Deep Dive into **Active Learning**

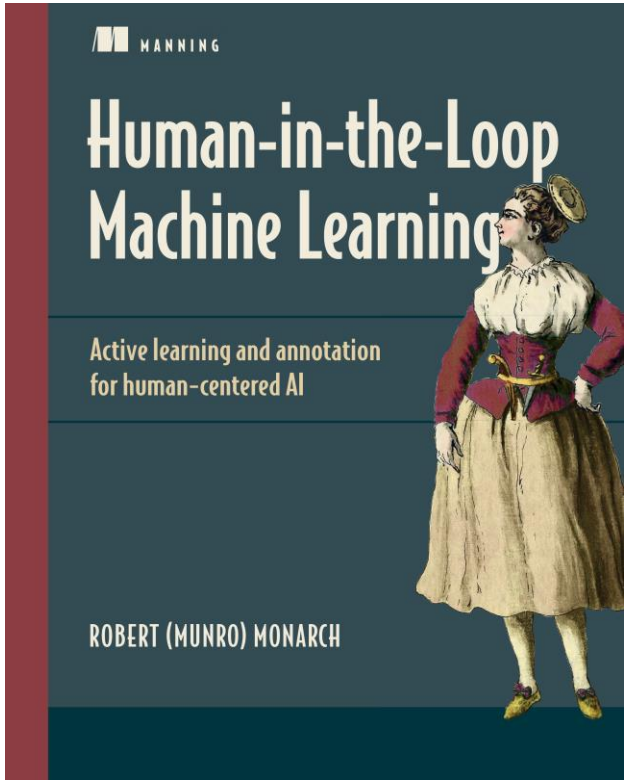
Train neural networks more efficiently with less data

Christian Normand

Roadmap

- Introduce problem statement
- Define active learning
- Review experiment
- Discuss results
- Reflect on lessons learned

Read this book



Source: Manning Publications Co.

Human-in-the-Loop Machine Learning
By Robert (Munro) Monarch.

This book provided the inspiration for this project. It's all about active learning, and more generally, how humans and machine learning algorithms interact.

Check it out here!

Problem statement:

Can active learning techniques be used to reduce the amount of training images required for a convolutional neural network (CNN) to reach target accuracy?

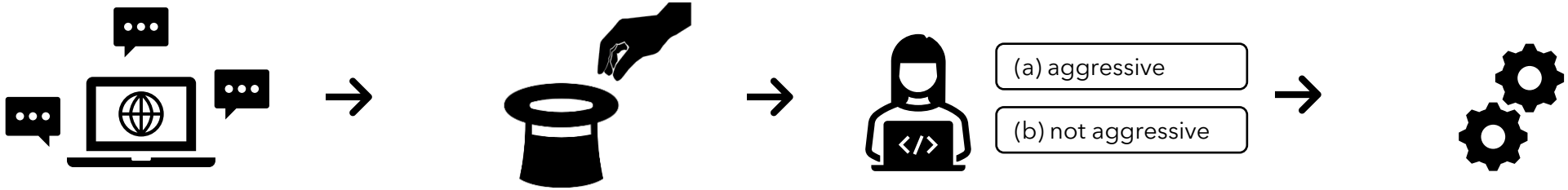
Define Active Learning

Active learning: What is it?

- Supervised learning requires **labeled data**.
- Active learning comprises a set of techniques for selecting the data you will use to train your model.
- Allows you to reach target accuracy with fewer labeled data points.
- Can be used to periodically retrain models to adapt to changing data.

Example: Consider a blogging website that's trying to detect aggression in its comment section.

A **traditional approach** to labeling data might look like this:



Data scientists sample comments from a large collection of blog posts

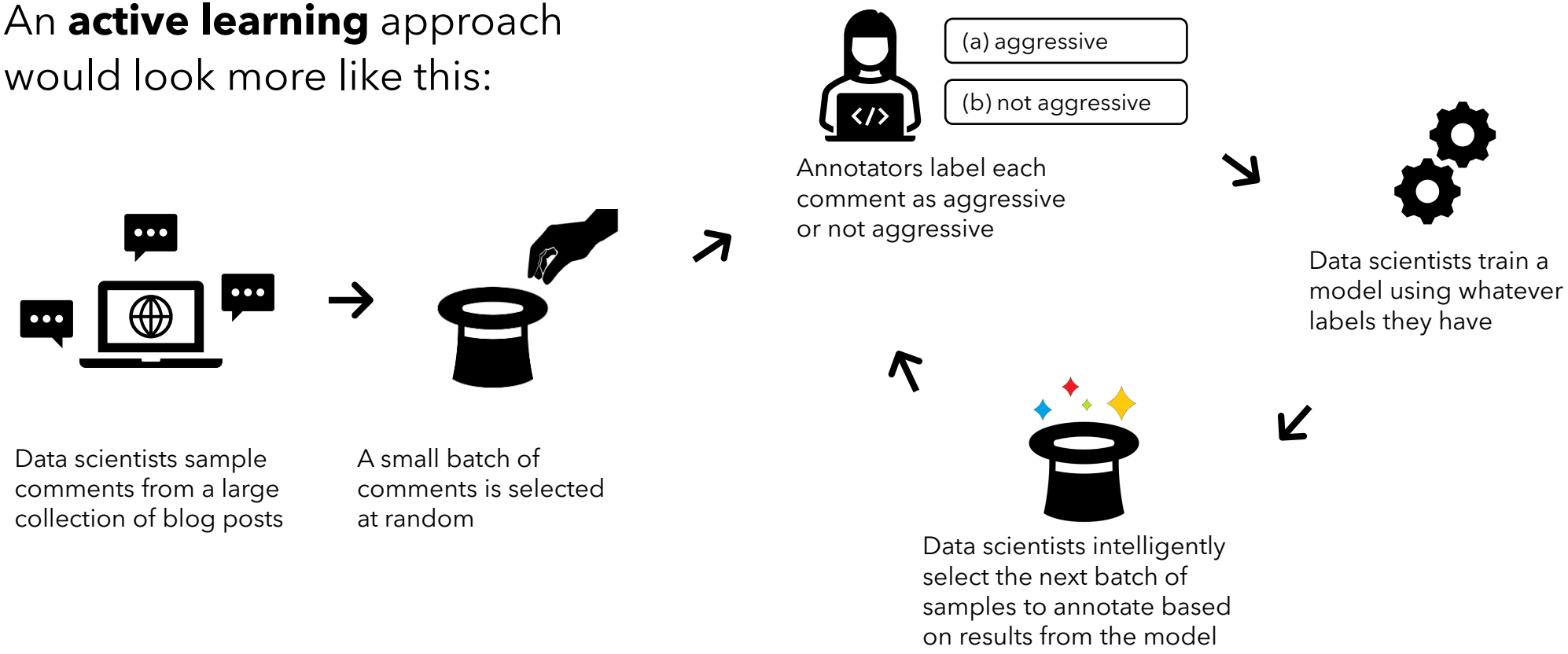
Comments are selected at random

Annotators label each comment as aggressive or not aggressive

Data scientists train a model using the labeled data

Example: Consider a blogging website that's trying to detect aggression in its comment section.

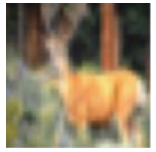
An **active learning** approach would look more like this:



Active learning techniques

Uncertainty sampling

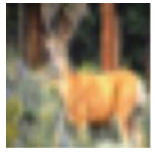
Sampling data that's most confusing for the model.

Input	Model says:
	91% sure that's a deer.
	99% sure that's a car.
	51% sure that's a cat… but maybe it's another deer.

Active learning techniques

Uncertainty sampling

Sampling data that's most confusing for the model.

Input	Model says:
	91% sure that's a deer.
	99% sure that's a car.
sample this → 	51% sure that's a cat… but maybe it's another deer.

Active learning techniques

Uncertainty sampling

Sampling data that's most confusing for the model.

Input	Model says:
	91% sure that's a deer.
	99% sure that's a car.
	51% sure that's a cat… but maybe it's another deer.

Diversity sampling

Sampling data that's the most different from what the model has already seen.

Input	Model says:
	That's a boat!
	That's another boat!!!
	That's a frog… but I haven't seen many frogs before

Active learning techniques

Uncertainty sampling

Sampling data that's most confusing for the model.

Input	Model says:
	91% sure that's a deer.
	99% sure that's a car.
	51% sure that's a cat… but maybe it's another deer.

Diversity sampling

Sampling data that's the most different from what the model has already seen.

Input	Model says:
	That's a boat!
	That's another boat!!!
	That's a frog… but I haven't seen many frogs before

sample it.

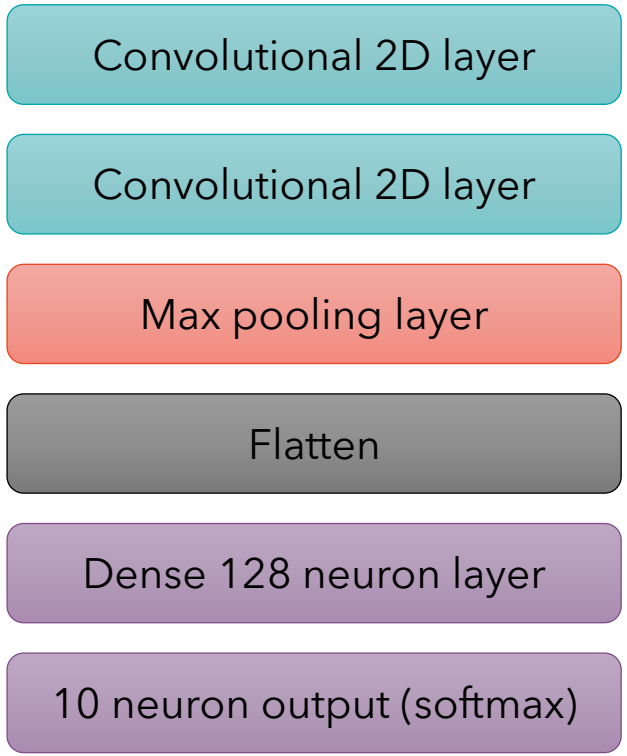
Review experiment

The CIFAR-10 Dataset

- 60,000 tiny images evenly split into ten categories
- I used 49,000 for training, 1,000 for validation, and 10,000 for testing.
- This is a commonly used dataset for image classification. There are plenty of models out there that have achieved ~90% accuracy.
- Get the dataset: https://www.cs.toronto.edu/~kriz/cifar.html

The model

- I used a convolutional neural network with one hidden layer.
- The model maxed out at 50% accuracy.
- I chose a simpler model to see if it would be faster to train, acknowledging it could impact accuracy.



Methodology

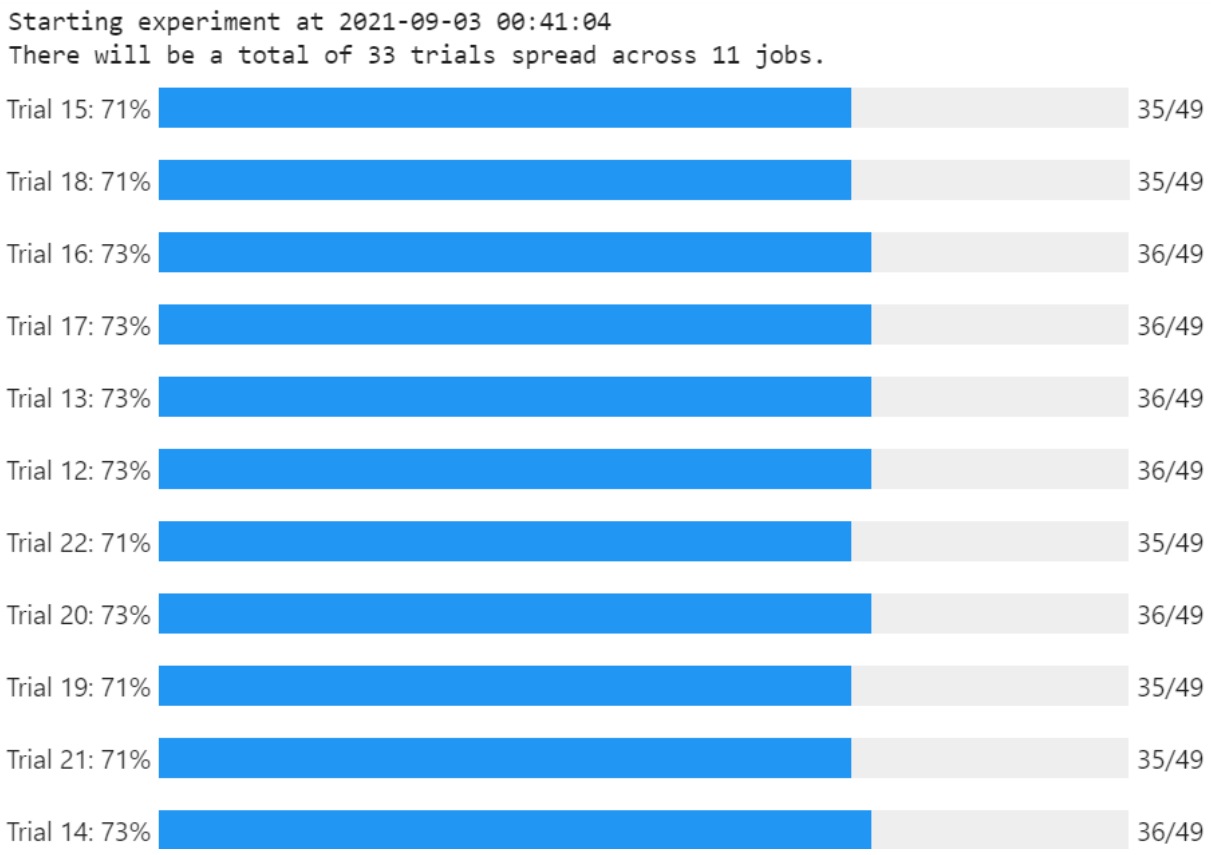
- Try different percentages of uncertainty sampling [0%, 10%, 20%... 100%], and see how that affects the time it takes for the model to reach 50% accuracy.
- Diversity sampling would have been interesting to implement as well, but that'll be a future project.
- Start each trial by training the model on a randomly selected batch of 1,000 images.
- After the first training batch, use the uncertainty sampling protocol for that trial to add another 1,000 images to the training set.
- I used *margin of confidence* to rank images by their level of uncertainty.
- Retrain the model and repeat until the training set includes all 49,000 training images.
- Run three trials for each sampling protocol.

Concern: Speed

My laptop

- The experiment consisted of 1,600+ CNNs.
- Using a GPU or TPU would have been ideal but was not an option.

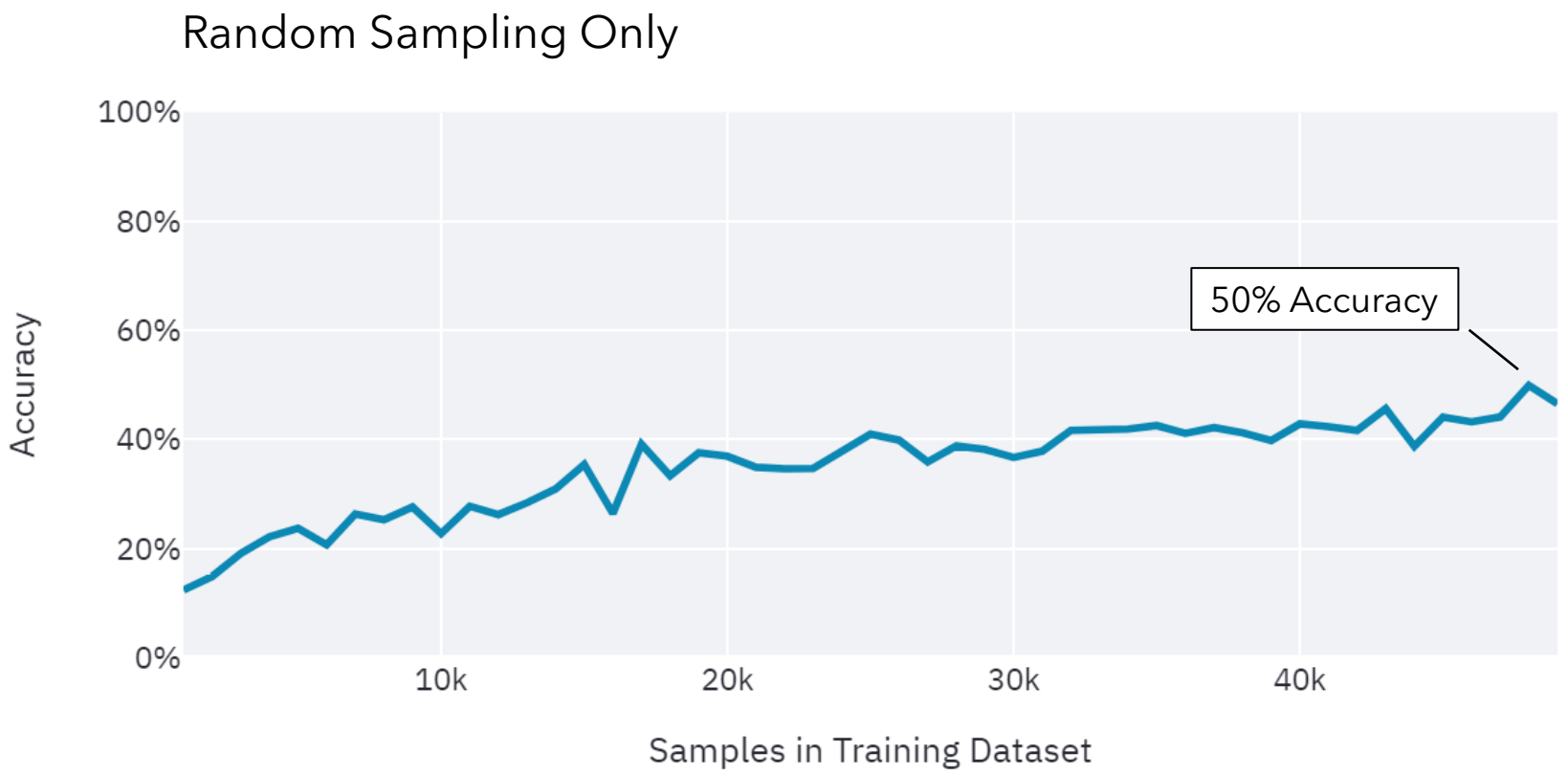
Solution: Multiprocessing



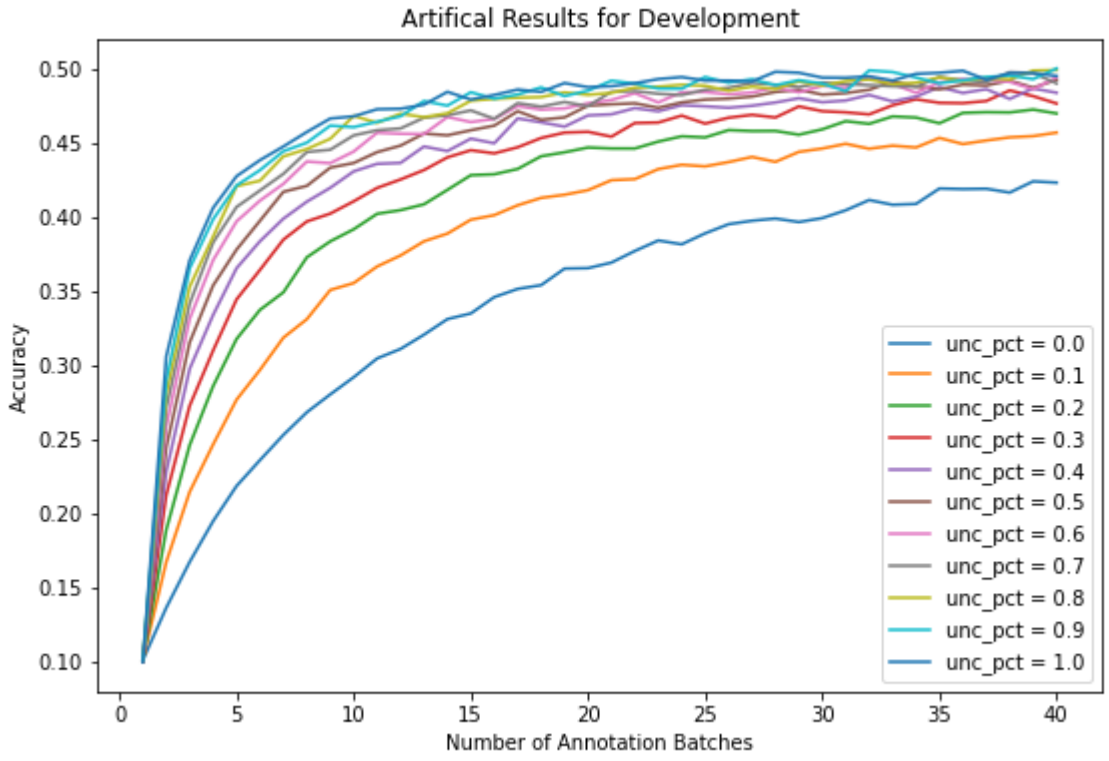
I refactored my code using the multiprocessing package and ran the experiment in a Google AI notebook with 16 CPUs

Discuss results

Baseline

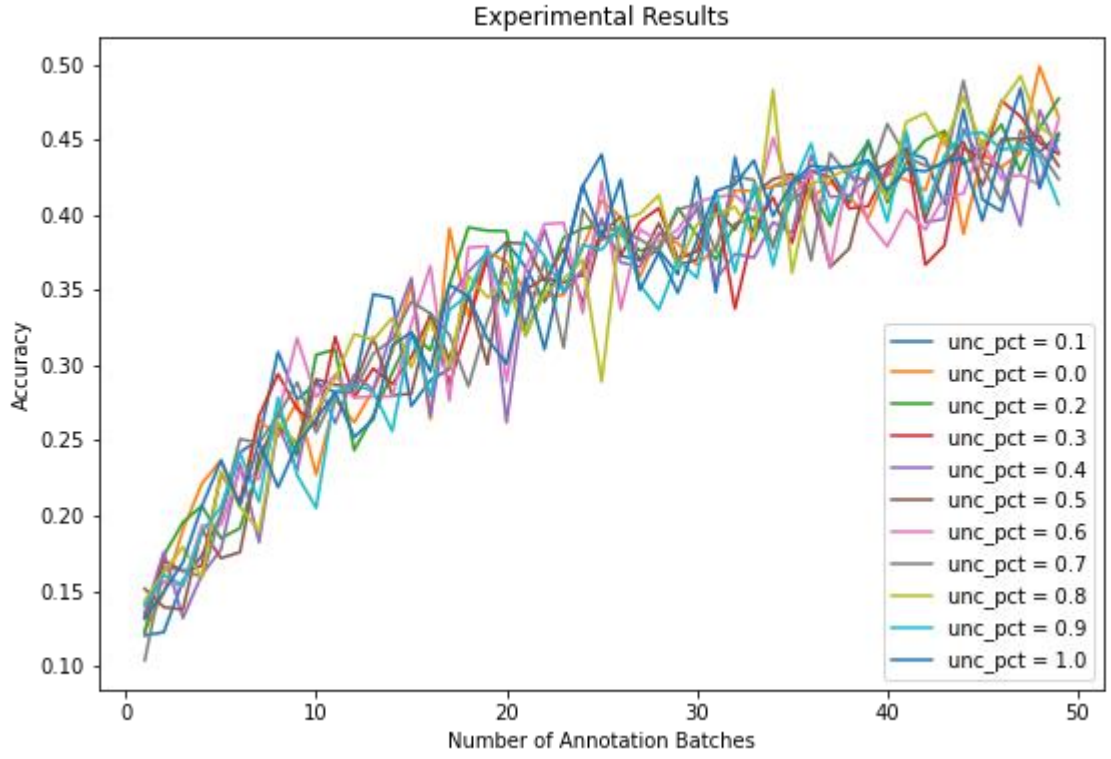

Random Sampling Only
100%
80%
60%
40%
20%
0%
Accuracy
50% Accuracy
10k
20k
30k
40k
Samples in Training Dataset

Results…?



- I derived a formula to generate dummy results so I could start working on charts while the experiment was running.
- This is what I hoped the results would look like.

Results



- I derived a formula to generate dummy results so I could start working on charts while the experiment was running.
- This is what I hoped the results would look like.
- **This is what the results actually looked like.**

Reflect on lessons learned

Why the unexpected outcome?

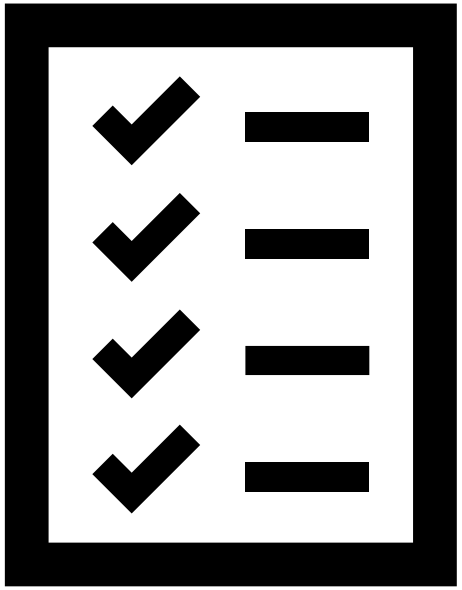
First guess: The model wasn't strong enough.

Second guess: A different method of uncertainty sampling might have led to better performance.

Third guess: Incorporating diversity sampling might have led to better performance.

Accomplishments

- Created an object-oriented framework for running iterative experiments using any type of model.
- Learned how to use the multiprocessing package and Google AI notebooks.
- Continuing to learn about active learning, a very useful technique in industry.



Check out the Streamlit app.

Thank you!