Using Deep Learning to Classify Lymphoma Types

Extraterrestrial Confounding Cats

Nicole Firestone, Keerthana Jegatheesan, Elisa Tau, Alejandro Cartes and Zachary Richards
TA: Daniela Grandón
Mentor: Mauricio Cerda

LA SERENA SCHOOL FOR DATA SCIENCE 2022

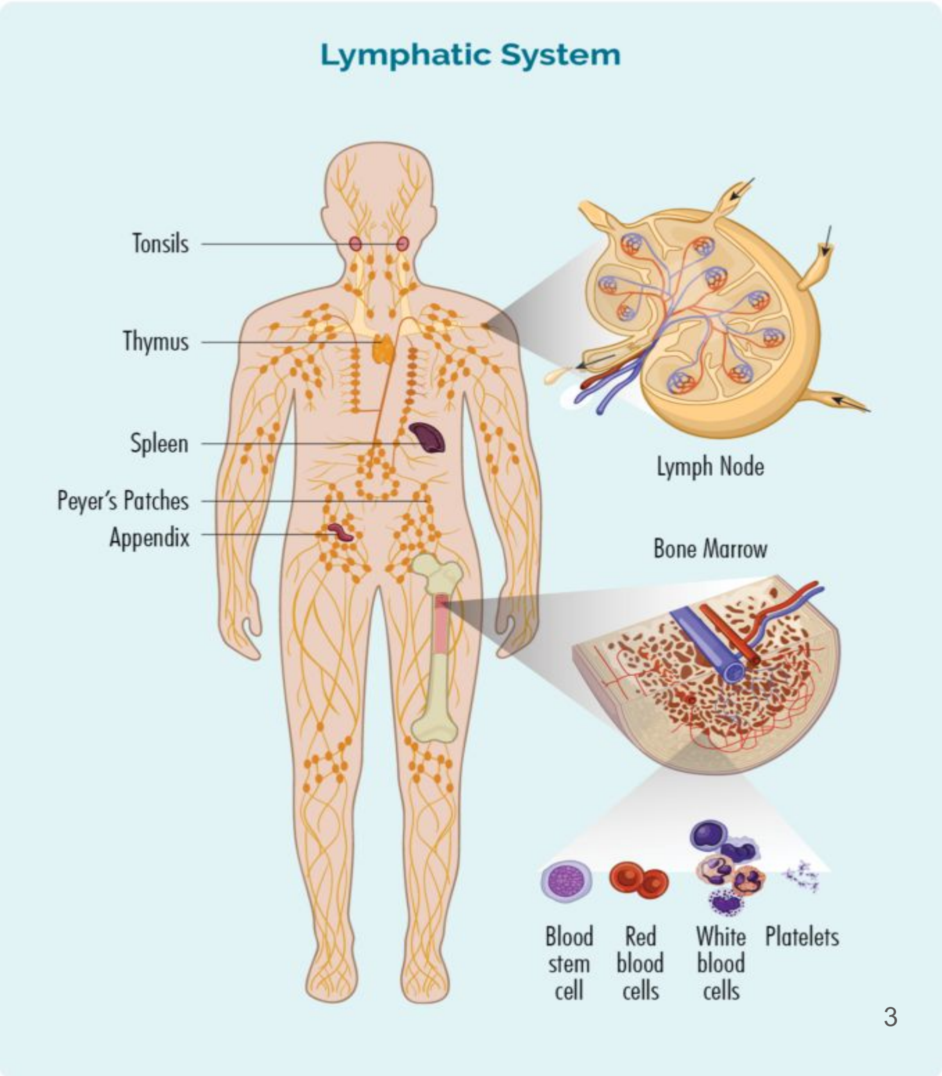
12/8/2022

Outline

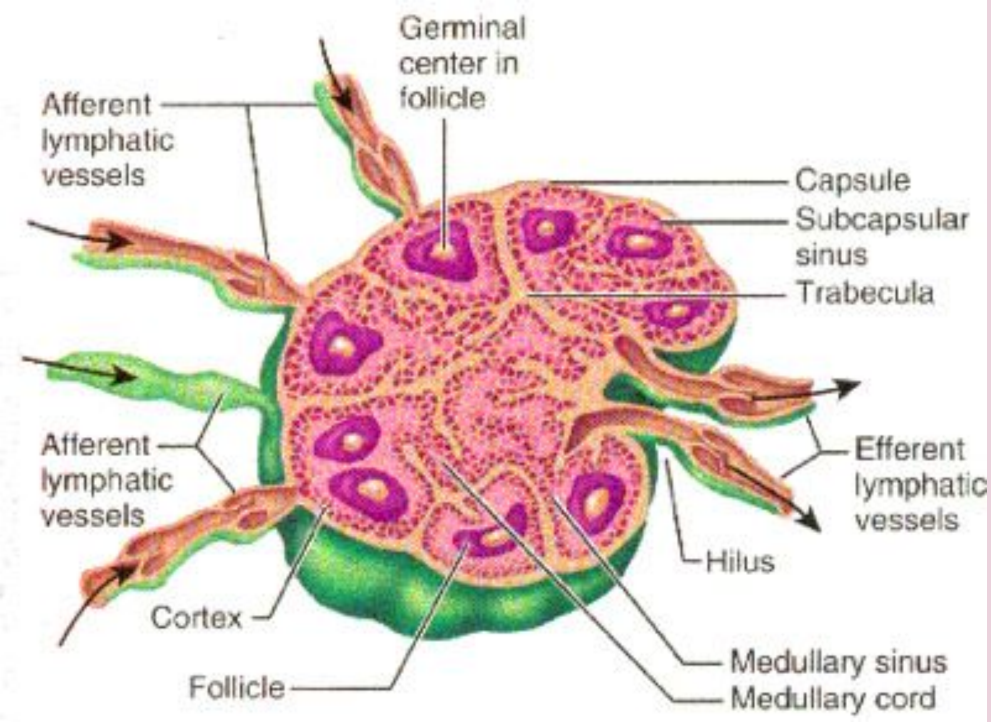
1. Scientific Context
2. Our data
3. Methods
4. Results
5. Conclusions
6. Future work and connections to astronomy

Scientific Context: What is lymphoma?

Lymphoma is a cancer of the lymphatic system.

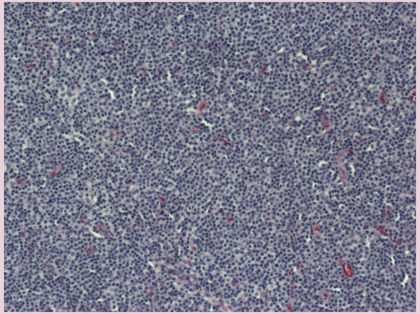


Scientific Context: Anatomy of a Lymph Node

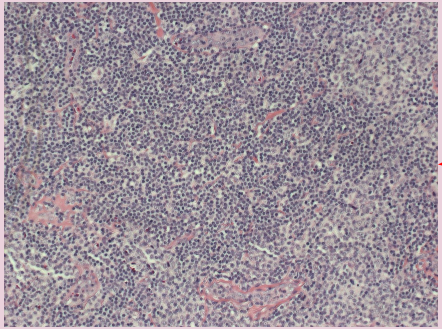


Scientific Context: Three Classes of Lymphoma

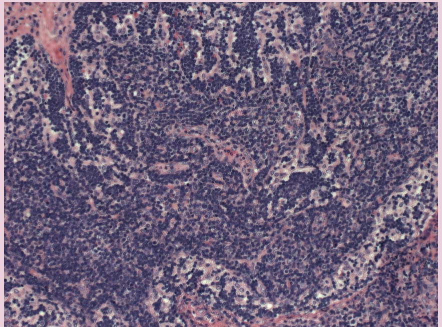
Chronic Lymphocytic Leukemia (CLL)



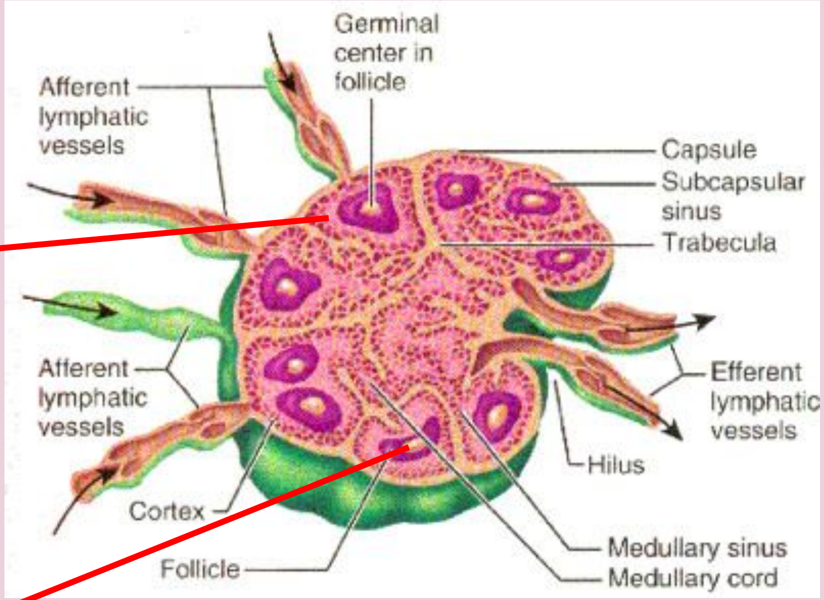
Mantle Cell Lymphoma (MCL)



Follicular Lymphoma (FL)

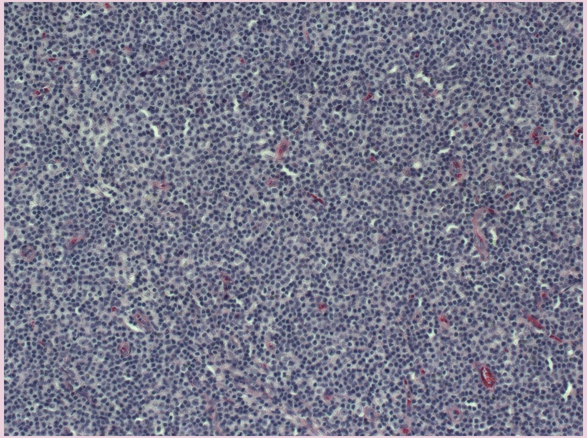


Anatomy of a Lymph Node

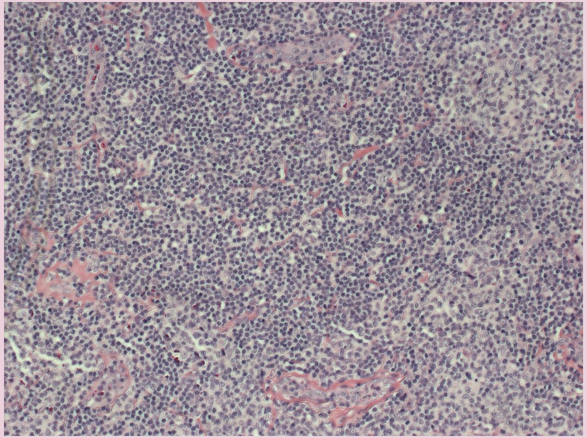


Scientific Context: Three Classes of Lymphoma

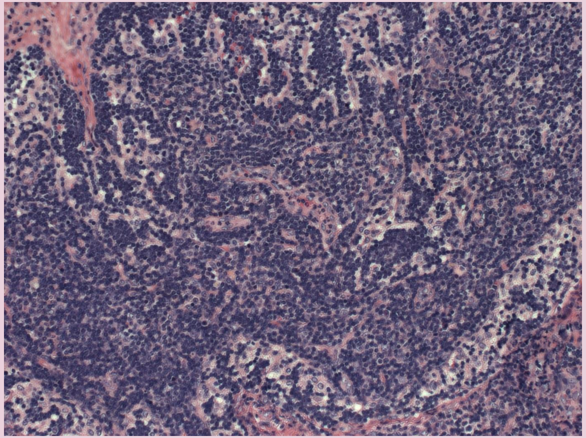
Chronic Lymphocytic Leukemia (CLL)



Mantle Cell Lymphoma (MCL)



Follicular Lymphoma (FL)



Biopsies stained with hematoxylin and Eosin → Lymphocytes appear blue

Motivation

Challenge:

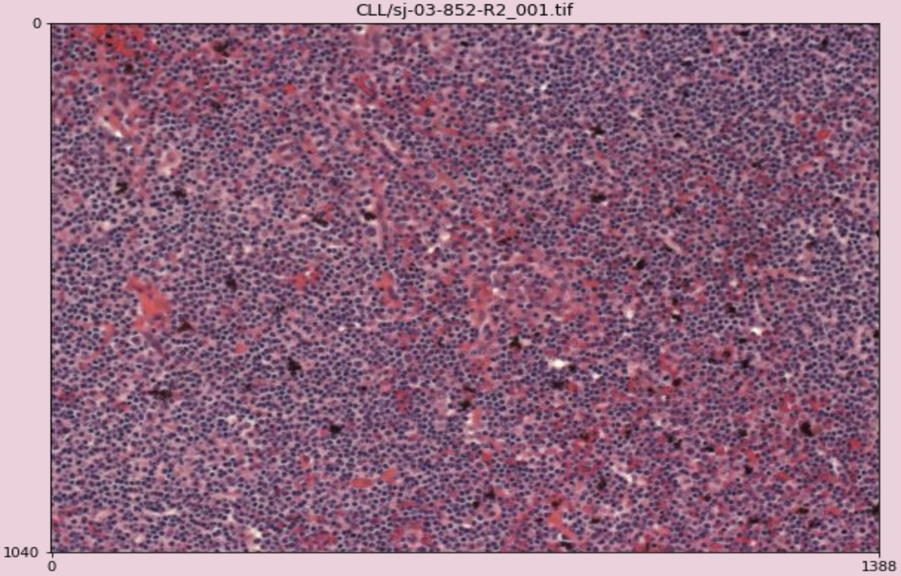
Lymphoma classification is inaccessible, unstandardized, and time sensitive- requires efficient and consistent classification to assign most promising treatment options.

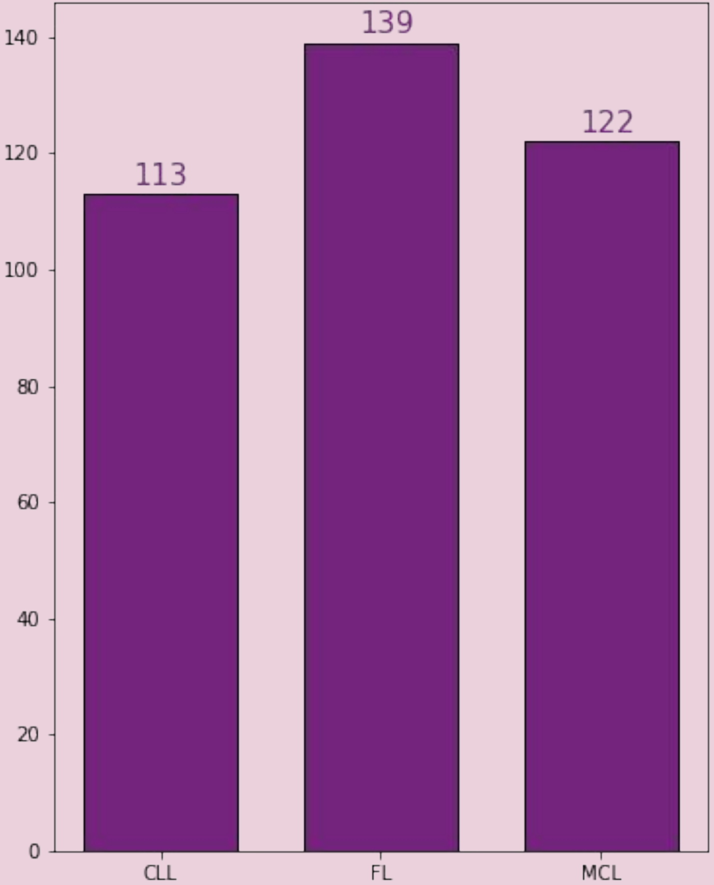
Solution:

Develop a convolution neural network to properly classify 3 types of lymphoma in order to speed up and standardize the diagnostic process, making diagnostics tools more accessible.

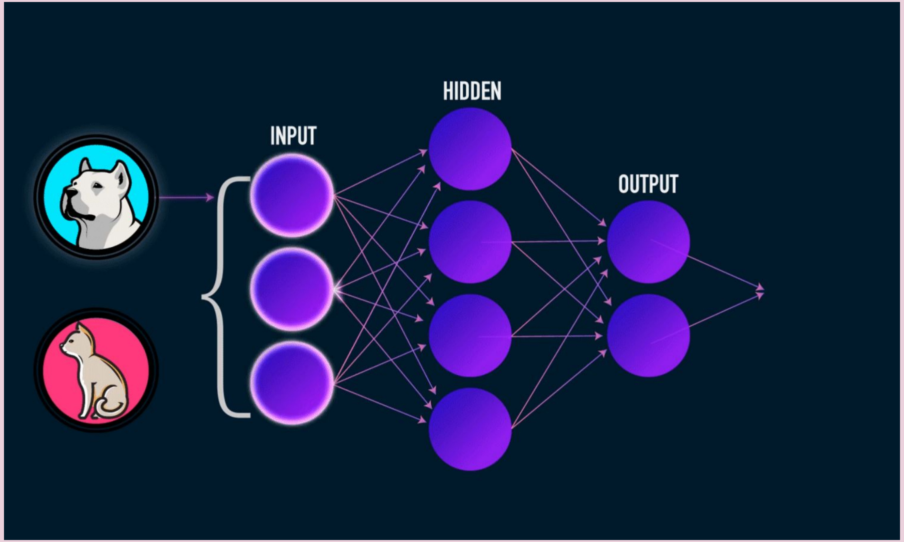
The Image Data

- Format: .tif
- Count: 374
- Classes 3
- Dimensions: 1040x1388 pixels
- Channels: 3 (rgb)





Why use a convolutional neural network?



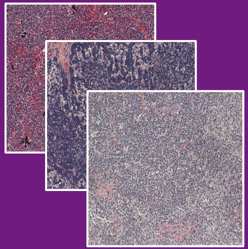
- Useful for processing images (as opposed to data sets)
- Robust and computationally efficient with large sample of data
- Does not require human supervision

Some common architectures: LeNet-5 (1998), AlexNet (2012), VGG-16 (2014)

Overview of Analysis Process

1. **Compile Data**: Create image and type arrays
2. **Optimize Data**: Generate patches and implement augmentation
3. **Prepare Data for CNN**: Divide and preprocess data
4. **Train CNN**: Use Tensorflow and Keras do develop CNN
5. **Inspect Results**: Plot confusion matrix and model accuracy
6. **Troubleshoot**: Adjust parameters until satisfied

Concatenate Image Files Into Array



`lymphoma_imgs`

```
[[[[123  98 114]
   [122  98 115]
   [120  96 113]
   ...
   [102  64  78]
   [105  69  85]
   [115  78  96]]

  [[123  98 121]
   [122  98 119]
   [119  96 114]
   ...
   [101  64  78]
   [105  69  81]
   [115  78  92]]

  [[134 106 125]
   [131 106 125]
   [129 109 125]
   ...
   [102  67  79]
   [106  71  80]
   [118  82  90]]

  ...
  [[ 71  54 105]
   [ 70  54 107]
   [ 70  55 109]
   ...
   [141 142 194]
   [138 130 189]
   [117 109 169]]

  [[ 66  49  99]
   [ 66  49 103]
   [ 69  52 110]
   ...
   [118 112 169]
   [108  93 147]
   [ 91  75 133]]

  [[ 56  41  88]
   [ 56  41  93]
   [ 62  45  99]
   ...
   [ 90  80 135]
   [ 86  68 119]
   [ 84  62 106]]]]
```

Construct Image Types Array

	Integer	One-Hot Encoding
CLL	0	[1 0 0]
FL	1	[0 1 0]
MCL	2	[0 0 1]

`lymphoma_types`

```
[[1. 0. 0.]
 [1. 0. 0.]
 [1. 0. 0.]
 ...
 [0. 0. 1.]
 [0. 0. 1.]
 [0. 0. 1.]]
```

Compile Data

Optimize Data

Prepare Data for CNN

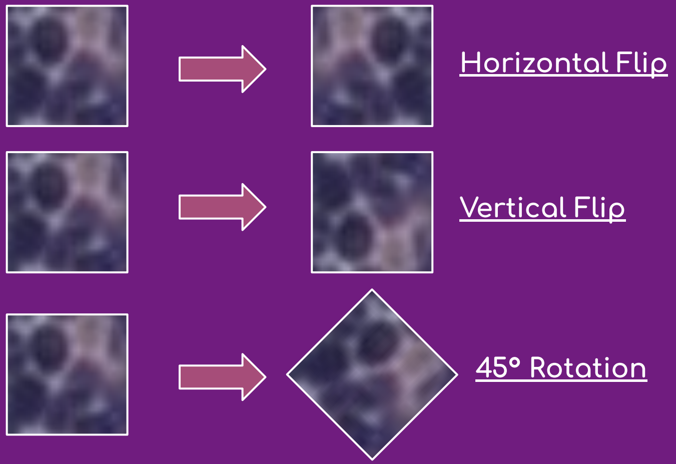
Train CNN

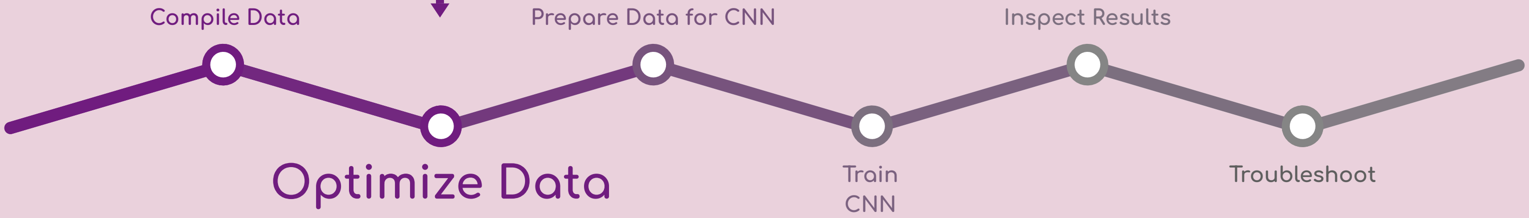
Inspect Results

Troubleshoot

Generate 32 x 32 Pixel Patches

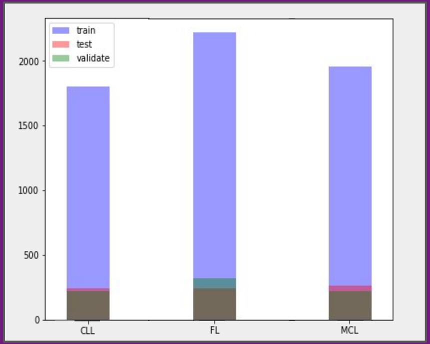
Apply Image Augmentations





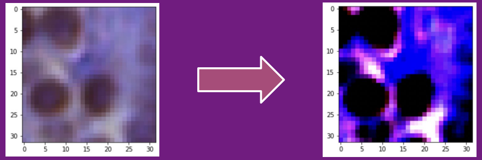
Optimize Data

Divide Data Into Subsets

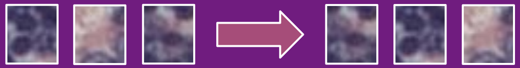


Training: 80% ; Validation: 10% ; Test: 10%

Preprocess Data



Normalize



Shuffle

Batch

Prepare Data for CNN

Compile Data

Optimize Data

Train CNN

Inspect Results

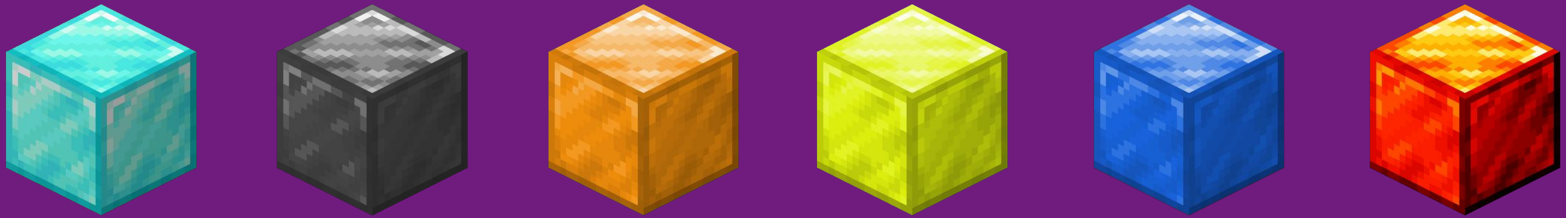
Troubleshoot

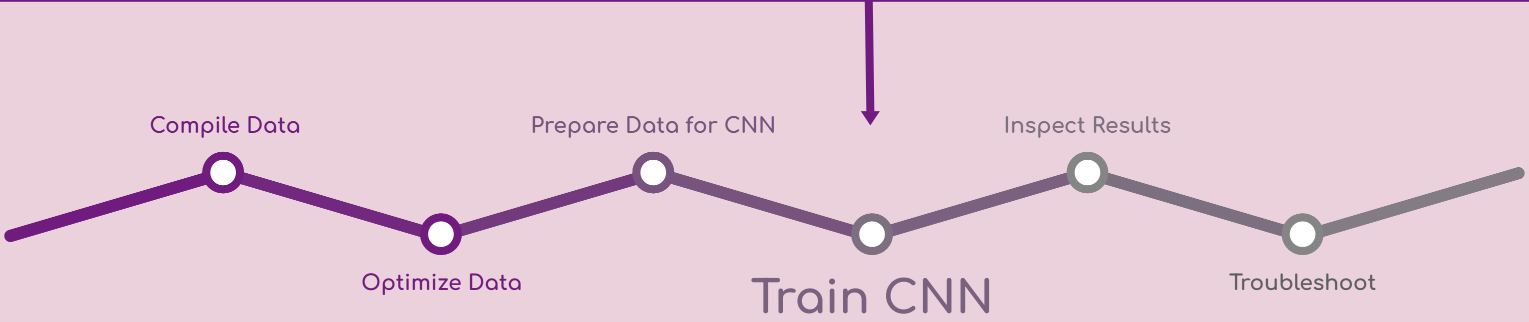
Building Blocks of a CNN

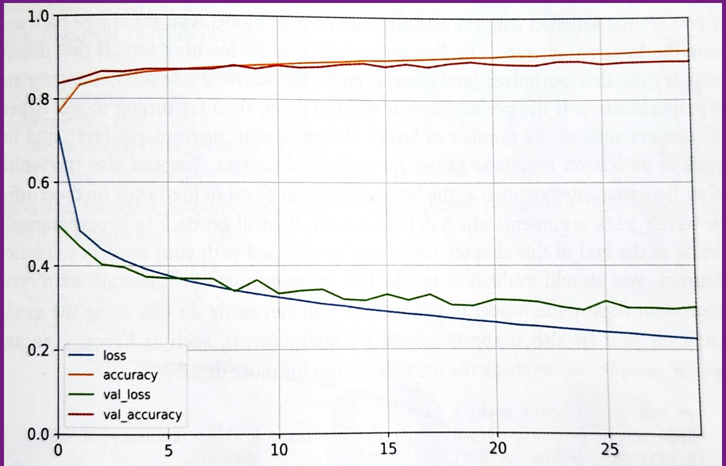
TensorFlow

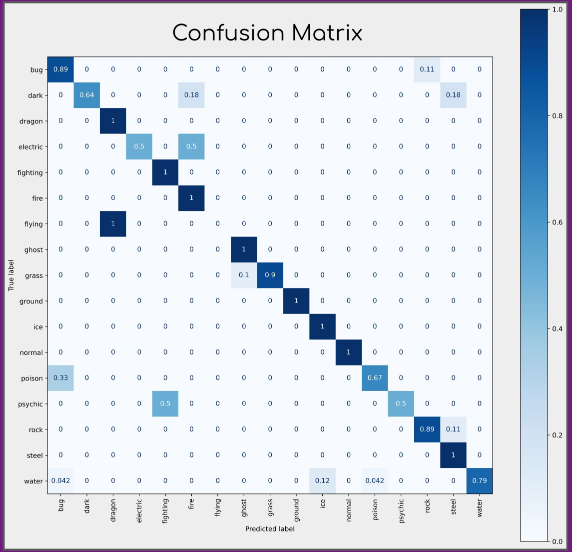
K

Convolution Activation Max Pooling Average Pooling Dense Flatten









Inspect Results

Compile Data

Optimize Data

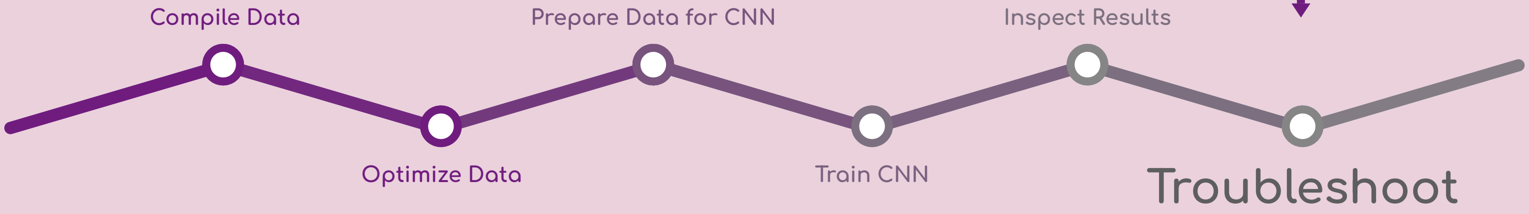
Prepare Data for CNN

Train CNN

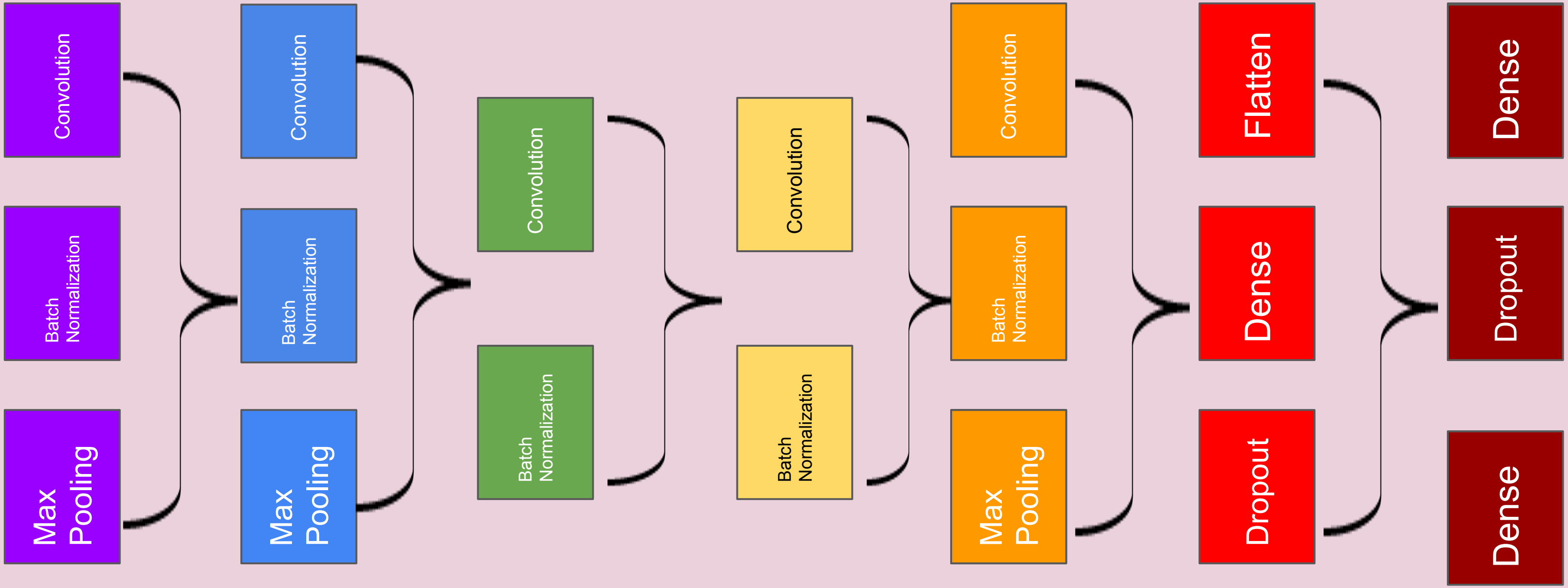
Troubleshoot

CNN Output

```
Epoch 72/150
8/8 [==============================] - 5s 496ms/step - loss: 1.4068 - accuracy: 0.5348 - val_loss: 1.0313 - val_accuracy: 0.5321
Epoch 73/150
8/8 [==============================] - 5s 577ms/step - loss: 1.4446 - accuracy: 0.5223 - val_loss: 1.0227 - val_accuracy: 0.5334
Epoch 74/150
8/8 [==============================] - 5s 595ms/step - loss: 1.3753 - accuracy: 0.5307 - val_loss: 1.0302 - val_accuracy: 0.5374
Epoch 75/150
8/8 [==============================] - 4s 492ms/step - loss: 1.3943 - accuracy: 0.5343 - val_loss: 1.0244 - val_accuracy: 0.5361
Epoch 76/150
8/8 [==============================] - 4s 488ms/step - loss: 1.3497 - accuracy: 0.5365 - val_loss: 1.0122 - val_accuracy: 0.5428
Epoch 77/150
8/8 [==============================] - 4s 492ms/step - loss: 1.3699 - accuracy: 0.5348 - val_loss: 0.9895 - val_accuracy: 0.5548
Epoch 78/150
8/8 [==============================] - 4s 490ms/step - loss: 1.4060 - accuracy: 0.5285 - val_loss: 1.0254 - val_accuracy: 0.5388
Epoch 79/150
8/8 [==============================] - 5s 493ms/step - loss: 1.3207 - accuracy: 0.5521 - val_loss: 1.0096 - val_accuracy: 0.5468
Epoch 80/150
8/8 [==============================] - 4s 491ms/step - loss: 1.3166 - accuracy: 0.5521 - val_loss: 0.9780 - val_accuracy: 0.5602
```



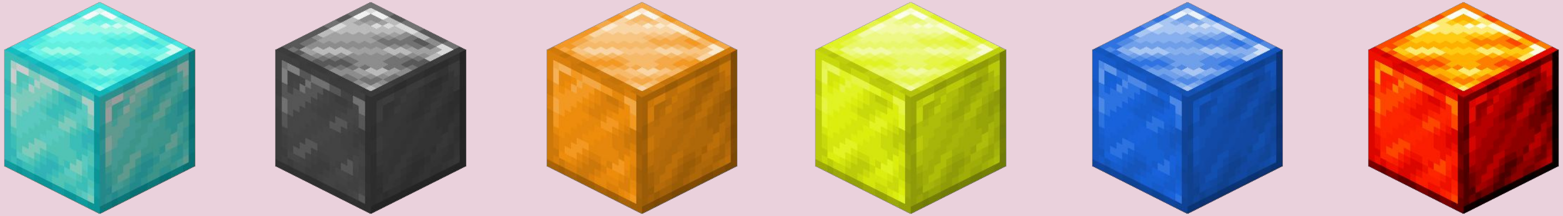
Extraterrestrial Confounding Cat Architecture



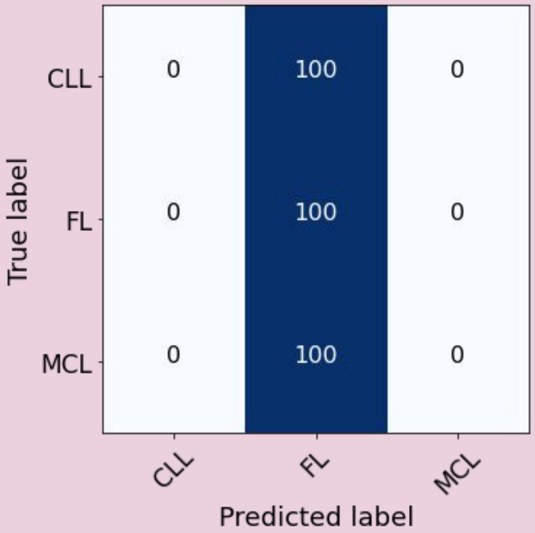
Results and Discussion

CNN Architectures

Convolution | Activation | Max Pooling | Average Pooling | Dense | Flatten



First CNN implementation: goes wrong



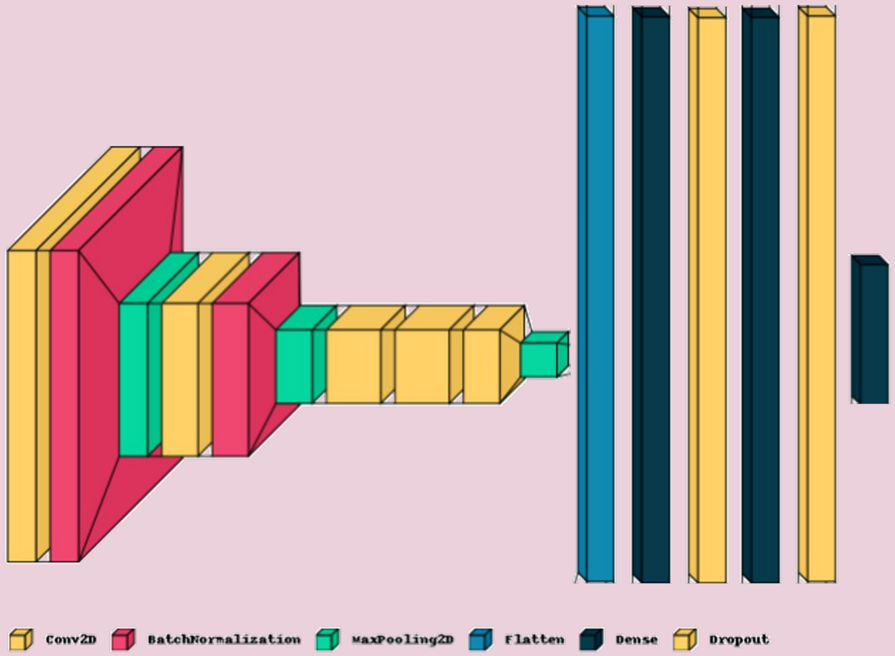
- Architecture adapted from morphological classification of SDSS galaxies activity

- A lot of changes, same performance

Solution: **AlexNet** CNN architecture

It consists of:

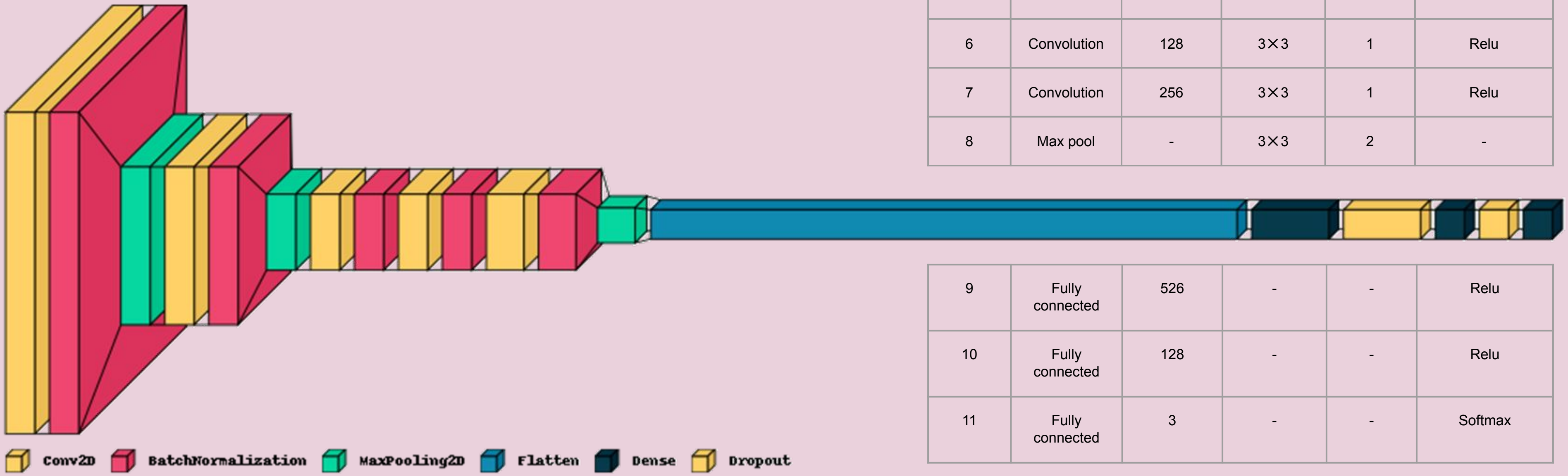
- 5 convolutional layers
- 3 max pooling layers
- 2 normalization leyers
- 2 fully connected layers
- 1 softmax layer



Extraterrestrial Confounding Cat Architecture

- Slightly different of the original AlexNet architecture

Layer	Type	Num Kernels	Kernel Size	Stride	Activation
0	Input	3	227 × 227	-	-
1	Convolution	64	11×11	4	Relu
2	Max pool	-	3×3	2	-
3	Convolution	128	5×5	1	Relu
4	Max pool	-	3×3	2	-
5	Convolution	128	3×3	1	Relu
6	Convolution	128	3×3	1	Relu
7	Convolution	256	3×3	1	Relu
8	Max pool	-	3×3	2	-
9	Fully connected	526	-	-	Relu
10	Fully connected	128	-	-	Relu
11	Fully connected	3	-	-	Softmax



Results and Discussion (continued)

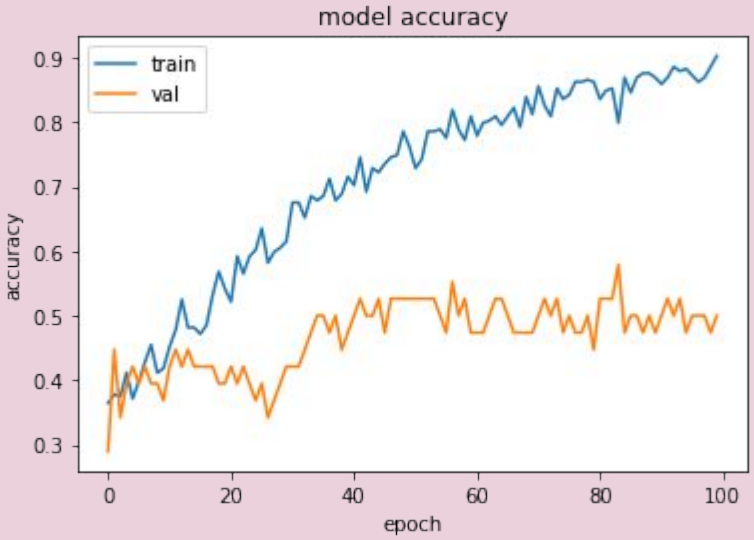
Learning curves - evaluate model performance

Confusion matrix - performance measurement for classification algorithms

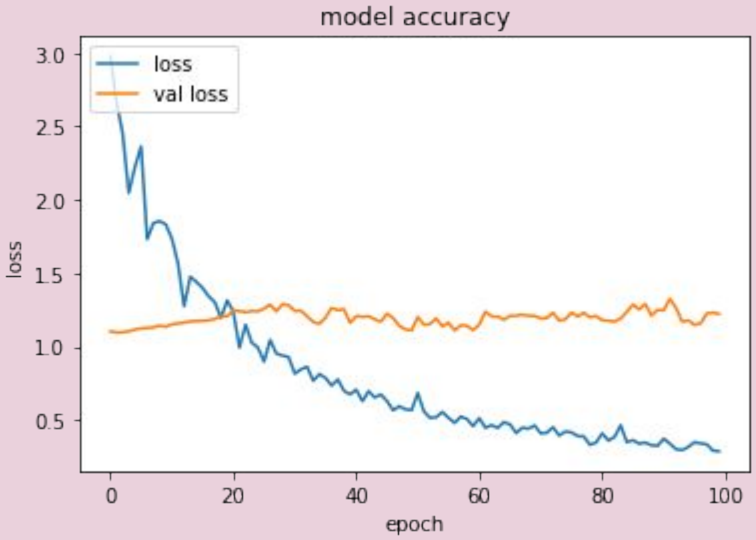
Simplest case: 1 patch per image

training	validation	test
299	37	38

- Training accuracy increases
- Validation accuracy constant



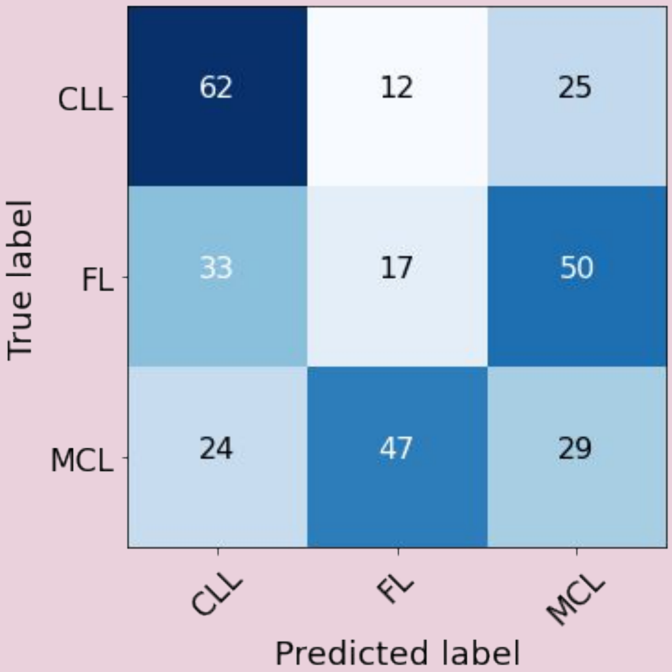
Accuracy curve over 100 epochs



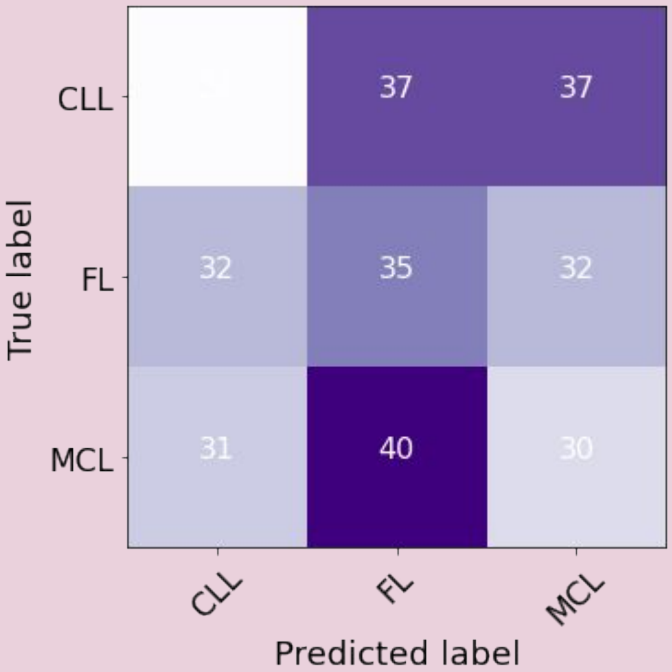
Loss curve over 100 epochs

Confusion matrices for test and training set

- Predictions for test set not very accurate

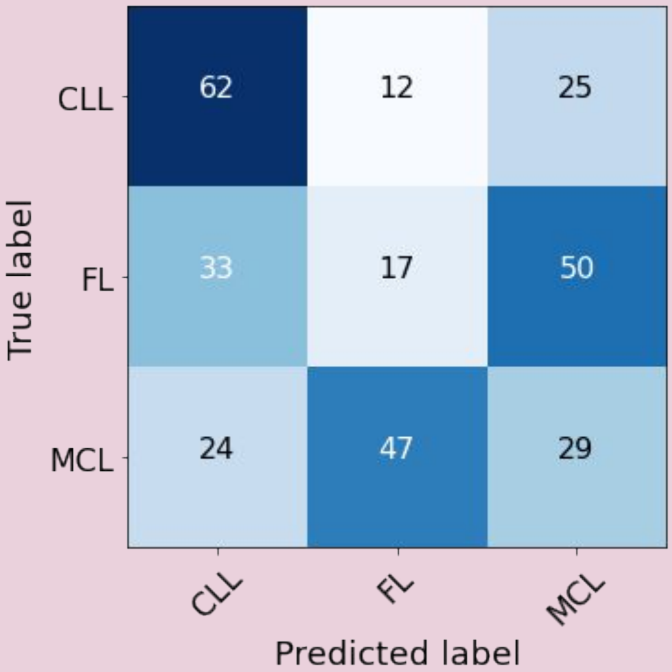


- Sanity check: predictions for training set

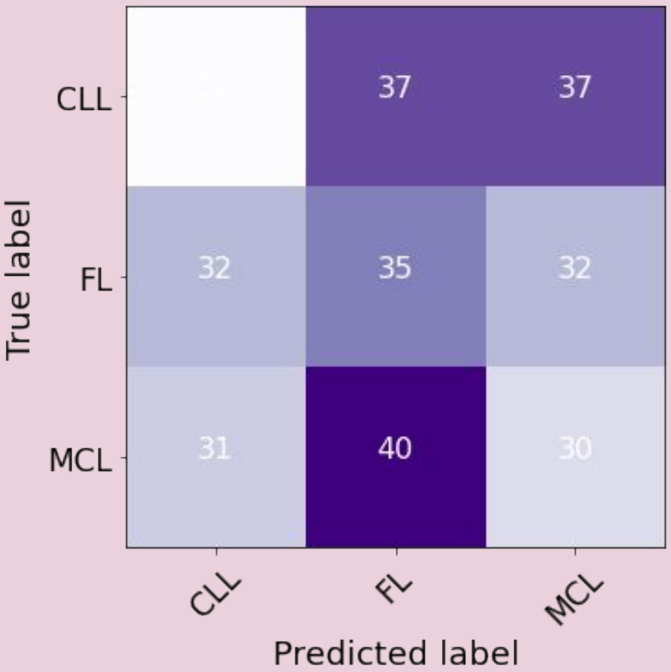


Is the model performance bad, or are other factors in play?

- Predictions for test set not very accurate



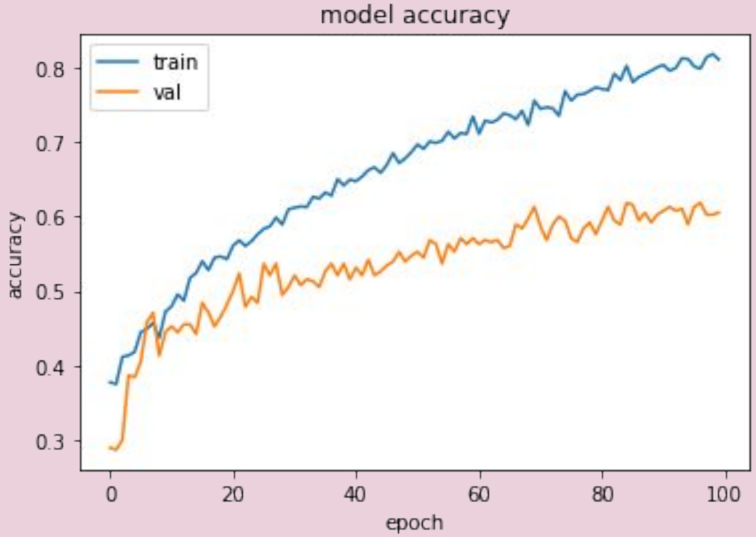
- Sanity check: predictions for training set



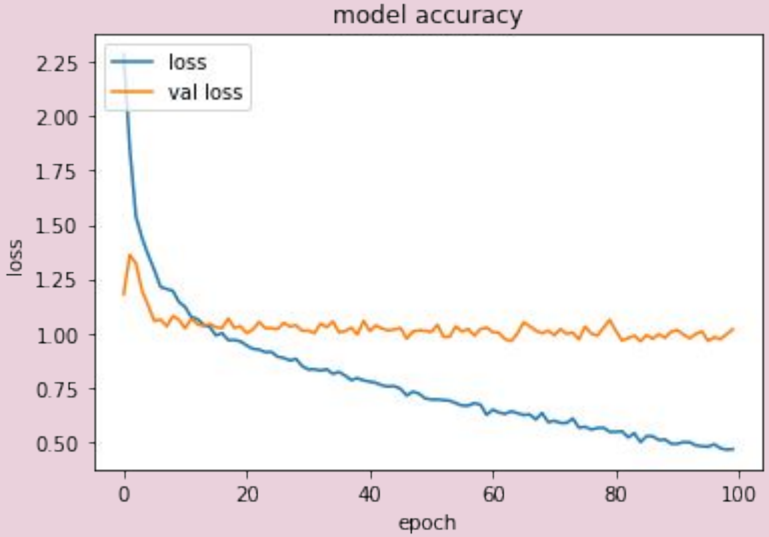
Increase sample size: 10 patch per image

training	validation	test
2990	370	380

Learning curves show indication of overfitting



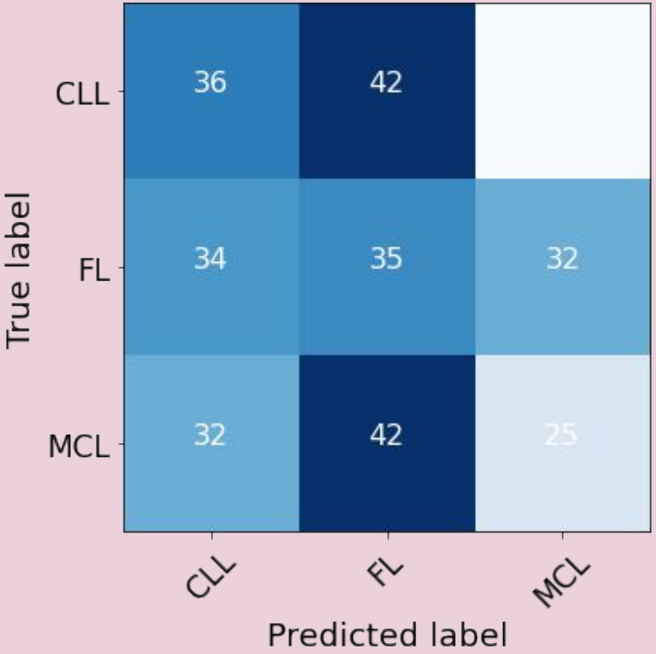
Accuracy curve over 100 epochs



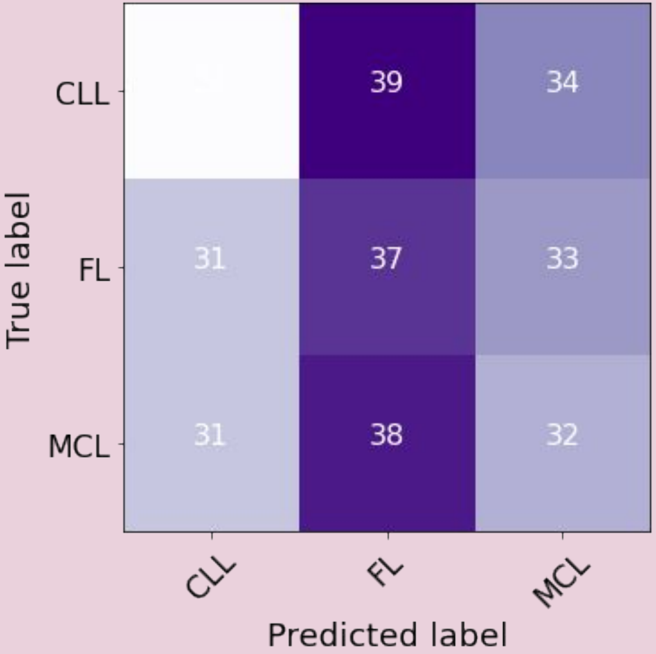
Loss curve over 100 epochs

Confusion matrices for test and training set

- Predictions for test set have not improved



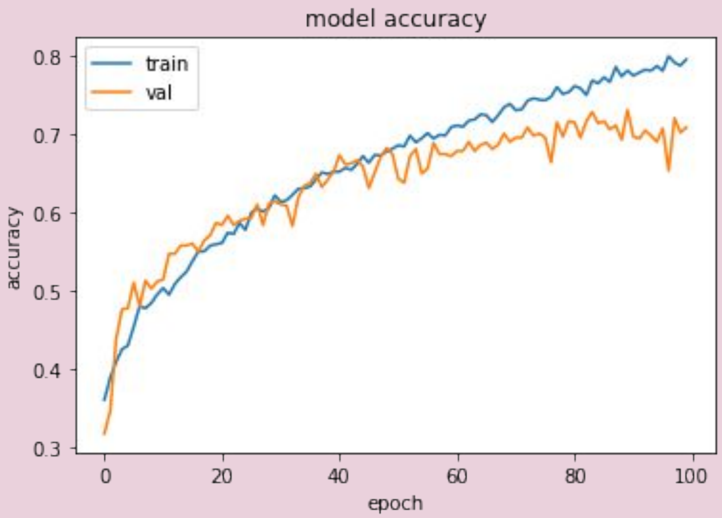
- Sanity check not so sane?



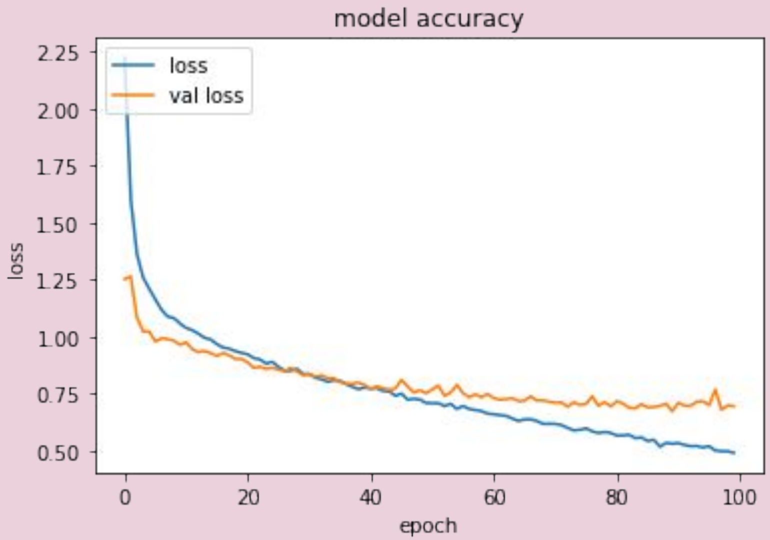
Increase sample size: 20 patch per image

training	validation	test
5980	740	760

- Validation accuracy curve shows slight improvement
- Training accuracy curve continues to rise



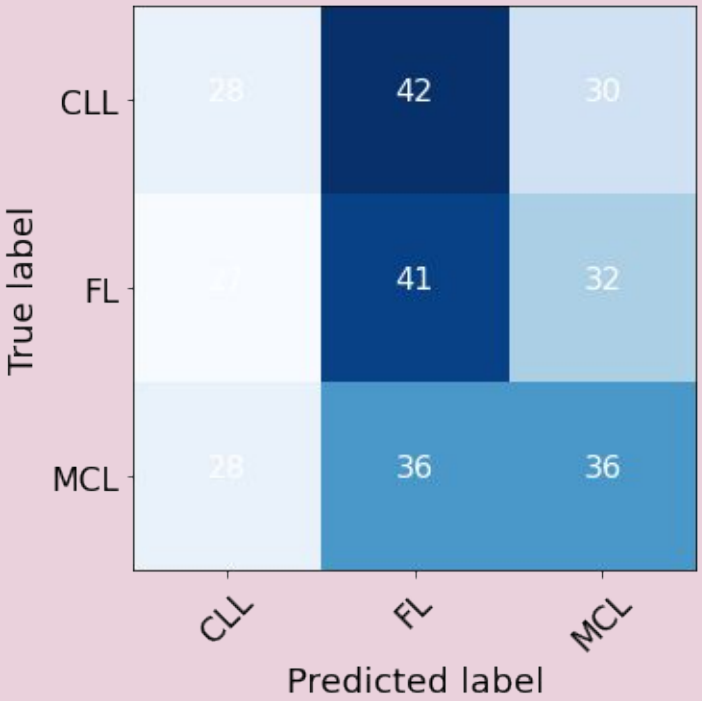
Accuracy curve over 100 epochs

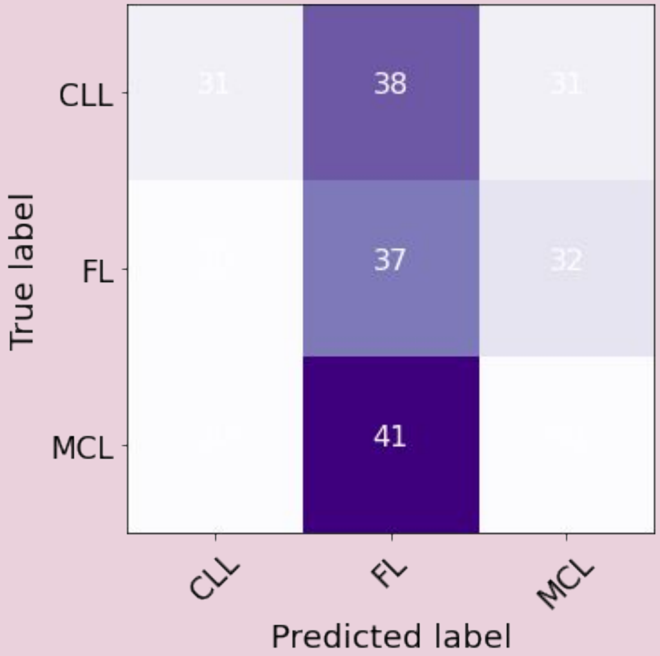


Loss curve over 100 epochs

Confusion matrices for test and training set

- Predictions for test set have not improved

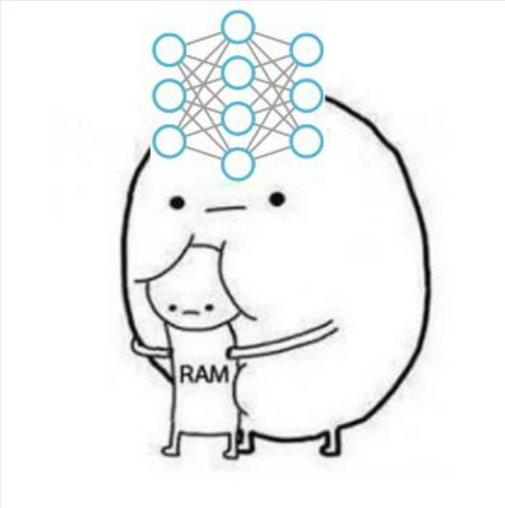




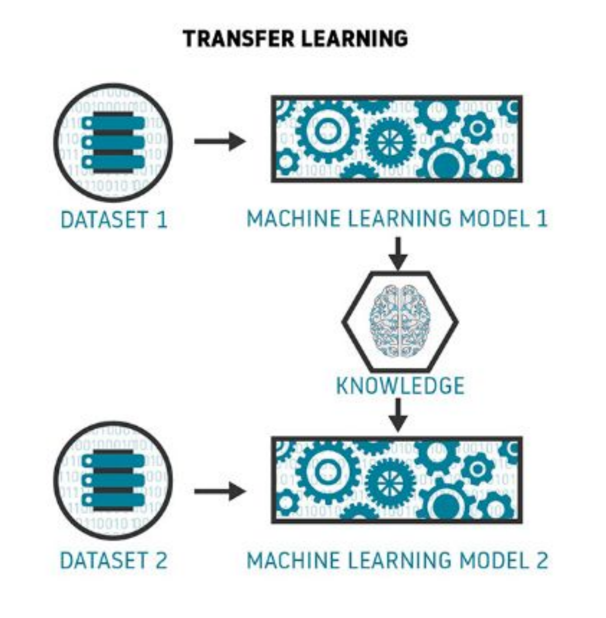
Tests with other parameters

- Increasing sample size by x20 → *no significant improvement*
- Decreasing the complexity of CNN → different architectures
- Dropout parameter → 0.1 - 0.5
- Image augmentation in training set → rotation - horizontal and vertical flipping - zooming
- Activation function, optimizer, kernels, learning rate
- Increasing batch size → *made things worse*

Optimization improvements for classification



- Better computing power - increase training sample without memory issues
- Implement AlexNet with transfer learning
- Segmentation/ feature extraction



Connections to Astronomy

The techniques implemented in this work can also be used in astronomy!

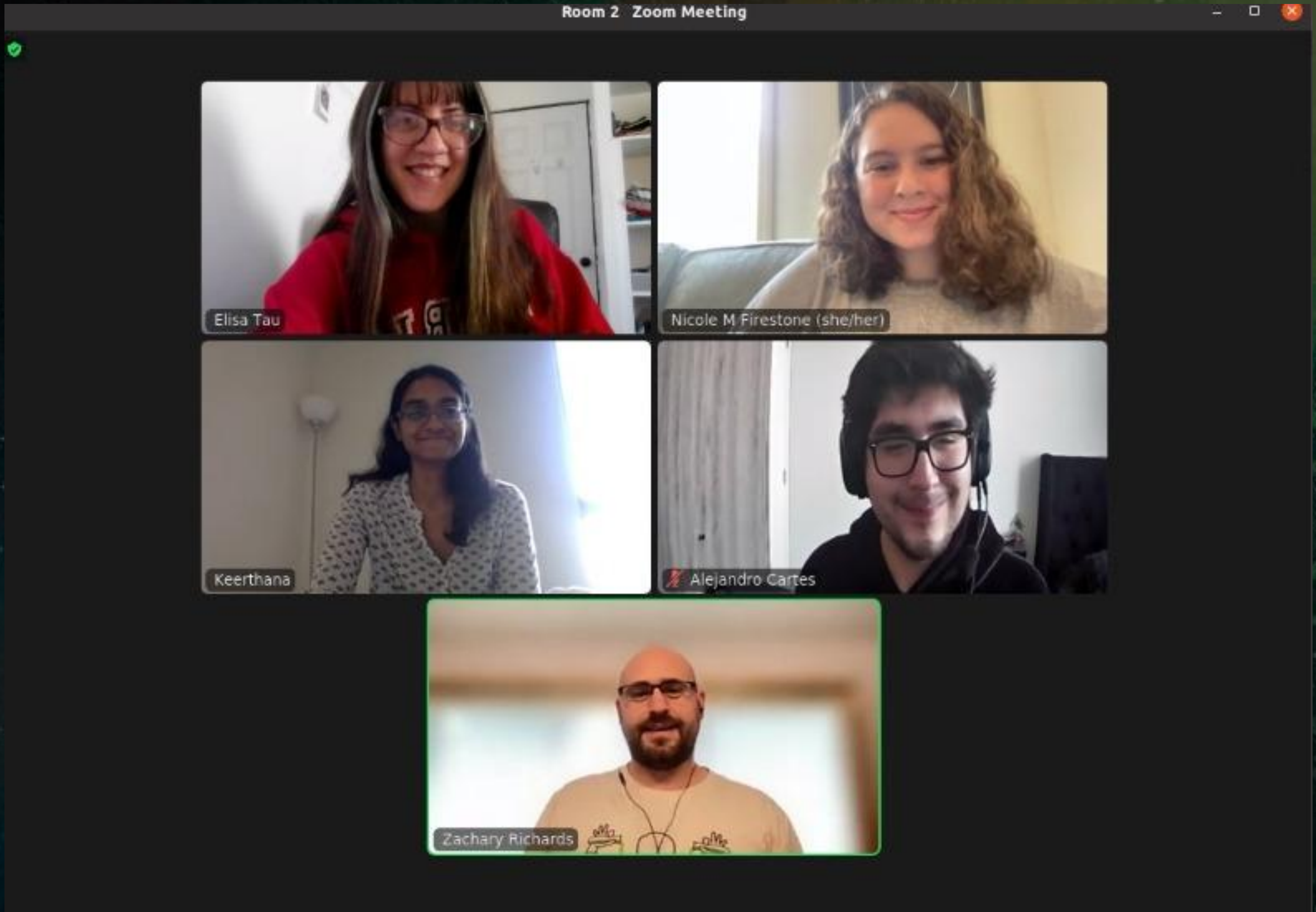
- CNNs: very useful for image processing; applicable for limiting the necessity of visual inspections in photometric works such as Lyman Alpha studies.
- Data augmentation: useful for those cases in which we do not have many observations (example: rotating galaxies with particular features that we want to study).
- Nowadays surveys can serve as training samples.
- Catalogs can be used as seeds for transfer learning techniques (e.g. JWST)

...and to other fields!

Summary

- Created CNN using TensorFlow and Keras in order to classify 3 different types of Lymphoma
- Improved results significantly from initial methods, but ultimately could achieve better results with less limitations in RAM related to the homogeneity in images
- Gained an exciting experience of working out of area
- Learned many valuable methods in deep learning, data augmentation, and image processing, which we hope to apply to our research in astronomy

Thank you for your attention!



References

J Pathol Inform

Original Article

Deep learning for digital pathology image analysis: A comprehensive tutorial with selected use cases

Andrew Janowczyk[1], Anant Madabhushi[1]

[1]Department of Biomedical Engineering, Case Western Reserve University, Cleveland, OH 44106, USA

E-mail: *Dr. Andrew Janowczyk - andrew.janowczyk@case.edu
*Corresponding author

Received: 18 November 2015 Accepted: 18 March 2016 Published: 26 Jul 2016

Abstract

Background: Deep learning (DL) is a representation learning approach ideally suited for image analysis challenges in digital pathology (DP). The variety of image analysis tasks in the context of DP includes detection and counting (e.g., mitotic events),

Histopathology

Histopathology 2018, 72, 227–238. DOI: 10.1111/his.13333

HER2 challenge contest: a detailed assessment of automated HER2 scoring algorithms in whole slide images of breast cancer tissues

Talha Qaiser,[1,*] Abhik Mukherjee,[2,*] Chaitanya Reddy PB,[3] Sai D Munugoti,[3] Vamsi Tallam,[3] Tomi Pitkäaho,[4] Taina Lehtimäki,[4] Thomas Naughton,[4] Matt Berseth,[5] Aníbal Pedraza,[6] Ramakrishnan Mukundan,[7] Matthew Smith,[8] Abhir Bhalerao,[1] Erik Rodner,[9] Marcel Simon,[9] Joachim Denzler,[9] Chao-Hui Huang,[10,11] Gloria Bueno,[6] David Snead,[12] Ian O Ellis,[2] Mohammad Ilyas[2,13] & Nasir Rajpoot[1,12]

[1]Department of Computer Science, University of Warwick, Coventry, UK, [2]Department of Histopathology, Division of Cancer and Stem Cells, School of Medicine, University of Nottingham, Nottingham, UK, [3]Department of Electronics and Electrical Engineering, Indian Institute of Technology, Guwahati, India, [4]Department of Computer Science, Maynooth

NOTES

Started to segmentation
Slide of failed approaches
Explain Alexnet and our architecture
Plots
Talk about one hot encoder
Image normalization
What did work
Diagram of steps in analysis - arrays, preprocessing, training etc (Flow chart)
Why using neural networks
Future works - application to astronomy