CS 529: Project 4: Kaggle Competition: Human Protein Atlas Image Classification

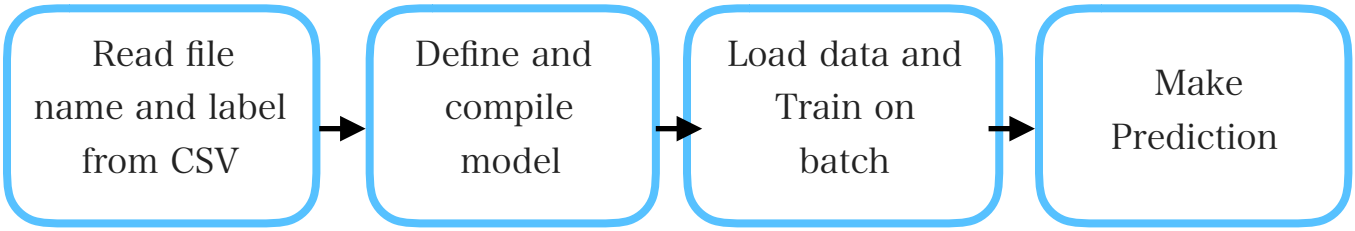
Farhan Asif Chowdhury
Md Amanul Hasan

Summary: In this project, we have tackled the problem of the "multiclass-multilabel" classification task on Human Protein Images. We have used ResNet34 Architecture for the Convolutional Neural Network (CNN) layers with the weights initialized from pertained ImageNet datasets. We have dropped the top fully connected layers from the ResNet34 architecture and modified the top layers to address multilabel classification task. We used newly introduced “Focal Loss” for loss calculation. Our top score in the public leaderboard is 45.3%.

Method:

Some of the major challenges were: (1) The multi-labels were not given to any particular segment of the image rather given to the image as a whole (2) High amount of class Imbalance in the training data; (3) Limited training sample data As Convolutional Neural Network (CNN) is currently the state of the art deep learning methods for Computer Vision and Image Processing, it was our first choice. We explored some of the kernels publicly available related to this kaggle competition to get some idea. We found few CNN architectures [1][2] which gave moderate accuracy (in the range of 30% in Public Leaderboard). But, one kernel that used ResNet34 architecture [3], achieved more than 46% accuracy. So, we tried to take ideas from their model to build on.

As our task is the multi-label prediction, primarily we used binary cross entropy for loss calculation, which is the most common which resulted in good enough accuracy. But, after doing some research, we found that recently "Focal loss for dense object detection" [4][5] as loss parameter has given better performance in similar multi-label classification task, which we observed in our own implementation and later used this for our loss calculation. It performed better as it can handle class imbalance in the training data efficiently.



In the final prediction layer we used sigmoid on each of the 28 nodes to predict probability for each class, and then we used thresholding value to convert that probability to 0 (present) and 1 (absent). We tried several thresholding combination for each class to maximize F1-Score [6] on the validation data set.

To prevent loading large-scale training data at a time in memory, we used batch data-generator to load data batch by batch during training. During the training period, we evaluated the model on validation data after a number of training iteration [7]. We used a batch size of 32, with image resized to 256 by 256 pixel to meet the memory constraint. We believe that, with more powerful computing power, with higher resolution, the accuracy can be increased.

We used Keras with tensorflow in the backend for implementing and running our deep learning model. We used Nvidia GTX-1070 8 GB GPU for our training purpose.

References:

[1] https://www.kaggle.com/rejpalcz/cnn-128x128x4-keras-from-scratch-lb-0-328
[2] https://www.kaggle.com/allunia/protein-atlas-exploration-and-baseline
[3] https://www.kaggle.com/aumyfarhan/pretrained-resnet34-with-rgby-0-460-public-3bdcbf/notebook
[4] Lin, Tsung-Yi, et al. "Focal loss for dense object detection." IEEE transactions on pattern analysis and machine intelligence (2018).
[5] https://github.com/mkocabas/focal-loss-keras
[6] https://www.kaggle.com/guglielmocamporese/macro-f1-score-keras
[7] https://stanford.edu/~shervine/blog/keras-how-to-generate-data-on-the-fly