**Breast Cancer Histopathology Image Classification**

*Dylan Boles*

Department of Electrical and Computer Engineering, Temple University

dylan.boles@temple.edu

**Introduction:** Breast cancer diagnosis relies heavily on the accurate classification of histopathology images. Machine Learning techniques can assist pathologists in this task, potentially improving both efficiency and diagnostic accuracy. This paper explores the effectiveness of two distinct machine learning approaches used for classifying breast cancer histopathology images using their Discrete Cosine Transform (DCT) coefficients: Convolutional Neural Networks (CNNs) and Random Forests. The dataset we were provided contains 3,505 images in RGB color format with a resolution of 32x32x3. Each image is represented as 1024 coefficients per color channel (red, green and blue). For Training and Development, the dataset includes nine different labels. The nine labels consisted of normal, artifact, non-neoplastic, inflation, suspicions, ductal carcinoma in situ, invasive ductal carcinoma in situ, null and background. However, for the purpose of this project, we only focused on six of these labels and excluded the null, artifact and suspicious classes.

Our primary goal was to achieve the highest classification accuracy possible, which could ultimately support faster and more reliable diagnoses by pathologists. I selected the Random Forest algorithm because it is well-suited to handling DCT-based features and us relatively computationally efficient. This allowed me to experiment with various parameters to optimize the model. I also chose a Convolutional Neural Network because I wanted to apply the inverse DCT to convert the signals from the frequency domain back to the spatial domain, resulting in 3,505 reconstructed 32x32 pixel images-ideal input for Convolutional Neural Networks. When visualizing the distributions of the classes in the dataset, we observed class imbalance, which posed an additional challenge in model training and evaluation.

**Algorithm No. 1 Description:** For my traditional method, I implemented a Random Forest classifier to perform multi-class classification on image-like data. By tuning parameters, we can control how the classifier divides the data into the nine classes. The model was trained on a combined dataset created from both the training and development sets. Using more labeled data benefits the model and can lead to improved performance, especially classes with fewer samples. Data preprocessing involved standardization using StandardScaler, applied consistently across training, development and evaluation sets. The Random Forest model is a relatively fast model to train, especially with the help of GPU support. This allowed me to experiment with multiple different combinations of hyperparameters efficiently. After testing various number of trees, maximum depth of the trees, minimum samples before a leaf, minimum samples before a split, and the maximum features, I found an optimal model with 800 trees, a maximum depth of 12, a minimum sample of 3 samples per leaf, the square root of the number of features, and a minimum of 3 samples per split. To help address class imbalance, I used the class weights variable and set to balanced in Scikit-learn’s Random Forests function. This automatically adjusts class weights, assigning higher weighs to underrepresented classes and helping the model learn them more effectively. Each decision tree in the forest uses subsets of our DCT coefficients to learn rules that distinguish between image classes based on their frequency patterns. Since DCT representations reduce redundancy and emphasize dominant features, Random Forests can perform robustly by capturing non-linear relationships between frequency components and class labels.

**Algorithm No. 2 Description:** The Neural Network method I used was a Convolutional Neural network (CNN). I began by applying the inverse discrete cosine transform (IDCT) to all the images to revert them back to the special domain. This allowed me to feed the CNN pixelated images, which is the standard input format for Convolutional Neural Networks. For preprocessing, I combined the training and development datasets to train on all the available labeled data. I applied data augmentation techniques including rotation, affiliate transforms, horizontal and vertical flips, and normalization using the standard deviation and mean values from ImageNet, a large-scale image dataset. This helped the model generalize better. Since the data was already annotated, applying augmentations like color jitter and contrast adjustments actually worsened performance. The Convolutional Neural Network model I built consist of five layers: four 2D convolutional layer and two fully connected layers. Each convolutional layer includes Batch normalization, a Max Pooling layer, a ReLU activation function, and dropout. The Max Pooling layer selects the largest values from each kernel to retain the most prominent features. ReLU introduces non-linearity, and dropout helps prevent overfitting. To handle complex classes like Ductal Carcinoma in situ-where images may have blur or indistinct edges-I increased the network complexity by starting with a 32x32 convolutional layer and expanding to a 64x64 convolutional layer with five kernels. This gave the model more spatial awareness to detect patterns. In the final fully connected layer, I used a higher dropout rate to further reduce overfitting. For training, I used a cross-entropy loss function, allowing us to calculate errors and update weights accordingly. I applied an Adam optimizer to control weight updates, along with a ReduceLOnPlatue function to reduce the learning rate when the model stopped improving. The model was trained for up to 100 epochs, but early stopping was triggered if performance failed to improve over five consecutive epochs, preventing overfitting and unnecessary computation. After training, the model proceeded to generate prediction.

**Results:** The results after running both my Random Forest and Convolutional Neural network models show a clear difference in performance. The Convolutional Neural network demonstrates a stronger ability to generalize, achieving a 21.32% error rate on the training set and a 22.29% on the development set. The evaluation set yielded an error rate of 38.78%, which is a strong result for a Convolutional neural network on this dataset. In contrast, the Random Forest model had lower error rates during training-8.85% on the training set and 10.49% on the development set. However, the evaluation error rate was 54.07%, indicating significant overfitting. This suggests that the CNN was better at learning features that generalize to new, unseen data, whereas the Random Forest model memorized the training data too closely. Both models showed the most classification errors in non-neoplastic, ductal carcinoma in situ, and invasive ductal carcinoma in situ data. This is understandable, as non neoplastic data and this is the most abundant and the hardest to classify due to its lack of clear edges and defining patterns.

|  |  |
| --- | --- |
|  | **Data Set** |
| **Algorithm** | **Train** | **Dev Test** | **Eval** |
| Convolutional Neural Network | 21.32% | 22.39% | 38.78% |
| Random Forest | 8.85% | 10.49% | 54.07% |

Table 1. Convolutional Neural Network vs. Random Forest Results.

**Conclusions:** In this project, we exploded the application of machine learning to the classification of breast cancer histopathology images using DCT coefficients. Both models-Random Forest and Convolutional Neural network-were implemented and evaluated. The Random Forest was chosen for its ability to efficiently handle high-dimensional data and capture non-linear relationships, while the Convolutional Neural Network was selected to leverage spatial information by processing the inverse DCT transformed images. The Convolutional Neural Network model demonstrated superior performance, suggesting that learning directly from the pixelated images resulted in a more effective classification. While the Random Forest offered the advantage of fast training and easier hyperparameter tuning, its performance dropped significantly on evaluation, overfitting from 8% and 10% to 54.07%. Further research could investigate hybrid approaches that combine the frequency domain and spatial domain features. There is strong potential in these models, especially with access to more labeled data and the use of transfer learning techniques.