11 月汇报

基于块的 VGG 网络结构—为什么最经典的网络架构之一?

VGG—Based on Block Construction—Why it is one of the classics network framework?

AlexNet 的成功不是偶然，一个更大更深的 LeNet 在那个年代的大数据中训练效果非常好。AlexNet 首次证明了通过神经网络训练学习得到的特征可以超越人类手工设计的特征。但它还不够完美，设计不够简洁，并且也不知道为什么要如此设计(具体每一层的超参数)。简洁的设计往往更容易被世人记住。如此，基于块的网络模型：VGG 被提出了。VGG 块的结构如下：

It is not accident for AlexNet' success, which is a bigger, deeper LeNet, it effect in big data which in that time is brilliant. AlexNet firstly prove feature through nerve network train to get, can beyond hand made. However, it is not perfect also, it's design is not concise, and we don't know how each of his layers are designed, (include value of hyper-parameter in each layer). So, the design of the profile is still easy to remember. Based on above, VGG—based on Block Network Construction is proposed. VGG Blovk's graph as follow:

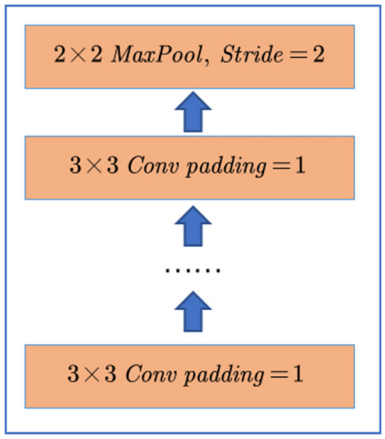


Figure 1 VGG Block Design

VGG Block 包括 n 个 3*3 卷积组合，然后最后经过 2*2 的最大池化. 实现高宽减半通道数翻倍的操作. 高宽减半通道数翻倍也是一个非常经典的设计. 通过组合不同的 VGG Block，我们可以得到不同的网络结构：例如：VGG-16, VGG-19.

综上, VGG 给我们带来了什么设计思路:

1. 块的结构往往更加利于理解. 一个块，或者一个 Stage，通常进行的是高宽减半，通道数翻倍.
2. 组合不同个数的块，和块中的超参数(这里指的就是卷积层的个数)可以得到不同复杂度的模型.

VGG Block include *n* 3*3 Convolution layer, than via a 2*2 MaxPool layer, which can get output that height and width are half and doubling the number of channels. which is also a significantly classical design. Through compose a vary of VGG Block, we can get different Net Construction, such as VGG-16 and VGG-19.

To sum up, what is brought by VGG's design thinking:

Block construction usually more easy to understand, a *block* or a *Stage*, which get half height and width, in the same time, doubling the number channels.

Combination a vary of Block, and hyper-parameter in block, which can a group of differently complexity model.

ResNet 网络核心思想—为什么能够训练超过 1000 层神经网络?

ResNet Core Thinking—Why it can train more than 1000 layer nerve network?

首先来看一下如果过深的网络有什么问题：首先过深的神经网络在结构上是不断堆叠的，通过反向传播后靠近数据的层中的参数的梯度不断叠加，那么很容易造成参数不稳定的问题，这导致在反传更新时，更容易出现梯度爆炸和梯度消失现象。同时随着网络层数不断叠加，进行反传时，梯度总是从上到下的，如果上层收敛较快，那么就会导致反传时梯度很小，不断反传，进行梯度累加，最终导致靠近数据的层会非常不容易收敛。

那么为什么 ResNet 可以训练深层网络：

从直观上看: View From Intuition:

上述提到，一个思路：将乘转换为加，$F(X)\longrightarrow F(X)+X$，来看看 ResNet 是怎么做的：

ResNet 引入了残差块，所谓残差块在之前的结构对比就是通过增加一个额外的“高速公路“来将梯度累成转换为梯度累加，以此来克服在梯度反传中梯度不断累加而造成的问题。ResNet 核心架构设计：Res Block:

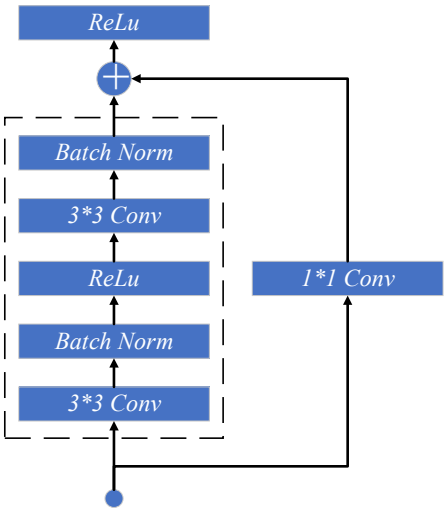


Figure 2 Res Block

请注意，residual 部分可以是什么层也不加，这里添加 1*1 的 Conv 是为了将X和$F(X)$的 shape 进行匹配。

Note: residual part can flat connect with finally output, in which add a *1*1 Conv* in order to match shape between X and $F(X)$

从梯度上看：View From gradient:

$$normal\ parameter\ step:$$

$$y=f(x),\ backward:\ for\ x'\ paramerate\ gradient:\ \frac{\partial y}{\partial w},$$

$$then\ step:\ w=w-lr\times\frac{\partial y}{D\times\partial w}$$

$$Origin:$$

$$y'=g(f(x))$$

$$backward:\ for\ x'\ gradient:$$

$$\frac{\partial y'}{\partial w}=\frac{\partial y'}{\partial y}\frac{\partial y}{\partial w}$$

$$=\frac{\partial g(y)}{\partial(y)}\cdot\frac{\partial y}{\partial w}$$

$$when\ above\ layer\ converge\ faster,\ \frac{\partial gs(y)}{\partial(y)}\ may\ be\ become\ small.$$

$$which\ case\ bottom\ layer'\ step\ not\ work.$$

$$so,\ how\ to\ solve\ it?$$

$is\ that\ transform\ \ form\ of\ multiplcation\ to\ addling$:

$Residual$:

$y'' = f(x) + g(f(x))$

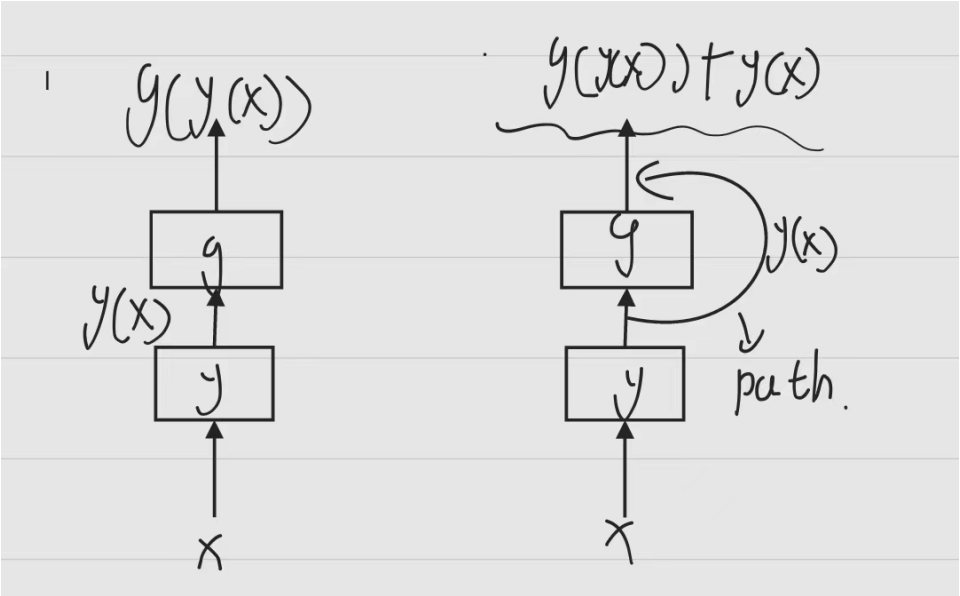
$backward$: $for\ x'\ gradient$:

$$\frac{\partial y^{'}}{\partial w} = \frac{\partial y^{'}}{\partial y} + \frac{\partial y}{\partial w}$$

$$= \frac{\partial g(y)}{\partial (y)} + \frac{\partial y}{\partial w}$$

$while\ above\ layer\ converge\ faster,\ bottom'\ layer\ is\ still\ work.$

$which\ guarantee\ \frac{\partial y^{'}}{\partial w}\ sitll\ can\ step.$



卷积神经网络发展简述:

Development of Convolutional neural network:

深度神经网络的复兴起源于 LeNet，LeNet 最开始是应用于一个手写数字识别的系统，在 MNIST 数据集上效果很好, LeNet 的架构：使用卷积和最大池化提取图片特征，并且识别模式，最终使用全连接层使其塌陷到最后的类别数。LeNet 带来了两个思想，使用卷积学习图片中的空间信息，然后使用全连接层转换到类别信息。真正给深度学习带来热潮的神经网络是 AlexNet，一个更大更深的 LeNet，在 LeNet 的基础上使用了更深层的网络结构，使用更小的 3*3 卷积核，以及 ReLU 和 Dropout, 缓解模型参数梯度消失及降低模型复杂度。AlexNet 在 ImageNet 竞赛中崭露头角，成为当时热门的神经网络。但上面我们说过 AlexNet 还不够好，不够简单，基于块的 VGG 改变了人们设计深度神经网络的观念，VGG 的核心思想就是 VGG 块，VGG 使用多个 3*3 的卷积核，最后通过 MaxPooling 层将高宽减半，其他和 AlexNet 设计相同，其中包括最后的全连接层。这样就有一个问题，这些模型都太复杂了。也就是模型参数太多，研究者发现，整个网络的模型参数量都集中在全连接层，全连接层本质上是一个矩阵，需要将输入通过矩阵坍塌到输出，如果输入和输入足够大的话，全连接层是积聚大量参数的，但是卷积层是一个不那么多参数的层。那么一个想法就是用卷积层代替全连接层，这就是 NiN 网络，NiN 中没有全连接层，核心思想是 NiN 块，包含一个卷积，然后二个 1*1 卷积用来对通道维做全连接，使得输出想要的形状，最后使用全局平均（Global AvgPooling）获得输出，塌陷到我们想要的类别数。NiN 拥有更少的参数量，不容易过拟合，但是缺少了全连接层，这也使得它需要更大的 Epoch，才能收敛到同样精度。不过基于这种 NiN 也就是“网络中的网络”思想，GoogLeNet 被提出了，在 NiN 的基础上他做了一个类似于集成学习思想，并行层的连接网络，就是我全都要了。Inception 块就是这样，他有一个输入，然后对这个输入做不同的操作，最后在通道数（dim=1）进行 concat，这样我们就可以吸收不同层（也就是不同处理）得到的输出了。不同的 Inception 有不同的效果，目前使用最多的就是 Inception V3,V4.GoogLeNet 也是第一个几乎达到百层的神经网络设计了。不过上面我们提到过参数稳定性的问题, 层数太多，

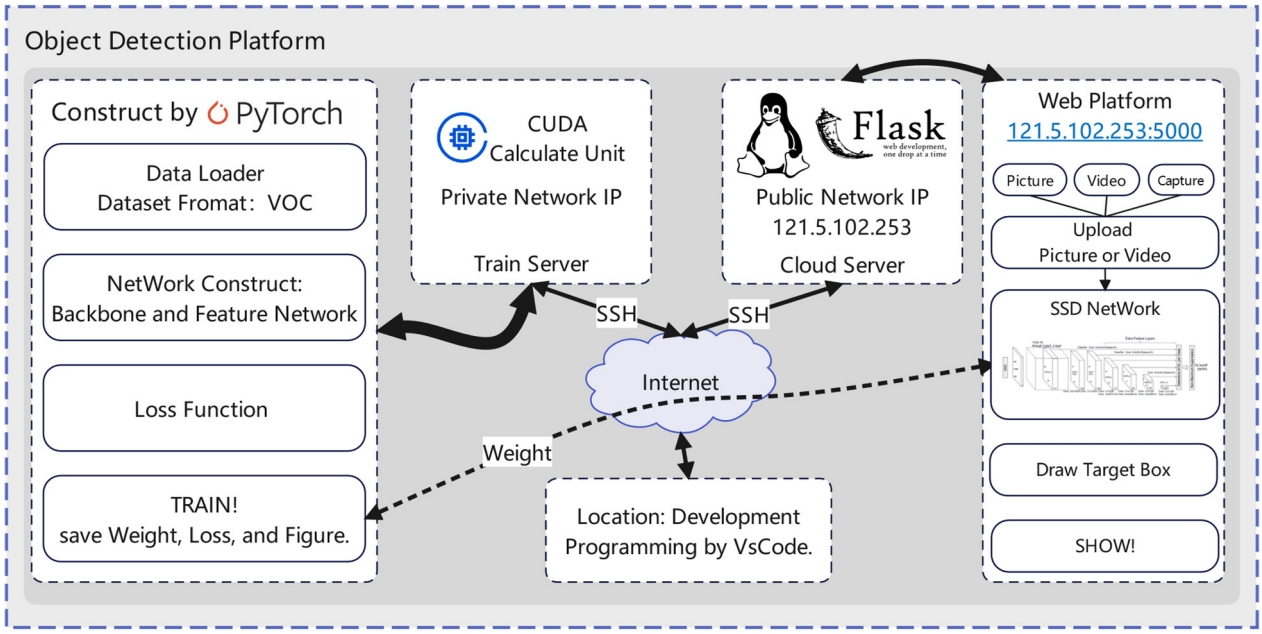
这种累计起来的效应是非常可怕的，前传：从 data 到 loss，反传：从 loss 到每层的 parameter，这个过程需要链式法则的参与，参数不稳定伴随着深度网络的层数越发明显，所以研究者提出了 Batch Norm，也就是小批量中将输出拉到一个合理的范围内，BN layer 不会改变精度，但是它可以使参数更稳定，这样我们可以设置更大的学习率，可以加速我们的训练。不过即使这样我们仍然无法做到更深的神经网络，参数稳定了，但是参数反传过程中累计的效应仍然存在，一个简单的思想是将乘法转换为加法，基于此 ResNet 被提出了，其思想是将乘法性质转换为加法形式，实现方式是使用跃层连接(Skip Connection)，在设计残差块的过程中通过设计一个“高速公路“，以实现输入能够跨层连接，这样就解决了参数在反传当中的稳定性问题。后面我们会看到有很多有意思的设计都是基于 ResNet 的思想，例如 Transformers 等。

The revival of deep neural networks originated with LeNet, which was first applied to a handwritten digit recognition system, it worked well on the MNIST dataset, LeNet’s architecture: use ConV2d and MaxPooling to extract image features recognize patterns, and half height and width , and eventually use a Full Connected layer(short as FC) to collapse category to the final number. LeNet offers researchers two way to thinking: Using a Convolution to learning spatial information in images, then use FC layer transformer to category we want. AlexNet, larger, deeper of LeNet, was the neural network that really propelled deep learning thriving, it used a deeper network structure, smaller kernel size such as 3*3 Conv2d based on LeNet, and it also use ReLU and Dropout to mitigate model parameter gradient disappearance and reduce module complexity. It become the most popular neural network at that time. This leads to the issue that all of these models are overly complicated, in other words, the models have too may parameters. Researchers discovered that the FC layer contains the majority of the model parameters. FC is essentially a matrix that needs to collapse the input through the matrix multiplication to the output. The FC layer is accumulating a large number of parameters if the inputs and output are large enough, but the convolutional layer is a less parametric layer. Then an idea is to use a convolutional layer instead of a FC layer, which is the NiN, NiN block include a Conv2d, and two 1*1 Conv2d are serve as full connection to channel dimension, making the output the desired shape, and finally the output is obtained using Global AvgPooling,, collapsing to the number of categories that we want. NiN has less number of parameters and less prone to overfitting, but which also makes it require a larger Epoch to converge reach same accuracy with full connection. But based on NiN, which is “network of networks” idea, GoogLeNet was proposed, and on the basis of NiN, it made a similar integrated learning idea, parallel layer of connected networks, is that I Want it ALL. The Inception block is just that, it has an input and then does different operations on this input and finally concat at the number of channer (dim=1), so that we can absorb the pattern obtained by different layers(i.e. different processing), different Inception Block with different effect, Inception V3, V4 are currently used most frequently, GoogLeNet is also the first neural network design to reach almost 100 layers. But above we mentioned the parameter stability problem, too many layers, this cumulative effect is very scary, forward: from Data to Loss, backward: from Loss to parameter of each layers. this process needs the involvement of Chain Law, parameter instability is accompanied by deep network layers more and more obvious. so researcher proposed Batch Norm, also It is small batch will pull the output to a reasonable range, BN layer will not change the accuracy of train, but it can make the parameters more stable, it allow we set a lager learning rate and can accelerate our training. but we still can’t use deeper network, the parameter are stable, but the cumulative effect of the parameter backpropagation is still in here. A simple thinking is to convert multiplication to addition, ResNet is that do this, it implementation is to use Skip Connection, in the process of designing the residual block by designing a “highway” so that the input can be connected across layers. which solves the accumulation effect in the backward. As we will see later, there are many interesting esigns based on ResNet ideas, such as Transformer.

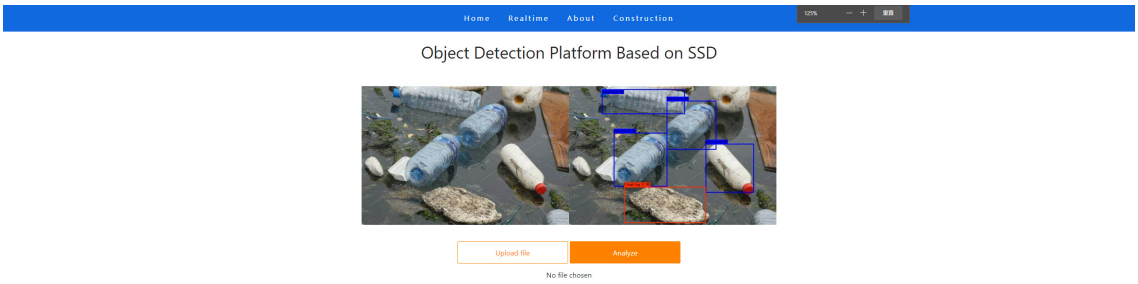
杂谈

10 末有个考试，复习了一段时间，因此那段时间没有产出，然后最近实验比较紧，同时距离 CET-6 还有一个月，所以也在学习英语。然后 11 月中旬在准备开题以及做毕设的 demo，因此这个月的报告交的有点晚了，不过整体进度还不错，毕设 demo 做了个简单一点的，然后也在继续学习深度学习知识的内容。

关于毕设：
架构：

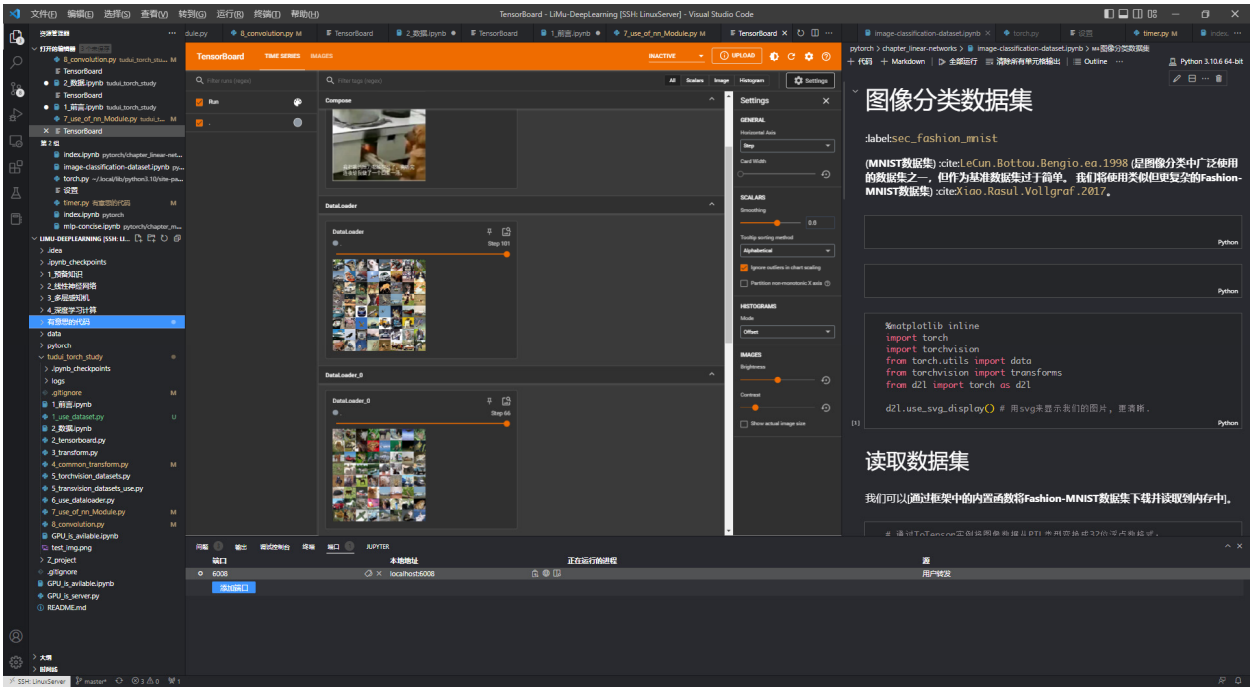


网址: http://121.5.102.253:5000/ 目前只能检测一些比较清晰的水面垃圾，经过测试，识别率还可以，漏检比较大，不过我也在利用自己标的数据做微调(fine turning)，又训练了 100epoch 吧，效果还可以，下图是主界面.

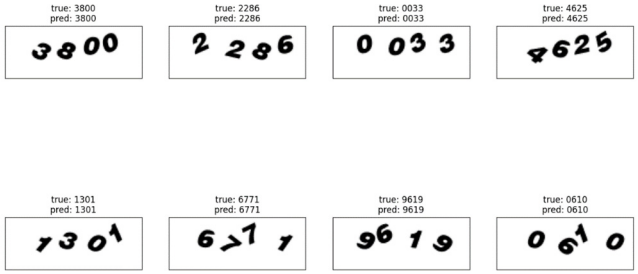


VSCODE 永远嘀神!!!

不仅支持直接在远程服务器上写代码，并且直接支持配置端口转发，以及集成 TensorBoard，这样可以直接在 VSCODE 中启动，非常方便。

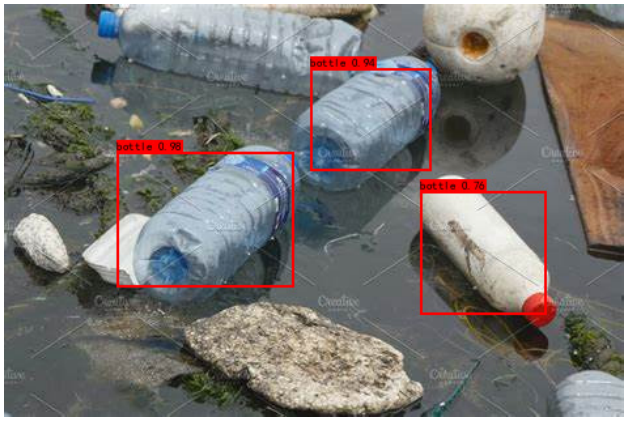


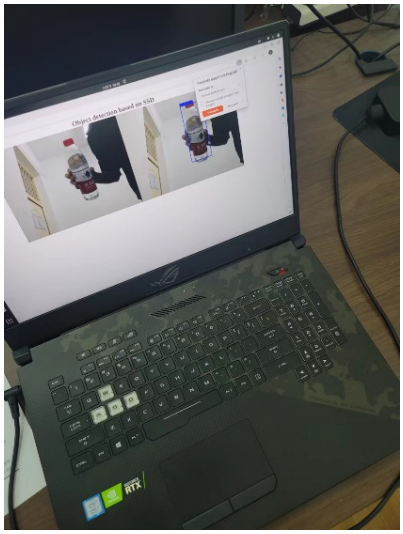
用 pytorch 实现了验证码的识别，算是回顾了一下，网络架构模仿的是 AlexNet 的网络架构。以及 nin 网络架构，减少了参数量.



工作

1. 仍然在学习李沐的《Dive to Deep Learning》，书和视频一直在边看边学。不过没有空去实现一遍他的代码，下个月补上。目前现代卷积神经网络基本已经了解完了，下面会看物体检测。以及实战一下 Kaggle 图片分类的比赛。
2. 这个月已经将毕设的 demo 实现了一下. 使用了 Flask 搭建了一个简单的目标检测平台，目前还没是实现上传图片或者视频等功能，简单做了个一个直接检测摄像头的。关于 SSD 算法，训练了个 300 epoch，数据集用的是在 CSDN 上面的一个数据集，效果有好有坏. 漏检现象有点严重，然后我自己也在爬取图片，不过因为是直接从 bing 上爬取图片，图片质量参差不齐. 后面可能会使用 labelimg 自己制作大约 1k 个数据集吧。





3. 毕设部署到我的云服务器上了，测试了一下感觉还行，目前只能检测图片，其他功能还没来得及做，也是非常紧急的，经常是连夜改 bug，里面的东西其实也是东拼西凑的，有点冗余，后面有时间可能会优化一下吧。网站网址：Object Detection Platform（http://121.5.102.253:5000/）
4. 英语，英语太差了，所以一直在学习英语，并且临近考试，时间有点紧，上午通常是在学英语。

问题

1. 多输出卷积是如何实现的，没有想太明白。是先将所有输入通道卷积后求和，形成一个输出矩阵，然后再多输出卷积，每个多输出卷积都是不同的，所以形成不同的输出通道吗。那么这样卷积核的个数就有输入卷积核+输出卷积核了。不知道我如此理解对不对。这样的话多输出通道的卷积就是：输入卷积数+输出卷积个数。

$$\cdot \text{输入}\ \mathbf{X}: c_i \times n_h \times n_w$$
$$\cdot \text{核}\ \mathbf{W}: c_o \times c_i \times k_h \times k_w$$

   对于 1*1 卷积的理解，对每个像素的通道进行全连接层，那么这个过程是怎么样的.
   能理解，是对通道做全连接，就是将输入通道数乘以矩阵得到想要的输出通道数，不过过程有点没想通.
2. 在做 Object Detection 时，遇到了算法选择的问题，目前是使用 SSD 算法，效果有好有坏,. 所以我在想要不要使用另一种热门的目标检测算法：YoLo，所以想问一下 SSD 算法和 YoLo v5 我应该选择那种算法。目前使用 SSD 的效果感觉还不错。
3. 我目前想要识别出河流区域，然后生成 Mask，这个地方是需要使用 Attention 吗，我之前是想用 OpenCV 的运动背景的 API 来做，不过对于河流来说好像并不是很好，所以如果我只关注河流区域我是需要 Attention 来做这方面的实现吗?
4. 还有就是数据，数据很重要，一般来说数据是自己爬，或者找到一些可靠的素材网站，然后爬取。不过这样图片质量参差不齐，不知道老师在数据方面有什么心得吗。
5. 深度学习中需要大量的矩阵推理，以及矩阵运算，有时候有些看不懂，例如矩阵，向量求导，需要学习下《矩阵分析》这种课程吗。