Analysis of Machine Learning Techniques for Cervical Cell Classification

Daniela Ferreira (14997), Pedro Mendes (103028)
Department of Electronics, Telecommunications and Informatics, University of Aveiro
Course: Foundations of Machine Learning, Course Instructor: Petia Georgieva Georgieva
Email: {daniela.f.m.c.f, pmiguelsilvamendes}@ua.pt
Work Load: 0% 100%

***Abstract*—Cervical cancer is a major health concern, and early detection is critical for improving survival rates. However, traditional diagnostic methods, like cytological analysis, are time-consuming and require a lot of expertise. This study uses machine learning techniques to automate cervical cell classification, making the diagnosis faster and more accurate.**

We used the SIPaKMeD dataset, which includes 4049 labeled images of cervical cells across five different types: dyskeratotic, koilocytotic, metaplastic, parabasal and superficial-intermediate. We applied three machine learning models, Convolutional Neural Networks, Light Gradient Boosting Machine, and Support Vector Machines, to predict cell types.

The results show how well these models perform at classifying cervical cells and highlight which methods might be the best fit for real-world medical use. By comparing each technique, we hope to provide insights into how machine learning can help improve cervical cancer diagnosis and make early detection more accessible.

***Index Terms*—Cervical Cancer, Machine Learning, Image Classification, CNN, LightGBM, SVM, SIPaKMeD Dataset**

I. Introduction

Cervical cancer is the fourth most common type of cancer among women, being a major health issue, especially in areas with limited resources. Early detection can greatly reduce mortality rates, but traditional diagnostic methods like cytological analysis are time-consuming and need a lot of expertise [1].

For this study, we used the SIPaKMeD dataset, one of the biggest collections of cervical cell images available. It contains 4049 labeled single cell images, categorized into five classes: dyskeratotic, koilocytotic, metaplastic, parabasal and superficial-intermediate, which are of extreme importance to determine the severity of the condition.

We applied and compared three machine learning techniques, Convolutional Neural Networks (CNN), Light Gradient Boosting Machine (LightGBM) and Support Vector Machines (SVM), with the goal of predicting the type of cell the algorithm is analysing.

Through the application of these methods, we aim to evaluate their effectiveness in classifying cervical cell images and identify the most suitable approach for this task. The study also explores the strengths and weaknesses of each method, providing insights into their performance and applicability in real-world clinical settings.

II. Motivation

Classifying cells from microscopic images is an important task in medical research, especially for detecting diseases, but manually identifying different cell types is time-consuming and can lead to mistakes. Machine learning can help by automatically learning patterns from images and making predictions, reducing human error and improving efficiency.

This project explores how machine learning can be applied to cell classification. By evaluating different models and analysing performance metrics, we aim to understand how well machine learning can assist in this task, where hopefully the findings could contribute to improving medical image analysis, making diagnosis faster and more reliable.

III. Problem Complexity

Cervical cell classification is a challenging task due to several factors. First, the dataset consists of high-dimensional image data, where each image is made up of thousands of pixels. Analysing this data requires algorithms that can find meaningful patterns without being overwhelmed by irrelevant information.

Additionally, the problem is non-linear because the features that distinguish different cell types, such as the shape and size of the nucleus or texture of the cytoplasm, are subtle and complex. Classical machine learning models may struggle with such patterns, as we will see once we compare all methods.

The dataset size also adds to the complexity. While SIPaKMeD is one of the largest datasets available for this type of study, it is still small compared to what some models typically require. This means there's a risk of overfitting, where the model performs well on the training data but poorly on new data.

Lastly, real-world factors introduce additional challenges. For example, in clinical settings, images can vary in quality due to differences in equipment or sample preparation. These inconsistencies make it difficult to apply models trained on a clean dataset like SIPaKMeD directly to real-world cases.

IV. Dataset Description

The SIPaKMeD dataset is one of the largest publicly available datasets for cervical cell classification. It contains 4049 labeled single-cell images, which have been categorized into five distinct classes: dyskeratotic and koilocytotic cells,

considered abnormal, metaplastic cells, considered benign, and parabasal and superficial-intermediate cells, considered normal. These categories are essential for diagnosing cervical cancer and assessing the severity of abnormalities [2].

The dataset also contains images of clusters of cells, where the labeled single-cell images were extracted from. Due to hardware issues, only the single cells were used as training for our models. These cells are found inside the folder *CROPPED*, within the folders of each class.

In Figure 1, we can see how many features each class contains.

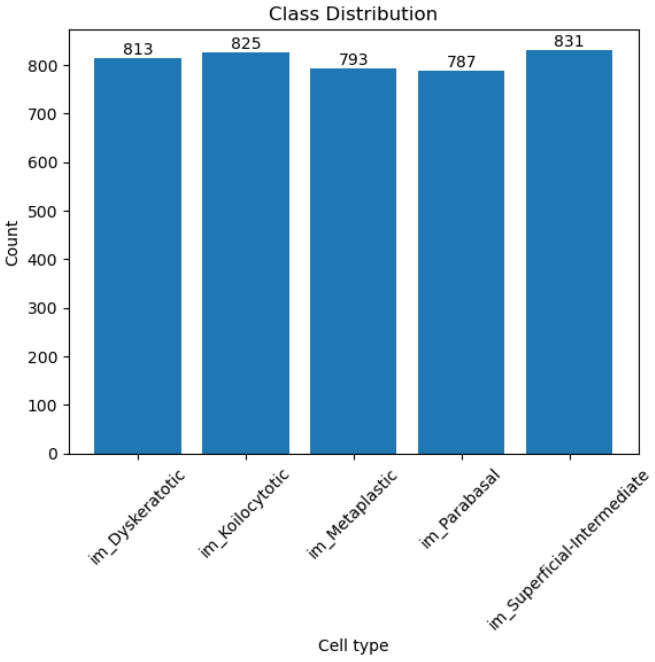

Fig. 1. Number of features for each cell type

Each image in the dataset is a color image captured using a high-resolution microscope. These images are stored in BMP format with various dimensions. They also provide detailed visual representations of individual cells, with key features like cell shape, nucleus size and texture, which are important for classification.

To analyse the data itself, a boxplot was created to visualize the distribution of mean pixel intensities for each class. The mean pixel intensity is calculated as the average grayscale value of all pixels in an image. Figure 2 shows the distribution of these values across the classes.

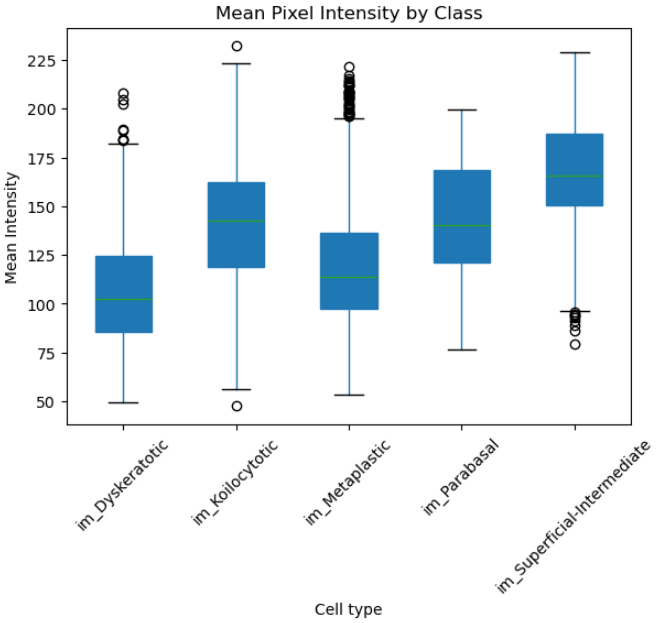

Fig. 2. Mean pixel intensity for each cell type

The x-axis represents the different classes of cervical cells and the y-axis shows the mean pixel intensity.

This plot provides valuable insights into the dataset. For instance, significant differences in the distributions between classes may make it easier for the machine learning algorithms to distinguish images and make more accurate predictions.

The dataset is split into two parts: 80% for training and 20% for testing. This ensures that the models are evaluated on unseen data, which helps in assessing their generalization ability.

V. Methodology

As previously mentioned, three machine learning algorithms were used for this project, those being Convolutional Neural Networks (CNN), Light Gradient Boosting Machine (LightGBM) and Support Vector Machines (SVM).

During training, we monitored the models' performance by checking the accuracy and loss for both the training and testing phases, while precision, recall and f1-score were calculated for the test set only. We also tracked the computation time required to run the entire algorithm.

A. Convolutional Neural Network

In the last project, we used Artificial Neural Networks (ANN) to solve a regression task, more specifically to predict a student's grade based on various factors. For this project, we switched to CNNs for image analysis.

CNNs are a specialized form of neural networks, particularly effective in dealing with image data due to their ability to capture spatial relationships in images through convolutions and pooling layers. CNNs are great at extracting features such as edges, textures, and shapes, which are crucial for tasks like image classification, object detection, and segmentation.

We started by setting up the file paths to the cell types, then proceeded to load and preprocess the images, resizing the images to a size of 128x128 pixels, and then normalizing the pixel values, between 0 and 1. After that, we converted the features (images) and labels into numpy arrays. The labels were encoded by mapping each cell type to a unique index, followed by converting the encoded labels into one-hot encoded labels. Next, we reshaped the data and ensured it was properly normalized before testing.

Then we built the CNN model, starting with a convolutional layer that extracts basic features from the input images, followed by MaxPooling to reduce their size. This process is repeated in subsequent layers, with the number of filters increasing to capture more complex features. Each convolutional layer uses the ReLU activation function to add non-linearity, allowing the model to learn more complex features. After the convolutional layers, the output is flattened and passed through a dense layer for further interpretation. A dropout layer is used to prevent overfitting. Finally, the output layer uses a Softmax activation to classify the image into one of the five cell types based on learned patterns. Finally, we compiled the model using the Adam optimizer, with a cross-entropy loss function, and trained it.

The following figures show the accuracy and loss values during the training and testing process:

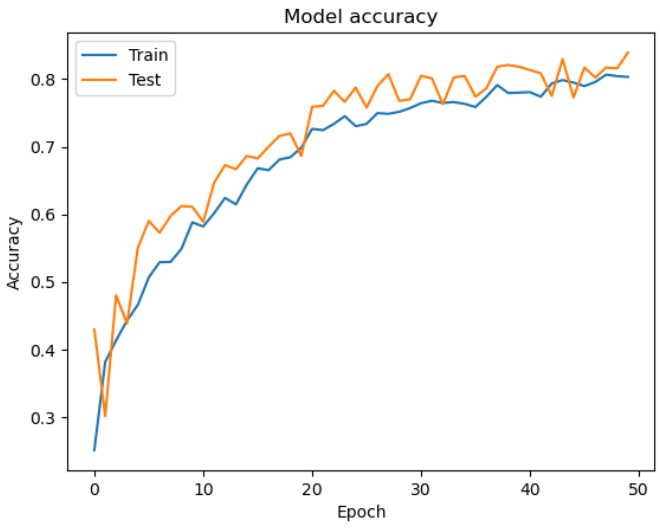

Fig. 3. Training and Testing Accuracy in the CNN model

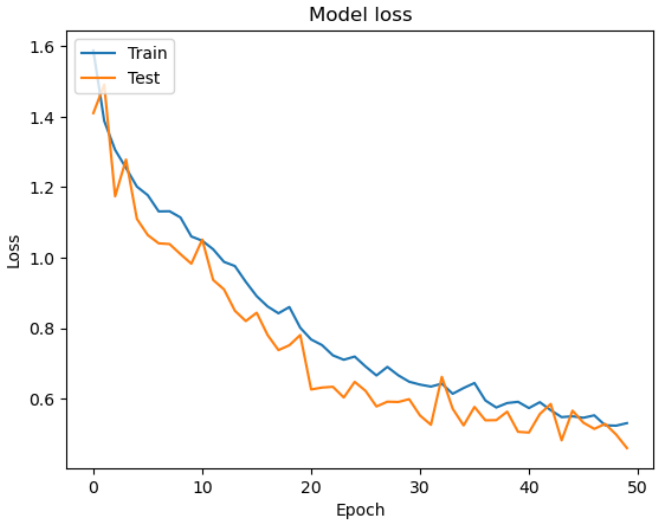

Fig. 4. Training and Testing Loss in the CNN model

The confusion matrix in Figure 5 summarizes the performance of the model by showing the true and predicted class distributions. Cell type *im_Dyskeratotic* shows the best classification results. In contrast, cell types *im_Metaplastic*, *im_Parabasal* and *im_Superficial-Intermediate* show a higher confusion among themselves.

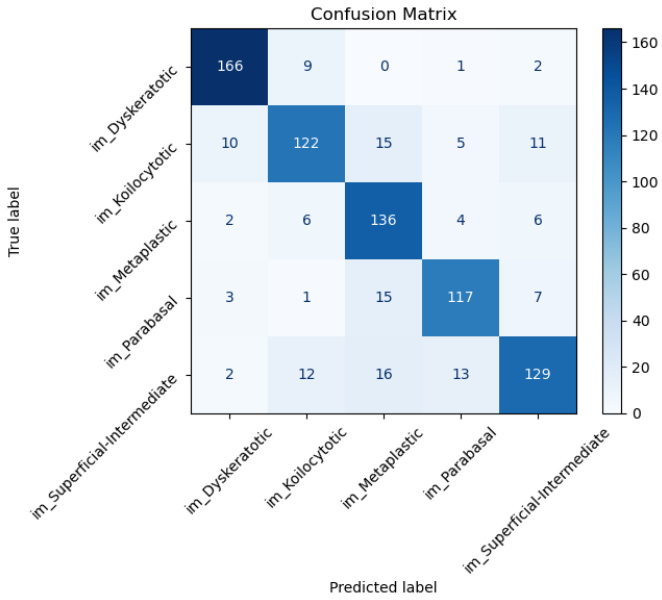

Fig. 5. Confusion matrix of the CNN model

The ROC curve in Figure 6 shows how well the model balances true positives against false positives for each class, where each class is compared with all the others. The Area Under the Curve (AUC) is used to measure the model's ability to distinguish between each class and all others, with values closer to 1 indicating almost perfect discrimination.

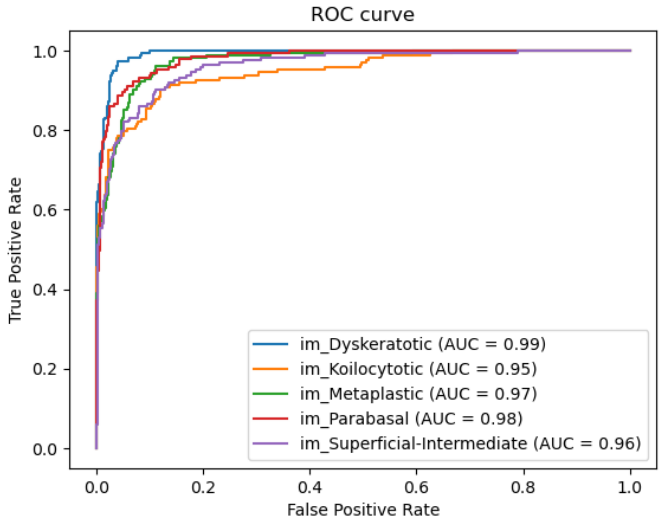

Fig. 6. ROC curve of the CNN model

The stacked bar plot in Figure 7 shows how well the model did for each cell type, showing correct vs. incorrect predictions. Each bar represents a different cell type, and the height of the bar shows the total number of predictions for that class.

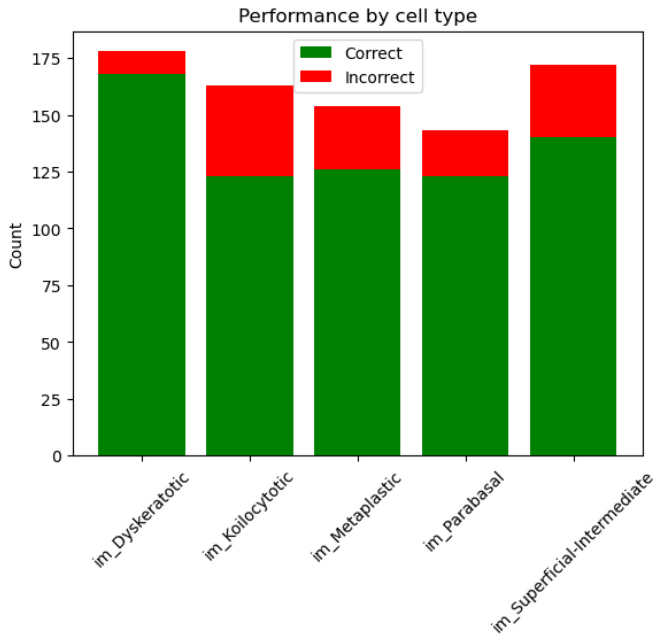

Fig. 7. Performance of each class in the CNN model

B. Light Gradient Boosting Machine

LightGBM is a gradient-boosting framework that uses decision trees for prediction tasks. It builds an ensemble of decision trees where each tree tries to correct the errors of the previous one. It is highly efficient, and is particularly effective for tabular datasets.

The file setup and preprocessing is very similar to the previous model, with a slight different in the image reading. Initially, we tried grayscaling, but the results ended up not being as good as we hoped, so we switched to colored images. Then we set them to 128x128 pixels like in the CNN model and flattened into a numpy array. The dataset is then wrapped into a Dataset object and the model hyperparameters are defined, with Gradient Boosting Decision Trees (GBDT) selected as the boosting method. Lastly, the model is trained.

The following figures show the accuracy and loss values during the training and testing process, as well as the confusion matrix, ROC curve and the performance for each class.

Fig. 8. Training and Testing Accuracy in the LightGBM model

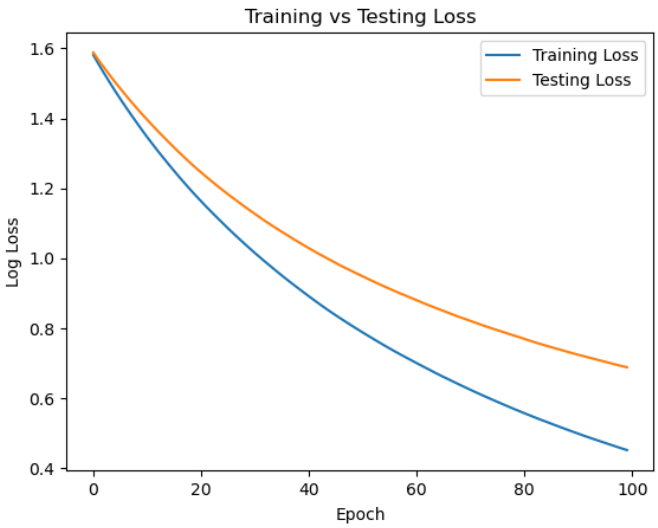

Fig. 9. Training and Testing Loss in the LightGBM model

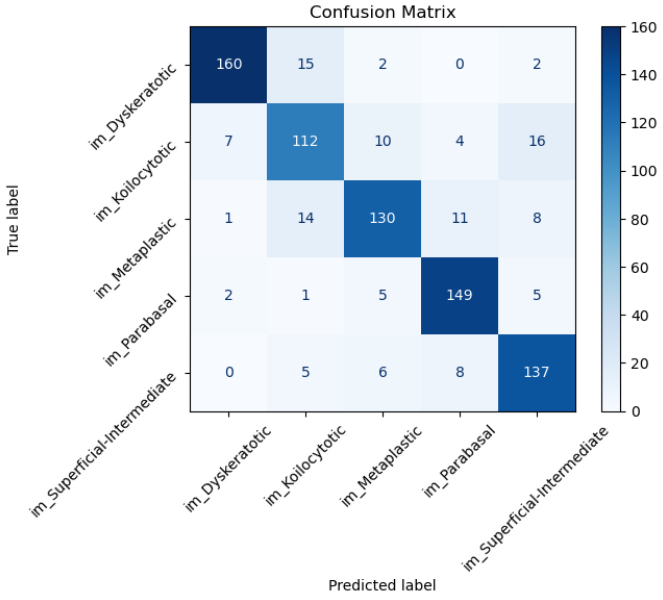

Fig. 10. Confusion matrix of the LightGBM model

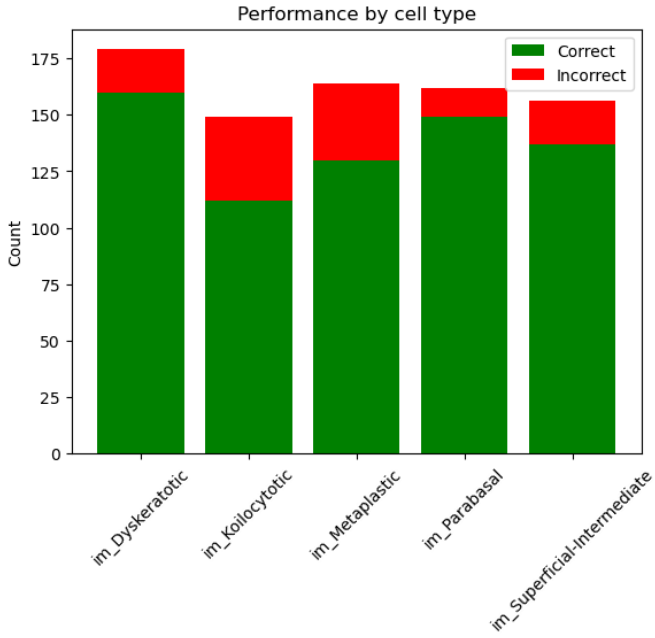

Fig. 12. Performance of each class in the LightGBM model

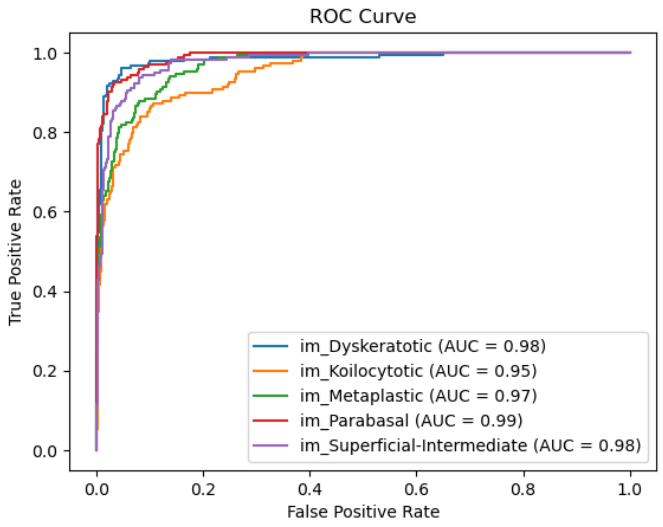

Fig. 11. ROC curve of the LightGBM model

C. Support Vector Machine

SVMs are supervised learning models that find the optimal hyperplane to classify data points in high-dimensional spaces. They are effective for smaller datasets with a clear margin of separation.

We started by loading Residual Network with 50 layers (ResNet50), a deep convolutional neural network pre-trained on ImageNet, a large visual database. ResNet50 is designed to handle deep architectures using skip connections, a ”shortcut” that bypasses one or more layers in a deep neural network and directly connects an earlier layer’s output to a later layer. For this project, it was used as a feature extractor. It starts off similarly to what we’ve previously done in image preprocessing and feature extraction, with a difference in resize, to 224x224 pixels, as required for ResNet50. Converts the data into numpy array as per usual, splits the data and normalizes the features. Then we used Stochastic Gradient Descent (SGD) as our classifier. While ResNet50 extracts meaningful features, SGDClassifier finds the best hyperplane using these features to classify the cell types. Lastly, we train the model.

The following figures show the accuracy and loss values during the training and testing process, as well as the confusion matrix, ROC curve and the performance for each class.

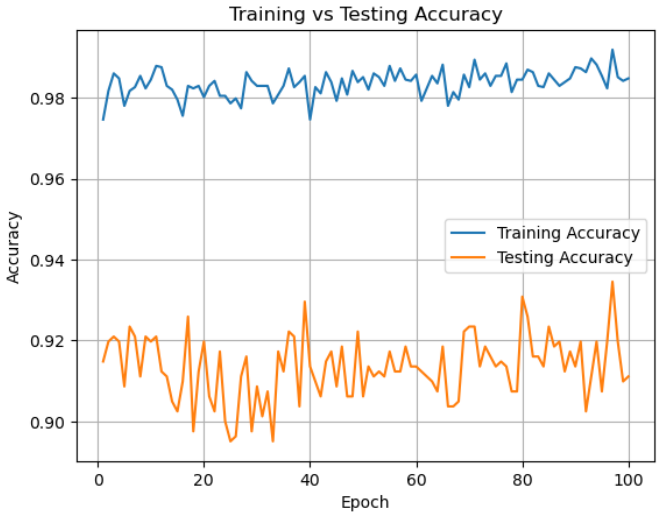

Fig. 13. Training and Testing Accuracy in the SVM model

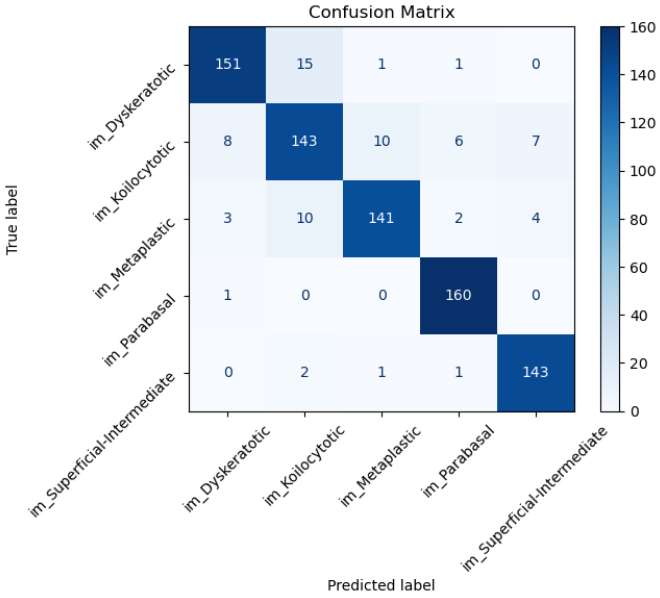

Fig. 15. Confusion matrix of the SVM model

Fig. 14. Training and Testing Loss in the SVM model

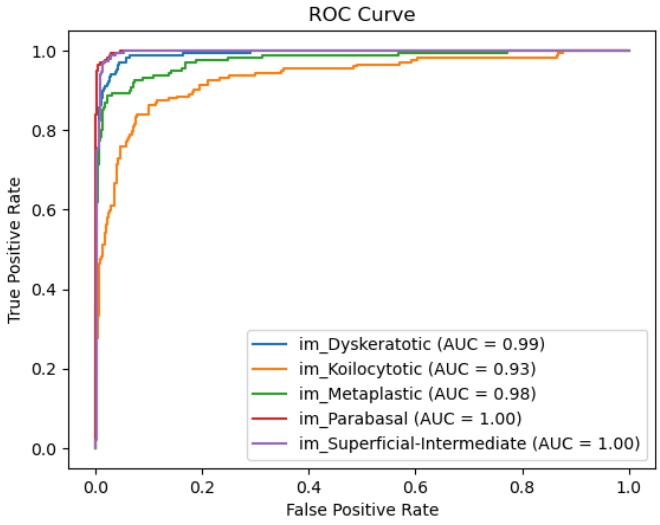

Fig. 16. ROC curve of the SVM model

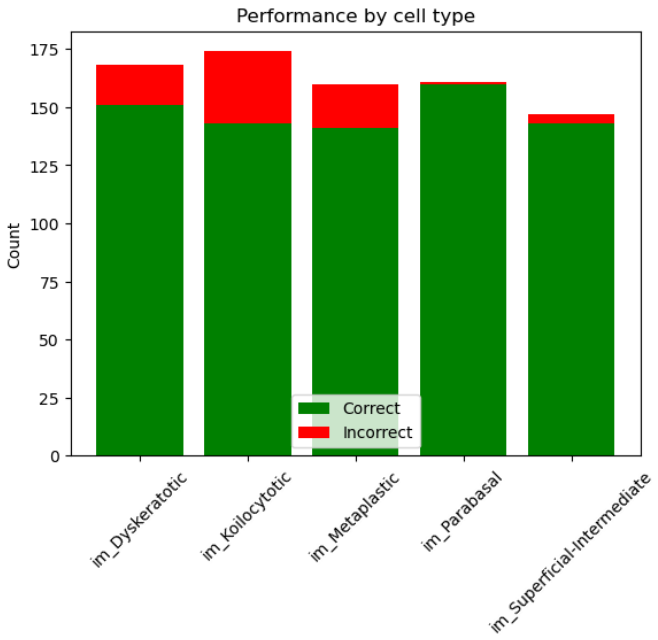

Fig. 17. Performance of each class in the SVM model

VI. Results

The performance of each method was evaluated using accuracy, loss, precision, recall and f1-score. Table I presents the results obtained from all models.

TABLE I
Performance Metrics

Metric	CNNs	LightGBM	SVMs
Accuracy	83.95%	84.94%	91.11%
Loss	45.99	0.6880	0.0889
Precision	84.08%	85.05%	91.05%
Recall	83.95%	84.94%	91.11%
F1 Score	83.82%	84.92%	91.04%
Computation Time	1653.13s	922.82s	999.86s

The results demonstrate that all models performed reasonably well, with SVM outperforming both CNN and LightGBM on all metrics, except for a small difference in computation time. This performance is most likely due to ResNet50, as it is already pre-trained in a much larger dataset.

CNN however has two big flaws, its loss value and computation time. One of the reasons for the loss value could be because of overconfidence in wrong predictions from the model. It also uses a categorical cross-entropy loss function, that could be sensitive to incorrect classifications. As for the computation time, CNNs tend to be computationally demanding because of its architecture. Using ResNet50 for this model most likely would have significantly improved the loss value, while also increasing all other metrics.

VII. Comparison With Related Work

We've also compared our work with other author's work using the same dataset. In the Kaggle notebook by Prakhar Pipersania [3], the author used a CNN model to make his predictions, also with the Adam optimizer but with different layers and activation functions, obtaining an accuracy of around 91.29%.

Another author is Ashish Ram [4], that also made a CNN model, and additionaly, used ResNet50 as a feature extractor, obtaining an accuracy of around 95% as shown in his classification report. The improvement in the CNN model in his case, could be once again, because of ResNet50's pre-trained nature.

Further improvements to our models could involve using a pre-trained model in classifying images, and then applying it to our own dataset. Defining better hyperparameters, as well as data augmentation would most likely lead to more pleasing results.

VIII. Conclusion

This study compared CNN, LightGBM, and SVM to solve an image classification problem, more specifically to analyse type of cells studied with diagnosing possible cervical cancer. While SVM showed the best overall metrics, LightGBM did not fall too far behind. CNN ended up being the worst model in terms of performance, where the solutions mentioned previously could most likely improve it.

Machine learning has become an essential tool in medical imaging, as well as in numerous other fields, completely chaning how data is analysed and interpreted. It aids in diagnosing diseases, segmenting medical scans or classifying abnormalities with high accuracy. These models help automate and enhance diagnostic processes, reducing human error and improving efficiency in healthcare.

References

[1] W. H. Organization, "Cervical cancer," https://www.who.int/news-room/fact-sheets/detail/cervical-cancer, 2023, accessed: 05-01-2025.
[2] P. Mehandiratta, "Cervical cancer largest dataset sipakmed," https://www.kaggle.com/datasets/prahladmehandiratta/cervical-cancer-largest-dataset-sipakmed, 2021, accessed: 03-01-2025.
[3] P. Pipersania, "Cervical cancer prediction," 2021, accessed: 11-01-2025. [Online]. Available: https://www.kaggle.com/code/prakharpipersania/cervical-cancer/notebook
[4] A. R. J. A, "Cervical cancer detection and classification," 2024, accessed: 12-01-2025. [Online]. Available: https://www.kaggle.com/code/ashishramja/cervical-cancer-detection-and-classification/notebook