lesson2-总结

一、上节课遗留的问题

2. 病理图片很大，怎么分 patch 来做了，一直不清楚，目前实际的工作中不分 patch 可能损失信息更多，比如缩小图片大小等，所以老师讲讲 patch 这个思路吧

3. 问题 1.目前分割 sota 前三是什么，大概讲下优缺点或者创新点 2. 打分割比赛常用 tricks

14. 请老师讲一下如何用 HPA pixel_size=0.4 的图像去造 HubMAP 的数据？pixel_size 的变化仅仅通过 resize 大小就可以实现吗？

15. SegFormer 训练太长了，如何加速收敛呢？

~~1.可以提供用 colab 或 Kaggle notebook 跑 baseline 的实操吗？目前老师提供的 baseline 小白无法用在 MAC…~~

看《如何打造舒适的 AI 环境》

~~2. 如何分别看对于不同器官的模型分割的效果。我看 kaggle 许多人测试都会看模型基于不同器官的成绩。老师能提供一些测试的 code 吗？~~

3.老师上节课你没有讲 patch 思路，讲一下，哈哈，毕竟现实我们医院病理切皮哦安很大，必须切分 patch，也弄个 patch 的 code 试试效果吧

~~4. 第一次接触分割，可以对 baseline 的结构能简单讲一下吗~~

~~5. 老师为什么用 ResNext101 作为 encoder 的 Unext, 提交分数远低于 ResNext50 作为 encoder 的模型？是过拟合了吗？~~

~~6. 老师我发现 baseline 自动下载的预训练模型好像是来自弱监督半监督的？这和全监督 imagenet 上得出的预训练模型有什么区别嘛，能不能简单讲一下呢？~~

~~7. 老师提供的 efficienet 参考应该是基于一个高星的 github pytorch 复现库，想请问一下这种实现方式和直接调 timm 从里面抽 FCN 的方式区别大吗？~~

8. ~~老师，class HuBMAPDataset 的__init__函数可以单步调试，但__getitem__函数无法单步调试，应该怎么单步调试？~~training 过程 img = cutout(img)的作用是什么，validating 过程为什么没有？validating 过程下面三行代码的作用是什么？

```
if img.shape[0] % 2 != 0:
    img = img[:-1]
    mask = mask[:-1]
```

二、代码结构

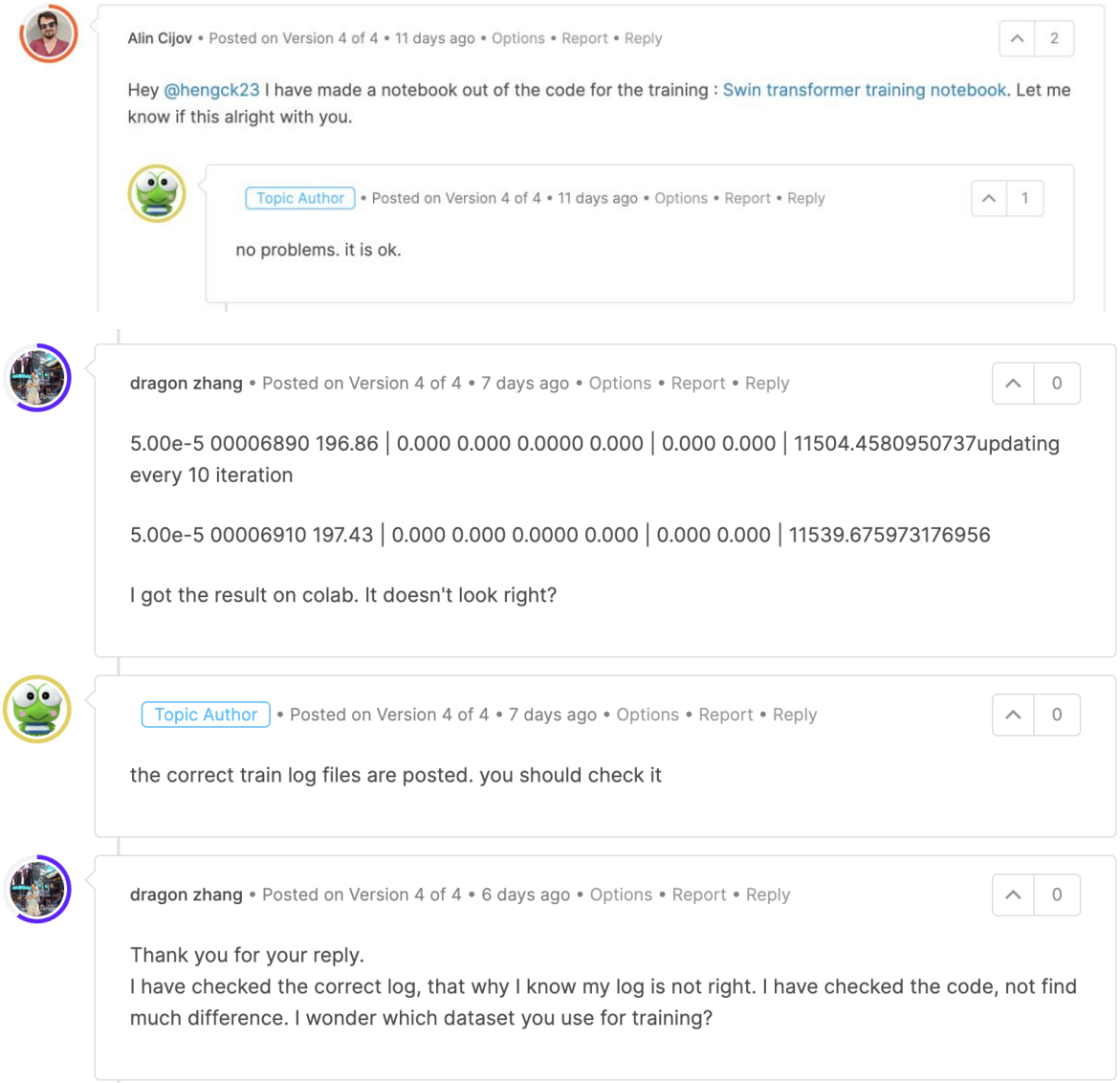
见代码

三、debug 教学

1. 遇到问题如何解决？

LB 0.75 Variable size swin transformer v1 and v2

https://www.kaggle.com/code/hengck23/lb-0-75-variable-size-swin-transformer-v1-and-v2/comments



Topic Author • Posted on Version 4 of 4 • 6 days ago • Options • Report • Reply 0

i just simply convert tiff file to png and rle from csv file to png.

i can only say you need to to do algorithim debugging.

according to my log file, if every thing is correct, you should see validation dice of 0.6+ within first 15 min. So you just need 15 min to check if you setup my code correctly or not.

things that can go wrong:

- data
- model (for get to load pretrain, etc)
- hyperparameters

you need to check of of those in sequence

e.g. to check if the dataset is correct, you need to
1) find dataset that has known results

- either use data set that has been by others in some public kernel
- create your own dataset and run only very simple framework (the objective is not to get good performance but is to use one framework that has no bugs and not sensitive to hyper-parameters. so we choose the simplest like resnet-unet or your old working code that your used before)

2) you will get baseline results in step1. at this step there should not be data problem (but you may be get optimal data). your model will not perform worse than baseline.

3) only after you do step1 and step2, you start to do hyper-parameters search

---

so if you suspect the dataset in my code is wrong setup, you should replace with your own correct data first.

If you want to still want to use my dataset and don't know if you setup correctly, you should interface it with some other simple framework first like unet-resnet (or your old code)

dragon zhang • Posted on Version 4 of 4 • 5 days ago • Options • Report • Reply 0

thank you very much for your patient explanation.

I am not running your code directly, because of lacking related module.

The code I am running is the public notebook based on your code.
The dataset is the public 256×256 mask dataset.

I will check where I can find your dataset.

Thank you again. my own due diligence.

[Training] HuBMAP LB 0.75 Swin Transformer v1 ：
https://www.kaggle.com/code/alincijov/training-hubmap-lb-0-75-swin-transformer-v1

再次告诉我们。深度学习不是一键 SH

Naman Tuli • Posted on Version 8 of 8 • 2 days ago • Options • Report • Reply

0

Hi @alincijov I tested this code but somehow I could not get the dice score in my logs

```
** net setting **
   initial_checkpoint = None

optimizer
   AdamW (
Parameter Group 0
    amsgrad: False
    betas: (0.9, 0.999)
    capturable: False
    eps: 1e-08
    foreach: None
    lr: 5e-05
    maximize: False
    weight_decay: 0.01
)

** start training here! **
   batch_size = 8
                          |-------------- VALID---------|---- TRAIN/BATCH ----------------
rate     iter  epoch | dice   loss   tp     tn      | loss           | time
-------------------------------------------------------------------------------------------
0.00e+0   00000000*   0.00 | 0.000  0.646  0.0000  0.000    | 0.000  0.000   | 20.75287890434265
5.00e-5   00000105*   3.00 | 0.000  0.182  0.0000  0.000    | 0.306  0.503   | 191.58397936820984
5.00e-5   00000210*   6.00 | 0.000  0.121  0.0000  0.000    | 0.155  0.263   | 365.4716787338257
5.00e-5   00000315*   9.00 | 0.000  0.086  0.0000  0.000    | 0.104  0.178   | 538.1359286308289
5.00e-5   00000420*  12.00 | 0.000  0.062  0.0000  0.000    | 0.080  0.134   | 710.7318749427795
5.00e-5   00000525*  15.00 | 0.000  0.050  0.0000  0.000    | 0.059  0.098   | 884.5983004570007
5.00e-5   00000630*  18.00 | 0.000  0.046  0.0000  0.000    | 0.050  0.080   | 1058.469259262085
5.00e-5   00000735*  21.00 | 0.000  0.040  0.0000  0.000    | 0.039  0.061   | 1231.55540060997
5.00e-5   00000840*  24.00 | 0.000  0.032  0.0000  0.000    | 0.031  0.049   | 1403.5147120952606
5.00e-5   00000945*  27.00 | 0.000  0.023  0.0000  0.000    | 0.029  0.045   | 1575.7540881633759
5.00e-5   00001050*  30.00 | 0.000  0.022  0.0000  0.000    | 0.023  0.036   | 1748.2336530685425
5.00e-5   00001155*  33.00 | 0.000  0.018  0.0000  0.000    | 0.020  0.030   | 1921.2369349002838
5.00e-5   00001260*  36.00 | 0.000  0.016  0.0000  0.000    | 0.017  0.026   | 2094.2854833602905
5.00e-5   00001365*  39.00 | 0.000  0.014  0.0000  0.000    | 0.015  0.023   | 2266.8450589179993
5.00e-5   00001470*  42.00 | 0.000  0.011  0.0000  0.000    | 0.013  0.020   | 2439.537337064743
5.00e-5   00001575*  45.00 | 0.000  0.011  0.0000  0.000    | 0.012  0.018   | 2612.5152032375336
5.00e-5   00001680*  48.00 | 0.000  0.010  0.0000  0.000    | 0.010  0.016   | 2783.8875954151154
5.00e-5   00001785*  51.00 | 0.000  0.008  0.0000  0.000    | 0.009  0.014   | 2955.7082829475403
5.00e-5   00001890*  54.00 | 0.000  0.008  0.0000  0.000    | 0.008  0.013   | 3126.2663884162903
5.00e-5   00001995*  57.00 | 0.000  0.007  0.0000  0.000    | 0.008  0.012   | 3297.9572010040283
5.00e-5   00002100*  60.00 | 0.000  0.006  0.0000  0.000    | 0.007  0.010   | 3469.562523841858
5.00e-5   00002205*  63.00 | 0.000  0.005  0.0000  0.000    | 0.006  0.009   | 3641.6275684833527
5.00e-5   00002310*  66.00 | 0.000  0.005  0.0000  0.000    | 0.006  0.009   | 3813.5894770622253
```

```
5.00e-5   00002625*  75.00 | 0.000  0.004  0.0000  0.000 | 0.004  0.007 | 4328.774290323257
5.00e-5   00002730*  78.00 | 0.000  0.004  0.0000  0.000 | 0.004  0.006 | 4499.7483768463135
5.00e-5   00002835*  81.00 | 0.000  0.004  0.0000  0.000 | 0.004  0.005 | 4671.31111073494
```

Also, where could I infer/test with the saved weights as it takes almost 9 hours for 1 fold of the code

• Posted on Version 8 of 8 • 8 hours ago • Options • Report • Reply

0

you cannot do machine learning just by running code and adjust hyperparameters.
you have to understand the numbers.

the bces loss for train and validation is
-log(p)= 0.004

this means that p = exp(-0.004)= 0.9960079893439915

if that is true, how can validation dice loss be zero?

i would stop training after 12 epoch. no need to waste GPU computation.

obviously, the model got stuck in some degnerate solution.
e.g. wrong initialisation, wrong data, fp16 numerical error, etc...

i would draw the input, predict and ground truth to debug the code. i suspect the model is predicting all zeros

[placeholder LB 0.79 (single fold?)] my experiment results

https://www.kaggle.com/competitions/hubmap-organ-segmentation/discussion/332941

we now enter phrase 2 of the competition: from 0.80 to ~0.83

- it is now the battle of external data, SSL masked pertaining, unlabelled learning ... weak/self/semi-supervised, domain adaption, ...

Ali Salhi • (21st in this Competition) • 9 days ago • Options • Report • Reply

0

Great 0.8 :-)
My 0.75 was achived with basline notebook and some new training with larger bone with some size modification I achived 0.7.
And a trick (or approch) that pushed the score from 0.70 to 0.75 , a boost that I belive you didn't yet try and I guess with your current solution it might give you a boost of 0.02 or so. Or maybe not! If segformer is doing great with HuBMap.

Gunes Evitan • (18th in this Competition) • 20 hours ago • Options • Report • Reply

0

That is very intuitive. It would probably work for all organs.

kingjohnson • (100th in this Competition) • 19 hours ago • Options • Report • Reply

0

I am curious about which way you choose for training, the whole image(resize) or image tiles.

四、切 patch 原因？

问下医学图像切 patch 的大致策略是什么？ - Master KangKang 的回答 - 知乎

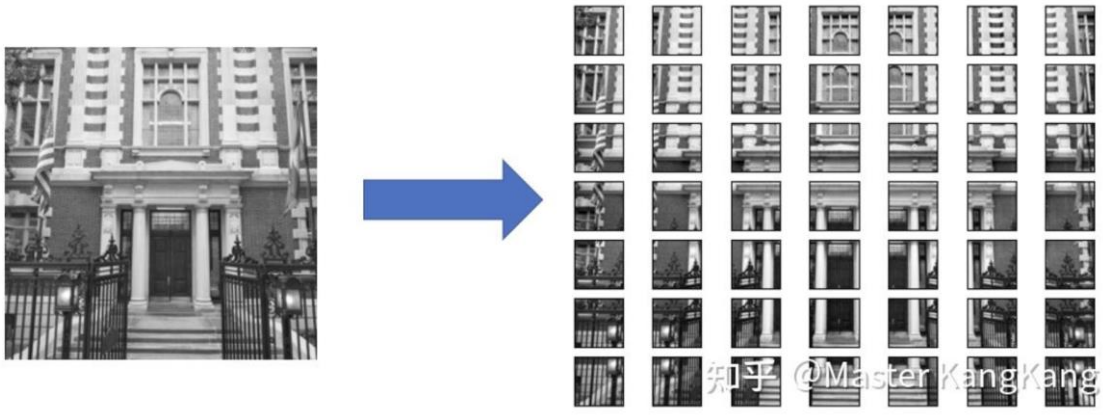
https://www.zhihu.com/question/436498354/answer/2513462682

在计算机视觉中，图像数据往往比模型对结果的影响更关键。一个泛化能力强的模型免不了要处理海量的数据，然而各种各样的图像可能会带来这些问题：

- 数据收集到的图像大小不一致，无法统一进行训练，如果强行Resize，会对图像产生扭曲，缩放，可能会丢失信息
- 图像规模非常大，显存不够
- 即使在显存充足的情况下，对于大图，模型可能难以捕捉到其中的细节，尤其对于动辄几个G的医学图像，可能病灶只有那么几十个像素的大小
- 数据集太小，数据难以收集。或者大量数据被污染，只有少量数据包含了明显的特征。
- ……

而为这些问题提供一个解决方案是：**切分Patch**

也就是用一个滑动窗口将二维或是三维的图像切分成许多大小一致的小块作为训练数据。这样一来图像大小得到了统一，并且小的patch相当于对图像局部的放大，降低显存并且大大丰富了数据集，例如，以55*55大小的窗口，5像素的步长滑动切分一张500*500大小的图像，就可以得到 $(\frac{500-55}{5}+1)^2=8100$ 张子图像。当然，可以根据实际需求量确定Patch大小和步长。



五、分析模型的误差

六、To do List

0. 本节课提供的放在：

(base) ➜ HuBMAP ls

alin_cijov hengck23 input working

1. 探索 alin_cijov & hengck23 的代码 （ debug 已经修好 ）

参考上节课讲的 hubmap_3rd 提交代码. & main_0724.py 完成自己的提交代码.

探索 256x256 & 不分 patch 的两种数据集哪个好

bce loss => bce loss + dice loss

2. 替换 swin + upernet 为 segformer

3. 对自己的模型进行误差分析

encode => pred mask ===== GT mask loss . BCE + DICE

decoder1 => pred mask ===== GT maks loss

decoder2 => pred mask ===== GT mask loss

decoder3(最后一层) => pred mask ===== GT maks loss