MRI Classification Project with Resnet vs. Hybrid Model

Piotr Brozek

1 Introduction

The main objective of this project was to develop an automated machine learning system that not only detects the presence of a brain tumor in MRI scans but also accurately classifies its type—glioma, meningioma, pituitary tumor, or healthy tissue—using a hybrid architecture rather than relying solely on a conventional, end-to-end neural network. Specifically, I used a pretrained ResNet-18 as a fixed feature extractor (freezing all but its final residual block) and paired its high-level representations with an AdaBoost ensemble of decision trees.

The main motivation behind this work was to understand if alternative approaches by combining different models such as a neural network and decision tree could beat conventional approaches such as neural networks. The original idea behind this project was that since neural networks usually tend to perform well on large datasets by learning massive amounts of patterns, but tend to overfit on smaller datasets, and decision trees can learn better with smaller amounts of data, could combining both of these models result in a better performing model than a conventional neural network?

The results showed that when we trained on small amounts of the MRI scans, the hybrid ResNet+AdaBoost model did better than the pure ResNet. As I moved to using all of the scans, the end-to-end ResNet caught up and eventually outperformed the hybrid model, which tells us that deep nets still have the edge when you have lots of data. By plotting accuracy against the size of the training set, I could clearly see the exact point where one model pulled ahead of the other. In other words, this means that in clinics or settings where it's hard or expensive to collect and label hundreds of MRIs, the hybrid approach can serve as a simple, reliable alternative.

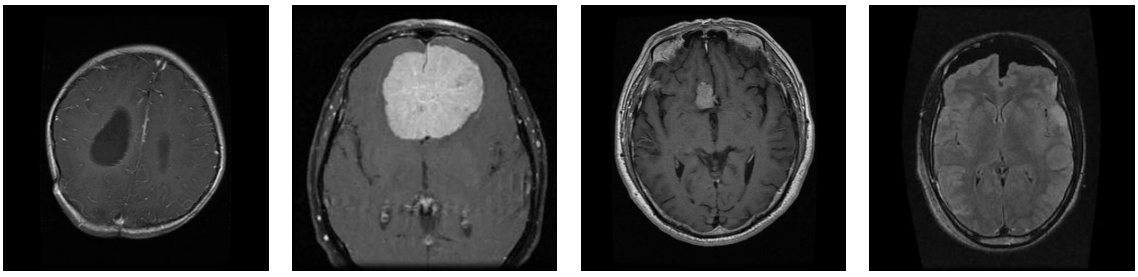

Figure 1: Brain tumor MRI examples: glioma, meningioma, pituitary tumor, and no tumor.

2 Methods/Case Study

The first step in conducting my study into hybrid models was to first set up a baseline accuracy. To set up a baseline, I first loaded the MRI scans from a public dataset, and resized each image to 224x224 pixels, turned it into a tensor, and normalized it using ImageNet's mean and standard deviation - the measurements used when Resnet was trained on ImageNet. I then created dataloaders with a batch size of 64—shuffling the training data and using four worker processes with pinned memory for faster transfers. Next, I grabbed a pretrained ResNet-18, swapped out its final fully connected layer for a new one with four outputs (one per class), and moved the model to the GPU if it was available. I chose the Adam optimizer (learning rate = 0.001) to update only the new layer's weights and used cross-entropy loss as my criterion. Over 15 epochs, I trained the model, tracking loss and accuracy on the training set with a tqdm progress bar, then evaluated its accuracy on the validation set after each epoch. At the end, I plotted both training and validation accuracy curves to visualize how well the pure ResNet model learned over time. The model performed relatively well, achieving about an 82% validation accuracy.

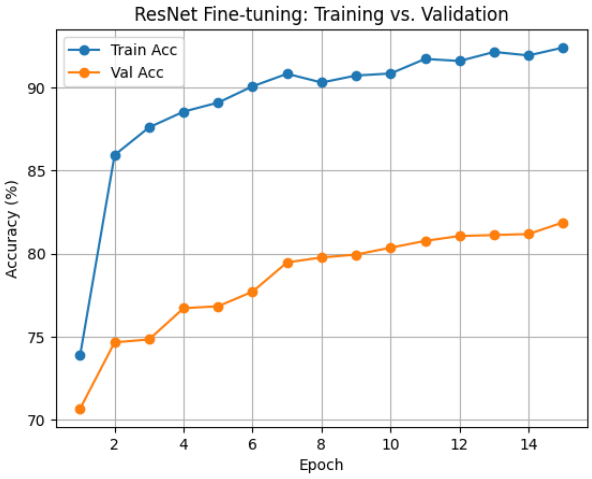

Figure 3: Performance of ResNet-18 Model

Next, I trained the ResNet-18 + AdaBoost model. I first took another ResNet-18 pretrained model, and removed its final classification layer, and used the remaining layers as a feature extractor. The approach to this ResNet-18 model for feature extraction was done similarly to the pure ResNet-18 model. Each MRI was resized to 224x224, normalized and converted to a tensor. The next thing necessary is the AdaBoost ensemble of decision trees. The AdaBoost portion is built with a depth of 3 with 75 n estimators, and a learning rate of 0.5. After extracting and flattening the feature vectors from each MRI scan, I standardized them to zero mean and unit variance using a StandardScaler fit on the training set. I then applied the same transform on the validation features. To evaluate consistency, I ran the AdaBoost training several times, each on a different random 80/20 split of the training data. In each run, the depth-3, 75 n estimator ensemble was fit on 80% of the samples and then used to predict both its training subset and

the held-out 20%, reading accuracy on each. The model did well achieving the highest accuracy at 85% validation accuracy.

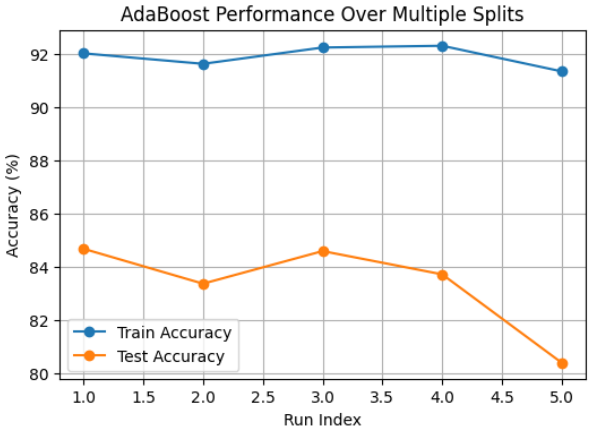

Figure 4: ResNet-18 + AdaBoost Hybrid Model Performance

To test how much the amount of training data mattered, I had to source an additional dataset and merge both datasets to create a larger set of MRI images. The final combined dataset contained 10,297 MRI scans, which was approximately 47% larger than the original dataset of around 7,000 scans.

I then wrote a looped program to train both models—ResNet-18 and the ResNet+AdaBoost hybrid—on different fractions of the data: 1%, 5%, 10%, 50%, and 100%. For each data proportion, I made sure the tumor type ratios (glioma, meningioma, pituitary tumor, and no tumor) stayed the same as in the full dataset. On each slice, I trained the hybrid model using frozen ResNet layers to extract features and passed them into AdaBoost, and trained the pure ResNet model normally for 15 epochs.

Both models were tested on the same validation set each time, so the results were easy to compare. Since I had to rebuild and retrain both models from scratch for every data size, the process was very GPU-heavy and took a lot of time and computing power.

Even though it was time-consuming, it was important to run these tests to see exactly how the models behave as data size changes. It helped me find the point where the hybrid model worked better and where the pure ResNet started to take over. This kind of testing is useful because it's similar to real situations where people may only have a small amount of labeled data and need to know which model is best depending on what's available.

Fraction of Data	Hybrid Accuracy (%)	Pure ResNet Accuracy (%)
1%	64.52	53.90
5%	71.91	72.55
10%	75.19	72.43
50%	83.17	81.82
100%	83.75	84.57

Table 1: Validation accuracy of hybrid (ResNet+AdaBoost) vs. pure ResNet-18 models at varying training-set fractions.

3 Results and Discussion

I entered the experiment believing that the original dataset would be more than sufficient for a deep network to learn all relevant tumor patterns end to end and beat the hybrid model from the beginning, but I was surprised when the hybrid model ended up performing better than the pure ResNet model on the original-sized dataset. From the last experiment with varying data sizes, it appears that from 5% to 50%, the hybrid model outperformed the pure ResNet model. Most likely this occurred because the ResNet model was not able to learn all of the patterns from the limited amount of data. This is especially evident when 5% of data is learned, where the ResNet-18 model scores a 53.9% and the hybrid model scores a 64.52% - about 11% higher, a significant advantage over the pure ResNet-18 model.

The difference in performance when using smaller parts of the dataset supports the idea that AdaBoost, which is good at handling small amounts of data, works well when paired with a deep model like ResNet. Since ResNet was already trained on a huge dataset, it can pull out useful features from the MRI scans. AdaBoost then takes those features and focuses on the ones the model struggled with, learning from mistakes and improving through its set of small decision trees.

At 5% of the data, both models actually perform pretty closely (about 72% accuracy), which shows that even with a small amount of training, ResNet can learn some useful patterns. But when I increased the data to 10%, the hybrid model pulled ahead again, proving it's more reliable with limited data. Once I used the full dataset, though, the pure ResNet model started to do better, slightly beating the hybrid. This shows that when there's a lot of data, deep learning models can really shine because they're built to learn complex patterns.

One thing I noticed is that the hybrid model kind of "leveled off" in performance, staying around 83–84% accuracy even as I gave it more data. This is likely because AdaBoost with decision trees doesn't improve much after a certain point, while deep models like ResNet keep getting better as they're fed more examples.

In the end, this project showed that in situations where you don't have a ton of MRI scans—like in many hospitals or clinics—the hybrid model can actually be a better and simpler choice. It trains faster, is less likely to overfit, and still gives solid results. More importantly, collecting and labeling large medical datasets is not only time consuming but also expensive, often requiring specialized staff, such as radiologists or physicians, to review and annotate each

image. This significantly increases the cost and effort involved. So if someone doesn't have the resources to train a large neural network—or to curate a massive labeled dataset—combining a hybrid model is a smart, efficient, and reliable option.

4 Conclusion

In this project, I set out to see if combining two different types of models—ResNet and AdaBoost—could create a more balanced system for detecting and classifying brain tumors in MRI scans, especially when the amount of training data is small. I first built a baseline using a regular ResNet-18 model trained end-to-end, then built a hybrid model where ResNet was used only to extract features, and AdaBoost handled the classification.

What I learned is that hybrid models can actually perform better than deep learning models when data is limited. In fact, for most of the smaller data sizes I tested (like 1% to 50%), the ResNet+AdaBoost model had better accuracy. This means that in real-world medical situations—where collecting a large, labeled dataset is tough—the hybrid model can be a better, more practical option.

However, when I trained both models on the full dataset, the pure ResNet model ended up doing slightly better. So in cases where there is enough data and resources, deep learning still has the upper hand. I also observed that the hybrid model's performance tends to plateau around 83–84% accuracy, even as more training data is provided. This suggests that while AdaBoost can capitalize on good feature representations early on, it eventually hits a performance ceiling that deeper neural networks can surpass when given enough data.

Overall, this project taught me that hybrid models are not only valid, but in some cases, they can actually be the smarter choice—especially in fields like healthcare, where data is often limited and expensive to label due to the need for expert annotations from trained professionals.

If someone is working on a brand new problem where labeled data is scarce—or expensive to produce—then a hybrid approach offers a reliable way to get meaningful results. But if the problem is well-established and there is plenty of data available, then deep learning models trained end-to-end will likely be more powerful and accurate in the long run.

More broadly, I also learned that there is probably no single "best" model. The ideal approach really depends on the specific use case, the size and quality of the available dataset, the computational resources, and even the constraints or goals. Model selection should always be grounded in context.

Reference

TorchVision Contributors. (n.d.). ResNet-18 — Torchvision Main Documentation. Retrieved May 9, 2025, from
`https://pytorch.org/vision/main/models/generated/torchvision.models.resnet18.html`

PyTorch Contributors. (n.d.). ResNet — PyTorch Hub. Retrieved May 9, 2025, from
`https://pytorch.org/hub/pytorch_vision_resnet/`

Abhishek. (n.d.). Fine-tuning a pre-trained ResNet-18 model for image classification on custom dataset with PyTorch. Medium. Retrieved May 9, 2025, from https://medium.com/@imabhi1216/fine-tuning-a-pre-trained-resnet-18-model-for-image-classific

H2O.ai. (n.d.). AdaBoost — H2O 3.46.0.7 Documentation. Retrieved May 9, 2025, from https://docs.h2o.ai/h2o/latest-stable/h2o-docs/data-science/adaboost.html

DigitalOcean Community. (n.d.). Boost your models with AdaBoost explained. Retrieved May 9, 2025, from https://www.digitalocean.com/community/tutorials/adaboost-optimizer

Into Deep Learning. (2022, June 9). Use tqdm to monitor model training progress. Retrieved May 9, 2025, from https://colab.research.google.com/github/intodeeplearning/blog/blob/master/_notebooks/2022-06-09-use-tqdm-to-monitor-model-training-progress.ipynb