Deep Learning for Medical Imaging
Clasification of lymphocytosis from blood cells

Julie Dessaint
Hugo Dalla-torre
Kaggle team name: Dessaint Julie

21/03/2021

Abstract

This paper aims at applying state of the art deep learning techniques to a multi-instance learning problem. The goal is to combine the information from both a bag of blood cells pictures and the clinical attributes of the patient. Our investigation show that a convolutional neural network (CNN) can reach a 84.93% **balanced accuracy on a test dataset on the diagnosis of lymphocytosis only by extracting information from the patient's bag of pictures, which demonstrate the relevance of including an image processing step into the diagnosis procedure. The proposed method is an end-to-end trainable pipeline which combines the features extracted from the patient's blood cell pictures by the CNN with his clinical attributes (age and lymphocyte concentration) that achieves a balanced accuracy of** 84.93%**.**

Introduction

Lymphocytosis is diagnosed whenever the concentration in lymphocytes exceeds $c_{lymph} = 4 \times 10^9$ per liter of blood. It is common in the population and can be of 2 natures: *reactive* when in response to an infection or acute stress and *tumoral* when the manifestation of a lymphoproliferation disorder which is a type of lymphocyte cancer.

In order to perform the diagnosis, a clinician will collect multiple samples from the patient's bloods, examine it under the microscope, assess the texture, form and size of the lymphocytes and combine this information with the clinical attributes of the patient, mainly age and lymphocyte concentration. Although this is a very quick and cheap process since only a blood sample is required, it is obviously very subjective and present poor reproducibility among the clinicians whose opinion can differ on a same case. If the patient needs additionnal analysis, tests are performed to validate the diagnosis, among which the standard one is flow cytometry, an expensive and tedious procedure. A fast, automatic, accurate and reproducible approach to the diagnosis of lymphocytosis would therefore bring a better approach to the determining of which patient should be referred to a flow cytometry.

Taking advantage of the momentum around deep learning solutions for medical imaging, this paper proposes a computer vision based approach to the diagnosis of lymphocytosis. First, a vanilla CNN is proposed to extract relevant features from the patient's blood cell pictures. Secondly, we refine the structure of the CNN by downsizing the ResNet (He et al. 2016) architecture. Finally, the features extracted by this CNN are combined with the patient's clinical attributes to extract the most information from the patient.

The paper is organized as follows. Section 1 describes the architecture chosen for our end-to-end trainable final solution as well as the multi-instance learning framework. Section 2 presents the preprocessing steps taken, the performance brought by the different bricks of the model as well as a discussion on the solutions tested but whose performance disappointed.

1 Architecture

1.1 Dataset and approaches

The dataset is composed of N subjects, each of them being represented by a bag of data $\left\{ (X_i j)_1^{N_i}, a_i, c_i, y_i \right\}_i$ where X_{ij}, a_i, c_i and y_i are respectively the j^{th} blood cell picture, the age in years, the lymphocyte concentration in number per litre and the label of the i^{th} patient.

The aggregation of multiple instances per label is called multi-instance learning and is a form weak supervision. To tackle this, we explored the performance of two approaches:

- **Features aggregation** : Feature vectors are extracted from each of the patient's bag pictures by a CNN. These features are then concatenated and aggregated by taking the mean along concatenation axis to give an ultimate feature vector that is representative of the whole bag. This vector will then be used to predict the diagnosis of the patient.

- **Prediction aggregation** : Predictions are done for each picture of the patient's bag and then aggregated by the mean operator to give the patient's diagnosis.

1.2 Proposed architecture

In this section we will present our final architecture. A CNN is used as a visual features extractor on each of

the patient's images. For each image, the feature vector obtained is combined with the age and lymphocyte concentration before being passed through a multi-layer perceptron (MLP) to give a prediction for this image. All the predictions are averaged to give the final diagnosis.

1.2.1 Feature Extractor

The CNN used is a downsizing of the resnet architecture. Since Resnet is too big of a network to be trained on such a small dataset ($\sim$ 13400 pictures in total), its fundamental architecture leveraging the benefits of including residuals in the forward pass is kept but the number of layers is dramatically decreased in order to better match the dataset. (Sahasrabudhe et al. 2020) showed that it is an efficient features extractor. Table 1 presents our architecture where each convolution is followed by batch normalisation and rectified linear unit (ReLU) and each fully connected layer is followed by ReLU and dropout except for the last layer which is followed by sigmoid to obtain an output between 0 and 1.

Layer name	**Output size**	**Filter size**
Conv1	112×112	$\begin{bmatrix} 7 \times 7, K \end{bmatrix}$
Conv2	56×56	$\begin{bmatrix} 3 \times 3, K \\ 3 \times 3, K \end{bmatrix}$
Conv3	28×28	$\begin{bmatrix} 3 \times 3, 2K \\ 3 \times 3, 2K \end{bmatrix}$
Conv4	14×14	$\begin{bmatrix} 3 \times 3, 4K \\ 3 \times 3, 4K \end{bmatrix}$
Conv5	7×7	$\begin{bmatrix} 3 \times 3, 8K \\ 3 \times 3, 8K \end{bmatrix}$
Flatten and Concatenate	$7 \times 7 \times 8K + 2$	
Fc1	1000	•
Fc2	500	•
Fc3	100	•
Fc4	1	•

Table 1: Network Architecture. The convolutional layers correspond to the ResNet like architecture while the fully connected layers correspond to the MLP. This architecture gives relevent performances for $K \in \{32, 64\}$.

1.2.2 Concatenating and MLP

A MLP is implemented in order to take into account both the features extracted by the CNN and the clinical attributes of the patient. In Table 1, the MLP correspond to the four fully connected layer. This is done via the concatenation of the features vector with the 2 quantities age and concentration, that is why the output size of the layer "Flatten and Concatenate" in Table 1 is $7*7*8K+2$. Formally, by noting $x_{ij} = CNN(X_{ij})$ the features vector of size $[1{,}7 \times 7 \times 8K]$ extracted by the CNN from picture X_{ij}, the diagnosis based on the single picture can be written as such:

$$y_{ij} = MLP\left(\begin{pmatrix} x_{ij} \\ a_i \\ c_j \end{pmatrix}\right)$$

.

1.2.3 End to end architecture

Predicting the diagnosis of a patient is done by averaging the predictions done on each of the pictures. Denoting y_i the diagnosis of patient i, the following relationship follows

$$y_i = \frac{1}{N_i} \sum_{j}^{N_i} y_{ij}$$

This may contradicts the heuristics of the clinician whose diagnosis on a single picture is irrelevant because the information relies on the whole bag of pictures rather than one particular lymphocyte, but our experience shows that prediction aggregation outperforms by far features aggregation. We tested our different architectures with features aggregation instead of predictions aggregation but it did not perform well. Thus, in the rest of this paper we will only present architectures that aggregate predictions.

1.3 Metrics

The algorithm selection is based mainly on the **balanced accuracy**, **recall** and **AUC**. This is due to the importance of having the fewest false negatives possible. We used the balanced accuracy instead of the accuracy because the two classes are unbalanced. We also draw the ROC curve for the different methods. Another important point is that, in order not to be biased by the test set, the algorithms tested are chosen only on their performance on the validation dataset. In order to design the most robust algorithm, a train and validation datasets with similar distributions are created at each training.

1.4 Relevance of the algorithm

Deep learning has brought an upscale of algorithm performance in computer vision and is greatly developping in the healthcare business. However, deep learning is not a requirement as in some cases a "regular" machine learning algorithm displays a better performance for far less trouble training it. The first thing the authors looked at was the baseline performance the deep learning solution was to beat.

We defined the baseline performance as the best performance that can be obtained with the usual machine learning algorithms on the clinical attributes. Taking only as

inputs the age and the lymphocyte concentration of the patients, **SVM**, **Random Forest** and **MLP** presented a validation balanced accuracy of 84.2%, 87.4% and 64.1%. Therefore, the deep learning solutions explored were evaluated keeping in mind that only a balanced validation accuracy superior to 87.4% would mean relevance for the architecture.

2 Model Tuning

The proposed architecture follows a serie of tryout. However, all these architectures took as input a dataset that went through a data augmentation process that is presented here.

2.1 Preprocessing

Data augmentation is performed on the dataset as it is standard procedure when the training set is small. This prevents overfitting and gives additionnal robustness to the algorithm. Random horizontal and vertical flips are introduced since the diagnosis of a lymphocyte is independant from the orientation of the blood cell picture.

(A cropping of the center of the picture is also introduced since the examination of the dataset shows that all pictures are centered on a lymphocyte and it is the intuition of the authors that the increase in signal to noise ratio will make up for the few pictures where the lymphocyte is not perfectly cropped.)

2.2 Loss function

Since the output of the network is binary, a natural loss function is the Binary Cross Entropy (BCE). When outputting the prediction $\hat{y}$ for target y, the network induces a loss

$$\mathcal{L}(\hat{y}, y) = -w(ylog(\hat{y}) + (1-y)log(1-\hat{y}))$$

with w a rescaling factor depending on the batch processed. For a batch size superior to 1, the BCE loss of the batch is simply the mean of the individual losses.

2.3 Implementation details

The code was written in Python with the PyTorch library, and executed with Google Colab. The models were trained using the Adam optimiser, since it proved to have better performance than Stochastic Gradient Descent, starting with a learning rate of $5e^{-6}$. The batch size is set to 1, a bigger batch size would have rendered the training smoother than a batch size of 1 but a a superior value doesn't fit in the RAM. To execute the code, you can run the notebook 'DLMI_Project.ipynb', in the notebook you will be able to choose from the different architectures we discuss in this paper. You need a GPU to run the notebook. For some of the models (ResNet with clinical attributes) we started with a learning rate of $5e^{-4}$ and then decay it to $5e^{-5}$ after the first 10 epochs and then decay it again to $5e^{-6}$ after the first 50 epochs. This helped the models converge more quickly at the beginning of training but it did not change the performance of the model overall.

Training is done for 120 epochs , which is enough for the model to start overfitting the dataset. We choose the best model found during training according to its performance (balanced accuracy, recall, AUC) on the validation dataset. Training the model takes about one to two hours depending on the model.

2.4 Deep learning methods investigated

In this section, we will present the different architecture we investigated and discuss their performance. We will present them from the simplest to the most complicated architecture. Each network consists a small architectural update, that was built up to tackle an aspect of the problem. This approach is valuable because it also presents us with an ablation study, since the usefulness of each components is directly observed. The results for the different models are presented in Table 2, ROC curves for the different methods are shown in Figure 2 and training loss and validation accuracies are shown in Figure 1.

2.4.1 Vanilla CNN

The first architecture used as a feature extractor is a small CNN composed of $\left[3 \times 3, M_i\right]$ with $M_i \in \{8, 16, 32, 64\}$ and a final fully connected layer. The training balanced accuracy plateaued at the same value than the validation one, which indicates that upscaling is necessary. The result we obtained on the Kaggle public leaderboard with this model is **63**% accuracy on the public leaderboard.

2.4.2 Deep CNN

In the CNN spectrum, opposed to the vanilla model just presented, lays the VGG, Inception, ResNet and the pretrained networks jungle accessible with the Pytorch library. These networks are fare too big for such a small training set and even finetuning a fully connected layer on top of these networks showed no significant results. Since Sahasrabudhe et al. 2020 found that a downsized ResNet makes a good features extractor for the task, we decided to take this as a model.

First, we implemented the CNN presented in Table 1 without the residual layers, in order to quantify the performance improvement that these would bring to the model,

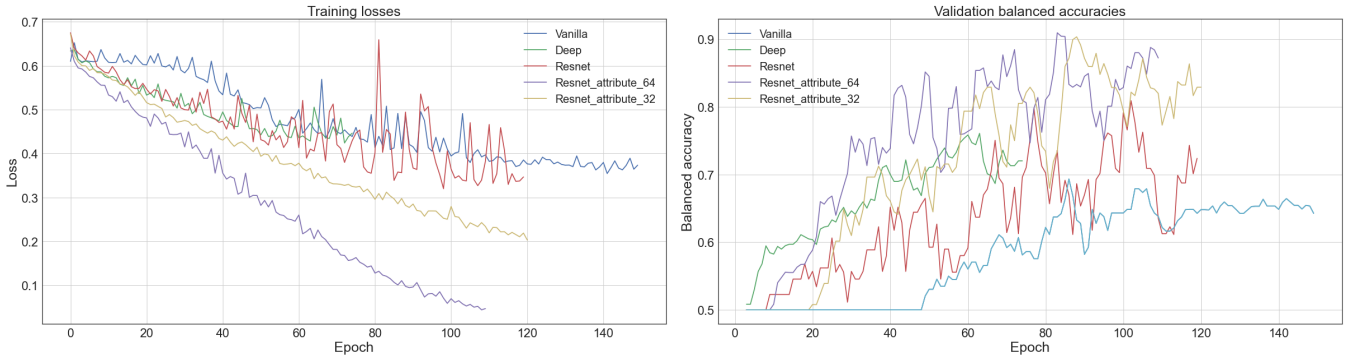

Figure 1: Training loss and balanced validation accuracies the different explored models

Methods	balanced accuracy	Recall	AUC	Accuracy on Kaggle
Deep CNN	0.80	0.84	0.84	0.78
ResNet, K= 32	0.78	0.73	0.84	0.71
ResNet, K= 64	0.86	0.88	0.88	0.84
ResNet with attr. K= 32	0.94	0.96	0.97	0.83
ResNet with attr. K=64	0.90	0.88	0.93	0.78

Table 2: Metrics for the different models, All the metrics were calculated on a validation dataset made of 38 patients, the Accuracy on Kaggle is the accuracy obtained on the public leaderboard.

with only 2 fully connected layers. This model attained **80.11**% balanced accuracy on the validation dataset and it yielded **78.18**% on the kaggle public leaderboard.

2.4.3 Downsized ResNet

Adding the residual layers mechanisms yields the architecture presented in Table 1, without the concatenation of the clinical attributes. This model still only deals with the blood cells pictures. We tested this model with K=32 and K=64. The balanced accuracy, for the model with K=64, is **86**%, which constitute an increase compared to the regular "plain" network. Surprisingly, this model doesn't perform the best on the validation dataset (see Table2 yet this model yielded our best score on the kaggle public leaderboard (**84.93**%). The same model with K=32, did not perform as well. This may be due to the fact that the training phase is not really stable, we can observe this instability in Figure 1. We can see that both the losses and balanced accuracies oscillates a lot and that results in different performances from one training session to an other. The ideal, in this case, would be to present average scores based on multiple training of our different models but we did not possess the computational power to do so. As a result, the downsized ResNet with K set 64 yielded performance on the kaggle public leaderboard from **75.58**% to **84.93**% which is the best result we obtained.

2.4.4 Mixed models

In the previous section we have explored different CNN architecture for feature extraction and classification. To this point, we have only used the lymphocytes images to perform the classification task. Now, we will present methods that combine the features extracted by the downsized ResNet to the clinical attributes (age and concentration). The full architecture is presented in Table 1. This model is composed of the downsized ResNet presented above that we use as feature extractor for the images, then we build a multi layer perceptron (MLP) for classification. The input of the first fully connected layer are the features extracted by the ResNet concatenated to the age and concentration of the patient. The activation function used for the fully connected layer is ReLU except for the last one where we used the sigmoid function to obtain an output between 0 and 1. We used dropout between all the fully connected layer, without dropout this architecture is subject to overfitting.

This architecture performed very well on the validation dataset, we can see in Table 2 that the balanced accuracy for this method is **0.94** and **0.90** for K=32 and K=64 respectively. This is a significant improvement compared to the other architectures. Figure 2 demonstrates that this architecture perform also better with respect to the AUC. We obtained, on the Kaggle public leaderboard, **0.83** and **0.78** accuracy on the test dataset for K=32 and K=64 respectively.

To compare this architecture with the downsized ResNet, we can note that the downsized ResNet performed slightly better on the public Kaggle leaderboard (see Table 2). Yet, the mixed model presented in Table 1 shows better results with respect to metrics such as balanced accuracy, recall or AUC on the validation dataset (see Ta-

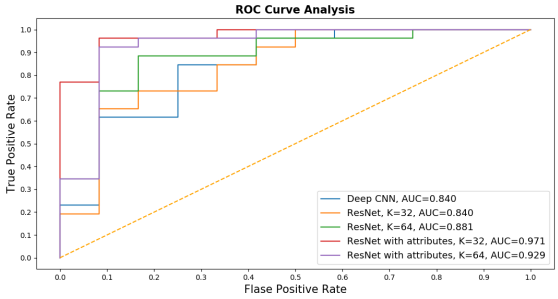

Figure 2: Roc curve and AUC for the different models

ble 2 and 2). For exemple the AUC for the ResNet with the clinical attributes and K=32 is **0.97** while the AUC for the ResNet which performed the best on the public leaderboard is **0.88**. Because the public leaderboard is calculated on such a short number of patient, we think that the mixed model is the best model we built for this classification task.

2.4.5 Gated attention models

We tried to implement a Gated attention models as presented in Sahasrabudhe et al. 2020. The full model is composed of three different blocks:

- Downsized ResNet: outputs a prediction y_i^{CNN} for each patient using the lymphocytes pictures.
- Multi Layer Perceptron for the clinical attributes: outputs a prediction y_i^{MLP} for each patient using its age a_i and lymphocytes concentration c_i.
- A Gated Network: output a probability π_{cnn} used to reconstruct the final prediction using the ResNet's prediction y_i^{CNN} and the patients clinical attributes a_i and c_i.

Finally, we reconstructed the final prediction as

$$y_i = \pi_{cnn} y_i^{CNN} + (1 - \pi_{cnn}) y_i^{MLP}$$

The idea behind this architecture is to use both the lymphocytes pictures and the clinical attributes for the classification and to let the model decide how much attention it puts on the clinical attributes or on the images. However this architecture seemed promising we did not manage to correctly train it.

Conclusion

We have successfully implemented a deep learning model for the classification of lymphocytes from blood cells. We managed to tackle this multi-learning instance learning problem and obtain satisfying results on the public leaderboard. Our best model on the bublic leaderboard, is the downsized ResNet which yielded a score of 84.93%, our second best model is the model presented in Table 1, that combines both the features extracted by the ResNet and the patient clinical attributes, which yielded a score of 83.11. Our conclusion is that deep learning models are able to provide relevant and good solutions for this classification task.

References

[He+16] Kaiming He et al. "Deep residual learning for image recognition". In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Vol. 2016-December. 2016. DOI: 10.1109/CVPR.2016.90.

[Sah+20] Mihir Sahasrabudhe et al. "Deep multi-instance learning using multi-modal data for diagnosis of lymphocytosis". In: *IEEE Journal of Biomedical and Health Informatics* (2020). ISSN: 21682208. DOI: 10.1109/JBHI.2020.3038889.