AN2DL - First Homework Report
The Bergers

Francesco Palma, Francesco Pellegrini, Luigi Raggi

palmfra, frapelle, luigiraggi

243800, 247907, 217882

November 22, 2024

1 Introduction

The first homework of the 2024-2025 Artificial Neural Networks and Deep Learning course consisted in a **blood cells image multiclass classification task**. Given a dataset of blood cells images, the goal was to analyze the data and train a *Convolutional Neural Network* based model able to achieve the best possible accuracy on a secret test set. The accuracy computation was done by Codabench, an external deep learning challenges platform that hosted the competition.

2 Problem Analysis

The first step taken to tackle the challenge was to load the dataset, stored in a NumPy archive file, and plot some of the contained images 1.

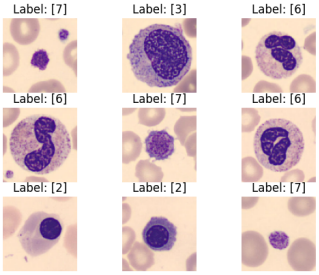

Figure 1: Dataset blood cells images with labels

Knowing how the images look like, the following step was to plot some significative statistics such as the average pixel brightness per image.

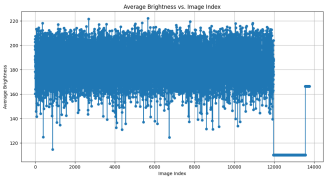

Figure 2: Average pixel brightness per image

From this graph 2 it was noted that some of the last images had the exact same value, leading to the deduction that those images were all the same, but repeated. After plotting those images, it was evident that they were outliers so in conclusione they were dropped from the dataset. Then, the distribution of classes across the dataset was plotted to check for an eventual class imbalance.

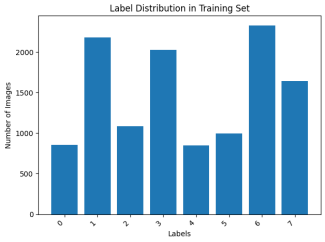

Figure 3: Classes distribution across dataset

The graph 3 confirmed that the dataset was indeed imbalanced.

3 Method

The model training was split in two parts: *feature extraction* and *fine tuning*. During the feature extraction phase the model layers have been freezed to train only the head while on the fine tuning phase some layers have been gradually unfreezed. **Backbone Model**: the model that was selected was *EfficientNetV2L*. The reasons are suggested from the paper [14] which includes optimized training speed and parameter efficiency that rivals much bigger models. **Model Head**: the model head includes two dense layers that follow two dense layers. This was done to ensure a balance between counter overfitting and enhancing learning. **Data Augmentation**: to ensure the maximum effect of data augmentation it was decided to adopt *online random augmentation*. For every epoch of the training, every image is augmented in a different way. The selected augmentation strategy was a method from *keras-cv* library called *RandAugment*. As described in the paper [3], this layer selects randomly three different transformations chosen randomly to apply sequentially in a single image.

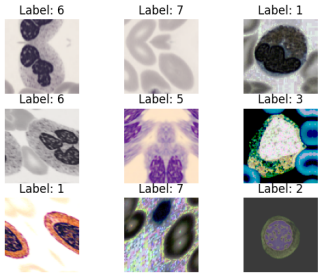

Figure 4: Examples of data augmentation

Optimizer: to apply an efficient regularization starting from the optimizer *AdamW* was selected. According to the paper [11], it improves Adam's generalization performance, allowing it to compete with SGD with momentum on image classification datasets. **Loss**: to tackle the problem of class imbalance the *FocalLoss* loss function was adopted. According to the paper [8], this loss function works by reshaping the standard cross entropy loss such that it down-weights the loss assigned to well-classified examples. This can be seen also in the following equation 1:

$$FL(p_t) = -(1-p_t)^{\gamma}\log(p_t) \tag{1}$$

where p_t is the model estimated probability and γ is a tunable parameter. **Learning Rate**: according to the AdamW paper [11], the optimal learning rate is 3-10 times smaller then the one of Adam, so we set a fixed learning rate of 1e-4 for the feature extraction phase, with the addition of *ReduceLROnPlateau*, a callback that reduces the learning rate when the validation accuracy stagnates. In the fine tuning phase it was used a cosine decay scheduler with restarts and initial learning rate 1e-5. This because the paper [10] suggests it is often useful to lower the learning rate as the training progresses, the smaller initial learning rate is to avoid the unfreezed layer to "forget" too much of the feature extraction phase.

4 Experiments

In the following are highlighted the experimental steps. The starting model takes inspiration from the transfer learning lab notebook of the fourth exercise lecture, the main difference resided in the VGG19 model [13], chosen for its good performance applied to medical classification challenges [6]. A first boost in accuracy was obtained cleaning the dataset from some evident outlayers. Standardization and normalization was let to the default procedure used for VGG19. Images were then properly resized and a sequential augmentation layer, that performed random flip and translation, was added at the beginning of the network. The imbalance in the classes distribution was handled using a weighted loss function. The focus then was shifted on the optimizer, were compared: Adam [7] and Lion [1]. No significant differences emerged, so Adam was selected due to the higher reliability. Fine tuning was done by defreezing the last ten layers of the convolutional part. The network reached an accuracy of 0.55 on the development testing dataset. So it was tested the model ConvNeXtLarge [9] that, altough heavier, has shown promising performance, like in brain tumor classification [12]. In this case, being the model very deep, a key parameter to tweak was the number of layers to defreeze in the fine tuning. To improve regularization the initial dataset was augmented using a mixup techinque, doubling its size. No significant

improvments were noticed. As a final adjustment the augmentation sequence was extended, including non-geometric manipulation techniques. In terms of accuracy this model largely exceeded VGG19, scoring 0.77. Having experienced that the accuracy was strongly related to the choice of the backbone, EfficientNetV2L [15] was tested. With similar parameters, this model scored 0.75 so it was picked due to its reduced size. Afterwards, changes were made to model head, adding a dense and dropout layer, augmentation, using RandAugment, and optimizer and loss, using AdamW and FocalLoss. In addition, tensorflow datasets were used due to compatibility with keras-cv randaugment. This boosted the accuracy to 0.93 and a change in the number of unfreezed layer let to 0.95.

5 Results

This table 1 shows the outcome of different models in terms of accuracy on the Codabench test set.

Table 1: Backbone model used

Model	Best Test accuracy
VGG19	0.55
ConvNeXtLarge	0.77
EfficientNetV2L	**0.95**

On the local test set, EfficientNetV2l produces 0.9833 accuracy and the following statistics.

Classification Report:

	precision	recall	f1-score	support
0	0.98	1.00	0.99	42
1	1.00	1.00	1.00	109
2	1.00	0.98	0.99	54
3	0.95	0.99	0.97	102
4	0.96	1.00	0.98	43
5	0.96	0.96	0.96	50
6	1.00	0.95	0.97	116
7	1.00	1.00	1.00	82
accuracy			0.98	598
macro avg	0.98	0.98	0.98	598
weighted avg	0.98	0.98	0.98	598

Figure 5: Results

6 Discussion

The most incisive fetaures in the network were in the order: the backbone model, the data augmentation method and the number of layers to unfreeze. VGG19, composed by simple convolutional layer, was outperformed by more sophisiticated network. ConvNeXtLarge and EfficientNetV2L use advanced features, to name a few: depthwise convolutional, batch/layer normalization and squeeze and excitation modules. As reported in literature [5], EfficientNetV2L arose as the more suitable for medical tasks. Having less parameters, the tuning may have been easier, considering the small and imbalanced dataset used in the competition. In addition, it incorporates SE layers that adaptively emphasize important features, which is a crucial aspect as highlighted in [4]. Besides the selection of backbone, also the number of layers to unfreeze played an important role during the fine tuning phase. The unfreezing was performed always starting from the head of the model and it was observed that there was a specific optimal depth for unfreezing; deviating from this point — either by unfreezing fewer or more layers — led to a decline in performance.
Moreover, it was observed that enhancing both the complexity and diversity of augmentation techniques consistently improved performance, regardless of the backbone architecture used. In [16] enhancements in regularization are gained using a wide variety of augmentation techniques, especially for blood cells' classification. The most reliable technique has proven to be RandAugment [2].

7 Conclusions

The results achieved by the group both in terms of the leaderboard and the resulting test accuracy were excellent. This was done thanks to a great cooperation by the team members as well as a great curiosity that pushed the members to go beyond the sufficient assignment. However, there is still room for improvement as the confusion matrix 6 shows that the majority of misclassifications on the test set are due to images classified as 3 while they should have been classified as 6. Addressing, this paves the way for further improvement.

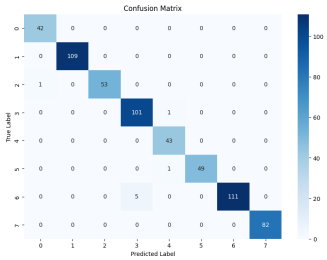

Figure 6: Confusion Matrix

References

[1] X. Chen, C. Liang, D. Huang, E. Real, K. Wang, Y. Liu, H. Pham, X. Dong, T. Luong, C.-J. Hsieh, Y. Lu, and Q. V. Le. Symbolic discovery of optimization algorithms, 2023.

[2] E. D. Cubuk, B. Zoph, J. Shlens, and Q. V. Le. Randaugment: Practical automated data augmentation with a reduced search space, 2019.

[3] E. D. Cubuk, B. Zoph, J. Shlens, and Q. V. Le. Randaugment: Practical data augmentation with no separate search. *CoRR*, abs/1909.13719, 2019.

[4] A. Iantsen, D. Visvikis, and M. Hatt. Squeeze-and-excitation normalization for automated delineation of head and neck primary tumors in combined pet and ct images. In *Head and Neck Tumor Segmentation: First Challenge, HECKTOR 2020, Held in Conjunction with MICCAI 2020, Lima, Peru, October 4, 2020, Proceedings 1*, pages 37–43. Springer, 2021.

[5] P. Jeevan and A. Sethi. Which backbone to use: A resource-efficient domain specific comparison for computer vision. *arXiv preprint arXiv:2406.05612*, 2024.

[6] I. A. Kandhro, S. Manickam, K. Fatima, M. Uddin, U. Malik, A. Naz, and A. Dandoush. Performance evaluation of e-vgg19 model: Enhancing real-time skin cancer detection and classification. *Heliyon*, 10(10):e31488, 2024.

[7] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization, 2017.

[8] T. Lin, P. Goyal, R. B. Girshick, K. He, and P. Dollár. Focal loss for dense object detection. *CoRR*, abs/1708.02002, 2017.

[9] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie. A convnet for the 2020s, 2022.

[10] I. Loshchilov and F. Hutter. SGDR: stochastic gradient descent with restarts. *CoRR*, abs/1608.03983, 2016.

[11] I. Loshchilov and F. Hutter. Fixing weight decay regularization in adam. *CoRR*, abs/1711.05101, 2017.

[12] Y. Mehmood and U. I. Bajwa. Brain tumor grade classification using the convnext architecture. *Digital health*, 10:20552076241284920, 2024.

[13] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition, 2015.

[14] M. Tan and Q. Le. Efficientnetv2: Smaller models and faster training. In M. Meila and T. Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 10096–10106. PMLR, 18–24 Jul 2021.

[15] M. Tan and Q. V. Le. Efficientnetv2: Smaller models and faster training, 2021.

[16] Y. Zhu, X. Cai, X. Wang, and Y. Yao. Bayesian random semantic data augmentation for medical image classification. *arXiv preprint arXiv:2403.06138*, 2024.