



# **Neural Network Classification of White Blood Cell Images**

***Capstone Project***

Michał Czapski  
[czapski.michal@gmail.com](mailto:czapski.michal@gmail.com)

# Contents

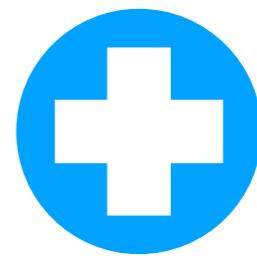
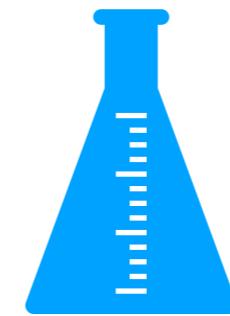
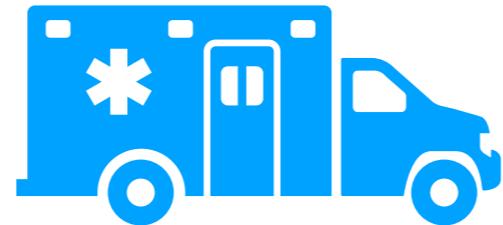
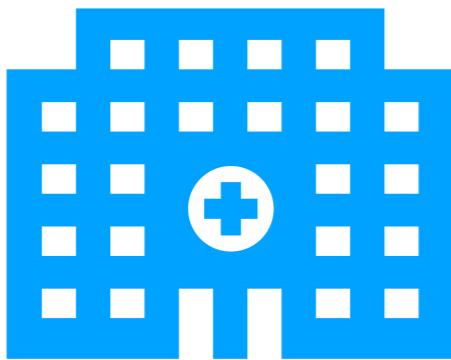
- **Introduction**
- **Data Set**
- **Image Classification**
  - **CNN**
  - **Learning Process**
  - **Initial Model**
  - **Model Optimization**
  - **Final Model**
- **Assumptions and Limitations**
- **Recommendations**

# Introduction

Blood cell classification is laborious work done manually by pathologist that could be automated to save time and resources



Automated process of image classification could be used by medical companies and institutions that could retrieve more information about blood related diseases or use it for other medical applications that require fast analysis



Hospitals | Clinics | Cell Laboratories | Medical Companies

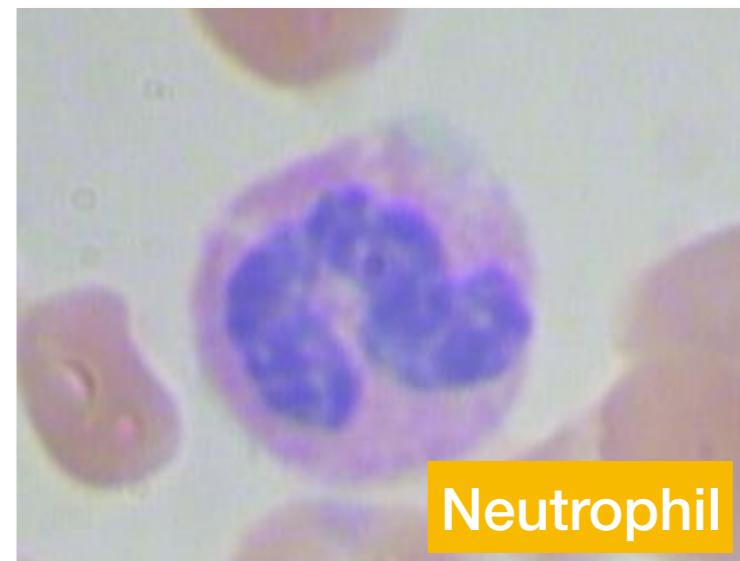
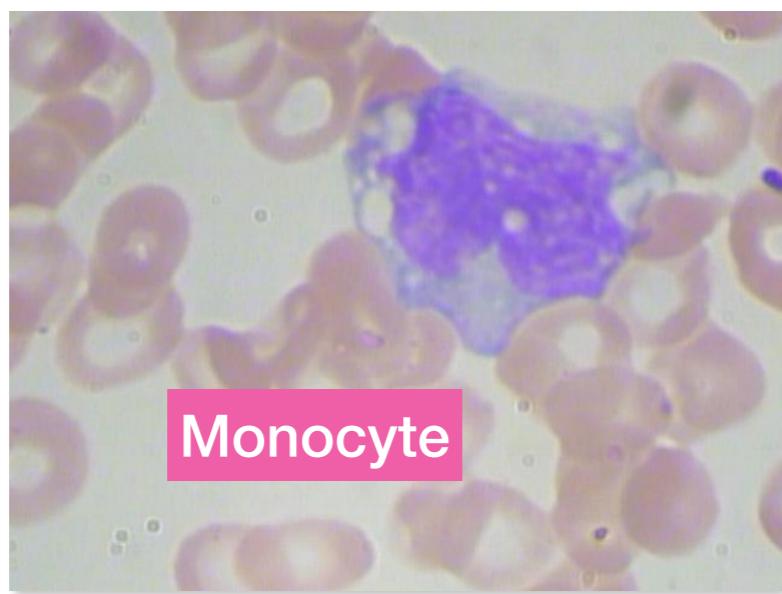
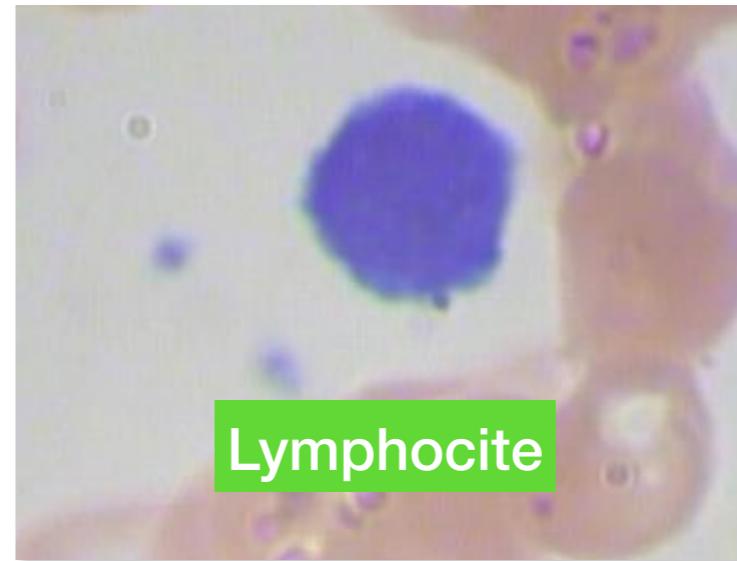
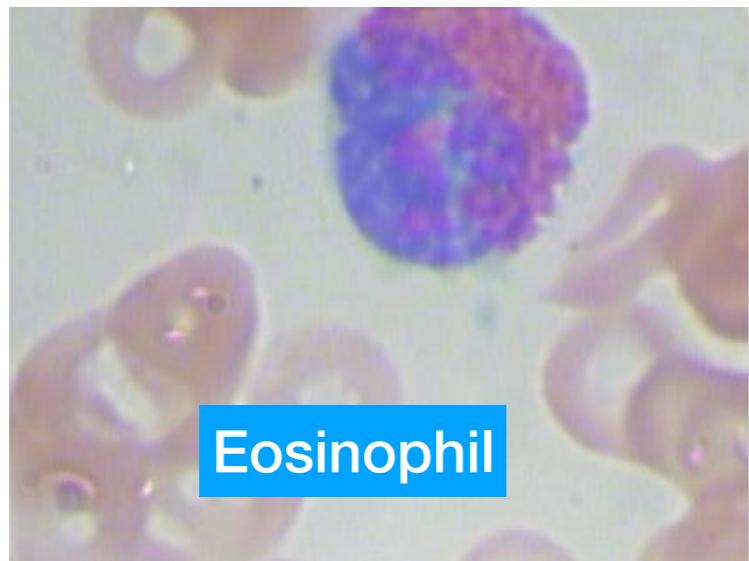


Samples Collection

Automated Image Classification

Decision

The goal of this study was to build a an optimized image classifier that can classify four types of white blood cells with accuracy around 90%

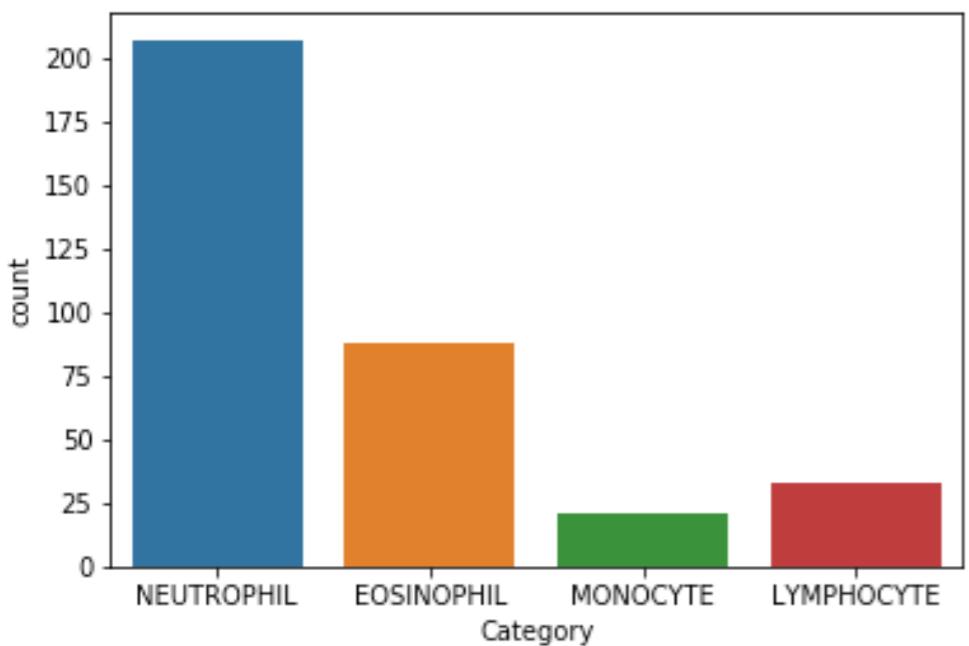
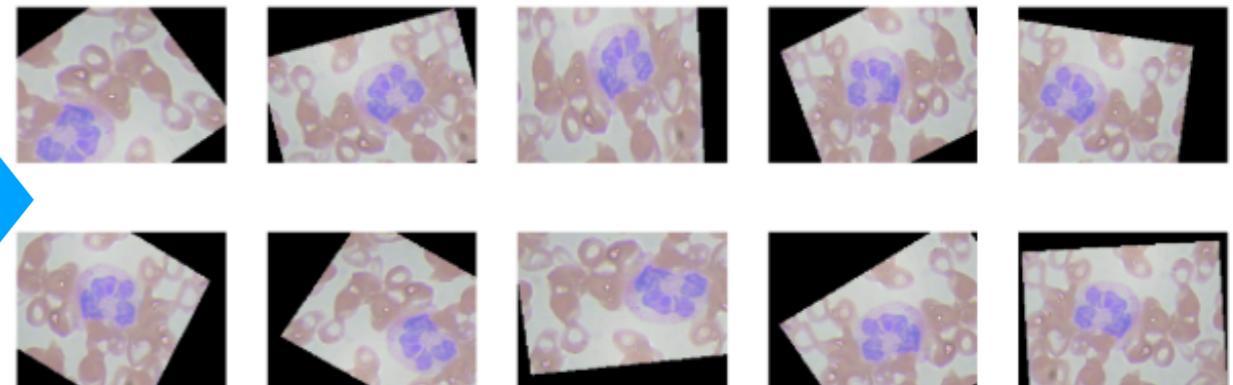
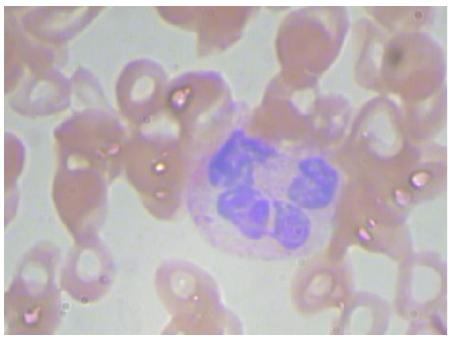


# Data Set

Image augmentation techniques were used to increase the number of original 366 images of four classes that differ significantly in number per class.

### Image Augmentation

- Rotation
- Width / Height Shift
- Shear
- Zoom
- Rescaling (1/255)
- Vertical /Horizontal Flip



8  
Image  
Augmentation

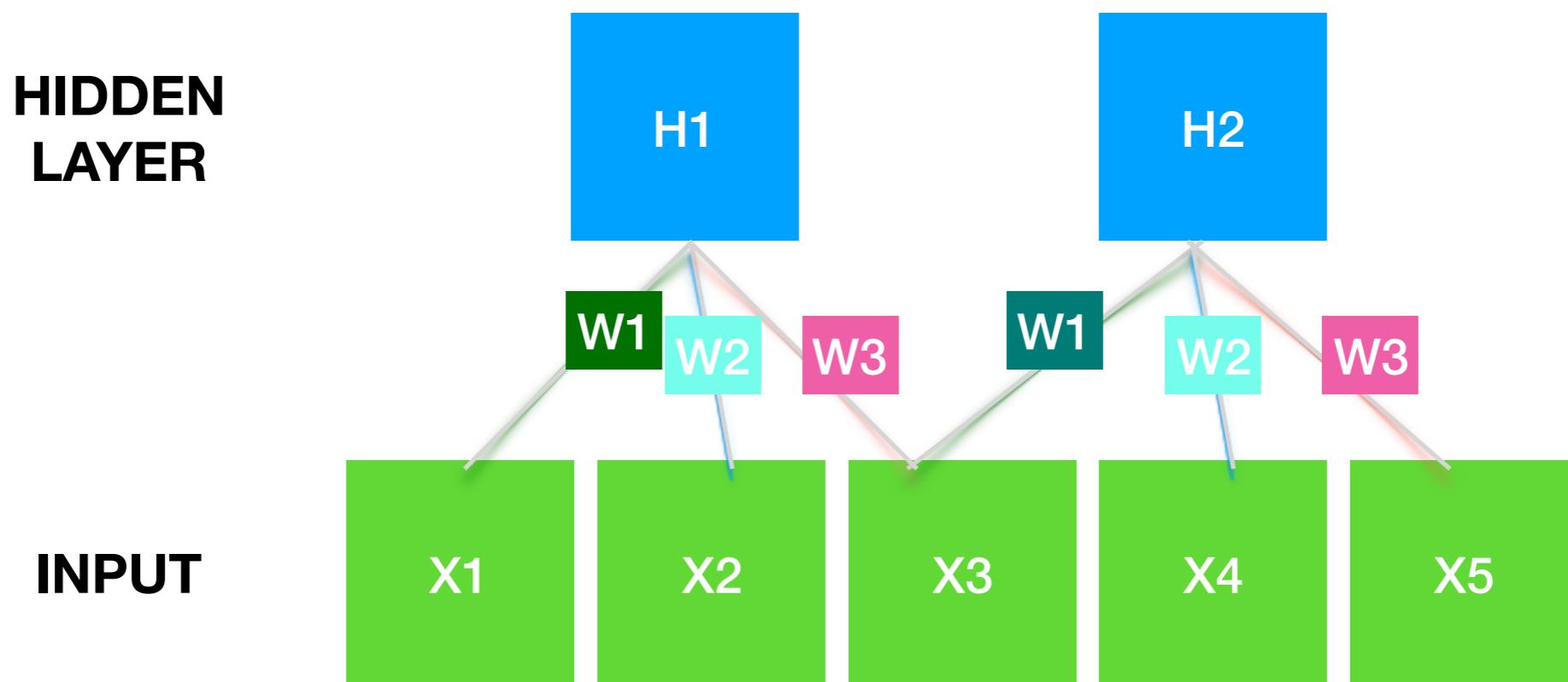
Random

Around 2600 images per class  
decided into  
train / validation / test sets in  
ratio  
60:20:20

# Image Classification

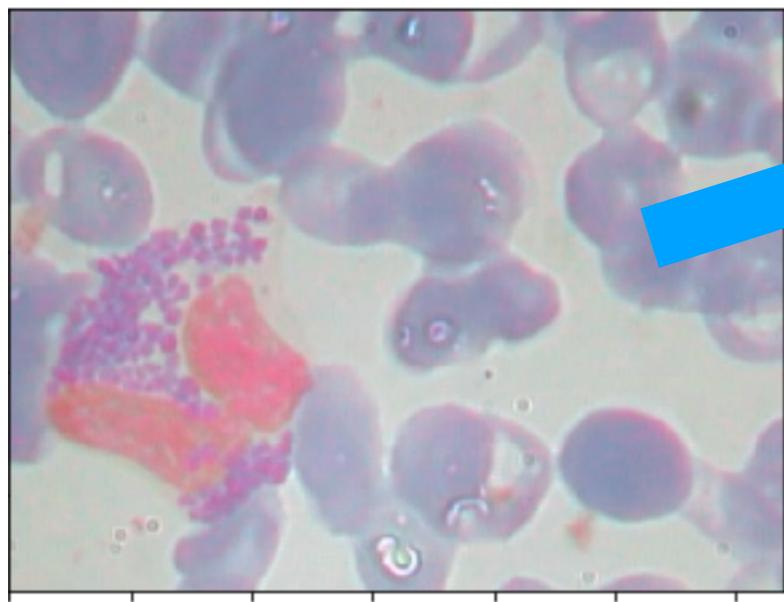
# *Convolutional Neural Networks*

A convolutional neural network is a class of deep, artificial neural networks that found application mostly in image and sound analysis, whose main feature is significant parameter drop through weight sharing between neurons



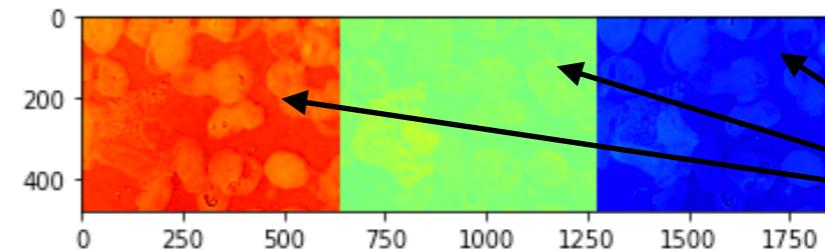
# *Learning Process*

CNN layers' output visualization technique of untrained simple network (C-R-P) demonstrated how image is processed from layer to layer



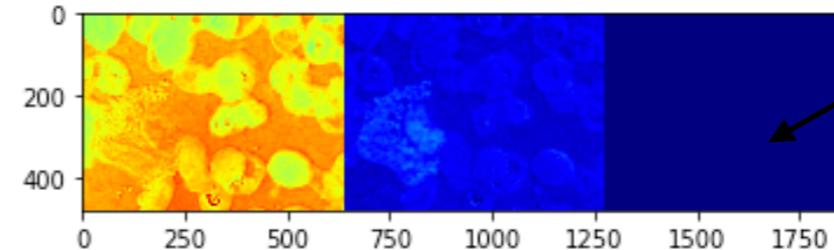
**INPUT**

**CONVOLUTIONAL LAYER  $3 \times 3 \times 3$**



**3 filters**

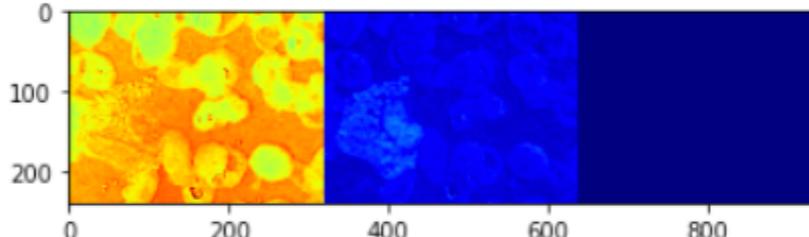
**RELU LAYER**



**Zero'ing negative activations**

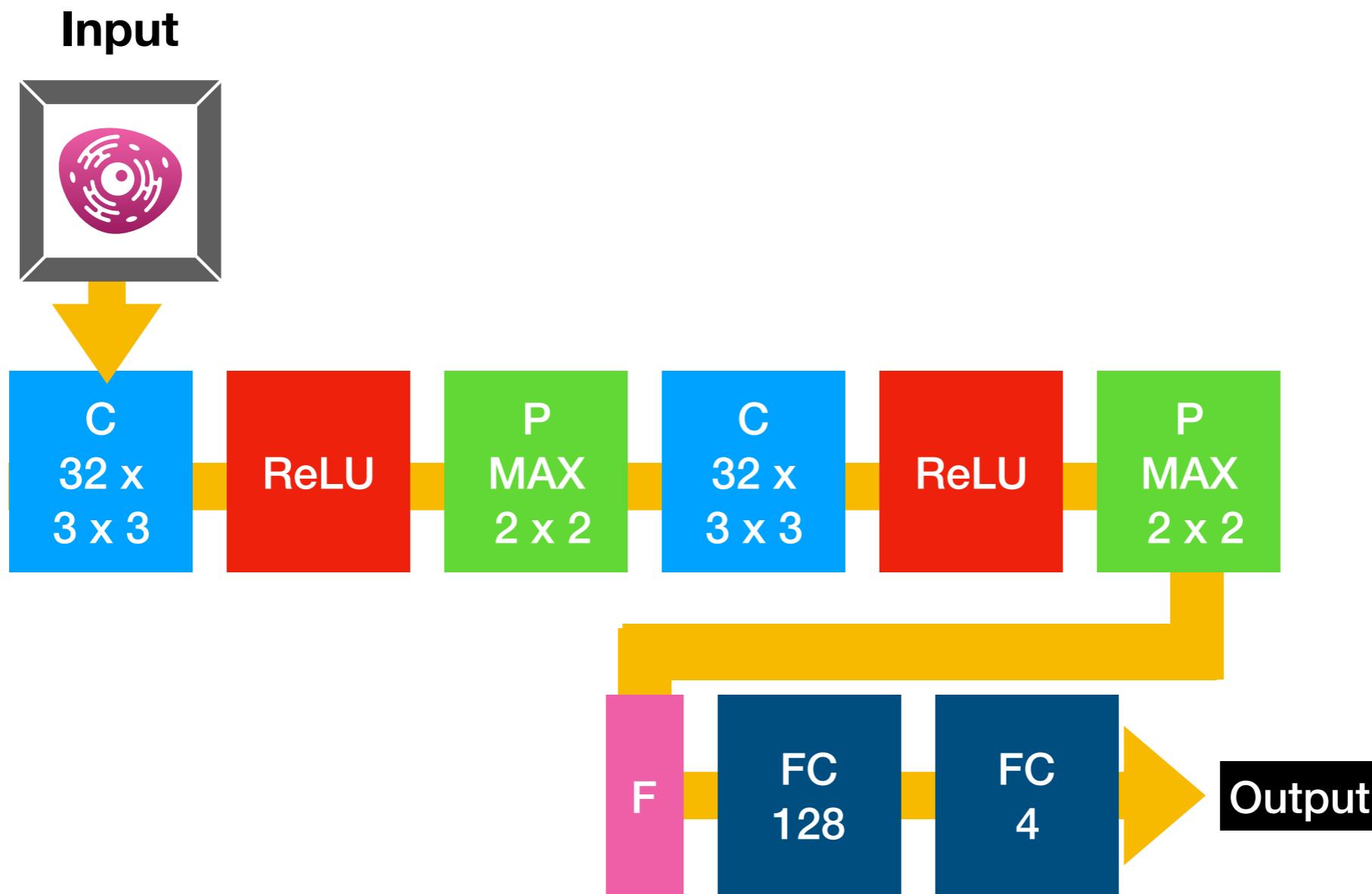
**Dimension change**

**MAX POOLING LAYER  $2 \times 2$**



# *Initial Model*

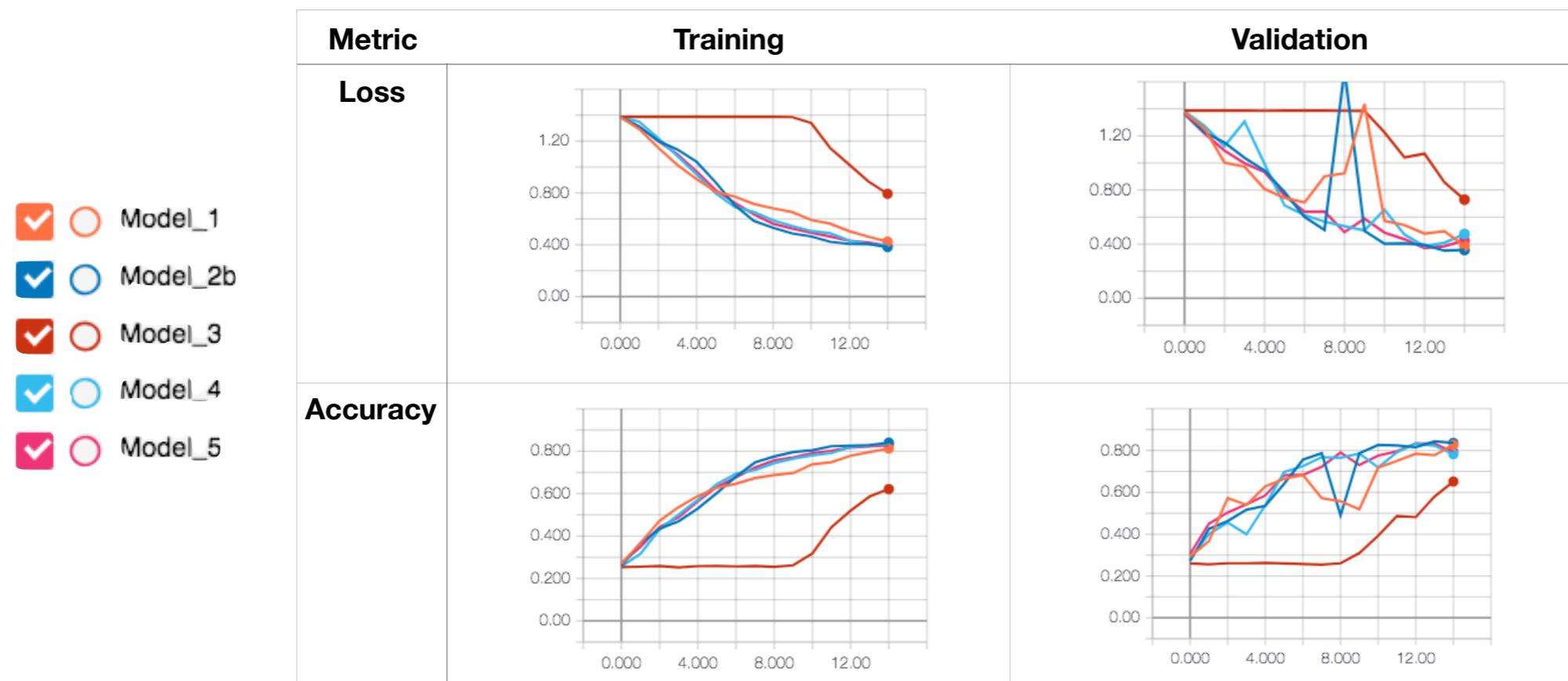
The initial model comprised of simple architecture based on most common practices i.e. several stacks of convolutional, ReLU and pooling layers followed by fully connected layers for final classification.



# *Model Optimization*

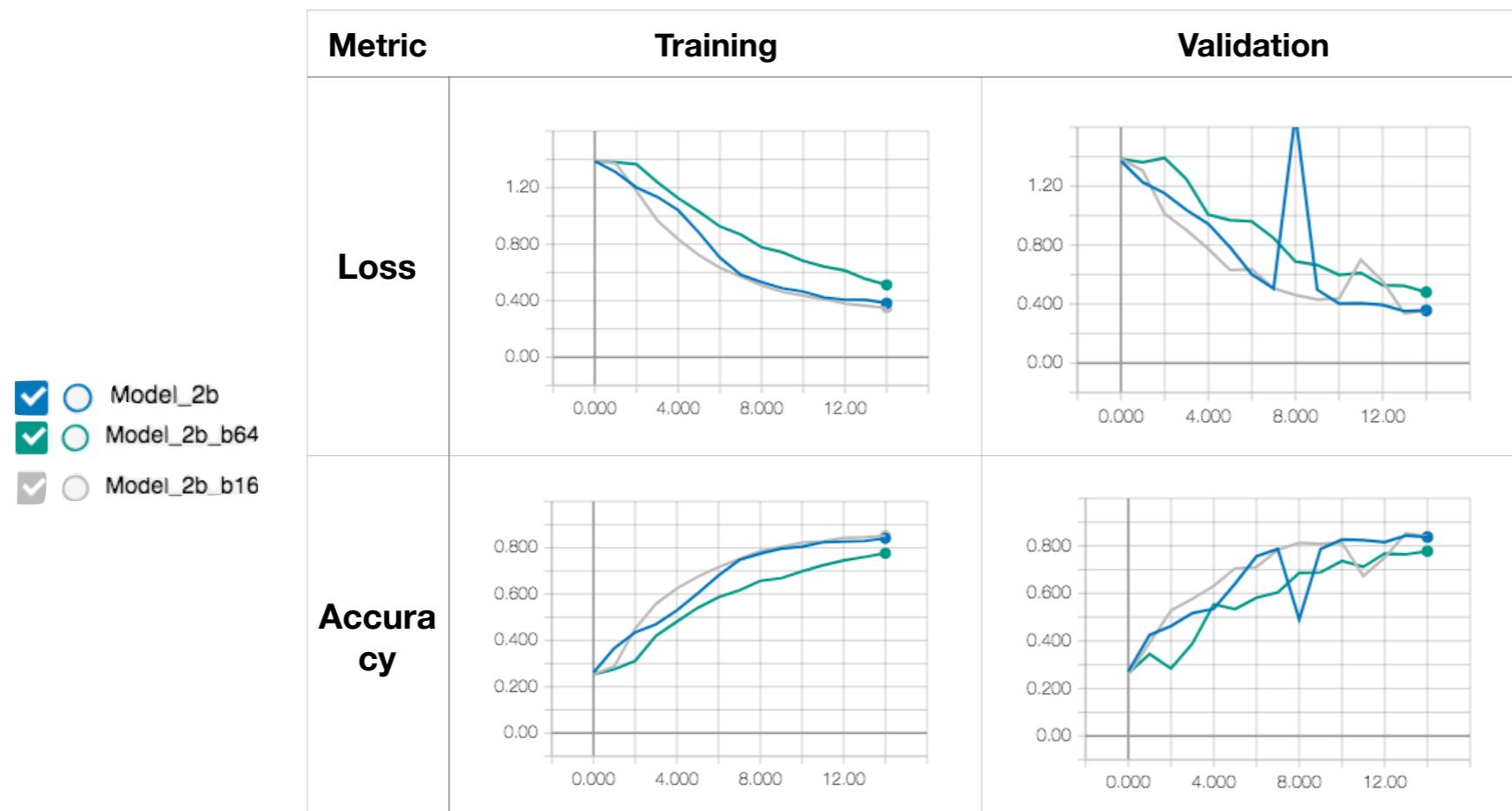
The initial model with additional C-R-P stack proved to have the lowest validation loss in contrary to the initial model with two additional C-R-P stacks to have the highest loss.

| Step | Description        | Architecture            | Model name | Validation Loss | Next step         |
|------|--------------------|-------------------------|------------|-----------------|-------------------|
| 1    | Initial model      | CRP-CRP-F-FC-FC         | model_1    | 0.50            | increase capacity |
| 2    | add CRP            | CRP-CRP-CRP-F-FC-FC     | model_2b   | <b>0.40</b>     | increase capacity |
| 3    | add CRP            | CRP-CRP-CRP-CRP-F-FC-FC | model_3    | <b>0.72</b>     | go back to step 2 |
| 4    | add FC to model_2b | CRP-CRP-CRP-F-FC-FC-FC  | model_4    | 0.46            | go back to step 2 |
| 5    | add D to model_2b  | CRP-CRP-CRP-F-FC-D-FC   | model_5    | 0.42            | go back to step 2 |



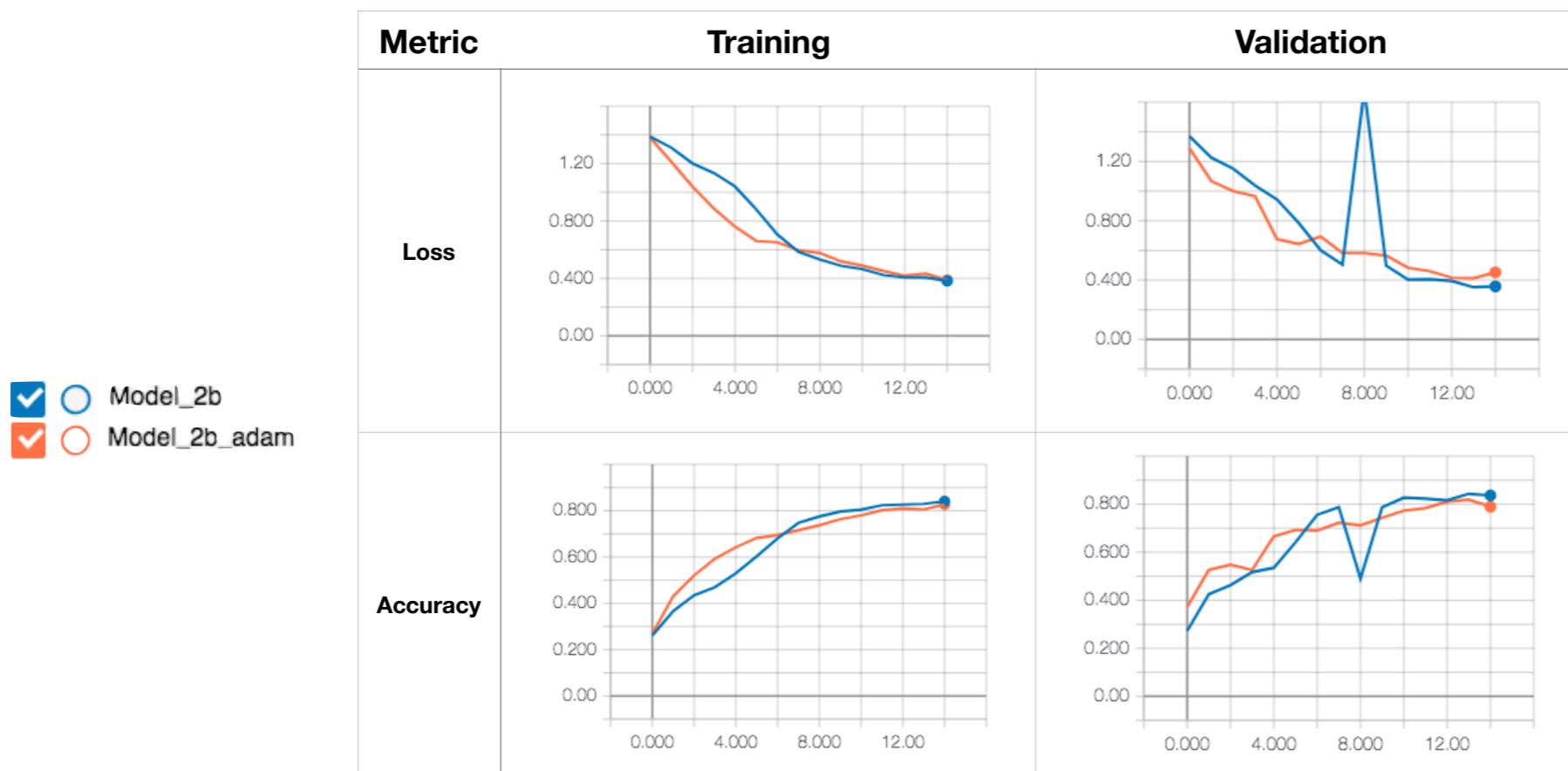
The increase in batch size increased the validation loss, on the other hand, the decrease in batch size speeded up the learning process, but the final result was 5% worse than the original batch size

| Batch size | Model name | Validation loss | Next step                  |
|------------|------------|-----------------|----------------------------|
| 32         | model_2b   | 0.40            | increase batch size        |
| 64         | model_b64  | 0.53            | decrease batch size        |
| 16         | model_b16  | 0.42            | go back to batch size = 32 |



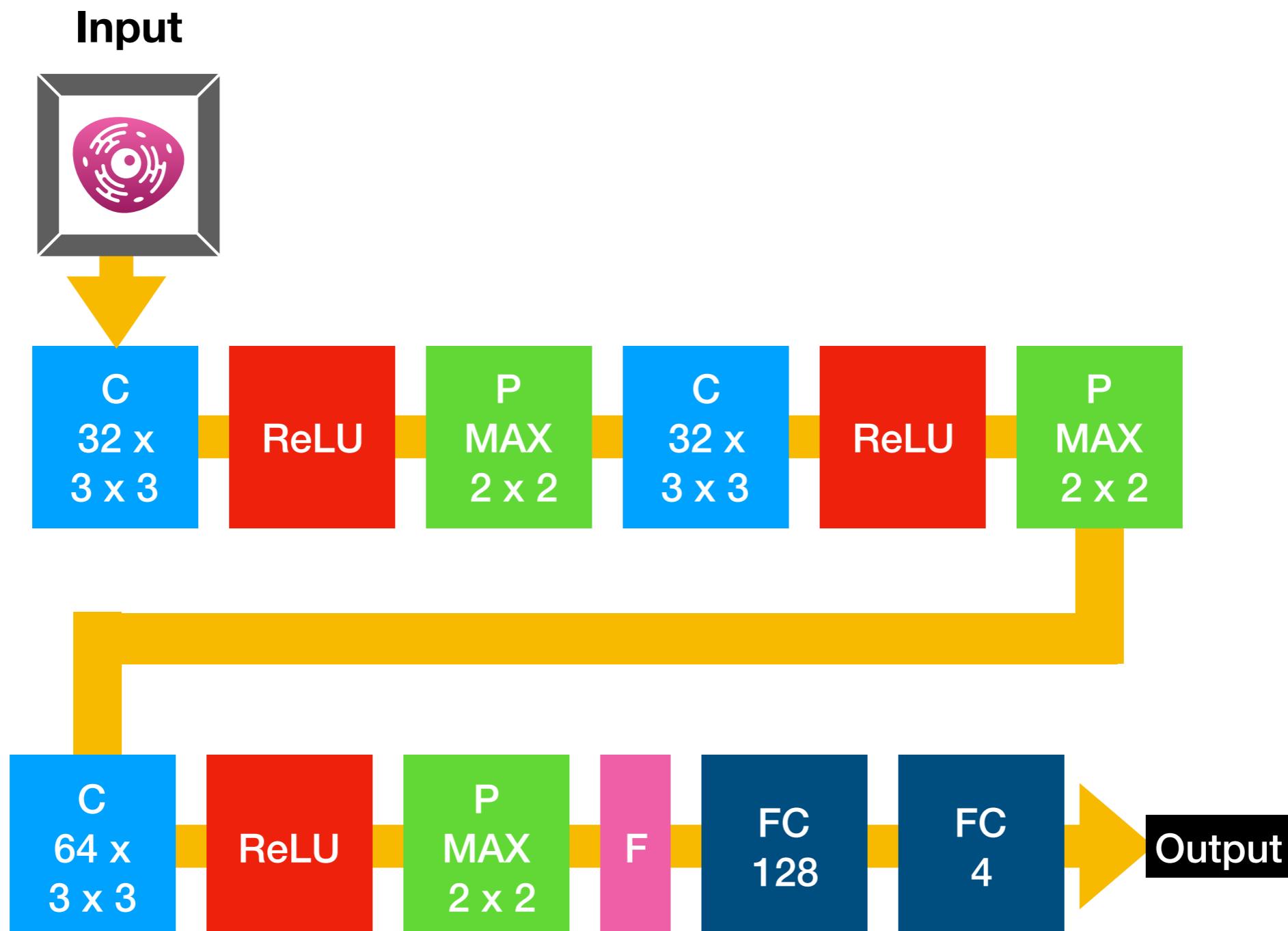
Even though Adam optimizer generally is recommended when training a new network, and the training process went faster, Adadelta achieved lower final validation loss

| Optimizer | Model name    | Validation Loss | Next step             |
|-----------|---------------|-----------------|-----------------------|
| Adadelta  | model_2b      | 0.40            | change optimizer      |
| Adam      | model_2b_adam | 0.45            | move back to Adadelta |

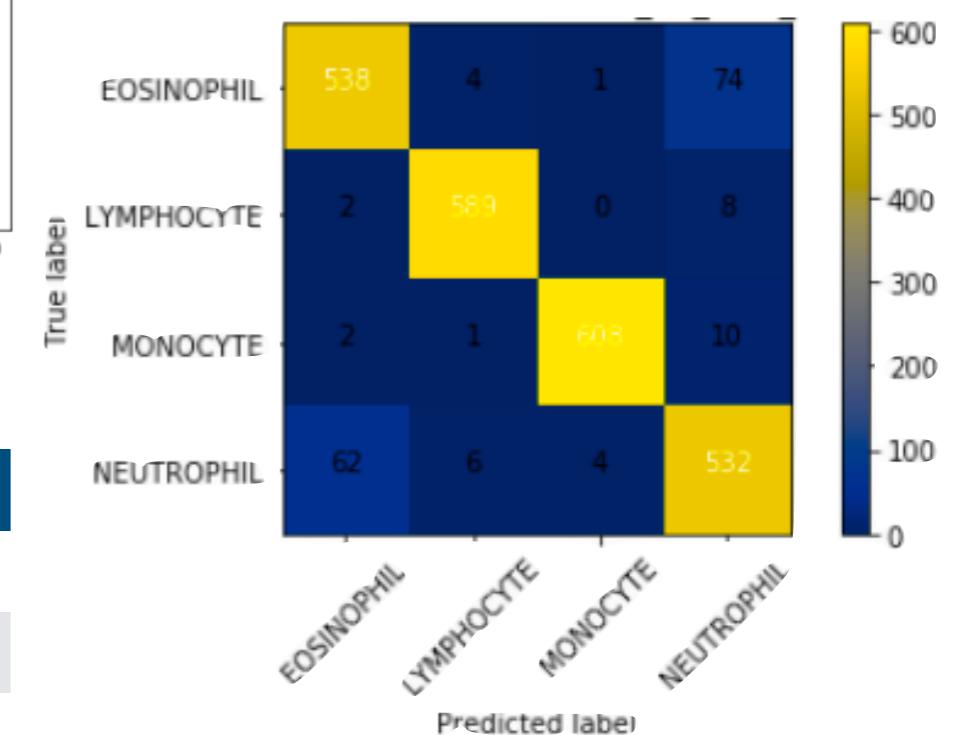
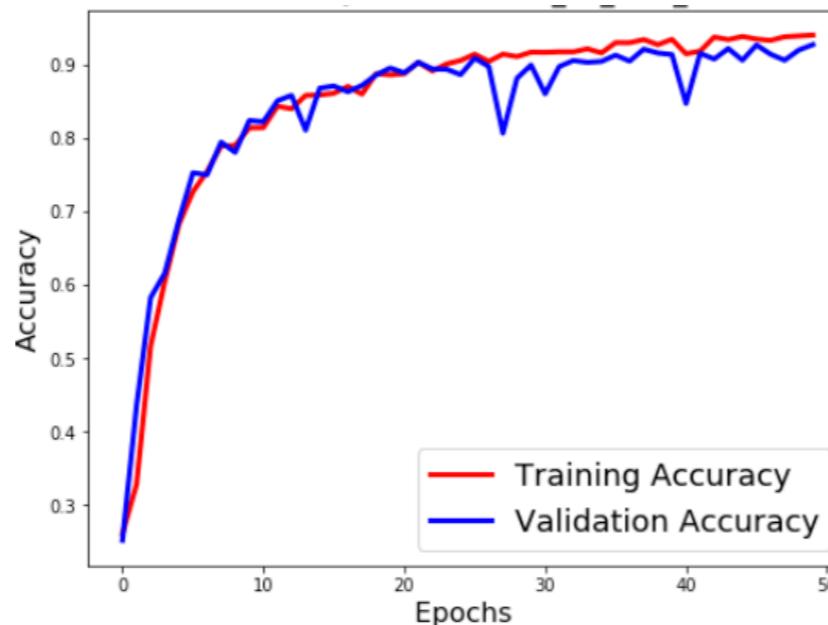
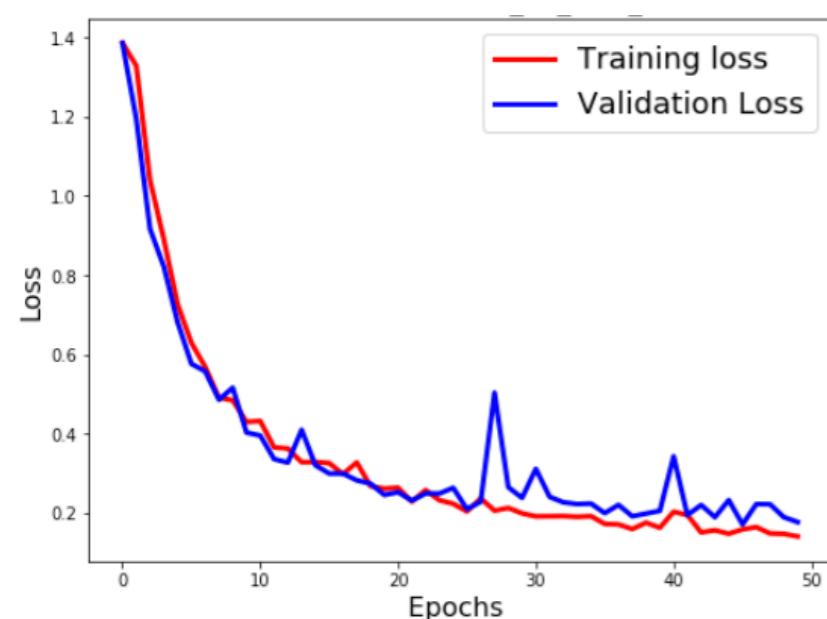


*Final Model*

The final model comprised of 3 CRP stacks, one flatten layer and two fully connected layers, trained during 50 epochs on the batch size = 30.



The final model despite of its simple architecture achieved accuracy above 90% on testing set, which is comparable to the results, found online, achieved on more complex models.

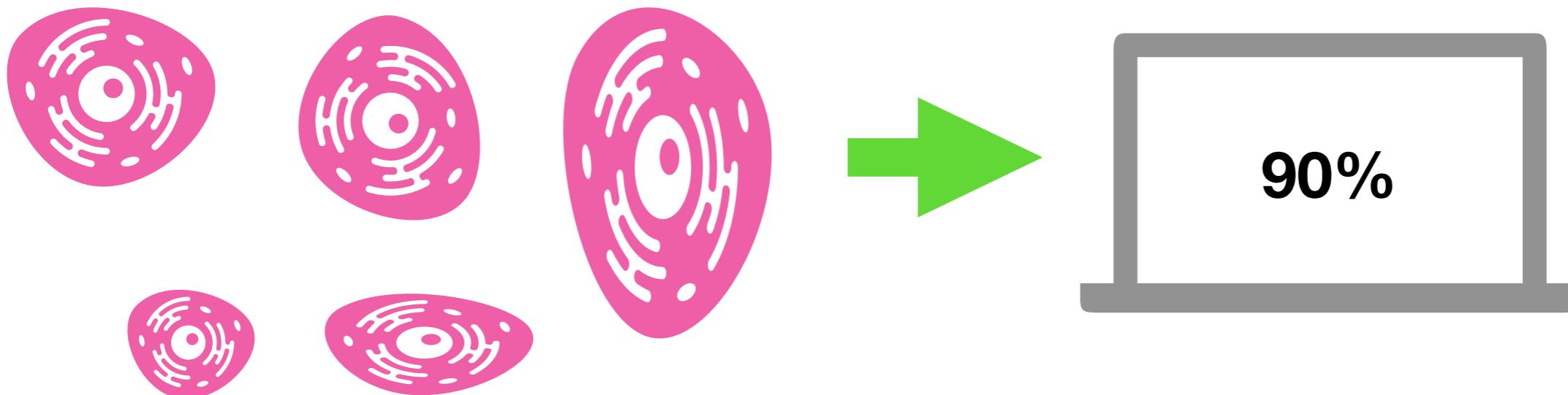


| Class                  | Precision   | Recall      | F-score     | Support     |
|------------------------|-------------|-------------|-------------|-------------|
| Eosinophil             | 0.89        | 0.87        | 0.88        | 617         |
| Lymphocyte             | 0.98        | 0.98        | 0.98        | 599         |
| Monocyte               | 0.99        | 0.98        | 0.99        | 621         |
| Neutrophil             | 0.85        | 0.88        | 0.87        | 604         |
| <b>Average / Total</b> | <b>0.93</b> | <b>0.93</b> | <b>0.93</b> | <b>2441</b> |

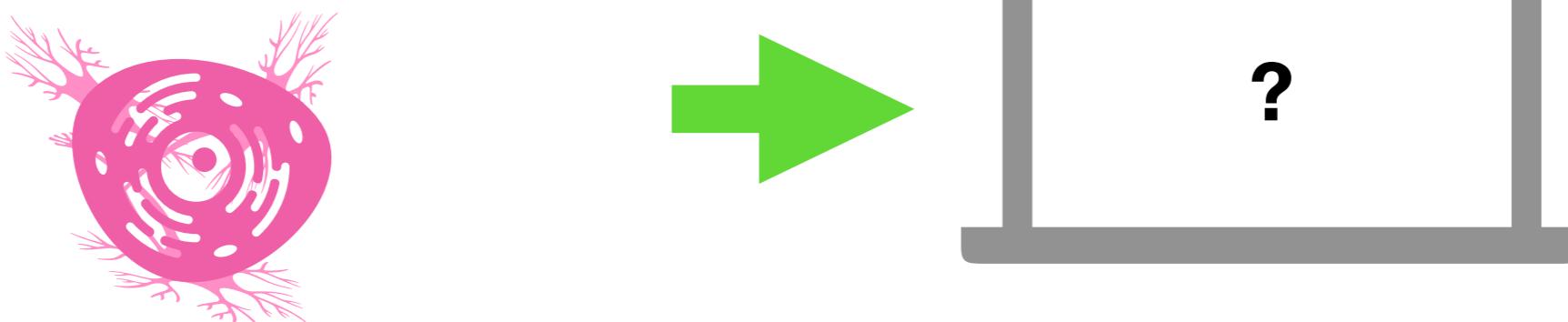
# **Assumptions and Limitations**

Small number of original images might have an impact the classifier's robustness and accuracy as well as on the model optimization results

**Original augmented data set**

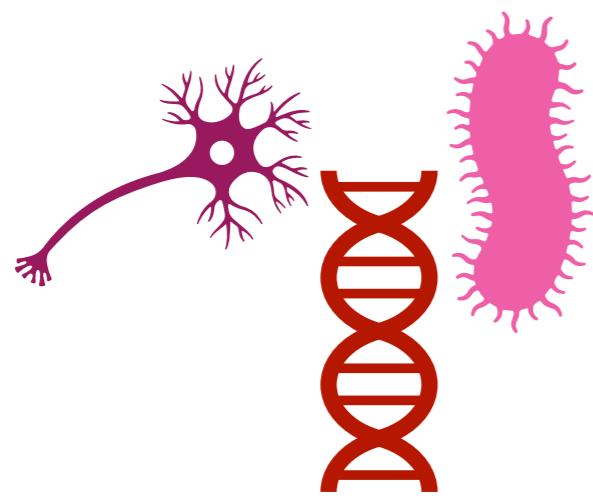
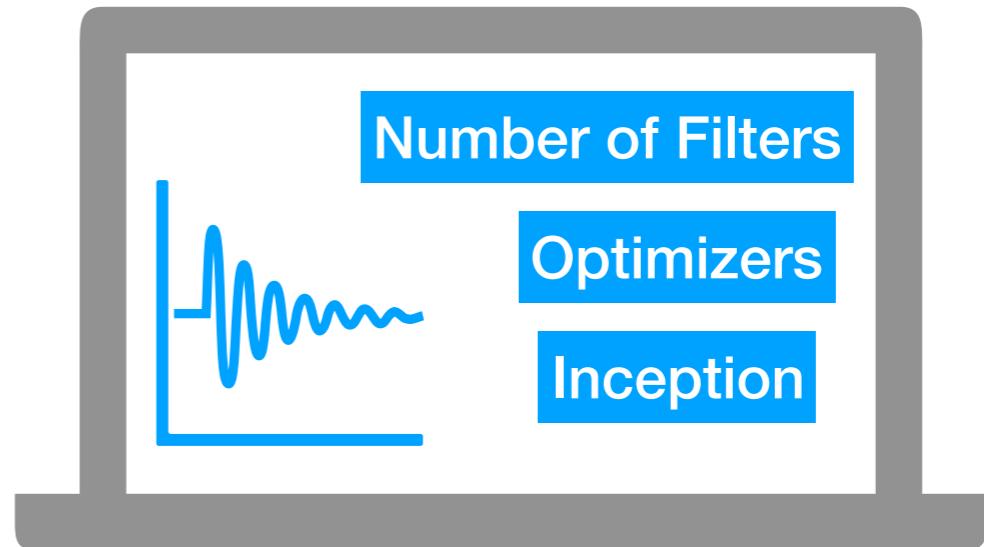


**New image**



# **Recommendations**

There are many other parameters that could be tested to improve the classifier even more



Transfer learning and fine-parameter tuning of the top layers could help to achieve even better result

Adding more classes to the training could make the classifier more versatile

