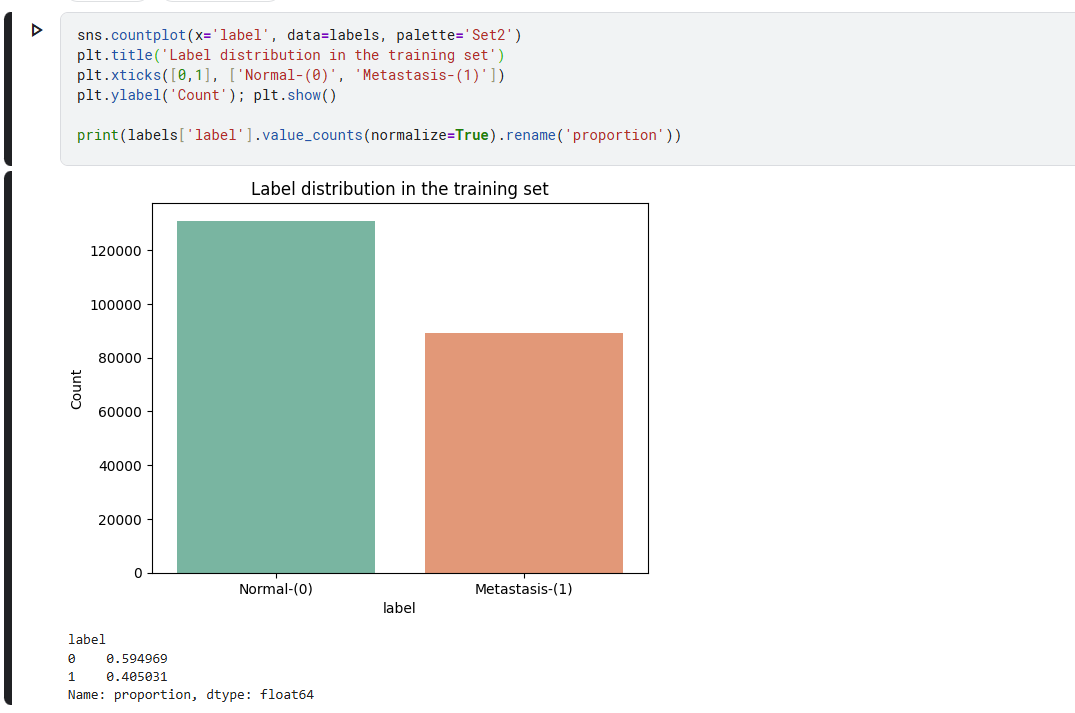
A close up of a text

AI-generated content may be incorrect.

A close-up of a computer screen

AI-generated content may be incorrect.

**Label Balance**



* **Comments:** Almost 60% percent of our dataset is normal, and 40% of them has Metastasis(cancer). Data is slightly imbalanced.

**Checking Image quality and pixel Size**

A screenshot of a computer program

AI-generated content may be incorrect.

* **Comments:** Here I checked pixel size of the portion of the train dataset, and it passed the test, data is clean sizes are correct.

**Creating Visual and Observing Image Samples**

A computer screen shot of a code

AI-generated content may be incorrect.

A collage of cells

AI-generated content may be incorrect.

A collage of cells with different cells

AI-generated content may be incorrect.

**Comments:** I couldn’t identify the cancer with bare eyes; it was supposed to be in the center of image.

**DModel Architecture (25 pts)**

A screenshot of a computer program

AI-generated content may be incorrect.

**DATA PREPROCESSING**

A screen shot of a computer code

AI-generated content may be incorrect.

A screenshot of a computer code

AI-generated content may be incorrect.

**MODEL ARCHITECTURES**

**Simple CNN**A screen shot of a computer code

AI-generated content may be incorrect.

**Enhanced CNN Model**

A screenshot of a computer program

AI-generated content may be incorrect.

**Transfer Learning Model**   
A computer screen shot of a program code

AI-generated content may be incorrect.

**Model Architecture Design and Reasoning**

**Problem Analysis**

For histopathologic cancer detection, I need to classify 96x96 pixel tissue images as cancer or non-cancer. This is a binary classification problem.

**Architecture Selection Strategy**

I chose to compare three different approaches to understand which works best for medical imaging:

1. **Simple CNN** - Basic baseline to test if the problem is learnable
2. **Enhanced CNN** - Improved version with modern techniques
3. **ResNet50 Transfer Learning** - Pre-trained model to leverage existing knowledge

**Architecture 1: Simple CNN (Baseline)**

**Design Reasoning:**

* Started with a basic 3-layer CNN to establish baseline performance
* Used progressive layers (32→64→128) to capture features from simple to complex
* Included MaxPooling for spatial reduction and computational efficiency
* GlobalAveragePooling instead of flatten to reduce overfitting risk

**Why This Architecture:**

I chose a simple and efficient baseline CNN model as a starting point to test whether basic architectures can learn cancer patterns and serve as a foundation for future improvements.

**Architecture 2: Enhanced CNN (Improved Design)**

**Design Reasoning:**

* Added BatchNormalization for training stability and faster convergence
* Included Dropout (0.25 and 0.5) to prevent overfitting on medical data
* increasing dropout rates: lower for conv layers, higher for dense layers

**Why These Improvements:**

I included BatchNorm, Dropout, and double convolution layers to handle variability in medical images, prevent overfitting to specific tissue samples, and better capture complex cellular patterns for fine-grained medical analysis.

**Expected Performance:**

It should outperform simple CNN due to better regularization and feature extraction capabilities.

**Architecture 3: ResNet50 Transfer Learning**

**Design Reasoning:**

* Used pre-trained ResNet50 from ImageNet as feature extractor
* Froze base weights to prevent overfitting with limited medical data
* Added custom classification head with dropout for medical domain
* Leveraged 50-layer depth for complex pattern recognition

**Why Transfer Learning:**

I used transfer learning with ResNet50 because its pre-trained features offer rich visual representations, it's a proven architecture for image tasks, it's more efficient than training from scratch, and its 50 layers can model complex tissue structures.

**Results and Analysis (35 pts)**

**MODEL TRAINING AND COMPARISON**

**TRAINING SIMPLE CNN**

A screenshot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**TRAINING ENHANCED CNN**

A computer code with many text

AI-generated content may be incorrect.

A grid of lines with small dots

AI-generated content may be incorrect.

**TRAINING RESNET50 (TRANSFER LEARNING)**

A computer screen shot of a computer code

AI-generated content may be incorrect.

A white background with black and white lines

AI-generated content may be incorrect.

INITIAL MODEL PERFORMANCE COMPARISON

A screenshot of a computer program

AI-generated content may be incorrect.

**Model Comparison Visuals**

**Model Performance Summary**

 I trained three different models for 10 epochs each: a simple CNN, an enhanced CNN with added layers and regularization, and a ResNet50-based transfer learning model. After evaluating each model on both the training and validation sets, I compared their AUC metrics.

The enhanced CNN consistently outperformed the others. It achieved the highest validation AUC, showing both strong generalization. The simple CNN performed reasonably well, but not as strongly. Suprasingly the ResNet50 model underperformed compared to both CNNs.

**Why did ResNet50 fail?**

* Medical histopathology images are very different from ImageNet's natural images
* Pre-trained features (animals, objects, scenes) don't transfer well to medical images, I should have only frezee low level layers which can be transfered more easily. this shows the importance of domain-specific features in medical imaging

**HYPERPARAMETER TUNING**

**Tunning Learning Rate**

A screenshot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**Tunning Optimizers**

A screenshot of a computer program

AI-generated content may be incorrect.

A screenshot of a graph

AI-generated content may be incorrect.

**Hyperparameter Tuning Conclusion**I am surprised that RMSprop actually beat Adam. I mean, Adam is usually performs better. But I guess this just shows that different problems might need different approaches. Also, the fact that a 0.01 learning rate worked almost as well as 0.001. The 0.0001 result honestly makes sense though, probably just too slow to learn much in only 10 epochs.

Overall, I think this tuning experiment confirmed that having a good model architecture like my Enhanced CNN, matters *way* more than perfectly tuning every little parameter. The differences between optimizers and learning rates were small.

**General Conclusion**I went into this thinking bigger models and transfer learning stuff would *obviously* be better, but I’ve realized domain knowledge and design actually matter a lot. Like, our 140K parameter Enhanced CNN totally outperformed the 23.8M parameter ResNet50, which I didn’t really expect. It could have perform better, if I was able train longer and with more data. The problem was I had limited resources and this training took very long time.

Also, I’m glad I did a proper comparison instead of just picking one and hoping it worked. Seeing ResNet50 kinda fail was actually more helpful than if it had just worked like I assumed it would. I guess I didn’t really think about whether ImageNet features even made sense for our problem now I know that’s something I need to check going forward.

The tuning part was helpful too, though maybe not too much difference, it mostly showed that the default settings we used were already pretty solid. I guess that kind of reinforces that sometimes the defaults are there for a reason.