Leon Sick

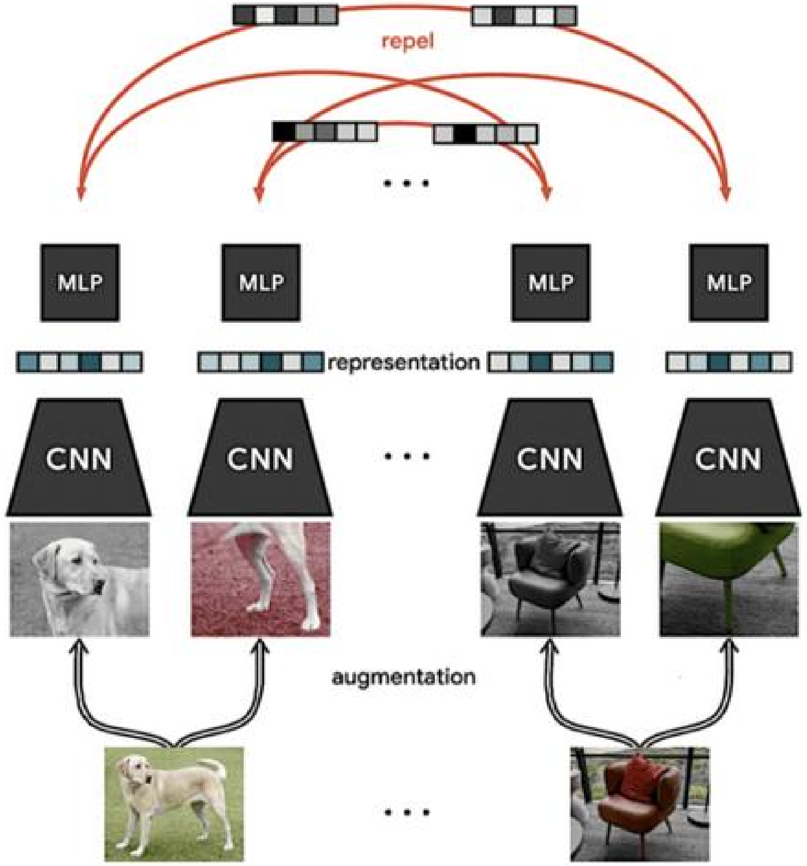
Dec 9, 2021 · 6 min read ★

Paper explained: A Simple Framework for Contrastive Learning of Visual Representations

Going over the ideas presented in the SimCLR paper

In this story, we will take a look at SimCLR: The architecture that led the computer vision research community to new heights in self-supervised pre-training for vision tasks.

SimCLR was presented in the Paper *"A Simple Framework for Contrastive Learning of Visual Representations"* by Chen et al. from Google Research in 2020. The ideas in this paper are relatively simple and intuitive, but there is also a novel loss function that is key for achieve great performance for self-supervised pre-training. I've tried to keep the article simple so that even readers with little prior knowledge can follow along. Without further ado, let's dive in!



An illustration of the SimCLR training procedure. Source: [1]

Pre-requisite: Self-supervised pre-training for computer vision

Before we go deeper into the SimCLR paper, it's worth quickly re-visiting what self-supervised pre-training is all about. **If you have been reading other self-supervised learning stories from me or you are familiar with self-supervised pre-training, feel free to skip this part.**

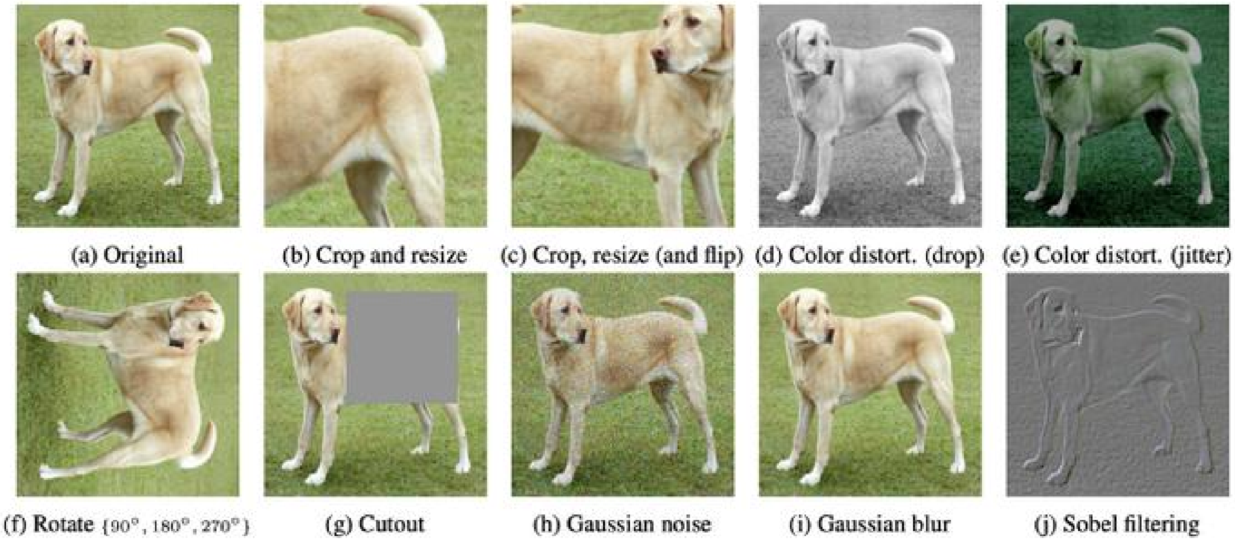
Traditionally, computer vision models have always been trained using **supervised learning.** That means humans looked at the images and created all sorts of **labels** for them, so that the model could learn the patterns of those labels. For example, a human annotator would assign a class label to an image or draw bounding boxes around objects in the image. But as anyone who has ever been in contact with labeling tasks knows, the effort to create a sufficient training dataset is high.

In contrast, **self-supervised learning does not require any human-created labels**. As the name suggest, **the model learns to supervise itself.** In computer vision, the most common way to model this self-supervision is to take different crops of an image or apply different augmentations to it and passing the modified inputs through the model. Even though the images contain the same visual information but do not look the same, **we let the model learn that these images still contain the same visual information**, i.e., the same object. **This leads to the model learning a similar latent representation (an output vector) for the same objects.**

We can later apply transfer learning on this pre-trained model. Usually, these models are then trained on 10% of the data with labels to perform downstream tasks such as object detection and semantic segmentation.

Learning image similarity with SimCLR

A key contribution by the paper is the use of **data augmentations**. SimCLR creates pairs of images to learn the similarity from. If we would input the same image twice, there would be no learning effect. Therefore, each pair of images is created by applying **augmentations or transformations to the image.**

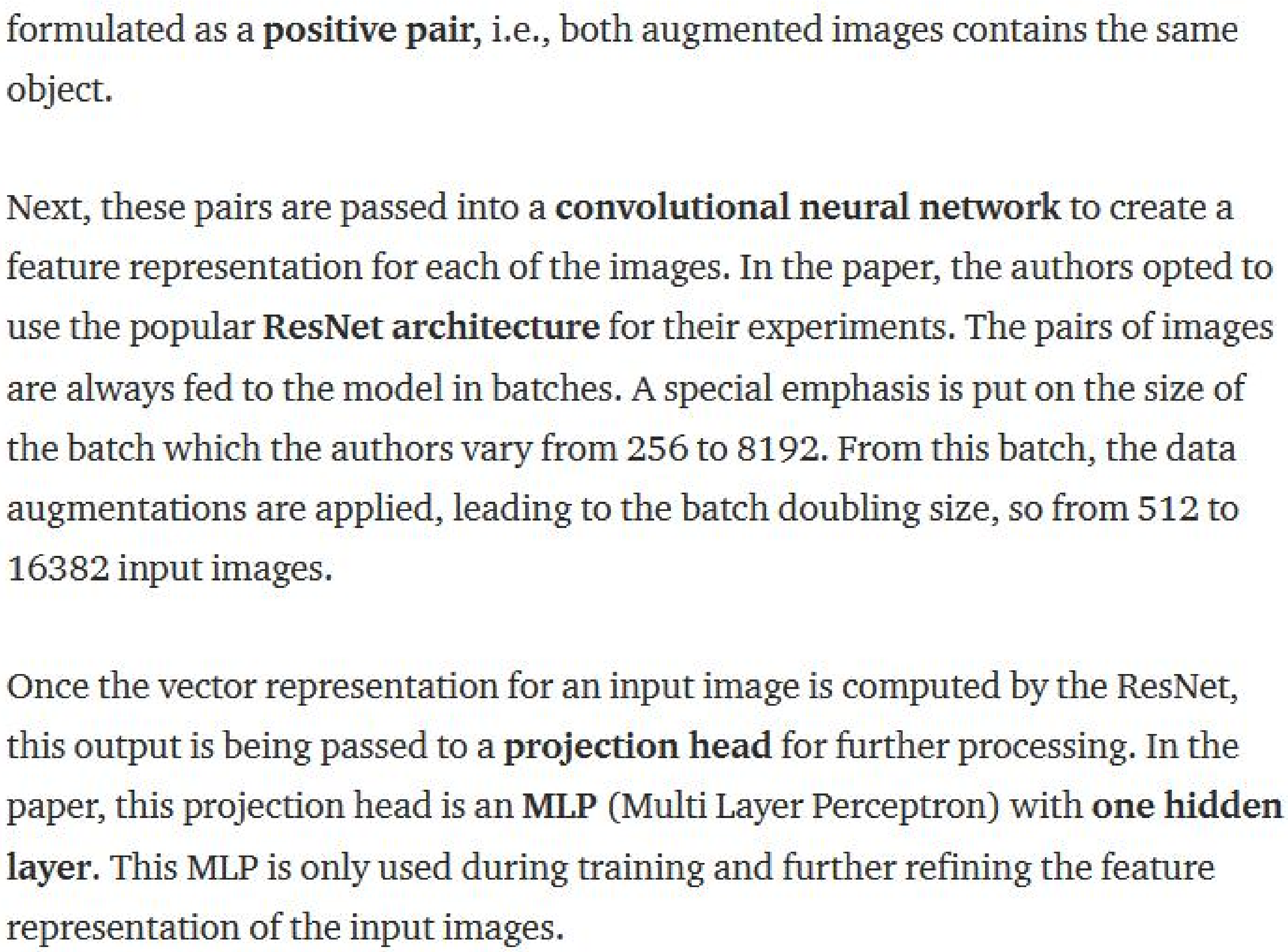


(a) Original (b) Crop and resize (c) Crop, resize (and flip) (d) Color distort. (drop) (e) Color distort. (jitter)

(f) Rotate {90°, 180°, 270°} (g) Cutout (h) Gaussian noise (i) Gaussian blur (j) Sobel filtering

Different data augmentations applied to an image of a dog. Source: [2]

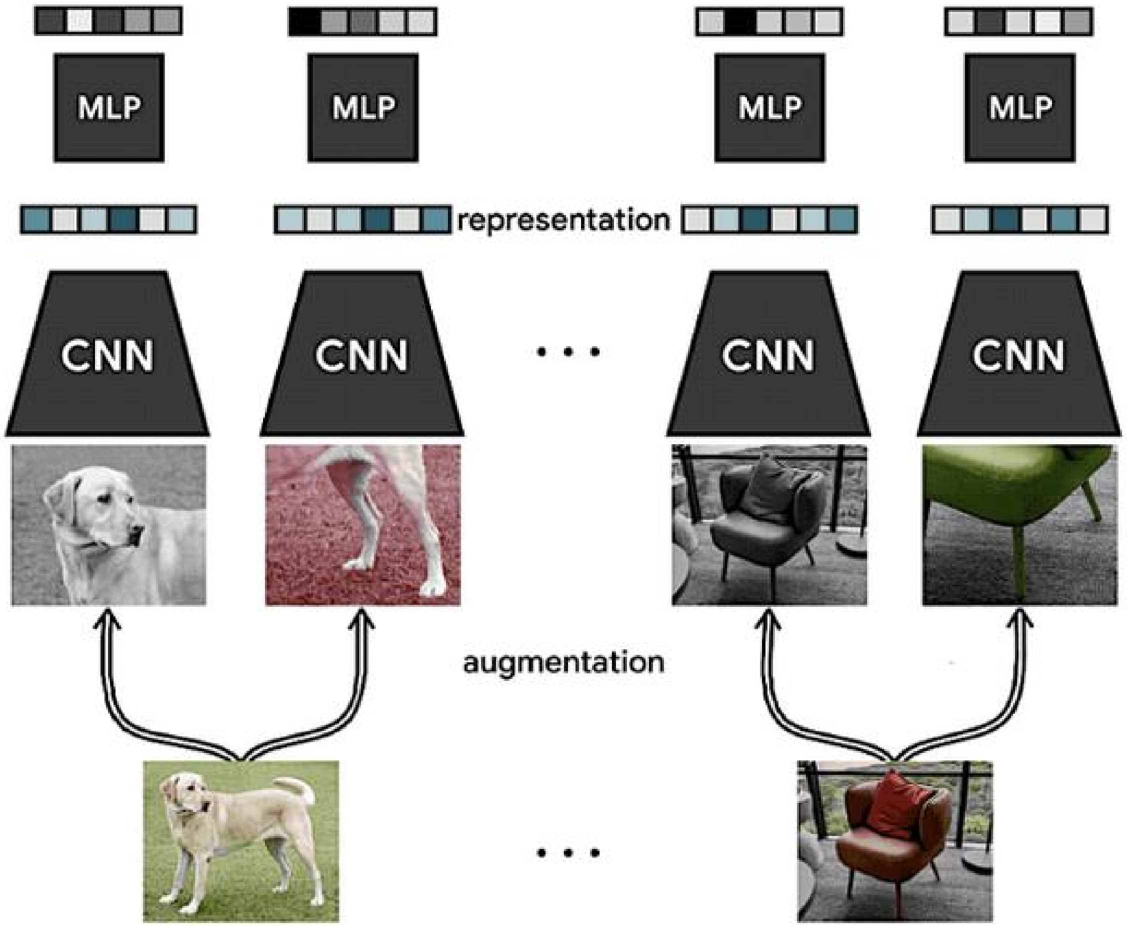
As can be seen in this excerpt from the paper, the authors apply different augmentations such as resizing, color distortion, blurring, noising and much more. They also take crops from different parts of the image which is important for the model to learn a consistent representation. The image can be cropped into a global and local view (full image and cropped part of the image) or adjacent views can be used (two crops from different parts of the image). Each pair is



formulated as a **positive pair,** i.e., both augmented images contains the same object.

Next, these pairs are passed into a **convolutional neural network** to create a feature representation for each of the images. In the paper, the authors opted to use the popular **ResNet architecture** for their experiments. The pairs of images are always fed to the model in batches. A special emphasis is put on the size of the batch which the authors vary from 256 to 8192. From this batch, the data augmentations are applied, leading to the batch doubling size, so from 512 to 16382 input images.

Once the vector representation for an input image is computed by the ResNet, this output is being passed to a **projection head** for further processing. In the paper, this projection head is an **MLP** (Multi Layer Perceptron) with **one hidden layer.** This MLP is only used during training and further refining the feature representation of the input images.

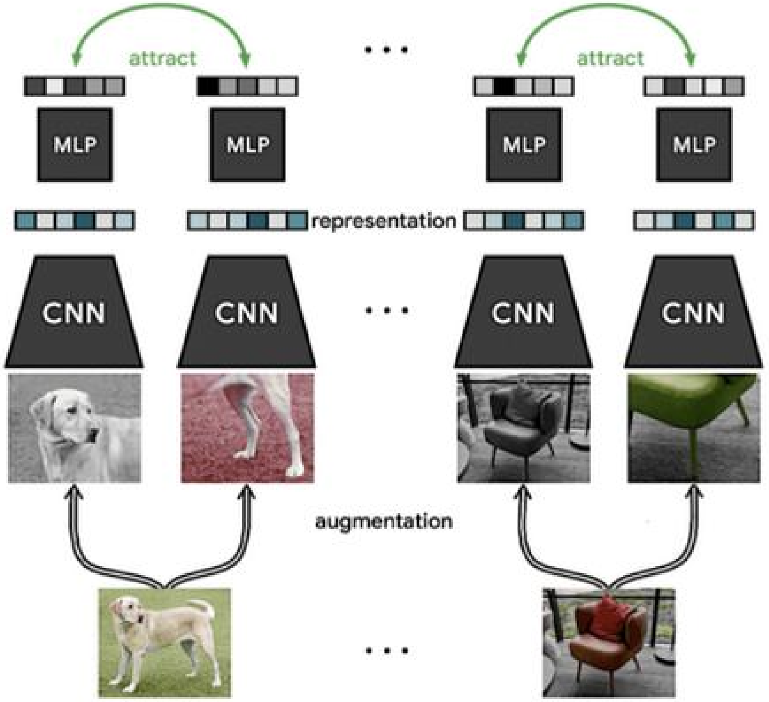


The SimCLR training process from the raw input image to the representation computed by the MLP. Source: [1]

Once the MLP computation is completed, the result reserves as the input into the loss function. **The learning goal of SimCLR is maximize agreement between different augmentations of the same image.** That means the model tried to **minimize the distance between images that contain the same object** and **maximize the distance between images that contain vastly different object.** This mechanism is also called **contrastive learning**.

One major contribution by the SimCLR paper the formulation of its **NT-Xent loss**. NT-Xent stands for normalized temperature-scaled cross entropy loss. This novel loss function has a property that is especially desirable: **Different examples are weighted effectively** allowing the model to **learn much more effectively from vector representations that are far away from each other** even though their origin is the same image. These examples the model perceives to be very different from each other are called **hard negatives.**

This loss effectively achieves an attraction of similar images, i.e., similar images are learned to be mapped closer together.

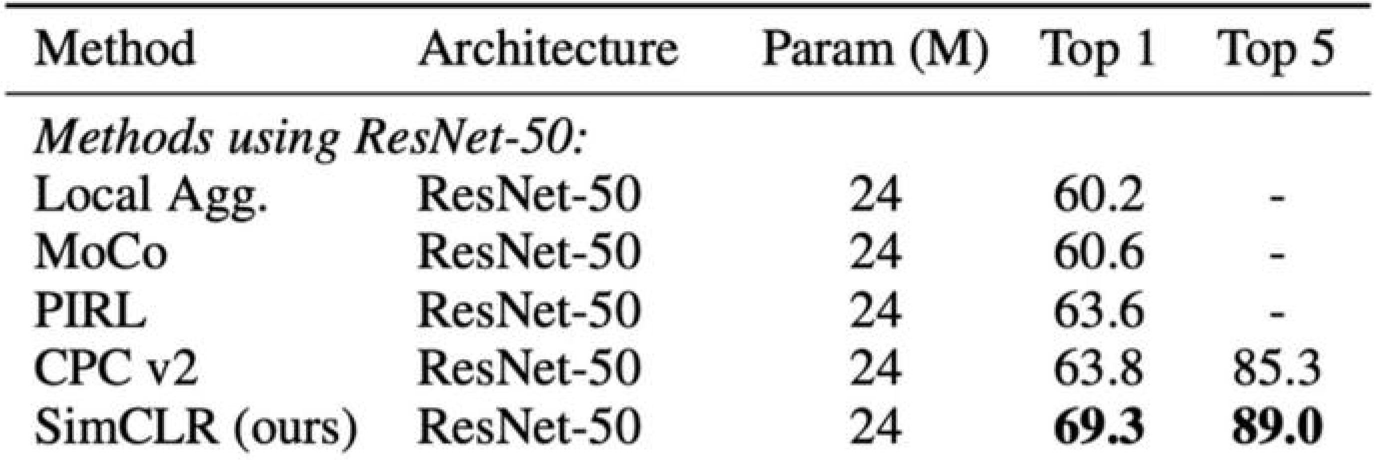


Similar images are attracted to each other. Source: [1]

Results

Once the network is fully trained, the MLP projection head is discarded and only the convolutional neural network is used for evaluation. In their paper, the authors performed different evaluations:

First, they measure the performance of SimCLR as a linear classifier on the ImageNet dataset. Their results show **SimCLR performing all other self-supervised methods**.

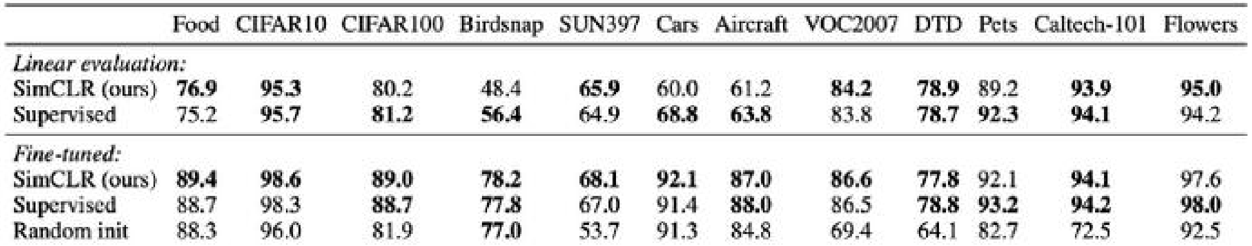


Method	Architecture	Param (M)	Top 1	Top 5
Methods using ResNet-50:				
Local Agg.	ResNet-50	24	60.2	-
MoCo	ResNet-50	24	60.6	-
PIRL	ResNet-50	24	63.6	-
CPC v2	ResNet-50	24	63.8	85.3
SimCLR (ours)	ResNet-50	24	**69.3**	**89.0**

Results of SimCLR and other self-supervised methods on ImageNet linear classification. Source: [2]

Please bear in mind that these results are not up-to-date anymore, since novel methods with better performance have come along. Feel free to read my other articles where I go over other self-supervised pre-training models.

Second, they evaluated the **performance of SimCLR on different image datasets versus training the same ResNet with labels**, i.e., with a supervised learning. Again, **SimCLR performs very well, beating the supervised training method on many datasets.** In the same table, they also looked at results from fine-tuning the self-supervised model with labeled data. In this row, they show that **SimCLR outperforms the supervised training approach on almost all datasets.**



	Food	CIFAR10	CIFAR100	Birdsnap	SUN397	Cars	Aircraft	VOC2007	DTD	Pets	Caltech-101	Flowers
Linear evaluation:												
SimCLR (ours)	**76.9**	**95.3**	80.2	48.4	**65.9**	60.0	61.2	**84.2**	**78.9**	89.2	**93.9**	**95.0**
Supervised	75.2	**95.7**	**81.2**	**56.4**	64.9	**68.8**	**63.8**	83.8	**78.7**	**92.3**	**94.1**	94.2
Fine-tuned:												
SimCLR (ours)	**89.4**	**98.6**	**89.0**	**78.2**	**68.1**	**92.1**	**87.0**	**86.6**	**77.8**	92.1	**94.1**	97.6
Supervised	88.7	98.3	**88.7**	**77.8**	67.0	91.4	**88.0**	86.5	**78.8**	**93.2**	**94.2**	**98.0**
Random init	88.3	96.0	81.9	**77.0**	53.7	91.3	84.8	69.4	64.1	82.7	72.5	92.5

Evaluation of SimCLR vs. supervised training of the ResNet. Source: [2]

Wrapping it up

In this article, you have learned about SimCLR, a paper that is one of the most popular self-supervised frameworks with a simple concept and promising results. SimCLR is constantly improved and there is even a second version of this architecture. While I hope this story gave you a good first insight into the paper, there is still so much more to discover. Therefore, I would encourage you to read the paper yourself, even if you are new to the field. You'll have to start somewhere ;)

If you are interested in more details on the method presented in the paper, feel free to drop me a message on Twitter, my account is linked on my Medium profile.

I hope you've enjoyed this paper explanation. If you have any comments on the article or if you see any errors, feel free to leave a comment.

And last but not least, if you would like to dive deeper in the field of advanced computer vision, consider becoming a follower of mine. I try to post a story once a week and and keep you and anyone else interested up-to-date on what's new in computer vision research!

. . .

References:

[1] SimCLR GitHub Implementation: https://github.com/google-research/simclr

[2] Chen, Ting, et al. "A simple framework for contrastive learning of visual representations." *International conference on machine learning*. PMLR, 2020. https://arxiv.org/pdf/2002.05709.pdf

97 2