

BERT

- 1、基本概念
- 2、源码简介

分词

附：

BERT

BERT是一种预训练语言表示的方法，这意味着我们在大型文本语料库（例如Wikipedia）上训练通用的“语言理解”模型，然后将该模型用于我们关心的下游NLP任务（例如问题回答）。BERT优于以前的方法，因为它是第一个用于预训练NLP的无监督，深度双向系统。

1、基本概念

BERT基本组成为Seq2Seq网络：输入和输出都为序列的网络，中间为Transformer

为什么要Transformer? 因为传统RNN网络每个循环体都需要上一个循环体的中间结果，无法并行运算（不独立）；采用Self-Attention机制并行计算，取代RNN

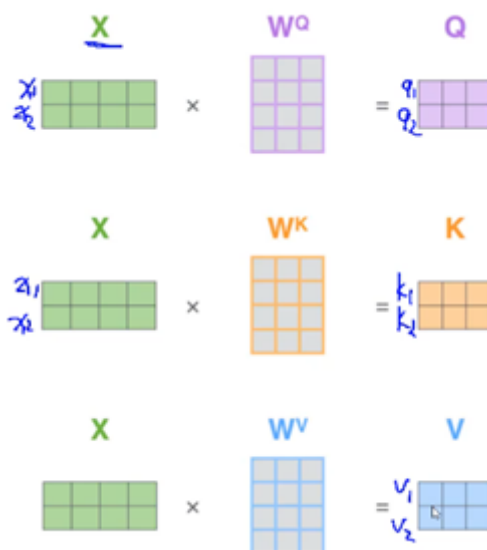
为什么不用word2vec? 相同的词表达含义一样，实际上不是这样的

Self-Attention机制：在不同的语境下不同的词有不同的权重。

方法：词->向量编码 x ->权重向量编码 z

预先定义三个矩阵（权重） W_q 、 W_k 、 W_v ，分别表示要被查询的、等待被查的和实际的特征信息，使输入词向量分别与三个矩阵进行运算。然后用softmax来进行归一化，求得每个词在当前句子中占有的权重（影响程度）

若要查询第一个词与每个词的关系，则用 W_q1W_k1 （内积：若无关系则内积为0）表示与第一个词的关系， W_q1W_k2 表示与第二个词的关系，以此类推





Multi-headed机制: 提取多种词向量特征（上述只是提取一种特征）

通过多个头机制（一般8个）得到多个特征表达，然后将所有特征拼接起来，再加一层全连接来降维

注：通过上述过程得到词特征向量，一般情况下还要堆叠多层来得到最终的词特征向量

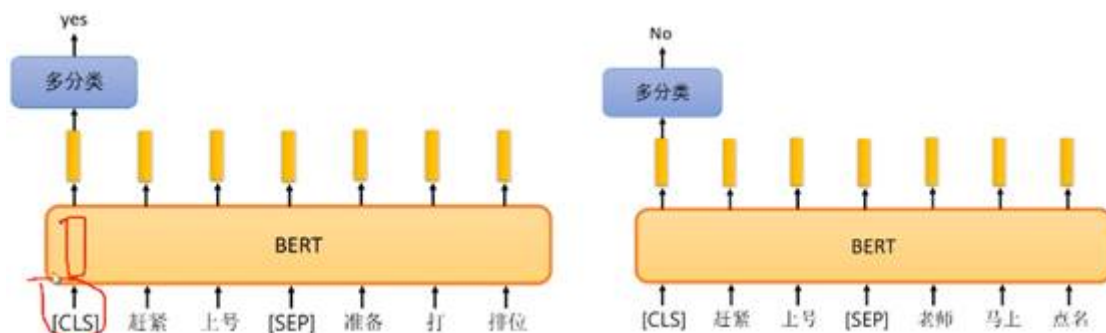
位置信息表达: 相对于上述得到的词特征向量再加上位置信息编码，一般是周期信号（正余弦等）

LayerNormalize: 对每个词的所有特征进行归一化处理（区别于对每批数据进行归一化），为了使得训练更快，更稳定。其中词的特征向量由上述处理得到

连接: 层归一化的同时，加入残差连接

Decoder: 相对于Encoder，Self-Attention计算不同（用q来查），加入了MASK机制。

训练BERT: 将句子中15%（替换为mask，随机变为其他词或不变）的词随机mask掉，让模型去预测被mask的是什么；预测两个句子是否应该连在一起（[seq]连接符，[cls]分类向量）



2、源码简介

数据:

1、加载数据（一般是两句话或一句话）==> 按模板格式化数据 + 定义标签

2、制作TF_record格式数据集==> 制作label字典（label: index）+ 数据分词 + 数据补全（并加CLS等标识符）+ 词典查ID（同时设置input_mask，作用是指示哪些位置的词是有效的）+ 段标识符（用来指示这是第几个话，0/1表示。最后用0补全长度）

```

546     # 生成tf_record文件
547     writer = tf.python_io.TFRecordWriter(output_file)
548
549     # 遍历每一条数据
550     for (ex_index, example) in enumerate(examples):
551         if ex_index % 10000 == 0:
552             tf.logging.info("Writing example %d of %d" % (ex_index, len(examples)))
553         # 对每个样本制作feature
554         feature = convert_single_example(ex_index, example, label_list,
555                                         max_seq_length, tokenizer)
556
557     def create_int_feature(values):
558         f = tf.train.Feature(int64_list=tf.train.Int64List(value=list(values)))
559         return f
560
561     features = collections.OrderedDict()
562     features["input_ids"] = create_int_feature(feature.input_ids)
563     features["input_mask"] = create_int_feature(feature.input_mask)
564     features["segment_ids"] = create_int_feature(feature.segment_ids)
565     features["label_ids"] = create_int_feature([feature.label_id])
566     features["is_real_example"] = create_int_feature(
567         [int(feature.is_real_example)])
568
569     tf_example = tf.train.Example(features=tf.train.Features(feature=features))
570     writer.write(tf_example.SerializeToString())
571 writer.close()

```

模型:

刚开始num_warmup_steps次迭代学习率偏小, 之后恢复设置的学习率

构建Embedding层==>

词向量编码 (和变换维度) + 位置和段从属关系嵌入 (相加) + 层归一化 (和dropout)

构建Transformer层 (encoder, 自注意机制) ==>

词向量扩维 (目的是为了使每一个词对应每一个句子; 同时创建mask标记有效性, 即实现自注意) + 多头机制 (每个头专注于N种特征/输出的提取, 后面是对每一个头的处理) + Q矩阵初始化 (len(输入) × len(特征)) + K矩阵初始化 (len(输出) × len(特征)) + V矩阵初始化 (len(输出) × len(特征)) + QK内积 (接着结合mask标记运算, 类似于单神经网络) + 结果与V内积 + (池化+训练等)

分词

| | | |
|----|------|-------|
| 1 | 嘉宝 | 20471 |
| 2 | 数量 | 16098 |
| 3 | 米粉 | 14468 |
| 4 | 营养 | 13848 |
| 5 | 单价 | 9770 |
| 6 | 小计 | 6091 |
| 7 | 原价 | 5995 |
| 8 | 收银员 | 5990 |
| 9 | 金额 | 5892 |
| 10 | 店 | 5773 |
| 11 | 合计 | 5698 |
| 12 | 品名 | 5465 |
| 13 | 元 | 5194 |
| 14 | 商品 | 5142 |
| 15 | 电话 | 4547 |
| 16 | 找零 | 4245 |
| 17 | 实收 | 4232 |
| 18 | 谢谢惠顾 | 4063 |
| 19 | 营业员 | 4005 |
| 20 | 现金 | 3863 |
| 21 | 号 | 3827 |
| 22 | 货号 | 3806 |
| 23 | 配方 | 3599 |
| 24 | 付款 | 3474 |
| 25 | 时间 | 3461 |
| 26 | 单号 | 3360 |

```

cut_words_util.py × test.py × handle.py × words_handle.txt ×
613 已买商品总数 100000 TN
614 已买商品总数 100000 TN
615 总计 100000 TN,TM_1
616 总计(件) 100000 TN
617 总计*件 100000 TN
618 总件数 100000 TN
619 总数 100000 TN
620 总数量 100000 TN
621 本单金额 100000 TM_1
622 标价金额 100000 TM_1
623 订单金额 100000 TM_1
624 合计 100000 TM_1,TN,PE
625 合计(RMB) 100000 TM_1
626 合计金额 100000 TM_1
627 合计总额 100000 TM_1
628 金额 100000 TM_1,MY
629 商品合计 100000 TM_1
630 商品总价 100000 TM_1
631 实价小计 100000 TM_1
632 现价 100000 TM_1,PE,OW

```

000b847ad1c04b81af8a3632a1ba69e4.txt - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

1
1
门售甲
1
品宝铁锌仍米粉2253.0058.00

000ba5d0ac3e40edbb8710cf82446bac.txt - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

贯日漂亮宝贝中央公园店(0040
[P]英氏美食加淮?135.8022.8122.81
[F]英氏美食加南?75.8022.0122.81
[P]黄氏有机牛肉?1480030.530.58
[F]英氏美食加南?35.0022.802230
大写叁佰叁拾伍元

27.txt - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

钙铁锌营养麦粉
59177
77
圆员员

附：

分布相似性==>密集型向量；与词汇相对位置无关

[skip-gram](#)：利用中心词来预测上下文。

1、利用one-hot编码将单词转换为向量，维度为[1, V] (V一般表示总次数)

2、与权重向量 $W[V, N]$ (N 是上下文窗口) 点积运算, 无激活函数, 最后得到 v 个向量 $H[1, N]$ (相对每一个上下文词都会生成一个向量)

3、与输出层权重向量 $W'[N, V]$ 点积得到向量 $U[1, V]$, 再经过softmax函数得到one-hot编码形式的预测向量。概率最大的那个单词就是结果, 若预测有误, 会通过反向传播算法来修正权重矩阵 W 和 W' 。

