深度之眼
deepshare.net

实战四：LSTM实现新闻分类算法

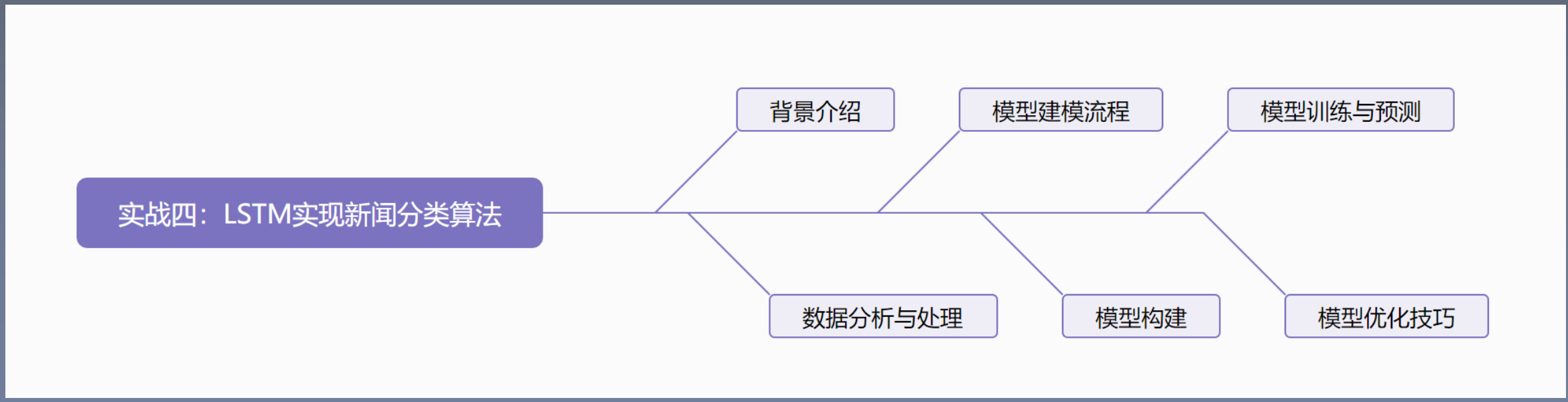
导师：GAUSS

深度之眼
deepshare.net

目录

知识树

Knowledge tree



深度之眼
deepshare.net

背景介绍

NLP任务

背景介绍

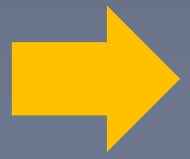
THUCNews是根据新浪新闻RSS订阅频道2005~2011年间的历史数据筛选过滤生成，包含74万篇新闻文档（2.19 GB），均为UTF-8纯文本格式。我们在原始新浪新闻分类体系的基础上，重新整合划分出14个候选分类类别：财经、彩票、房产、股票、家居、教育、科技、社会、时尚、时政、体育、星座、游戏、娱乐。

任务参考： http://thuctc.thunlp.org/

http://thuctc.thunlp.org/message

数据案例

马晓旭意外受伤让国奥警惕 无奈大雨
格外青睐殷家军记者傅亚雨沈阳报道
来到沈阳，国奥队依然没有摆脱雨水的
困扰。7月31日下午6点，国奥队的日
常训练再度受到大雨的干扰，无奈之下
队员们只慢跑了25分钟就草草收场。
31日上午10点，国奥队在奥体中心外
场训练的时候，天就是阴沉沉的，气象
预报显示当天下午沈阳就有大雨，但幸
好队伍上午的训练并没有受到任何干扰。
下午6点，当球队抵达训练场时，大雨
已经下了几个小时，而且丝毫没有停下
来的意思。抱着试一试的态度，球队开
始了当天下午的例行训练，25分钟过
去了，天气没有任何转好的迹象，为了
保护球员们，国奥队决定中止当天的训
练，全队立即返回酒店。在雨中训练对
足球队来说并不是什么稀罕事……



- 体育
- 娱乐
- 游戏
- 科技
- 房产
- 家居
- 教育

……

应用场景

参考：https://ai.baidu.com/tech/nlp/topictagger

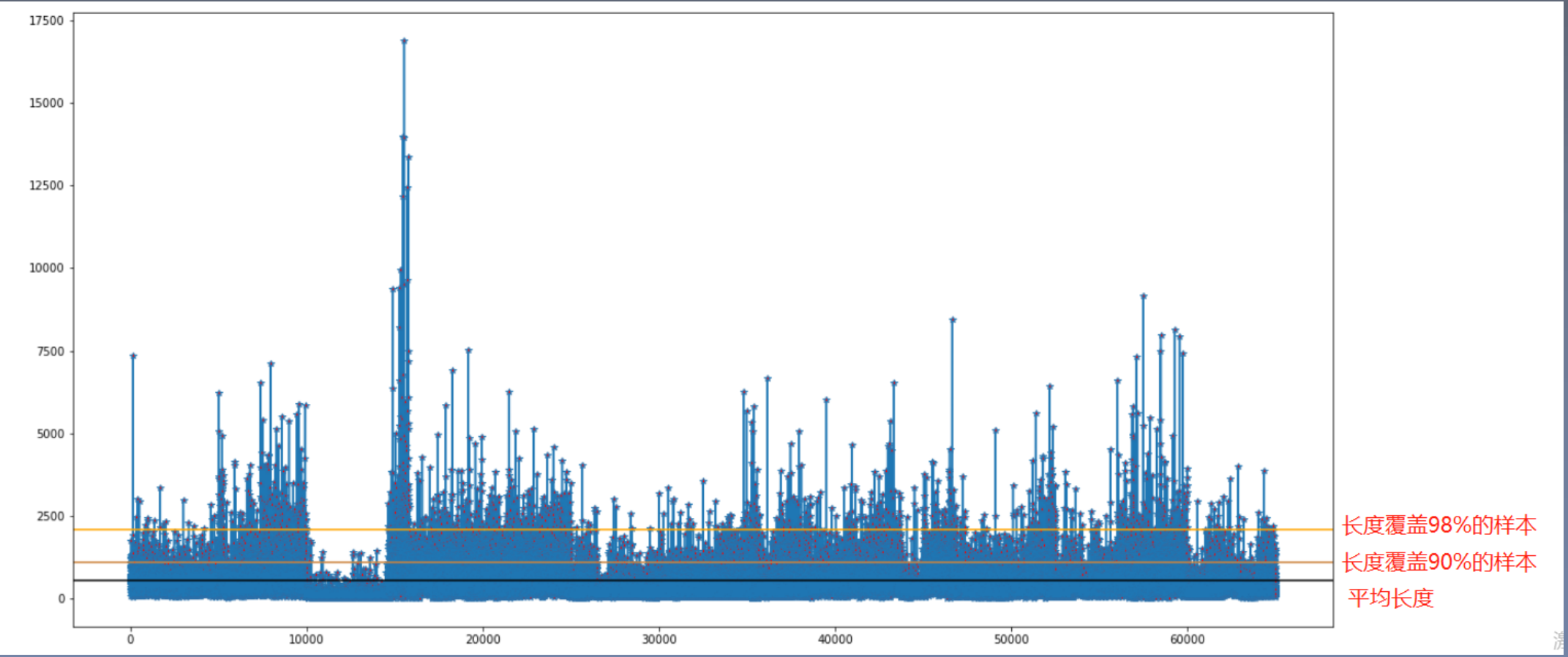
深度之眼
deepshare.net

数据分析与处理

文本长度统计


```
import matplotlib.pyplot as plt
import numpy as np
plt.figure(figsize=(20,10))
plt.plot(df['content_len'].tolist(),marker='*',markerfacecolor='red')
plt.axhline(y=np.mean(df['content_len'].tolist()),color="black",)
plt.axhline(y=np.percentile(df['content_len'].values,90),color="peru")
plt.axhline(y=np.percentile(df['content_len'].values,98),color="orange")
```

文本长度分布：

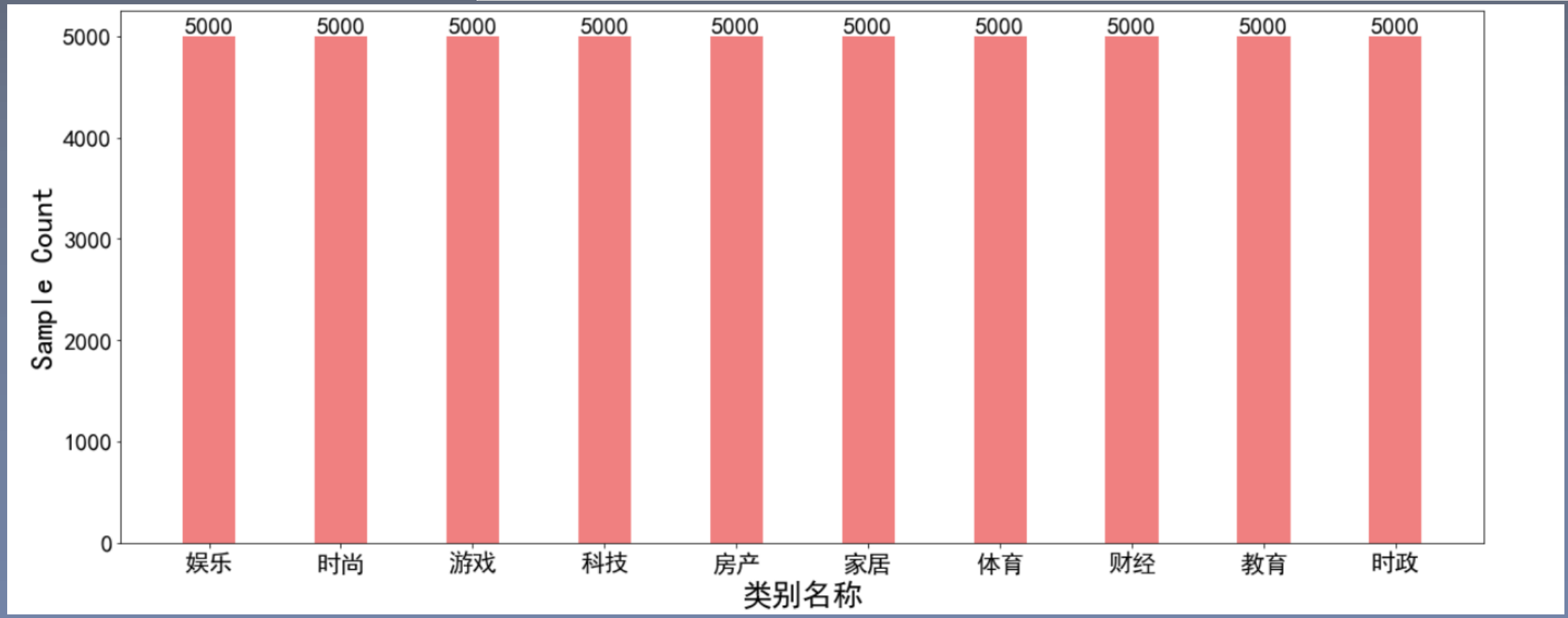


label统计

label分布统计：

```
import matplotlib.pyplot as plt
import numpy as np
plt.rcParams['font.family'] = ['sans-serif']
plt.rcParams['font.sans-serif'] = ['SimHei']
count_class = train['label'].value_counts()
plt.figure(figsize=(20,8))
class_bar = plt.bar(x=count_class.index, height=count_class.tolist(),width=0.4,color='lightcoral')
plt.xticks(fontsize=20)
plt.yticks(fontsize=20)
for bar in class_bar:
    height = bar.get_height()
    plt.text(bar.get_x() + bar.get_width() / 2, height+1, str(height), ha="center", va="bottom",fontsize=20)

plt.ylabel("Sample Count",fontsize=25)
plt.xlabel("类别名称",fontsize=25)
```



文本到序列转化

文本转化到序列

欢迎大家来到深度之眼参加NLP比赛课，希望大家可以学习到NLP比赛的技巧

jieba分词

欢迎 大家 来到 深度 之眼 参加 NLP 比赛 课 ， 希望 大家 可以 学习 到 NLP 比赛 的 技巧

构建词典

0 1 2 3 4 5 6 7 8 9 10 1 11 12 13 6 7 14 15

{'欢迎': 0,
'大家': 1,
'来到': 2,
'深度': 3,
'之眼': 4,
'参加': 5,
'NLP': 6,
'比赛': 7,
'课': 8,
', ': 9,
'希望': 10,
'可以': 11,
'学习': 12,
'到': 13,
'的': 14,
'技巧': 15}

文本截断、补全

文本截断

- 前截断
- 后截断

文本补全

- 前补全
- 后补全

文本截断长度如何选择?

数据截断

```
data_length = list(map(lambda x: len(x),train_))
np.percentile(data_length,85)
```

```
1899.0
```

```
train_
```

```
array([[  0,   0,   0, ..., 175, 217,  17],
       [437, 768, 134, ..., 175, 217,  17],
       [311,   1, 197, ..., 175, 217,  17],
       ...,
       [  0,   0,   0, ..., 175, 217,  17],
       [  0,   0,   0, ..., 175, 217,  17],
       [  0,   0,   0, ..., 175, 217,  17]], dtype=int32)
```

截断

补全

文本截断、补全

Tensorflow.keras接口提供一套文本编码、处理的工具。

```
tf.keras.preprocessing.text.Tokenizer(
    num_words=None, filters='!"#$%&()*+,-./:;<=>?@[\\]^_`{|}~\t\n', lower=True,
    split=' ', char_level=False, oov_token=None, document_count=0, **kwargs
)
```

- num_words: 需要保留的最大词数，基于词频。只有最常出现的 num_words 词会被保留。
- filters: 一个字符串，其中每个元素是一个将从文本中过滤掉的字符。默认值是所有标点符号，加上制表符和换行符，减去 ' 字符。
- lower: 布尔值。是否将文本转换为小写。
- split: 字符串。按该字符串切割文本。
- char_level: 如果为 True，则每个字符都将被视为标记。
- oov_token: 如果给出，它将被添加到 word_index 中，并用于在 text_to_sequence 调用期间替换词汇表外的单词。

文本截断、补全

tf.data.Dataset同样地提供padded_batch的方法

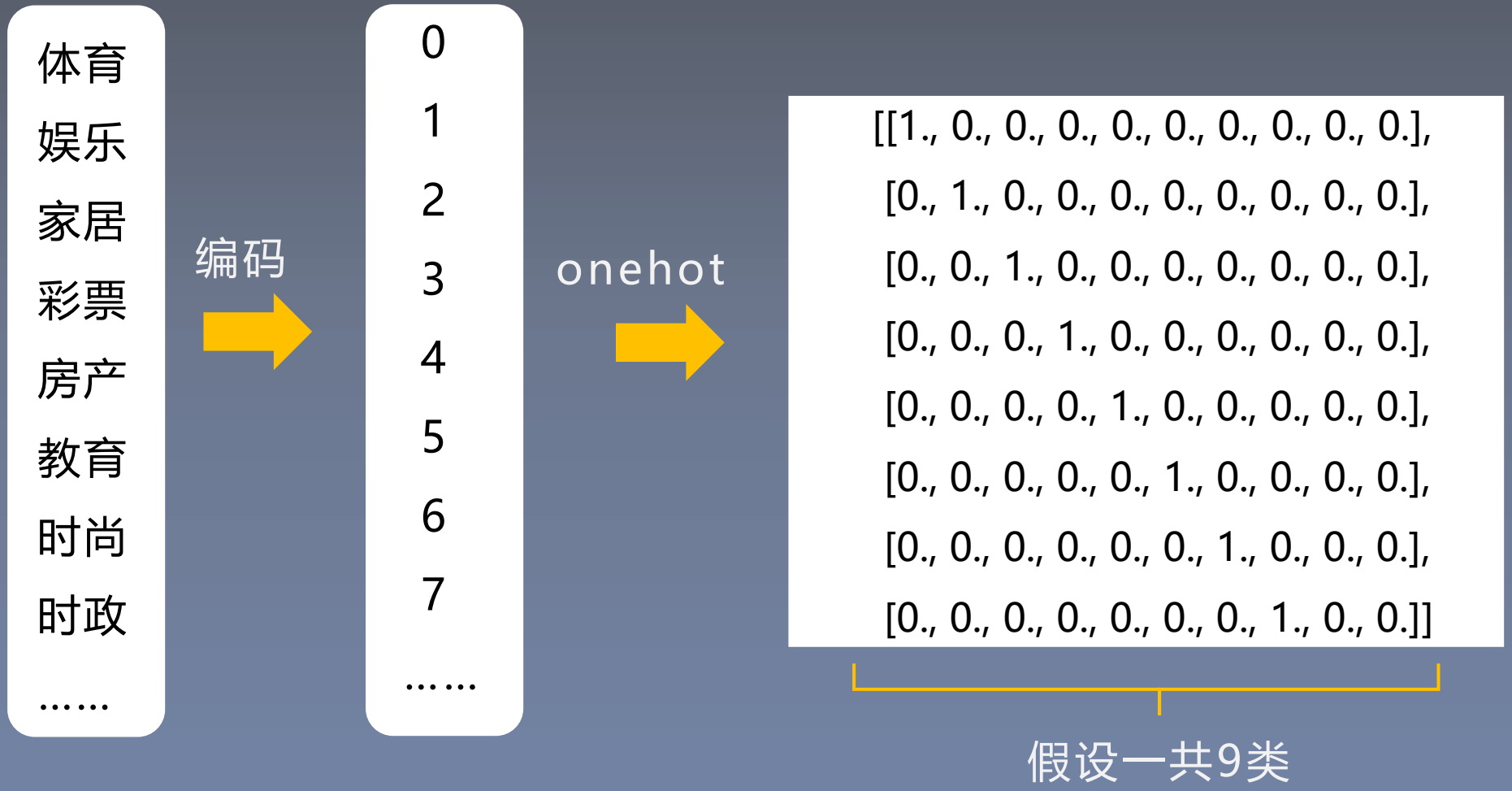
优点：直接在tf.data中补全序列（后补全）；可按照batch，固定序列长度。

缺点：不支持截断，前补全

```
padded_batch(
    batch_size, padded_shapes, padding_values=None, drop_remainder=False
)
```

- batch_size：每次喂入数据样本大小
- padded_shapes：表示在批处理之前应将每个输入元素的各个成分填充到的形状。任何未知的尺寸（如（None,））将被填充到在每个批次该维度的最大大小。
- padding_values：默认值0用于数字类型，空字符串用于字符串类型。
- drop_remainder：表示在batch_size元素少于元素的情况下是否应删除最后一批；默认行为是不删除较小的批次。

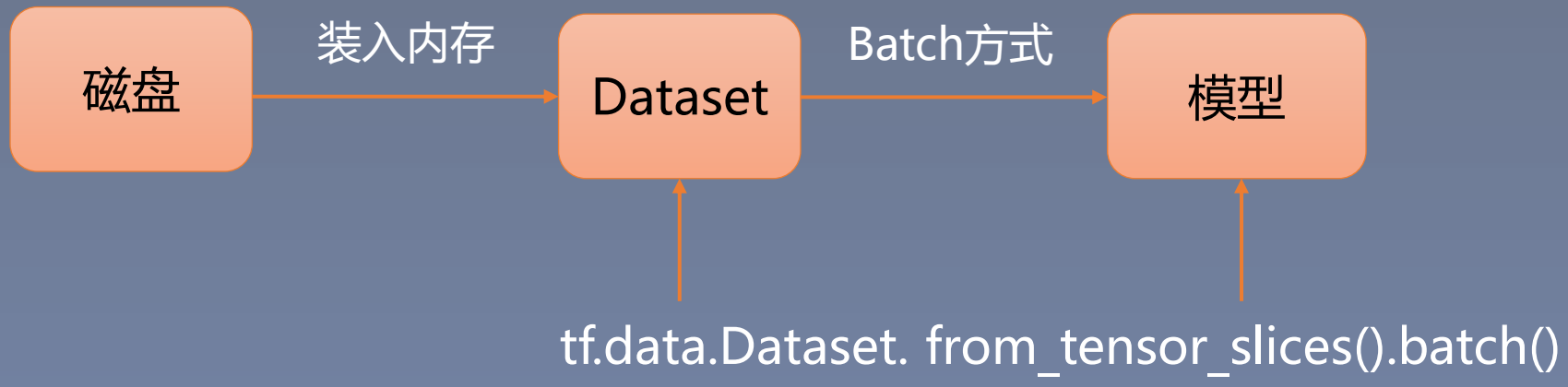
Label处理


体育
娱乐
家居
彩票
房产
教育
时尚
时政
……
编码
0
1
2
3
4
5
6
7
……
onehot
[[1., 0., 0., 0., 0., 0., 0., 0., 0.],
[0., 1., 0., 0., 0., 0., 0., 0., 0.],
[0., 0., 1., 0., 0., 0., 0., 0., 0.],
[0., 0., 0., 1., 0., 0., 0., 0., 0.],
[0., 0., 0., 0., 1., 0., 0., 0., 0.],
[0., 0., 0., 0., 0., 1., 0., 0., 0.],
[0., 0., 0., 0., 0., 0., 1., 0., 0.],
[0., 0., 0., 0., 0., 0., 0., 1., 0.]]
假设一共9类

Dataset数据读取

TensorFlow全新的数据读取方式：Dataset API

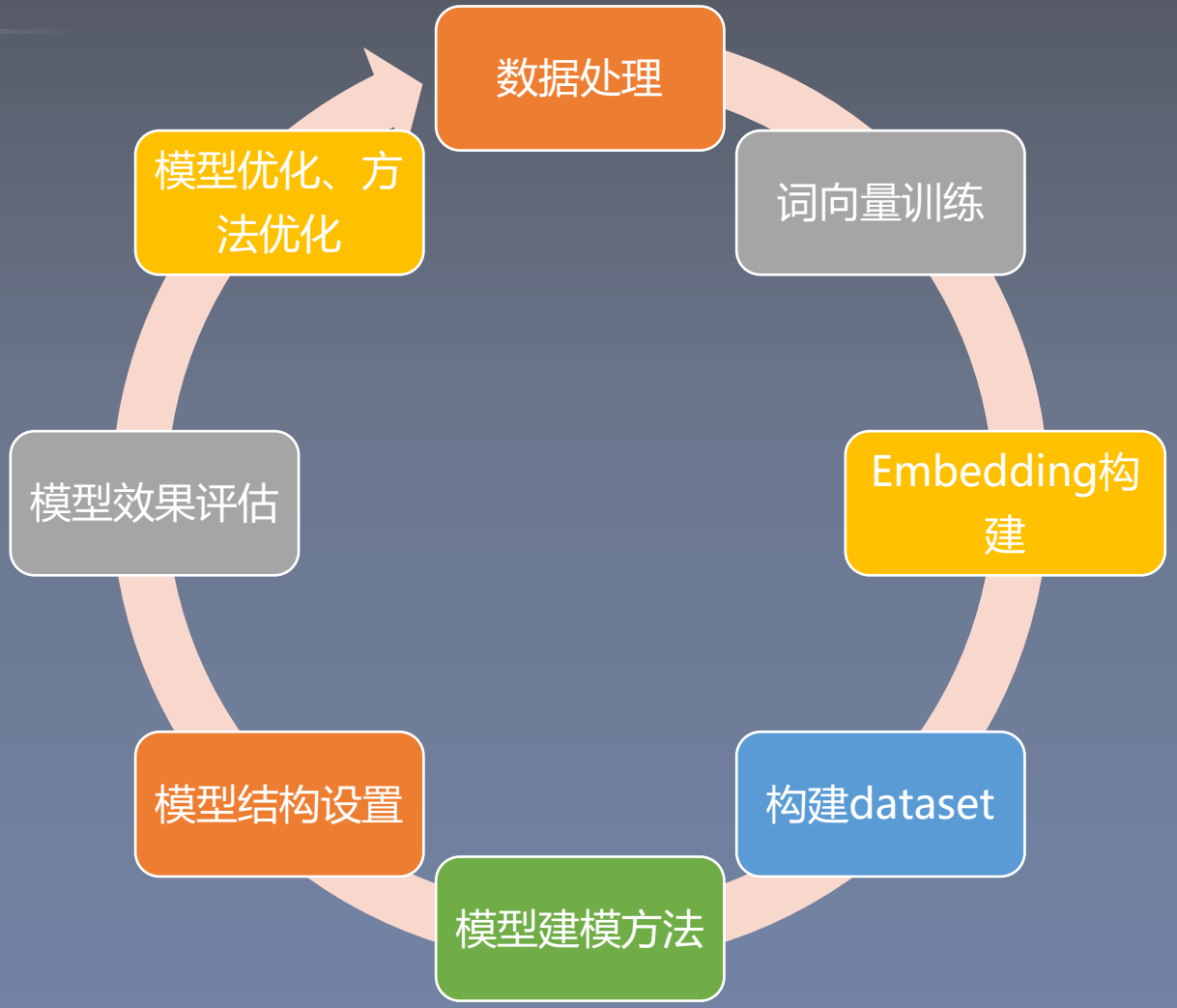
tf.data.Dataset.
from_tensor_slices



深度之眼
deepshare.net

模型建模流程

模型建模



深度之眼
deepshare.net

模型构建

Word Embedding构建流程

Build Word Embedding

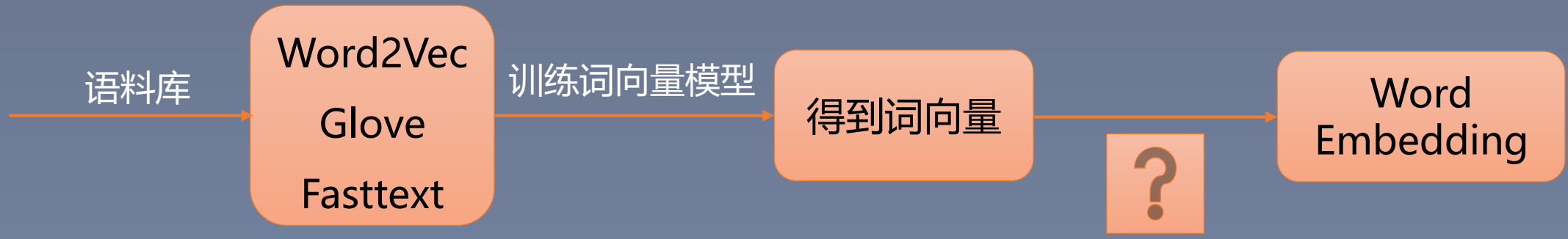
```
import sys
```

```
sys.getsizeof(np.zeros((64,1800,200)))
```

```
184320128
```

```
sys.getsizeof(np.zeros((64,1800,1)))
```

```
921728
```



Word Embedding构建

Build Word Embedding


```
{', ': 1,
 '的': 2,
 '。': 3,
 '在': 4,
 '、': 5,
 '了': 6,
 '是': 7,
 '“': 8,
 '”': 9,
 '和': 10,
 ': ': 11,
 '也': 12,
 '有': 13,
 '\xa0': 14,
 '》': 15,
 '《': 16,
 ')': 17,
 '(': 18,
 '我': 19,
 '都': 20,
 '他': 21,
 '中': 22,
 '月': 23,
 '将': 24,
 '就': 25,
 '我们': 26,
```

补全

1044285

7368

```
array([[ 0.        ,  0.        ,  0.        , ...,  0.        ,
         0.        ,  0.        ],
       [-0.80249637,  2.09233642,  2.41174316, ..., -1.5269419 ,
         0.16863775, -0.40093648],
       [ 0.29422134, -2.1090436 ,  0.75257933, ...,  0.55900234,
        -0.91687071,  0.35363957],
       ...,
       [ 0.18558396, -0.00910257,  0.24313086, ...,  0.09952487,
        -0.08558567, -0.13468956],
       [-0.02052246,  0.24421895, -0.09710397, ...,  0.1536202 ,
         0.15063426, -0.08361475],
       [ 0.13947084,  0.05973758,  0.0057267 , ..., -0.12683548,
        -0.04992896,  0.05894343]])
```

模型框架

The framework of model

- **Bidirectional RNNs** (GRU、LSTM)
- CNN
- Transformer
- Capsule等

Embedding层

Embedding Layer

Embedding层是如何计算的?

欢迎 大家 来到 深度 之眼 参加 Tensorflow 实战 课，希望 .....

[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

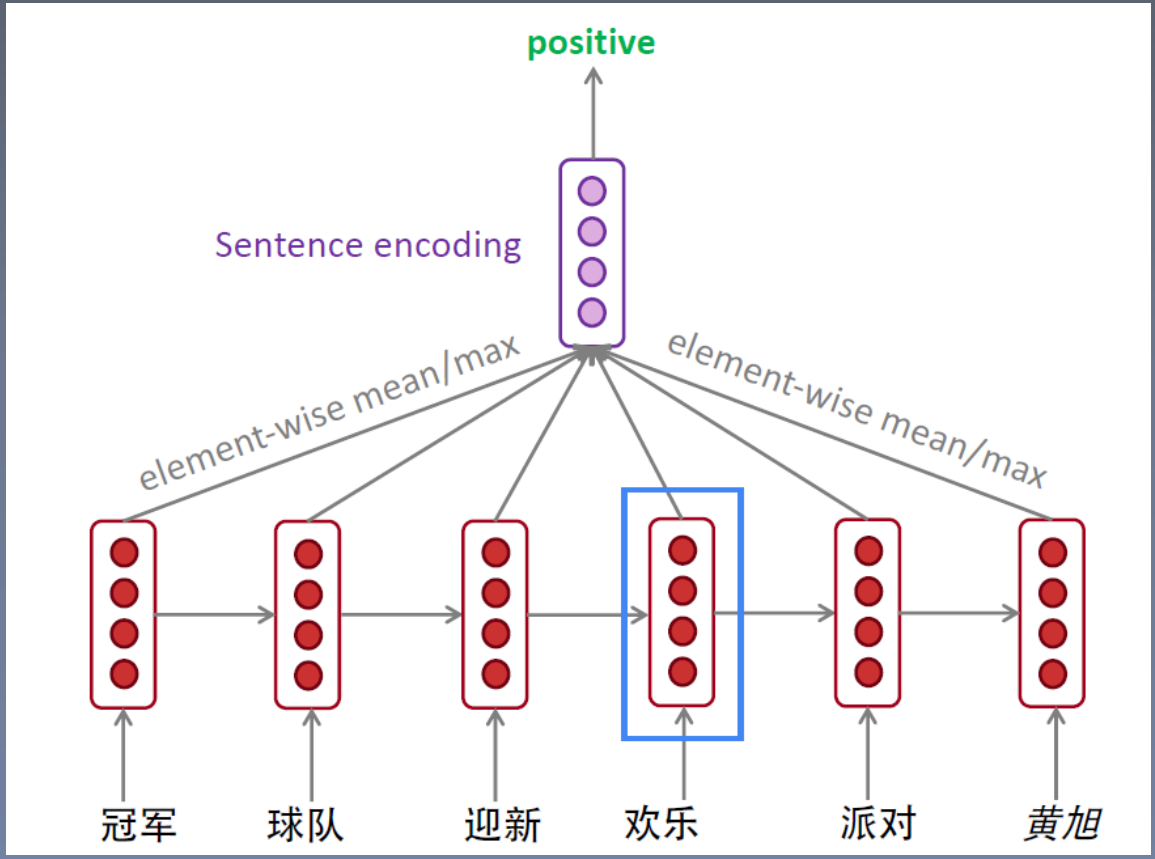
×

```
[[0.2886175 , 0.80598666, 0.81430982, 0.34097919],
 [0.51068579, 0.90726968, 0.31009823, 0.26969601],
 [0.06610848, 0.47337733, 0.80782722, 0.18688145],
 [0.71365487, 0.4570546 , 0.74953238, 0.93002451],
 [0.44649379, 0.08139837, 0.30183416, 0.15376385],
 [0.70414365, 0.43174657, 0.61739017, 0.81355264],
 [0.67939095, 0.47823112, 0.61334507, 0.05838023],
 [0.05895247, 0.84656122, 0.07303169, 0.05458424],
 [0.65914415, 0.19221401, 0.96600996, 0.62232913],
 [0.95428438, 0.81280639, 0.49041015, 0.89731348],
 [0.67071844, 0.28264647, 0.02227384, 0.10535961],
 [0.8836602 , 0.87078514, 0.24007419, 0.18909079],
 [0.08396557, 0.28140216, 0.33899817, 0.37048985],
 [0.05626022, 0.22627358, 0.543999  , 0.7997996 ],
 [0.95389475, 0.99723204, 0.91763436, 0.46344343],
 [0.28422473, 0.66377275, 0.40676082, 0.85823169]]
```

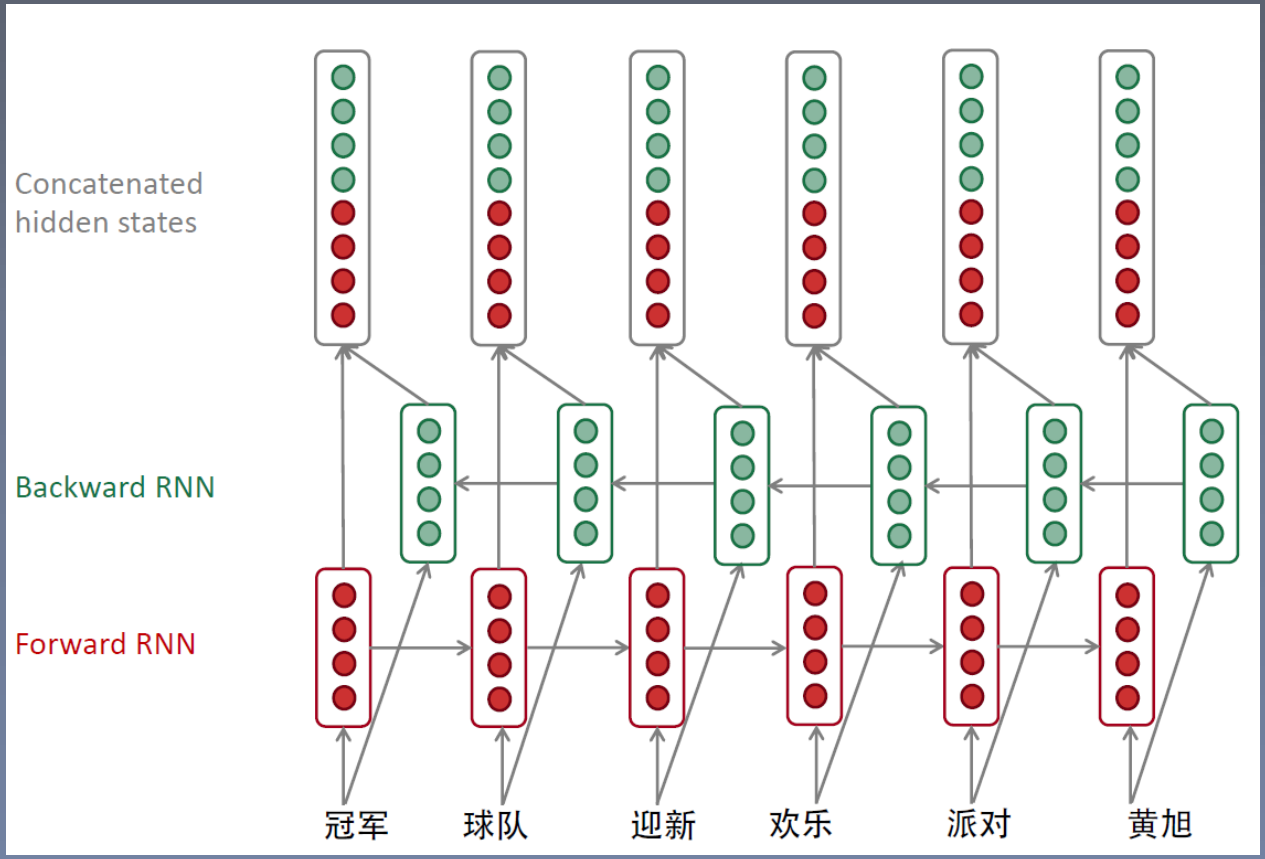
=

```
[[0.30528183, 0.44330278, 0.830672  , 0.40084265],
 [0.77597367, 0.21495795, 0.86622477, 0.28538754],
 [0.66355725, 0.19148728, 0.01173291, 0.34239544],
 [0.64620611, 0.54421693, 0.99350563, 0.13253877],
 [0.83564585, 0.03547649, 0.92305862, 0.22914557],
 [0.66759431, 0.2815053 , 0.80458928, 0.95565446],
 [0.22181455, 0.17954265, 0.72992925, 0.05537759],
 [0.93069185, 0.56526825, 0.19636648, 0.90954081],
 [0.70893121, 0.24656699, 0.02993562, 0.79396246],
 [0.65473165, 0.79020451, 0.15830648, 0.96676224]]
```

Bidirectional RNNs



单层RNN



多层RNN

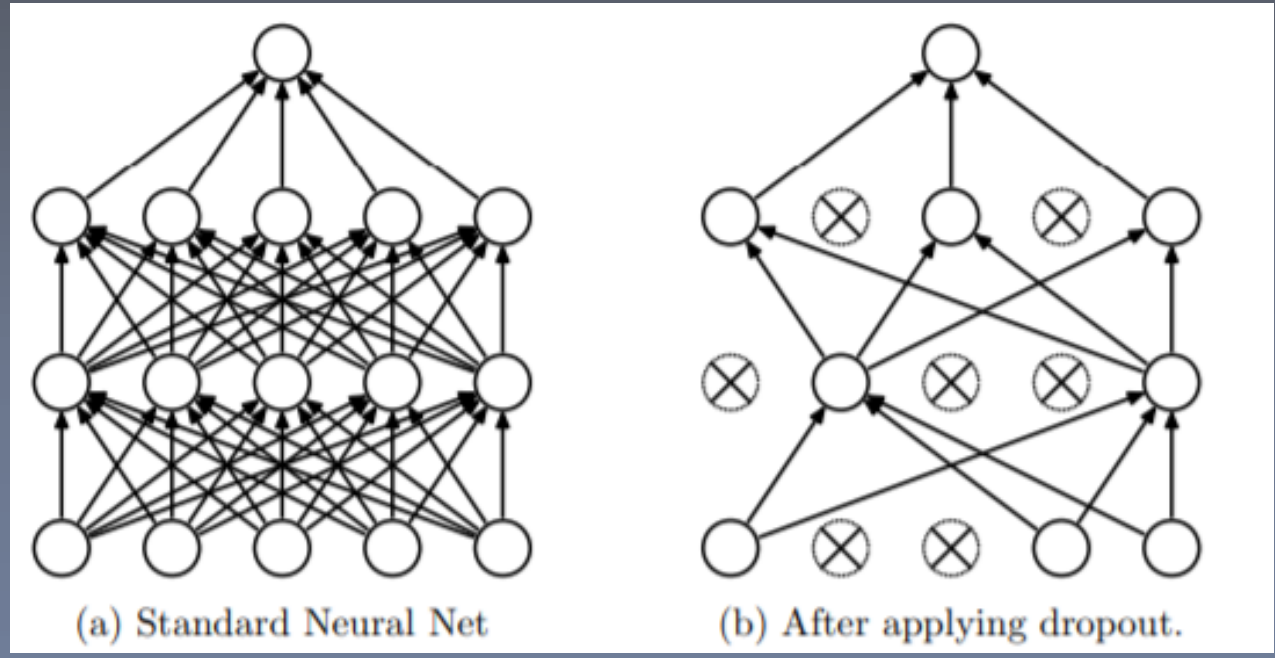
池化层

- 最大池化　tf.keras.layers.GlobalMaxPool1D
- 平均池化　tf.keras.layers.GlobalAveragePooling1D
- Attention池化 https://arxiv.org/pdf/1512.08756v3.pdf

Dropout

Dropout

作用：减少过拟合的风险



参考：
http://www.cs.toronto.edu/~rsalakhu/papers/srivastava14a.pdf

Dropout


```
tf.keras.layers.Dropout(
    rate, noise_shape=None, seed=None, **kwargs
)
```

rate：在 0 和 1 之间浮动。需要丢弃的输入比例。

noise_shape: 1D 整数张量， 表示将与输入相乘的二进制 dropout 掩层的形状。 例如，如果你的输入尺寸为 (batch_size, timesteps, features)，然后 你希望 dropout 掩层在所有时间步都是一样的， 你可以使用 noise_shape=(batch_size, 1, features)。

seed: 一个作为随机种子的 Python 整数。

Batch Normalization

Batch Normalization

- 提升了训练速度，收敛过程大大加快
- 增加分类效果

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1...m}\}$;
Parameters to be learned: γ, β

Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma\widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

Algorithm 1: Batch Normalizing Transform, applied to activation x over a mini-batch.

Batch Normalization


```
tf.keras.layers.BatchNormalization(
    axis=-1, momentum=0.99, epsilon=0.001, center=True, scale=True,
    beta_initializer='zeros', gamma_initializer='ones',
    moving_mean_initializer='zeros', moving_variance_initializer='ones',
    beta_regularizer=None, gamma_regularizer=None, beta_constraint=None,
    gamma_constraint=None, renorm=False, renorm_clipping=None, renorm_momentum=0.99,
    fused=None, trainable=True, virtual_batch_size=None, adjustment=None, name=None,
    **kwargs
)
```

- axis: 整数，需要标准化的轴 （通常是特征轴）。 例如，在 data_format="channels_first" 的 Conv2D 层之后， 在 BatchNormalization 中设置 axis=1。
- momentum: 移动均值和移动方差的动量。
- epsilon: 增加到方差的小的浮点数，以避免除以零。
- center: 如果为 True，把 beta 的偏移量加到标准化的张量上。 如果为 False， beta 被忽略。

Batch Normalization

- scale: 如果为 True，乘以 gamma。 如果为 False，gamma 不使用。 当下一层为线性层（或者例如 nn.relu），这可以被禁用，因为缩放将由下一层完成。
- beta_initializer: beta 权重的初始化方法。
- gamma_initializer: gamma 权重的初始化方法。
- moving_mean_initializer: 移动均值的初始化方法。
- moving_variance_initializer: 移动方差的初始化方法。
- beta_regularizer: 可选的 beta 权重的正则化方法。
- gamma_regularizer: 可选的 gamma 权重的正则化方法。
- beta_constraint: 可选的 beta 权重的约束方法。
- gamma_constraint: 可选的 gamma 权重的约束方法。

深度之眼
deepshare.net

模型训练与预测

Build Model

模型训练

Training Model

自定义训练 or keras训练

- Early Stopping
- 学习率衰减
- 保存最优权重

深度之眼
deepshare.net

模型优化与提升

模型优化与提升

- 数据增强
- 模型结构调整
- 训练参数
- 多层RNN编码器

深度之眼
deepshare.net

面试中可能会问的问题

面试中可能会问的问题

- 文本分类中还有哪些没有解决的问题？
- 文本分类有些些模型？
- 一些细节的原理技术等
- CNN 、RNN、Transformer各有什么优劣？

深度之眼
deepshare.net

总结

本节小结

Summary

实战四：LSTM实现新闻分类算法	背景介绍	
	数据分析与处理	
	模型建模流程	
	模型构建	
	模型训练与预测	
	模型优化与提升	

结语

我 说

看过千万代码，不如实践一把！

深度之眼
deepshare.net

联系我们：

电话：18001992849

邮箱：service@deepshare.net

Q Q： 2677693114

公众号

客服微信