Name: Mayank Sahu

Email address: sahu.mayank0101@gmail.com

Contact number: 9424995835

Anydesk address:

Years of Work Experience: **0**

Date: **18th Jun 2021**

Self Case Study -2: Medical Image Segmentation

"After you have completed the document, please submit it in the classroom in the pdf format."

Please check this video before you get started:
https://www.youtube.com/watch?time_continue=1&v=LBGU1_JO3kg

Overview

*** Write an overview of the case study that you are working on. ***(MINIMUM 200 words)***
***

Computer vision is a field of study focused on the problem of helping computers to see. It is a multidisciplinary field that could broadly be called a subfield of artificial intelligence and machine learning, which may involve the use of specialized methods and make use of general learning algorithms. The goal of computer vision is to understand the content of digital images. Typically, this involves developing methods that attempt to reproduce the capability of human vision. Understanding the content of digital images may involve extracting a description from the image, which may be an object, a text description, a three-dimensional model, and so on. Many popular computer vision applications involve trying to recognize things in photographs; for example:

- **Object Classification**: What broad category of object is in this photograph?
- **Object Identification**: Which type of a given object is in this photograph?
- **Object Verification**: Is the object in the photograph?
- **Object Detection**: Where are the objects in the photograph?

- **Object Landmark Detection**: What are the key points for the object in the photograph?
- **Object Segmentation**: What pixels belong to the object in the image?
- **Object Recognition**: What objects are in this photograph and where are they?

In digital image processing and computer vision, image segmentation is the process of partitioning a digital image into multiple segments (sets of pixels, also known as image objects). The goal of segmentation is to simplify and/or change the representation of an image into something that is more meaningful and easier to analyse. Image segmentation is typically used to locate objects and boundaries (lines, curves, etc.) in images. More precisely, image segmentation is the process of assigning a label to every pixel in an image such that pixels with the same label share certain characteristics. There are two types of segmentation- semantic segmentation and instance segmentation. If there are 5 people in an image, semantic segmentation will focus on classifying all the people as a single instance. Instance segmentation, on the other hand. will identify each of these people individually.

Medical image segmentation is the task of segmenting objects of interest in a medical image. Image segmentation is considered the most essential medical imaging process as it extracts the region of interest (ROI) through a semiautomatic or automatic process. It divides an image into areas based on a specified description, such as segmenting body organs/tissues in the medical applications for border detection, tumour detection/segmentation, and mass detection. Because segmentation partitions the image into coherent regions, clustering procedures can be applied for segmentation by extracting the global characteristics of the image to professionally separate the ROI from the background.

Research-Papers/Solutions/Architectures/Kernels

*** Mention the urls of existing research-papers/solutions/kernels on your problem statement and in your own words write a detailed summary for each one of them. If needed you can include images or explain with your own diagrams. ***

1. Medical Transformer: Gated Axial – Attention for Medical Image Segmentation (url: https://arxiv.org/abs/2102.10662)

In this paper authors proposed a gated position-sensitive axial attention mechanism that works well even on smaller datasets, also introduced Local-Global (LoGo) training methodology for transformers which is effective, and further proposed Medical-Transformer (MedT) which is built based on the above two concepts proposed specifically for medical image segmentation. In the proposed architecture Medical Transformer (MedT) uses gated axial attention layer as the basic building block and uses LoGo strategy for training. MedT has two branches - a global branch and local branch. The input to both of these branches are the feature maps extracted from an initial conv block. This block has 3 conv layers, each followed by a batch normalization and ReLU activation. In the encoder of both branches, they used proposed transformer layer while in the decoder, they used a conv block. The encoder bottleneck contains a 1X1 conv layer followed by normalization and two layers of multi-head attention layers where one operates along height axis and the other along width axis. Each multi-head attention block is made up of the proposed gated axial attention layer. Note that each multi-head attention block has 8 gated axial attention heads. The output from the multi-head attention blocks are concatenated and passed through another 1X1 conv which are added to residual input maps to produce the output attention maps. In each decoder block, they used conv layer followed by an upsampling layer and ReLU activation. They also implemented skip connections between each encoder and decoder blocks in both the branches. In the global branch of MedT, there are 2 blocks of encoder and 2 blocks of decoder. In the local branch, there are 5 blocks of encoder and 5 blocks of decoder. They used three datasets: Brain Anatomy segmentation, Gland Segmentation and MoNuSeg. Binary cross entropy was used as loss function and Adam as optimizer with learning rate of 0.001. Model was trained for 400 epochs with 4 images as batch size, also the gates were not trained for first 10 epochs. F1 score and IoU score were used as metrics for this work. Their best score for each dataset were: Brain Anatomy- 88.84 and 81.34, Gland Segmentation- 81.02 and 69.61 and MoNuSeg- 79.55 and 66.17 as F1 and IoU score respectively.

2. AG-CUResNeSt: A Novel Method for Colon Polyp Segmentation (url: https://arxiv.org/abs/2105.00402)

In this paper authors proposed a novel deep network, called AG-CUResNeSt, where the encoders of coupled UNets are strengthened by residual connections and split-attention blocks in ResNeSt. Attention gates are integrated into skip connections within each UNet to suppress the redundant low-level information from the encoders. They also leveraged skip connections across the two UNets to reduce gradient vanishing and promote feature reuse. Their proposed network consists of two coupled UNets with a similar architecture. Each UNet has an encoder and a decoder with skip connections between them. The encoder takes an image input of size 512 X 512, then passes through five top-down blocks to produce a high-level semantic feature map of size 16 X 16. This feature map is gradually up-sampled through five bottom-up blocks of the decoder and fused with low-level information in the encoder via gated skip connections called attention gates. Next, they used a 1 X 1 conv layer followed by a sigmoid layer at the end of each UNet to yield its output. The last feature map of the first UNet is combined with the raw input image and then fed to the second UNet for further refinement. They also used skip connections across the two UNets to enhance the information ow and promote feature reuse in the network. Authors used Tversky loss (alpha=0.5 and beta=0.7) as loss function. They used four datasets: CVC-ClinicDB, ETIS-Larib, Kvasir-SEG and CVC-ColonDB. SGD optimizer was used for optimization with learning rate as 5*10-3 and momentum as 0.9. Authors also performed data augmentation like rotation(90,180,270), horizontal and vertical flip, resizing, blurring, random brightness and random contrast. Recall, Precision, Dice and IoU were used as performance metrics. They performed various experiments with different combinations of the datasets.

3. FANet: A Feedback Attention Network for Improved Biomedical Image Segmentation(url: https://arxiv.org/abs/2103.17235)

In this work, authors leveraged the information of each training epoch to prune the prediction maps of the subsequent epochs. They proposed a novel architecture called feedback attention network (FANet) that unifies the previous epoch mask with the feature map of the current training epoch. The previous epoch mask is then used to provide a hard attention to the learnt feature maps at different convolutional layers. The network also allows to rectify the predictions in an iterative fashion during the test time. The network uses an encoder-decoder design common

to many semantic segmentation architectures. They combined the strength of a residual network enhanced with SE as SE-Residual block, MixPool block that facilitates the attention and propagation of information flow from the current learning paradigm and that of the previous epoch. The MixPool block uses the previous segmentation map, which contains the information from prior training and uses it to improve the semantic representation of the feature maps.
Their proposed network architecture is a Fully Convolutional Neural Network (FCNN) consisting of four encoder and four decoder blocks. The encoder takes the input image, downsamples it gradually, and encodes it in a compact representation. Then, the decoder takes this compact representation and tries to reconstruct the semantic representation by gradually upsampling it and combining the features from the encoder. Finally, a pixel-wise categorization of the input image is received. Both the encoder and the decoder are built using the SEResidual block, and an additional concatenation of the original resolution feature representation in the encoder is added at each resolution scale. This mechanism minimizes the loss of feature representations during downscaling and upscaling processes. Each encoder network starts with two SE-Residual blocks, which consist of two 3 X 3 convolutions and a shortcut connection known as identity mapping, connecting the input and output of the two convolution layers. Each convolution is followed by a BN and a ReLU activation function. The output of the second SE-Residual block acts as skip connection for the corresponding decoder block. After that, it is followed by the MixPool block, which has the previous epoch segmentation mask and provides a hard-attention over the incoming feature maps. This process is repeated for each of the downscaled layers.
Each decoder network starts with a 4 X 4 transpose convolution that doubles the spatial dimensions of the incoming feature maps. These feature maps are concatenated with feature maps from the corresponding encoder block through skip connections. The skip connections help to propagate the information from the upper layers, which are sometimes lost due to the depth of the network. The skip connections are followed by two SE-Residual blocks, which help to eliminate the problem of vanishing gradient. The MixPool block that utilizes the segmentation mask from the previous epoch is then applied creating a hard-attention over the learnt feature maps. Next, they concatenated the feature maps from the last decoder block and the segmentation mask from the previous epoch. Finally, they applied a 1 X 1 convolution with the sigmoid activation function. The output of this is used to both minimize the training loss, using a combined binary cross-entropy and dice loss, and to generate segmentation masks that are stored as a run-length encoded (RLE) compression for each sample and propagated during the next epoch. The RLE is updated every epoch. Similarly, the network learns to adapt the weights in iterative training, this mechanism is also utilized during the test time. They worked on seven datasets: DRIVE, CHASE-DBI, ISIC 2018, Kvasir-SEG, CVC-ClinicalDB, 2018 Data Science Bowl and EM Dataset. Dice coefficient, IoU, Precision and Recall were used as performance metrics. The model was trained for 100 epochs with Adam as optimizer having learning rate as 1e-4. ReduceLRonPlateau was also used as callback and various data augmentations techniques were also used.

4. HarDNet-MSEG: A Simple Encoder-Decoder Polyp Segmentation Neural Network that Achieves over 0.9 Mean Dice and 86 FPS (url: https://arxiv.org/abs/2101.07172)

Authors proposed HarDNet-MSEG based on the backbone of HarDNet68. HarDNet, have improved dense block of Densenet. It reduces shortcuts to increase the inference speed, and at the same time increases its channels' width for the key layer to make up for the loss of accuracy.

It also uses a small amount of Conv1x1 to increase computational density. Authors used Cascaded partial decoder. A Receptive Field Block was also used. It can strengthen the deep features learned from a lightweight CNN backbone. By using multi-branch with different kernel size convolution and dilated convolution layers, it generates features with the different receptive fields. Afterwards, it applies a 1x1 convolution to merge these features and generate the final representation. They added this module to the skip connection, so that they could enlarge their receptive fields from each different resolutions' feature maps. They performed aggregation by element-wise multiplication. After up-sampling to the same scale, the feature is multiplied with another input feature of the corresponding scale. They performed experiments on datasets: Kvasir-SEG, CVC-ColonDB, EndoScene, ETIS-Larib, PolypDB and CVC-ClinincalDB. The metrics they used were Mean Dice, Mean IoU, Recall, Precision, F2, Accuracy. They performed experiments on different datasets.

First Cut Approach

*** Explain in steps about how you want to approach this problem and the initial experiments that you want to do. ***(MINIMUM 200 words)*** ***

1. I would first start by performing some EDA on the dataset.
2. I will then start performing image segmentation of the data using pretrained image segmentation models like UNET, RESNET, etc.
3. I will then try to replicate the architecture, experiments and results of the chosen paper.
4. Since the authors implemented the code in pytorch, I will try to build the model from scratch using tensorflow.

Notes when you build your final notebook:

1. You should not train any model either it can be a ML model or DL model or Countvectorizer or even simple StandardScalar
2. You should not read train data files

3. The function1 takes only one argument “X” (a single data points i.e 1*d feature) and the inside the function you will preprocess data point similar to the process you did while you featurize your train data
    a. Ex: consider you are doing taxi demand prediction case study (problem definition: given a time and location predict the number of pickups that can happen)
    b. so in your final notebook, you need to pass only those two values
    c. def final(X):

preprocess data i.e data cleaning, filling missing values etc

compute features based on this X

use pre trained model

return predicted outputs

final([time, location])

    d. in the instructions, we have mentioned two functions one with original values and one without it
    e. final([time, location])   # in this function you need to return the predictions, no need to compute the metric
    f. final(set of [time, location] values, corresponding Y values) # when you pass the Y values, we can compute the error metric(Y, y_predict)
4. After you have preprocessed the data point you will featurize it, with the help of trained vectorizers or methods you have followed for your train data
5. Assume this function is like you are productionizing the best model you have built, you need to measure the time for predicting and report the time. Make sure you keep the time as low as possible
6. Check this live session: https://www.appliedaicourse.com/lecture/11/applied-machine-learning-online-course/4148/hands-on-live-session-deploy-an-ml-model-using-apis-on-aws/5/module-5-feature-engineering-productionization-and-deployment-of-ml-models