Quiz for Deep Learning in Computer Vision

1　简述有监督学习中，分类问题和回归问题的区别。（4 分）

（2 分，意思接近即可得分）分类问题—— label 为离散取值

（2 分，意思接近即可得分）回归问题—— label 为连续型取值

2. 将

-1	-2	-1
0	0	0
1	2	1

卷积核以 stride=1 作用于小灰度图像

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

上，请计算其结果。（3 分）

33	35
28	32

3. 假设运用目标检测算法对某一幅包含 m 只猫的图像进行“猫“检测，并返回 n 个猫的 bounding boxes， 其中只有 k 个 boxes 命中猫。试写出此次目标检测结果的 precision 和 recall 表达式。（6 分）

（3 分）Recall = k / m

（3 分）Precision = k / n

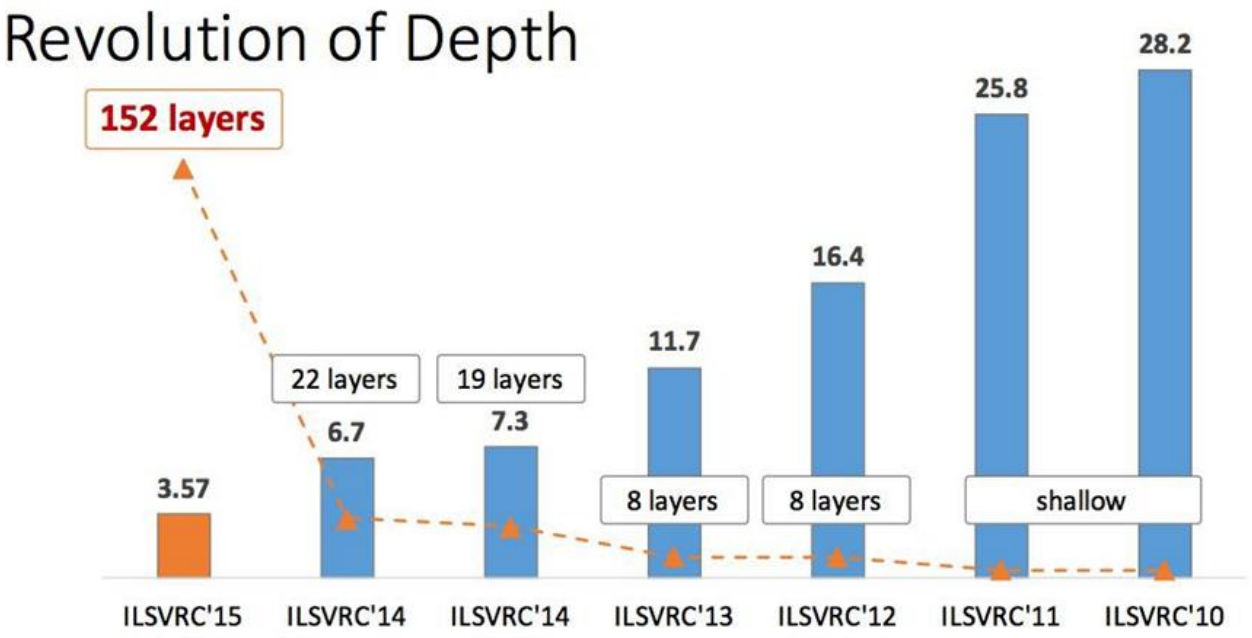
4. 写出图中在 2012 年、2014 年（19 layers）、2014 年（22layers）和 2015 年 ILSVRS 竞赛中获得最好名次的网络模型名称。（4 分）

(1 分) 2012 年——AlexNet

(1 分) 2014 年（19 layers）——VGG

(1 分) 2014 年（22layers）——GoogleNet

(1 分) 2015 年——ResNet



5. 如何计算目标检测中预测的 bounding box 和真实 bounding box 的重合程度？如何计算多目标检测的 mAP 评价指标？（3 分）

(1 分) IoU

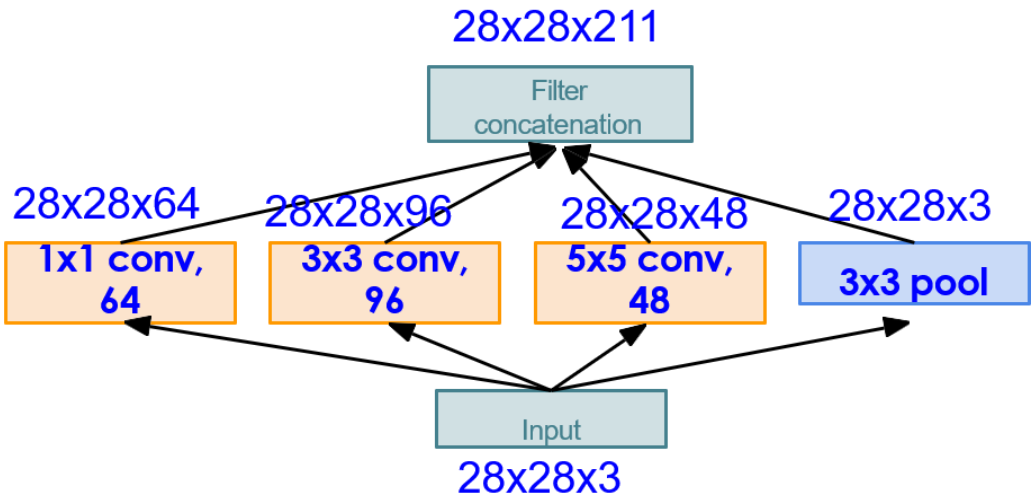
(2 分，意思接近即可得分) 分别为每类检测结果绘制 recall-precision 曲线计算其面积 AP，并平均每类 AP 得到 mAP。

6. 两层 kernel size 为 5×5 的卷积核叠加作用后，与什么尺寸的卷积核具有等效的感受野？ 三层 kernel size 为 3×3 的卷积核叠加作用后，与什么尺寸的卷积核具有等效的感受野?（4 分）

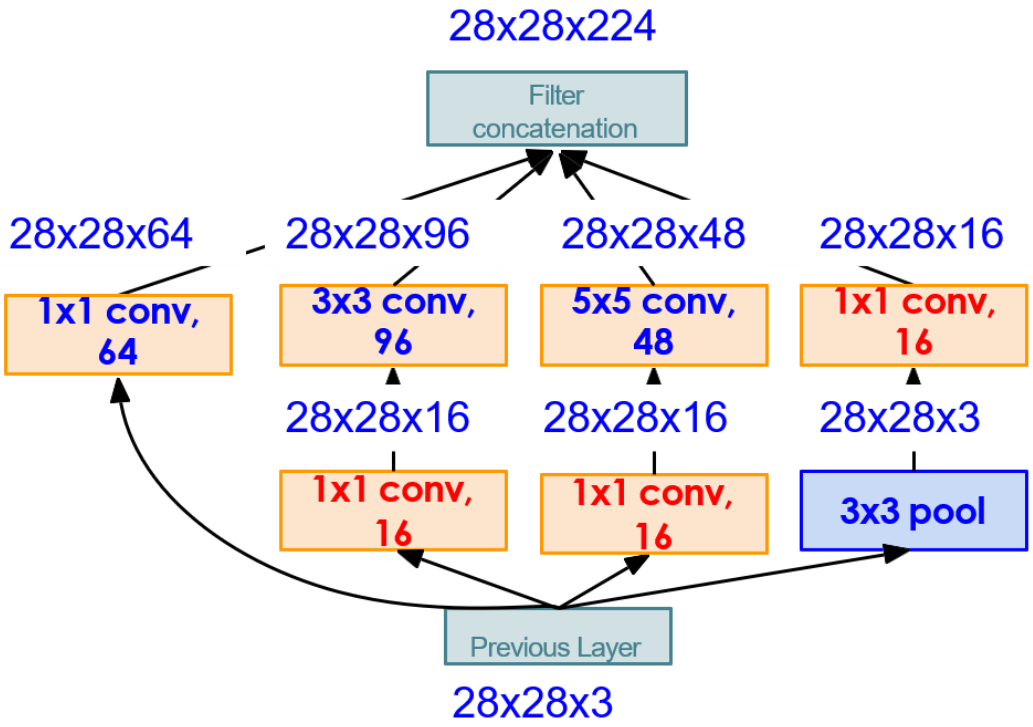
（2 分）9×9

（2 分）7×7

7. 如图所示两种 inception 结构（卷积层中 stride=1，padding = ‘same’），假设输入 shape = 28×28×3 的图像，请分别计算两图中每一个卷积层输出的 feature map 的 shape。（请在答题纸上画出对应结构，并在图上标出计算结果，每层计算结果 2 分，共 26 分）



图一



图二

8.请分别计算 1 个 5×5 卷积层和 2 个堆叠的 3×3 卷积层需要训练的网络权值数（假设在代码中指定卷积层参数 bias = False）（6 分）

（3 分, C 代表输入的通道数，若无此信息不得分）1 个 5×5 卷积层—— 25C

（3 分, C 代表输入的通道数，若无此信息不得分）2 个堆叠的 3×3 卷积层——18C

9. 对一幅 64*64 图像进行目标检测，返回结果是什么形式？若对其进行语义分割，返回结果是什么形式？（4 分）

（2 分，意思接近即可得分）目标检测结果——类别号+bounding box

（2 分，意思接近即可得分）语义分割结果——64*64 的矩阵，其中每一个元素表示原图像相应位置像素的类别号

10. 请简述为何 fast-RCNN/faster-RCNN 中需要进行 Roi pooling 操作（5 分）

（5 分，意思接近即可得分）将不同尺寸的候选框对应的 feature maps 统一尺寸，便于进一步预测

11. 请比较 RCNN，fast-RCNN 和 faster-RCNN 三种目标检测方法中，候选框生成方法的不同之处（9 分）

（3 分，意思接近即可得分）RCNN——在原图像上使用传统图像处理方法搜索候选框，截取候选框子图像

（3 分，意思接近即可得分）fast-RCNN——在原图像上使用传统图像处理方法搜索候选框，在全图 feature maps 上截取候选框子对应的 feature maps

（3 分，意思接近即可得分）faster-RCNN——将全图 feature maps 送入 region proposal network 子网预测候选框

12. 请使用函数方式改写以下使用序列方式定义的卷积网络模型，可使用的函数包括：Input、Conv2D、Activation、Flatten、Dense、Model、compile 等。（10 分）

```
input_shape = (img_width, img_height, 3)

model = Sequential()

model.add(Convolution2D(64, 3, 3, activation='relu'))

model.add(MaxPooling2D((2,2), strides=(2,2)))

model.add(Convolution2D(128, 3, 3, activation='relu'))

model.add(MaxPooling2D((2,2), strides=(2,2)))

model.add(Flatten())

model.add(Dense(256, activation='relu'))

model.add(Dense(1, activation='sigmoid'))

model.compile(loss='binary_crossentropy', optimizer='rmsprop',  metrics=['accuracy'])
```

改写后（所有层都具备，且函数参数设置和返回值均正确即可得分）：

```
input_img = Input (img_width, img_height, 3)
x = Conv2D (64, 3, 3) (input_img)
x = Activation(‘relu’)(x)
x = MaxPooling2D((2,2), strides=(2,2))) (x)
x = Conv2D (128, 3, 3)(x)
x = Activation(‘relu’)(x)
x = MaxPooling2D((2,2), strides=(2,2))(x)
x = Flatten()(x)
x = Dense(256)(x)
x = Activation(‘relu’)(x)
x= Dense(1)(x)
out = Activation(‘'sigmoid')(x)
model = Model(input = input_img, output = out )
model.compile(loss='binary_crossentropy', optimizer='rmsprop',  metrics=['accuracy'])
```

13.请补全下列代码中缺失的部分。（每空 1 分，共 16 分）

```
img_width, img_height = 150, 150
if K.image_data_format() == 'channels_first':
   input_shape = __(3, img_width, img_height)__
else:
   input_shape = __(img_width, img_height, 3)____

model = __Sequential()__________________
model.add(Convolution2D(64, 3, 3, activation='relu'))
model.add(MaxPooling2D((2,2), strides=(2,2)))
```

```
model.add(Convolution2D(512, 3, 3, activation='relu'))
model.add(MaxPooling2D((2,2), strides=(2,2)))

model.add(____Flatten()_____)
model.add(Dense(4096, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(_1 或 2___, activation='__sigmoid 或 softmax_____'))

model.compile(loss='__binary_crossentropy_或__ categorical_crossentropy _',
          optimizer='rmsprop',
          metrics=['accuracy'])
train_data_dir = r'./dogs-vs-cats/train'
validation_data_dir = r'./dogs-vs-cats/validation'
nb_train_samples = 10835
nb_validation_samples = 4000
epochs = 500
batch_size = 20


train_datagen = __ImageDataGenerator___(rescale=1. / 255,
   shear_range=0.2,
   zoom_range=0.2,
   horizontal_flip=True)

# this is the augmentation configuration we will use for testing:
# only rescaling
test_datagen = __ImageDataGenerator___(rescale=1. / 255)

train_generator = train_datagen.flow_from_directory(
```

```
    train_data_dir,
    target_size=(___img_width,    __img_height__)
    batch_size=batch_size,
    class_mode='__binary 或 categorical __')

validation_generator = test_datagen.flow_from_directory(
    validation_data_dir,
    target_size==(___img_width,    __img_height,
    batch_size=batch_size,
    class_mode='_binary 或 categorical___')

model.___fit_generator___(
    train_generator,
    steps_per_epoch=nb_train_samples // batch_size,
    epochs=epochs,
    validation_data=validation_generator,
    validation_steps=nb_validation_samples // batch_size)
```