# Detection of Acute Lymphocytic Leukemia (ALL) with Convolutional Neural Network

Unnithan, Ragi| Dr. Liu, Xinlian | M.S. Computer Science | Hood College | Frederick, MD, USA | ru3@hood.edu

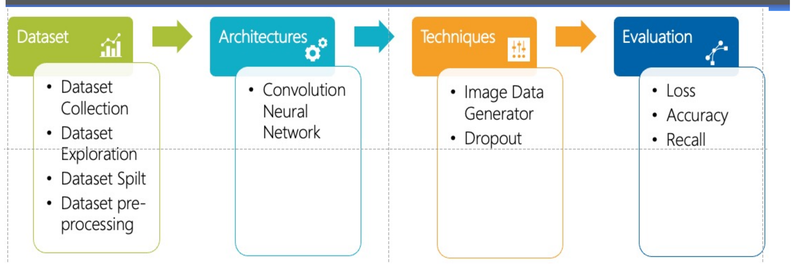
## ABSTRACT

Acute Lymphocytic Leukemia (ALL) is a deadly cancer that not only affects adults but also accounts for 25% of childhood cancers. Timely and accurate diagnosis of the cancer is an important factor for an effective treatment to improve survival rate. The image of ALL cells under the microscope is very similar in morphology to that of the normal cells. In this work, we are trying to answer whether a Convolutional Neural Network (CNN) can reliably classify cancer and normal cell images to assist in the diagnosis of ALL. This project aims to build a CNN model that can classify ALL from normal cells as diagnosing ALL is difficult even for trained medical operator.

## INTRODUCTION

Acute Lymphocytic Leukemia (ALL) is a type of cancer of the blood and bone marrow. ALL is the most common cause of pediatric cancer and the most frequent cause of death from cancer before 20 years of age. Cell classification usually depends on morphological characteristics of the cell and requires a skilled medical operator. These procedures are time consuming, tedious, costly and error prone. In this project, we are trying to see whether a **Convolutional Neural Network can classify Normal and ALL cells** from microscopic blood images

## IMPLEMENTATION

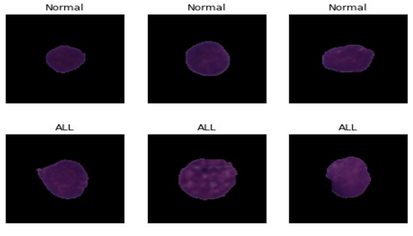



## DATA OVERVIEW

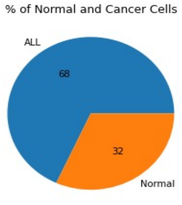
Obtained from Kaggle – C_NMC_Leukemia. Images are segmented from microscopic images. Total of 15,135 images from 118 patients with 2 labels.

| Cell Type | Train | Validation | Test |
|---|---|---|---|
| ALL | 47 | 13 | 9 |
| Normal | 26 | 15 | 8 |
| Total | 73 | 28 | 17 |

| Dataset - subtype | Total | Normal | ALL |
|---|---|---|---|
| Training | 7996 | 2542 | 5454 |
| Validation | 2665 | 847 | 1818 |
| Test | 1867 | 1219 | 648 |



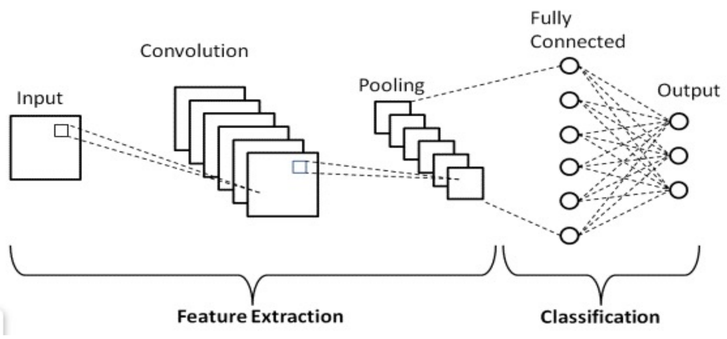

Normal cells– Spherical and non- clefted



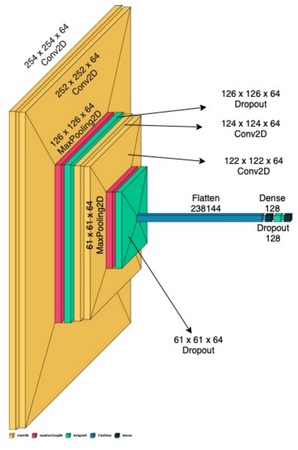

Cancer cells – Non-spherical and clefted

## ARCHITECTURES

Convolutional Neural Network



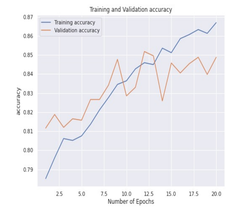

- Three common layers are used, convolution, pooling, and fully connected
- Input layers accepts 256x256x3 input (RGB images)
- Four convolution layers, with 64 filters of 3 X3.
- All convolution layers use *relu* activation function and 3x3 stride.
- The max pooling layer uses 2x2 pooling window.
- Three Dropout layers with a value of 0.5
- Flatten layer
- Two Dense layer




```
Model: "sequential_12"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_41 (Conv2D)           (None, 254, 254, 64)      1792
_________________________________________________________________
conv2d_42 (Conv2D)           (None, 252, 252, 64)      36928
_________________________________________________________________
max_pooling2d_32 (MaxPooling (None, 126, 126, 64)      0
_________________________________________________________________
dropout_44 (Dropout)         (None, 126, 126, 64)      0
_________________________________________________________________
conv2d_43 (Conv2D)           (None, 124, 124, 64)      36928
_________________________________________________________________
conv2d_44 (Conv2D)           (None, 122, 122, 64)      36928
_________________________________________________________________
max_pooling2d_33 (MaxPooling (None, 61, 61, 64)        0
_________________________________________________________________
dropout_45 (Dropout)         (None, 61, 61, 64)        0
_________________________________________________________________
flatten_12 (Flatten)         (None, 238144)            0
_________________________________________________________________
dense_24 (Dense)             (None, 128)               30482560
_________________________________________________________________
dropout_46 (Dropout)         (None, 128)               0
_________________________________________________________________
dense_25 (Dense)             (None, 1)                 129
=================================================================
Total params: 30,595,265
Trainable params: 30,595,265
Non-trainable params: 0
```

## EVALUATION


- With 100 epochs
  - Training accuracy – 94%
  - Validation accuracy – 85%
  - Testing accuracy – 61%




| Model | Optimizer | Training | | | Validation | | | Test | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Accuracy | Recall | Loss | Accuracy | Recall | Loss | Accuracy | Recall | Loss |
| 2X2C1D - Dropout(0.5) | RMSprop | 0.8112 | 0.598 | 0.446 | 0.83 | 0.621 | 0.418 | 0.617 | 0.108 | 0.8515 |

## CONCLUSION

CNN can be used for classification of ALL cells from normal cells and can be used as a tool by a physician for verifying his diagnosis.

performance of the model can be improved by implementing different methods. In this project, we were able to improve the performance of the model by increasing the number of epochs to 100. We also tried two different optimizers – RMSprop and Adam optimizer. But we see increase in performance when using RMSprop. Batch Normalization and Dropout were also implemented to improve the performance. We experimented with different dropout values ranging from 0.1 to 0.5 and we settled on 0.5 as this increased the accuracy.

## FUTURE WORK

In future we can implement early stopping and cross-validation for avoiding overfitting. Image data augmentation can be implemented to the normal dataset for the eliminating class im-balance. Try implementing Alexnet or other pre-trained models to improve the accuracy

## ACKNOWLEDGEMENT

I would like to express sincere thank to Professor Dr. Xinlian Liu for this course and guidance throughout the semester as it helped me in learning various unknown topics related to deep learning, image classification.

## REFERENCE

1. https://www.hindawi.com/journals/complexity/2021/6658192/
2. https://www.hindawi.com/journals/cin/2021/7529893/
3. https://www.upgrad.com/blog/basic-cnn-architecture/
4. https://towardsdatascience.com/detecting-leukemia-with-a-cnn-af699b19ab99