机器学习

深 度 强 化 学 习

深度学习工具入门简介
(Keras/TensorFlow)

龚平华　　12月22日

滴滴内部
学习资料
请勿外传

扫钉钉群，加入我们

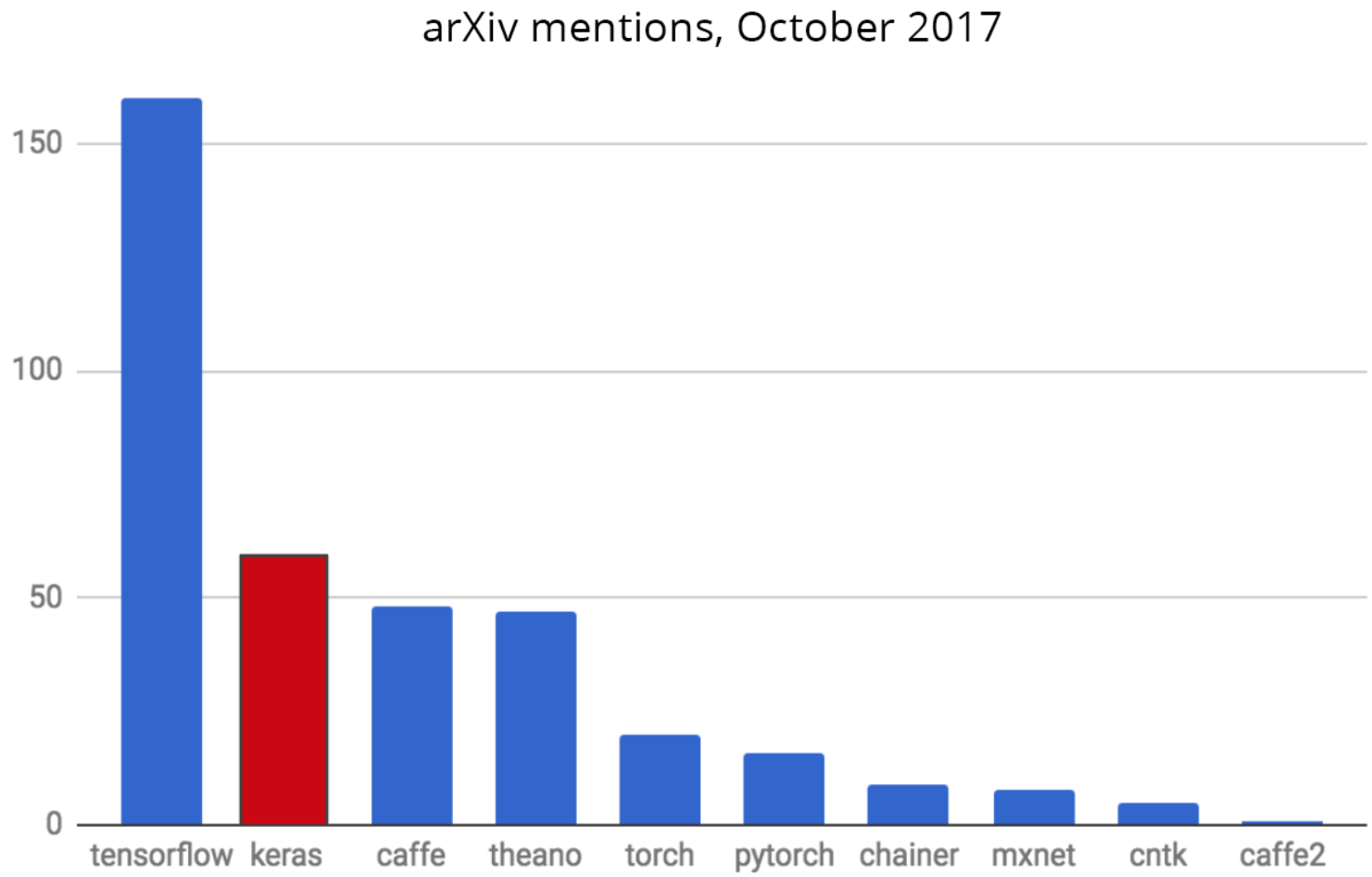
学习受益

- 对TensorFlow/Keras有一个初步的认识
- 能用TensorFlow/Keras构建深度学习网络

内容提纲

- TensorFlow简介和示例
- Keras简介和示例

深度学习工具使用概况


arXiv mentions, October 2017
150
100
50
0
tensorflow
keras
caffe
theano
torch
pytorch
chainer
mxnet
cntk
caffe2

TensorFlow简介

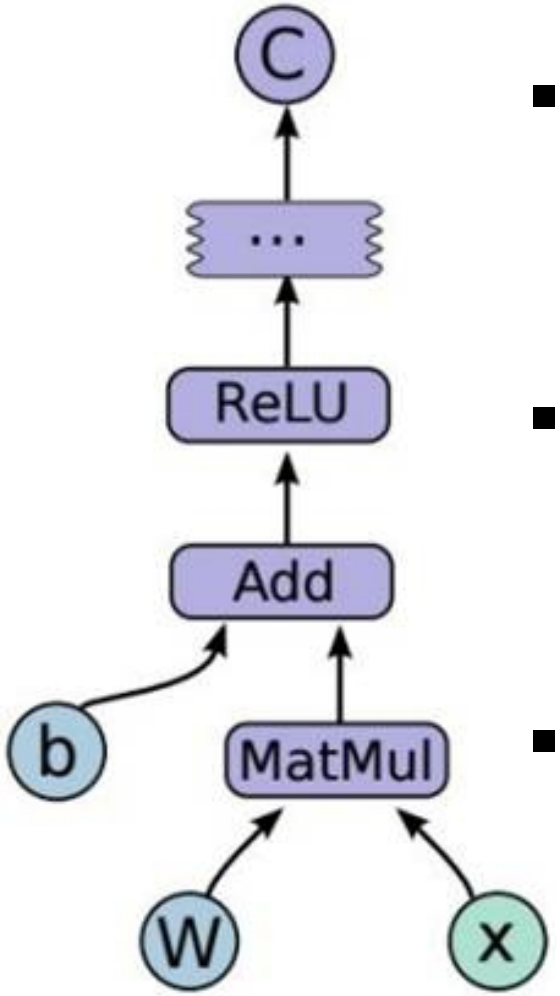
- TensorFlow™ 是一个采用数据流图（data flow graphs），用于数值计算的开源软件库
- TensorFlow可以在多种平台上展开计算，例如台式计算机中的一个或多个CPU（或GPU），服务器，移动设备等等。
- TensorFlow 最初由Google大脑的研究员和工程师们开发出来，用于机器学习和深度神经网络方面的研究，但这个系统的通用性使其也可广泛用于其他计算领域。

以上内容来自TensorFlow中文社区：http://www.tensorfly.cn

TensorFlow基本概念

- 张量 (Tensor)
- 操作 (Operation)
- 会话 (Session)

TensorFlow基本概念



- “节点”(Nodes)一般用来表示施加的数学操作，但也可以表示数据输入的起点，输出的终点，或者是读取/写入变量的起/终点。
- “边”(Edges)表示“节点”之间的输入/输出关系。这些数据“边”可以输运“大小可动态调整”的多维数据数组，即“张量”（tensor）。
- 张量从图中流过的直观图像是这个工具取名为“TensorFlow”的原因

TensorFlow特性

- 高度的灵活性
- 可移植性
- 自动求微分
- 多语言支持
- 性能优化

TensorFlow入门示例


```
import tensorflow as tf
import numpy as np
# Model parameters
W = tf.Variable(tf.zeros([10, 1]), dtype=tf.float32)
b = tf.Variable(tf.zeros([10, 1]), dtype=tf.float32)
# Model input and output
x = tf.placeholder(tf.float32, [None, 10])
linear_model = tf.matmul(x, W) + b
y = tf.placeholder(tf.float32, [None, 1])
```

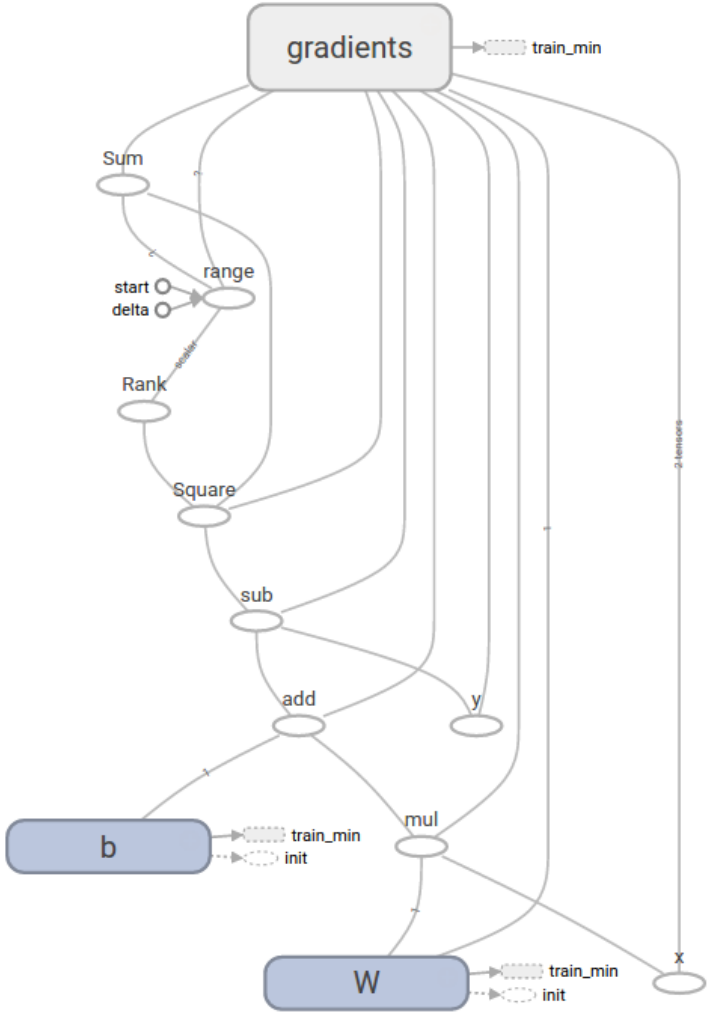
TensorFlow入门示例

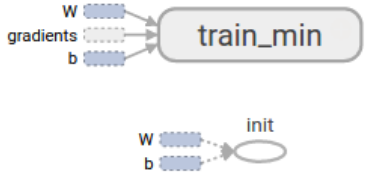

```
# loss and optimizer
loss = tf.reduce_sum(tf.square(linear_model - y)) # sum of the squares
optimizer = tf.train.GradientDescentOptimizer(0.01)
train = optimizer.minimize(loss)

# training data
x_train = np.random.random([10, 10])
y_train = np.random.random([10, 1])
```

TensorFlow入门示例

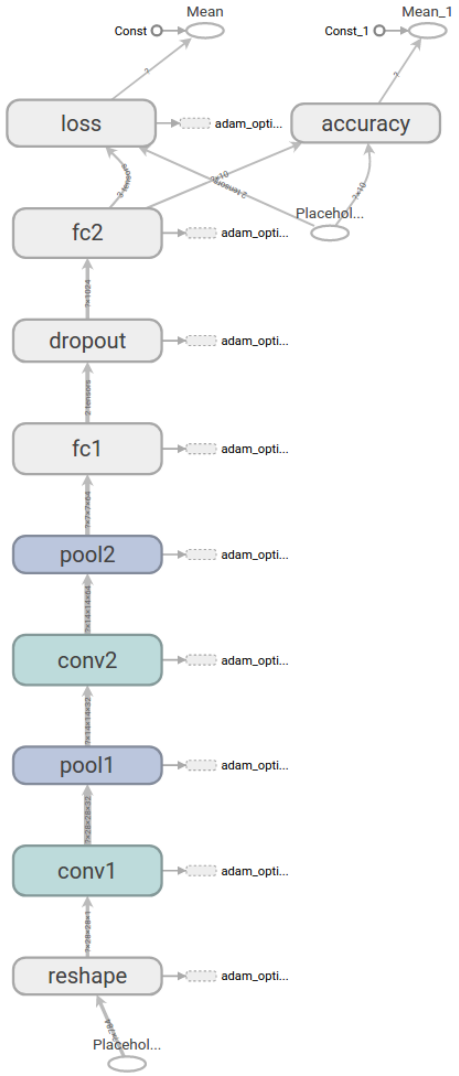

```
# training loop
init = tf.global_variables_initializer()
sess = tf.Session()
sess.run(init) # reset values to wrong
for i in range(1000):
  sess.run(train, {x: x_train, y: y_train})

# evaluate training accuracy
curr_W, curr_b, curr_loss = sess.run([W, b, loss], {x: x_train, y:
y_train})
print("W: %s b: %s loss: %s"%(curr_W, curr_b, curr_loss))
```

TensorFlow入门示例





TensorFlow卷积神经网络例子



TensorFlow卷积神经网络示例


```
def weight_variable(shape):
  initial = tf.truncated_normal(shape, stddev=0.1)
  return tf.Variable(initial)

def bias_variable(shape):
  initial = tf.constant(0.1, shape=shape)
  return tf.Variable(initial)
```

TensorFlow卷积神经网络示例


```
def conv2d(x, W):
  return tf.nn.conv2d(x, W, strides=[1, 1, 1, 1],
padding='SAME')


def max_pool_2x2(x):
  return tf.nn.max_pool(x, ksize=[1, 2, 2, 1],
                        strides=[1, 2, 2, 1], padding='SAME')
```

TensorFlow卷积神经网络示例


```
x = tf.placeholder(tf.float32, [None, 28*28])
x_image = tf.reshape(x, [-1, 28, 28, 1])

W_conv1 = weight_variable([5, 5, 1, 32])
b_conv1 = bias_variable([32])

h_conv1 = tf.nn.relu(conv2d(x_image, W_conv1) + b_conv1)
h_pool1 = max_pool_2x2(h_conv1)
```

TensorFlow卷积神经网络示例


```
W_conv2 = weight_variable([5, 5, 32, 64])
b_conv2 = bias_variable([64])

h_conv2 = tf.nn.relu(conv2d(h_pool1, W_conv2) + b_conv2)
h_pool2 = max_pool_2x2(h_conv2)
```

TensorFlow卷积神经网络示例


```
W_fc1 = weight_variable([7 * 7 * 64, 1024])
b_fc1 = bias_variable([1024])

h_pool2_flat = tf.reshape(h_pool2, [-1, 7*7*64])
h_fc1 = tf.nn.relu(tf.matmul(h_pool2_flat, W_fc1) + b_fc1)

keep_prob = tf.placeholder(tf.float32)
h_fc1_drop = tf.nn.dropout(h_fc1, keep_prob)
```

TensorFlow卷积神经网络示例


```
W_fc2 = weight_variable([1024, 10])
b_fc2 = bias_variable([10])

y_conv = tf.matmul(h_fc1_drop, W_fc2) + b_fc2
```

TensorFlow卷积神经网络示例


```
cross_entropy = tf.reduce_mean(
    tf.nn.softmax_cross_entropy_with_logits(labels=y_, logits=y_conv))
train_step = tf.train.AdamOptimizer(1e-4).minimize(cross_entropy)
correct_prediction = tf.equal(tf.argmax(y_conv, 1), tf.argmax(y_, 1))
accuracy = tf.reduce_mean(tf.cast(correct_prediction, tf.float32))
```

TensorFlow卷积神经网络示例


```
with tf.Session() as sess:
  sess.run(tf.global_variables_initializer())
  for i in range(20000):
    batch = mnist.train.next_batch(50)
    if i % 100 == 0:
      train_accuracy = accuracy.eval(feed_dict={
          x: batch[0], y_: batch[1], keep_prob: 1.0})
      print('step %d, training accuracy %g' % (i, train_accuracy))
    train_step.run(feed_dict={x: batch[0], y_: batch[1], keep_prob: 0.5})
  print('test accuracy %g' % accuracy.eval(feed_dict={
      x: mnist.test.images, y_: mnist.test.labels, keep_prob: 1.0}))
```

TensorFlow Python API r1.4

Keras简介

- Keras是一个高层神经网络API，Keras由Python编写而成并基于TensorFlow, Theano, CNTK后端(Amazon正在开发支持Keras的MXNet后端)
- Keras能支持快速实验
- 高度模块化
- 可扩展性强
- 无缝CPU和GPU切换

Keras简介

- Models
  - About Keras models
  - Sequential
  - Model (functional API)
- Layers
  - About Keras layers
  - Core Layers
  - Convolutional Layers
  - Pooling Layers
  - Locally-connected Layers
  - Recurrent Layers
  - Embedding Layers
  - Merge Layers
  - Advanced Activations Layers
  - Normalization Layers
  - Noise layers
  - Layer wrappers
  - Writing your own Keras layers

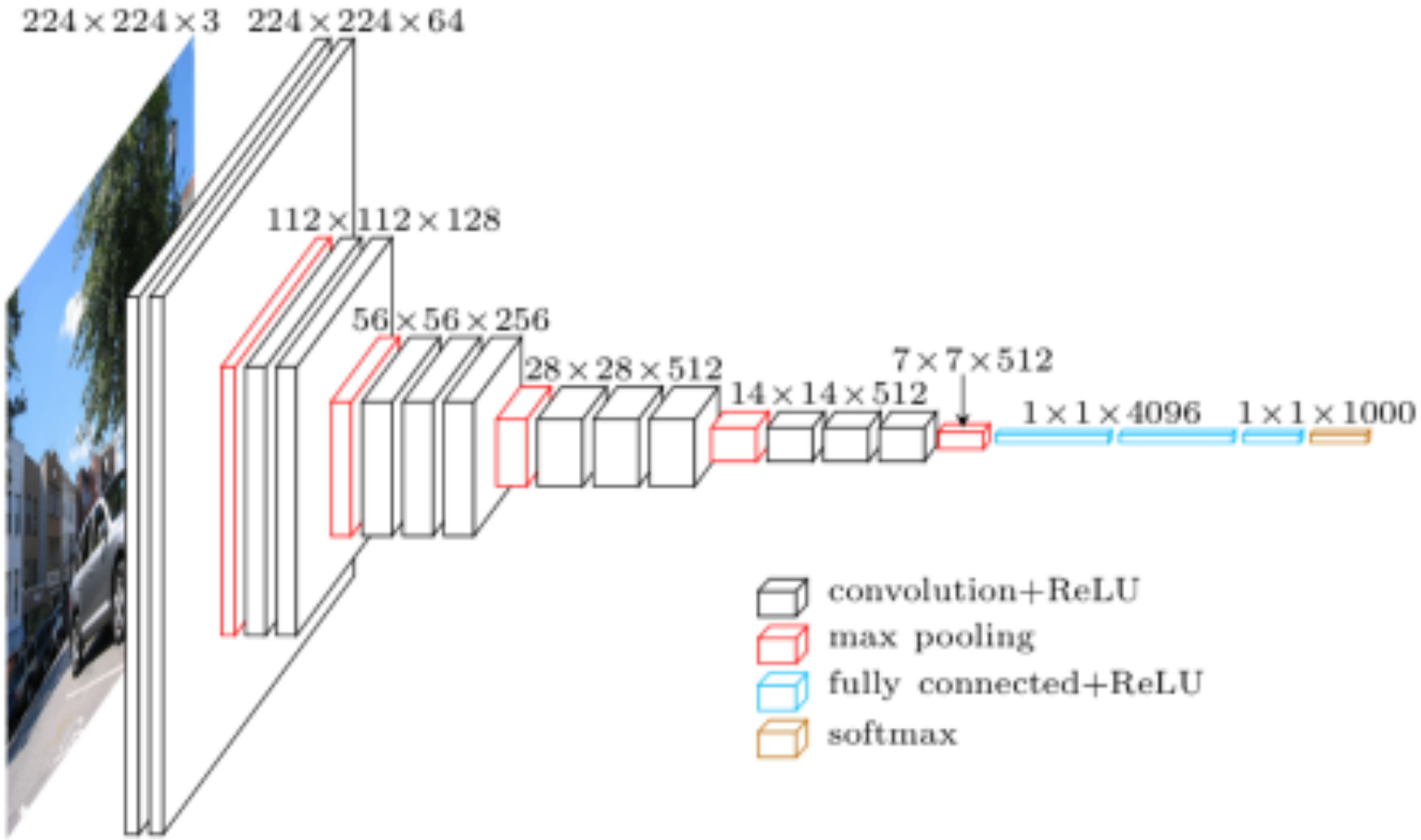
Keras入门示例


```
import numpy as np
from keras.models import Sequential
from keras.layers import Dense

data_train = np.random.random((1000, 100))
labels_train = np.random.randint(2, size=(1000, 1))
data_test = np.random.random((2000, 100))
labels_test = np.random.randint(2, size=(2000, 1))
```

Keras入门示例


```
model = Sequential()
model.add(Dense(32, activation='relu', input_dim=100))
model.add(Dense(1, activation='sigmoid'))
model.compile(optimizer='rmsprop', loss='binary_crossentropy', metrics=['accuracy'])
model.fit(data_train, labels_train, epochs=10, batch_size=32)
loss_and_metrics = model.evaluate(data_test, labels_test, batch_size=128)
print "loss_and_metrics %s" % loss_and_metrics
```

Keras：VGG主要模块示例



Keras：VGG主要模块示例


```
import numpy as np
import keras
from keras.models import Sequential
from keras.layers import Dense, Dropout, Flatten
from keras.layers import Conv2D, MaxPooling2D
from keras.optimizers import SGD
```

Keras：VGG主要模块示例


```
# Generate data
x_train = np.random.random((1000000, 224, 224, 3))
y_train = keras.utils.to_categorical(np.random.randint(1000,
size=(1000000, 1)), num_classes=1000)
x_test = np.random.random((20000, 224, 224, 3))
y_test = keras.utils.to_categorical(np.random.randint(1000,
size=(20000, 1)), num_classes=1000)
```

Keras：VGG主要模块示例


```
model = Sequential()
# input: 224x224 images with 3 channels -> (224, 224, 3) tensors.
# this applies 64 convolution filters of size 3x3 each.
model.add(Conv2D(64, (3, 3), activation='relu', input_shape=(224, 224, 3)))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
```

Keras：VGG主要模块示例


```
model.add(Conv2D(128, (3, 3), activation='relu'))
model.add(Conv2D(128, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Conv2D(256, (3, 3), activation='relu'))
model.add(Conv2D(256, (3, 3), activation='relu'))
model.add(Conv2D(256, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
```

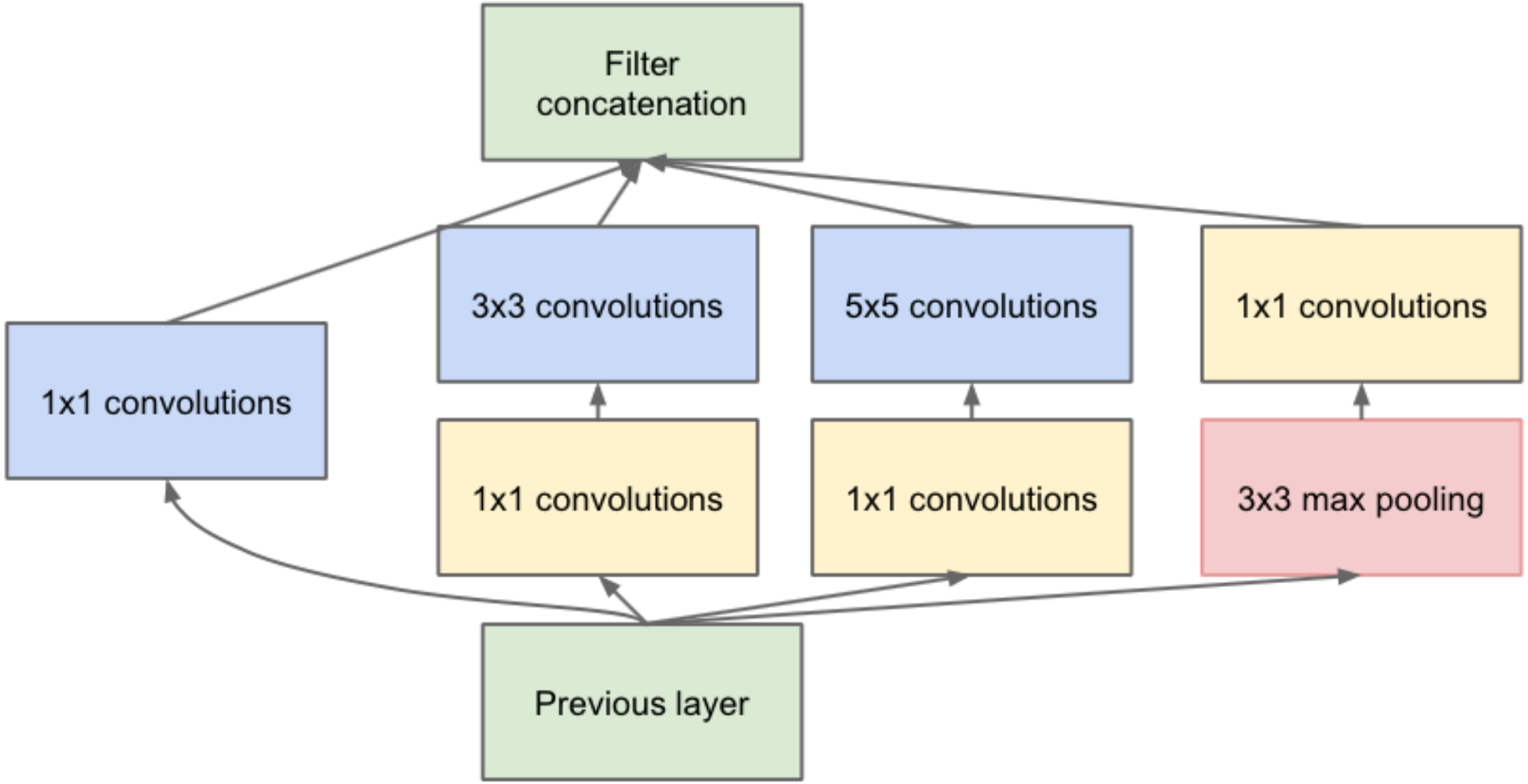
Keras：VGG主要模块示例


```
model.add(Conv2D(512, (3, 3), activation='relu'))
model.add(Conv2D(512, (3, 3), activation='relu'))
model.add(Conv2D(512, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Conv2D(512, (3, 3), activation='relu'))
model.add(Conv2D(512, (3, 3), activation='relu'))
model.add(Conv2D(512, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
```

Keras：VGG主要模块示例


```
model.add(Flatten())
model.add(Dense(4096, activation='relu'))
model.add(Dense(4096, activation='relu'))
model.add(Dense(4096, activation='relu'))
model.add(Dense(1000, activation='softmax'))
sgd = SGD(lr=0.01, decay=1e-6, momentum=0.9, nesterov=True)
model.compile(loss='categorical_crossentropy', optimizer=sgd)
model.fit(x_train, y_train, batch_size=32, epochs=10)
score = model.evaluate(x_test, y_test, batch_size=32)
```

Keras：Inception v1 主要模块示例



Keras：Inception v1 主要模块示例


```
from keras.layers import Conv2D, MaxPooling2D, Input, Dense, Flatten
from keras.models import Model, Sequential
import numpy as np
from keras.optimizers import SGD
import keras
```

Keras：Inception v1 主要模块示例


```
# Generate data
x_train = np.random.random((10, 26, 26, 3))
y_train = keras.utils.to_categorical(np.random.randint(10,
size=(10, 1)), num_classes=10)


x_test = np.random.random((20, 26, 26, 3))
y_test = keras.utils.to_categorical(np.random.randint(10,
size=(20, 1)), num_classes=10)
```

Keras：Inception v1 主要模块示例


```
input_img = Input(shape=(26, 26, 3))

tower_0 = Conv2D(64, (1, 1), padding='same',
activation='relu')(input_img)

tower_1 = Conv2D(64, (1, 1), padding='same',
activation='relu')(input_img)
tower_1 = Conv2D(64, (3, 3), padding='same',
activation='relu')(tower_1)
```

Keras：Inception v1 主要模块示例


```
tower_2 = Conv2D(64, (1, 1), padding='same',
activation='relu')(input_img)
tower_2 = Conv2D(64, (5, 5), padding='same',
activation='relu')(tower_2)


tower_3 = MaxPooling2D((3, 3), strides=(1, 1),
padding='same')(input_img)
tower_3 = Conv2D(64, (1, 1), padding='same',
activation='relu')(tower_3)
output = keras.layers.concatenate([tower_0, tower_1, tower_2,
tower_3], axis=3)
```

Keras：Inception v1 主要模块示例


```
output = Flatten()(output)
output = Dense(10, activation='softmax')(output)
model = Model(inputs = input_img, outputs = output)
sgd = SGD(lr=0.01, decay=1e-6, momentum=0.9, nesterov=True)
model.compile(loss='categorical_crossentropy', optimizer=sgd)
model.fit(x_train, y_train, batch_size=32, epochs=1)
score = model.evaluate(x_test, y_test, batch_size=32)
print 'score: %s' % score
```

THANK YOU

滴滴
滴滴一下 美好出行

www.xiaojukeji.com