For context, I am currently working on an application of GPT for college admissions essay writing. A project that I can derive from this, as I've discussed with Pranav, is to investigate fine-tuning GPT for better responses. Although Pranav mentioned that this is still a good project to work on as long as I keep the discussion scientific, it is difficult to discuss the project in the terms of the MNIST example. I'll discuss what I believe are the scientific processes I would like to take below.

There are multiple models that OpenAI allows users to access. The most famous of which is Davinci-03. However, although it is the best performing compared to the other models, it is also the slowest and most expensive. What is possible though, is to identify a specific application of text generation, and fine-tune a "worse" model (such as Curie or Babbage) and train it to mimic the behavior and results of the better model. This would be cheaper and faster, **but there is a lack of knowledge on this cost-benefit analysis** ("fine-tuning Curie with *x* examples results in performance *y%* of Davinci and costs *z%* of it"). I want to fill this gap.

Dataset
We can use the Davinci-03 API to create {prompt, response} pairs that could be used to finetune Curie and validate Curie's responses. There is no limit, and no real time constraint as the prompts and responses we will be validating will be at most 50 words long. I will probably stick to the action "make this one sentence sound better."

"Model" Input
Curie can take in fine-tune training (prompt:response pairs) and then a single prompt.

"Model" Output
A single response.

"Model" Architecture
There are parameters we can tweak on both Davinci and Curie: Temperature, Top P, Frequency, Presence, top P (all these refer to randomness), or max response length.

Evaluation Metric

Similarity
I will score the responses of a finetune Curie against Davinci's responses ("validation set") and see how similar they are. I can use Dandelion as one way to compare the responses via semantics, or other models that can compare text.

There's also I think valuable insights in discussing the costs of training, how much finetuning was needed to achieve specific results, and providing the cost-benefit analysis of all of this.