POLITECNICO
MILANO 1863

AN2DL - First Homework Report
LionIA

Malvin Noël, Léandre Le Bizec, Satvik Bisht, Subadip Banerjee

malvin, leandre, satvikbisht, subhadip

11077886, 10990212, 10886954, 11045700

November 24, 2024

1 Introduction

In this project, we address a *multi-class classification problem* involving the categorization of blood cells into eight distinct classes, each representing a unique cell state. The objective is to develop robust neural network models capable of accurately predicting the class label for each blood cell image. This requires overcoming challenges such as dataset imbalances, image quality variations, and limited resolution.

To achieve this, we followed a systematic approach comprising several key steps:

- **Dataset Analysis:** Identifying and addressing outliers, duplicates, and class imbalances.
- **Baseline Model Development:** Building a neural network from scratch as a starting point.
- **Data Augmentation:** Applying augmentation techniques to improve model generalization and performance.
- **Transfer Learning:** Leveraging pre-trained architectures to enhance classification accuracy.
- **Fine-tuning:** Optimizing pre-trained models for the specific task at hand.

This report focuses on the implementation, evaluation, and comparison of both custom-built and transfer learning-based deep learning models to achieve high classification accuracy in this challenging task.

2 Problem Analysis

Our initial analysis revealed several key characteristics and challenges:

2.1 Dataset Characteristics

- 13,759 images (96x96x3 pixels)
- 8 distinct cell classes
- Significant class imbalance (ranging from 1,049 to 2,530 samples per class)
- Image quality variations

2.2 Main Challenges

1. Class imbalance affecting model bias
2. Image quality variations and noise
3. Limited image resolution (96x96)
4. Presence of duplicates and outliers

3 Method

Our methodology consisted of three main components: data preprocessing, model development, and iterative improvement.

3.1 Data Preprocessing

We implemented a comprehensive preprocessing pipeline:

$$I_{processed} = S(C(L(I_{raw}))) \tag{1}$$

where L is illumination correction, C is contrast enhancement, and S is cell segmentation.

- **Illumination Correction:** Used morphological operations and Gaussian blurring to adjust uneven lighting.
- **Contrast Enhancement:** Applied CLAHE to improve feature visibility.
- **Cell Segmentation:** Isolated cells with thresholding and morphological techniques.
- **Augmentation:** Included random transformations, such as flipping and rotation, to enhance generalization.

3.2 Model Evolution

We progressively developed more sophisticated models:

1. **Basic CNN with cross-entropy loss:**

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N} y_i \log(\hat{y}_i) \tag{2}$$

   We started with a simple CNN architecture, experimenting with different configurations of layers, neurons, and activation functions. While this model performed well on the training set, it showed significant overfitting on Codabench due to its limited capacity to generalize across complex class distributions.

2. **ConvNeXt-Base:** We transitioned to ConvNeXt-Base for transfer learning, utilizing dropout and early stopping to address overfitting. However, despite unfreezing layers and adding a dense layer, the model continued to overfit and achieved a lower performance of **38%** on Codabench, indicating minimal improvement over the baseline.

3. **Advanced architectures with RandAugment**:

$$x_{aug} = T(x, N, M) \tag{3}$$

   where N is augmentations per image and M is magnitude.

4. **ResNet50V2:** ResNet50V2 was introduced for transfer learning due to its robust residual connections, which mitigate vanishing gradient issues and enhance training depth. We fine-tuned the last layers and used class weights to handle imbalances. This approach improved stability and generalization, achieving a performance of **77%** on Codabench, but still fell short of the desired accuracy.

5. **EfficientNetV2B0:** As our final model, we employed EfficientNetV2B0 due to its scalable architecture and efficient handling of small image resolutions (96x96). We conducted extensive hyperparameter tuning, adjusting learning rates, dropout rates, and freezing different layers to optimize performance on our preprocessed dataset. This model delivered the best results, achieving a test accuracy of **87%**. The model accuracy and loss are shown in figure 1.

4 Experiments and Results

4.1 Data Cleaning Results

The dataset was analyzed and cleaned to improve model performance. Table 1 summarizes the data cleaning statistics.

Table 1: Dataset Cleaning Statistics

Category	Count
Original Images	13,759
Duplicates Removed	2,915
Outliers Removed	314
Final Dataset	10,530

4.2 Model Performance

Extensive experiments were conducted with various architectures and training strategies. Table 2 highlights the performance of different models.

Table 2: Model Accuracy Comparison.
Best model is highlighted in bold.

Model	Training	Validation	Test
Basic CNN	99.18	96.29	28
ConvNeXtBase	87.60	93.04	38
ResNet50V2	96.39	98.05	77
EfficientNetV2B0	**91.62**	**98.01**	**87**

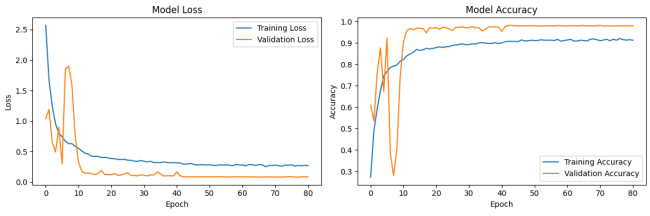

Figure 1: Model Loss and Accuracy During Training

5 Discussion

5.1 Key Improvements

The following improvements contributed significantly to the results:

- **Enhanced data augmentation with increased magnitude:** Data augmentation techniques such as random rotations, flips, and brightness adjustments improved model generalization by making the training data more representative of real-world variations.

- **Optimizer selection (Lion vs Adam):** The Lion optimizer provided better convergence and improved performance on validation data compared to the Adam optimizer, especially for fine-tuning pre-trained models.

- **Architecture selection for small images:** EfficientNetV2B0 architectures were particularly effective for small image resolutions like 96x96 due to their scalability and ability to capture fine details without overfitting. [3]

- **Thorough preprocessing pipeline:** Preprocessing steps, including duplicate and outlier removal, ensured cleaner and more balanced data for training, which helped the model focus on meaningful patterns. [2]

5.2 Limitations

Despite the improvements, some limitations remain:

- **Limited image resolution:** The 96x96 resolution constrained the level of detail available for the model to learn, potentially limiting classification accuracy for complex cell structures.

- **Class imbalance effects:** Although weighted loss functions and oversampling were used, the significant class imbalance in the dataset still affected the model's ability to generalize well across all classes.

- **Computational constraints:** Fine-tuning large pre-trained architectures required significant computational resources, which limited the exploration of more complex models or ensemble techniques [1].

6 Conclusions

We successfully developed a robust blood cell classification system achieving **87% accuracy** on the submission set. Key contributions include:

- **Effective preprocessing pipeline:** The pipeline addressed data quality issues and ensured a cleaner training set.

- **Optimized augmentation strategy:** Data augmentation improved the generalization ability of the models, reducing overfitting.

- **Architecture selection methodology:** Leveraging scalable architectures like EfficientNetV2B0 provided strong performance even with low-resolution images.

- **Comprehensive model comparison:** A detailed evaluation of different architectures helped identify the most effective model for the task.

Future work could explore:

- **Ensemble approaches:** Combining predictions from multiple architectures could improve accuracy by leveraging the strengths of each model [1].

- **Advanced segmentation techniques:** Implementing cell-specific segmentation could provide cleaner inputs, improving classification accuracy further.

- **Class-specific augmentation strategies:** Tailoring augmentation to specific classes could help balance the performance across all categories.

- **Higher resolution images:** Using higher-resolution images might provide more details, enabling the models to achieve better accuracy.

7 Team Contributions

The project was a collaborative effort, with each member contributing significantly to its successful completion:

- **Malvin Noël:** Optimized data preprocessing by removing outliers, resizing images, and enhancing quality with bilateral filtering, illumination correction, and contrast adjustment. Explored CutMix and fine-tuned data augmentation strategies. Developed a CNN and fine-tuned a transfer learning model using EfficientNetV2B0 to enhance performance.

- **Léandre Le Bizec:** Explored various architectures without transfer learning to establish a baseline for the project. Investigated the impact of dropout and regularization techniques, as well as data augmentation strategies. Subsequently joined the team in leveraging transfer learning, fine-tuning models like EfficientNetV2B0. Additionally, identified optimal hyperparameters for final submissions, contributing to enhanced model performance.

- **Satvik Bisht:** Performed preprocessing with outlier removal, applied data augmentation, built CNNs with batch normalization and dropout, and used ConvNeXt-Base for transfer learning with early stopping. Fine-tuned EfficientNetV2B0 with hyperparameter tuning.

- **Subadip Banerjee:** Assisted in refining the model's architecture and contributed to its development process. By utilizing a confusion matrix, I helped analyze misclassifications, leading to improvements in the model's performance. Moreover, supported the documentation and packaging of the model for deployment by fine-tuning the EfficientNetV2B0 model for multi-class classification.

References

[1] T. G. Dietterich. Ensemble methods in machine learning. *Multiple classifier systems*, pages 1–15, 2000.

[2] Keras. Kerascv: Computer vision extensions for keras. `https://keras.io/guides/keras_cv/`, 2024. Accessed: 2024-11-24.

[3] M. Tan and Q. V. Le. Efficientnetv2: Smaller models and faster training. *arXiv preprint arXiv:2104.00298*, 2021.