbol{z}_j)/\tau)}{\sum_{k=1}^{2N} \mathbb{I}_{[k \neq i]} \exp(\text{sim}(\boldsymbol{z}_i, \boldsymbol{z}_j)/\tau)}, \tag{1}$$

with $\exp(\boldsymbol{u}, \boldsymbol{v})$ the standard cosine similarity $\boldsymbol{u}^\top \boldsymbol{v} / \|\boldsymbol{u}\| \|\boldsymbol{v}\|$, $\mathbb{I}$ an indicator function returning 1 when $k \neq i$ and 0 otherwise. Conceptually, this loss function aims to minimize the distance between positive pairs (augmented versions of the same image), and maximize the distance between negative pairs (augmented versions of different images). For each minibatch of size N, the loss generates a positive pair for each sample and considers all other pairs as negative examples. In the code, this loss function is defined as `info_nce_loss()` in the `SimCLR` class. The final loss, $\mathscr{L}$, is a summation of the pairwise loss across all positive pairs in the minibatch. We then optimize the encoder network $f(\cdot)$ to minimize this loss, and discard the projection head $g(\cdot)$ after training. The resulting model can then be used to extract high-quality visual representations.

Figure 2 depicts the model architecture graphically.

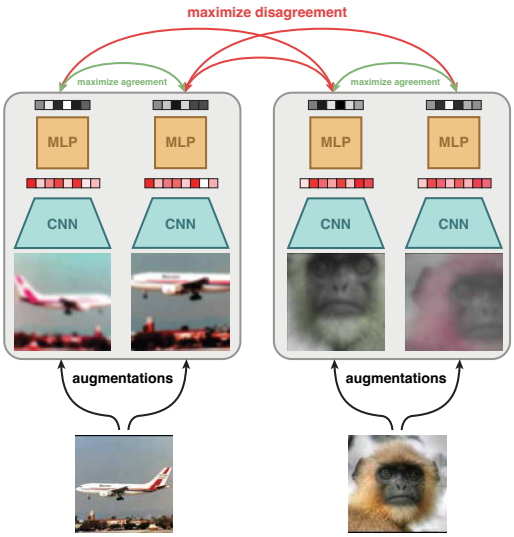

Figure 2: SimCLR model architecture

3 Experiment

We trained our model on the STL-10 dataset (Coates et al., 2011) using a ResNet-18 backbone. Training was performed for 300 epochs using the Adam optimizer with a batch size of 256. The learning rate was initially set to 3×10^{-4} and scheduled with cosine annealing. Our implementation[1] was based on a slightly modified version of the PyTorch code provided by Silva (2020). Experiments were run on an NVIDIA GTX 1070 Ti GPU over approximately 10 hours.

Figure 3 shows the training loss and Top-1/Top-5 accuracy over the course of training. The accuracy reported here reflects the proportion of augmented pairs for which an element of the pair is closer to its corresponding counterpart than to any other negative sample. This accuracy measure is based on the similarity of augmented pairs and is not related to class labels. The model converges steadily over the first 200 epochs, after which improvements in accuracy plateau.

To better understand how the model's representations evolved during training, we visualized the learned embeddings at several epochs using t-SNE (van der Maaten and Hinton, 2008). We used a labeled subset of the training data and a perplexity value of 50. The results are shown in Figure 4.

At epoch 1, the embeddings are largely unstructured, with significant overlap across classes. By epoch 100, we observe the emergence of distinct clusters, which become more pronounced by epoch

[1] `https://github.com/bennettzug/SimCLR`

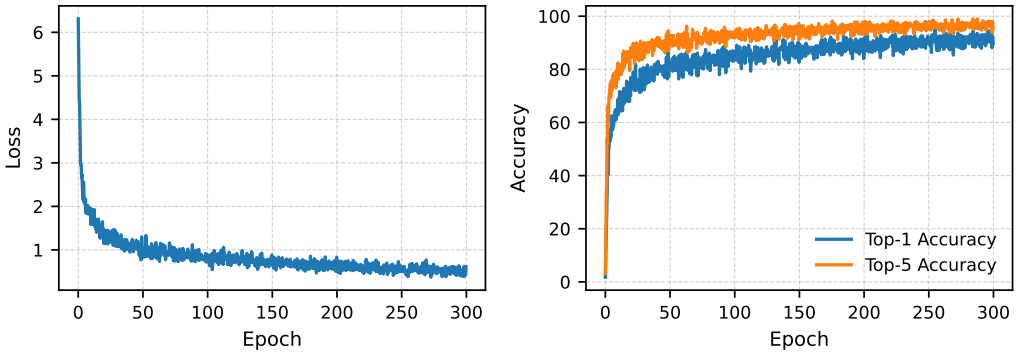

Figure 3: Training loss and Top-1/Top-5 accuracy over 300 epochs.

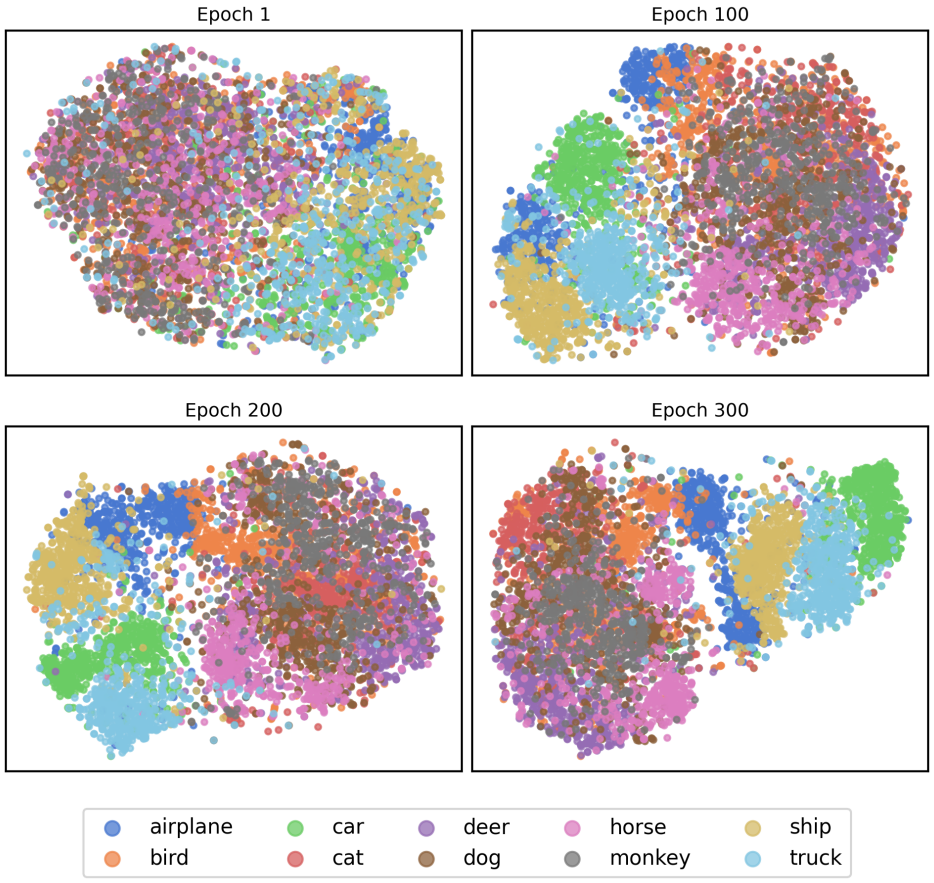

Figure 4: t-SNE visualizations of learned STL-10 embeddings at selected training epochs (1, 100, 200, and 300), projected into 2D space. Each point represents a labeled image colored by class.

200. At epoch 300, some clusters are clearly well-formed, although some classes remain entangled. Specifically, all of the vehicle related classes are in distinct clusters, but there remains considerable overlap between the animal clusters, indicating that the model has learned to distinguish certain classes better than others. Some of this difficulty might be because the unlabeled version of STL-10 contains more classes, and specifically more animals, than the labeled training variant we created the visualizations on, which could be affecting the learned representations.

In addition to evaluating the learned representations using t-SNE visualizations, we performed a linear probe evaluation on the learned embeddings. Linear probing involves training a linear classifier on top of the representations, to assess their quality for downstream tasks. This allows us to gauge how well the learned features generalize to a classification task. The results are summarized in Table 1. These results seem to confirm what we see in the qualitative t-SNE data, showcasing that the vehicle classes have relatively high recall values, with for example "Car" obtaining a recall of 60.75%. On the other hand, some of the animal classes have more modest results, with "Dog" having a recall of just 20.5%.

Class	**Precision**	**Recall**	**F1-Score**
Airplane	0.5375	0.5637	0.5503
Bird	0.3411	0.3675	0.3538
Car	0.4880	0.6075	0.5412
Cat	0.2272	0.1588	0.1869
Deer	0.3911	0.3975	0.3943
Dog	0.2966	0.2050	0.2424
Horse	0.3617	0.4512	0.4016
Monkey	0.3128	0.3738	0.3405
Ship	0.5341	0.5575	0.5456
Truck	0.4024	0.2963	0.3413
Average	0.3892	0.3979	0.3898

Table 1: Classification Report for Linear Probe Evaluation on STL-10

4 Conclusion

This study effectively demonstrates the potential of the SimCLR framework for generating high-quality, generalized visual representations through self-supervised learning. Training on the STL-10 dataset revealed notable progress in the organization of learned feature embeddings, as shown by t-SNE visualizations. Initially, the embeddings were unstructured. However, as training progressed, they shifted to form more distinct clusters, particularly for vehicle classes, while animal classes still exhibited considerable overlap. This highlighted the challenges in distinguishing visually similar image categories. By epoch 200, top-1 accuracy reached 86.7%, plateauing soon after at around 89.6%, indicating that the model had converged.

The linear probe evaluation supported these findings, with higher recall scores for vehicle classes, such as "Car" (60.75%), compared to animal classes like "Dog" (20.5%). These results emphasize the effectiveness of contrastive pretraining in learning valuable features from limited, unlabeled data. However, the challenges in fine-grained animal classification suggest potential avenues for improvement, such as introducing domain-specific augmentations or addressing class imbalance through more balanced sampling between classes.

Looking ahead, future work could explore these avenues through the use of stronger or more targeted augmentations, as well as scaling up to a deeper architecture such as ResNet-50, to test the robustness of the trends observed. Additionally, the application of SimCLR to real-world datasets, such as those in medical imaging or satellite photography, could provide valuable insights into its practical applicability in data-scarce domains where data collection is costly, making efficient feature learning even more critical. While further training with more of epochs could also be used, the plateau exhibited after the first 200 epochs suggests that the training had already reached its optimal state with the current setup. However, the proposed improvements could change this outcome. Furthermore,

the hardware used to train could be improved, as working with a home computer-level of processing power is far more limited than the resources that large companies, such as Google, are utilizing to efficiently train their models.

References

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 1597–1607. PMLR.

Adam Coates, Andrew Ng, and Honglak Lee. 2011. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pages 215–223, Fort Lauderdale, FL, USA. PMLR.

Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778.

Thalles Santos Silva. 2020. Pytorch SimCLR: A simple framework for contrastive learning of visual representations.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. In *Journal of Machine Learning Research*, volume 9, pages 2579–2605.