AN2DL - First Homework Report
Hyperbee

Cemal Gunduz, Sila Saraoglu, David-Alexandru Fota

cemalgndzz, silasaraoglu, fotadavid23

270347, 257732, 276388

November 24, 2024

1 Introduction

This report aims to demonstrate the Image Classification project in Artificial Neural Network and Deep Learning course. The goal is to create a Convolution Neural Network that classifies eight different blood cell classes from the given dataset with high accuracy. The dataset is composed of 13759 images in the shape of 96 x 96-pixel size in RGB color format.

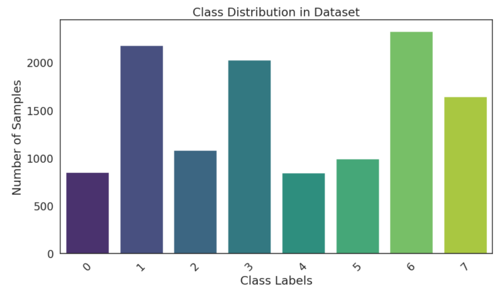
Chosen approach is to start with simple CNN models and check their performance. Afterward, we develop our bigger model step by step to more advanced forms by using several techniques

2 Problem Analysis

2.1 Dataset Characteristics

The dataset comprises 13,759 images with dimensions of 96 x 96 pixels in RGB color format. Initial analysis revealed the presence of outlier images that were not representative of typical blood cell images, including images of characters such as Shrek and trolls, which were intentionally included in the dataset. These outliers were removed prior to any further pre-processing. Additionally, duplicate images were eliminated, resulting in a refined dataset containing 11,959 images.

A significant characteristic of the dataset influencing model performance is the class imbalance among the blood cell image categories. The class distribution is illustrated in the following figure:



2.2 Main Challenges

Several challenges were encountered in achieving high model accuracy. One of the primary challenges was the class imbalance in the dataset, which necessitated pre-processing steps to create a balanced dataset suitable for training. Various pre-processing techniques, including data augmentation, were considered to address this issue. Furthermore, computational storage was limited thus it was a challenge to use deeper model types for the training.

2.3 Initial Assumptions

It was assumed that the provided dataset, after the removal of evident outliers (e.g., non-blood cell images such as trolls), accurately represents the blood cell types intended for classification. Additionally, all labels associated with the images were assumed to be correct and free from significant errors. Also models trained on the local dataset were assumed to generalize well to the evaluation dataset, provided sufficient augmentation and fine-tuning strategies were applied.

3 Method

Initially, before progressing to advanced models, a basic CNN model was tested to evaluate its performance on the dataset. Multiple CNN architectures were constructed with convolution layers followed by max-pooling, a flattening layer, and a classification layer. Early experiments with this models demonstrated high accuracy on the local dataset but poor performance on the test set, indicating potential over-fitting. To address this, dropout layers were incorporated, resulting in an accuracy of 20%. These results indicated that a simple CNN model was insufficient for achieving high accuracy. Developing a custom feature extraction approach was deemed impractical due to time and resource constraints, leading to the adoption of transfer learning.

3.1 Transfer Learning

For transfer learning, ImageNet-pretrained weights were utilized, and several architectures were explored, including Inception [1], ResNet [2], MobileNetV2 [3], and ConvNeXtLarge [4]. All models employed a Softmax activation function in the output layer, while hidden layers used ReLU or Swish activations. As noted in [5], the combination of Softmax and ReLU can improve accuracy, and [6] demonstrates that Swish outperforms ReLU in specific scenarios. To connect the feature extraction layers with the classifier, a Global Average Pooling layer was implemented. Dropout layers were also included to mitigate over-fitting.

3.2 Optimizer Selection

The Adam optimizer was primarily employed due to its robust performance across various scenarios, as outlined in [7]. Adam's adaptive learning rate mechanism enables efficient training, making it a popular choice. For the final model, which achieved the best performance, the AdamW optimizer was adopted. As described in [8], Adam's weight regularization scales gradients by the adaptive learning rate, leading to uneven regularization effects. AdamW addresses this limitation by decoupling weight decay from gradient updates, applying it directly to parameters. This ensures consistent regularization, enhancing generalization and stability during training. The use of AdamW significantly contributed to the improved performance of the final model.

Additionally, the Lion optimizer with weight decay was explored due to its innovative approach to gradient updates. However, it was observed that for highly augmented datasets, such as the one used in this project, the convergence capabilities of Lion were less effective compared to AdamW which is also described in [9]. This limitation made Lion less suitable for the final model configuration.

3.3 Fine Tuning

To enhance the performance of the models, fine-tuning was implemented. Experiments involved unfreezing specific layers, particularly the last layers, to make them trainable.

3.4 Class Imbalance and Augmentation

Class imbalance posed a significant challenge in the dataset. Various strategies were explored to address this issue. Initial attempts included undersampling, which, while computationally efficient, yielded unsatisfactory accuracy and was not pursued further. Class weights were then calculated based on each class's representation in the dataset and incorporated into the training process. This approach improved accuracy by balancing the model's attention across minority and majority classes but did not achieve the desired results.

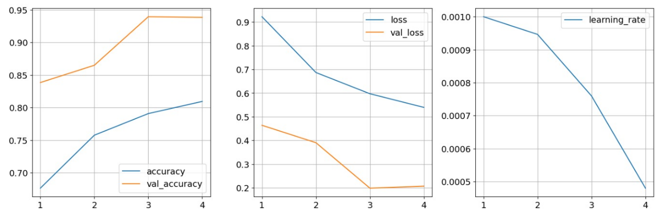
The most effective approach involved a combination of offline and online data augmentation. Offline augmentation was applied to equalize class sample counts before training, utilizing mixed and random augmentation techniques. These methods effectively balanced the dataset, achieving the highest accuracy among all strategies. Mixed augmentation techniques blended images or replaced portions of one image with patches from another, enriching the training samples' diversity [10, 11].

For our best-performing model, we used the Keras library's RandAugment and RandFlip as augmentation techniques. RandAugment has been shown to improve image classification results across numerous datasets and implements augmentations such as identity, autoContrast, equalize, rotate, solarize, color, posterize, contrast, brightness, sharpness, shear-x, shear-y, translate-x, and translate-y.

In our training pipeline, we preprocessed the data with RandAugment and RandFlip and applied the augmentation to approximately 80% (8/10) of the images by adjusting the RandAugment parameter values to control the severity and number of applied augmentations. This approach significantly enhanced the variability of the training samples, leading to better model generalization and improved classification performance.

4 Experiments

This section presents experiments conducted with the best-performing model, ConvNeXt-Large. The performance of the models is compared to illustrate progress and highlight the differences in accuracy across architectures.



The experiment visualizes the accuracy and loss for the training and validation sets over epochs. As shown, increasing the number of epochs improves accuracy. Additionally, a learning rate scheduler was employed, causing the learning rate to decrease as accuracy improved. At then end, our best model ConvNeXtLarge trained 10 epochs with frozen layers (because of the long training time as RandAugment is bottleneck), then fine-tuned 3 epochs by unfreezing the last 30 layers of the model.

Below table shows the experimented models and their peak accuracies from the experiments.

Table 1: Models and their peak Accuracy

Model Name	Accuracy [%]
Custom CNN	20±1
InceptionV3 (without augmentation)	21
MobileNetV2(without augmentation)	20±1
NasNetMobile	40
ResNet50V2	67
ConvNeXtLarge	88

5 Results

As shown in the table in experiments part, we followed a step-by-step approach to develop our models, progressively refining their performance. Among the architectures we tested, ConvNeXtLarge achieved the highest accuracy, primarily due to the integration of data augmentation and fine-tuning.

In this project, we sometimes observed a discrepancy between local and evaluation set performance. While our models achieved high accuracy locally, their scores were comparatively lower on the evaluation set. Despite the evaluation results suggesting potential over-fitting, we did not believe this was the root cause. To address the issue, we adopted more aggressive data augmentation strategies, aiming to bridge the gap between local and evaluation performance and enhance the robustness of our models.

6 Discussion

One of the strengths of our model is the pre-processing part. We discussed all possible options and decided to implement online and offline augmentation. Furthermore, we used random and mixed augmentation together to strengthen our dataset. We believe that we try a good range of models from the simplest to more complex ones thus the best-performing model will be efficient for any other evaluation test set as well. In terms of weaknesses, since we use augmentation and complex models to achieve high accuracy, computational resources to train our model are high and the time needed is much. The computational resources and time played a role as a limitation to us. If we had more computational resources in Colab or any other platform, we believe that we could achieve higher accuracy results in general.

7 Conclusions

To conclude, we conducted extensive experiments to classify eight distinct blood cells, achieving the highest accuracy of 86% using ConvNeXtLarge with online RandAugment and RandFlip augmentations from keras to address data imbalance. For future work, exploring models like ConvNeXtXLarge, applying more extensive augmentation, or utilizing datasets like TXL-PBC[12] could enhance performance.

Our team collaborated on training models with diverse pre-processing techniques and approaches, sharing methods to improve outcomes. While Cemal with the ConvneXtLarge achieved the highest submission accuracy, with detailed information provided in earlier sections, Sila and David achieved moderate results. Finally, Sila and Cemal gathered all the results and wrote the report.

References

[1] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, ”Going Deeper with Convolutions,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1–9.

[2] K. He, X. Zhang, S. Ren, J. Sun, ”Deep Residual Learning for Image Recognition” *arXiv preprint arXiv:1512.03385*, 2015.

[3] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, ”MobileNetV2: Inverted Residuals and Linear Bottlenecks,” *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 4510–4520.

[4] Z. Liu, H. Mao, C. Wu, C. Feichtenhofer, T. Darrell, and S. Xie, ”A ConvNet for the 2020s,” *arXiv preprint arXiv:2201.03545*, 2022.

[5] D. P. Kingma and J. Ba, ”Adam: A Method for Stochastic Optimization,” *arXiv preprint arXiv:1412.6980*, 2014.

[6] X. Yi, P. Zhang, and R. T. Qiu, ”A New Augmentation Strategy for Improving Deep Neural Networks in Natural Language Processing,” *arXiv preprint arXiv:2002.04060*, 2020.

[7] S. DeVries and G. W. Taylor, ”Improved Regularization of Convolutional Neural Networks with Cutout,” *arXiv preprint arXiv:1710.05941*, 2017.

[8] I. Loshchilov, F. Hutter, ”Exploring the Benefits of Augmentation in Deep Learning Models,” *OpenReview.net*, 2024. [Online]. https://openreview.net/pdf?id=Bkg6RiCqY7.

[9] Xiangning Chen et al. ”Symbolic Discovery of Optimization Algorithms”. *arXiv: 2302.06675*, 2023.

[10] Y. Wang, Z. Chen, and H. Lee, ”Enhancing Image Recognition with Advanced Augmentation Techniques,” *Multimedia Tools and Applications*, vol. 83, no. 4, pp. 1–18, 2024. [Online]. Available: https://link.springer.com/article/10.1007/s11042-024-18868-8.

[11] M. Zhang and T. Huang, ”A Survey of Modern Optimizers and Regularization Techniques,” *Artificial Intelligence Review*, vol. 55, no. 2, pp. 345–368, 2022. [Online]. Available: https://link.springer.com/article/10.1007/s10462-022-10227-z.

[12] Y. Gao, X. Xue, J. Jiang, J. Li, and Z. Chen, ”TXL-PBC: a freely accessible labeled peripheral blood cell dataset,” 2024. [Online]. Available: https://arxiv.org/abs/2407.13214.