de文章翻译完成，讨论了一下，发现deep speckle 这篇文章限制了散射范围在焦点附近，需要看一下那部分的文章，需要看一下resnet和denseblock。明天需要把这篇文章的引文找出来。需要一个resnet的ppt。

安装了环境。下载了anaconda tensorflow keras

试了一下源代码需要的1.4的环境：

从pypi.org 下载了tensorflow.whl文件到下载文件夹

进入cmd进入下载文件夹

Pip uninstalled tensorflow

然后安装失败

在别的电脑上是可以安装的，但是在这台电脑上没办法

同时我决定去学tensorflow2的语言规范，这样应该能够把代码移植

总是出现装了tensorflow以后keras显示没装或者反过来的问题

当时建立虚拟环境的时候名为tensorflowwl不知道出什么问题

2.0的教程也无法运行并且出现相同的错误，找人帮忙搞，准备重装python然后时间特别长，

重新装的时候发现是装anaconda的时候装给自己，但是安装包的时候是默认装给所有人

新装的时候装在F

需要python和tensorflow版本对应

但是还是不行，安装了tensorflow2.4.1以后会提示找不到DLL模块，查询了以后说是因为python的版本问题，但是anaconda自带的版本是3.8，所以可能需要搞一个新的环境。

最后感觉还是很麻烦，决定重装ananconda从blog.csdn.net/heivy/article/datails/92992887看版本号然后在清华anaconda镜像下载旧版本3.6py的

然后面临的问题是pip版本过于老旧，解决方案是在cnpython.com/pypi/pyhamcrest下载最新版本然后在anaconda prompt中cd文件所在路径然后pip install 文件名就升级了pip

上面的网址是pypi中文社区

然后由于目前的文章的代码都是在1.x下跑的我们决定下载低版本，只能从清华镜像源下载，然后安装的时候记得加—users

再装keras

3.20-21学了两天tensorflow

之后打算从kaggle上翻一些新的论文

然后打算读nature子刊多模光纤的那篇文章，因为代码比较完全

然后出现了电脑只使用CPU而不使用GPU 计算非常的情况，所以把普通版本卸载了

但是运行一个教学代码的时候的时候发现没办法运行，所以去查了一下需要安装cuda，但是又找到了知乎上的一篇文章有不用cuda的anacondagpu的使用方式：zhuanlan.zhihu.com/p/51224259

然后发现这个方法里没有keras pip install keras 查了一下版本对应2.3.1

然后运行又发现缺少PIL就又安装了pillow pip install pillow 但是GPU利用率只有百分之10到20，而且是直接在prompt里面运行的（从jupyter navigator里面的env中的tf-gpu的箭头的terminal打开），要在jupyter notebook里面打开还需要一些流程：blog.csdn.net/qq\_36895854/article/details/108780379

也就是安装新的环境之后必须要安装一个新的核心ipykernel才能在jupyter notebook里面打开，py3.5以上都可以直接安装不用指定kernel版本

在kernel里面可以选择换核心

然后2000张图片的数据使用的内存可能就占三分之二了，后面进行正式光学图片处理可能需要更大的内存

注意下载package的时候的vpn是否是打开的

也就是有两种运行方法，一种是直接从navigator进入terminal另一种是打开jn换核心

但是发现这台电脑上的硬件配置远远不能满足计算需求，所以尝试向其他实验室借，向徐老师组（GTX 1080）借了之后用xshell登录，在文件新建ssh，主机写给了的ip，然后点击新建的文件会提示登录，输入用户名和密码，然后发现他们没有tensorflow环境，就在上面

Python3 -m pip install tensorflow-gpu==1.15.0

python3 -m pip install –upgrade pip

python -m pip install tensorflow-gpu==1.4.0

python -m pip install keras==2.1.2

pip list

python -m pip install matplotlib==3.0.5

mkdir speckle

ls

rz

但是一直由于和电脑上原有的python冲突一直没有办法运行

现在尝试复现多模光纤的代码，发现给的训练数据过于巨大，有六万张图片，被重新编码成了bin文件，需要使用matlab还原

发现不需要还原也能运行代码，就是可能需要将得到的数据进行还原

重新在服务器上尝试同步运行代码，但是在服务器上的Python2下面有一个问题就是weakref无法调用，所以在python3下面安装了tensorflow1.15并且将tensorboard和tensorflow-tensorborad都进行了更新，更新到最低版本就能跑了，但是速度差别不大

并且在复现代码的时候发现代码里面指定了需要使用cpu进行计算，所以非常慢，发邮件问作者为什么指定cpu但是还没得到回复

如果发现navigator无法运行就升级 conda update anaconda-navigator，但是升级以后就无法直接下载vscode了：[www.jb51.net/article/192890.htm](http://www.jb51.net/article/192890.htm) blog.csdn.net/u011930054/article/details/112383654

Skimage安装的时候换成scikit-image

终于解决了gpu显存不够的问题了，可以动态分配显存，或者减小batchsize，虽然这样可以训练了但是显示的时候显存还是不够。

今天尝试使用jupyter连接服务器的主机：blog.csdn.net/weixin\_40512956/article/details/104618156，但是连接的时候是自动连接到python2，后来突然好了，重新输入指令如果不自动打开的话就按ctrl+c

然后发现地空没有装cuda和cudnn，尝试到地空的上面看看有没有环境，发现有gpu版本但是没有jupyter notebook也没有cuda和cudnn，所以还是尝试在徐老师机子上做，但是没有sudo权限无法开展工作

通过获得sudo权限，我们可以顺利安装cuda。将下载好的文件

<https://pan.baidu.com/s/1GlI00s2xBQOKYzXgOTJvAg#list/path=%2F> 提取码: vyg5放在主目录下面

运行代码 sudo sh 文件名

顺利安装。

利用代码 cat /usr/local/cuda/version.txt可以查看当前版本号

现在数据集的来源是，一方面尝试向相关的论文作者借数据集，我自己也尝试在matlab上

通过仿真产生数据集，参考的论文就是Exploiting the speckle-correlation scattering

matrix for a compact reference-free holographic

image sensor补充材料中给出的代码，代码并不能直接运行，需要进行一定的修改，我需要找到标准的字母图片才能制作散斑图片，被我使用作为标准字母的数据库是mnist数据库，可以从官网上直接下载，下载mnist数据库http://yann.lecun.com/exdb/mnist/以后再反编码回原来的数据，由于4个文件都是使用linux系统下gunzip打包好的.gz文件，要在matlab中做，在https://zhuanlan.zhihu.com/p/66280431找到了相应的代码，现在就是尝试用老师给的TM仿真的代码来产生训练数据对

代码中由于它的训练数据非常多，我调试程序的时候是先取了其中第一个数据先进行尝试，但是发现无法直接绘制mnist的图片，所以我又尝试了一下另外一个方法：<https://blog.csdn.net/k3v1n1990s/article/details/67645343>，直接使用这部分代码会报错：

错误使用 fread

文件标识符无效。使用 fopen 生成有效的文件标识符。

出错 visualisation (第 10 行)

images\_data = fread(fid1,'uint8');

需要加上文件后缀名，这可以在官网上找到，也就是加上.gz的后缀名，但是加上了之后发现运行结果如下：



并且运行很久还不结束，所以我决定还是不使用这个代码，参考斯坦福大学Andrew Ng教授的课件，通过loadMNISTImages.m和loadMNISTLabels.m两个文件读取mnist数据集。

**loadMNISTImages.m**

function images = loadMNISTImages(filename)

%loadMNISTImages returns a 28x28x[number of MNIST images] matrix containing

%the raw MNIST images

fp = fopen(filename, 'rb');

assert(fp ~= -1, ['Could not open ', filename, '']);

magic = fread(fp, 1, 'int32', 0, 'ieee-be');

assert(magic == 2051, ['Bad magic number in ', filename, '']);

numImages = fread(fp, 1, 'int32', 0, 'ieee-be');

numRows = fread(fp, 1, 'int32', 0, 'ieee-be');

numCols = fread(fp, 1, 'int32', 0, 'ieee-be');

images = fread(fp, inf, 'unsigned char');

images = reshape(images, numCols, numRows, numImages);

images = permute(images,[2 1 3]);

fclose(fp);

% Reshape to #pixels x #examples

images = reshape(images, size(images, 1) \* size(images, 2), size(images, 3));

% Convert to double and rescale to [0,1]

images = double(images) / 255;

end

**loadMNISTLabels.m**

function labels = loadMNISTLabels(filename)

%loadMNISTLabels returns a [number of MNIST images]x1 matrix containing

%the labels for the MNIST images

fp = fopen(filename, 'rb');

assert(fp ~= -1, ['Could not open ', filename, '']);

magic = fread(fp, 1, 'int32', 0, 'ieee-be');

assert(magic == 2049, ['Bad magic number in ', filename, '']);

numLabels = fread(fp, 1, 'int32', 0, 'ieee-be');

labels = fread(fp, inf, 'unsigned char');

assert(size(labels,1) == numLabels, 'Mismatch in label count');

fclose(fp);

end

但是这个使用还不是很方便，最后我决定使用Nikolay Chumerin

制作的手写识别项目的代码进行改进<https://www.pianshen.com/article/2006707798/>

几种神经网络和它所完成的任务

Unet：

Def unet(pretrained\_weights=None, input\_size1=(None, None, 1)):

Input = Input(input\_size1, name=’input’)

conv1 = Conv2D(16, (3,3), activation=’relu’, padding=’same’)(input)

conv1 = Conv2D(16, (3,3), activation=’relu’, padding=’same’)(conv1)

pool1 = MaxPooling2D(pool\_size=(2,2))(conv1)

conv2 = Conv2D(32, (3,3), activation=’relu’, padding=’same’)(pool1)

conv2 = Conv2D(32, (3,3), activation=’relu’, padding=’same’)(conv2)

pool2 = MaxPooling2D(pool\_size=(2,2))(conv3)

conv3 = Conv2D(64, (3,3), activation=’relu’, padding=’same’)(pool2)

conv3 = Conv2D(64, (3,3), activation=’relu’, padding=’same’)(conv3)

pool3 = MaxPooling2D(pool\_size=(2,2))(conv3)

conv4 = Conv2D(128, (3,3), activation=’relu’, padding=’same’)(pool3)

conv4 = Conv2D(128, (3,3), activation=’relu’, padding=’same’)(conv4)

up5 = concatenate([UpSampling2D(size=(2,2))(conv4), conv3], axis=-1)

conv5 = Conv2D(64, (3,3), activation=’relu’, padding=’same’)(up5)

conv5 = Conv2D(64, (3,3), activation=’relu’, padding=’same’)(conv5)

up6 = concatenate([UpSampling2D(size(2,2))(conv5), conv2], axis=-1)

conv6 = Conv2D(32, (3,3), activation=’relu’, padding=’same’)(up6)

conv6 = Conv2D(32, (3,3), activation=’relu’, padding=’same’)(conv6)

up7 = concatenate([UpSampling2D(size=(2,2))(conv6), conv1], axis=-1)

conv7 = Conv2D(16, (3,3), activation=’relu’, padding=’same’)(up7)

conv7 = Conv2D(16, (3,3), activation=’relu’, padding=’same’)(conv7)

conv8 = Conv2D(1, (1,1), activation=’sigmoid’)(conv7)

model = Model(inputs=[input], outputs=[conv8])

model.summary()

return model

然后是神经网络的训练方面还是决定使用华为云，可能比较麻烦的就是需要随时保存神经网络训练的模型，因为每隔一个小时就会间断一次，需要从上一次保存下来的模型里面读取数据继续训练，我参考<https://tellyouwhat.cn/p/machine-learning-tensorflow-keras-how-to-gracefully-continue-training-from-where-it-drops/>实现了这个目标，下一步的目标就是将波士顿大学的模型移植过来进行训练并使得能够达到目标，然后使得他们给的数据文件能够跑通，然后就是开始创建自己的数据集

LeCun杨乐春 我们选取batch size时不妨这样操作：

1. 当有足够算力时，选取batch size为32或更小一些。
2. 算力不够时，在效率和泛化性之间做trade-off，尽量选择更小的batch size。
3. 当模型训练到尾声，想更精细化地提高成绩（比如论文实验/比赛到最后），有一个有用的trick，就是设置batch size为1，即做纯SGD，慢慢把error磨低。

神经网络的结构：

网络结构不是作者自行设计的

原作者的论文中对于实际网络结构的作图中也有许多错误

首先在实际训练中并没有dropout层，并且使用的不是真正的反卷积，而是用上采样加反卷积操作替代的反卷积，首先是作一个正确的图，然后从作者参考的网络设计的原始文章中解释一下网络的设计思路，然后看一下原始文章有没有提供代码，观察一下他们真正的网络结构

一个讲解DenseNet很清晰的博客：<https://blog.csdn.net/qq_37555071/article/details/108377880?utm_medium=distribute.pc_relevant.none-task-blog-2%7Edefault%7EBlogCommendFromMachineLearnPai2%7Edefault-3.control&depth_1-utm_source=distribute.pc_relevant.none-task-blog-2%7Edefault%7EBlogCommendFromMachineLearnPai2%7Edefault-3.control>

一个讲解matlab绘图的博客

<https://blog.csdn.net/zy122121cs/article/details/49761307>

<https://blog.csdn.net/qq_41375609/article/details/100854662>

**卷积核（kernels）与滤波器（filters）的关系 这篇文章不错**

感觉效果上限是因为损失函数的设置，所以修改进行训练