Urchade Zaratiana, k11932165

Microscopy Imaging Challenge: Cell Line Identification

This challenge aims at developing a Deep Learning model which can classify nine regularly misidentified cell lines based on microscopy images. For that, we are given a dataset in which each sample contains at least one cell and consists of 3 separate images showing different parts (nucleus, microtubules, endoplasmic reticulum) of the same cell.

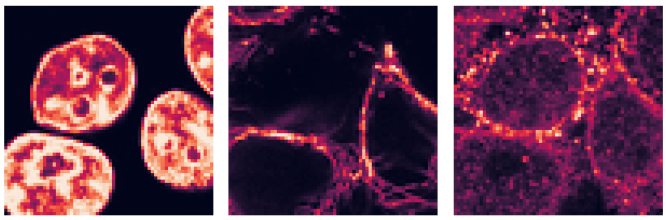

Figure 1: One sample from the training set.

1. Data preprocessing

To make one training sample, we stack the three separate images (nucleus, microtubules and endoplasmic reticulum) and we treat them as an RGB channel, resulting in a tensor of dimension $3 * 64 * 64$.

Moreover, we compute the overall mean and standard deviation for each of the three channels, and we use the results to normalize our data. Finally, we split the data into a training and validation set. We used 1000 samples for validation and the remaining samples for training.

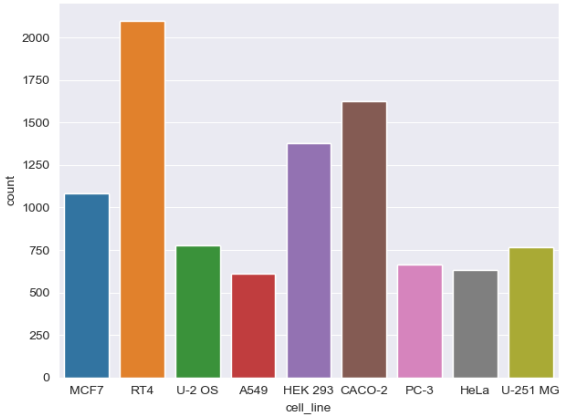

Figure 2: Label count for the training set.

2. Model architecture

During the experiments, we tried different architectures like ResNet (He et al., 2015) or VggNet (Zeiler and Fergus, 2013) but we got our best result with a 161-layered DenseNet (Huang et al., 2017) pretrained on ImageNet dataset (we also tried non-pretrained architectures as baseline). A DenseNet is composed of several dense blocks which connect each layer to every other layer in a feed-forward fashion. We adapted the pretrained model by removing the last layer, and we replaced it by a dropout layer (Srivastava et al., 2014) with $p = 0.5$ and a fully connected layer.

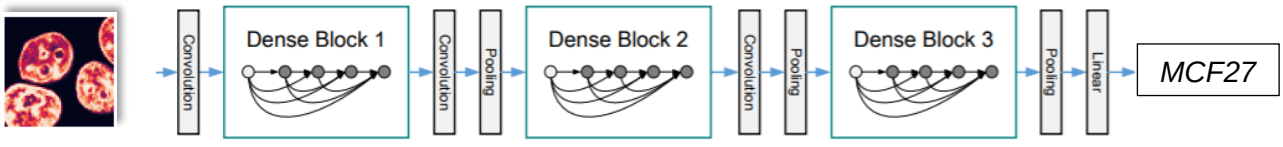

Figure 3: Architecture of our model. Picture adapted from (Huang et al., 2017).

3. Model training

We trained our model by optimizing its cross-entropy loss using an Adam optimizer (Kingma and Ba, 2017) with a learning rate of 0.001. We trained for only 7 epochs since the model is very big and required. At the start, we employed a batch size of 128 but we later noticed that lower batch size (64 samples) yield to a better result. The training lasted 46 minutes with a single GPU (Nvidia GeForce MX150).

Furthermore, we saved the model that got the highest balanced accuracy on the validation set during the training (6th epoch) and we loaded it for testing and submission.

All the implementations were done using PyTorch and torchvision library.

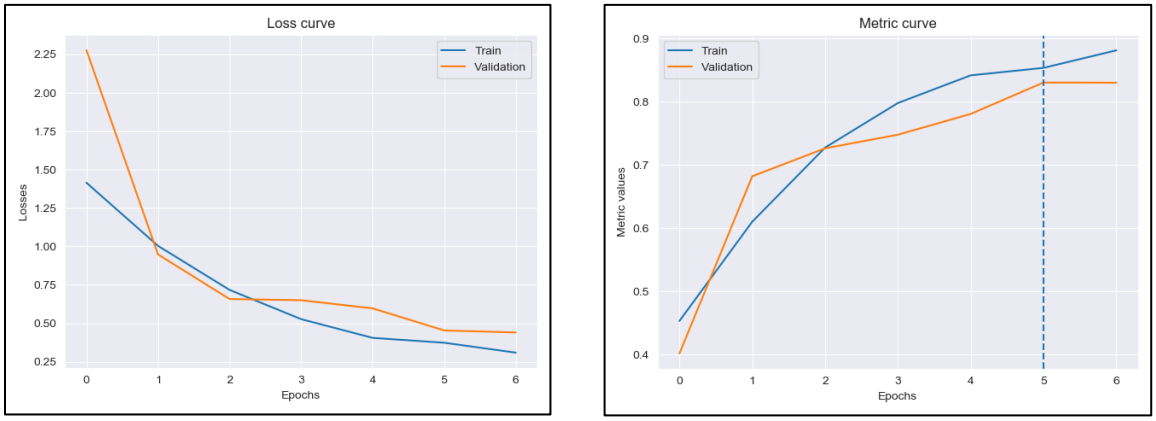

Figure 4 : Learning curves. (At the left the loss curves and at the right the balanced accuracy curves)

4. Results and conclusion

We got a balanced accuracy (BAC) of 86% on the training set, 83% on the validation set and approximately 75% on the test set. We can notice that the difference between the validation and test BAC is quite high. We think that it may be caused by the fact that we have normalized the training data before the train/validation splitting. Consequently, the training set contains some information about the validation set.

However, we can see that the model performs poorly on labels that have few samples like **HeLa** and **U-2 OS** which is normal (imbalanced dataset) even though it could be alleviated by using a weighted loss or techniques like over-sampling or under-sampling.

Finally, the learning curves show that we could get better results by training longer since the model is not overfitting yet.

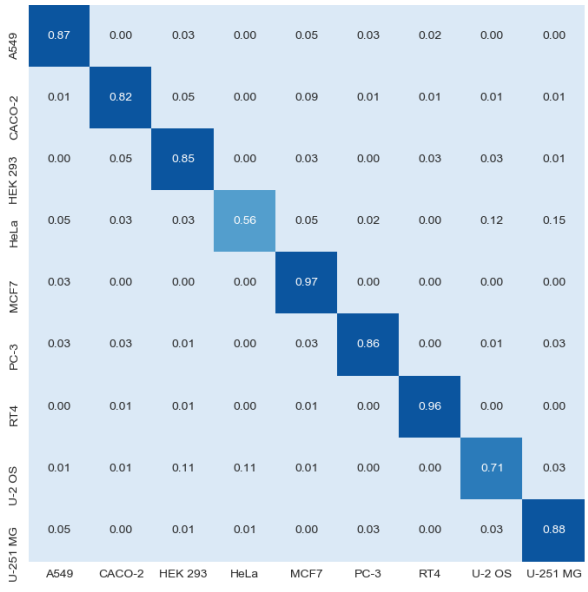

Figure 5: Confusion matrix on the validation set

5. Bibliography

He, K., Zhang, X., Ren, S., Sun, J., 2015. Deep Residual Learning for Image Recognition. CoRR abs/1512.03385.

Huang, G., Liu, Z., Weinberger, K.Q., 2017. Densely Connected Convolutional Networks. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2261–2269.

Kingma, D.P., Ba, J., 2017. Adam: A Method for Stochastic Optimization. arXiv:1412.6980 [cs].

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R., 2014. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. Journal of Machine Learning Research 15, 1929–1958.

Zeiler, M.D., Fergus, R., 2013. Visualizing and Understanding Convolutional Networks. CoRR abs/1311.2901.