# Report

## 1. Dataset

### 1.1. Choice of the project

For this task I decided to take some subset of pandas library, because this is such a broad library and having a model that can help with finding a needed function would be very helpful

### 1.2. Creating dataset

Pandas is a huge library, so I want to take only the core functionality: manipulations with Series and DataFrame, etc. This is located in `pandas/core` folder.

To parse the functions I used ast python library. For each found function I create an instance in json, storing needed information, such as docstring, function name, code of the function, start and end lines, and path.

## 2. Fine-tuning

### 2.1. Choosing a model

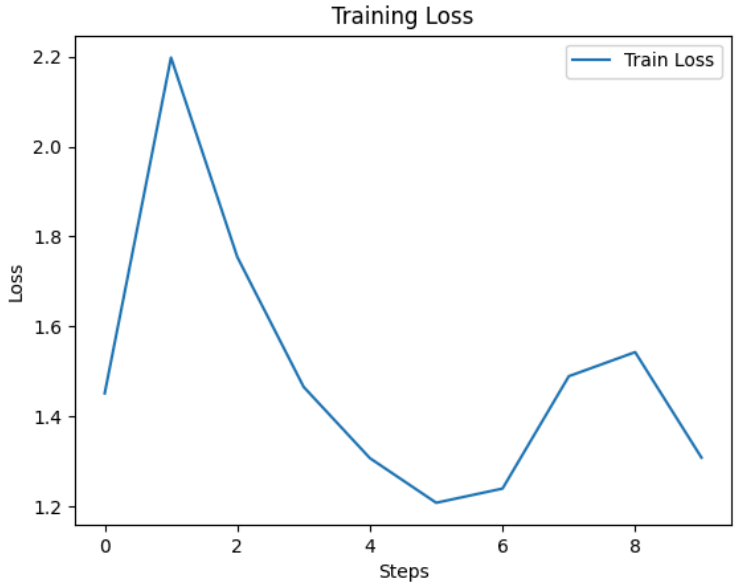
To fine tune the model, I'm going to use some pretrained code generation model, in particular Salesforce/codegen-350M-mono. This is the smallest model I was able to find, but unfortunately even with such model I've ran into issues with training it.

### 2.2. Training

So, when trying to train it locally I obviously didn't succeed, because my laptop does not have GPU. So I switched to Google Colab, in order to be able to use GPU, but then I ran into the issue that CUDA was out of memory, so this option didn't work either

So, I had to drastically decrease the amount of samples in the dataset, and some of the training arguments, like `max_steps = 50`, and only then I could run the code locally

## 3. Results



Looking at this learning curve we can’t tell much about the model, simply because we trained it on a really small dataset and with hyperparameters that are far from being optimal, just due to the hardware that I have. Still, we can see that there is a trend that loss on average decreases, so I can suppose that by increasing `max_steps = 50` and allowing a broader dataset will really increase the performance

P.S. The steps values are from 0 to 50, with logging_step = 5.