

# Self-Supervised meets Active-Learning

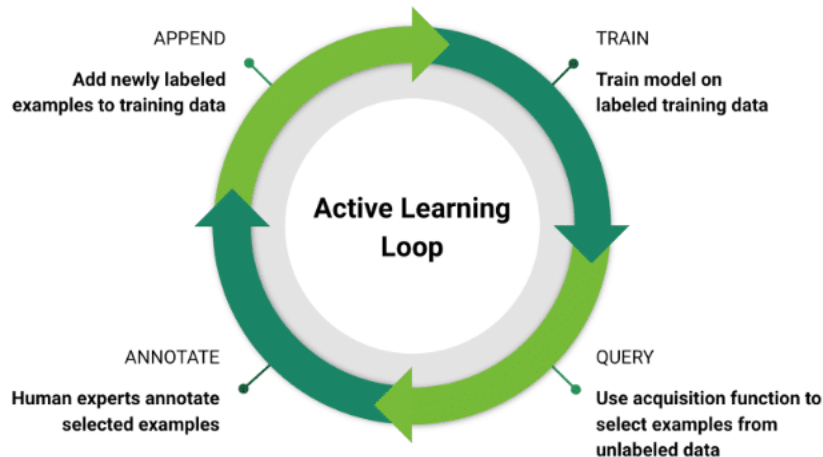
Benjamin Barda, Francesco Danese, Alessandro Vecchi

March 6, 2023

## Introduction

While Data is getting generated at an ever increasing speed and can be retrieved with not much effort, making this data useful and usable is still not a trivial task at all. One of the main bottlenecks is the need of vast amount of labeled data, which, for a high quality dataset, needs to be manually labeled with care, a process that requires a lot of man-hours to complete. This is the fundamental problem that researchers in the field of Active Learning are trying to answer. How can we get competitive performance with less data?

Many approaches are possible and we invite to refer to this repository for a comprehensive list of publications on the subject. The core idea of Active Learning is to get more out of the human in the loop. But how can we do this? Clearly is not possible to force someone to annotate faster, but what we can do is make sure that every sample that he labels have as much impact as possible.



The process is relatively simple from an overview perspective. Starting from an existing dataset of unlabeled data we select via an acquisition function the *most informative* samples to be presented to the oracle (a human expert for example) for he to annotate them and add them to the labeled dataset. We then train the model on the labeled data. We repeat this process until exhausting the labelling budget (time, money, ecc..).

## Active Learning: Not only one way

There are three main approaches to the problem :

- Stream based selective sampling
- Pool-Based sampling
- Membership query synthesis

In this project we investigate Pool-Based sampling but as an overview of the various methods this article might be a good starting point.

### Our approach

Our approach can be divided in two main steps:

- Pretrain
- Active learning loop

### Pretrain

This is a crucial part of the method proposed, in fact the quality of the features learned during this phase have a great impact on the final performance, so we put great consideration in selecting the appropriate pretext task.

The one thing that remained constant during all of our experiments was the choice of the architecture. We decided to use a Residual Network for two reasons:

1. Low number of learnable parameters
2. Proved history of great performance on CIFAR10, the dataset we chose for this project.

During the first phase of the project we considered using the non contrastive task described in SimSiam. Even though the quality of the features were excellent we faced the problem of limited resources in terms of computing and time, and since to have an effective pretrain we would need to run at least 800 epochs we decided to discard this option.

We then resorted to a simpler task, more specifically rotation prediction, as described in the RotNet paper. The task is pretty straight forward. Given an unlabeled image, we rotate it by 90, 180 and 270 degrees, associating to each rotation a label (0 for no rotation, 1 for 90deg rotation, etc). We then train a classifier over the rotated images.

The idea is that the network is forced to focus on important features of the image in order to recognize the rotation thus learning useful features representations. The big advantage over the SimSiam approach is the relative low number of epochs it needs. In fact around 100 epochs are needed to have an effective pretrain, allowing us to complete the training in a single colab session in less than 2 hours.