Comparative Analysis of Machine Learning Algorithms for Digital Pathology Image Classification

Sadia Afrin Purba
Department of Electrical and Computer Engineering, Temple University
sadia.afrin.purba@temple.edu

Introduction: This project addresses the classification problem using a dataset consisting of Discrete Cosine Transform (DCT) coefficients extracted from pathology images. The dataset has nine distinct classes. Each image is transformed by 3072 features: 1024 coefficients for each RGB channel. For experimentation, we applied gaussian normalization across each RGB channel of the DCT coefficients which is essential for deep learning-based model convergence and performance. The dataset included a total of 10,066 images for training and 5,958 images as development set. An additional evaluation set consisting of 6,259 images was used as a blind evaluation to test model generalization. This paper focuses on evaluating two machine learning algorithms for classification tasks: the gradient boosting method XGBoost and Convolution Neural Network (CNN) integrated with skip connections. To establish a strong reference point for evaluating our two primary models, we first explored different baseline algorithms. These included both simple non-neural network-based methods, such as Random Forest (RNF), Naïve Bayes (NB), K-Nearest Neighbors (KNN), and K-Means clustering, as well as neural network-based architectures, such as Multi-Layer Perceptron (MLP), encoder-only Transformer, and a basic CNN with five layers. All models, except the CNN variants, were implemented using NEDC Vector Classify library with the default parameter file. Among all these models, NB and the 5-layer CNN gave the lowest score on the development sets (64.11% and 56.51%, respectively). As a result, these two were selected for comparison against the more advanced methods presented later in the paper.

XGBoost: XGBoost is a widely-used gradient boosting algorithm renowned for its performance in classification task specially on tabular datasets. Unlike RNF, which creates multiple decision trees in parallel, XGBoost builds trees sequentially, each learning from the mistakes of its predecessor. It initiates predictions with an initial model, identifies residuals (difference between predictions and actual values), and iteratively constructs new trees focused on correcting these errors. The final prediction aggregates all individual tree outputs. In this project, we used multilabel softmax as the objective function, a maximum tree depth of 6, and a learning rate of 0.1 to train the XGBoost model.

CNN with Skip Connections: CNN is a powerful architecture in image-related tasks due to their spatial hierarchical feature extraction. We integrate skip connections inspired by ResNet architecture which resolves issues such as vanishing gradient and improves information flow across layers. The implemented Residual CNN in this project has multiple convolutional layers, each coupled with skip connections. To make the data compatible with the CNN-2D architecture, we reshaped the original 1D vector input into 32×32 matrices for each of the RGB channels, resulting in a three-channel image-like input. Initially, the network applies 3x3 convolution operation on the input, enhancing feature extraction through increasing the depth as each layer (32, 64, 128, 256, and 512 filters). We created the skip connections through 1x1 convolutions, which helps to ensure the number of channels matches between the original and shortcut path. The final layer uses a fully connected structure, classifying the flattened features into one of the nine classes. The architectural overview of the Residual CNN is illustrated in Figure 1.

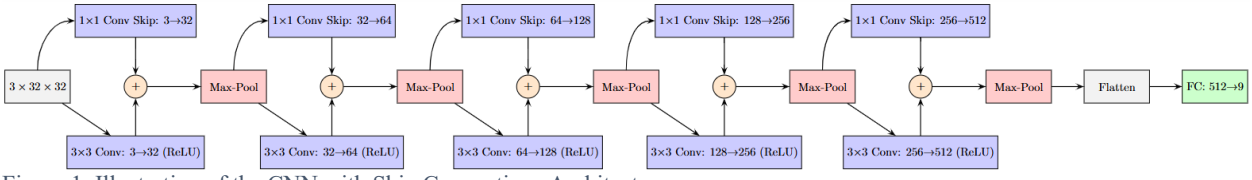

Figure 1. Illustration of the CNN with Skip Connections Architecture.

We used Optuna library to identify the optimal learning rates, weight decay, and batch sizes for training the Residual CNN architecture. The model used Adam optimization and Cross-Entropy loss function during the training phase. To prevent overfitting, we leveraged early stopping when the performance on the development set stops improving after a few epochs.

Results: The XGBoost model produced moderate performance, but showed clear sign of overfitting. It achieved a very low training error of 2.78% on the unnormalized dataset, indicating that the model fit the training data almost perfectly. However, we get higher error rates on the development and evaluation sets, suggesting poor generalization. This overfitting arises because, by default, XGBoost builds deep trees that can capture relationship including noise. We did not properly regulate the training process which led to the overtraining. On the other hand, CNN model with skip connections showed the most promising results for DCT coefficient features on the development set. When trained on gaussian-normalized data using both the training and development sets, the model achieved a low training score of 2.13% and development score of 3.93% with corresponding label-specific errors of 2.16% and 4.16%, respectively. On the blind evaluation set we got 52.81% error rate. However, like XGBoost, this architecture is also overfitted. Since DCT coefficients are already a compressed form of the original images, much of the spatial detail has been lost. Applying max pooling in every CNN layer further reduces spatial resolution, potentially worsening this information loss. We also experimented with replacing max pooling with average pooling to preserve more contextual information; however, this change did not result in significant improvement in dev set performance. So, we stick with the max-pooling implementation. We also explore advanced architecture during the experimental phase, including MLP-Mixer, a transformer-inspired architecture using only multilayer perceptions; MLP-Mixer with Reinforcement Learning (RL); Vision Transformer (ViT), utilizing attention mechanisms for image classification; and EfficientNet, known for balancing network depth, width, and resolution. Among these EfficientNet-b0 (EN-b0) delivered the strongest performance on the dev set. When trained on gaussian-normalized data from both the train and dev sets, it achieved a dev set error rate of 32.55%, with average label error of 35.57% and background error of 5.32%. Other architectures like MLP-Mixer variants and ResNet18 also showed moderate success, achieving dev set scores between 53.00% to 59.00%. However, ViT (with and without digital pathology pretrained weights) performed worst, with error rates between 68.00% to 78.00% on the dev set. The ViT model, which is originally optimized for raw pixel-value inputs, was applied here on DCT coefficient features and that led to the poor performance on the dev set.

			Data Set		
Algorithm	**Normalize**	**Train Data**	**Train**	**Dev**	**Eval**
NB	No	\train	56.87%	64.11%	-
CNN	Yes	\train	84.70%	56.51%	-
XGB	No	\train	2.78%	64.18%	62.18%
CNN + Skip	Yes	\train \dev	2.13%	3.93%	52.81%
EN-b0	Yes	\train \dev	33.77%	32.55%	52.94%

Table 1. Comparison of classification error rates for XGBoost and CNN with skip connection models across training, development, and evaluation sets.

Conclusions: This paper comprehensively evaluated two distinct ML approaches, XGBoost and CNN with skip connection, alongside advanced methods for classifying digital pathology images represented by DCT coefficients. Both advanced algorithms showed moderate success, with CNN outperforming XGBoost in feature extraction but suffer with overfitting issue. However, this comparative study concludes that DCT coefficients are not an ideal choice for digital pathology image classification. The inherent compression in DCT discards valuable spatial information, which is crucial for accurately capturing complex pathological patterns. For this problem, analyzing pixel-level features is essential for achieving lower error rates, ideally in the range of 30% or less.