2.1 Implement the WordPiece algorithm

(1)

Tokenization algorithm 是计算机理解人类语言的必要步骤，大致可以分为 word, subword, character-base 的方法。 WordPiece algorithm 属于 subword-based 方法，解决了 word-based tokenization 方法产生 vocabulary 过大，容易发生 out of vocabulary 以及相似词意义不同的问题；也解决了character-based tokenization 方法序列太长，单个 token 表示意思太少的问题。

WordPiece algorithm的核心在于，在构建子词词表的同时保证训练语料在该词表下的总似然最大化，可以用下面的公式表示：

$$\mathcal{L} = \sum_{i=1}^{N} \log\ P(w_i) = \sum_{i=1}^{N} \sum_{j=1}^{m_i} \log\ P(t_{ij})$$

其中，

w_i : 第i个词
t_{ij} : 第i个词被分解出的第j个子词
$P(t_{ij})$: 子词t_{ij}的概率
m_i : 词w_i的子词个数

算法主要分为两个部分：训练和分词。

在训练中，先把训练语料中的词分为单个字符序列，非开头字符要加上前缀##，比如 word 被分解为：

["w", "##o", "##r", "##d"]

然后，每次按某种标准选择合并后似然最大的子词，接着将合并后的新子词加入词表，一直重复直到词表大小达到预设值。这里所说的某种标准指的是计算如下两个待合并子词的得分：

$$score = \frac{frequence\ of\ pair}{frequence\ of\ first\ element \times frequence\ of\ second\ element}$$

例如，"h"在训练语料中出现4次，"##u"在训练语料中出现4次，"hu"在训练预料中出现4次，那么合并 "h"和"##u"的得分是4/(4*4)=0.25。

在分词中，在已经构建好的词库中寻找子词，使之与输入单词有最长公共前缀，重复这个步骤，直到完成分词。如果在分词中的某一步发现词库中没有与之相匹配的子词，那么这个输入单词的分词结果是["[UNK]"]。

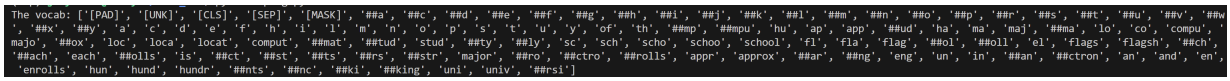
Reference: WordPiece: Subword-based tokenization algorithm, Word, Subword, and Character-Based Tokenization: Know the Difference, huggingface course

(2)

根据huggingface官方文档补全wpalg.py。

(3)

运行补充完整的wpalg.py，得到用所给语料训练的词汇表：



用这个词表对所给句子的分词结果是：

['n', '##o', '##u', '##s', 'e', '##tud', '##i', '##o', '##n', '##s', 'a', 'l', 'univ', '##e', '##rsi', '##t', '##e', 'd', '##e', 'p', '##e', '##ki', '##n']

(4)

对文本：I love apples 进行 tokenize 会产生带有 [UNK] 的 token 序列: ['[UNK]', 'lo', '##v', '##e', 'app', '##l', '##e', '##s']

Llama 的 tokenizer 使用了基于UTF-8编码的BPE算法，词表包含了所有可能的UTF-8字节，当遇到超出词汇表的未知词时，tokenizer 会把它拆解到字节级别，然后使用字节级别的token表示每个字节，无需使用[UNK]来处理未知词，从而避免了[UNK]的产生。

2.2 Expand BERT's tokenizer with WordPiece

1.

在step1中，我先初始化了一个WordPiece实列，然后用tokenizers库中的WordPieceTrainer，以 pubmed_sampled_corpus.jsonline作为语料库训练我的tokenizer，训练参数为：词表大小预设值是30000，一对词语要被合并的最小频率是2，特殊符号包含"[UNK]", "[CLS]", "[SEP]", "[PAD]", "[MASK]"，最终的到的词表长度为30000。

```
The size of resulting vocabulary:  30000
```

2.

选择策略：先从训练好的tokenzier词表中过滤掉原始bert-base-uncased词表中已经存在的token，去除特殊符号 "[UNK]", "[CLS]", "[SEP]", "[PAD]", "[MASK]"；然后按照token的长度从大到小排序，选择最长的5000个 token。

采样结果如下：

['##letion', 'multicentre', 'semiconduc', '##plantation', '##triction', 'Measurement', 'obstructive', 'Examination', 'Treatments', '##vities', 'Reporting', '##ographical', '##oprotective', 'Formation', 'genotypes', 'Experiments', 'intestine', 'Transcription', '##oprecip', 'Jorgensen', 'photoreceptor', 'aspiration', 'synchronous', '##aploid', '##International', 'Anderson', '##ogenetics', 'cytomegalovirus', '##related', '##empfer', '##otechn', 'Pseudomonas', 'Variations', '##tification', 'Problems', '##olytic', '##ashima', 'bicarbonate', 'inferences', 'artificially', '##assium', 'Musculoskeletal', 'angiography', 'transplanted', '##ulatory', '##emiology', 'Strength', 'clinically', 'percentile', 'Rosenberg']

特别的观察：

抽样结果中的token包含很多完整的长单词和有意义的前缀后缀，其中有一些医学专业名词，比如： genotypes 基因型，Jorgensen 巨病毒细胞，Pseudomonas 假单胞菌 等，也有一些有意义的人名，比如Jorgensen，根据资料显示(Figure 1)，这个人名可能对下游分类任务有用。

注：训练好的 expanded BERT tokenizer 保存在 ./expanded_BERT_tokenizer 中，可以通过transformers.BertTokenizer.from_pretrained()来加载。

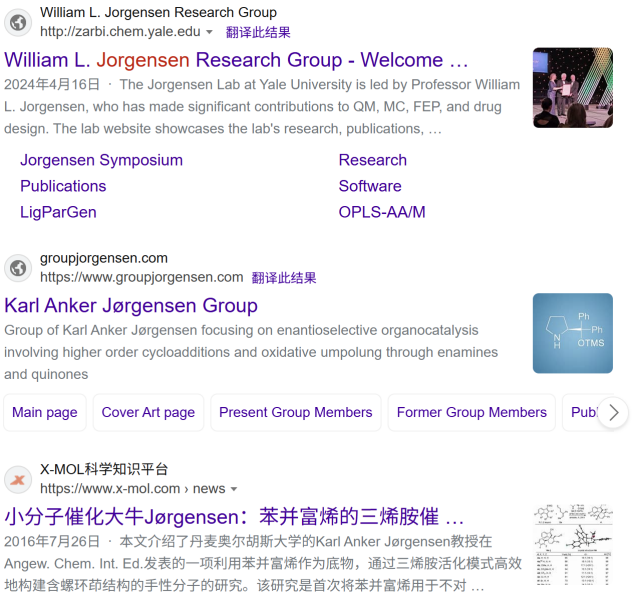


图 1: 疑似Jorgensen相关的资料

3.

此问题的测试代码在test.py中。

句子：The enhancement of frequency of activation-induced apoptosis (up to 244%) was observed at 4.2-83 nM Onc concentration , which is at least an order magnitude lower than its minimal concentration reported to affect proliferation or induce apoptosis of leukemic and solid tumor cell lines .

original bert tokenizer: `['the', 'enhancement', 'of', 'frequency', 'of', 'activation', '-', 'induced', 'ap', '##op', '##tosis', '(', 'up', 'to', '244', '%', ')', 'was', 'observed', 'at', '4', '.', '2', '-', '83', 'nm', 'on', '##c', 'concentration', ',', 'which', 'is', 'at', 'least', 'an', 'order', 'magnitude', 'lower', 'than', 'its', 'minimal', 'concentration', 'reported', 'to', 'affect', 'proliferation', 'or', 'induce', 'ap', '##op', '##tosis', 'of', 'le', '##uke', '##mic', 'and', 'solid', 'tumor', 'cell', 'lines', '.']`

expanded bert tokenizer: ['the', 'enhancement', 'of', 'frequency', 'of', 'activation', '-', 'induced', 'apoptosis', '(', 'up', 'to', '244', '%', ')', 'was', 'observed', 'at', '4', '.', '2', '-', '83', 'nm', 'on', '##c', 'concentration', ',', 'which', 'is', 'at', 'least', 'an', 'order', 'magnitude', 'lower', 'than', 'its', 'minimal', 'concentration', 'reported', 'to', 'affect', 'proliferation', 'or', 'induce', 'apoptosis', 'of', 'le', '##uke', '##mic', 'and', 'solid', 'tumor', 'cell', 'lines', '.']

句子：BACKGROUND/AIMS by reducing the number of ATP molecules produced via aerobic glycolysis , the inhibition of lactic dehydrogenase (LDH) should hinder the growth of neoplastic cells without damaging the normal cells which do not rely on this metabolic pathway for their energetic needs .

original bert tokenizer: ['background', '/', 'aims', 'by', 'reducing', 'the', 'number', 'of', 'atp', 'molecules', 'produced', 'via', 'aero', '##bic', 'g', '##ly', '##col', '##ysis', ',', 'the', 'inhibition', 'of', 'lac', '##tic', 'de', '##hy', '##dro', '##genase', '(', 'ld', '##h', ')', 'should', 'hind', '##er', 'the', 'growth', 'of', 'neo', '##pl', '##astic', 'cells', 'without', 'damaging', 'the', 'normal', 'cells', 'which', 'do', 'not', 'rely', 'on', 'this', 'metabolic', 'pathway', 'for', 'their', 'energetic', 'needs', '.']

expanded bert tokenizer: ['background', '/', 'aims', 'by', 'reducing', 'the', 'number', 'of', 'atp', 'molecules', 'produced', 'via', 'aero', '##bic', 'glycolysis', ',', 'the', 'inhibition', 'of', 'lac', '##tic', 'dehydrogenase', '(', 'ld', '##h', ')', 'should', 'hind', '##er', 'the', 'growth', 'of', 'neoplastic', 'cells', 'without', 'damaging', 'the', 'normal', 'cells', 'which', 'do', 'not', 'rely', 'on', 'this', 'metabolic', 'pathway', 'for', 'their', 'energetic', 'needs', '.']

句子：TNFalpha mediated increase in Akt phosphorylation was dependent on oxidative stress as Akt phosphorylation was abrogated in the presence of ROS inhibitor and elevated in cells transfected with SOD-1 siRNA .

original bert tokenizer: ['tn', '##fa', '##lp', '##ha', 'mediated', 'increase', 'in', 'ak', '##t', 'ph', '##os', '##ph', '##ory', '##lation', 'was', 'dependent', 'on', 'ox', '##ida', '##tive', 'stress', 'as', 'ak', '##t', 'ph', '##os', '##ph', '##ory', '##lation', 'was', 'ab', '##rogated', 'in', 'the', 'presence', 'of', 'ro', '##s', 'inhibitor', 'and', 'elevated', 'in', 'cells', 'trans', '##fect', '##ed', 'with', 'so', '##d', '-', '1', 'sir', '##na', '.']

expanded bert tokenizer: ['tn', '##fa', '##lp', '##ha', 'mediated', 'increase', 'in', 'ak', '##t', 'phosphorylation', 'was', 'dependent', 'on', 'oxidative', 'stress', 'as', 'ak', '##t', 'phosphorylation', 'was', 'ab', '##rogated', 'in', 'the', 'presence', 'of', 'ro', '##s', 'inhibitor', 'and', 'elevated', 'in', 'cells', 'transfected', 'with', 'so', '##d', '-', '1', 'sir', '##na', '.']

容易看出与 original bert tokenizer 相比，扩展后的 tokenzier 对长单词尤其是一些专有名词的识别能力增强，tokens中对一个单词的拆分减少, 即##xx的结构变少。

在HoC训练集上，original bert tokenizer 分词后的平均句子长度更长，expanded bert tokenizer 分词后的平均句子长度更短，如图：

```
original average length:  67.53925925925925
expanded average length:  62.260740740740744
```

4.

在bert-base-uncased模型中，每个token的embedding size是768，我为词表新加入了5000个token，所以新增的参数量是5000*768=3840000。我使用了huggingface提供的model.resize_token_embeddings()方法扩展embedding层，根据官方文档的说明，这里采用了原embedding层参数的均值和方差确定的正态分布初始化这些新的参数。

5.

在HoC数据集上训练，训练参数为:，得到的结果如图(fig:2)。分类的结果比original BERT model要差一些。

分析：可以看出虽然 expanded bert tokenizer 对句子的 tokenize 结果要更好，但是最后在HoC数据集上的分类效果要比直接使用 original bert tokenizer 要差不少，我认为主要原因是我在 step 4 中对新参数的初始化方法不恰当，直接使用resize_token_embeddings()用原有embedding层参数均值和方差确定的正态分布初始化新token对应的参数，这样明显不妥，可能会使得新加入的token无法正确表达意思，甚至影响原有token的表意能力；其次，HoC数据集比较小，使用了expanded bert tokenizer 的 bert-base-uncased 模型的参数量更大，可能会在训练中发生一定程度的过拟合，导致最终在测试集上的表现变差。

图 2: expanded bert model 在HoC上的分类结果