AN2DL - First Homework Report
Overfitting Gods

Bonanomi Emanuele, Dante Riccardo, Derme Francesco, Fumagalli Pietro

emanu, riccardodante, tunamayo, pietrofumagalli

278166, 274136, 274175, 279605

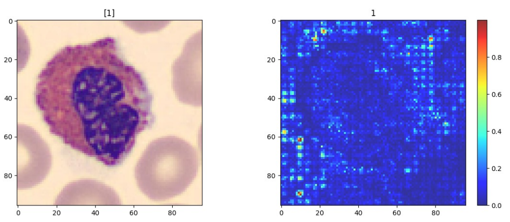
November 24, 2024

1 Introduction

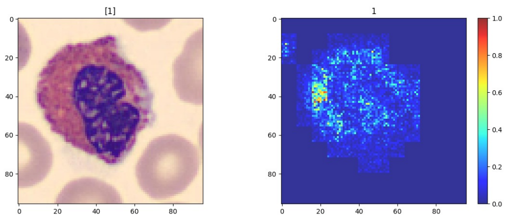
This report explains the development and implementation of a deep learning-based solution to an image classification problem. The proposed network is capable of accurately distinguishing between eight blood cell types while being highly resilient to perturbations in the testing data. The following outlines the data preprocessing, model architecture selection and training process that led to robust and reliable classification results.

2 Experiments

This section qualitatively outlines the thought process that guided the development of the final model, shedding light on the key considerations, challenges and adjustments that shaped its progression. Transfer learning was the obvious starting point: the backbone of the model was initially MobileNet, a relatively small pre-trained architecture that combined the benefits of a reliable feature extractor and a hasty training process. This was crucial in order to allow for fast iterations with different techniques such as fine-tuning, k-fold cross-validation, augmentation and alternative prediction heads. Since the augmentation is an essential part of all the developed models, the following section is entirely dedicated to it. As far as prediction heads go, the first attempts were based on a classical approach with flattening and one to three dense layers, but this type of classifier proved to be highly inconvenient: the addition of many trainable parameters wasn't compensated by greater accuracy. A GAP (Global Average Pooling) layer appeared to be a reasonable compromise: without adding complexity it could boost the translational invariance property of the network. The combination of MobileNet, GAP and a light augmentation pipeline reached 47% accuracy on the test set. At this stage it wasn't clear that augmentation was crucial in order to obtain higher performance: from a human point of view the main features that separate the classes were size and color, so the augmentation was only composed of transformations that wouldn't affect these properties. Switching to a more complex pre-trained model, such as ConvNeXtTiny, brought the accuracy to 60% and it was difficult to figure out, due to a biased view of the role of augmentation, what was causing this low precision. In order to better understand how the model worked, a cam technique was employed, the results of which are shown in the image below.



Findings like these strengthened the belief that the model was not extracting the most intuitive features. From this point on the development of new models took an alternative and arduous route that basically led to nothing, except for a deeper understanding of how neural networks work. The search for a strategy to make the network focus on the cell brought in masking techniques. As described in this paper [1], masking is an algorithmic process that filters out from the input everything but the points of interest, in this case, the cell. It is performed in 3 steps. Firstly, a copy of the image is *binarized*: only pixels within certain threshold values are set to 1. Then erosion and dilation are applied in order to smooth out the boundaries. Since existing layers for such morphological operations were incompatible with GPUs, they were replaced by pooling and up-sampling. Finally, a tensor product is performed between the obtained mask and the original image. To test this new technique, two models were trained on a completely masked dataset: the transfer learning model described before and one built from scratch taking inspiration from the paper [1]. Those network achieved respectively 55% and 14% test-accuracy. At this point the first doubts on this strategy arose but, before completely abandoning the approach, masking was implemented as a proper custom layer of the network to test the assumption that training on a masked dataset was causing misclassification errors due to the difference between training set and test set images. As shown in the heatmap below the masking layer worked pretty well. Unfortunately this attempt too resulted in disheartening performance on the test set.



There were problems with masking (the heavy dependence on uniform brightness for instance), as well as with the approach itself: it became apparent that the network wasn't focusing on the wrong features, it was the human perspective and inherent biases on which features the model should focus on to be flawed. At a certain point one has to let his deep networks become adults, move out of their parents' house and extract **their own** features, this is what being a deep network is all about. At this point a model built with transfer learning and a very strong augmentation pipeline seemed the only possible way forward. The final result is described in all it's details in this section, and it reached an accuracy of 96% on the test set.

3 Augmentation

Of the 13759 96x96 RGB images, the last 1800 were of no use: being simply the superimposition of two fixed cells and unrelated perturbations, they were removed. A balanced dataset was produced through random rotations and flips of images belonging to underrepresented classes, to avoid the potential issues of an uneven distribution. An inspection of the dataset highlighted that the color and the dimensions of the cells could be discriminative features, thus, the working assumption was that the problem was neither color nor scale invariant. According to this assumption, only augmentation techniques that don't significantly alter the scale and the color channels of the images were initially applied. As suggested in this paper [2], to simulate possible conditions under which blood smear images may be taken, **Gaussian blurring**, **Gaussian noise** (up to 0.05 times the maximum pixel intensity), **random contrast** and **random brightness** (both with factors sampled from a uniform distribution in the interval [0.8, 1.2]), were implemented, followed by geometric transformations, such as **random horizontal and vertical flips**, **random rotations** (of an angle in [-0.2 π, 0.2 π]) and **random translations** (with height and width factors equal to 0.2). Further attempts included **CutMix** augmentation, which led to a significant drop of the performance over the test set, and the introduction of a **random zoom** up to 10% of the image size, in order to weaken the assumption of scale invariance. Techniques were applied during training or to create a new augmented dataset according to the availability of a corresponding *keras* layer. After the masking flop, the approach to augmentation needed to be revolutionized: the key to obtaining a satisfactory score in the competition was to realize how different the test set was from the training data provided. While the significance of such deviation was clear from the discrepancy between local validation accuracy and submission scores, an idea on what caused it exactly arose from an experiment with a heavily augmented test set. In particular, applying sequentially two *keras cv* layers, namely **AugMix** and **RandAugment** (with default parameters), the poor results achieved until then were replicated. In subsequent models those two operations replaced the former (lighter) pipeline, and were applied to the dataset at each epoch in a different way. That was achieved (in parallel, as to avoid unreasonable processing time) via *TensorFlow* dataset's operations: map, batch and prefetch. The validation set used for early stopping was also augmented (just once) in the same way.

4 Final models and Results

The final model is built upon *ConvNeXtXLarge* used as a pretrained feature extractor with a new prediction head based on the *inception module* (although originally thought of as part of a feature extractor, it yields good results thanks to it's ability to account for features at different scales and to the invariance to shifts provided by the *GAP* layer). Going into details, the output of the feature extractor is fed into a max-pooling layer from which 4 paths branch out: 3 of them with two convolutional layers each and the last one with max-pooling. The feature maps of the four branches are joined back together and concatenated depth-wise, they're fed to a GAP and finally to a dense layer, with some dropout added for regularization. Here the 1x1 filters are used as bottlenecks not to increase computational complexity too much. The training uses severe sample-wise augmentations and a custom loss function made by a variant of the usual categorical cross-entropy called focal loss with label smoothing. Label smoothing is the technique of assigning a non-zero probability to wrong classes in order not to let the network get too sure about it's predictions and thus limit overfitting. Focal loss is employed to make the network place more focus on challenging examples.

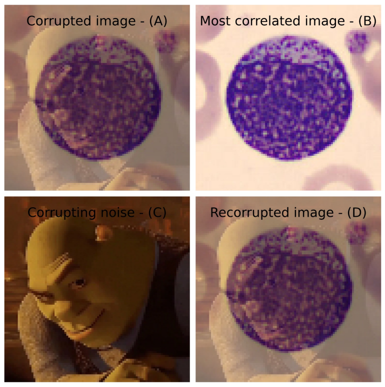
$$y_s = y_t \cdot (1 - S - \frac{S}{8-1}) + \frac{S}{8-1} \qquad (1)$$

$$\mathcal{L} = \sum_{dimension} \alpha \cdot (1 - y_p)^{\gamma} \cdot (-y_s) \cdot log(y_p) \qquad (2)$$

Where y_t are the true labels, y_p are the labels predicted by the model, $S = 0.05$, $\alpha = 1$ and $\gamma = 0.5$. The training process is divided into 2 macro-phases. Initially the feature extractor is frozen and only the prediction head is trained, to do this we first find the ideal number of epochs to train for using K-fold cross validation and early stopping, then the model is trained without a validation set up until the number of epochs computed at the previous step. The second macro-phase is the fine-tuning stage: this is made up of several micro-phases in which we gradually unfreeze more and more convolutional and dense layers of the pretrained feature extractor. Each phase has it's own early stopping criterion and the learning rate decreases exponentially among phases. Finally a light test-time augmentation is performed at inference time, this only includes flips and rotations. This model achieves: 96% test accuracy on *Codabench* and a very good resistance to over-fitting thanks to a training process that has been engineered not to let the model see the same samples twice.

5 Bonus section

Observing that the last 1800 images of the dataset were out-of-distribution samples created by superimposing the standard cells with one of two *distortions*, we sought for a way to make sense of them. The images themselves had to be discarded because their label was also corrupted (there were identical images with different labels and identical labels assigned to images of different classes) so we had to find a workaround. We wrote a python script to analyze the whole dataset and find the mostly correlated image with a certain corrupted input image $\mathcal{A}$ among those that were not identical to $\mathcal{A}$ itself, this yielded $\mathcal{B}$. Having both $\mathcal{A}$ and $\mathcal{B}$ we were able to extract a precise representation of the corrupting noise $\mathcal{C} = \frac{\mathcal{A} - \alpha \cdot \mathcal{B}}{1 - \alpha}$ where $\alpha = 0.5$ was the best choice. We applied this process 2 times to get the two images representing the distortions present in the dataset, then we could superimpose them to the training as a form of augmentation. To us, this was both a way to test if this kind of augmentations had been employed in constructing the test set and a standalone augmentation process that was worth testing, indeed our assumption was that this kind of distortion preserved labels, but this technique did not yeld the desired results and was ultimately discarded.



Contibutions, Bonanomi Emanuele: data inspection, balanced dataset, saliency map and augmentation; Dante Riccardo: custom layer implementation, masking and augmentation; Derme Francesco: Fine tuning, loss function, final model implementation and bonus section; Fumagalli Pietro: K-fold, test time augmentation, masking and report structure.

References

[1] I. Oahidul, M. Assaduzzaman, and Z. Hasan. An explainable ai-based blood cell classification using optimized convolutional neural network. *Journal of Pathology Informatics*, 2024.

[2] O. Pako Mmileng, A. Whata, M. Olusanya, and P. Mhlongo. Application of convnext with transfer learning and data augmentation for malaria parasite detection in resource-limited settings using microscopic images. *MedRxiv*, 2024.