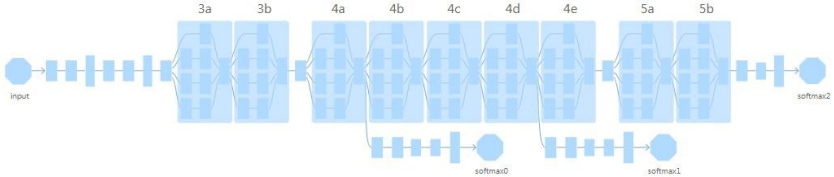
Google TensorFlow™

浙江大学城市学院 ZHEJIANG UNIVERSITY CITY COLLEGE



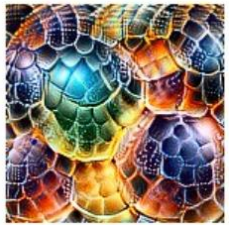

# 深度学习应用开发
## 基于TensorFlow的实践

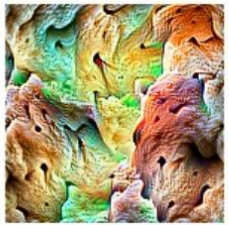
吴明晖 李卓蓉 金苍宏

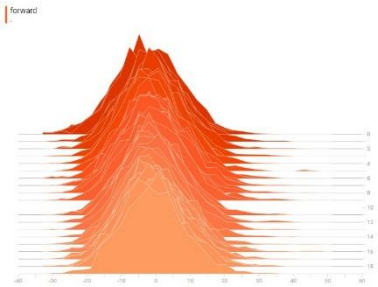
浙江大学城市学院

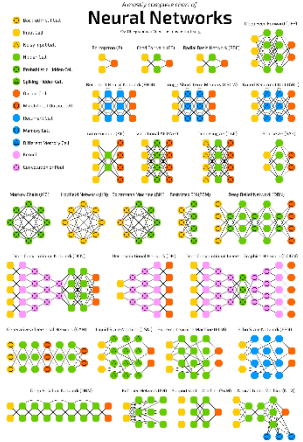
计算机与计算科学学院

Dept. of Computer Science
Zhejiang University City College







# 课程内容安排

# 课程内容安排

# 第一讲 人工智能导论

1. 人工智能 未来已来
2. 人工智能发展史，跌宕起伏的60+年
3. 人工智能、机器学习与深度学习
4. 深度学习框架

# 第二讲 环境搭建和Python快速入门



**环境复杂，我装Anaconda**

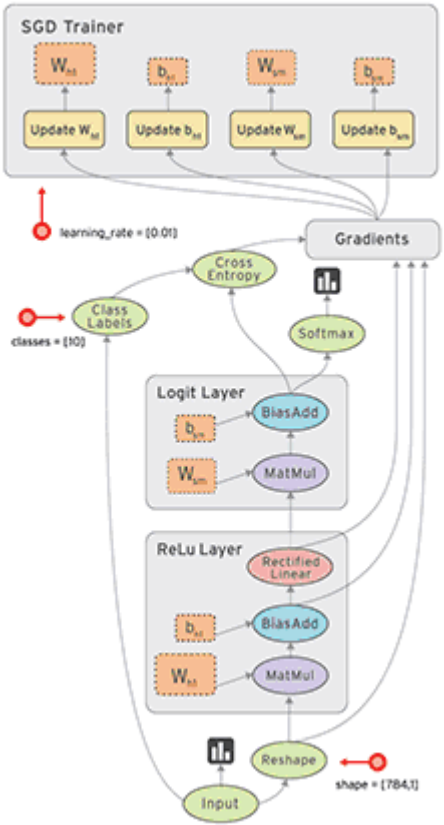
**人生苦短，我用Python**


# 第三讲 TensorFlow 编程基础

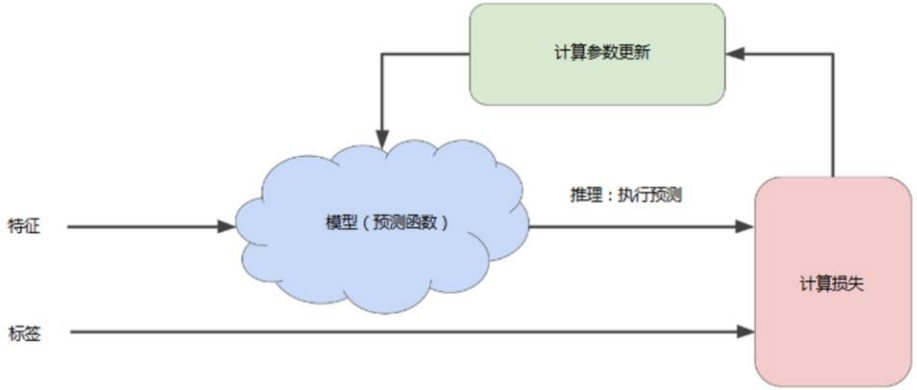

**磨刀不误砍柴工!**

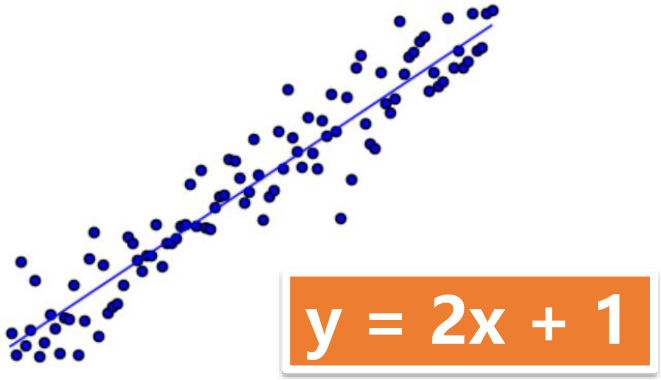
```
import tensorflow as tf

scalar = tf.constant(100)
vector = tf.constant([1, 2, 3, 4, 5])
matrix = tf.constant([[1, 2, 3], [4, 5, 6]])
cube_matrix = tf.constant([[[1], [2], [3]], [[4], [5], [6]], [[7], [8], [9]]])

print(scalar.get_shape())
print(vector.get_shape())
print(matrix.get_shape())
print(cube_matrix.get_shape())
```

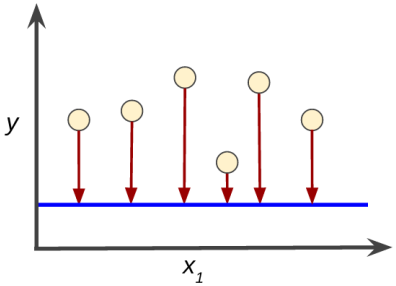
```
()
(5,)
(2, 3)
(3, 3, 1)
```

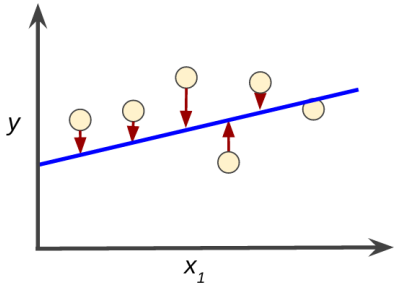


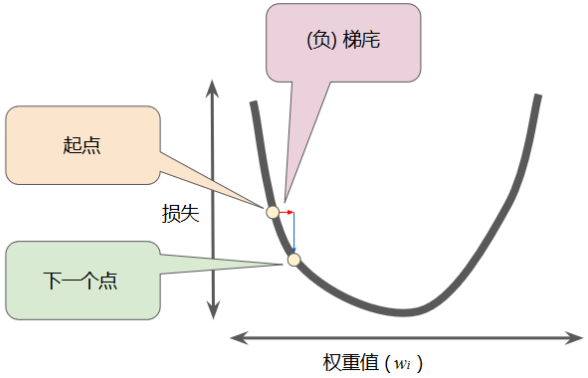
# 第四讲 线性回归问题？用一个神经元搞定！



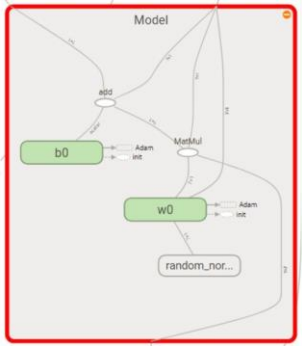
# 第五讲 房价预测问题：多元线性回归及TensorFlow编程进阶

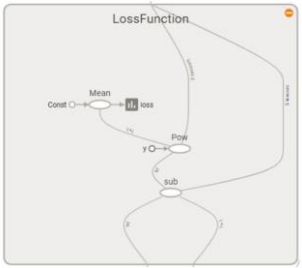
浙江大学城市学院 ZHEJIANG UNIVERSITY CITY COLLEGE

| | CRIM | ZN | INDUS | CHAS | NOX | RM | \ |
|---|---|---|---|---|---|---|---|
| count | 506.000000 | 506.000000 | 506.000000 | 506.000000 | 506.000000 | 506.000000 | |
| mean | 3.613524 | 11.363636 | 11.136779 | 0.069170 | 0.554695 | 6.284634 | |
| std | 8.601545 | 23.322453 | 6.860353 | 0.253994 | 0.115878 | 0.702617 | |
| min | 0.006320 | 0.000000 | 0.460000 | 0.000000 | 0.385000 | 3.561000 | |
| 25% | 0.082045 | 0.000000 | 5.190000 | 0.000000 | 0.449000 | 5.885500 | |
| 50% | 0.256510 | 0.000000 | 9.690000 | 0.000000 | 0.538000 | 6.208500 | |
| 75% | 3.677082 | 12.500000 | 18.100000 | 0.000000 | 0.624000 | 6.623500 | |
| max | 88.976200 | 100.000000 | 27.740000 | 1.000000 | 0.871000 | 8.780000 | |

| | AGE | DIS | RAD | TAX | PTRATIO | LSTAT | \ |
|---|---|---|---|---|---|---|---|
| count | 506.000000 | 506.000000 | 506.000000 | 506.000000 | 506.000000 | 506.000000 | |
| mean | 68.574901 | 3.795043 | 9.549407 | 408.237154 | 18.455534 | 12.653063 | |
| std | 28.148861 | 2.105710 | 8.707259 | 168.537116 | 2.164946 | 7.141062 | |
| min | 2.900000 | 1.129600 | 1.000000 | 187.000000 | 12.600000 | 1.730000 | |
| 25% | 45.025000 | 2.100175 | 4.000000 | 279.000000 | 17.400000 | 6.950000 | |
| 50% | 77.500000 | 3.207450 | 5.000000 | 330.000000 | 19.050000 | 11.360000 | |
| 75% | 94.075000 | 5.188425 | 24.000000 | 666.000000 | 20.200000 | 16.955000 | |
| max | 100.000000 | 12.126500 | 24.000000 | 711.000000 | 22.000000 | 37.970000 | |


**Boston房价预测**



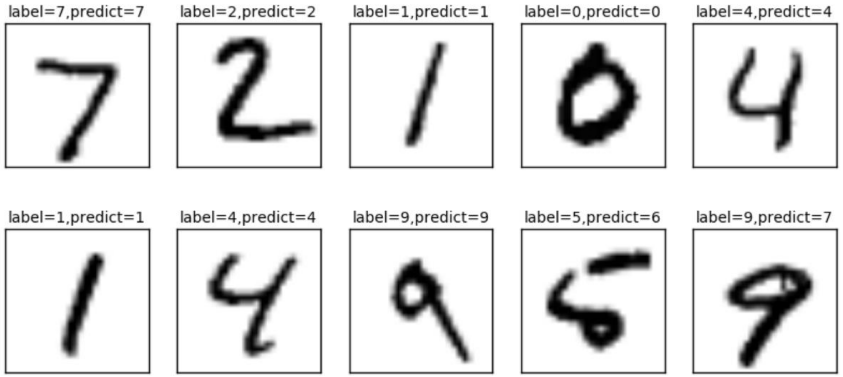




```
epoch= 46 b= 33.2241 w= [[-0.14541905]
 [-0.44425079]
 [-0.81856412]]
38.0116727172
epoch= 47 b= 33.4952 w= [[-0.14832546]
 [-0.47865671]
 [-0.8287307 ]]
37.8562009301
epoch= 48 b= 33.7531 w= [[-0.15111288]
 [-0.51134986]
 [-0.83840436]]
37.7182320901
epoch= 49 b= 33.9985 w= [[-0.15378238]
 [-0.54241031]
 [-0.84760654]]
37.5959297923
epoch= 50 b= 34.2318 w= [[-0.15633498]
 [-0.57191145]
 [-0.85635585]]
```

```
print("y=",w0temp[0], "x1+",w0temp[1], "x2+",w0temp[2], "x3+", [b0temp])
print("y=",w0temp[0], "CRIM+", w0temp[1], 'DIS+', w0temp[2], "LSTAT+", [b0temp])
```

```
y= [-0.15633498] x1+ [-0.57191145] x2+ [-0.85635585] x3+ [34.231792]
y= [-0.15633498] CRIM+ [-0.57191145] DIS+ [-0.85635585] LSTAT+ [34.231792]
```

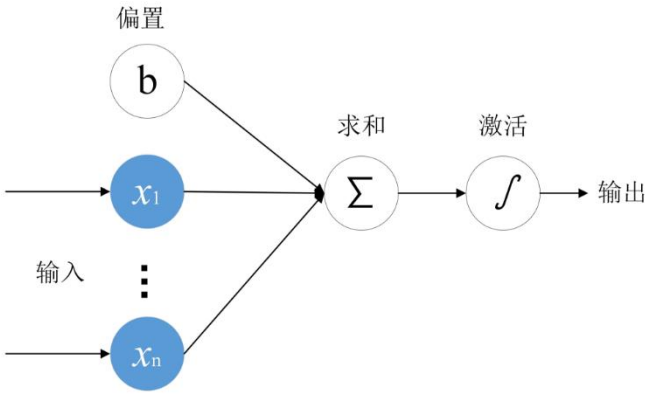
# 第六讲 MNIST手写数字识别：分类应用入门



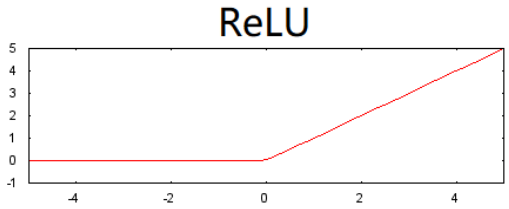

```
from tensorflow.examples.tutorials.mnist import input_data
mnist = input_data.read_data_sets("MNIST_data/", one_hot=True)
```

```
Extracting MNIST_data/train-images-idx3-ubyte.gz
Extracting MNIST_data/train-labels-idx1-ubyte.gz
Extracting MNIST_data/t10k-images-idx3-ubyte.gz
Extracting MNIST_data/t10k-labels-idx1-ubyte.gz
```

**一个神经元处理分类问题**

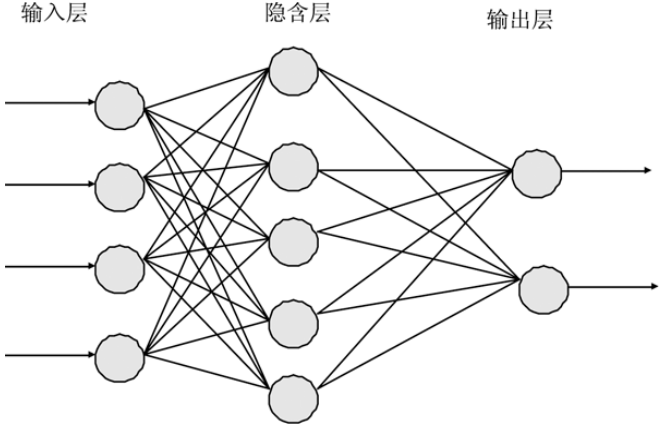



$$\text{output} = f(z) = f(\sum_{i=1}^{n} w_i \times x_i + b)$$

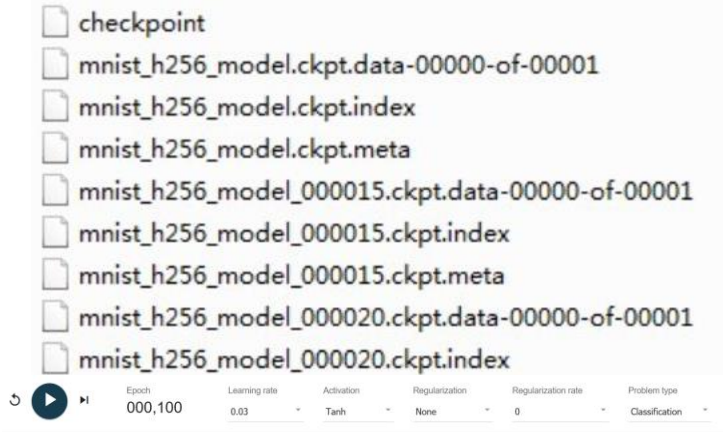


# 第七讲 MNIST手写数字识别进阶：多层神经网络与应用

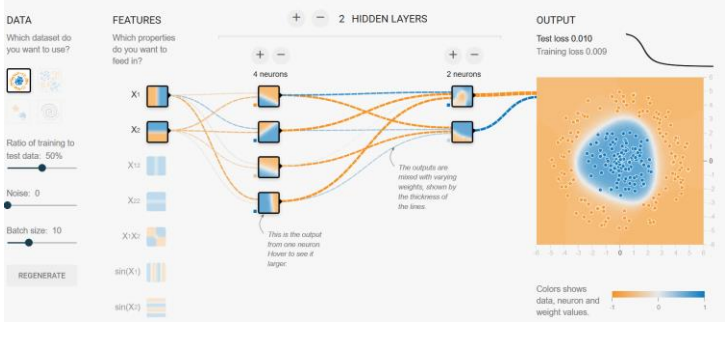
浙江大学城市学院 ZHEJIANG UNIVERSITY CITY COLLEGE




```
saver.save(sess, os.path.join(ckpt_dir, 'mnist_h256_model.ckpt'))#生成检查点文件
print("Model saved!")
```

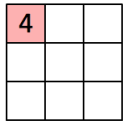

```
def fcn_layer(output_dim,input_dim,inputs, activation=None):
#input_dim为输入神经元数量，output_dim为输出神经元数量
#inputs是输入的二维数组placeholder，activation是激活函数
    W = tf.Variable(tf.random_normal([input_dim, output_dim])) #以正态分布的随机数初始化W
    b = tf.Variable(tf.random_normal([1, output_dim])) #以正态分布的随机数初始化b
    XWb = tf.matmul(inputs, W) + b # 建立表达式：inputs x W + b
    if activation is None: # 默认不使用激活函数
        outputs = XWb
    else: # 若传入激活函数，则用其对输出结果进行变换
        outputs = activation(XWb)
    return outputs
```



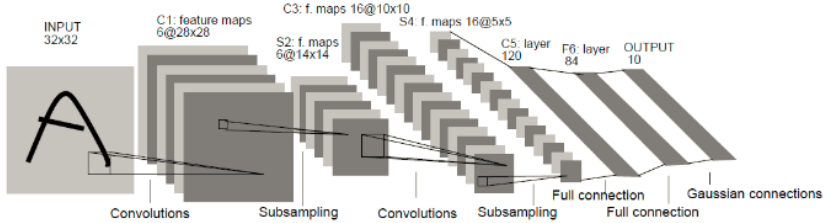
# 第八讲 图像识别问题：卷积神经网络与应用

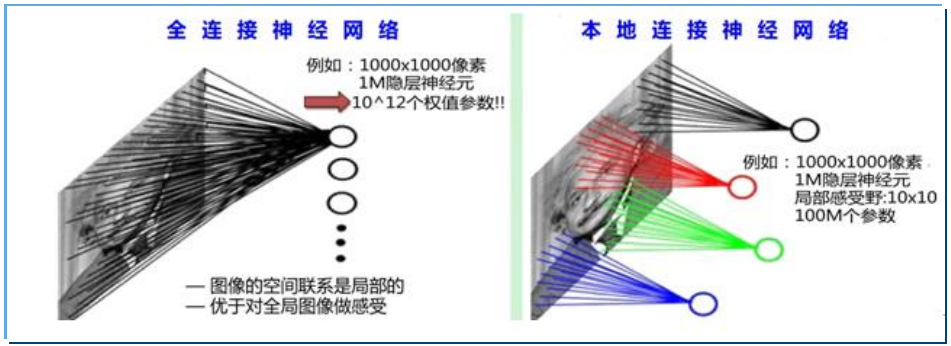

**CIFAR10**

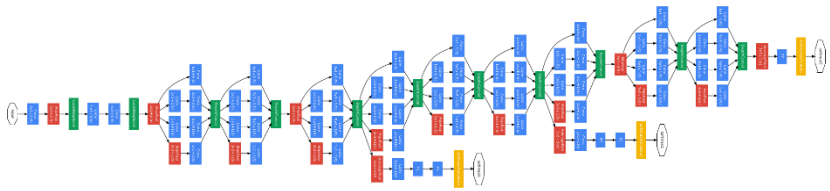

Image




Convolved Feature



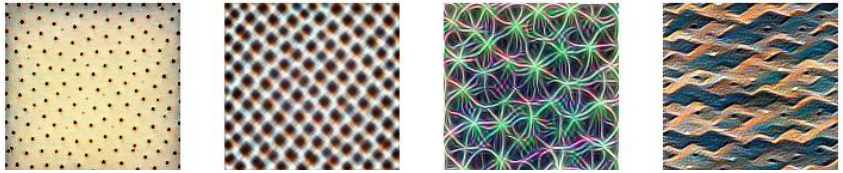



# 第九讲 Deep Dream：理解深度神经网络结构及应用

**VGG、Inception**

Layer 3a

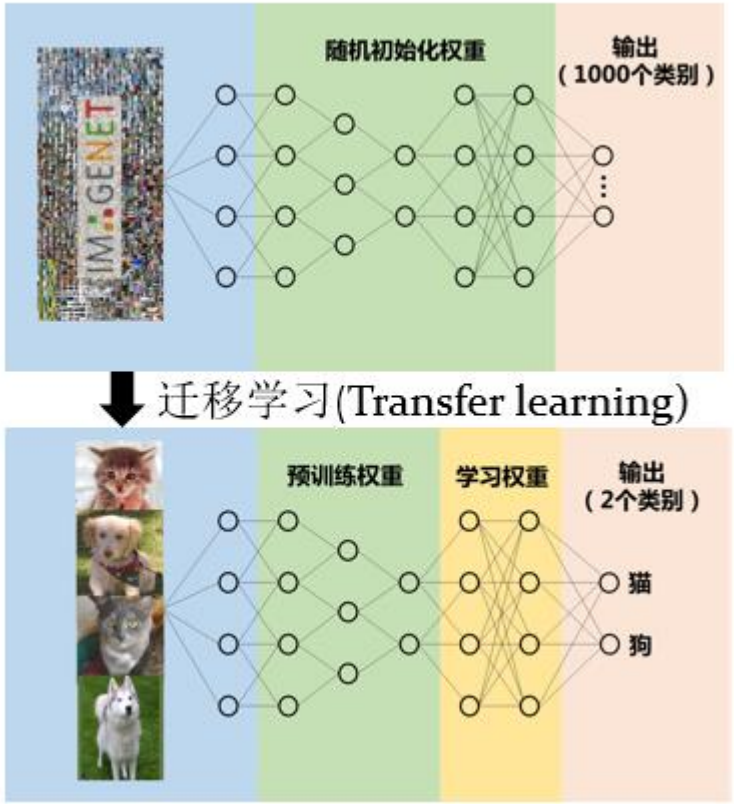


Layer 4a

Layer 4d

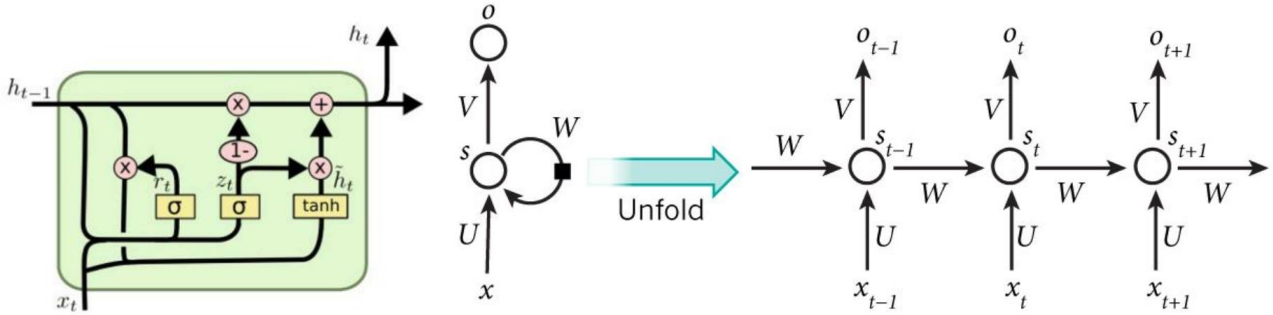
Layer 5a


# 第十讲 猫狗大战：迁移学习及应用




# 第十一讲 电影评论情感分析：RNN循环网络原理及应用

IMDb



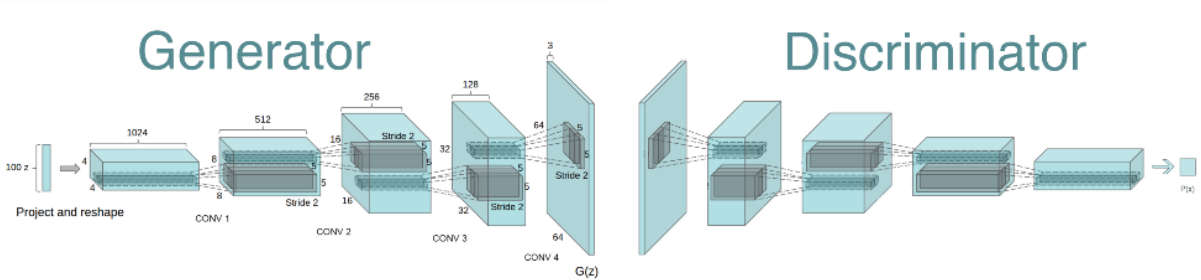


```
predict_review('''
If you let yourself enjoy a live action remake of a classic Picture, you will actually pretty much like (if not love) this movie.
Yes, the CGI isn't first class always and in many cases is a point-to-point copy of the old one, but still The Disney Magic is all over this Fi
A great spectacle and a family movie 100%! ''')
```

```
1/1 [==============================] - 0s
正面的
```

浙江大学城市学院
ZHEJIANG UNIVERSITY CITY COLLEGE

# 第十二讲 给动画自动上色：生成对抗网络原理及应用




浙江大学城市学院 ZHEJIANG UNIVERSITY CITY COLLEGE

# 第十三讲 泰坦尼克号上的旅客生存概率预测：TensorFlow的高级框架Keras

```
In [33]: pd[-2:]
```

Out[33]:

| | survived | name | pclass | sex | age | sibsp | parch | fare | embarked | probability |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 0 | Jack | 3 | male | 23.0 | 1 | 0 | 5.0 | S | 0.150541 |
| **1** | 1 | Rose | 1 | female | 20.0 | 1 | 0 | 100.0 | S | 0.969736 |


| 字段 | 字段说明 | 数据说明 |
|---|---|---|
| pclass | 舱等 | 1 头等舱，2 二等舱，3 三等舱 |
| survival | 是否生存 | 0 否，1 是 |
| name | 姓名 | |
| sex | 性别 | Female 女性，male 男 |
| age | 年龄 | |
| sibsp | 手足或者配偶也在船上的数量 | |
| parch | 双亲或者子女也在船上的数量 | |
| ticked | 船票号码 | |
| fare | 船票费用 | |
| cabin | 舱位号码 | |
| embarked | 登船港口 | C=Cherbourg, Q=Queenstown, S=Southampton |

# 第十四讲 鸢尾花品种识别：TensorFlow.js 应用开发

浙江大学城市学院 ZHEJIANG UNIVERSITY CITY COLLEGE



```
async function trainModel(xTrain, yTrain, xTest, yTest) {
  ui.status('Training model... Please wait.');

  const params = ui.loadTrainParametersFromUI();

  // 模型定义，二层全连接
  const model = tf.sequential();
  model.add(tf.layers.dense(
      {units: 10, activation: 'sigmoid', inputShape: [xTrain.shape[1]]}));
  model.add(tf.layers.dense({units: 3, activation: 'softmax'}));

  const optimizer = tf.train.adam(params.learningRate);
  model.compile({
    optimizer: optimizer,
    loss: 'categoricalCrossentropy',
    metrics: ['accuracy'],
  });
```

鸢尾花分类案例 

鸢尾属（拉丁学名：Iris L.），单子叶植物纲，百合目，鸢尾科多年生草本植物，有块茎或匍匐状根茎；叶剑形，嵌叠状；花美丽，状花序或圆锥花序；花被花瞵状，有一[illegible]颜色，外展而覆盖着雄蕊；子房下位，胚珠多数，果为蒴果。本属模式种：德国鸢尾（iris germanica L.）原产欧洲，中国各地常见栽培。

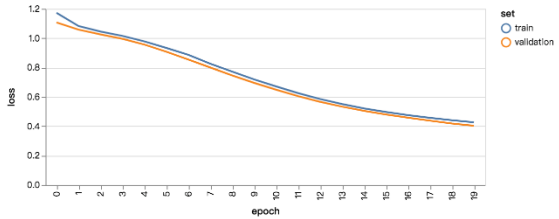
**训练模型**

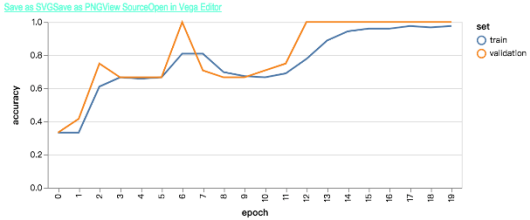
训练轮次： 20

学习率： 0.01

模型训练

Model training complete.



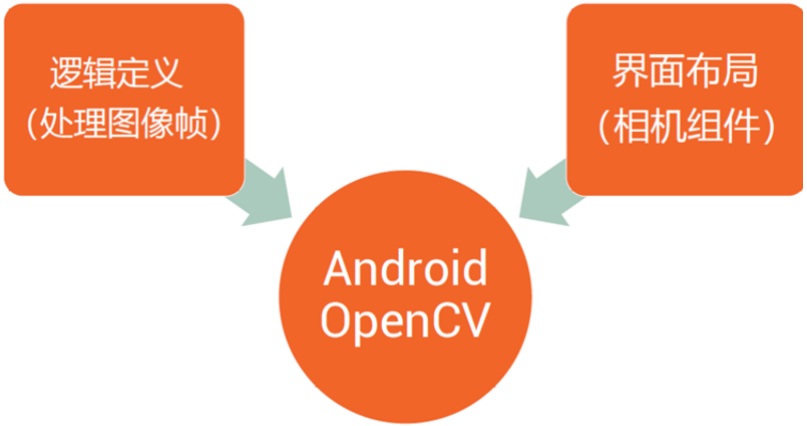



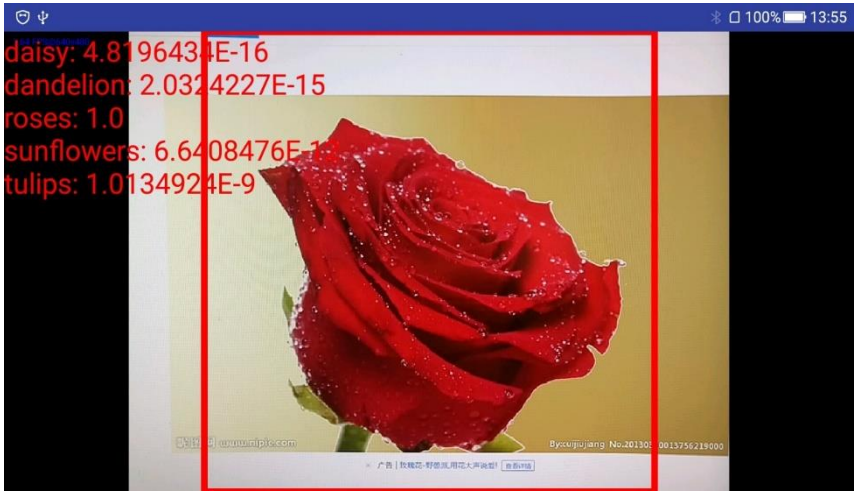

Text Examples

| 花瓣长度 | 花瓣宽度 | 萼片长度 | 萼片宽度 | 真实分类 | 预测分类 | 概率 |
|---|---|---|---|---|---|---|
| 5.1 + - | 3.5 + - | 1.4 + - | 0.2 + - | | Iris-setosa | **0.952** 0.043 0.005 |
| 4.4 | 3.2 | 1.3 | 0.2 | Iris-setosa | Iris-setosa | **0.945** 0.048 0.007 |
| 5.0 | 3.5 | 1.6 | 0.6 | Iris-setosa | Iris-setosa | **0.912** 0.077 0.011 |
| 5.1 | 3.8 | 1.9 | 0.4 | Iris-setosa | Iris-setosa | **0.929** 0.062 0.009 |
| 4.8 | 3.0 | 1.4 | 0.3 | Iris-setosa | Iris-setosa | **0.918** 0.072 0.010 |
| 5.1 | 3.8 | 1.6 | 0.2 | Iris-setosa | Iris-setosa | **0.955** 0.040 0.005 |
| 4.6 | 3.2 | 1.4 | 0.2 | Iris-setosa | Iris-setosa | **0.940** 0.053 0.007 |
| 5.3 | 3.7 | 1.5 | 0.2 | Iris-setosa | Iris-setosa | **0.955** 0.040 0.005 |

# 第十五讲 花卉识别App：TensorFlow Lite与移动应用开发

```
// 初始化 TFLite
public TensorFlowLiteDetector(Map<String, Object> othersMap) {
    try {
        Activity activity = (Activity) othersMap.get("activity");
        tfLite = new Interpreter(loadModelFile(activity));
        labelList = loadLabelList(activity);
        imgData = ByteBuffer.allocateDirect(
                4 * DIM_BATCH_SIZE * DIM_IMG_SIZE_X * DIM_IMG_SIZE_Y * DIM_PIXEL_SIZE);
        imgData.order(ByteOrder.nativeOrder());
        labelProbArray = new float[1][labelList.size()];
        filterLabelProbArray = new float[FILTER_STAGES][labelList.size()];
    } catch (IOException e) {
        e.printStackTrace();
    }
}
```



# 第十六讲 课程回顾与大作业

## Final Project

Talk is cheap
show me the code