# AN2DL - First Homework Report
# OverfittingNinjas

Alessandro Griffanti, Andrea Ciarallo, Eleonora Bellè, Luca Masiero

alessandro, mose17, eleonorabelle, luca15

250114, 226500, 225900, 245458

November 24, 2024

## 1 Introduction

This project focuses on image classification using deep learning techniques. The main objective is to design a robust model that classifies 96x96 RGB images of blood cells into their respective categories, assigning the correct class label to each RGB image.

Our approach, that will be described in detail in Section 3, utilizes a convolutional neural network (CNN) built upon a pre-trained architecture, enhanced by custom data augmentation techniques. This strategy leverages the strengths of transfer learning while addressing the unique characteristics of the dataset.

## 2 Problem Analysis

The dataset for this project consists of 13,759 RGB images of size 96x96, each representing a blood cell, organized into the following eight categories: Basophil, Eosinophil, Erythroblast, Immature Granulocytes, Lymphocyte, Monocyte, Neutrophil, and Platelet.
After analysing the dataset, we noticed some corrupted images —beginning at index 11,959— where the original visuals are overlaid with unrelated content, introducing noise that needs to be addressed to ensure data integrity.

The main challenges encountered include class imbalance, as some cell types are underrepresented, which makes it difficult for the model to learn their distinct features effectively. Additionally, given the limited dataset size, there is a considerable risk of overfitting, particularly when training on high-dimensional RGB images.

Our initial assumptions were that transfer learning with pretrained CNNs, such as ConvNeXtLarge or VGG, would enable robust feature extraction. Moreover, the local dataset does not necessarily reflect a realistic class distribution; therefore, we anticipated that class weights would be essential to rebalance it, in addition to augmentation techniques like rotations, zooms and flips, providing the model with a wider and more effective set of training samples. Lastly, given the presence of corrupted images as stated, we considered essential to remove them to maintain dataset integrity, reduce bias, and improve model reliability.

## 3 Method

The following approach represents our best performing solution, though we also evaluated a range of alternative strategies, which we will discuss in detail in Section 4.

The proposed approach utilizes a convolutional neural network (CNN) architecture that combines *transfer learning* with advanced data augmentation techniques and class weights to address the specific challenges identified in Section 2, including class imbalance and the risk of overfitting. The model leverages ConvNeXtLarge, a state-of-the-art, pre-trained neural network from Keras, which, initialized with ImageNet weights, serves as the primary feature extractor. This enables us to benefit from the generalizable features learned on a large dataset while adapting the network to the unique characteristics of our blood image dataset through *fine-tuning.* Specifically, we fine-tuned a portion of the convolutional layers of ConvNeXtLarge, allowing the model to adjust the pre-trained weights to better capture the specific features of blood images.

As said, to mitigate class imbalance and increase data diversity, we implemented a comprehensive data augmentation pipeline incorporating several layers such as *AugMix* and *RandAugment*, each contributing a variety of transformations, enhancing the model's robustness. These augmentations generate diverse samples that help the model, along with the use of class weights, both to generalize better and reduce the effect of class imbalance. Additionally, we applied a simpler augmentation layer, including random rotations, zooms and contrast adjustments, just before the images are passed into ConvNeXtLarge, introducing variability at the input stage.

The model architecture then incorporates a fully connected dense layer with softmax activation to produce class probabilities, facilitating accurate classification. A dropout layer is also added before this dense layer, randomly deactivating neurons during training with a certain probability to help prevent overfitting.

For further robustness, we employed *test-time augmentation* (TTA) to improve model performance on unseen data, generating multiple transformations of each test sample to capture a more comprehensive feature set. The final prediction for each test sample is obtained by averaging predictions across these augmented copies, enhancing reliability.

Our approach achieved an accuracy, computed as:

$$\text{accuracy} = \left[\frac{\sum\left(y_{\text{ground\_truth}} == y_{\text{pred}}\right)}{\text{len}\left(y_{\text{ground\_truth}}\right)}\right] \times 100$$

where $y_{\text{ground\_truth}}$ is the set of real labels and $y_{\text{pred}}$ is the set of predictions made by the model, of **83%** on a hidden test set, demonstrating the effectiveness of combining transfer learning, targeted data augmentation, and fine-tuning techniques.

Detailed improvements attributed to each technique —data augmentation pipeline, fine-tuning, and test-time augmentation— are presented in later sections.

## 4 Experiments

The initial approach involved implementing a small convolutional neural network (CNN) comprising three convolutional layers with ReLU activations and MaxPooling operations, followed by a flattening layer and a dense output layer. This simple architecture provided a baseline for evaluating the dataset's performance characteristics, revealing a tendency toward overfitting.

Building on this initial experiment, we explored transfer learning by testing multiple pretrained networks, including MobileNetV3, VGG19, EfficientNetV2L, and ConvNeXtLarge. Consistent augmentation techniques were applied across models, and the training, validation, and test sets maintained similar sizes to ensure fair comparisons. The results of this experimental phase, summarized in Table 1, supported our decision to focus on the most promising network: ConvNeXtLarge. A detailed explanation of this network is provided in the previous section.

Table 1: Performance of the basic model and different pre-trained networks.

| Model | Validation accuracy[%] | Test Accuracy[%] | Codabench Accuracy[%] |
|---|---|---|---|
| Custom CNN | 94.49 | 94.98 | 25 |
| VGG19 | 75.17 | 76.49 | 35 |
| MobileNetV3Small | 96.24 | 89.63 | 41 |
| EfficientNetV2L | 96.15 | 96.49 | 66 |
| **ConvNeXtLarge** | **95.88** | **97.83** | **83** |

## 5 Results

Our model's results significantly surpass those of simpler approaches, such as a *Zero Rule* (ZeroR) classifier or a basic *custom network*.
While a ZeroR classifier achieves an accuracy close to 19.5%, our custom baseline network, described at the beginning of Section 4, achieved 25% accuracy, and our final transfer learning approach reached an impressive 83%.
Among the most relevant achievements in developing our best model, we can highlight:

- **Balanced Classes with AugMix + Class Weights:**
  The introduction of AugMix for data augmentation was a pivotal turning point. Even when applied to our custom network, it improved accuracy to 34%. This augmentation technique was consistently used across all experiments, laying the foundation for further improvements. Also, in order to balance the different number of occurrences for each class, we implemented class weights, making the model automatically assigning the class weights inversely proportional to their respective frequencies.

- **Transfer Learning:**
  Leveraging pre-trained models transformed the results, elevating accuracy from 34% to 74%. This demonstrates the significant advantage of starting with learned feature representations.

- **Fine-Tuning and Test-Time Augmentation:**
  Fine-tuning the pre-trained model and incorporating test-time augmentation further boosted performance, pushing accuracy from 74% to 83%. This combined strategy proved to be the most effective enhancement.

Not all the approaches yielded the desired results; the following unexpected outcomes were observed:

- **Ineffective Augmentation Techniques:**
  Certain augmentation methods, such as GridMask and CutMix, unexpectedly degraded the performance. These techniques possibly disrupted crucial spatial relationships in the blood cell images, leading to reduced model accuracy.

- **Limited Gains from Optuna:**
  Hyperparameter optimization using Optuna did not yield the expected improvements.

- **Validation Accuracy higher than Training Accuracy:**
  The validation accuracy consistently exceeded the training accuracy. This anomaly could be attributed to the extensive data augmentation used, which introduced substantial variability during training. This phenomenon may arise from the augmentation pipeline acting as a form of implicit regularization.

## 6 Conclusions

While our approach achieved good results, there is room for further refinement:

**Model Ensembles:**
Combining predictions from multiple models could potentially boost performance. However, preliminary experiments with ensembles exceeded the time constraints allowed for this project. With more computational resources, this avenue could be explored further.

**Different Data Augmentation:**
Exploring different data augmentation techniques and integrating them with the pipeline we used could boost even further the performance of our model;

**Contributions:**

- Alessandro Griffanti:
  Data augmentation pipeline definition, hyperparameters exploration with Optuna, ConvNeXtLarge, fine tuning, test time augmentation;

- Andrea Ciarallo:
  Data augmentation exploration, VGG, class weights, fine tuning;

- Eleonora Bellè:
  Data augmentation exploration, MobileNetV3, fine tuning;

- Luca Masiero:
  Custom CNN, outliers removal, data augmentation exploration, EfficientNet, fine tuning.