Erwin Cheng

ITAI 1378

September 25, 2025

Journal / Summary

Patricia Mcmanus

Working on this exercise felt like I was teaching a computer to see, which is pretty cool. This journal is all about my experience building a Convolutional Neural Network (CNN) to tackle this challenge, what I learned along the way, and how it stacks up against the other neural networks we've explored.

The structure was way different from the traditional neural networks we used before. The structure was way different from the traditional neural networks we used before because instead of just flattening the image into one giant, messy line of pixels, the CNN is much smarter. It uses a couple of "convolutional blocks" first, which act like a series of special magnifying glasses. The convolutional layers scan the image for basic patterns—like edges and corners—using the same filter across the whole image to save a ton of work. After that, the pooling layers come in to summarize the most important bits and shrink the image down. This makes everything run faster and helps the network not sweat the small stuff. Only after all this clever feature-spotting does the data get flattened and sent to the regular "fully-connected" layers for the final vote: chihuahua or muffin?

After letting the model train for 15 rounds (epochs), the results were pretty solid! It managed to get a validation accuracy of about 91%, which I was really happy with. The training accuracy was a little higher at 96%, which hints that it might have been "memorizing" the training photos just a tiny bit. But honestly, the most fun part was looking at the mistakes it made. The model got tripped up by photos of light-brown chihuahuas curled up into a ball, especially when you couldn't see their faces. The network seemed to focus on the overall color and texture, and in those cases, it just screamed "muffin!" The best, though, was when blueberry muffins were mistaken for chihuahuas. The model would see the cluster of dark blueberries and think it was looking at the eyes and nose of a tiny dog, which shows how a CNN can sometimes fixate on small details and miss the bigger picture.

Putting this new CNN model head-to-head with the traditional MLP networks we've used before isn't even a fair fight when it comes to images. The CNN is just awesome for these kinds of tasks because it's built from the ground up to understand spatial patterns, so it just *gets* images in a way MLPs can't. An old-school MLP starts by flattening the image, which throws away all the important info about what pixels are next to each other. Because of its smarter design, the CNN is also way faster to train. It smartly shares parameters, it has much less to learn and gets to a good answer a lot quicker. An MLP, on the other hand, would be painfully slow, with millions of parameters that would take forever to train. Finally, the MLP is a total memorization machine; with so many parameters, it's almost guaranteed to just memorize the training photos instead of actually learning. While you still have to watch out for overfitting with a CNN, it's naturally much less of a problem.

I did encounter some challenges while working on this activity. My first big hurdle was getting the settings right. I initially set the "learning rate" way too high, and the model's learning process was all over the place. To fix this, I dialed the learning rate back to be more conservative and slowly bumped up the number of training epochs, watching the learning curves until the loss started to go down smoothly. I also noticed that slight overfitting I mentioned earlier. The lab notebook had a great suggestion: data augmentation. If I were to continue, my next step would be to add random transformations to the training images—flipping them horizontally or rotating them slightly. That way, the model would be forced to learn what a "chihuahua" *really* is, not just what the specific chihuahuas in the training photos look like.

As much fun as telling dogs from baked goods is, the technology behind it is the same stuff that powers some seriously important real-world tools. It's wild to think that the same principles could be used for medical imaging to help doctors spot cancer cells, for self-driving cars to act as their "eyes" on the road, for factory robots to check products for tiny defects, or even in farming to analyze pictures of crops for diseases.

This project also got me thinking about some heavier topics. This kind of technology is incredibly powerful, and it's important to be careful about how it's used. The biggest concern is biased data; if you train a model on a dataset that isn't diverse, it's going to make biased decisions, which could have huge consequences in the real world. There's also the potential for misuse, like in mass surveillance, and tricky ethical questions about who's to blame if an AI model makes a mistake. It really hammered home that building these models isn't just about code; it's about being responsible for what we create.

References

Stanford University. (n.d.). *CS231n: Convolutional Neural Networks for Visual Recognition*. Retrieved from http://cs231n.stanford.edu/