Progress Report

1. Introduction and problem statement

In Computer Vision (CV), Convolutional Neural Networks (CNNs) play a crucial role in processing image and video data, allowing machines to effectively understand and analyze visual information. Challenges include handling variability within classes, assessing transfer learning across domains, and tuning hyperparameters for CNN models. We expect to achieve better model performance through transfer learning in closely related domains compared to distantly related domains.

This project addresses the multifaceted challenges of image classification through CNNs, employing a ResNet-18 architecture, with Task 1 involving CNN model selection, hyperparameter tuning, and evaluating the use of pre-trained networks compared to training from scratch. Task 2 involves transfer learning, utilizing pre-trained CNN encoders to extract features from new datasets, subsequently integrating them into machine learning models for image classification. Transfer learning improves performance in new tasks by using prior knowledge from a previous task to reduce the need for extensive data collection and optimize resource usage. Feature extraction results are visualized with t-SNE, and model performance is assessed using confusion matrices and standard metrics.

2. Proposed Methodologies

Dataset 1, the Colorectal Cancer Classification dataset, comprises image patches categorizing three tissue types: smooth muscle (MUS), normal colon mucosa (NORM), and cancer-associated stroma (STR). This dataset originated from the NCT Biobank and the UMM pathology archive in Germany.

Dataset 2, the Prostate Cancer Classification dataset, includes image patches identifying three types of prostate tissues: Prostate Cancer Tumor Tissue, Benign Glandular Prostate Tissue, and Benign Non-Glandular Prostate Tissue. This data is associated with a research paper [3].

Dataset 3, the Animal Faces Classification dataset, consists of images classified into three classes representing animal types: Cats, Dogs, and wildlife animals. The dataset is sourced from AFHQ (Animal FacesHQ).

For Data Preprocessing, we used PyTorch's transforms module to preprocess images for deep learning. It resizes input images to 224x224 pixels, performs a center crop, converts them to PyTorch tensors, and normalizes pixel values based on ImageNet statistics ([0.485, 0.456, 0.406], [0.229, 0.224, 0.225]). These steps ensure standardized input sizes, focus on relevant image regions, compatibility with PyTorch tensors, and numerical stability during model training.

	D1	D2	D3
# Images	6000	6000	6000
Dim	224x224 pixels (RGB)	300x300 pixels (RGB)	512x512 pixels (RGB)
Format	TIFF	JPG	JPG
# classes	3	3	3
# images original D.	100000	6 dataset each containing 120000	16130
# classes original D.	9	3	3
Source	[2]	[3]	[1]

Table 1. Datasets description

In our hyperparameter optimization, we selected values for learning rate (lr), batch size (bs), and optimizers. We experimented 27 hyperparameter configurations (lr = [0.1, 0.01, 0.001], bs = [16, 32, 64], optimizer = ["SGD", "Adam", "RMSprop"]), training ResNet-18 models from scratch for 5 epochs. The top 5 configurations were selected based on performance. These five configurations underwent further training for 15 epochs, and the best-performing one (lr = 0.01, bs = 32, SGD optimizer), determined by validation accuracy, was identified. Additionally, a configuration (lr = 0.001, bs = 64, SGD optimizer) showing consistent improvement with each epoch was chosen.

We chose the ResNet18 CNN architecture for its proven effectiveness in capturing intricate features through residual learning, as demonstrated in benchmarks like the ImageNet Challenge. ResNet18's moderate depth strikes a balance between complexity and computational efficiency, making it suitable for resource-limited scenarios.

ResNet18, a variant of the ResNet architecture, comprises 18 layers organized into stages with residual blocks containing skip connections. These connections mitigate the vanishing gradient problem, enabling effective training of deeper networks. The architecture, with two convolutional layers in each building block, hierarchical feature capture. Downsampling is achieved through strided convolutions or pooling layers, and global average pooling reduces spatial dimensions before fully connected layers. This design supports efficient feature extraction and transfer learning, making ResNet18 powerful for image classification.

We trained ResNet18 from scratch and employed transfer learning with two approaches: first, fine-tuning the entire network with pre-trained ImageNet weights, and second, freezing all layers except the final fully connected layer, which was replaced and trained with random weights. These strategies allowed us to adapt the model to our task

while benefiting from pre-trained knowledge. In all training scenarios, we used optimized hyperparameters from the tuning process.

For feature analysis, we applied t-SNE to the output features (512 x 1) from trained CNN encoders. This technique visually represents feature distribution and separability in a lower-dimensional space (2 x 1), providing insights into the model's ability to distinguish between classes. The t-SNE output qualitatively assessed the model's performance, enhancing interpretability and understanding of feature relationships.

3. Attempts at solving the problem

We trained models from scratch and pre-trained models using two transfer learning approaches for the selected configurations. Below are the results for the six models.

Training Data Results:

Table 2. lr = 0.01, batch size = 32, optimizer = 'SGD'

	Scratch Model	Pretrained Model	
		Aproach1	Aproach2
Loss	0.002202	0.00032	0.117913
Accuracy (%)	99.976190	100.00	95.54761
Precision	1.00	1.00	0.973333
Recall	1.00	1.00	0.973333

Table 3. lr = 0.001, batch size = 64, optimizer = 'SGD'

	Scratch Model	Pretrained Model	
		Aproach1	Aproach2
Loss	0.023551	0.007248	0.17127
Accuracy (%)	99.428571	99.952380	94.8095
Precision	0.94	1.00	0.95
Recall	0.94	1.00	0.95

Testing Data Results:

Table 4. lr = 0.001, batch size = 64, optimizer = 'SGD'

	Scratch Model	Pretrained Model	
		Aproach1	Aproach2
Loss	0.154925	0.049607	0.1279
Accuracy (%)	95.50	98.38709	95.0555
Precision	0.973333	0.993333	0.95
Recall	0.973333	0.993333	0.95

The results indicate that the pre-trained model using transfer learning approach 1, with a learning rate of 0.01, batch size of 32, and optimizer SGD, achieves the best performance. The choice of learning rate and batch size had a significant impact on the models performance. Lower learning rates and larger batch sizes tend to have slightly higher losses but maintain a good accuracy. According to the results the use of the pretrained model has a high accuracy but may have slightly higher losses compared to the model trained from scratch.

Table 5. lr = 0.001, batch size = 64, optimizer = 'SGD'

	Scratch Model	Pretrained Model	
		Aproach1	Aproach2
Loss	0.141247	0.127920	0.1821
Accuracy (%)	95.50	95.055555	94.2222
Precision	0.886667	0.983333	0.94
Recall	0.883333	0.983333	0.95

Feature visualizations (t-SNE) for the output features of two CNN encoders, the best scratch model, and the best pretrained model are provided below.

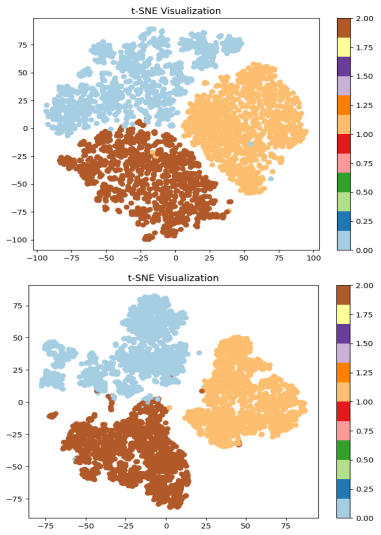

Figure 1. t-SNE for pre-trained model vs t-SNE for model from scratch

4. Future Steps

The model trained on Dataset 1 and the model pretrained on ImageNet are going to be used for feature extraction on Datasets 2 and 3. These four scenarios are going to be analyzed and visualized using t-SNE. After that, the data from the scenario that better separates the classes after applying t-SNE is going to be used to train two Machine Learning techniques, K-nearest neighbors and Random Forest, the performance of the ML techniques is going to be compared using metrics such as confusion matrix, accuracy, precision, recall, etc.

References

[1] Yunjey Choi, Youngjung Uh, Jaejun Yoo, and Jung-Woo Ha. Stargan v2: Diverse image synthesis for multiple domains. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 1

[2] Jakob Nikolas Kather, Niels Halama, and Alexander Marx. 100,000 histological images of human colorectal cancer and healthy tissue, May 2018. 1

[3] Yuri Tolkach. Datasets digital pathology and artifacts, part 1, June 2021. 1

5. Appendix

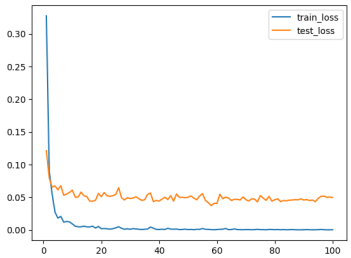

Figure 2. Loss of pre-trained model with 32 batch size and Learning rate 0.01 with Optimizer = 'SGD'

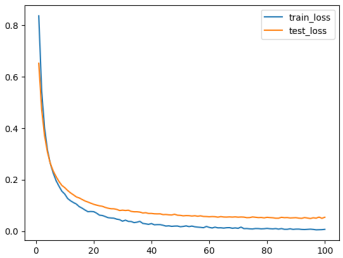

Figure 3. Loss of pre-trained model with 64 batch size and Learning rate 0.001 with Optimizer = 'SGD'

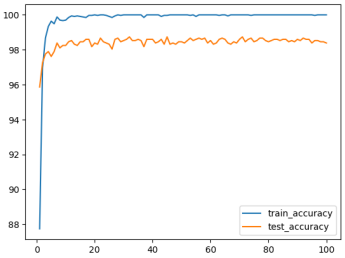

Figure 4. Accuracy of pre-trained model with 32 batch size and Learning rate 0.01 with Optimizer = 'SGD'

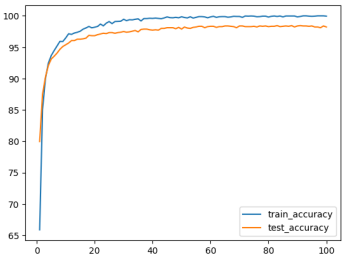

Figure 5. Accuracy of pre-trained model with 64 batch size and Learning rate 0.001 with Optimizer = 'SGD'

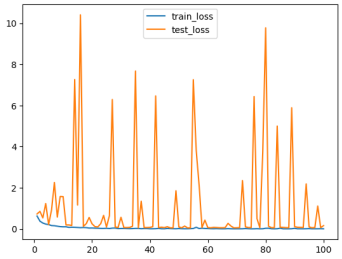

Figure 6. Loss of model from scratch with 32 batch size and Learning rate 0.01 with Optimizer = 'SGD'

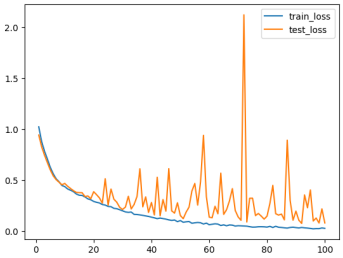

Figure 7. Loss of model from scratch with 64 batch size and Learning rate 0.001 with Optimizer = 'SGD'

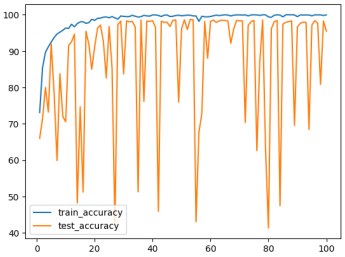

Figure 8. Accuracy of model from scratch with 32 batch size and Learning rate 0.01 with Optimizer = 'SGD'

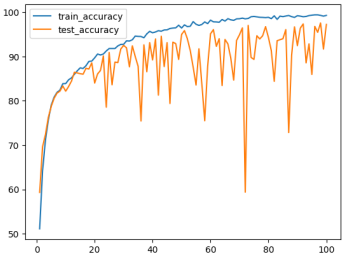

Figure 9. Accuracy of model from scratch with 64 batch size and Learning rate 0.001 with Optimizer = 'SGD'