TensorBoard应用进阶

TensorBoard

构建输入层

```
x = tf.placeholder(tf.float32, [None, 784], name="X")
```

```
image_shaped_input = tf.reshape(x, [-1, 28, 28, 1])
tf.summary.image('input', image_shaped_input, 10)
```

```
<tf.Tensor 'input:0' shape=() dtype=string>
```

输入的样本图像加入summary

TensorBoard

TensorBoard：前向输出值在以直方图显示

```
tf.summary.histogram('forward',forward)
```

```
<tf.Tensor 'forward:0' shape=() dtype=string>
```

定义好前向计算操作后就可以加入summary

TensorBoard

TensorBoard: 将loss损失以标量显示

```
tf.summary.scalar('loss', loss_function)
```

```
<tf.Tensor 'loss:0' shape=() dtype=string>
```

定义好loss_function操作后就可以加入summary

TensorBoard

TensorBoard: 将accuracy准确率以标量显示

```
tf.summary.scalar('accuracy', accuracy)
```

```
<tf.Tensor 'accuracy:0' shape=() dtype=string>
```

定义好accuracy操作后就可以加入summary

TensorBoard

训练模型

```
sess = tf.Session()
sess.run(tf.global_variables_initializer())
```

TensorBoard: 合并所有summary

```
merged_summary_op = tf.summary.merge_all()
writer = tf.summary.FileWriter('log/', sess.graph) #创建写入符
```

准备开始训练，合并所以summary，写入计算图

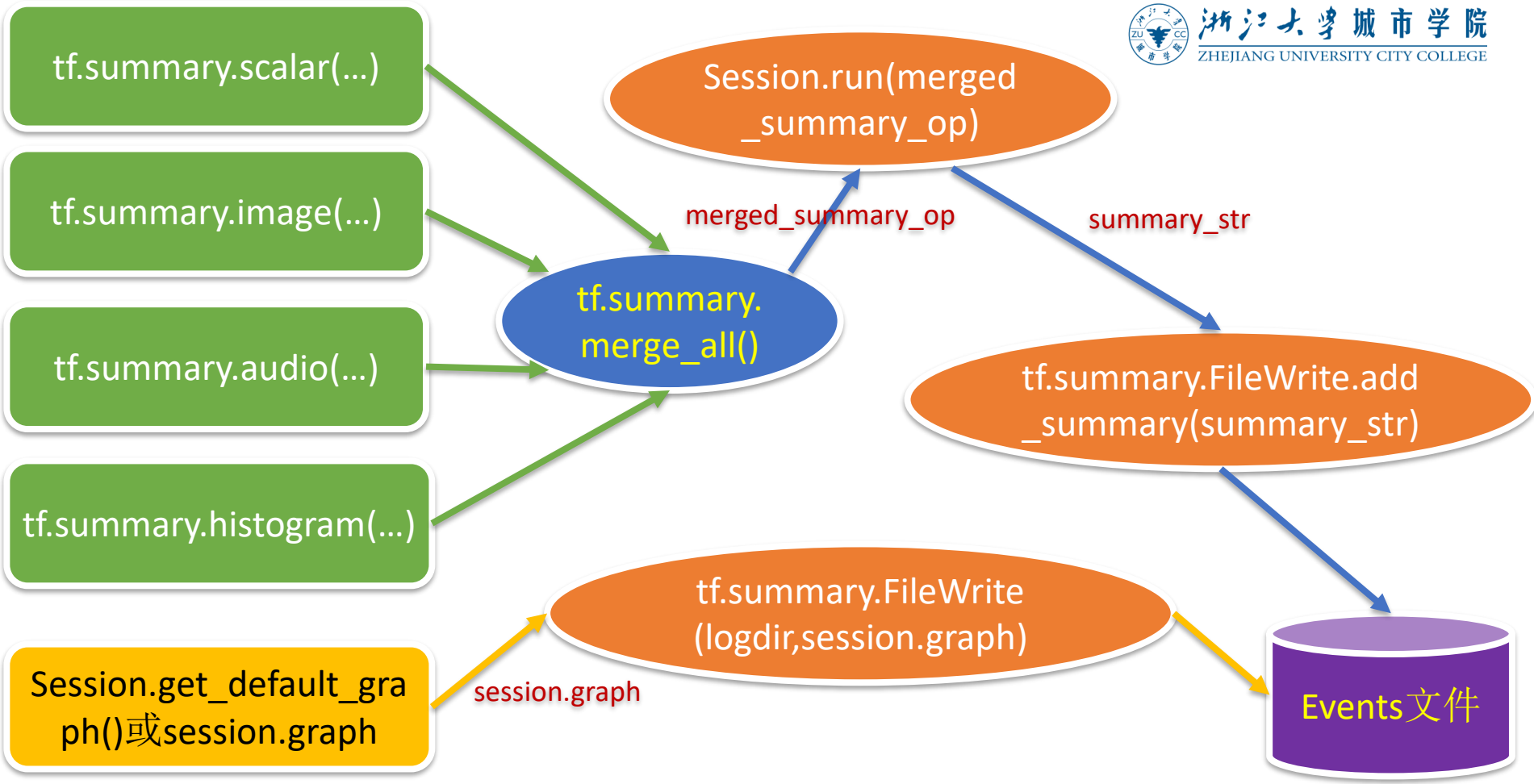
TensorBoard

```
for epoch in range(train_epochs):
    for batch in range(total_batch):
        xs, ys = mnist.train.next_batch(batch_size)# 读取批次数据
        sess.run(optimizer,feed_dict={x: xs,y: ys}) # 执行批次训练

        #生成summary
        summary_str = sess.run(merged_summary_op,feed_dict={x: xs,y: ys})
        writer.add_summary(summary_str, epoch) # 将summary 写入文件

    #total_batch个批次训练完成后，使用验证数据计算误差与准确率
    loss,acc = sess.run([loss_function,accuracy],
                        feed_dict={x: mnist.validation.images,
                                   y: mnist.validation.labels})
```

训练中，生成每次要写入的summary信息，加入summary


tf.summary.scalar(...)
tf.summary.image(...)
tf.summary.audio(...)
tf.summary.histogram(...)
Session.get_default_graph()或session.graph
tf.summary.merge_all()
merged_summary_op
Session.run(merged_summary_op)
summary_str
tf.summary.FileWrite.add_summary(summary_str)
session.graph
tf.summary.FileWrite(logdir,session.graph)
Events文件

TensorBoard


```
(C:\Users\mingh\Anaconda3) C:\Users\mingh\Documents\TensorFlowCodes\TF_ZUCC_6_MNIST_H256>tensorboard --logdir=.\log\
Starting TensorBoard b'54' at http://MinghuiWU:6006
(Press CTRL+C to quit)
```

进入日志所在目录，启动tensorboard，本例采用相对目录

Fit to screen

Download PNG

Run (1) .

Session runs (0)

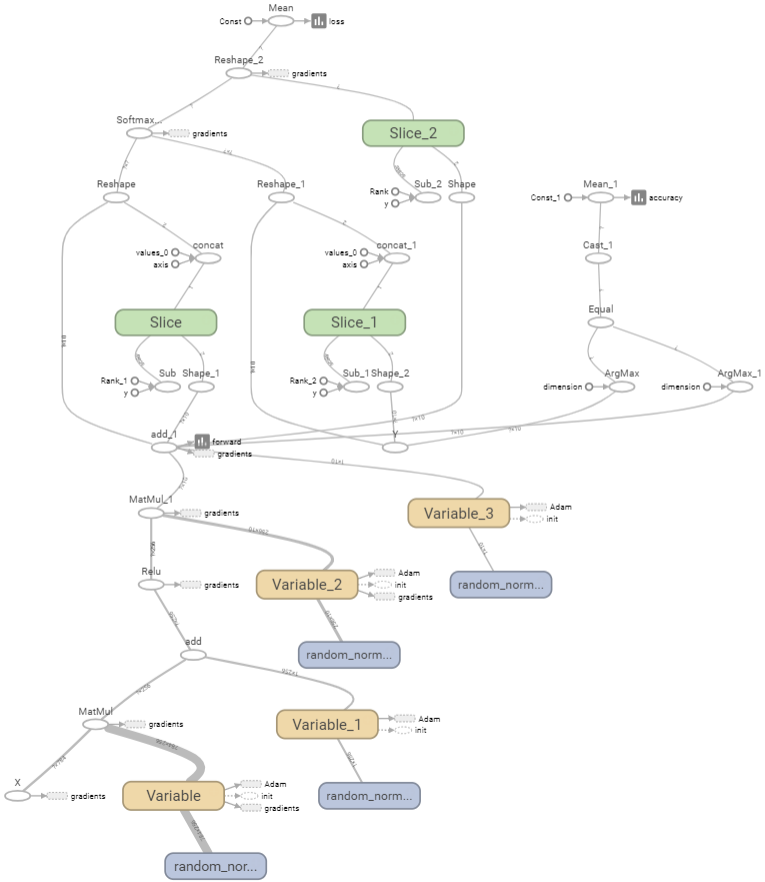
Upload Choose File

Trace inputs

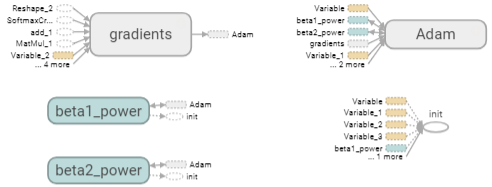
Color
- Structure
- Device
- XLA Cluster
- Compute time
- Memory

Graph (* = expandable)
- Namespace*
- OpNode
- Unconnected series*
- Connected series*
- Constant
- Summary
- Dataflow edge
- Control dependency edge
- Reference edge

Main Graph



Auxiliary Nodes



TensorBoard SCALARS IMAGES AUDIO GRAPHS DISTRIBUTIONS HISTOGRAMS EMBEDDINGS TEXT

Write a regex to create a tag group

Runs

Write a regex to filter runs

.

TOGGLE ALL RUNS

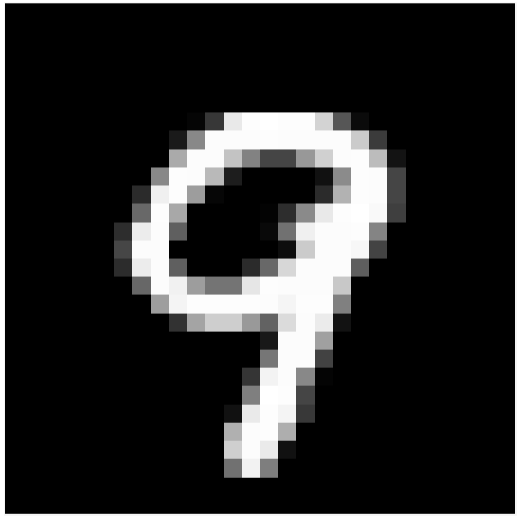
./log/

input 10

input/image/0

.

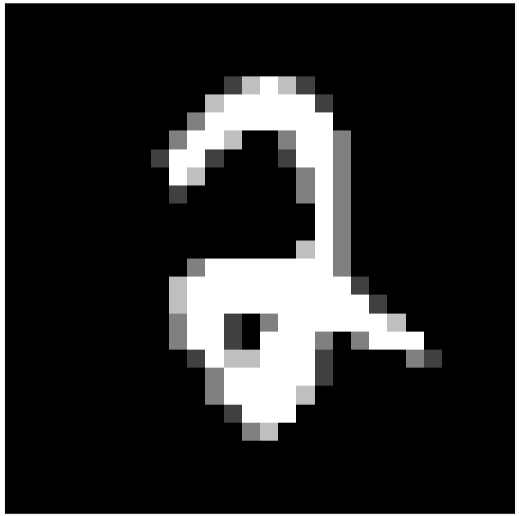
step **39** (Sun Nov 11 2018 08:42:49 GMT+0800 (中国标准时间))



input/image/1

.

step **39** (Sun Nov 11 2018 08:42:49 GMT+0800 (中国标准时间))



input/image/2

.

step **39** (Sun Nov 11 2018 08:42:49 GMT+0800 (中国标准时间))

input/image/3

.

step **39** (Sun Nov 11 2018 08:42:49 GMT+0800 (中国标准时间))

TensorBoard SCALARS IMAGES AUDIO GRAPHS DISTRIBUTIONS HISTOGRAMS EMBEDDINGS TEXT

Write a regex to create a tag group

Show data download links

Ignore outliers in chart scaling

Tooltip sorting method: default

Smoothing

0.6

Horizontal Axis

STEP RELATIVE WALL

Runs

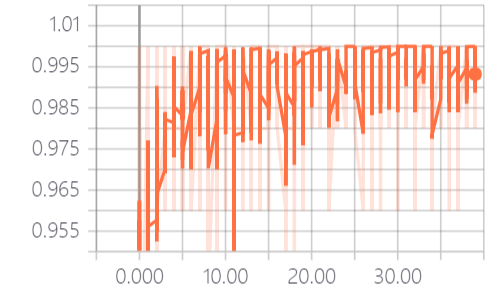
Write a regex to filter runs

TOGGLE ALL RUNS

.\log\

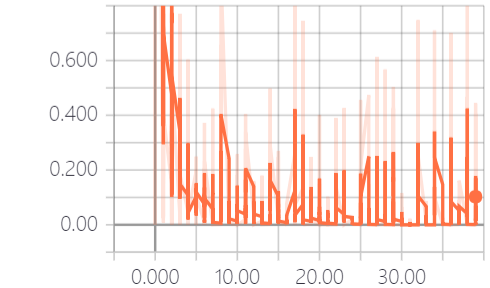
accuracy 1

accuracy



loss 1

loss



TensorBoard SCALARS IMAGES AUDIO GRAPHS DISTRIBUTIONS HISTOGRAMS EMBEDDINGS TEXT

Write a regex to create a tag group

Histogram Mode

OVERLAY OFFSET

Offset Time Axis

STEP RELATIVE WALL

Runs

Write a regex to filter runs

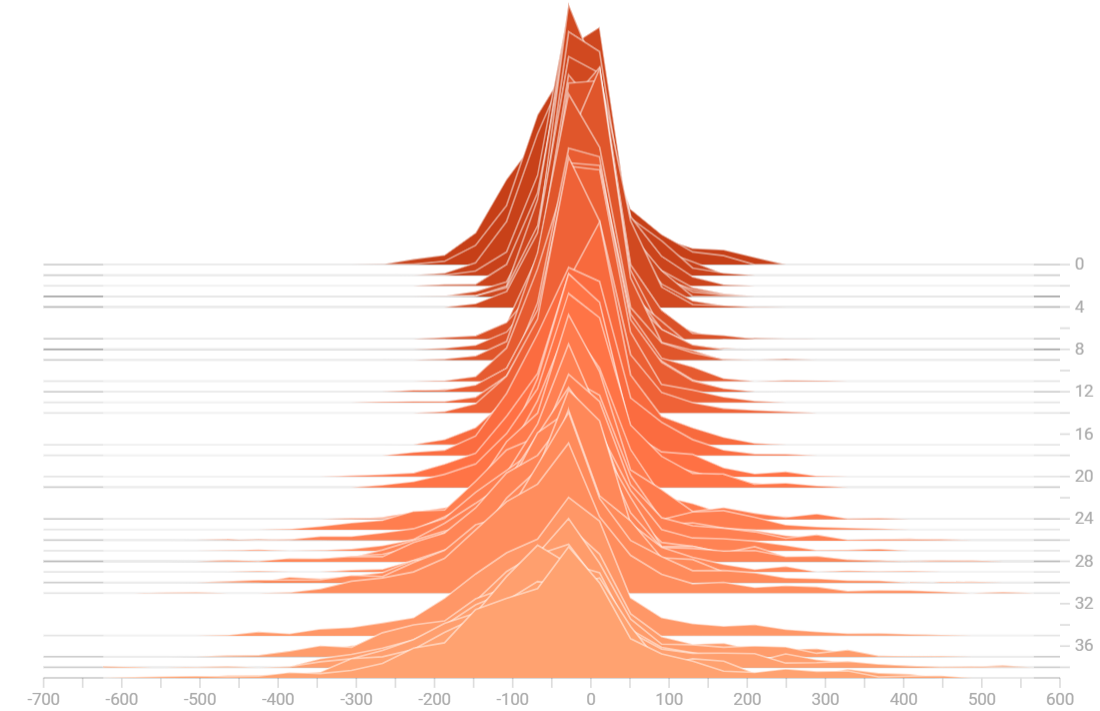
.

TOGGLE ALL RUNS

.\log\

forward 1

forward
.



TensorBoard SCALARS IMAGES AUDIO GRAPHS DISTRIBUTIONS HISTOGRAMS EMBEDDINGS TEXT

Write a regex to create a tag group

Horizontal Axis

STEP RELATIVE WALL

Runs

Write a regex to filter runs

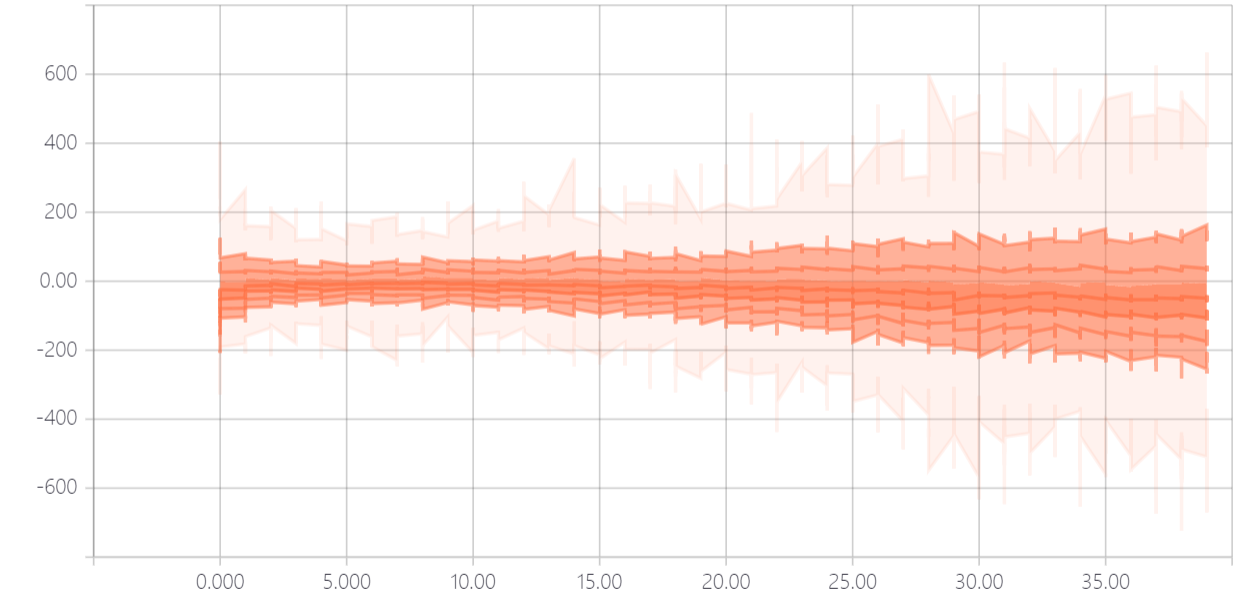
.

TOGGLE ALL RUNS

.\log\

forward 1

forward
.

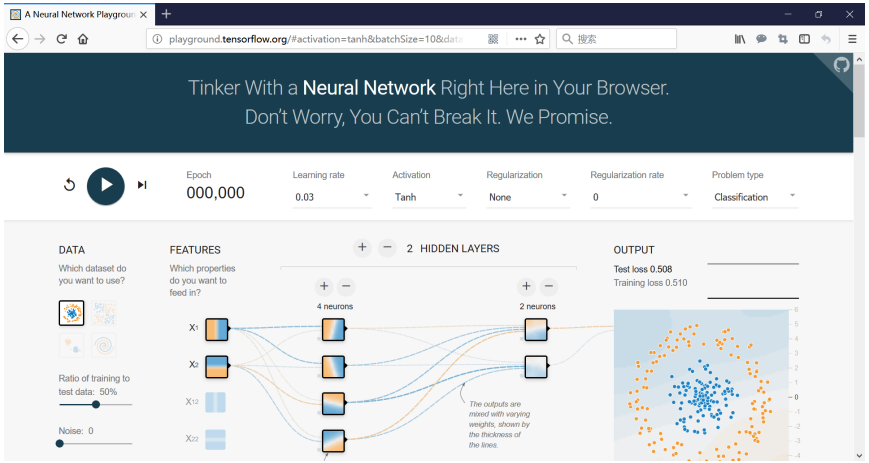


TensorFlow游乐场

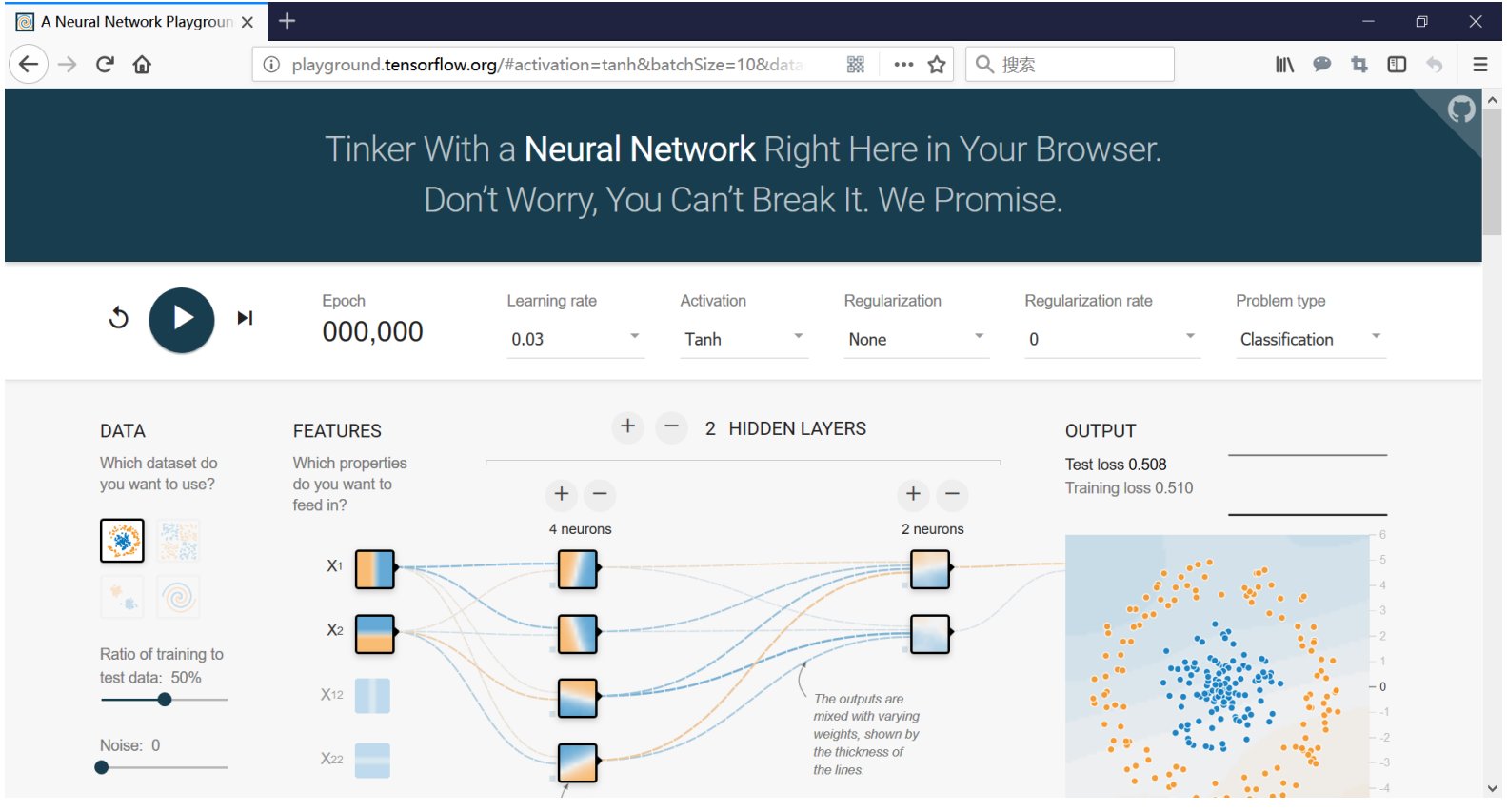
TensorFlow游乐场

TensorFlow游乐场是一个通过网页浏览器就可以训练的简单神经网络并实现了可视化训练过程

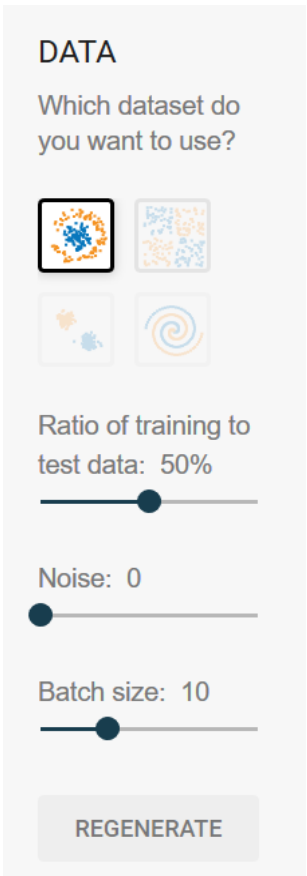
网址：http://playground.tensorflow.org

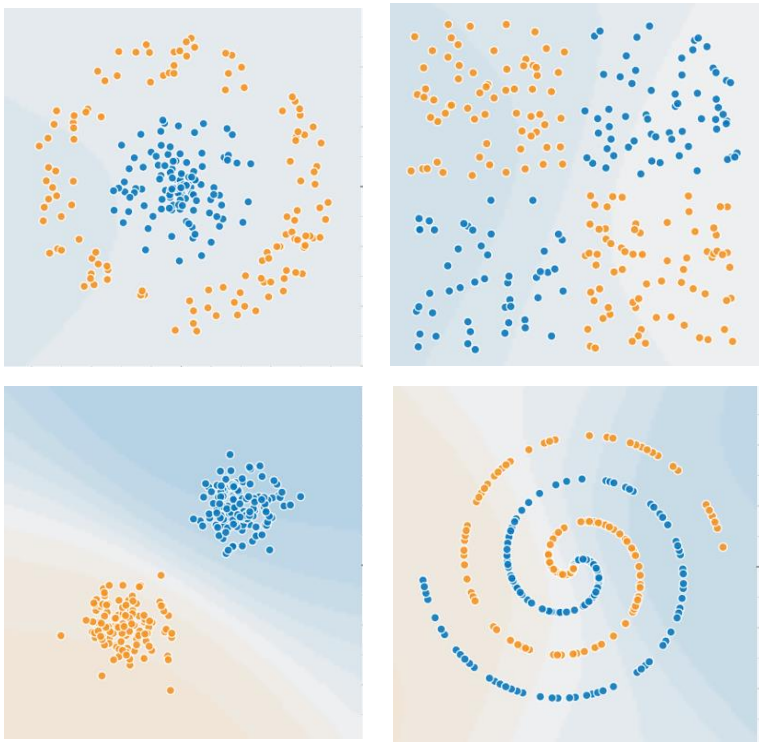


Playground


A Neural Network Playgroun
playground.tensorflow.org/#activation=tanh&batchSize=10&data
搜索
Tinker With a Neural Network Right Here in Your Browser.
Don't Worry, You Can't Break It. We Promise.
Epoch
000,000
Learning rate
0.03
Activation
Tanh
Regularization
None
Regularization rate
0
Problem type
Classification
DATA
Which dataset do you want to use?
Ratio of training to test data: 50%
Noise: 0
FEATURES
Which properties do you want to feed in?
X1
X2
X12
X22
2 HIDDEN LAYERS
4 neurons
2 neurons
The outputs are mixed with varying weights, shown by the thickness of the lines.
OUTPUT
Test loss 0.508
Training loss 0.510

数据





每个点代表一个样本

点的颜色代表样本标签

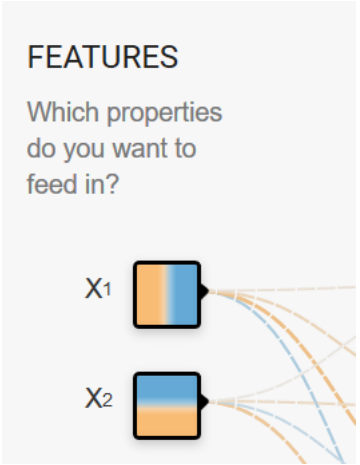
提供了4种不同的数据集，可以设置训练数据比例、噪音、批处理大小

特征提取

为了将一个实际问题对应到平面上不同颜色点的划分，需要将实际问题种的实体变成平面上的一个点；点的颜色只有两种，这是一个二分类的问题

以判断零件是否合格为例，**长度**和**质量**就是**特征**

在机器学习中，所有用于描述实体的数字的组合就是一个实体的**特征向量**（feature vector）



X1: 代表**长度**

X2: 代表**质量**

神经网络

神经网络是分层的机构

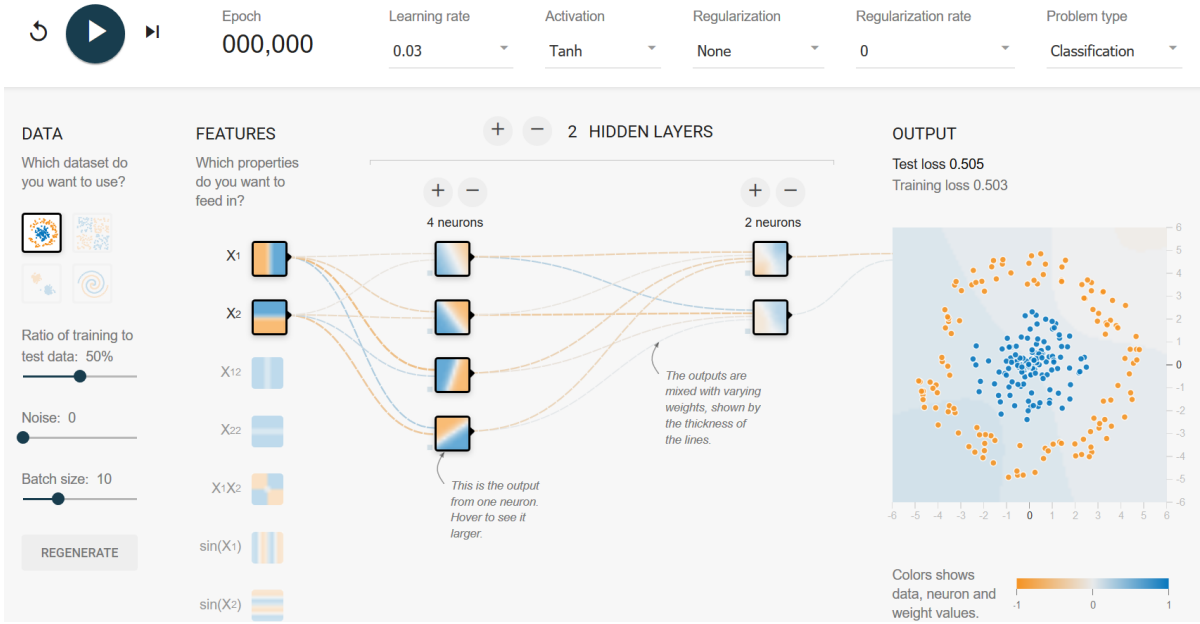
特征向量是神经网络的输入层

同一层的节点不会互相连接

每一层只和下一层连接

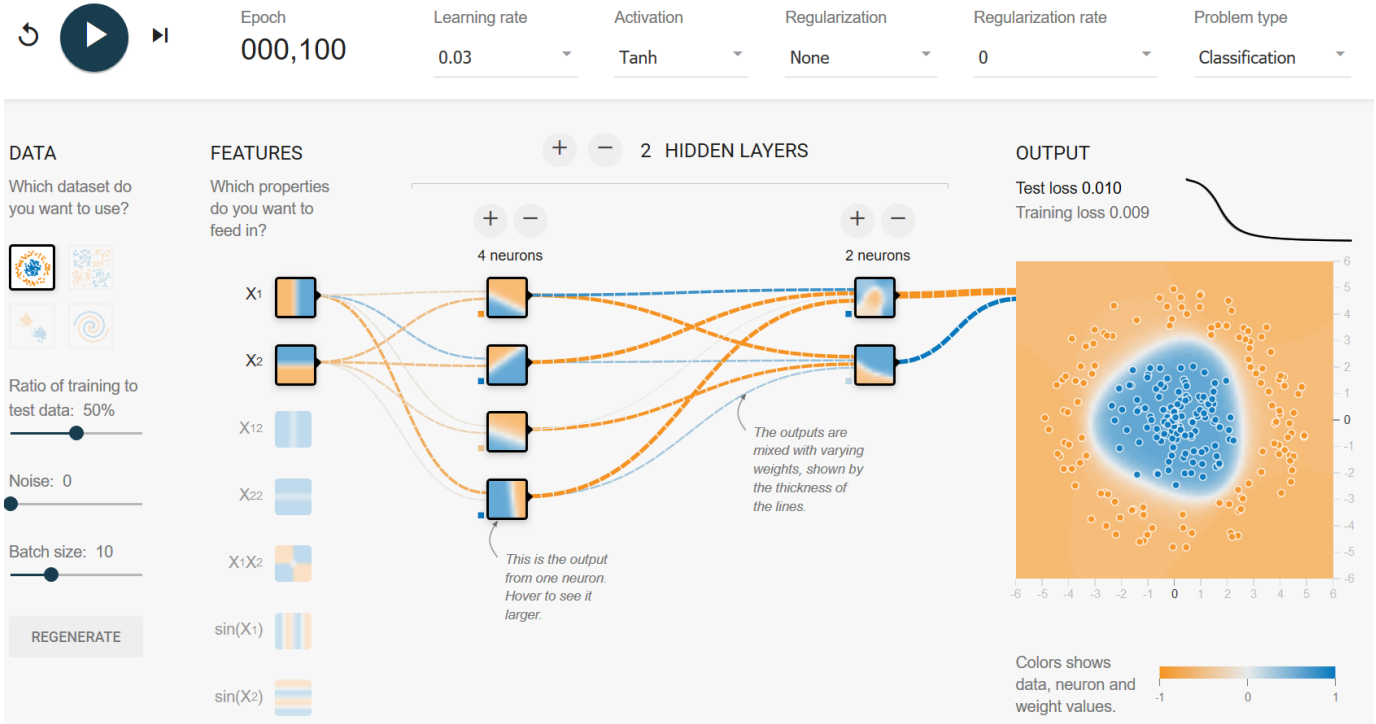
最后一层作为输出层得到结果

输入和输出层之间的是隐藏层



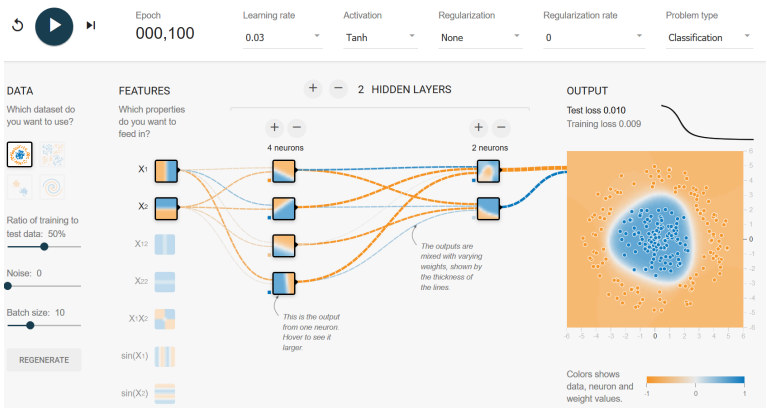
学习率（learning rate）、激活函数（activation）、正则化（regularization）

神经网络训练



训练100轮的结果

游乐场的神经网络训练解读



一个小格子代表神经网络中的一个节点

边代表节点之间的连接

节点和边都有或深或浅的颜色

边代表了神经网络的一个参数，可以是任意实数

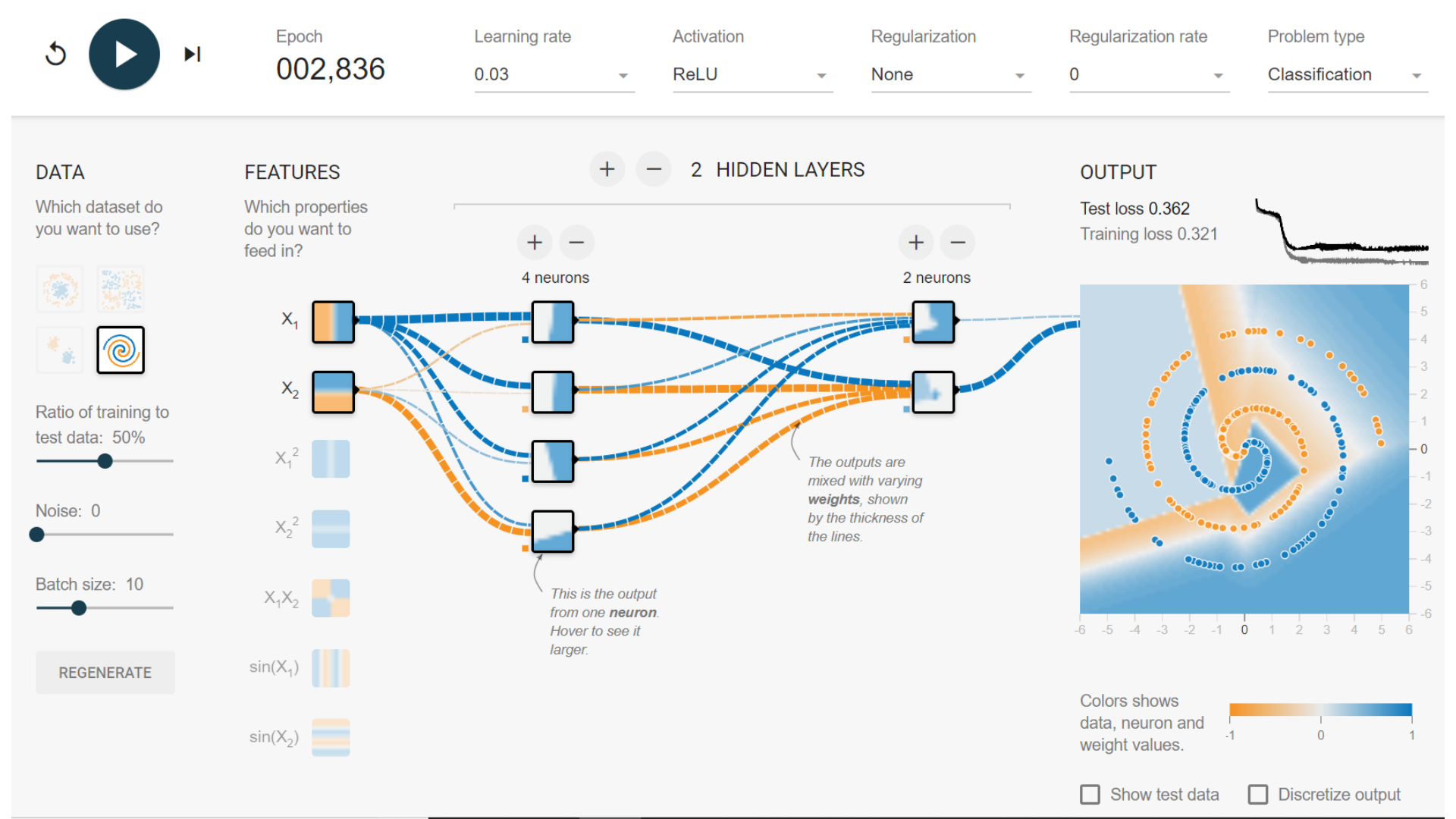
神经网络就是通过对参数的合理设置来解决分类或者回归问题的

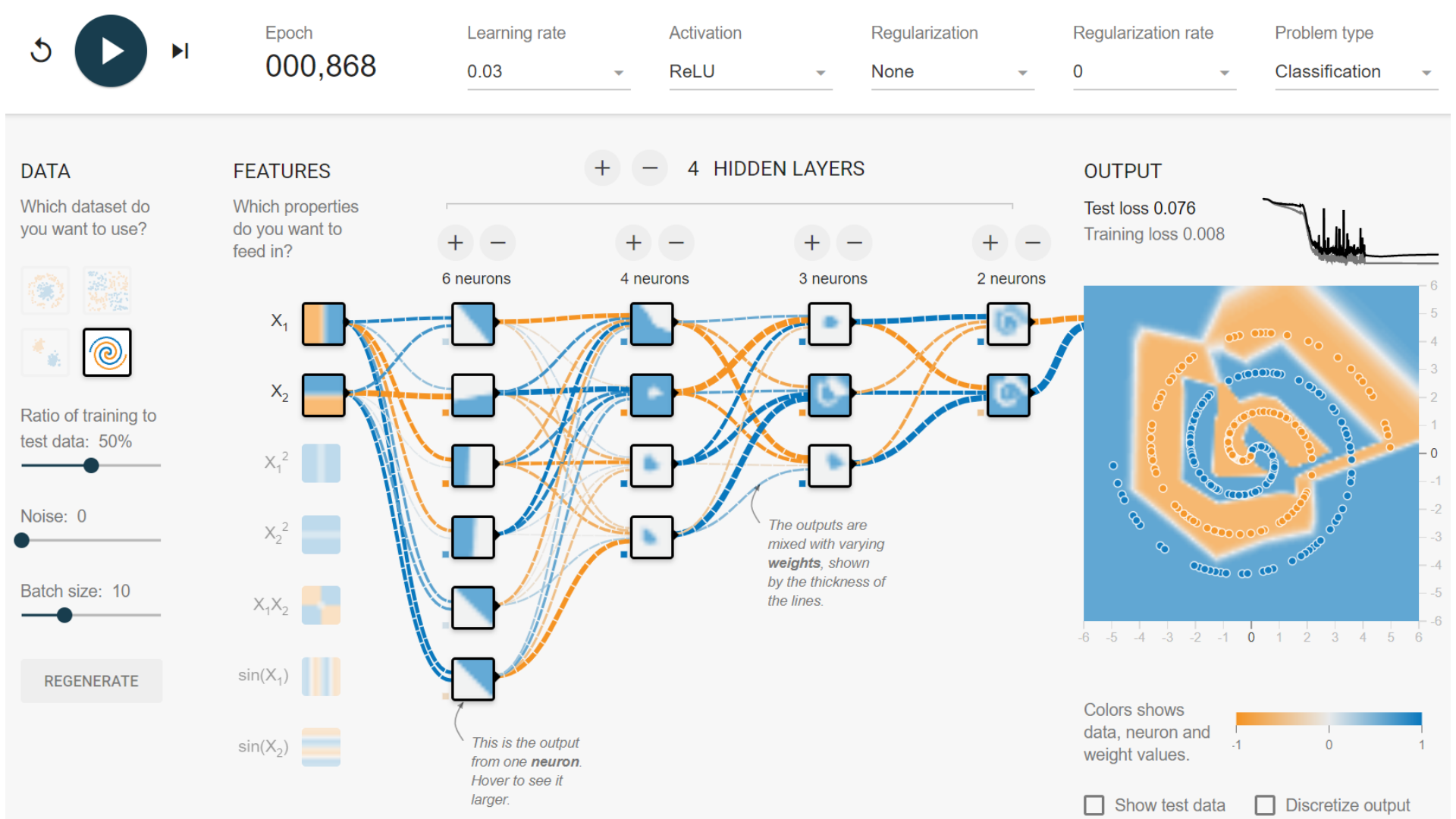
边的颜色体现了这个参数的取值，颜色越深，绝对值越大

节点的颜色代表了区分平面。这个平面上的每个点就代表了（x1,x2）的一种取值，当节点的输出值的绝对值越大时，颜色越深（蓝色）

例：x1节点的区分平面就是y轴

输出节点除了显示区分平面外，还显示了训练数据


Epoch
002,836
Learning rate
0.03
Activation
ReLU
Regularization
None
Regularization rate
0
Problem type
Classification
DATA
Which dataset do you want to use?
Ratio of training to test data: 50%
Noise: 0
Batch size: 10
REGENERATE
FEATURES
Which properties do you want to feed in?
X_1
X_2
X_1^2
X_2^2
X_1X_2
$\sin(X_1)$
$\sin(X_2)$
2 HIDDEN LAYERS
4 neurons
2 neurons
The outputs are mixed with varying weights, shown by the thickness of the lines.
This is the output from one neuron. Hover to see it larger.
OUTPUT
Test loss 0.362
Training loss 0.321
Colors shows data, neuron and weight values.
-1
0
1
Show test data
Discretize output


Epoch
000,868
Learning rate
0.03
Activation
ReLU
Regularization
None
Regularization rate
0
Problem type
Classification
DATA
Which dataset do you want to use?
Ratio of training to test data: 50%
Noise: 0
Batch size: 10
REGENERATE
FEATURES
Which properties do you want to feed in?
4 HIDDEN LAYERS
6 neurons
4 neurons
3 neurons
2 neurons
X1
X2
X1²
X2²
X1X2
sin(X1)
sin(X2)
This is the output from one neuron. Hover to see it larger.
The outputs are mixed with varying weights, shown by the thickness of the lines.
OUTPUT
Test loss 0.076
Training loss 0.008
Colors shows data, neuron and weight values.
-1
0
1
Show test data
Discretize output