SVHN_RNN 模型介绍

——基于Tensorflow实现的
街景门牌号码识别
深度学习模型

李煊
9月15日

深度学习关键词

- Approximation(Universal approximation theorem)
  - 激活函数
    - 非线性，有界，单调递增，连续可导
    - SGD(Stochastic Gradient Descent),BP(Back Propagation)
    - Cost function
- Representation(表达能力，特征；discriminative)
  - Sparse data → Dense representation(feature or embedding)
    - 图像数据→CNN
    - 时序数据→RNN
    - 离散数据→NN
- Balance the precision and the computational cost
  - Overfitting & regularization(提升精度)
  - Normalization (加速学习)

Content

- Introduction
- Project Source
  - 运行环境
- Architecture
- 网络结构定义思路
- RNN层
- 训练超参数
- 输出
  - 模型测试
- 改进思路

Introduction

- SVHN_RNN模型是利用深度神经网络实现识别街景门牌号码（Street View House Numbers）。

- SVHN_RNN 工程是用Tensorflow实现的一个”encode-decode”式的神经网络模型。
- 其中卷积层(the convolution layer)将输入图像编码成一个固定长度的特征向量，又称embedding.
- RNN层将定长的embedding解码成一个数字序列。

Project Source

任务

1. **设计并测试一个模型架构，使之能够识别出图像中的数字序列。**
   - 你的模型应基于 深度神经网络 或 卷积神经网络。
   - 你还可以尝试在 深度神经网络 中尝试使用 递归网络 来替换其中的分类层，并且将数字序列里的数字一个一个地输出。
2. **根据实际数据训练模型。**
   - 用人工合成的数据集，会有助于你较快地测试模型。
3. **将模型封装到一个安卓应用中。(可选)**

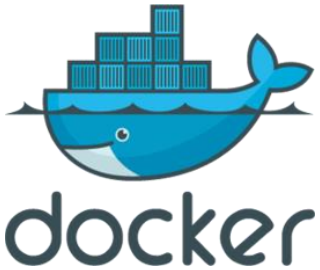
数据

Street View House Numbers (SVHN): （基于谷歌街景的）一个大规模的房屋门号数据库。

运行环境

- Ubuntu 14.04
- Python 3.5
- Docker
- Tensorflow

- Nvidia Tesla K20c



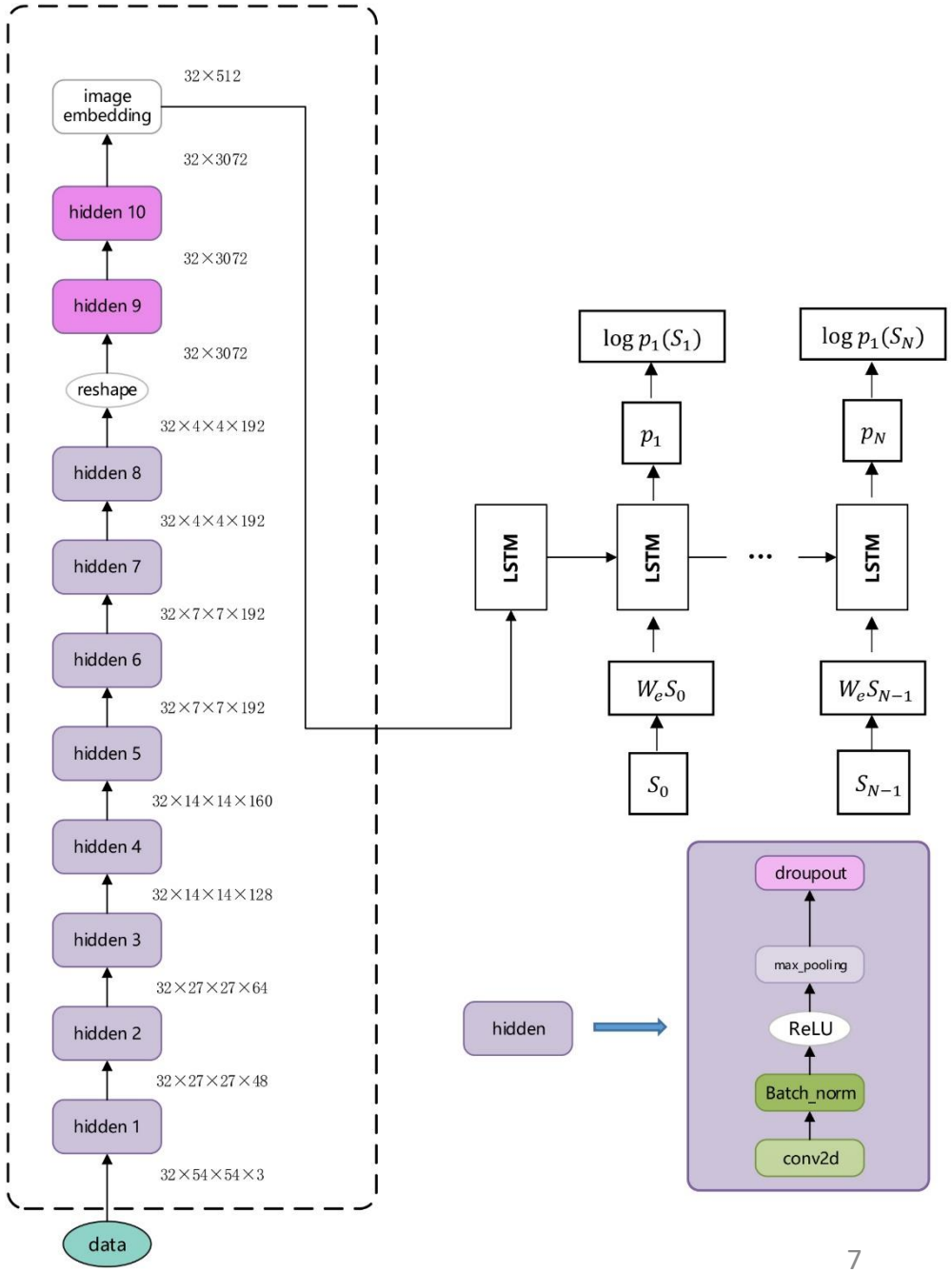
Architecture

Hidden layer 1-8，结构如右下角。

卷积层：提取特征
Batch_norm：正则(态)化输入数据
ReLU：激活函数
Max_pooling：特征筛选或降维
Dropout: 正则化网络，通过改变**网络结构**减少过拟合

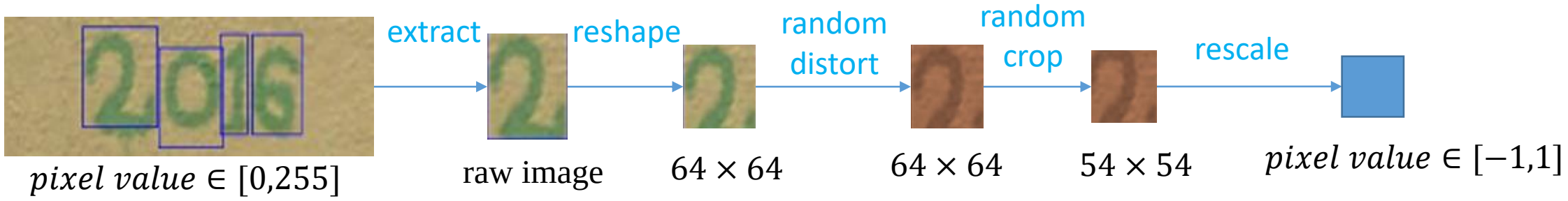
Hidden layer 9-10，全连接。
Full connection：聚合特征向量

RNN层，LSTM cell
输入：序列向量seq_embedding，由数字0~9,[10],[11]生成。
输出：下一个数字的概率(softmax层)



数据处理

- Image

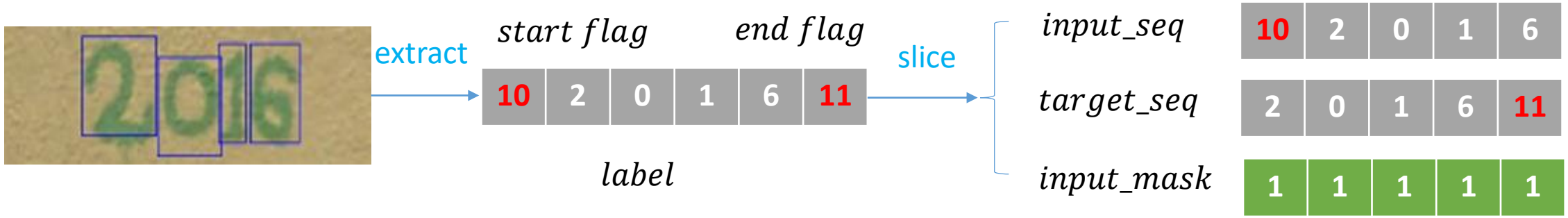


random distort
- 亮度
- 饱和度
- 色调
- 对比度

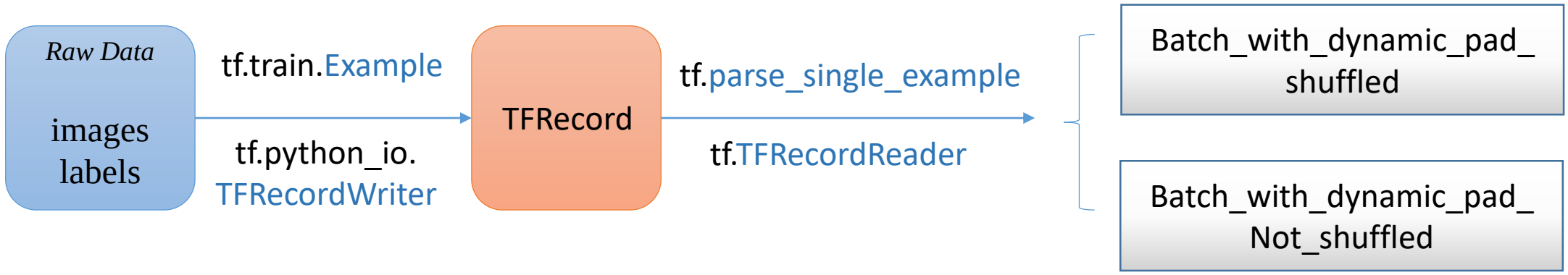
```
image = tf.image.random_brightness(image, max_delta=32. / 255.)
image = tf.image.random_saturation(image, lower=0.5, upper=1.5)
image = tf.image.random_hue(image, max_delta=0.032)
image = tf.image.random_contrast(image, lower=0.5, upper=1.5)
```

数据处理

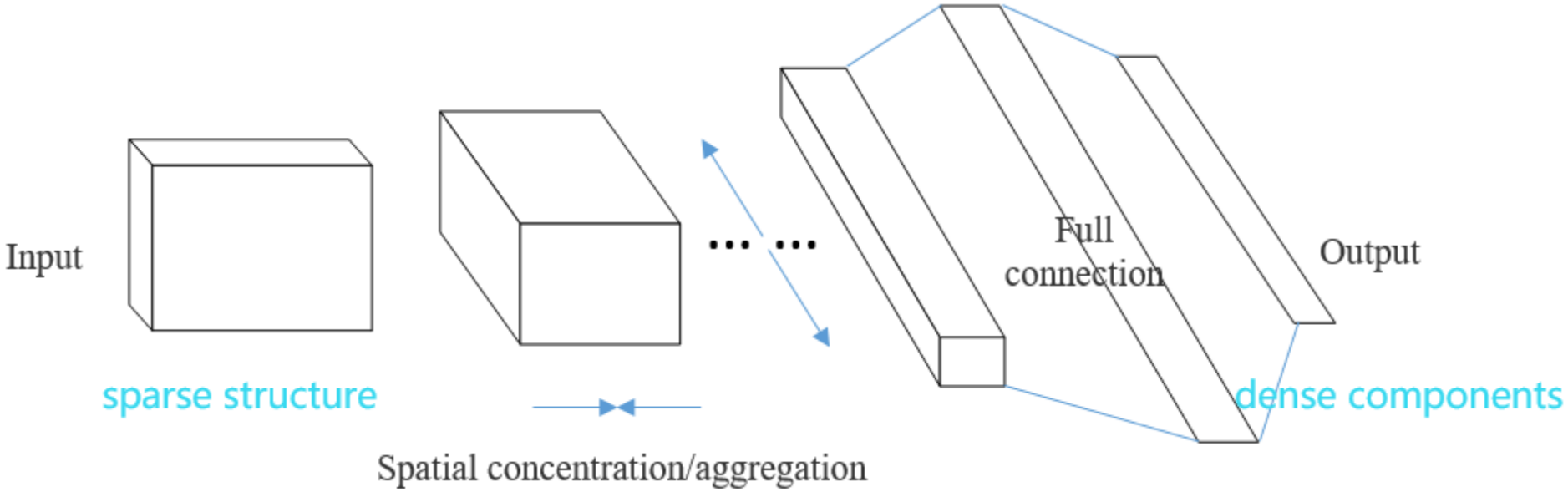
- Digital number



- Mini batch



网络结构定义思路



深度卷积网络是将 稀疏的输入图像经过空间聚合，生成一个紧致的特征向量。

网络设计准则：

1. 特征维度下降不要太快
2. 越高维特征越容易进行局部处理
3. 在全连接(或embedding)前的空间卷积，对于特征的表达能力不会造成太大的负作用
4. 平衡网络的深度和宽度.(精度与计算量的权衡)

网络结构定义思路

filters 依次增加，一般是上一层的1.5~2倍。

stride 一般取1~2。

stride =2时，特征减少1/4，所以对应增加filters来抵消维度的过度下降

Drop rate 一般取0.2~0.5。

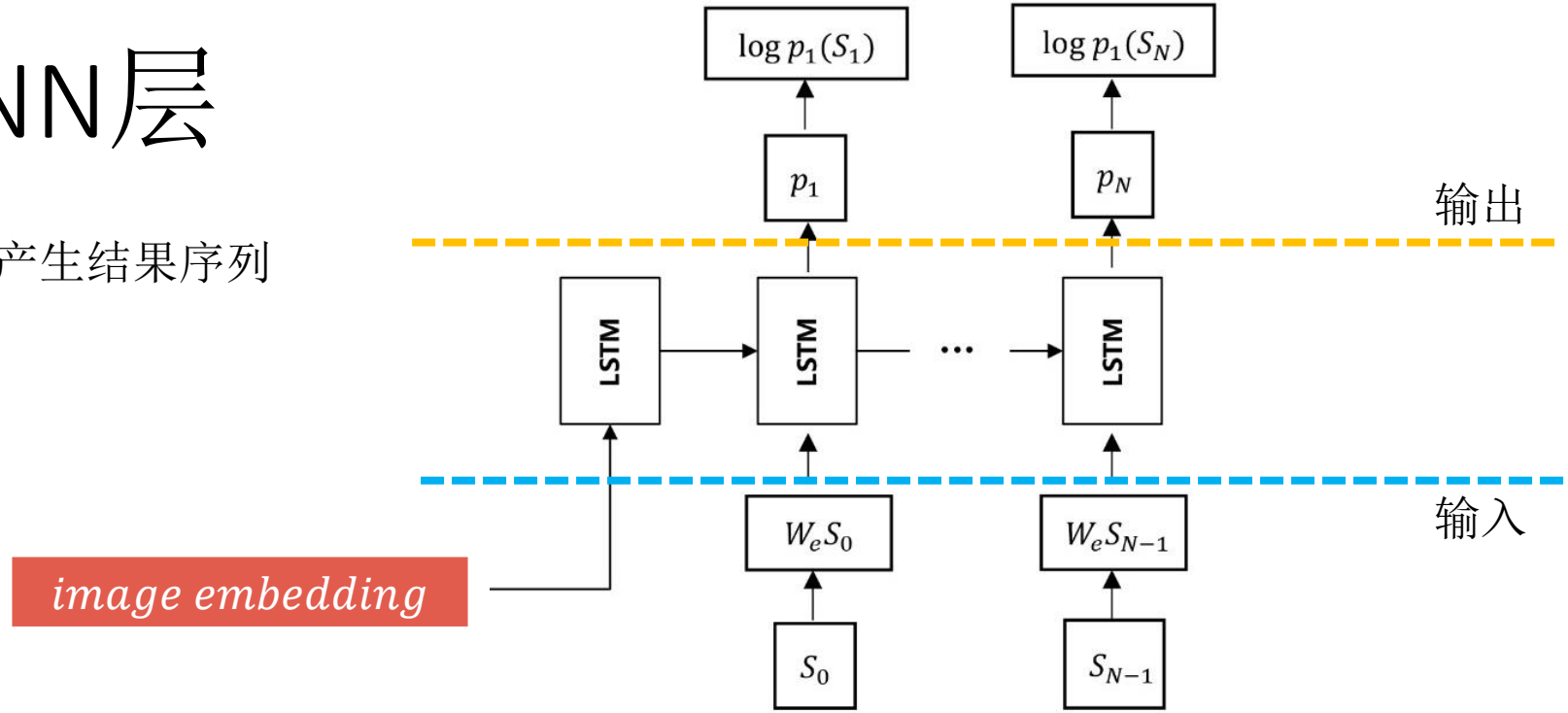
可得卷积层的层数至少为$log_2 n$。

```
with tf.variable_scope('hidden1'):
    conv = tf.layers.conv2d(x, filters=48, kernel_size=[
                                5, 5], padding='same')
    norm = tf.layers.batch_normalization(conv)
    activation = tf.nn.relu(norm)
    pool = tf.layers.max_pooling2d(
        activation, pool_size=[2, 2], strides=2, padding='same')
    dropout = tf.layers.dropout(pool, rate=drop_rate)
    hidden1 = dropout

with tf.variable_scope('hidden2'):
    conv = tf.layers.conv2d(hidden1, filters=64, kernel_size=[
                                5, 5], padding='same')
    norm = tf.layers.batch_normalization(conv)
    activation = tf.nn.relu(norm)
    pool = tf.layers.max_pooling2d(
        activation, pool_size=[2, 2], strides=1, padding='same')
    dropout = tf.layers.dropout(pool, rate=drop_rate)
    hidden2 = dropout
```

RNN层

用于产生结果序列



	Vocabulary mapping
0	$embedding_0$
1	$embedding_1$
...	
11	$embedding_{11}$

×

One-hot	0	1	...	11
S_k				

=

$embedding_k$

训练超参数

- 学习率
  - 指数衰减
- Batch size
  - 32 in train mode
  - 128 in evaluate mode
- 早停
  - no-improvement-in-*n(100)*
- 交叉验证

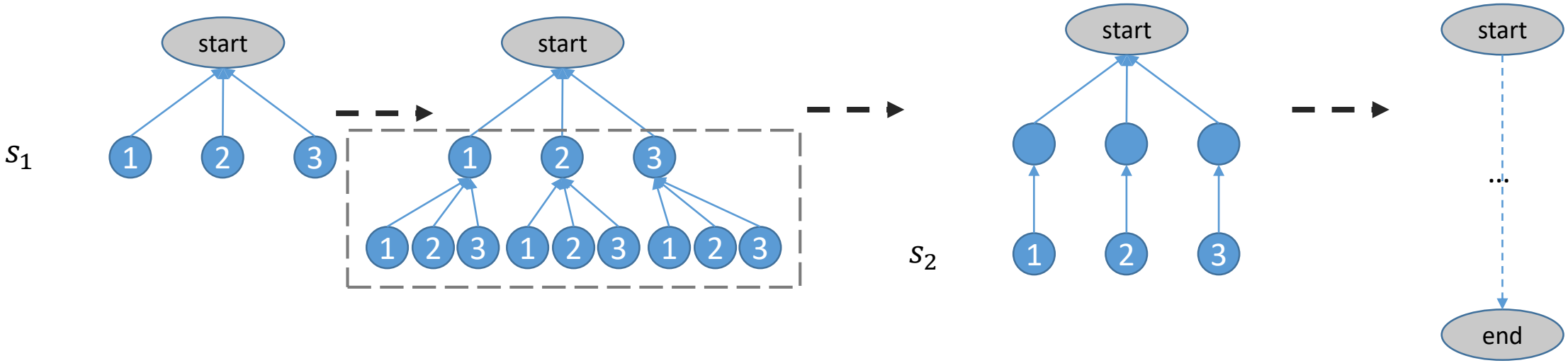
```
initial_learning_rate = training_options['learning_rate']  # 1e-2
decay_steps = training_options['decay_steps']  # 10000
decay_rate = training_options['decay_rate']  # 0.9

learning_rate = tf.train.exponential_decay(initial_learning_rate,
                                           global_step=global_step,
                                           decay_steps=decay_steps,
                                           decay_rate=decay_rate,
                                           staircase=True)
```

输出

Beam search

Top N



$$beam\begin{cases} numbers \\ state \\ score \end{cases}$$

候选数字序列

LSTM 状态

评分

$$score = \sum_i \ln p(s_i)$$

模型测试

```
The candidate result as follow.
No.      result  probability
 1        257     0.981
 2        255     0.005
 3        254     0.004
```

Branch: master ▾ SVHN_RNN / run_inference.ipynb

Find file | Copy path

lixuan0023 update run_inference 16ec8cc 21 days ago

1 contributor

139 lines (138 sloc) | 34.4 KB

Raw | Blame | History

In [1]:
```
from inference import Inference
%pylab inline
```

```
Populating the interactive namespace from numpy and matplotlib
```

In [2]:
```
path_to_checkpoint_file = '/notebooks/dataVolume/workspace/logs/train/model.ckpt-578000'
inference_model = Inference(path_to_checkpoint_file)
```

```
INFO:tensorflow:Restoring parameters from /notebooks/dataVolume/workspace/logs/train/model.ckpt-578000
```

In [3]:
```
path_to_image_file = './scratch_function/data/10.png'
image, inference_list = inference_model.run(path_to_image_file)
imshow(image)
inference_model.output(inference_list)
```

```
The candidate result as follow.
No.      result  probability
 1        16      0.578
 2        18      0.388
 3        10      0.013
```

控制台信息

每100 epoch 显示loss

每1000 epoch 交叉验证

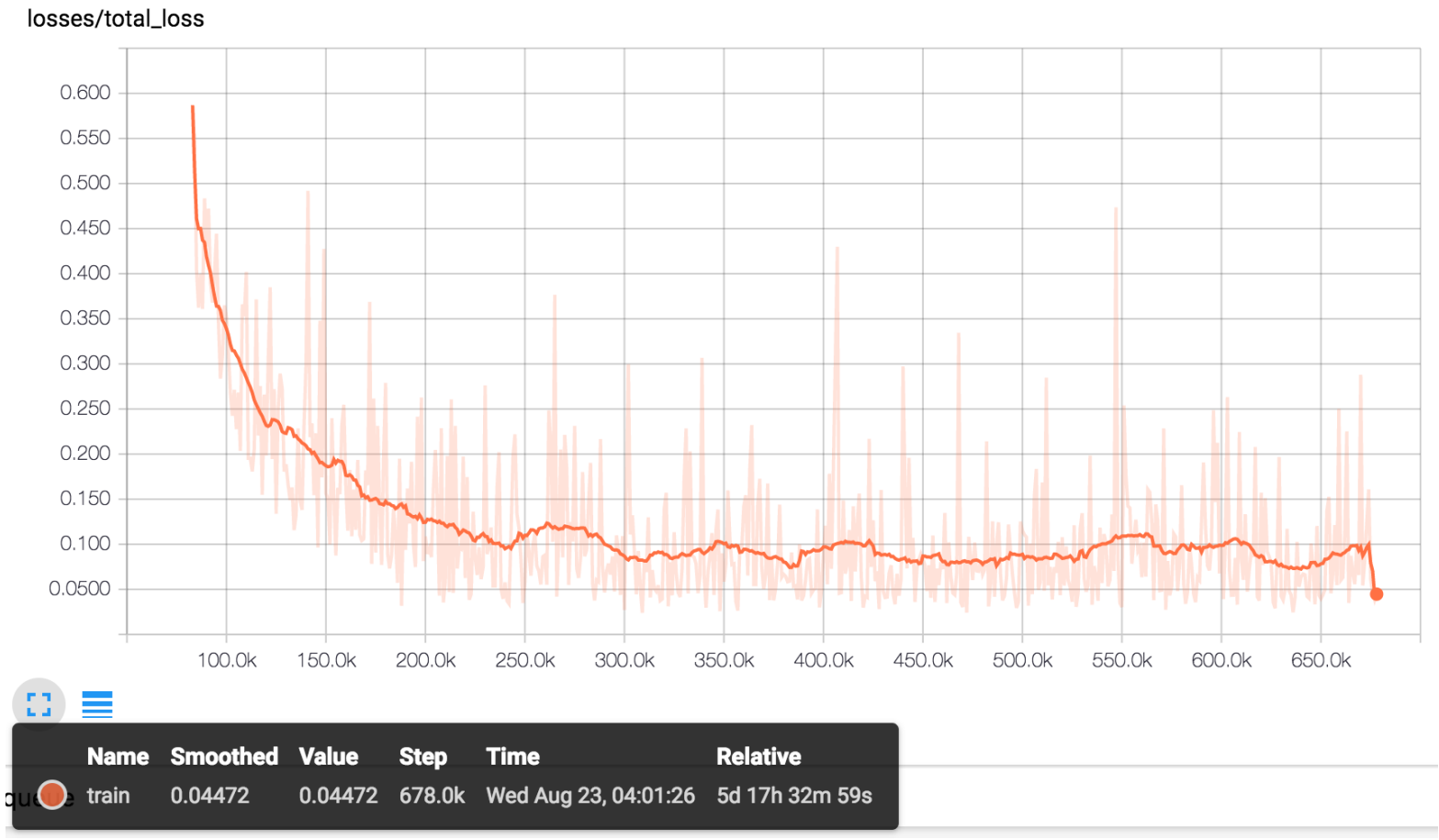
```
=> 2017-08-15 07:37:03.166343: step 100, loss = 2.412788 (38.8 examples/sec)
=> 2017-08-15 07:38:24.061638: step 200, loss = 2.363992 (39.6 examples/sec)
=> 2017-08-15 07:39:44.460196: step 300, loss = 2.316829 (39.8 examples/sec)
=> 2017-08-15 07:41:05.297330: step 400, loss = 2.270639 (39.6 examples/sec)
=> 2017-08-15 07:42:26.149749: step 500, loss = 2.215395 (39.6 examples/sec)
=> 2017-08-15 07:43:46.720148: step 600, loss = 2.242262 (39.7 examples/sec)
=> 2017-08-15 07:45:07.660818: step 700, loss = 2.235604 (39.5 examples/sec)
=> 2017-08-15 07:46:28.245084: step 800, loss = 2.237282 (39.7 examples/sec)
=> 2017-08-15 07:47:49.150330: step 900, loss = 2.204473 (39.6 examples/sec)
=> 2017-08-15 07:49:09.993308: step 1000, loss = 2.190285 (39.6 examples/sec)
=> Evaluating on validation dataset...
WARNING:tensorflow:Error encountered when serializing LAYER_NAME_UIDS.
Type is unsupported, or the types of the items don't match field type in CollectionDef.
'dict' object has no attribute 'name'
2017-08-15 07:49:11.289509: I tensorflow/core/common_runtime/gpu/gpu_device.cc:996] Crea
0, name: Tesla K20c, pci bus id: 0000:04:00.0)
2017-08-15 07:49:11.289585: I tensorflow/core/common_runtime/gpu/gpu_device.cc:996] Crea
1, name: Tesla K20c, pci bus id: 0000:05:00.0)
2017-08-15 07:49:11.289605: I tensorflow/core/common_runtime/gpu/gpu_device.cc:996] Crea
2, name: Tesla K20c, pci bus id: 0000:08:00.0)
2017-08-15 07:49:11.289622: I tensorflow/core/common_runtime/gpu/gpu_device.cc:996] Crea
3, name: Tesla K20c, pci bus id: 0000:09:00.0)
2017-08-15 07:49:11.289640: I tensorflow/core/common_runtime/gpu/gpu_device.cc:983] Igno
0, pci bus id: 0000:82:00.0) with Cuda multiprocessor count: 1. The minimum required cou
h the env var TF_MIN_GPU_MULTIPROCESSOR_COUNT.
2017-08-15 07:49:11.289659: I tensorflow/core/common_runtime/gpu/gpu_device.cc:996] Crea
5, name: Tesla K20c, pci bus id: 0000:85:00.0)
2017-08-15 07:49:11.289675: I tensorflow/core/common_runtime/gpu/gpu_device.cc:996] Crea
6, name: Tesla K20c, pci bus id: 0000:86:00.0)
2017-08-15 07:49:11.289691: I tensorflow/core/common_runtime/gpu/gpu_device.cc:996] Crea
7, name: Tesla K20c, pci bus id: 0000:89:00.0)
2017-08-15 07:49:11.289707: I tensorflow/core/common_runtime/gpu/gpu_device.cc:996] Crea
8, name: Tesla K20c, pci bus id: 0000:8a:00.0)
2017-08-15 07:49:15.102107: I tensorflow/core/common_runtime/gpu/pool_allocator.cc:247]
count=5056 evicted_count=1000 eviction_rate=0.197785 and unsatisfied allocation rate=0.2
2017-08-15 07:49:15.102235: I tensorflow/core/common_runtime/gpu/pool_allocator.cc:259]
==> accuracy = 0.280354, best accuracy 0.000000
WARNING:tensorflow:Error encountered when serializing LAYER_NAME_UIDS.
Type is unsupported, or the types of the items don't match field type in CollectionDef.
'dict' object has no attribute 'name'
=> Model saved to file: /notebooks/dataVolume/workspace/logs/train/model.ckpt-1000
=> patience = 100
```

训练精度

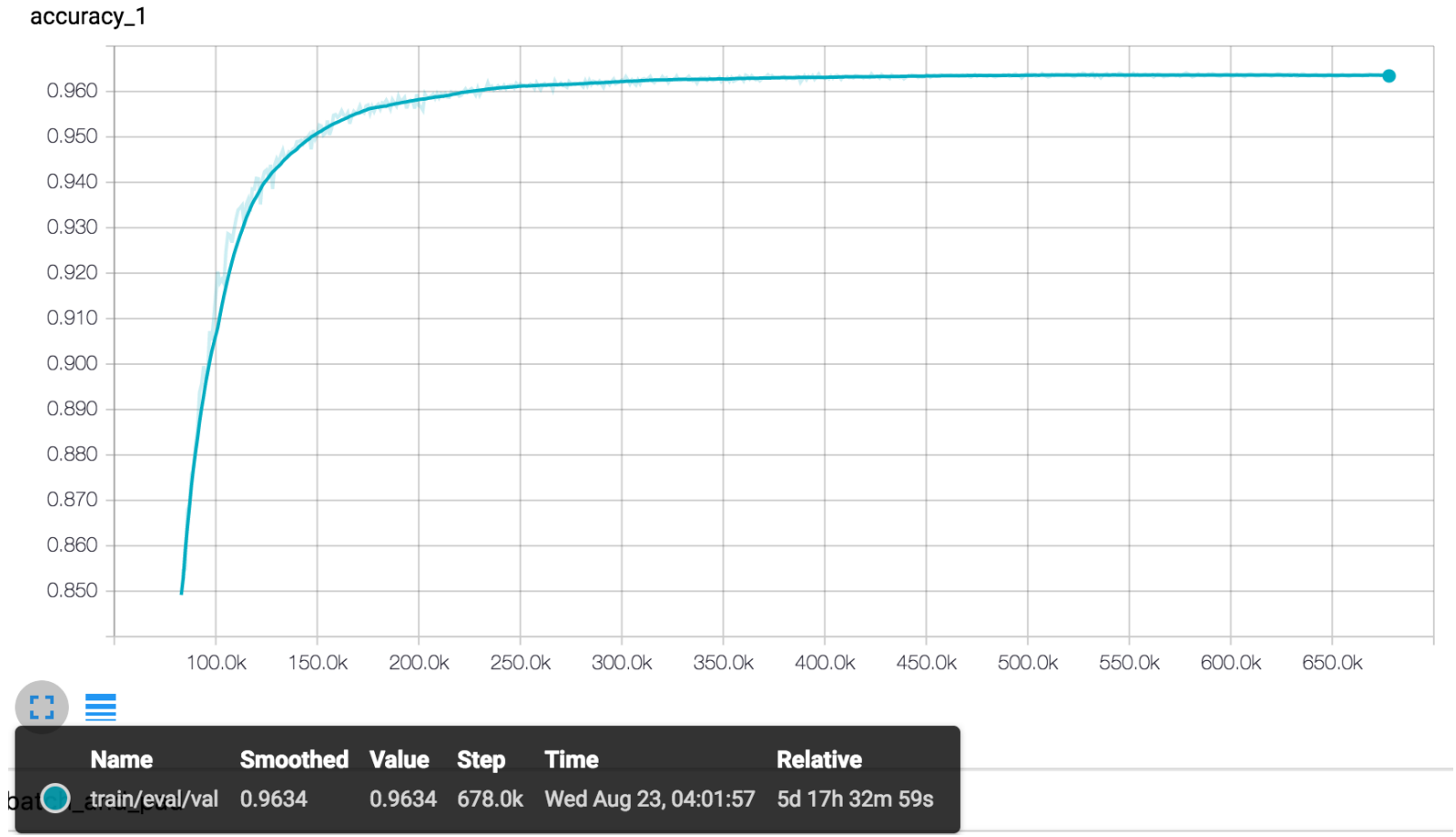
保存模型参数

早停参数

Loss


losses/total_loss
0.600
0.550
0.500
0.450
0.400
0.350
0.300
0.250
0.200
0.150
0.100
0.0500
100.0k
150.0k
200.0k
250.0k
300.0k
350.0k
400.0k
450.0k
500.0k
550.0k
600.0k
650.0k
Name Smoothed Value Step Time Relative
train 0.04472 0.04472 678.0k Wed Aug 23, 04:01:26 5d 17h 32m 59s

精度


accuracy_1
0.960
0.950
0.940
0.930
0.920
0.910
0.900
0.890
0.880
0.870
0.860
0.850
100.0k
150.0k
200.0k
250.0k
300.0k
350.0k
400.0k
450.0k
500.0k
550.0k
600.0k
650.0k
Name
Smoothed
Value
Step
Time
Relative
train/eval/val
0.9634
0.9634
678.0k
Wed Aug 23, 04:01:57
5d 17h 32m 59s

改进思路

- 优化模型
  - 全连接层前将max_pooling替换成average_pooling.
  - 尝试减少RNN中cell数量。
- 使用RCNN等网络结构来实现对门牌的目标检测，再进行识别。