

# 实验三：机器翻译

## 一、使用的数据集

训练集

网站<http://www.statmt.org/wmt16/translation-task.html> 中从 WMT Newscrawl 2007-14 语料库合并作为德语->英语神经机器翻译的训练数据。

概况：

Sentence: 4208440

Token(English): 83971668

测试集

- [Development sets \(22 MB\)](#)
- [Updated Romanian dev. with fixed diacritics \(0.5 MB\)](#)
- **NEW: [Test sets \(3.4 MB\)](#)** Test sets (source and references)

实际运行中抽取了50000句子进行训练

## 二、预处理

### tokenize

#### 1.建立词典库

构建training data的词典库（包含所有的单词 每个token占一行

安装subword-nmt进行将文本分割为子词单元的预处理操作

每种语言选择一个词汇表。通常，会选择一个词汇量为  $v$  的词，并且只有最常用的  $v$  词才会被视为唯一的。

```
usage: subword-nmt learn-joint-bpe-and-vocab [-h] --input PATH [PATH ...]
--output PATH [--symbols SYMBOLS]
[--separator STR]
--write-vocabulary PATH
[PATH ...] [--min-frequency FREQ]
[--total-symbols] [--verbose]
```

运行过程

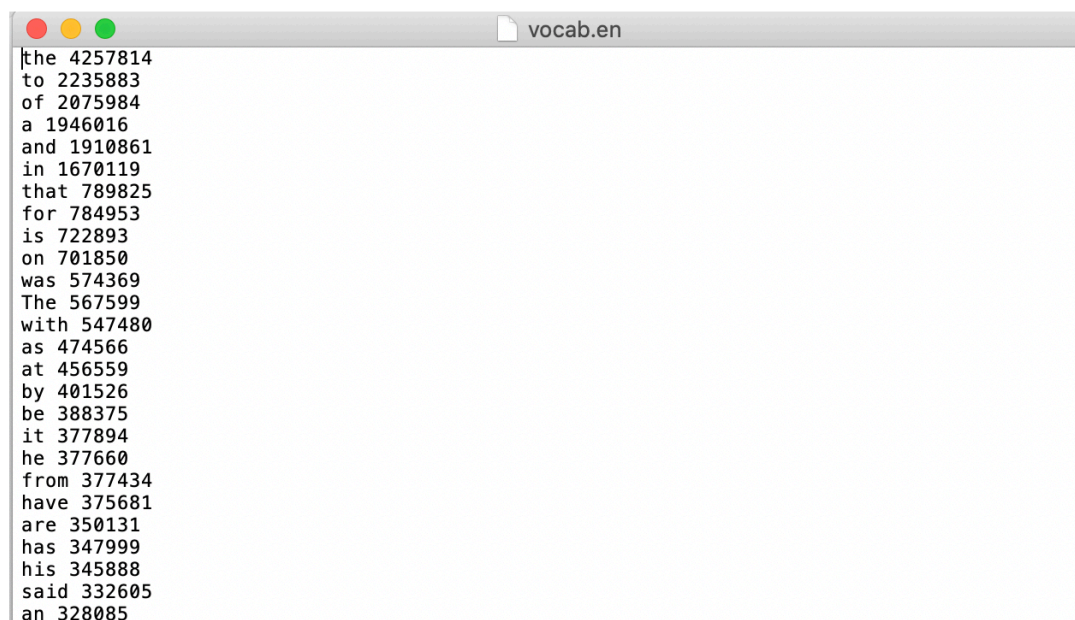
第一轮先尝试训练得到最高频的32000个单词

```
(base) wangwenqingdeMacBook-Pro:Machine-translation wangwenqing$ subword-nmt learn-joint-bpe-and-vocab -i ./news.bt.de-en.en -s 32000
-o ./bpe/bpe32k.en --write-vocabulary ./bpe/vocab.en
(base) wangwenqingdeMacBook-Pro:Machine-translation wangwenqing$ subword-nmt learn-joint-bpe-and-vocab -i ./news.bt.de-en.de -s 32000
-o ./bpe/bpe32k.de --write-vocabulary ./bpe/vocab.de
(base) wangwenqingdeMacBook-Pro:Machine-translation wangwenqing$
```

后续还尝试训练得到最高频的50000个单词

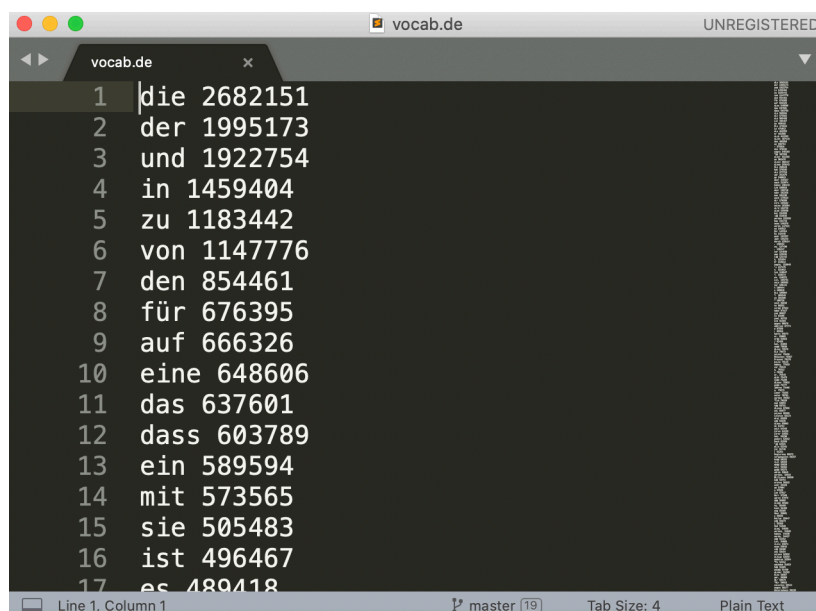
查看运行结果

## 英文词典



```
the 4257814
to 2235883
of 2075984
a 1946016
and 1910861
in 1670119
that 789825
for 784953
is 722893
on 701850
was 574369
The 567599
with 547480
as 474566
at 456559
by 401526
be 388375
it 377894
he 377660
from 377434
have 375681
are 350131
has 347999
his 345888
said 332605
an 328085
```

## 德文词典



```
1 die 2682151
2 der 1995173
3 und 1922754
4 in 1459404
5 zu 1183442
6 von 1147776
7 den 854461
8 für 676395
9 auf 666326
10 eine 648606
11 das 637601
12 dass 603789
13 ein 589594
14 mit 573565
15 sie 505483
16 ist 496467
17 es 489418
```

另外在词典里加入三个特殊标记

**<unk>** 标记没有出现在词典里的词

**<s>** 标记一句话的开始

**</s>** 标记一句话结束

## 2. 将字典的单词和数字进行映射

对于两种字典的数据格式需要进行不同的处理。对于上面的字典，其中包含了词频所以需要取出单词

src\_dict将德语字典中的单词映射为数字，src\_rev\_dict将数字映射成单词，目标字典的映射同理

以将源中的单词进行映射为例

```

src_dict = dict()
with open(filename_de, encoding='utf-8') as f:
    for line in f:
        # save the first word, not the number
        line=line.split(" ")
        line=line[0]
        # print(line)
        src_dict[line[:-1]] = len(src_dict)

src_rev_dict = dict(zip(src_dict.values(),src_dict.keys()))

```

### 3.对训练的句子进行tokenize

将句子拆分为其组成文字（单词、符号），省略最后一个换行符。

用source字段来区分token是在源字典还是目标字典中

在调用时对应设置为true/false

```

src_sent_tokens = tokenize(src_sent,True,src_dictionary,tgt_dictionary)
tgt_sent_tokens = tokenize(tgt_sent,False,src_dictionary,tgt_dictionary)

```


如果key未在字典中出现则将当前token设置为

```

def tokenize(sentence,source,src_dict,tgt_dict):
    sentence = sentence.replace('-', ' ')
    sentence = sentence.replace('\n', ' ')
    sentence = sentence.replace(',', ' ,')
    sentence = sentence.replace('.', ' .')
    tokens = sentence.split(' ')
    for t, key in enumerate(tokens):
        if source:
            if key not in src_dict.keys():
                tokens[t] = '<unk>'
        else:
            if key not in tgt_dict.keys():
                tokens[t] = '<unk>'
    return tokens

```

### 3.保证句子长度相同

通过填充  使所有相同长度的句子达到最大长度，如果超过，则将其截断

对于源和目标的句子处理类似不做赘述

```

num_src_sent = []
for tok in src_sent_tokens:
    num_src_sent.append(src_dictionary[tok])
num_src_set = num_src_sent[::-1]
num_src_sent.insert(0,src_dictionary['<s>'])

train_input_size.append(min(len(num_src_sent)+1,src_max_sentence_length))

if len(num_src_sent)<src_max_sentence_length:
    num_src_sent.extend([src_dictionary['</s>'] for _ in
range(src_max_sentence_length - len(num_src_sent))])

elif len(num_src_sent)>src_max_sentence_length:
    num_src_sent = num_src_sent[:src_max_sentence_length]

```

#### 4.产生batch of data

使用 unroll 函数以 [source\_sequence\_length, batch\_size, embedding\_size] 格式生成一批数据。其输入数据来自 next\_batch 函数。

```

def next_batch(self, sent_ids, first_set):

    if self._is_source:
        max_sent_length = src_max_sentence_length
    else:
        max_sent_length = tgt_max_sentence_length
    batch_labels_ind = []
    batch_data = np.zeros((self._batch_size),dtype=np.float32)
    batch_labels = np.zeros((self._batch_size),dtype=np.float32)

    for b in range(self._batch_size):

        sent_id = sent_ids[b]
        if self._is_source:
            sent_text = self.train_inputs[sent_id]

            batch_data[b] = sent_text[self._pointer[b]]
            batch_labels[b]=sent_text[self._pointer[b]+1]
        else:
            sent_text = self.train_outputs[sent_id]
            if sent_text[self._pointer[b]]!=self.src_dictionary['<s>']:
                batch_data[b] = sent_text[self._pointer[b]]
            else:
                batch_data[b] = sent_text[self._pointer[b]]
                batch_labels[b] = sent_text[self._pointer[b]+1]

        self._pointer[b] = (self._pointer[b]+1)%(max_sent_length-1)

    return batch_data,batch_labels

```

```

def unroll_batches(self, sent_ids, train_inp_lengths):

    if sent_ids is not None:

        self._sent_ids = sent_ids

        self._pointer = [0 for _ in range(self._batch_size)]

    unroll_data, unroll_labels = [], []
    inp_lengths = None
    for j in range(self._num_unroll):
        # The first batch in any batch of captions is different
        if self._is_source:
            data, labels = self.next_batch(self._sent_ids, False)
        else:
            data, labels = self.next_batch(self._sent_ids, False)

        unroll_data.append(data)
        unroll_labels.append(labels)
        inp_lengths = train_inp_lengths[sent_ids]
    return unroll_data, unroll_labels, self._sent_ids, inp_lengths

```

## 三、训练过程

### 1. 相关参数设置

batch size 设置为 60

最长读取的句子长度 60

词典单词总数 50000

读取的句子数 50000

### 2. Embedding

为了使这个嵌入层能够工作，首先为每种语言选择一个词汇表(这个在预处理中已经完成)。除词表中的词外，所有其他单词都被转换为“未知”标记，并且得到相同的嵌入。每种语言一组嵌入权值通常是在训练过程中学习的。

#### 初始化嵌入权重

使用 word2vec

pip install --upgrade gensim

```

model=Word2Vec("preprocess.en",vector_size=100,window=5,min_count=2,negative=3
,sample=0.001,hs=1,workers=4)
model.save("englishembedding.npy")
print("embedding model en saved")

```

### 3.Encoder（利用Istm）

一旦检索到词嵌入，将词嵌入作为主网络的输入，主网络由两个multi-layer RNN组成——源语言的编码器和目标语言的解码器。这两个神经网络，原则上，可以共享相同的权重；然而，在实践中，我们经常使用两个不同的神经网络参数(这种模型在拟合大型训练数据集时效果更好)。编码器 RNN 使用零向量作为其起始状态，构建如下：

```

# Build RNN cell
encoder_cell = tf.nn.rnn_cell.BasicLSTMCell(num_units)
initial_state = encoder_cell.zero_state(batch_size, dtype=tf.float32)
# Run Dynamic RNN
encoder_outputs, encoder_state = tf.nn.dynamic_rnn(
    encoder_cell, encoder_emb_inp, initial_state=initial_state,
    sequence_length=source_sequence_length,
    time_major=True, swap_memory=False)

```

为了避免浪费计算，句子有不同的长度，通过 source\_sequence\_length 告诉 dynamic\_rnn 准确的源句长度。因为我们的输入是时间主要，我们设置time\_major=True

#### encoder中的数据预处理

Unroll batch产生[source\_sequence\_length, batch\_size, embedding\_size] 形式的数据

```

eu_data, eu_labels, _, eu_lengths =
enc_data_generator.unroll_batches(sent_ids=sent_ids,train_inp_lengths =
train_inp_lengths)

feed_dict = {}
feed_dict[enc_train_inp_lengths] = eu_lengths
for j,(dat,lbl) in enumerate(zip(eu_data,eu_labels)):
    feed_dict[enc_train_inputs[j]] = dat

```

### 4.Decoder（利用Istm+Luong Attention）

步骤如下

第一步是定义attention机制

第二步是定义基础的RNNCell

第三步是使用AttentionWrapper进行封装

**解码器**

解码器也需要访问源信息，一个简单的实现方法是用编码器的最后一个隐藏状态来初始化它。

```
# Build RNN cell
decoder_cell = tf.nn.rnn_cell.BasicLSTMCell(num_units)

# Helper
helper = tf.contrib.seq2seq.TrainingHelper(
    decoder_emb_inp, [tgt_max_sentence_length-1 for _ in range(batch_size)],
    time_major=True)
# Decoder
training_decoder = tf.contrib.seq2seq.BasicDecoder(
    cell = decoder_cell,
    helper = helper,
    initial_state =
decoder_cell.zero_state(batch_size,tf.float32).clone(cell_state=encoder_state)
,
    output_layer = projection_layer)
# Dynamic decoding
outputs, _, _ = tf.contrib.seq2seq.dynamic_decode(
    training_decoder, output_time_major=True,
    swap_memory=False
)
```

这里解码的核心decoder接收decoder\_cell， helper以及encoder\_state作为输入

helper用来确定decoder部分的输入，训练过程应直接使用上一时刻的真实值作为下一时刻输入，预测过程中可以使用贪婪的方法选择概率最大的那个值作为下一时刻的输入

## 投影层

它是一个稠密矩阵，可以将最高隐藏状态转化为维数  $v$  的对数向量

```
projection_layer = Dense(units=vocab_size, use_bias=True)
```

## Attention

AttentionMechanism 中保存了 memory bank（在 seq2seq 中就是 encoder 各个时间步的输出组成的张量），同时其中还定义了注意力的计算方法（这里为 Luong's style）

AttentionWrapper 构造时要接受一个 AttentionMechanism 的实例

```
attention_mechanism = tf.contrib.seq2seq.LuongAttention(
    num_units = 128,
    memory = encoder_outputs,
    memory_sequence_length = source_sequence_length)
```

## decoder中的数据预处理

```

du_data, du_labels, _, du_lengths =
dec_data_generator.unroll_batches(sent_ids=sent_ids, train_inp_lengths =
train_inp_lengths)

feed_dict[dec_train_inp_lengths] = du_lengths
for j, (dat, lbl) in enumerate(zip(du_data, du_labels)):
    feed_dict[dec_train_inputs[j]] = dat
    feed_dict[dec_train_labels[j]] = lbl
    feed_dict[dec_label_masks[j]] = (np.array([j for _ in
range(batch_size)]) < du_lengths).astype(np.int32)

```

## 5.损失(交叉熵)

`sparse_softmax_cross_entropy_with_logits`函数传入的logits为神经网络输出层的输出，shape为[batch\_size, num\_classes]，传入的label为一个一维的vector，长度等于batch\_size，每一个值的取值区间必须是[0, num\_classes)，其实每一个值就是代表了batch中对应样本的类别。

在损失中除了批量大小，因此我们的超参数对批量大小是“不变的”。同时还除了目标句子的长度。

```

crossent =
tf.nn.sparse_softmax_cross_entropy_with_logits(labels=dec_train_labels,
logits=logits)
loss = (tf.reduce_sum(crossent*tf.stack(dec_label_masks)) /
(batch_size*target_sequence_length))

```

## 6.梯度计算和优化

反向传播

```

adam_gradients, v = zip(*adam_optimizer.compute_gradients(loss))
adam_gradients, _ = tf.clip_by_global_norm(adam_gradients, 25.0)

```

优化器选择adam。

```

adam_optimizer = tf.compat.v1.train.AdamOptimizer(learning_rate)
adam_optimize = adam_optimizer.apply_gradients(zip(adam_gradients, v))

```

学习率的值通常在0.0001到0.001之间，并且可以设置为随着训练的进展而减少。这里设置为指数级减少。

```

learning_rate = tf.compat.v1.train.exponential_decay(
0.01, global_step, decay_steps=10, decay_rate=0.9, staircase=True)

```

## 7.训练记录

本次训练共设置为10500 step



由于使用的是mac电脑，没有gpu,所以用cpu跑了两天

对于训练结果每100轮打印一次实际的语句和翻译后语句

每500轮使用 tf.compat.v1.train.Saver()保存一次模型

```
if (step+1)%500==0:
    print('Step ', str(step+1))
    print('\t Loss: ', avg_loss/500.0)

    save_path = saver.save(sess, "logs/testmodel")
    print("Model saved in path: %s" % save_path)
    avg_file.write(str(avg_loss))
```

训练到800 step的时候已经初有成效

```
Actual: If cancelled up to 28 days before date of arrival <unk> , no fee will be charged <unk> . <unk> </s>
Predicted: The cancelled up to the : <unk> date of arrival <unk> , the fee will be charged <unk> . <unk> </s>
```

bit ('base': conda) 0 0 Ln 104, Col 9 Spaces: 4 UTF-8 LF Python

训练到1000 step

```
.....

Step 1000
Actual: In case of no ##AT## ##AT## show <unk> , the total price of the reservation will be charged <unk> . <unk> </s>
Predicted: <unk> the of the <unk> ##AT## <unk> <unk> , the <unk> price of the <unk> <unk> be <unk> <unk> . <unk> </s>

Actual: If cancelled up to 4 days before date of arrival <unk> , no fee will be charged <unk> . <unk> </s>
Predicted: Hotel cancelled or to 12 days before date of arrival <unk> , the fee will be charged <unk> . <unk> </s>

Step 1000
Loss: 1.945605410337448
Model saved in path: logs/testmodel
```

bit ('base': conda) 0 0 Ln 104, Col 9 Spaces: 4 UTF-8 LF Python

训练2200 step后

```
Actual: All children under 12 years are charged GBP 20 <unk> per night for extra beds <unk> . <unk> </s>
Predicted: All older from 3 to stay charged 70 8 <unk> per night and extra beds <unk> . <unk> </s>
```

bit ('base': conda) 0 0 Ln 104, Col 9 Spaces: 4 UTF-8 LF Python

训练4000 step后

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 2: Python

.....

Step 4000
Actual: Tennis Court <unk> , Sauna <unk> , Fitness Centre <unk> , Golf Course ( within 3 km ) <unk> , Games R
oom <unk> , Solarium <unk> , Spa & Wellness Centre <unk> , Children &s Playground <unk> , Jacuzzi <u
nk> , Turkish / Steam Bath <unk> , Hammam <unk> , Mini Golf <unk> , Outdoor Swimming Pool <unk> . <unk>
Predicted: Tennis <unk> <unk> , Sauna <unk> , <unk> Centre <unk> , <unk> Course <unk> 24 3 km ) <unk> , Table
de <unk> , Table <unk> , Children & Wellness Centre <unk> , Spa <unk> Playground <unk> , Table and , Hik
ing / coffee and <unk> , Outdoor <unk> , Hammam Exchange <unk> , Car Swimming Pool <unk> , <unk>
```

bit ('base': conda) 0 0 Ln 84, Col 82 (153 selected) Spaces: 4 UTF-8 LF Python

训练5100 step后

```
Actual: The LateRooms rates for White Hart Inn in <unk> are the total price of the room and not the &apos; per person &apos; rate <unk> . <unk> </s>
Predicted: The LateRooms rates for Number House in Plymouth <unk> are the total price of the room and not the &apos; per person &apos; rate <unk> . <unk> </s>
```

训练7900 step后

```
Actual: Wireless Internet Hotspot is available in public areas and costs EUR 2 <unk> per 30 minutes <unk> . <unk> </s>
Predicted: Wireless internet Hotspot is available in public areas and costs EUR 6 <unk> per minute <unk> <unk> > . <unk> </s>
```

训练9000 step后

```
Actual: The hotel offers comfortable and attractively furnished rooms fitted with all modern comforts for both business and leisure guests <unk> . <unk> </s>
Predicted: The <unk> &apos;s a accommodation attractively rooms rooms <unk> with a modern comforts <unk> the business and leisure travellers <unk> . <unk> </s>

Step 9000
Loss: 1.413284642457962
Model saved in path: logs/testmodel
```

训练9200 step后

```
Actual: When would you like to stay at the Hotel <unk> Royal ? <unk> </s>
Predicted: When would you like to stay at the Hotel <unk> ? Gardens <unk> </s>
```

训练9700 step后

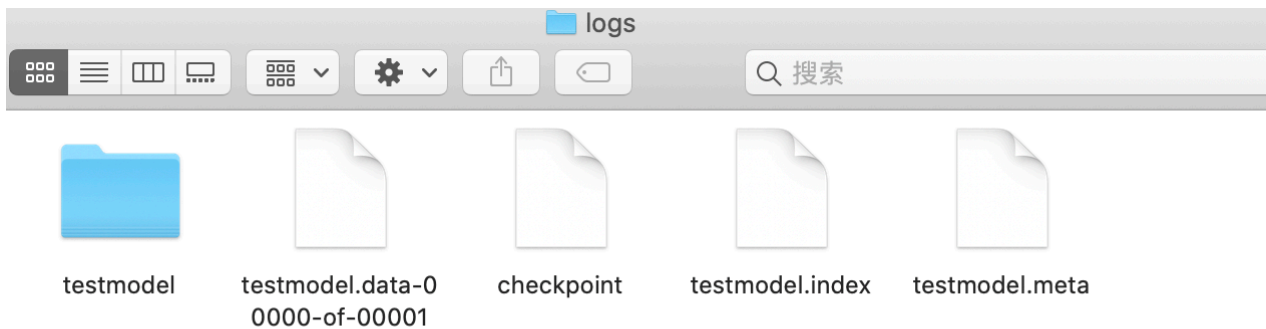
```
Step 9700
Actual: A rich breakfast buffet is served every morning in the <unk> restaurant <unk> , from 06 : 30 until 10 : 30 <unk> . <unk> </s>
Predicted: A buffet breakfast buffet breakfast served from morning and the morning Restaurant and , and the : 30 until 11 : 00 <unk> . <unk> </s>
```

训练10400 step后

训练出的翻译已经和正确的翻译十分相近

```
Step 10400
Actual: The hotel offers a wide range of facilities and services <unk> . <unk> </s>
Predicted: The hotel offers a wide range of facilities and a <unk> . <unk> </s>
```

保存的模型



## 四、测试和实验评测

### 数据预处理

测试数据中如图可以看到很多类似html格式的非文本数据，需要将他们预先处理成这里读入的 de\_dev.txt,en\_dev.txt

```
newstest2016-ende-ref.de.sgm
<krefset setid="newstest2016" srclang="any" trglang="de">
<doc sysid="ref" docid="Wirtschaftsblatt.at.22162" genre="news" origlang="de">
<p>
<seg id="1">Obama empfängt Netanyahu</seg>
<seg id="2">Das Verhältnis zwischen Obama und Netanyahu ist nicht gerade
freundschaftlich.</seg>
<seg id="3">Die beiden wollten über die Umsetzung der internationalen Vereinbarung sowie
über Teherans destabilisierende Maßnahmen im Nahen Osten sprechen.</seg>
<seg id="4">Bei der Begegnung soll es aber auch um den Konflikt mit den Palästinensern und
die diskutierte Zwei-Staaten-Lösung gehen.</seg>
<seg id="5">Das Verhältnis zwischen Obama und Netanyahu ist seit Jahren gespannt.</seg>
<seg id="6">Washington kritisiert den andauernden Siedlungsbau Israels und wirft Netanyahu
mangelnden Willen beim Friedensprozess vor.</seg>
<seg id="7">Durch den von Obama beworbenen Deal um das iranische Atomprogramm hat sich die
Beziehung der beiden weiter verschlechtert.</seg>
<seg id="8">Im März hatte Netanyahu auf Einladung der Republikaner vor dem US-Kongress
eine umstrittene Rede gehalten, die teils als Affront gegen Obama gewertet wurde.</seg>
<seg id="9">Die Rede war mit Obama nicht abgesprochen, ein Treffen hatte dieser mit
Hinweis auf die seinerzeit bevorstehende Wahl in Israel abgelehnt.</seg>
</p>
</doc>
<doc sysid="ref" docid="abcnews.151477" genre="news" origlang="en">
<p>
<seg id="1">In einem Notruf gesteht Professor, seine Freundin erschossen zu haben</seg>
<seg id="2">In einem Notruf erzählte Professor Shannon Lamb mit einer etwas zitterigen
Stimme der Polizei, dass er seine Freundin erschossen habe und dass die Beamten zu seinem
Haus kommen müssten.</seg>
<seg id="3">Lamb war es wichtig zu betonen, dass sein "süßer Hund" aber noch lebe und
```

处理后

de\_dev.txt

Obama empfängt Netanyahu  
Das Verhältnis zwischen Obama und Netanyahu ist nicht gerade freundschaftlich.  
Die beiden wollten über die Umsetzung der internationalen Vereinbarung sowie über Teherans destabilisierende Maßnahmen im Nahen Osten sprechen.  
Bei der Begegnung soll es aber auch um den Konflikt mit den Palästinensern und die diskutierte Zwei-Staaten-Lösung gehen.  
Das Verhältnis zwischen Obama und Netanyahu ist seit Jahren gespannt.  
Washington kritisiert den andauernden Siedlungsbau Israels und wirft Netanyahu mangelnden Willen beim Friedensprozess vor.  
Durch den von Obama beworbenen Deal um das iranische Atomprogramm hat sich die Beziehung der beiden weiter verschlechtert.  
Im März hatte Netanyahu auf Einladung der Republikaner vor dem US-Kongress eine umstrittene Rede gehalten, die teils als Affront gegen Obama gewertet wurde.  
Die Rede war mit Obama nicht abgesprochen, ein Treffen hatte dieser mit Hinweis auf die seinerzeit bevorstehende Wahl in Israel abgelehnt.  
In einem Notruf gesteht Professor, seine Freundin erschossen zu haben  
In einem Notruf erzählte Professor Shannon Lamb mit einer etwas zitterigen Stimme der Polizei, dass er seine Freundin erschossen habe und dass die Beamten zu seinem Haus kommen müssten.  
Lamb war es wichtig zu betonen, dass sein "süßer Hund" aber noch lebe und wahrscheinlich aufgeregt sei, und er sagte, die Familienkontakte der toten Frau könnten auf dem Handy gefunden werden.

## 测试

加载checkpoint, 恢复session, 并运行

```
with tf.Session() as sess:
    sess.run(tf.initialize_all_variables())
    ckpt = tf.train.get_checkpoint_state('./logs')
    if ckpt and ckpt.model_checkpoint_path:
        saver.restore(sess, ckpt.model_checkpoint_path)
        test_accuracy = evaluate(src_test_sent, tgt_test_sent)
        print(test_accuracy[0])

    else:
        pass
```

## bleu结果

由于选定的为随机选取step中的一条predicted sentence进行打印和比对, 所以对这个语句进行bleu计算

```
rand_idx = np.random.randint(low=1, high=batch_size)
print_str_predict = ' '
for w in np.concatenate(du_labels, axis=0):
    [rand_idx::batch_size].tolist():
        print_str_predict += tgt_reverse_dictionary[w] + ' '
        if tgt_reverse_dictionary[w] == '</s>':
            break
print(print_str_predict)

print_str_reference = ' '
for w in tr_pred[rand_idx::batch_size].tolist():
    print_str_reference += tgt_reverse_dictionary[w] + ' '
    if tgt_reverse_dictionary[w] == '</s>':
        break
```

```
print(print_str_reference)
print("bleu score",bleu(print_str_predict,print_str_reference,[1]))
```

并未专门对于测试集生成字典并加入训练，所以bleu效果不是特别好，这里是随机测试的bleu结果截图

bleu值多在0.15+的区间中

```
Until 31 May 2009 <unk> , all bookings for a classic room will be upgraded to a deluxe room <unk> , and all
bookings for a deluxe room will be upgraded to a junior suite <unk> . <unk> </s>
The the <unk> 2009 <unk> , the bookings are rent week hotel service be served to the 24 hotel <unk> , with t
he the for your day hotel <unk> be served to the day <unk> <unk> . <unk> </s>
bleu score 0.14210526315789473
```

```
The hotel has 54 rooms <unk> , well appointed and decorated in a contemporary style <unk> . <unk> </s>
The guest is a <unk> <unk> , the <unk> <unk> the <unk> the <unk> of <unk> , <unk> </s>
bleu score 0.15384615384615385
```

```
When would you like to stay at the <unk> Resort & Suites ? <unk> </s>
The would you like to stay at the heart <unk> <unk> <unk> <unk> <unk> </s>
bleu score 0.25333333333333335
```

```
We also have conference facilities for important business meetings and a fitness centre just 50 metres away
<unk> . <unk> </s>
The offer enjoy a rooms <unk> a facilities facilities <unk> services modern area <unk> <unk> metres from <un
k> . <unk> </s>
bleu score 0.1875
```

(base) conda

Ln 423, Col 9 Spaces: 4 UTF-8 LF Python 57

```
Please also read carefully the terms and conditions which apply for all bookings <unk> . <unk> </s>
Please note have the the <unk> of conditions of is to your the <unk> , <unk> </s>
bleu score 0.2079207920792079
```

```
You can also allow our native ##AT## ##AT## speaker agents to process client orders directly <unk> . <unk> <
/s>
The can choose find you information to ##AT## clip <unk> <unk> be <unk> <unk> <unk> <unk> . <unk> </s>
bleu score 0.20353982300884954
```

## 五、异常处理

1. Cannot load file containing pickled data when allow\_pickle=False

```
Traceback (most recent call last):
Accounts : "/Users/wangwenqing/Desktop/当代人工智能/Machine-translation/translate.py", line 292, in <module>
encoder_emb_layer = tf.convert_to_tensor(np.load('germanembedding.npy'))
File "/Users/wangwenqing/opt/anaconda3/lib/python3.7/site-packages/numpy/lib/npio.py", line 457, in load
raise ValueError("Cannot load file containing pickled data ")
ValueError: Cannot load file containing pickled data when allow_pickle=False
(base) wangwenqingdeMacBook-Pro:Machine-translation wangwenqing$
```

在np.load中添加allow\_pickle=True

2. Failed to convert object of type <class 'gensim.models.word2vec.Word2Vec'> to Tensor.  
Contents: Word2Vec(vocab=4, vector\_size=300, alpha=0.025)

```
allow_broadcast=allow_broadcast))
File "/Users/wangwenqing/opt/anaconda3/lib/python3.7/site-packages/tensorflow/python/framework/tensor_util.py", line 562, in make_t
ensor_proto
"supported type." % (type(values), values))
TypeError: Failed to convert object of type <class 'gensim.models.word2vec.Word2Vec'> to Tensor. Contents: Word2Vec(vocab=4, vector_s
ize=300, alpha=0.025). Consider casting elements to a supported type.
(base) wangwenqingdeMacBook-Pro:Machine-translation wangwenqing$
```

由于数据量和训练效率影响，实际训练的单词数改为50000，并将词向量维度减小

## 六、参考链接

Tutorial :<https://github.com/tensorflow/nmt>

<https://google.github.io/seq2seq/nmt/>

<https://arxiv.org/pdf/1606.02891v2.pdf>

<https://github.com/rsennrich/subword-nmt>

<https://blog.csdn.net/guolindonggld/article/details/56966200>