遥感图像分类

1.2 AlexNet

@tm9161

L2

AlexNet

1. 遥感图像数据集载入
2. AlexNet结构与创新
3. AlexNet搭建、训练与预测

遥感影像分类——NWPU-RESISC45

喜爱 10

NWPU-RESISC45数据集，用于遥感影像分类，源地址：http://www.escience.cn/people/JunweiHan/NWPU-RESISC45.html

trainer 2 83 2020-09-01

详情 相关项目 创建项目

数据集介绍

数据集背景：

- NWPU-RESISC45数据集
- 包含45种土地利用类型的遥感影像
- 提取自Google Earth
- 由西北工业大学于2016年发布

数据集内容：

包含45个类别文件夹，每个文件夹下对应各自700幅遥感影像

影像信息：

- size: 256*256*3
- pixel resolution: 30m ~ 0.2m
- total number: 45*700=31500

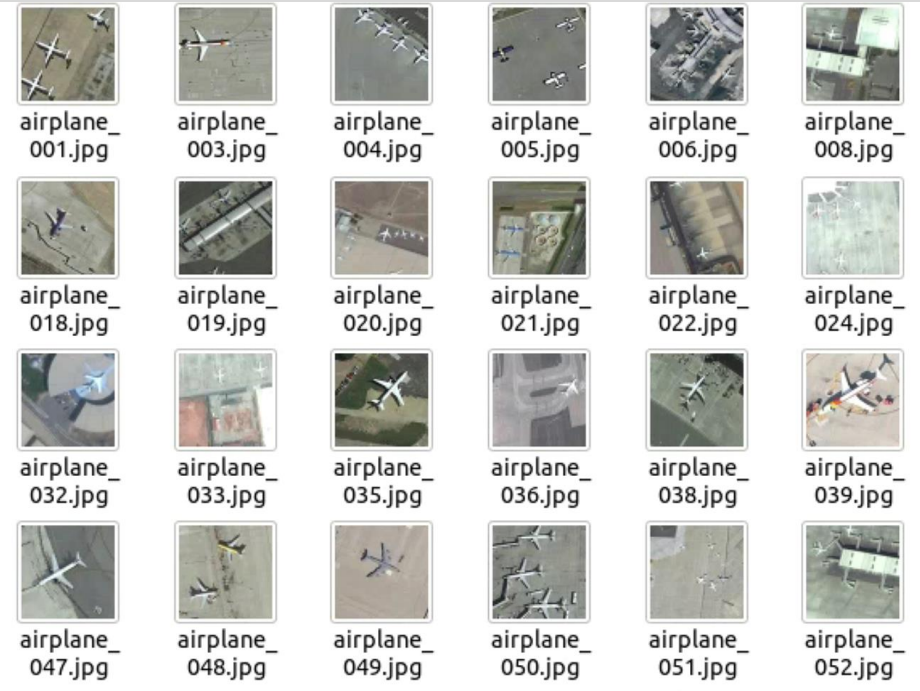
标注信息：

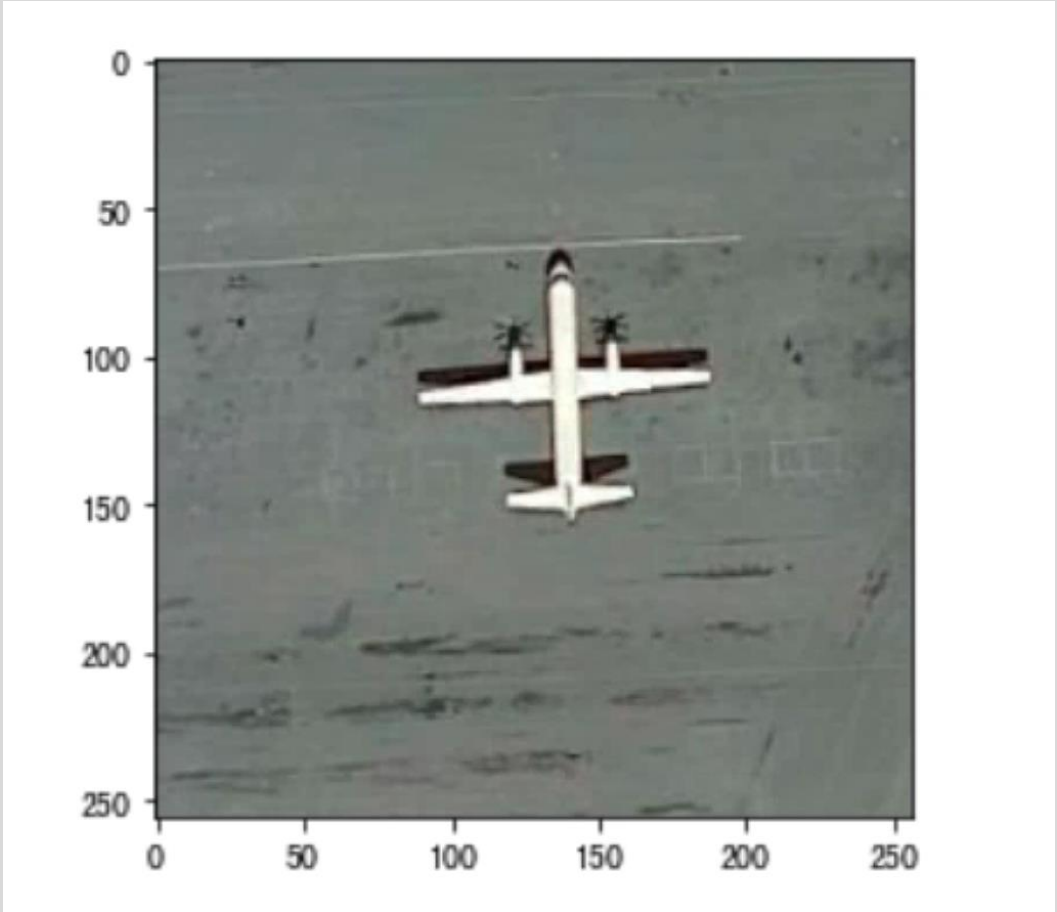
45类：

- airplane,
- airport,

https://aistudio.baidu.com/aistudio/datasetdetail/51873

遥感图像数据集





包含31500张遥感图像（45类*700张），256x256像素的彩色图。

本次使用其中的5类，划分每类630张为训练集，70张为测试集。

载入数据

1.按路径读取

2.预处理
a.归一化
b.水平翻转
c.批大小
d.随机
e.尺寸
f.独热编码

```
train_dir = 'sat2/train'
test_dir = 'sat2/val'

im_size = 224
batch_size = 16
```

```
train_images = ImageDataGenerator(rescale = 1/255,horizontal_flip=True)
test_images = ImageDataGenerator(rescale = 1/255)
```

```
train_gen = train_images.flow_from_directory(directory=train_dir,
                                             batch_size=batch_size,
                                             shuffle=True,
                                             target_size=(im_size, im_size),
                                             class_mode='categorical')
```

查看索引

```
classes = train_gen.class_indices
```

```
classes
```

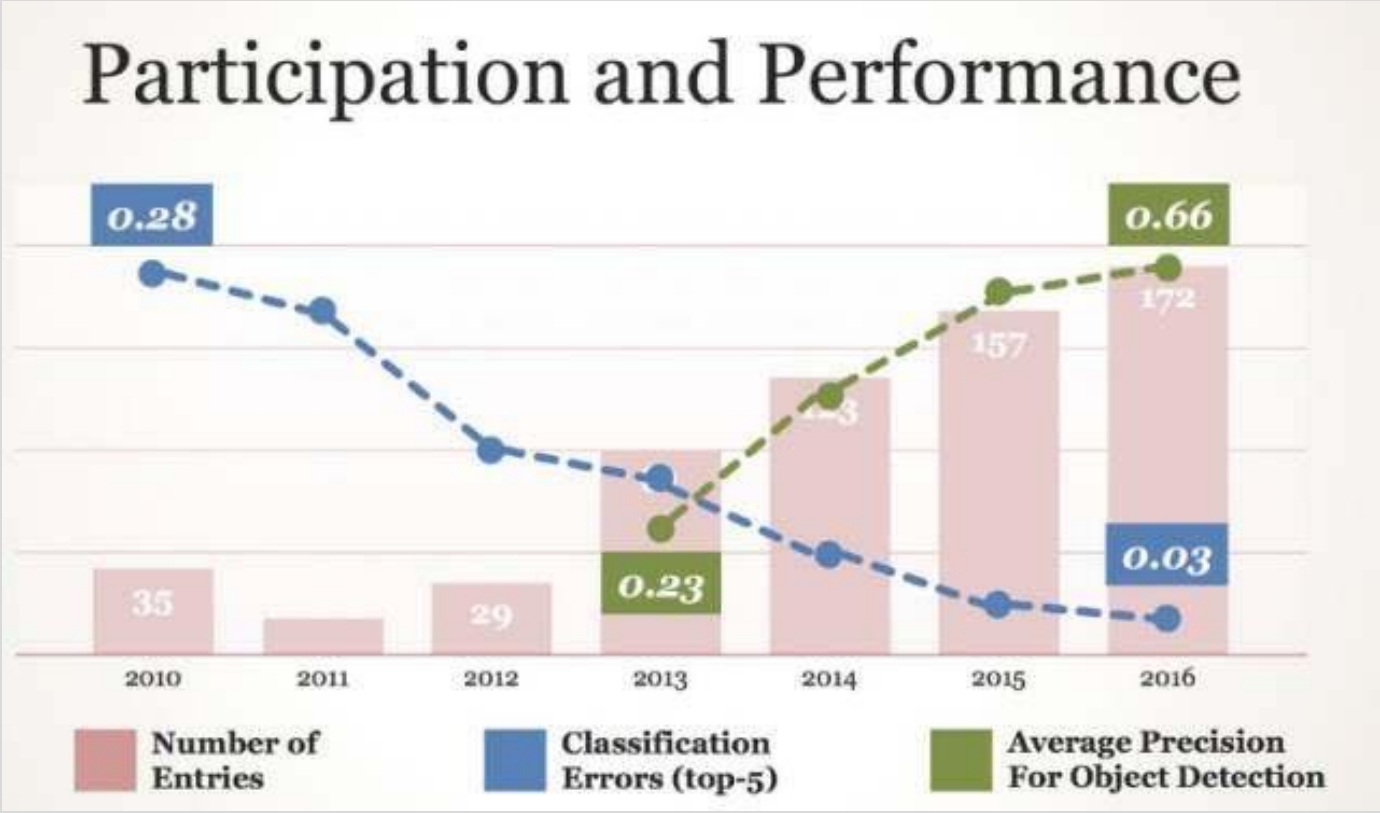
```
{'airplane': 0, 'bridge': 1, 'palace': 2, 'ship': 3, 'stadium': 4}
```

ILSVRC

ILSVRC（ImageNet Large Scale Visual Recognition Challenge）是近年来机器视觉领域最受追捧也是最具权威的学术竞赛之一，代表了图像领域的最高水平。

ImageNet数据集是ILSVRC竞赛使用的是数据集，由斯坦福大学李飞飞教授主导，包含了超过1400万张全尺寸的有标记图片。

AlexNet是2012年ImageNet竞赛冠军获得者Hinton和他的学生Alex Krizhevsky设计的。



激活函数：ReLU

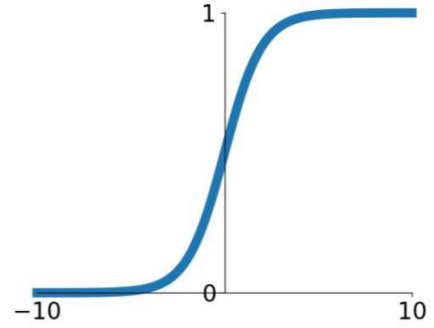
1.模型收敛快，避免梯度消失。

2. 计算简单，运算速度快。

Activation functions

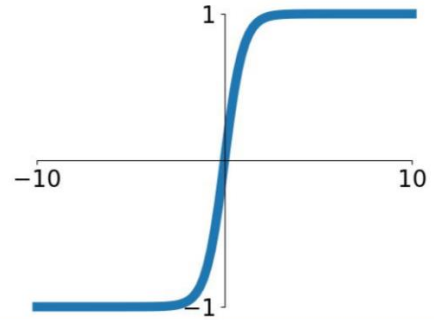
Sigmoid

$$\sigma(x) = \frac{1}{1+e^{-x}}$$



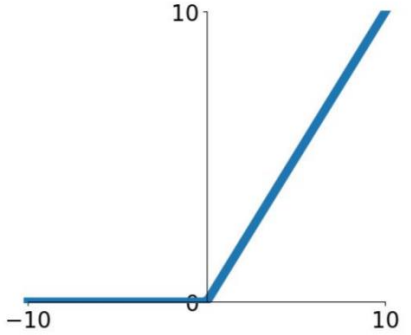
tanh

$$\tanh(x)$$

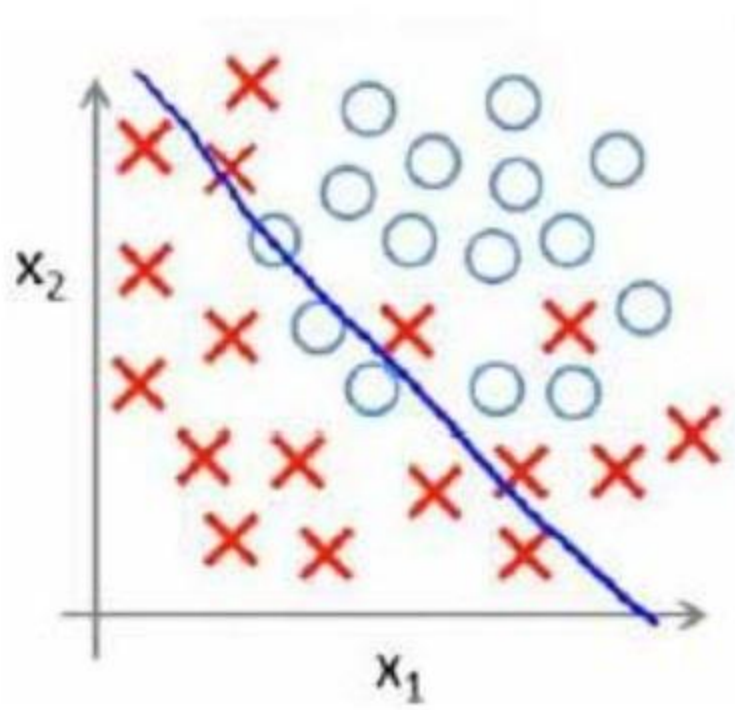


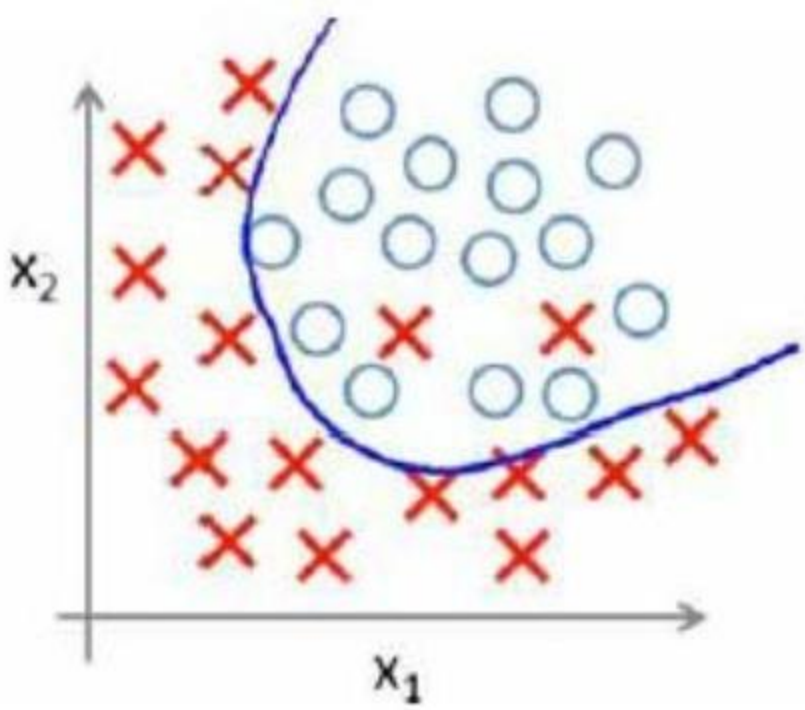
ReLU

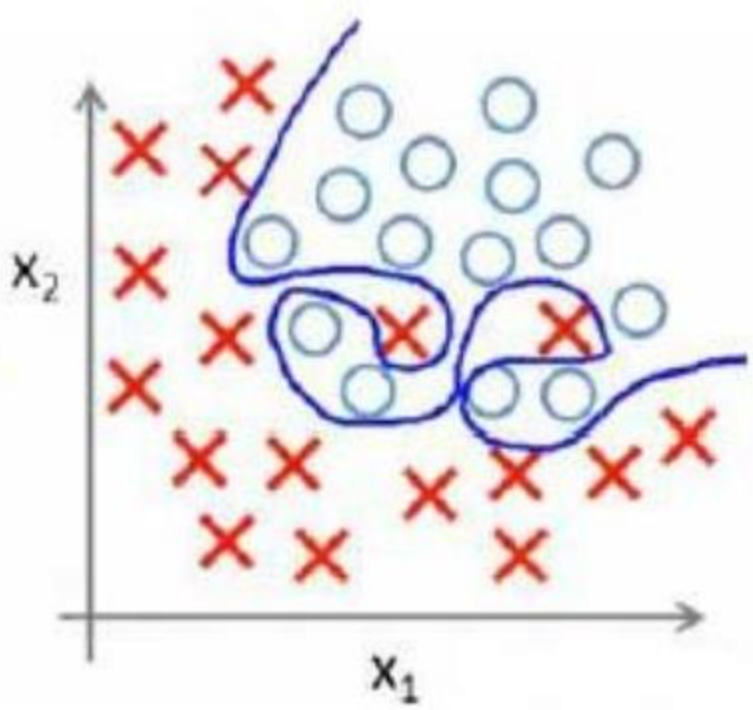
$$\max(0, x)$$



过拟合 Overfitting





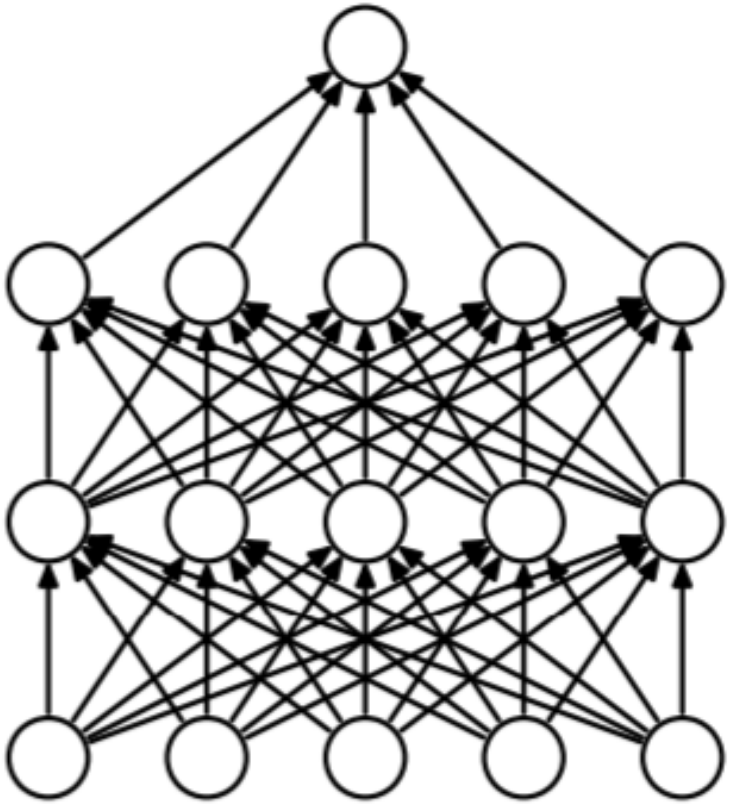


欠拟合（Underfitting）

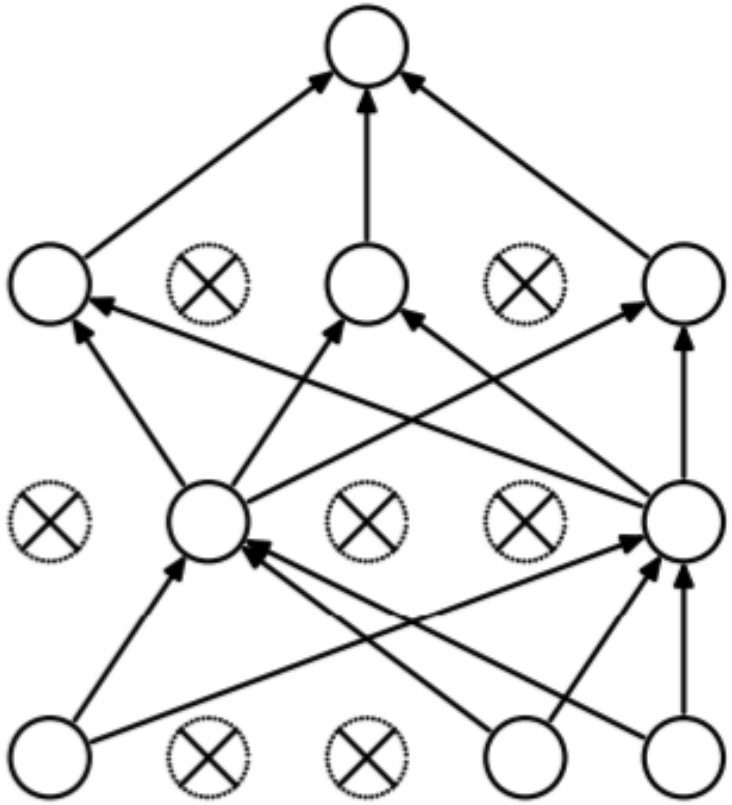
过拟合（Overfitting）

Dropout

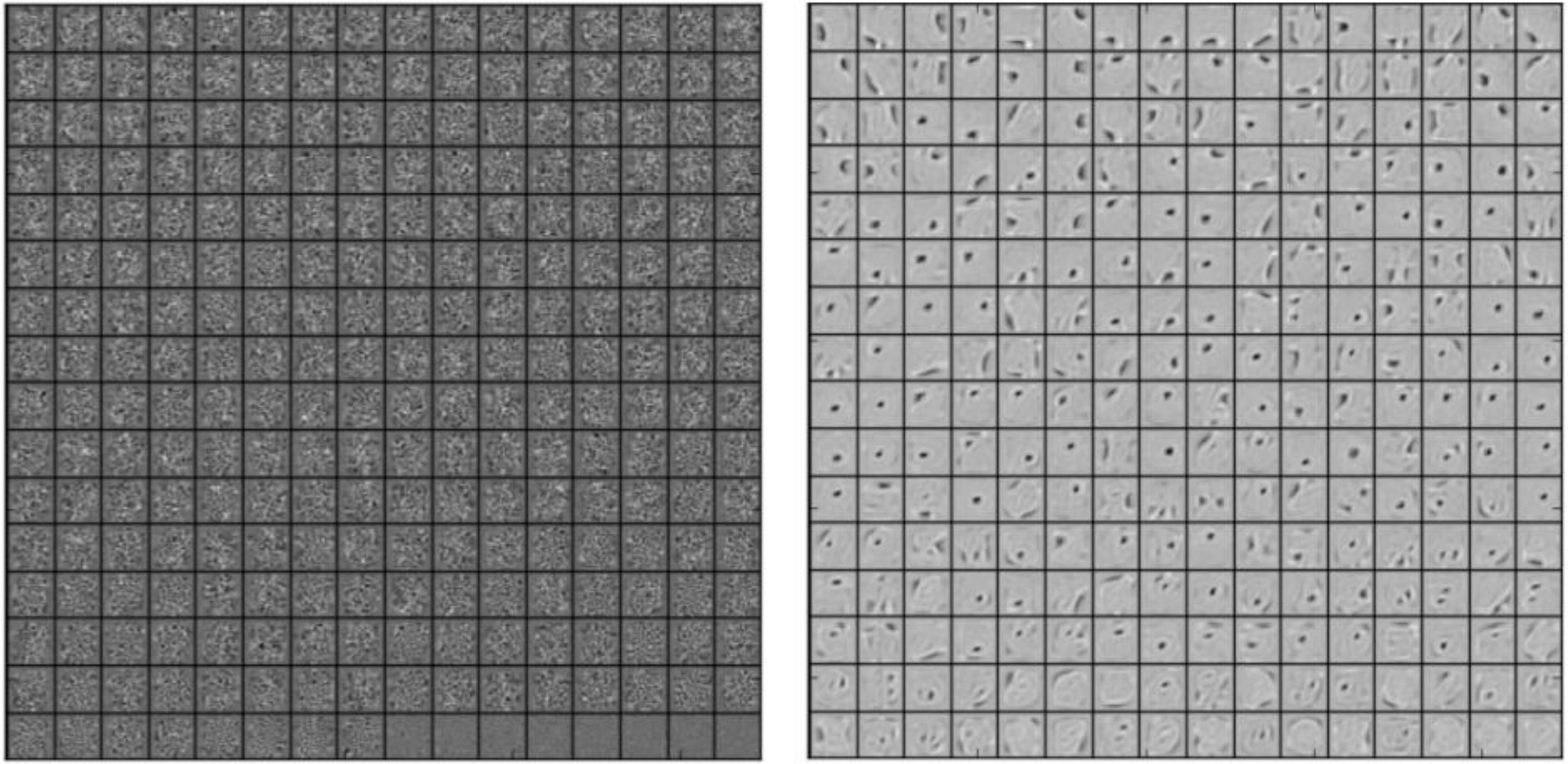
每次训练都随机让一定神经元停止参与运算，增加模型的泛化能力、稳定性和鲁棒性，避免过拟合。



(a) Standard Neural Net



(b) After applying dropout.

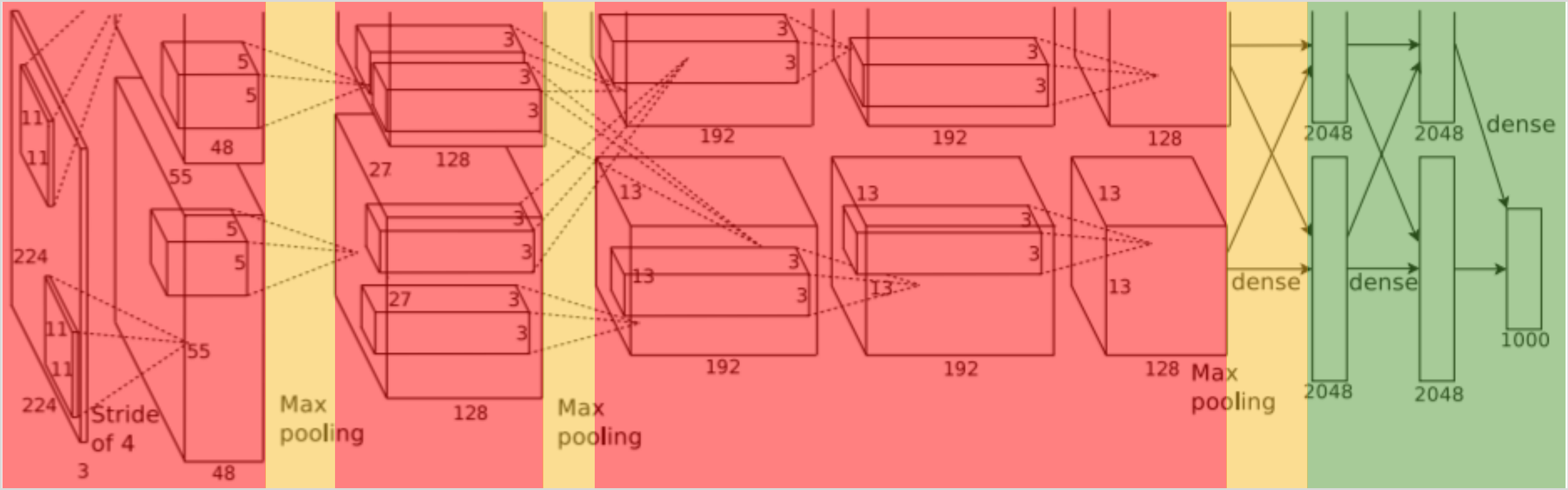


(a) Without dropout

(b) Dropout with $p = 0.5$.

Figure 7: Features learned on MNIST with one hidden layer autoencoders having 256 rectified linear units.

网络结构



卷积层　　下采样层（池化层）　　神经网络（全连接层）

步长 Stride & 加边 Padding &参数 Param

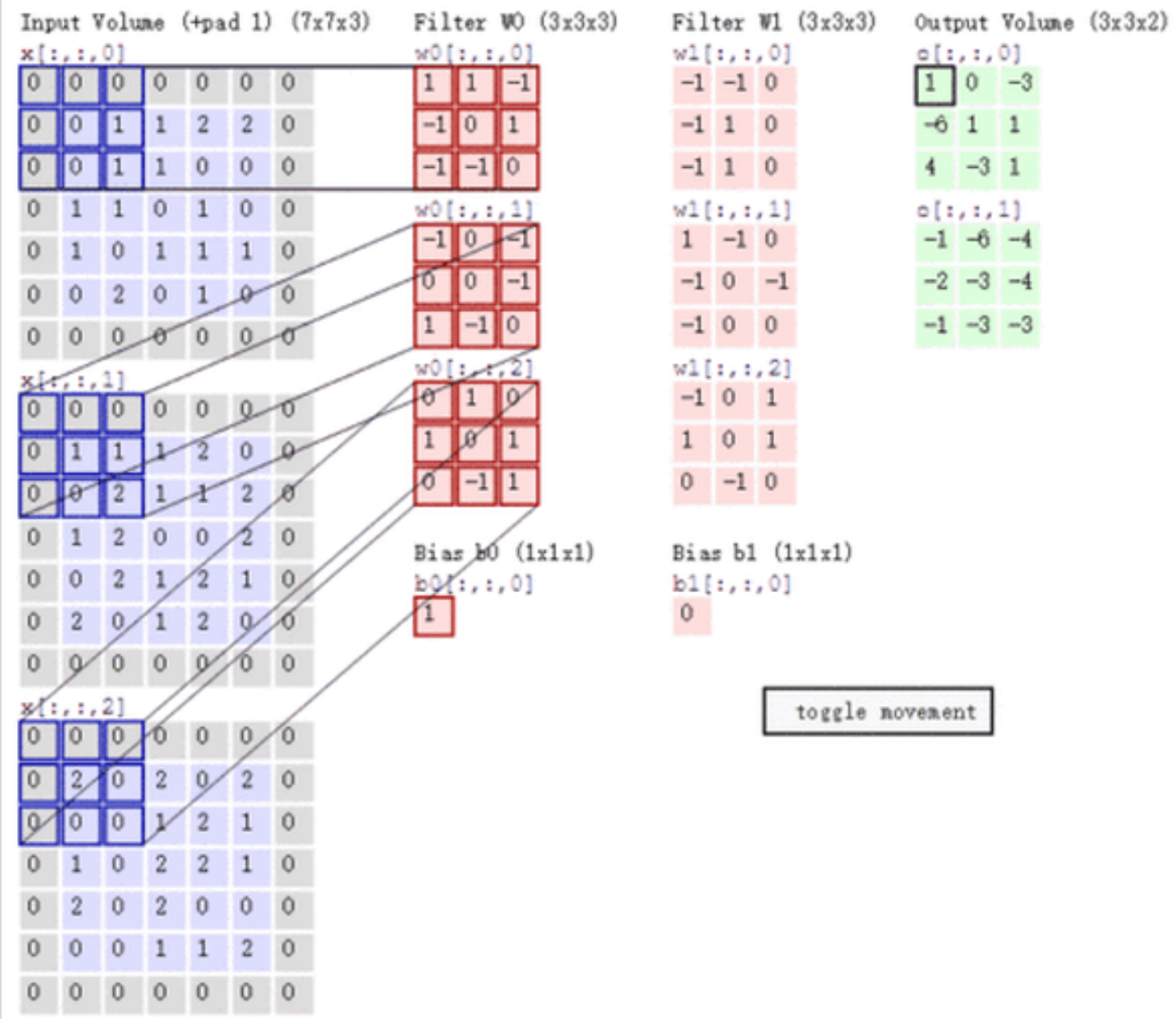
卷积后尺寸=（输入图像大小-卷积核大小+加边像素数）/步长 +1

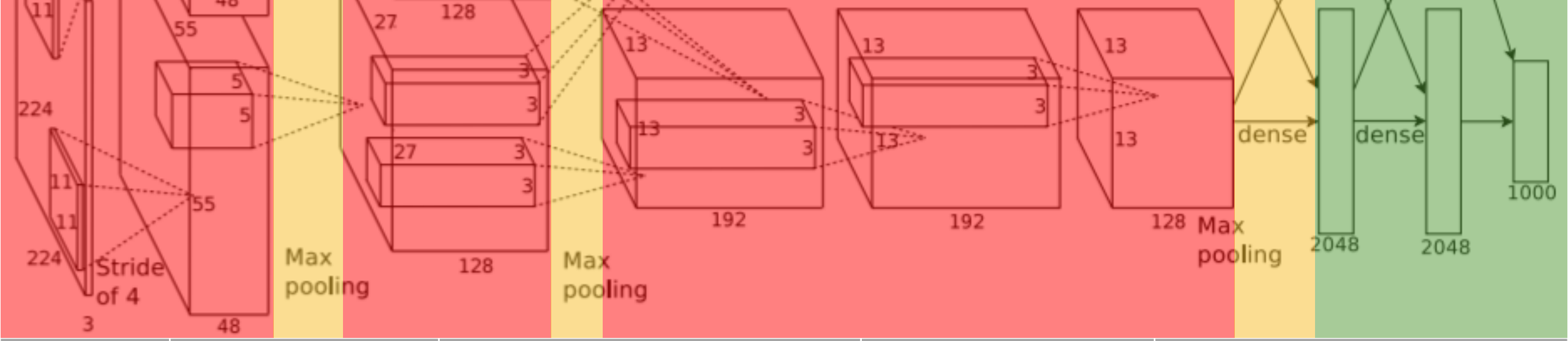
Tensorflow 默认：Padding= ‘valid ‘（丢弃），strides=1
设置：Padding= ‘same’ ：保证输出和输出尺寸不变，自动设置填充

参数：
卷积层：（卷积参数（卷积核各部分）+偏置参数）*卷积核的个数
池化层：无需要训练参数
全连接层：神经元连接权重+偏置参数

卷积层参数：

（3层*3x3大小 +1*偏置）*2个卷积核





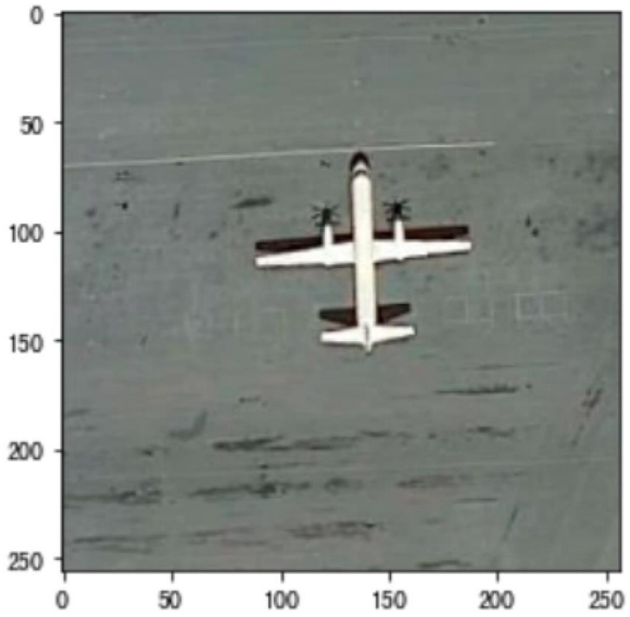
	输入	卷积、池化、神经元	输出	训练参数
输入层	224*224*3	Padding((1,2),(1,2))	227*227*3	0
卷积层1	227*227*3	48个 11*11卷积核 步长为4 relu	48*55*55（227-11）/4+1	3*（11*11）*48 +48=17472
池化层1	48*55*55	3*3 步长为2	48*27*27	0
卷积层2	48*27*27	128个 5*5卷积核 步长为1 relu same	128*27*27	48*（5*5）*128 +128=153728
池化层2	128*27*27	3*3 步长为2	128*13*13	0
卷积层3	128*13*13	192个 3*3卷积核 步长为1 relu same	192*13*13	128*（3*3）*192+192=221376
卷积层4	192*13*13	192个 3*3卷积核 步长为1 relu same	192*13*13	192*（3*3）*192+192=331968
卷积层5	192*13*13	128个 3*3卷积核 步长为1 relu same	128*13*13	192*（3*3）*128+128=221312
池化层3	128*13*13	3*3 步长为2	128*6*6	0
全连接层1	128*6*6（Dropout 0.5）	2048个神经元 relu	2048	128*6*6*2048+2048=9439232
全连接层2	2048（Dropout 0.5）	2048个神经元 relu	2048	2048*2048+2048=4196352
输出层	2048	5个神经元 softmax	5	2048*5+5=10245

图片读取&预处理

```
In [3]: img = cv2.imread('1.jpg',1)
        #读取图片

In [4]: plt.imshow(img)

Out[4]: <matplotlib.image.AxesImage at 0x7fdb782b0ac0>
```



1.图片读取：cv2.imread

2.图片大小调整：cv2.resize

3.图片维度调整：reshape

4.归一化：/255

```
In [5]: img.shape

Out[5]: (256, 256, 3)

In [6]: img = cv2.resize(img,(224,224))
        img = img.reshape(1,224,224,3)
        img = img/255
        #图片预处理

In [7]: img.shape

Out[7]: (1, 224, 224, 3)
```

模型预测

```
In [8]: predict = new_model.predict(img)

In [9]: predict
Out[9]: array([[9.9936479e-01, 7.3952382e-09, 2.1086180e-07, 6.3472008e-04,
                1.7892945e-07]], dtype=float32)

In [10]: label = ['airplane','bridge','palace','ship','stadium']

In [11]: label[np.argmax(predict)]
Out[11]: 'airplane'
```

1

数据采集

```
train_dir = 'sat2/train'
test_dir = 'sat2/val'

im_size = 224
batch_size = 16
```

```
train_images = ImageDataGenerator(rescale = 1/255,horizontal_flip=True)
test_images = ImageDataGenerator(rescale = 1/255)
```

```
train_gen = train_images.flow_from_directory(directory=train_dir,
                                             batch_size=batch_size,
                                             shuffle=True,
                                             target_size=(im_size, im_size),
                                             class_mode='categorical')
```

2

建立模型

```
model = tf.keras.Sequential()
```

```
model.add(tf.keras.layers.ZeroPadding2D(((1,2),(1,2)),input_shape=(224,224,3)))
model.add(tf.keras.layers.Conv2D(filters = 48,
                                 kernel_size = (11,11),
                                 strides = 4,
                                 activation = "relu"))
model.add(tf.keras.layers.MaxPooling2D(pool_size = (3, 3),strides = 2))
model.add(tf.keras.layers.Conv2D(filters = 128,
                                 kernel_size = (5,5),
                                 padding = 'same',
                                 activation = "relu"))
model.add(tf.keras.layers.MaxPooling2D(pool_size = (3, 3),strides = 2))
model.add(tf.keras.layers.Conv2D(filters = 192,
                                 kernel_size = (3,3),
                                 padding = 'same',
                                 activation = "relu"))
model.add(tf.keras.layers.Conv2D(filters = 192,
                                 kernel_size = (3,3),
                                 padding = 'same',
                                 activation = "relu"))
```

3

模型训练

```
model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=0.0005),
              loss='categorical_crossentropy',
              metrics=['acc'])
```

```
history = model.fit(train_gen,epochs=15,validation_data=val_gen)
```

```
Epoch 1/15
197/197 [==============================] - 8s 28ms/step - loss: 1.5417 - acc: 0.2492 - val
_loss: 1.3921 - val_acc: 0.3314
Epoch 2/15
197/197 [==============================] - 5s 26ms/step - loss: 1.3198 - acc: 0.3603 - val
_loss: 1.3553 - val_acc: 0.3629
Epoch 3/15
197/197 [==============================] - 5s 26ms/step - loss: 1.2456 - acc: 0.4091 - val
_loss: 1.0470 - val_acc: 0.5200
Epoch 4/15
197/197 [==============================] - 5s 26ms/step - loss: 1.1184 - acc: 0.5236 - val
_loss: 1.0447 - val_acc: 0.5400
Epoch 5/15
197/197 [==============================] - 5s 26ms/step - loss: 1.0050 - acc: 0.5632 - val
_loss: 0.9982 - val_acc: 0.5886
```

4

模型测试

```
predict = new_model.predict(img)
```

```
predict
```

```
array([[9.9936479e-01, 7.3952382e-09, 2.1086180e-07, 6.3472008e-04,
        1.7892945e-07]], dtype=float32)
```

```
label = ['airplane','bridge','palace','ship','stadium']
```

```
label[np.argmax(predict)]
```

```
'airplane'
```

总结:

1.数据载入：ImageDateGenerator

```
train_dir = 'sat2/train'
test_dir = 'sat2/val'

im_size = 224
batch_size = 16
```

```
train_images = ImageDataGenerator(rescale = 1/255,horizontal_flip=True)
test_images = ImageDataGenerator(rescale = 1/255)
```

```
train_gen = train_images.flow_from_directory(directory=train_dir,
                                             batch_size=batch_size,
                                             shuffle=True,
                                             target_size=(im_size, im_size),
                                             class_mode='categorical')
```

2.模型搭建：ReLU&Dropout

```
                                        activation = "relu"))
model.add(tf.keras.layers.Conv2D(filters = 128,
                                 kernel_size = (3,3),
                                 padding = 'same',
                                 activation = "relu"))
model.add(tf.keras.layers.MaxPooling2D(pool_size = (3, 3),strides = 2))
model.add(tf.keras.layers.Flatten())
model.add(tf.keras.layers.Dropout(0.5))
```

3.模型训练：learning_rate & batch_size

```
model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=0.0005),
              loss='categorical_crossentropy',
              metrics=['acc'])
```

参考资料：

1.数据集：遥感影像分类——NWPU-RESISC45
https://aistudio.baidu.com/aistudio/datasetdetail/51873

2. ImageNet Classification with Deep Convolutional Neural Networks
http://www.cs.toronto.edu/~fritz/absps/imagenet.pdf

3. AlexNet网络结构详解与花分类数据集下载
https://www.bilibili.com/video/BV1p7411T7Pc?t=377

4. Dropout: A Simple Way to Prevent Neural Networks from Overfitting
http://www.cs.toronto.edu/~hinton/absps/JMLRdropout.pdf

5.斯坦福大学2014（吴恩达）机器学习教程中文笔记
https://github.com/fengdu78/Coursera-ML-AndrewNg-Notes