Visualizing Representations Found in Simple Siamese Representation Learning

Emma Lind
emmal8@kth.se

Nathan Bosch
nbosch@kth.se

Yiming Hu
yimingh@kth.se

Abstract

This report aims to re-implement a simplified version of SimSiam, a siamese network modelling approach to generate high-quality representations of images, and perform experiments to present the representations learned by the network and explain its decisions. Firstly, the model's performance is evaluated in order to be compared to the results in the original paper. Then investigations of the latent space are performed by reducing the dimensionality and visualizing the features in 2-dimensions space to observe if data with similar labels form clusters. In addition, investigations of the stop-gradient are performed. Lastly, the model's decisions are visualized using explainable AI approaches (SHAP). The re-implementation of SimSiam and investigations were performed with smaller models (Resnet-18 encoder as opposed to Resnet-50) and simpler datasets (CIFAR-10 as opposed to Imagenet). The result has shown that both sufficient pre-training for the encoder and stop-gradient are very crucial for SimSiam. By visualizing the model's decision-making process and the representations found, we further identify drawbacks and weaknesses in the found image representations. The code is available on GitHub [1].

1 Introduction

Advancements in representation learning have enabled the use of deep learning (DL) methods for a variety of complicated tasks. Notably, DL models that can effectively learn in a self or semi-supervised manner have gained traction when applied to tasks where limited, high quality labelled data is available. The aim of these approaches is often to train a model which finds meaningful embedded representations of the data, e.g., images, which can then be used for a number of downstream tasks, such as classification and image segmentation. For images, contrastive learning methods, such as SimCLR (1) and SimCLRv2 (2), and Siamese network training structures have achieved strong performance in several computer vision tasks with limited labels available.

A notable constraint with many existing self-supervised representation learning approaches is the high degree of complexity in both the model and training. This is undesirable, especially when computational resources are limited. The focus of this report, therefore is on the paper (3), in which the authors outline a simple yet effective method for self-supervised learning of images. In principle, their approach is similar to other approaches such as BYOL (4), SimCLR (1), and SwAV (5), but reduces the structure to a simpler Siamese network. To avoid gradients collapsing throughout model training, a simple stop-gradient operation in the loss term is employed. Using a linear classifier that leverages the found image representations, they show that their method SimSiam performs competitively with other approaches and does not produce collapsing solutions.

This report aims to re-implement the self-supervised algorithm SimSiam presented in the paper (3) and to perform experiments on the latent space (i.e., the representation found by the encoder in

[1] https://github.com/NatBos99/DL_Advanced_Simsiam

Project report for DD2412 or FDD3412

SimSiam) in order to investigate and visualize the representations learned by the network. The experiments on the latent space encompass four components. Firstly, to investigate the impact of the model's performance when reducing the dimensionality of the latent space. Secondly, visualize the latent space to examine if clusters can be formed for each respective label. Thirdly, observe the performance when using/not using the stop gradient. Lastly, visualize the model's decision using SHAP (6). These experiments are relevant to interpret and understand the model. Investigations of the impact of the stop-gradient are interesting as the original paper concludes that it is a core component to avoid collapsing.

Our experiments show that the re-implementation and its adoptions for lower computational resources could not achieve as high accuracy as the original paper. Nevertheless, decent accuracy (>70%) was achieved when fine-tuning the whole model after SimSiam pre-training. The representations found without fine-tuning were, in general, insufficient. This is reflected in the visualization of features in 2-dimensional space using PCA and t-SNE. Using SHAP, we found that a linear classifier that uses the representations found in the self-supervised training (i.e., no fine-tuning of the entire model) does not focus on relevant sections of the image, instead frequently focusing on the top left of the image. This again indicates that the model finds low-quality representations of the input data. Upon increasing the length of training (to 400 epochs) and significantly increasing the size of the model, we did achieve far more acceptable representations, which is reflected both in the visualization of results and the prediction accuracy.

2 Related Work

The article (7) studies why methods using non-contrastive SSL do not collapse, such as BYOL and SimSiam. Their study provides insights into how these models learn and how multiple factors, such as stop-gradients, exponential moving average, weight decay, and predictor network, come into play to avoid representational collapse. Furthermore, they emphasize that the existence of the predictor and stop-gradient are vital for preventing collapse. Therefore, this project conducts the experiment of observing the latent space when using and removing the stop gradient.

Furthermore, the method FixMatch (8) also implements a simplification of already existing methods. Despite its simplicity, FixMatch achieved state-of-the-art performance. This is yet another confirmation that a more complex model architecture does not necessarily imply better performance. Thus, simplified variants of existing methods are of interest to explore.

3 Methods

The method presented in the original paper (3) consists of an encoder network f and an MLP projection head h. During training, an image x is augmented in two separate ways to form x_1 and x_2. Both augmented images represent the same underlying image as x (and hence the same concept and label), but will look completely different to a deep learning model. Both images are passed through the encoder network, i.e., $f(x_1) = z_1$ and $f(x_2) = z_2$. After which, an MLP head, also known as the predictor layer h, is used to compute $h(z_1) = p_1$ and $h(z_2) = p_2$. Negative cosine similarity loss is used for both prediction and encoder pairs (p_1 and z_2, p_2 and z_1), and the loss equals the average of $D(p_1, z_2)$ and $D(p_2, z_1)$. The core component in their method is the stop-gradient, which is applied to the view not processed by h. i.e., When computing $D(p_1, z_2)$, we do not allow the gradients to backpropagate for z_2. Note that by minimizing the negative cosine similarity, we aim to increase the similarity in the different augmentations of the same image. After training, similar images should therefore have similar latent representations.

To apply the trained model described above to a classification task, we add a layer on top of the encoder, e.g., prediction = $g(f(x))$. This layer, $g(x)$ is a standard linear layer followed by the softmax activation function, where the number of nodes corresponds to the number of classes in the dataset. In the fine-tuning process, we investigate two scenarios: 1) Freeze the encoder ($f(x)$) layer or 2) fine-tune the entire network. Scenario 1 allows us to investigate how strong the image representations found by the Siamese network are by default. Scenario 2, on the other hand, allows us to observe the potential of the model to learn with limited data. Comparing both scenarios allows us to gain a deeper understanding of the quality found in SimSiam representations.

3.1 Visualizing SimSiam Representations

The t-distributed stochastic neighbour (t-SNE) can be used to project high-dimensional data into lower dimensions and will be used in this project to visualize the features learned by the network. Basically, the projection is made by converting similarities of data to their joint probabilities and then minimizing the Kullback-Leibler divergence between the joint probabilities of the low-dimensional embedding and the high-dimensional data (9).

Principal Component Analysis (PCA) is a method to project high-dimensional data into lower dimensions. It will also be used in this project to visualize the features learned by the network. PCA analyses the high-dimensional data described by several dependent variables, often inter-correlated. The goal is to extract the important information from the data to express this information as a set of new orthogonal variables called principal components (10).

3.2 SHAP DeepExplainer

SHapley Additive exPlanations (SHAP) is a framework for interpreting predictions of a machine learning model. Basically, the method assigns each feature an importance value for a particular prediction (6). In visualization, the features marked with red colors are those important ones that make a positive contribution to the final prediction towards the assigned class. While features marked blue mean, they have a reversed effect upon the prediction. In the experiment, SHAP is used to reveal the distribution of the important values along with features, which demonstrates the model's horizon and feature-wise judgment. If the model is not well trained, it tends to be short-sighted in terms of its limited view throughout the image space.

4 Data

The original paper (3) performs unsupervised pre-training on the 1000-class ImageNet training set. Then the learned representations are evaluated by training a supervised linear classifier on the frozen representations and then testing it on the validation set. The best configuration of the network achieved an accuracy of 68.1% on the test set. This project implements the same procedure but on the CIFAR-10[2] dataset due to constraints of computational resources. However, as aforementioned, SimSiam can be seen as a simplified version of SimCLR. According to the article (1), SimCLR achieved an accuracy of 94.0% on the CIFAR-10 dataset. The authors argue that the accuracy can be improved by incorporating more data augmentations.

5 Experiments and findings

The following section describes the experiments conducted in this project and the corresponding findings. Five experiments were added on top of the re-implementation, and each is described below. All experiments were done using ResNet18 as a backbone network, batch size 16, and pre-training was done during 50 epochs. The stochastic gradient descent (SGD) optimizer was used with weight decay and momentum. We used a learning rate of 0.05, a weight decay of $5e^{-4}$, and momentum 0.9. Fine-tuning was done for 50 epochs with the same optimizer and parameters outlined previously. The fine-tuning was done with the entire training dataset (50000 labelled images).

5.1 Model performance & reduction of the latent space

Firstly, the model's performance on CIFAR-10 was evaluated to investigate if the model achieved similar performance to what was outlined in the original paper. Important to note here is that the original paper used ResNet50, ImageNet instead of CIFAR-10, pre-trained for 100 epochs. The results are presented in Table 2. When training for 50 epochs, the model achieves a Top-1 accuracy of the highest 34.2% when freezing the features learned from the pre-training. This is a lower accuracy in comparison to the results of the original paper, where the model achieved an accuracy of 91.8%. The reason for the low accuracy is probably due to the lack of training since in the original paper the model is trained for 800 epochs with batch size 512. In order to prove this, we pre-trained the model with an encoder dimensionality of 2048 for 400 epochs. Its Top-1 accuracy on the test data reaches

[2]https://pytorch.org/vision/stable/datasets.html#cifar

60.88% which is a significant improvement compared to the previous ones. It offers strong evidence that a fully pre-trained encoder is very important for SimSiam.

Secondly, the dimensionality of the latent space was reduced to observe the impact of the model's performance in terms of accuracy. The dimensionality was tested using five different dimensions (2048, 512, 128, 32, and 2). The results are presented in Table 2 as well. What can be observed here is that for Top-1 accuracy, when reducing latent space and freezing the encoder, the accuracy seems to decrease slightly; hence, the model is not as good to capture the latent representations when the dimension is decreasing.

Table 1: Results of testing the model performance, reduction of the latent space and analysis of frozen encoder

Encoder Dimensionality	Frozen Encoder	Top-1 Accuracy	Top-5 Accuracy
512	✓	34.1%	85.1%
512	✗	69.3%	96.0%
128	✓	34.2%	85.0%
128	✗	70.8%	96.0%
32	✓	30.8%	83.0%
32	✗	69.2%	95.0%
2	✓	22.8%	80.0%
2	✗	62.7%	91.0%
2048*	✓	60.88%	96.0%

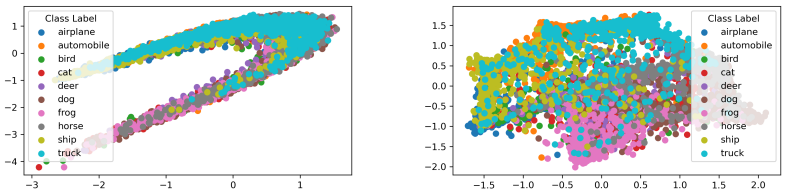
Model with symbol * is trained for 400 epochs

5.2 Visualization of the latent space

The features learned by the network are visualized in the 2-dimension space in order to investigate if the features could form clusters corresponding to each label. The projection to the 2-dimension space was made using PCA and t-SNE.

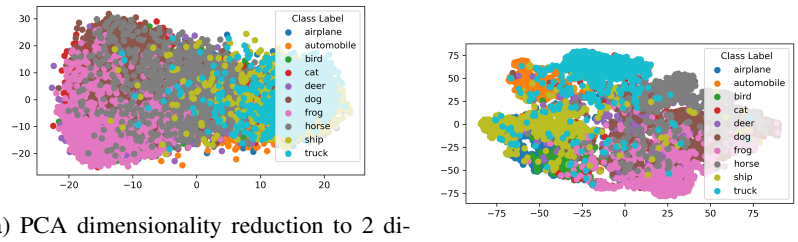
Figure 1 visualizes the features found with an encoder of two dimensions when freezing the encoder and when fine-tuning the whole model using insufficiently pre-trained encoder . Figure 2 visualizes the features found when projecting the latent space from 2048 dimensions to two dimensions using PCA and t-SNE when freezing the encoder.

First and foremost, one can clearly see that no fine-tuning of the encoder result in poor cluster properties for Figure 1, although this is significantly improved when increasing the training time and model size, as can be seen in Figure 2. In Figure 1, more clusters can be distinguished when fine-tuning the entire model, even though there is still overlap between data from different classes. Comparing Figure 1a with Figure 2, one can see that the encoder with only two dimensions has very overlapping clusters, whereas projecting from 2048 dimensions to two dimensions using PCA or t-SNE results in more clearly separated clusters. In Figure 2, we also see that similar concepts are closer together. For example, the clusters for cat and dog overlap more than cat and truck.



(a) No fine-tuning of the encoder (b) Fine-tuning of the entire model

Figure 1: Visualization of CIFAR-10 features found with an encoder which has 2 dimensions



(a) PCA dimensionality reduction to 2 dimensions. Around 38% of the variance is explained.

(b) t-SNE dimensionality reduction to 2 dimensions.

Figure 2: Visualization of CIFAR-10 features found with an encoder which has 2048 dimensions. PCA and t-SNE were used to visualize the results.

5.3 Analysis of stop-gradient

The impact of the stop-gradient was investigated by observing the model performance when using/not using the stop-gradient. As shown in table 2, if the output of both networks is treated as a non-constant (without stop-gradient), the model performance including Top-1 accuracy and Top-5 accuracy will significantly decrease. Our result proves that using the stop-gradient can effectively prevent the model collapse, which is accorded with the original paper. Whereas if the entire model is fine-tuned, then the problem is reasonably avoided.

Table 2: Results of model performance on stop-gradient analysis

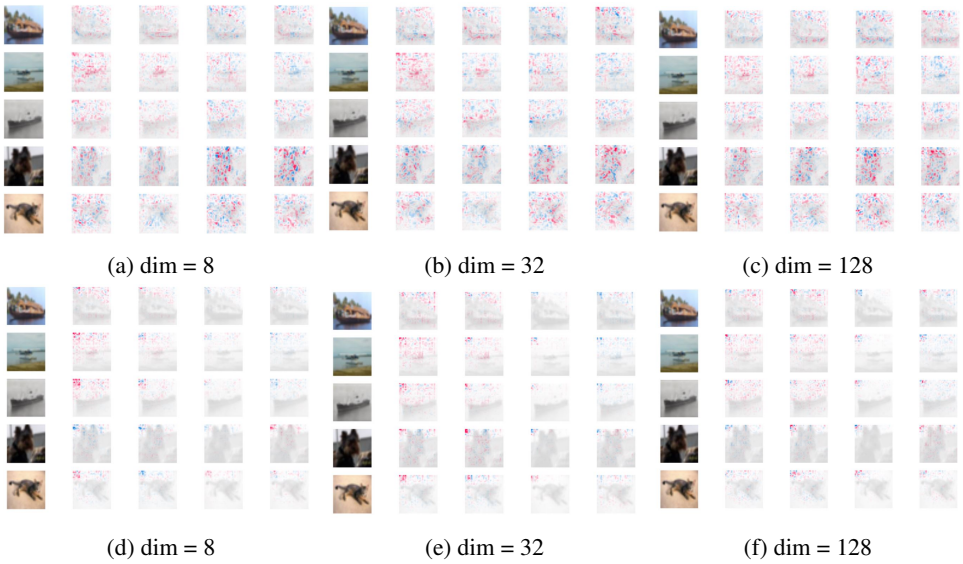
Encoder Dimensionality	Frozen Encoder	Stop gradient	Top-1 Accuracy	Top-5 Accuracy
512	✓	✓	34.1%	85.1%
512	✓	✗	10.4%	50.1%
512	✗	✓	69.3%	96.0%
512	✗	✗	69.2%	96.0%

5.4 Model interpretation using SHAP

Lastly, the SHAP DeepExplainer was implemented in order to increase the interpretability of the model and to visualize the model's decisions. Figure 3 present the result of the SHAP method using three different dimensions of the latent space. The figure explains four outputs for five images. The blue pixels decrease the model's output, whereas the red pixels instead increase the model's output. The result shows that fine-tuning the whole model Figure 3a-3c enables the model to achieve better representations. When only fine-tuning the classification layer Figure 3a-3c, the representations learned, seem to end up in the left upper corner, which indicates that the model is not able to learn the sufficient representations using only the self-supervised algorithm. Nevertheless, a reason for this could be that more training is needed. Moreover, the dimensionality of the latent space does not seem to significantly impact the ability to learn the representations. However, this could probably be different if another dataset were used. Such as ImageNet, which might need a larger latent space to express the representations.

6 Challenges

To re-implement the paper required a deep understanding of the method and architecture. We utilized both the explanation of the method in the paper and the published code to understand and re-implement the method. We had to make adoptions for lower computational resources; hence, we used the CIFAR-10 instead of the ImageNet dataset. Moreover, ResNet18 was used instead of ResNet50 to keep a light network architecture to make training and inference faster to efficiently tune hyperparameters and compare models. Another challenge is to perform the comparative study where we mainly focus on different dimensionality of the latent space and freeze/activate the backbone during fine-tuning. Several visualization methods were used to facilitate the analysis.



(a) dim = 8 (b) dim = 32 (c) dim = 128

(d) dim = 8 (e) dim = 32 (f) dim = 128

Figure 3: The result of using SHAP Deep Explainer for five different images and three dimensions of the latent space (8, 32 and 128) when fine-tuning the entire model (Figure 3a-3c) and when fine-tuning only the last layer (Figure 3d-3f).

7 Conclusion

One key finding found during this project is that representation learning can only be utilized if enough training data is present, computational power, and time to perform throughout pre-training. Representation learning might be more useful if the data is more complex, thus finding the underlying structure of the data. CIFAR-10 includes images that are not very sophisticated; hence, learning the features using the method presented in the paper might not be very suitable for CIFAR-10. Notably, PCA found that much of the variance in the model's image representations could be explained through only 2 dimensions, which is not the case when fine-tuning the model for a classification task. This indicates that the model representations are too simple to be very useful in downstream tasks. However, as argued in Section 5 more training would probably have resulted in better-learned representations, which was also proved running the pre-traning for 400 epochs. Moreover, another key finding is that the model's performance in terms of accuracy is highly correlated to the cluster properties generated by plotting the feature vector in the 2-dimensional space. Higher accuracy is achieved when more distinct clusters are present. Additionally, we study the effect of stop-gradient and find out it does effectively prevents the network from collapsing, which verifies the claim in the paper.

A future extension of this project could be to evaluate the SimSiam model on more complex data to see if the model is able to learn the underlying structure. For example, a suggestion could be to evaluate the performance of medical data, such as x-ray images. In addition to this, performing direct comparisons between SimSiam and other self-supervised algorithms with the same simple base models (e.g., Resnet-18) and data would be a worthwhile extension to this work.

8 Ethical consideration, societal impact, alignment with UN SDG targets

The authors of the article (3) create a simpler version of a model, thus, requires less computational power, which in turn can enable more people or organizations to utilize the method. In that sense, the purpose of the article can be seen as ethical.

The experiments conducted in this report increase the understanding of what the model learns by visually presenting the representations found in the latent space. Increasing the understanding of the black box in deep networks might enable developing even better models or understanding why a model has good or poor performance. The understanding might also help prevent situations where a model makes deplorable decisions. In the long run, these types of experiments could probably

contribute to better and more robust models and be more useful for real-world problems. Thus, it could possibly lead to pushing the UN SDG targets as the space of solvable tasks increases.

9 Self Assessment

According to the project grading guideline[3], an excellent project (grade B-A) requires re-implementing the paper and going beyond the scope of the papers method and experiments. This project reproduces the original paper and extends the results by visualizing the representations and the decisions made by the SimSiam network. Thus, this report provides novel insights into the decision making and latent representations of a SimSiam model. Hence, this project deserves a grade A.

References

[1] T. Chen, S. Kornblith, M. Norouzi, and G. E. Hinton, "A simple framework for contrastive learning of visual representations," *CoRR*, vol. abs/2002.05709, 2020. [Online]. Available: https://arxiv.org/abs/2002.05709

[2] T. Chen, S. Kornblith, K. Swersky, M. Norouzi, and G. Hinton, "Big self-supervised models are strong semi-supervised learners," *arXiv preprint arXiv:2006.10029*, 2020.

[3] X. Chen and K. He, "Exploring simple siamese representation learning," *CoRR*, vol. abs/2011.10566, 2020. [Online]. Available: https://arxiv.org/abs/2011.10566

[4] J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. H. Richemond, E. Buchatskaya, C. Doersch, B. A. Pires, Z. D. Guo, M. G. Azar *et al.*, "Bootstrap your own latent: A new approach to self-supervised learning," *arXiv preprint arXiv:2006.07733*, 2020.

[5] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin, "Unsupervised learning of visual features by contrasting cluster assignments," *arXiv preprint arXiv:2006.09882*, 2020.

[6] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. Curran Associates, Inc., 2017, pp. 4765–4774. [Online]. Available: http://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions.pdf

[7] Y. Tian, X. Chen, and S. Ganguli, "Understanding self-supervised learning dynamics without contrastive pairs," *CoRR*, vol. abs/2102.06810, 2021. [Online]. Available: https://arxiv.org/abs/2102.06810

[8] K. Sohn, D. Berthelot, C. Li, Z. Zhang, N. Carlini, E. D. Cubuk, A. Kurakin, H. Zhang, and C. Raffel, "Fixmatch: Simplifying semi-supervised learning with consistency and confidence," *CoRR*, vol. abs/2001.07685, 2020. [Online]. Available: https://arxiv.org/abs/2001.07685

[9] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne." *Journal of machine learning research*, vol. 9, no. 11, 2008.

[10] H. Abdi and L. J. Williams, "Principal component analysis," vol. 2, no. 4, pp. 433–459, 2010. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/wics.101

[3] https://canvas.kth.se/courses/28866/pages/project-grading-guideline