Benchmark for Modified UNet Models with Medical Images - A Case Study with Eosinophilic Esophagitis (EoE) Images and Future Research Opportunities

Ian Liu, University of Virginia

Abstract

This report studies the efficiency of several modified UNet models in processing and handling biopsy images of Eosinophilic Esophagitis (EoE), which is a chronic disease characterized by the prevalence of a type of white blood cell (eosinophil) in the esophagus . Modifications to the original UNet Model are made to attempt to improve performance as well as decrease the training time. Cutting-edge convolutional mechanisms such as Dilated Convolutional Layers and Separable Convolutional Layers are implemented. To further investigate the issue of efficiency on a hardware level, a series of experiments are conducted on a variety of GPUs in order to obtain the benchmark comparison. Further possible research opportunities are proposed and discussed.

Before Getting Started(Reproducibility Notes for Team 3's Code)

The code is well organized and documented. Despite some functions are hard-coded, the program is overall pretty intuitive to run. Here are some key components or parameters that need to adjusted in order to get the experiments running:

- Environments
    - Tensorflow 2.8.0/Keras Py3.9
    - Rivanna A100 and V100 GPUs
    - DS--6013 allocation
    - RAM: 150Gb
- Important functions/lines of code
    - def createModel(input_tensor, bnorm_axis, n_filters, drop_rate, drop_train):
    - def createDilatedModel(input_tensor, bnorm_axis, n_filters, drop_rate, drop_train, dilation):
    - def createPyramidDilatedModel(input_tensor, bnorm_axis, n_filters, drop_rate, drop_train):
    - def createSeparableModel(input_tensor, bnorm_axis, n_filters, drop_rate, drop_train):

        - def createDepthwiseModel(input_tensor, bnorm_axis, n_filters, drop_rate, drop_train):
- Which model to run? (comment out the model you want to run) **- Depthwise is not working**
    - model=createSeparableModel(input_tensor, bnorm_axis, n_filters, drop_rate, drop_train)
    - model=createModel(input_tensor, bnorm_axis, n_filters, drop_rate, drop_train)
    - model=reateDilatedModel(input_tensor, bnorm_axis, n_filters, drop_rate, drop_train, dilation)
    - model= createPyramidDilatedModel(input_tensor, bnorm_axis, n_filters, drop_rate, drop_train)
    - model=createDepthwiseModel(input_tensor, bnorm_axis, n_filters, drop_rate, drop_train)
- Important (hyper)parameters
    - epoch = 100
    - lr = N/A
    - drop_rate = 0.5
    - Batch_size = 4 and 5 (unclear comments)

Introduction

Eosinophilic esophagitis (EoE) is a chronic disease characterized by the prevalence of a type of white blood cell (eosinophil) in the esophagus. EoE affects approximately 5 to 10 people in every 10,000 people and can be seen in 2-7% of patients that undergo endoscopy for any reason. Identifying EoE has been heavily dependent on experienced and knowledgeable professionals, which is not available

The U-Net is convolutional network architecture for fast and precise segmentation of images. It has been shown to outperform what was previously considered the best method (a sliding-window convolutional network) on the ISBI challenge for segmentation of neuronal structures in electron microscopic stacks. For the UNet model, we used 23 convolutional layers with batch normalization. In both the encoder and decoder steps, we used a ReLU activation function, and for the final layer we used a sigmoid activation function. Loss was computed using binary cross entropy and ADAM was used as the optimizer. To prevent overfitting, we implemented early stopping and data augmentation.

When performing image segmentation tasks in medical imaging, one of the biggest challenges is not only to achieve high accuracy but also obtaining faster inference. This is often achieved by lighting the model by the number of layers or parameters. The previously discussed U-Net architecture is comprised of many 2D convolutional layers, where a 2D convolution is applied over an input signal that is composed of many input planes. Current research has identified many other convolution mechanisms that can improve performance as well as decrease training time in U-Nets. Having said that, the following modifications are proposed to achieve faster training time:

- Dilated Convolutional Layers with Dilation Factor of 2, 4, 8, 16 and Pyramid
- Spatially separable convolutional layer

- Depth-wise convolutional layer

Some preliminary experiments have been conducted using A100 GPUs, however, the similar experiments haven't been done on the other types of GPUs to obtain the benchmarking results.

Methodology

An experiment plan is listed in the following tables. Due to the lack of useful comments and explanations in the code this group submitted, I wasn't able to tune the important hyperparameters and run the experiments under diverse settings. Thus running the models on different GPUs is the best I could do:

GPU	Original UNet Model	UNet with Dilation Factor of 2	UNet with Dilation Factor of 4	UNet with Dilation Factor of 8	UNet with Dilation Factor of 16	UNet with Dilation Factor of Pyramid	UNet with Spatially Separable Layer	UNet with Depth-Wise
A100	138.98	141.28	140.59	141.11	142.40	141.482	322.190	N/A
V100	169.23	180.74	175.17	174.934	181.436	175.31	375.75	N/A
P100	Err:OOM	Err:OOM	Err:OOM	Err:OOM	Err:OOM	Err:OOM	Err:OOM	N/A
K80	Err:OOM	Err:OOM	Err:OOM	Err:OOM	Err:OOM	Err:OOM	Err:OOM	N/A

Table 1: Benchmark Results in terms of Training Time(in s)

GPU	Original UNet Model	UNet with Dilation Factor of 2	UNet with Dilation Factor of 4	UNet with Dilation Factor of 8	UNet with Dilation Factor of 16	UNet with Dilation Factor of Pyramid	UNet with Spatially Separable Layer	UNet with Depth -Wise
A100	0.673	0.560	0.434	0.423	0.211	0.535	0.534	N/A
V100	0.582	0.492	0.565	0.412	0.186	0.399	0.598	N/A
P100	Err:OOM	Err:OOM	Err:OOM	Err:OOM	Err:OOM	Err:OOM	Err:OOM	N/A
K80	Err:OOM	Err:OOM	Err:OOM	Err:OOM	Err:OOM	Err:OOM	Err:OOM	N/A

Table 2: Benchmark Plan in terms of Median DICE Score

GPU	Original UNet Model	UNet with Dilation Factor of 2	UNet with Dilation Factor of 4	UNet with Dilation Factor of 8	UNet with Dilation Factor of 16	UNet with Dilation Factor of Pyramid	UNet with Spatially Separable Layer	UNet with Depth-Wise
A100	2.472	2.503	2.570	2.538	2.685	2.426	2.430	N/A
V100	3.345	3.364	3.360	3.462	3.432	3.251	3.473	N/A
P100	Err:OOM	Err:OOM	Err:OOM	Err:OOM	Err:OOM	Err:OOM	Err:OOM	N/A
K80	Err:OOM	Err:OOM	Err:OOM	Err:OOM	Err:OOM	Err:OOM	Err:OOM	N/A

Table 3: Benchmark Plan in terms of Data_Loading Time

GPU	Original UNet Model	UNet with Dilation Factor of 2	UNet with Dilation Factor of 4	UNet with Dilation Factor of 8	UNet with Dilation Factor of 16	UNet with Dilation Factor of Pyramid	UNet with Spatially Separable Layer	UNet with Depth-Wise
A100	43.28	54.506	55.289	57.103	58.318	54.315	43.457	N/A
V100	74.10	90.010	90.535	92.607	96.427	89.283	73.962	N/A
P100	Err:OOM	Err:OOM	Err:OOM	Err:OOM	Err:OOM	Err:OOM	Err:OOM	N/A
K80	Err:OOM	Err:OOM	Err:OOM	Err:OOM	Err:OOM	Err:OOM	Err:OOM	N/A

Table 4: Benchmark Plan in terms of T Model Loading Time(T=20)

Explanations Regarding the Errors and Unreproducible Models

From the charts above, you should've noticed that the models were not successfully reproduced on P100 and K80 GPUs due to the error message of OOM, which indicates that the GPU resources were either exhausted or ran out of memory.

I've attempted to resolve the issues by adjusting the essential hyperparameters such as batch size. but

Results

According to the preliminary results, we state that making modifications to the convolutional mechanisms don't help with significantly decreasing the training time. The new performance after the adjustment doesn't show significant improvement. More investigation is needed to prove the correctness of the direction.

However, from the benchmark results, we can see that xxx GPU does perform the best among other GPUs in terms of having the shortest training time. This is due to the following reasons.

The metric is not a universally accepted metric, which made the benchmark or results comparison particularly problematic.

Future Work: Federated Learning and Data Privacy

UNET or the modified UNET models may not be the best candidates that will produce the most satisfactory results. Due the fact the we are dealing the biomedical images where the target is not only detect and classify an infection, but also to identify the area infected by the disease, more models can be implemented to complete the task.

UNet is a convolutional neural network architecture that expanded with few changes in the CNN architecture. It was invented **to deal with biomedical images where the target is not only to classify whether there is an infection or not but also to identify the area of infection**.