Classification of Healthy, Peripheral Blood Cells

Lee Perkins, Cinthya Rosales, Faye Titchenal
Data Science 281 Final
Section 3
Spring 2024

Abstract

In this study, we explore the application of computer vision technology to the classification of healthy peripheral blood cells. Using a dataset comprising over 17,000 images of eight distinct blood cell types, we implement various preprocessing techniques to address issues such as class imbalance and variable image dimensions. We employ feature extraction methods including Histogram of Oriented Gradients (HOG), ResNet features, and luminance histograms to effectively distinguish between cell types based on their textural and optical properties. Our classification approach leverages several machine learning models, including logistic regression, support vector machines, random forests, and neural networks, to evaluate the potential of these techniques in enhancing the efficiency and accuracy of blood cell analysis in medical laboratories.

1. Introduction

The intersection of healthcare and technology has perpetually been a forefront of innovation, particularly within the realm of medical diagnostics. Advancements in computer vision technology offer promising avenues for enhancing medical diagnostics, particularly through the automation of routine tasks such as blood cell classification. This tool will not only streamline the blood cell count and identification processes but also lay groundwork for future applications in personalized medicine and the early detection of hematologic diseases. Our project is motivated by the potential to significantly reduce diagnostic times, thereby enhancing patient outcomes and the efficiency of healthcare services.

This research leverages a large dataset of peripheral blood cell images, addressing challenges like class imbalance and varied image dimensions through systematic preprocessing. We employ a combination of traditional and novel feature extraction methods to effectively capture the unique properties of each cell type. Subsequent classification is performed using a range of machine learning models, assessing their ability to improve both the speed and reliability of blood cell analysis in medical settings.

2. Data

The dataset used in this study, Blood Cells Image Dataset [1], is sourced from Kaggle and comprises a comprehensive collection of 17,092 high-resolution images. These images depict individual normal cells captured using the CellaVision DM96 analyzer at the Core Laboratory of the Hospital Clinic of Barcelona. The dataset categorizes these images into eight distinct groups: neutrophils, eosinophils, basophils, lymphocytes, monocytes, immature granulocytes (promyelocytes, myelocytes, and metamyelocytes), erythroblasts, and platelets or thrombocytes. Each image measures 360 x 363 pixels, is in JPG format, and has been meticulously annotated by expert clinical pathologists. The specimens were derived from individuals free of infection, hematologic or oncologic diseases, and were not under any pharmacological treatment at the time of blood collection. This dataset, notable for being the first publicly available collection of such a scale of normal peripheral blood cells, offers a unique resource for training and benchmarking machine learning and deep learning models aimed at recognizing various types of normal peripheral blood cells.

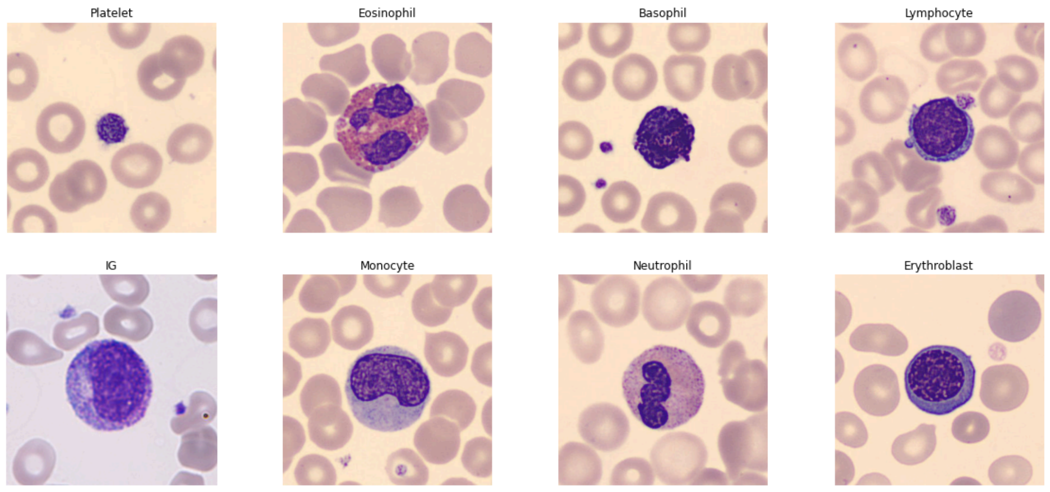

Fig 1: Example of original images for each Blood Cell class

3. Data Preprocessing and Augmentation

The original dataset contains a varied number of images across the eight categories, which reflects the natural occurrence rates of these cell types in peripheral blood samples. To address the imbalance across the eight different cell types in our dataset, we employed a class balancing technique by randomly sampling 1,200 images per class, ensuring uniformity for machine learning input. Each image underwent a center-cropping process to a fixed dimension, followed by the application of a specialized library for background removal, to remove noise prior to feature extraction. The images were then converted to grayscale and normalized to a zero-one scale to streamline the feature processing while retaining necessary luminance and textural features. We divided the dataset into training, validation, and testing segments with a proportion of 70/15/15.

Due to the nature of the data collection process, the image dataset was generated under highly controlled and consistent lighting and magnification conditions. These conditions produced blood cell images of representative scale and consistent position. However, the rotation of the cell within the image was not controlled. Therefore, to account for this variance and enhance the robustness of our classification models, we augmented the training dataset with rotated and reflected images. This process resulted in a balanced training dataset of 2,547 images per class.

As mentioned above, a source of significant variation in the image dataset was background noise. Due to the nature of blood samples, each image in the dataset contains extra cells in the background which makes feature extraction difficult. Therefore, we leveraged the Rembg python library [2] to systematically replace the values of the "background" pixels with zeros to simulate a black background in the image. The Rembg library is generalizable to a wide range of images and is built with U-Net architecture that programmatically segments images into "foreground" and "background".

We tested this method on a sub-sample of training images and confirmed that it worked well for background removal for the majority of images. The algorithm, however, struggled to correctly remove the background from the Platelet class. This is likely because Platelets are the smallest blood cell type making it difficult to distinguish between the background cells and the actual Platelet cell. This resulted in the pixels for multiple cells or the wrong cell being retained in the final image (reference Fig 2).

Despite this issue we chose to use the background removal function as part of the pre-processing pipeline as it was capable of correctly removing the background from the majority of the images and greatly reduced the noise captured during feature extraction.

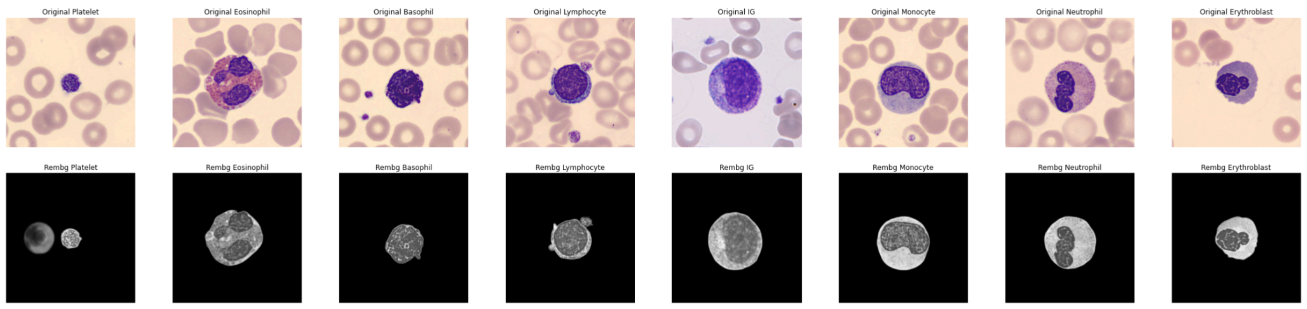

Fig 2: Example of original and final pre-processed images with "rembg" background removal. Background removal was incomplete for Platelet class example.

4. Feature Extraction

A review of existing literature [3] indicated that both luminance and texture were useful and descriptive features for blood cell classification tasks. One main reason for this is that each type of blood cell scatters and absorbs light differently and those differences can be quantified with luminance analysis. Therefore, to extract both luminance and texture features, we employed techniques to extract luminance histograms and HOG. Additionally, based on prior research [4], we chose ResNet50 as a method for extracting more detailed patterns in cell textures, edges, and granule shapes. ResNet50 is a pre-trained convolutional neural network containing 50 dense layers which can be used for both feature extraction and image classification.

4.1. Luminance Histogram

While the luminance histogram was capable of describing the patterns in light absorption and scatter between classes, we found that this feature is also highly influenced by the size of the blood cell. Because the images were taken under constant magnification, the size of the cell is represented by the quantity of pixels with a value of zero. After background removal, the majority of the pixel values in the images were remapped to black (zero pixel value), significantly skewing the luminance histograms (Figure 3).

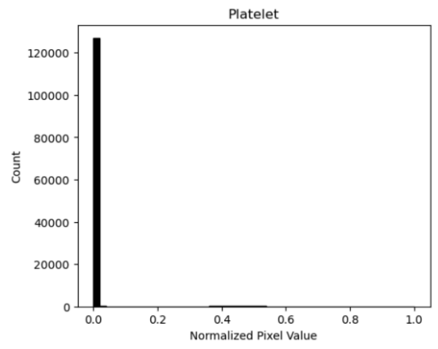

Fig 3: Example histogram demonstrating significant skew of luminance histograms due to background

Despite the imbalance in the histograms introduced by the black background, we ultimately decided to retain the histogram bin representing pixel values of zero based on our dimensionality reduction analysis (reference section 5.4). Excluding the black background, the variation in luminance across classes is demonstrated in Figure 4. To refine this feature vector, future experimentation would include further cropping the image to reduce the amount of skew introduced by the background pixels and evaluating the impact on model performance by excluding the background pixel values from the feature vector completely.

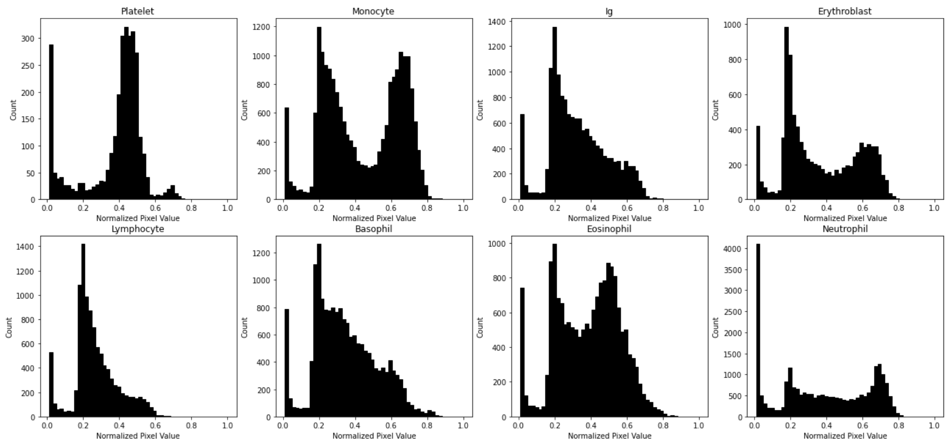

Fig 4: Example of luminance histograms for each class excluding black background pixels

4.2. HOG

Histogram of Oriented Gradients (HOG) is a method to quantify the occurrences of gradient orientations in a localized portion of an image. The gradient orientations are defined as the magnitude and angle of edges within each section of the image. The occurrences of each gradient orientation are counted and binned to form the final feature vector. For our analysis, the Scikit-Image HOG function was leveraged to extract the feature vectors. As shown in the HOG visualization (Figure 5), the cell and granule sizes and shapes are defined by the HOG feature vector.

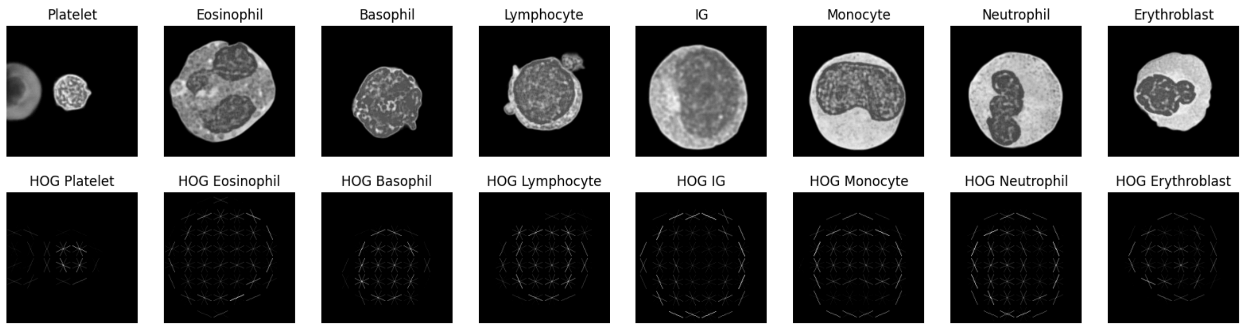

Fig 5: Example of pre-processed images compared with HOG feature vector representations

The distribution of HOG features across the training dataset (Figure 6), shows a similar skew towards zero. Again, these correspond with the black background that did not contain any edges or changes in frequency. Each class, however, contains a different rate of zero values indicating that this feature vector is also useful for describing the overall size of the cell. There is clear differentiation in the distribution of occurrences between classes for values above zero. These differences indicate that the feature is capturing the differences in granule shape and size between classes.

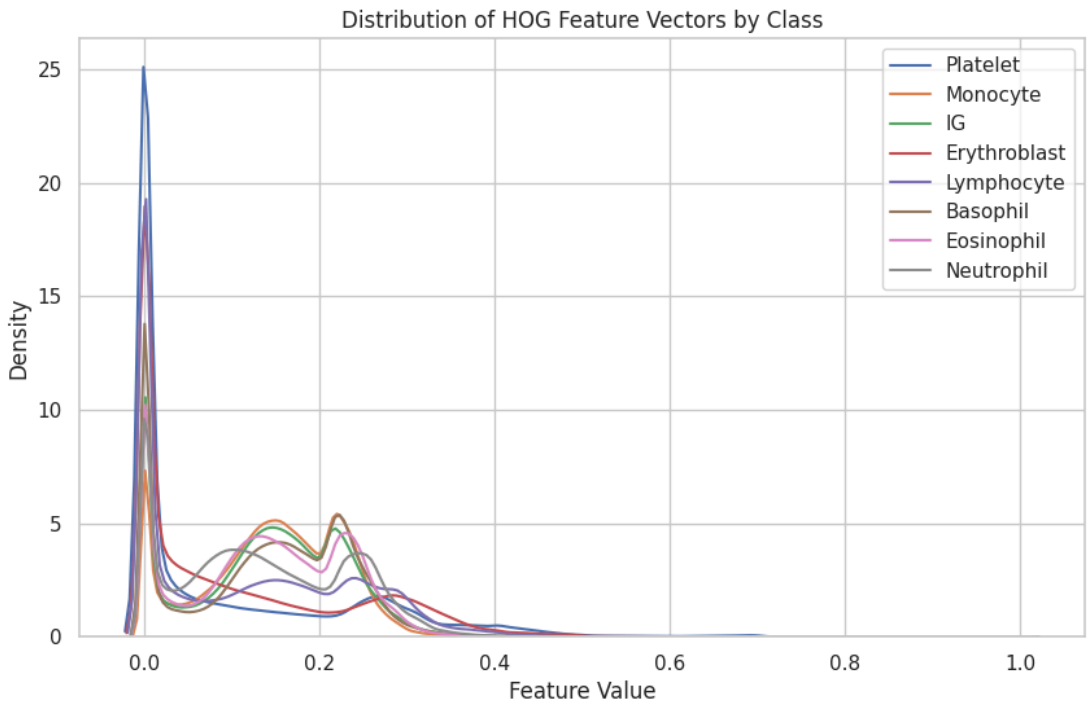

Fig 6: Distribution of HOG feature vectors for training dataset

4.3. ResNet50

To extract the feature vectors from the ResNet50 model, the grayscale images were processed further to meet the requirements of ResNet50 input. The images were cropped to 224 by 224 pixels, converted to 3-channel images by duplicating the grayscale values, and zero-centering each channel with respect to the ImageNet dataset (the dataset used for original training of ResNet50). Since the ResNet model was being used for feature extraction rather than image classification, we built and ran the model without the final, fully connected layer used for image classification. The remaining final layer then produced 2,048-length embeddings that we used as our final feature vector. The distribution of the embeddings for each class in the training dataset was plotted and demonstrates the differences in these embeddings between classes.

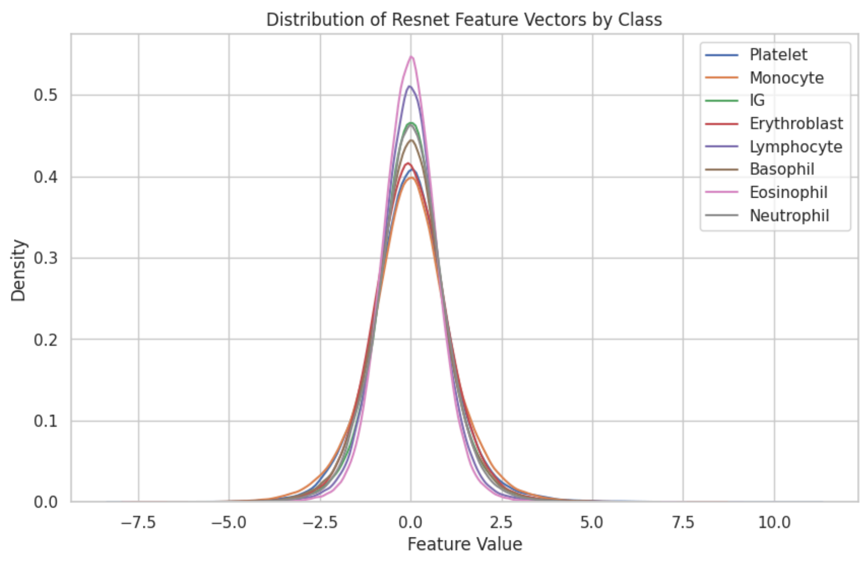

Fig 7: Distribution of ResNet50 feature vectors for training dataset

4.4. PCA and tSNE

Principal Component Analysis (PCA) was performed on each individual feature vector to extract the components of the feature vector that contributed the most to the explained variance within the extracted feature vectors. The cumulative explained variance plot (Figure 8), demonstrated that almost 90% of the variance in the luminance histogram feature vector was explained by a single component of the feature vector. This is expected given that the histograms were highly skewed based on the histogram bin representing the zero-value pixels. Ultimately, five components with a cumulative explained variance of approximately 99% were selected for transformation of the luminance feature vector. We believed these five components would capture both the differences in cell size and light scatter between cell classes.

The HOG and ResNet50 feature vectors required more components to converge on an optimal explained variance. Based on the cumulative explained variance curves, 77 and 93 components were selected from the HOG and ResNet50 feature vectors, respectively, to achieve approximately 90% explained variance.

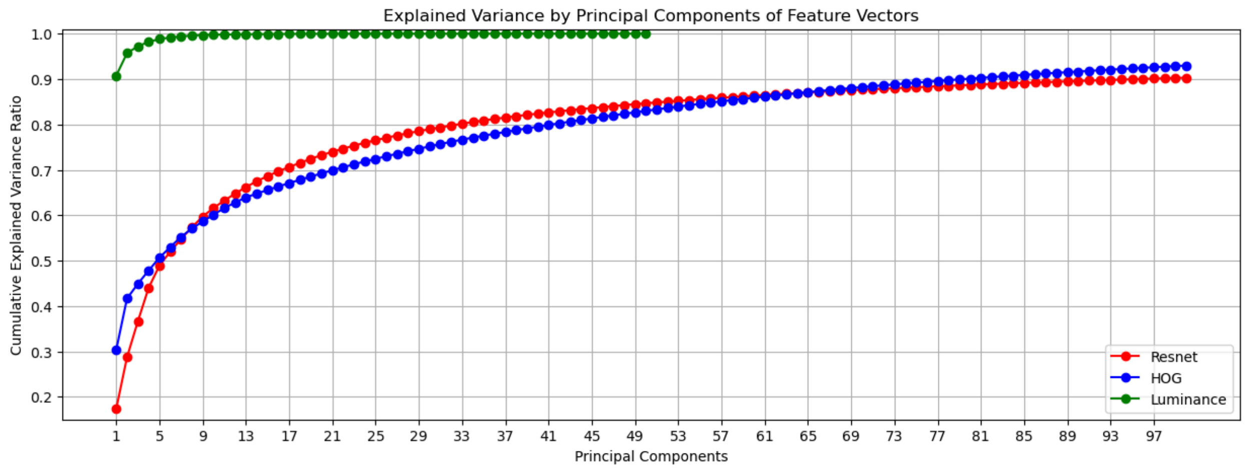

Fig 8: PCA cumulative explained variance curves for each feature vector

T-distributed Stochastic Neighbor Embedding (tSNE) was performed on a subset of the training data (500 random data points per class), to visualize the clustering of the training data after PCA dimensionality reduction. Our experimentation confirmed that luminance histograms are a very simple yet powerful descriptor of the grayscale blood cell images as demonstrated by the defined clustering of classes (reference Figure 9).

Despite our initial hypothesis, defined clustering of classes was not immediately evident as part of the tSNE visualization. There appears to be some clustering of the Platelet and Lymphocyte classes, however, the remainder of classes have little definitive separation. This result is likely due to non-optimal HOG extraction parameters such as too few orientations or too many pixels-per-cell. For future refinement of this feature, we would experiment with alternate parameters as a means to optimize both the resolution of the HOG feature vectors and the compute resources required to extract these features.

Unsurprisingly, the tSNE visualization demonstrated that the principal components of the ResNet50 feature extraction were fairly successful at clustering the classes into distinct groups. While there appeared to be overlap between the Monocyte, IG, and Eosinophil classes, we refrained from drawing definitive conclusions from the tSNE visualizations alone based on the limitations of visualizing this high dimensional feature vector in the 2D space.

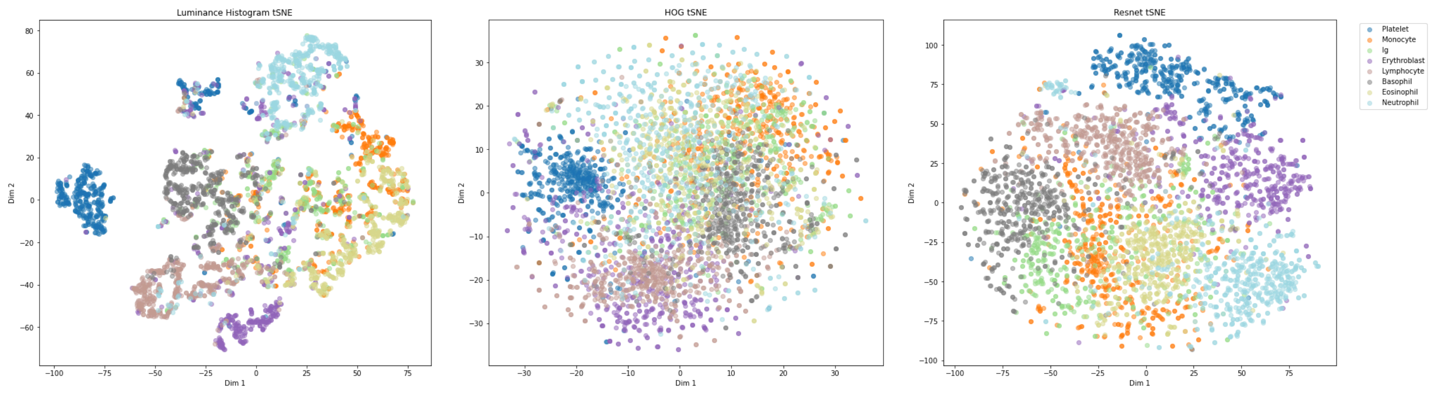

Fig 9: tSNE visualization of feature vectors on a subset of training data

5. Model Building and Evaluation

We employed five different models: Logistic Regression, Support Vector Machine (SVM), Random Forest, Perceptron, and RestNet50 (pre-trained CNN). Table 1 summarizes the train, validation, and test accuracies for all the models.

Please refer to the notebooks folder here https://github.com/BaiKongQue/281-final-project to look at any code and processes for our model implementation. You will find Logistic Regression and Random Forest in Faye's notebook. The Support Vector Machine in Cynthia's notebook. The Perceptron and ResNet50 models can be found in Lee's notebook under the "Classification" section.

Model	Train Accuracy	Val Accuracy	Test Accuracy	Train Time
Logistic Regression	0.9302	0.7438	0.8190	30.6s
SVM	0.9459	0.7466	0.8347	102.5s
Random Forest	0.9382	0.7857	0.8067	44.3s
Perceptron	0.9131	0.9258	0.9358	35.2s
ResNet50	0.9782	0.9087	0.8955	183.2s

Table 1: Train, Validation, and Test accuracies of all the models

5.1. Logistic regression

Logistic regression is a linear, supervised machine learning technique used for binary classification tasks. We chose a logistic regression model as an initial baseline model due to its simplicity, relatively low computational cost, and explainability.

In our case, we adapted a logistic regression model for multiclass classification by utilizing the softmax activation function which assigns a probability to each class for every data point. The concatenated feature vectors (Luminance, HOG, and ResNet50) were used as input. Hyperparameter optimization using Random Search was used to find the best learning rate, decay steps, and decay rate for the model. After hyperparameter optimization, we obtained a train accuracy of 93.02%, validation accuracy of 74.38%, and a test accuracy of 81.90%. However, even after hyperparameter tuning there was still about 20% difference between the train accuracy and validation accuracy indicating overfitting to the training dataset.

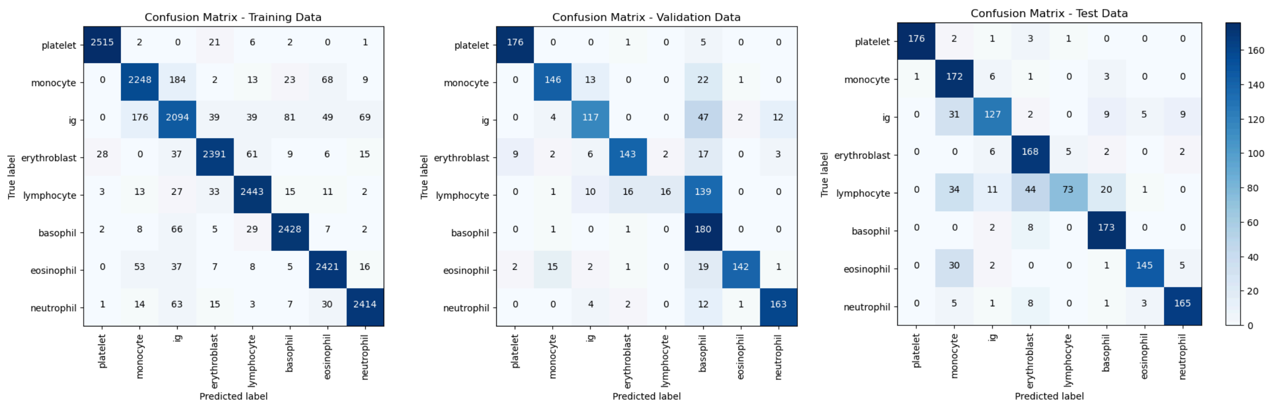

Fig 10: Logistic Regression confusion matrices for train, val, and test datasets

The model struggled to correctly classify the Lymphocyte and Basophil classes for both the validation and test datasets. This result is likely due to either the model's sensitivity to outliers caused by the imperfect background removal pre-process or underlying nonlinear relationships between the classes that could not be captured by the linear model.

5.2. Support Vector Machine (Linear)

Support Vector Machines (SVM) are robust, supervised machine learning algorithms renowned for their ability to classify data by finding an optimal hyperplane that distinctly separates different classes. Given their effectiveness in handling outliers and their adaptability to high-dimensional spaces, we chose a linear SVM as an advanced step in our classification strategy after establishing a baseline with logistic regression.

For this task, we adapted the SVM to manage the multiclass classification by employing a linear kernel, due to its computational efficiency and straightforward implementation. The input for this model was the same concatenated feature vectors used in logistic regression, including Luminance, Histogram of Oriented Gradients (HOG), and ResNet50 features.

Hyperparameter tuning was essential to optimize the SVM’s performance, with a focus on the regularization parameter C. We utilized GridSearchCV with a StratifiedKFold cross-validation strategy to methodically explore different values of C and ensure the model was not overly fitted to the training data. The best parameters found significantly improved the model's generalizability.

The final model achieved a training accuracy of 94.59%, a validation accuracy of 74.66%, and a test accuracy of 83.47%. Despite these promising results, the model exhibited some overfitting, as indicated by the difference between the training and validation accuracies. Additionally, similar to the logistic regression model, the SVM struggled with accurately classifying the Lymphocyte and Basophil classes, suggesting that the linear kernel might not fully capture the complex, nonlinear relationships present in the data.

This outcome stresses the potential need to explore SVMs with non-linear kernels in future work to better address the intricacies of the dataset. The linear SVM, while less computationally intensive than more complex models, still offers a significant improvement in handling outliers and high-dimensional data, making it a valuable tool in our analytical arsenal.

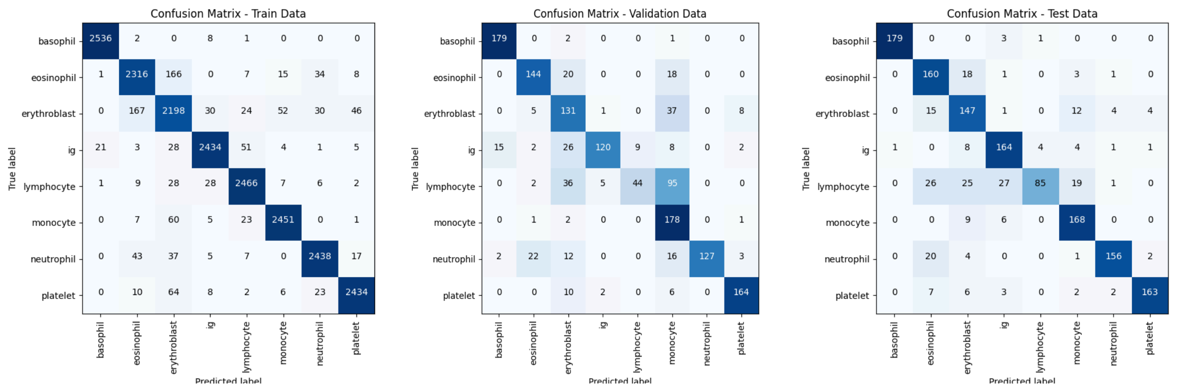

Fig 10: SVM confusion matrices for train, val, and test datasets

5.3. Random Forest

Random Forest is a machine learning algorithm that combines the output of multiple decision trees to produce a single result and is capable of modeling nonlinear relationships within the data. Therefore, we opted to build and evaluate a Random Forest model as a next step in improving model performance.

To optimize and finetune the hyperparameters, a grid search optimization was utilized. The optimizations found that 200 estimators and a maximum depth of 10 produced the best results with a train accuracy of 93.82%, a validation accuracy of 78.57%, and a test accuracy of 80.67%. The train and validation accuracy did not converge to the training accuracy, suggesting that there was still overfitting present even after limiting the maximum depth of the forest model. While the validation and test confusion matrices suggest that this model was capable of correctly classifying the Lymphocyte class, there was little improvement in the overall accuracy of the model compared with the linear logistic regression and SVM models.

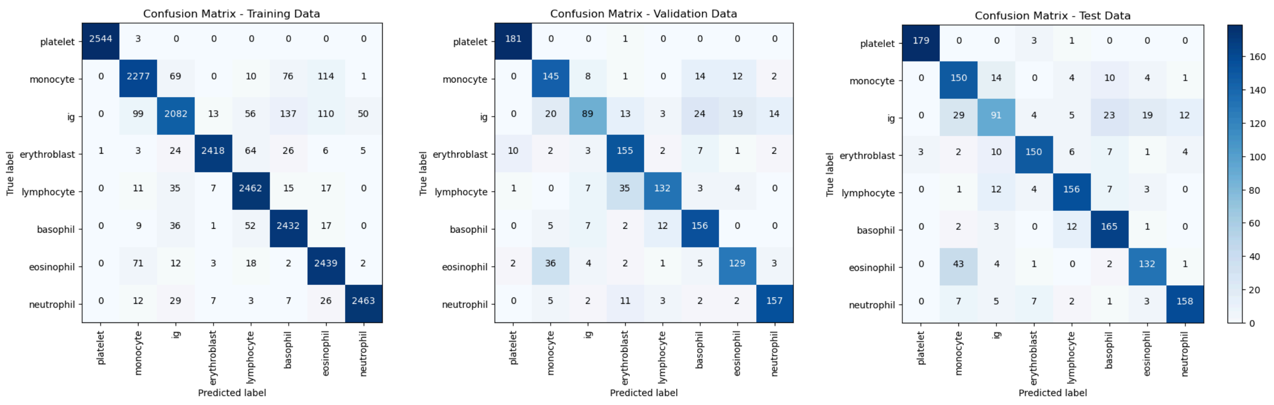

Fig 11: Random Forest confusion matrices for train, val, and test datasets

5.4. Perceptron

Perceptrons are a single layer neural network that are considered the simplest form of a neural network used for supervised learning of binary classifiers. Our perceptron was developed with a dropout layer, a single hidden layer, and an output layer with a softmax activation function for multiclass classification. The dropout layer is added to reduce overfitting and the single hidden layer uses a Relu activation function to equip the model with the ability to learn nonlinear patterns in the data.

Using an adam optimizer with a learning rate of 0.0001, the model achieved a training accuracy of 91.31%, validation accuracy of 92.58%, and a test accuracy of 93.58%.

For future refinement of the model, we would perform hyperparameter optimization to evaluate whether the train accuracy could be further improved. However, with the initially selected hyperparameters, the model performed very well while having the train and test accuracies converge. This suggested that the model not only learned the training set well, but also generalized well for classification of both the validation and test datasets.

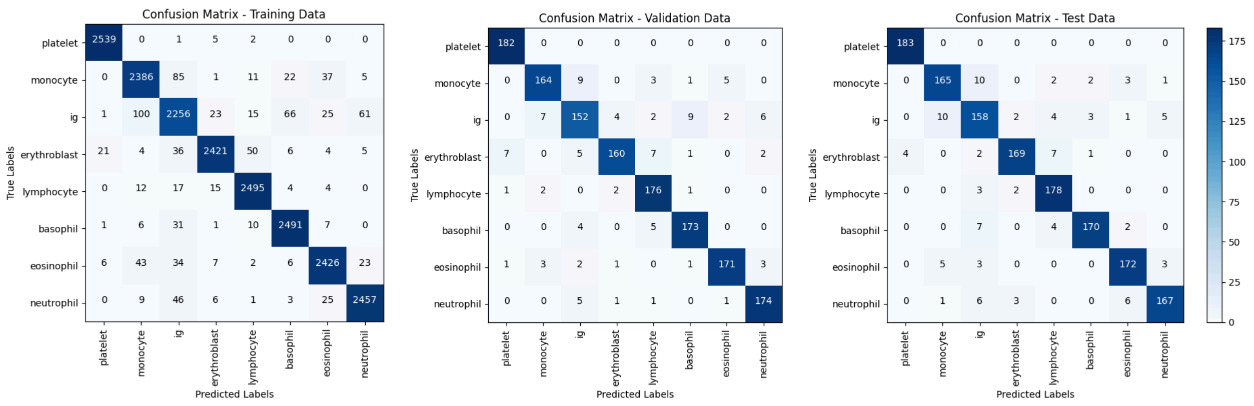

Fig 12: Perceptron confusion matrices for train, val, and test datasets

5.5. ResNet50

ResNet50 is a 50-layer convolutional neural network that introduces skip connections, allowing the network to bypass certain layers during training, combating the vanishing gradient problem. Another benefit ResNet50 provides is pretrained weights, saving lots of time and computational effort. A single dropout layer was added to help with overfitting, and a single hidden layer with a Relu activation function was added for additional learning.

Using an adam optimizing with a learning rate of 0.001, we achieved a training accuracy of 97.82%, validation accuracy of 90.87%, and a test accuracy of 89.55%. The train and validation accuracies did not converge, and they could have been closer with more parameter tuning. One of the challenges with the model was reshaping the vector to fit a four dimensional expectation of being a three channel image. ResNet50 was trained to classify images, not extracted features. The high validation and test accuracies suggest that the ResNet50 did a pretty good job at generalizing and learning from the features even though they were not images.

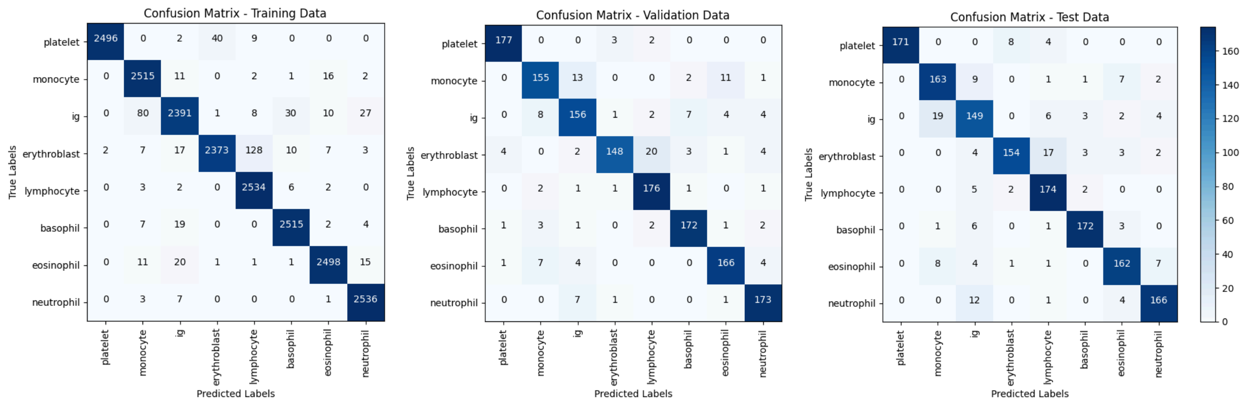

Fig 13: ResNet50t confusion matrices for train, val, and test datasets

6. Conclusion

This study demonstrated the application of advanced machine learning techniques to automate the classification of healthy peripheral blood cells, marking a significant step forward in the integration of computer vision into medical diagnostics. Through meticulous preprocessing, robust feature extraction, and the deployment of multiple machine learning models, we achieved notable success in identifying various blood cell types with high accuracy.

Our findings highlight the Perceptron model as the most effective, achieving a remarkable test accuracy of 93.58%. This success underscores the potential of neural networks in handling complex classification tasks that involve nuanced differences between cell types. Conversely, the logistic regression and SVM models provided valuable insights into the challenges of using linear methods for such complex data, particularly in terms of their limitations in capturing the intricate relationships within the blood cell features.

The Random Forest and ResNet50 models, while robust in their performance, also revealed the critical balance between model complexity and computational efficiency. Each model played a crucial role in advancing our understanding of how different algorithms handle the subtleties of medical image data. Particularly, the ResNet50 model demonstrated the advantage of deep learning in extracting and learning from high-dimensional data, albeit at a higher computational cost.

However, the study was not without its challenges. The issues of overfitting and the struggle to generalize across different clinical scenarios highlighted inherent limitations in the models' capabilities. Moreover, the discrepancies in performance between training and validation phases suggest a need for improved methods in model training and validation to enhance their reliability and accuracy.

Moving forward, this research lays a robust foundation for further exploration into automated systems for diagnosing and understanding hematologic diseases. Future studies should focus on refining preprocessing techniques, exploring more sophisticated models that can handle the non-linearities of medical data, and expanding the dataset to include diseased cell types. Additionally, addressing the generalizability of these models to operate effectively across various magnifications and imaging conditions will be critical.

In conclusion, this project not only enhances the efficiency of medical diagnostics but also opens new avenues for advanced healthcare solutions, thereby contributing significantly to the field of digital pathology. The insights gained from this study encourage ongoing innovation in the automation of medical image analysis, ultimately leading to improved patient care and outcomes.

7. References

[1] Blood Cells Image Dataset. www.kaggle.com. https://www.kaggle.com/datasets/unclesamulus/blood-cells-image-dataset.

[2] Gatis D. Rembg. GitHub. Published October 12, 2022. https://github.com/danielgatis/rembg

[3] Rustam F, Aslam N, De La Torre Díez I, et al. White Blood Cell Classification Using Texture and RGB Features of Oversampled Microscopic Images. *Healthcare*. 2022;10(11):2230. doi:https://doi.org/10.3390/healthcare10112230

[4] 1.Zhu Z, Wang SH, Zhang YD. ReRNet: A Deep Learning Network for Classifying Blood Cells. *Technology in Cancer Research & Treatment*. 2023;22:15330338231165856. doi:https://doi.org/10.1177/15330338231165856