

# Subword Models

2020年6月26日 14:49

## ##00\_背景

机器翻译就是把一种语言翻译成另外一种语言，如下图所示。句子用source标记的，即源语言，用target标记的，即目标语言，机器翻译的任务就是把源语言的句子翻译为目标语言的句子。在基于神经网络的机器翻译中，将卷积网络和subword加入其中能够提取丰富的语义和结构信息。SMT是在神经网络之前最主流的翻译模式，统计机器翻译。

### 机器翻译做什么？

Source:

我 在 周 日 看 了 一 本 书



Target:

I read a book on Sunday

在目前的机器翻译面临的挑战：

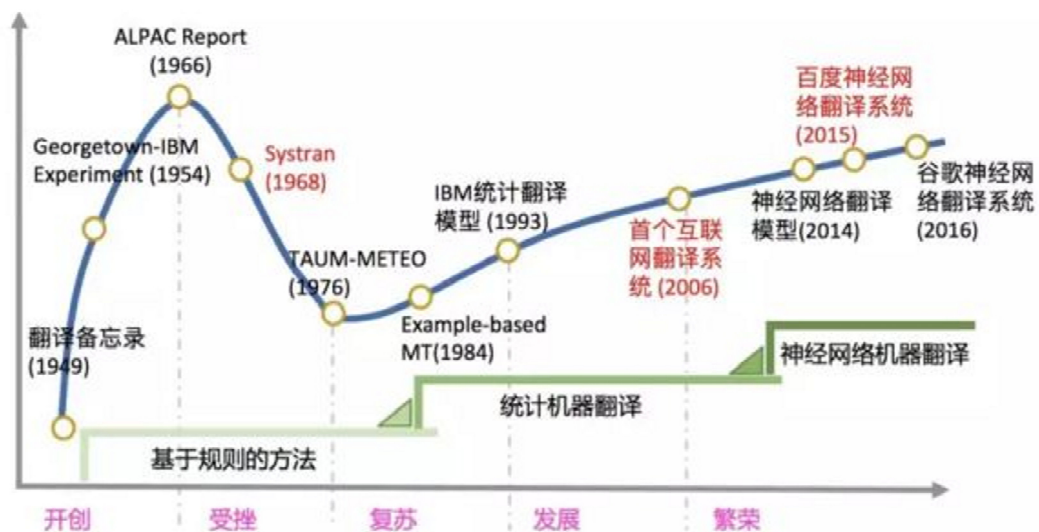
- 译文选择。在翻译一个句子时会面临很多选词的问题，因为语言中存在一词多义的现象比较普遍
- 词语顺序的调整。由于文化及语言发展上的差异，我们在表述的时候，有时候先说这样一个成份，后面说另外一个成份，但是，在另外一种语言中，这些语言成分的顺序可能是完全相反的。
- 数据稀疏。

现在的机器翻译技术大部分都是基于大数据的，只有在大量的数据上训练才能获得一个比较好的效果。而实际上，语言数量的分布非常不均匀的。右边的饼图显示了中文相关语言的一个分布情况，大家可以看到，百分之九十以上的都是中文和英文的双语句对，中文和其他语言的资源呢，是非常少的。在非常少的数据上，想训练一个好的系统是非常困难的。

## ##01\_相关工作

- 机器翻译的发展历程

# 机器翻译发展历程



## ➤ 基于规则

翻译知识来自人类专家。找人类语言学家来写规则，这一个词翻译成另外一个词。这个成分翻译成另外一个成分，在句子中的出现在什么位置，都用规则表示出来。这种方法的优点是直接用语言学专家知识，准确率非常高。缺点是什么呢？它的成本很高，比如说要开发中文和英文的翻译系统，需要找同时会中文和英文的语言学家。要开发另外一种语言的翻译系统，就要再找懂另外一种语言的语言学家。因此，基于规则的系统开发周期很长，成本很高。此外，还面临规则冲突的问题。随着规则数量的增多，规则之间互相制约和影响。有时为了解决一个问题而写的一个规则，可能会引起其他句子的翻译，带来一系列问题。而为了解决这一系列问题，不得不引入更多的规则，形成恶性循环。

## 基于规则的翻译

Source: 我 在 周 日 看 了 一 本 书

翻译知识：人工撰写规则	
专家知识 准确率高	成本高 周期长 规则冲突

Target: I read a book on Sunday

## ➤ 统计方法

大约到了上世纪九十年代出现了基于统计的方法，我们称之为统计机器翻译。统计机器翻译系统对机器翻译进行了一个数学建模。可以在大数据的基础上进行训练。

它的成本是非常低的，因为这个方法是语言无关的。一旦这个模型建立起来以后，

对所有的语言都可以适用。统计机器翻译是一种基于语料库的方法，所以如果是在数据量比较少的情况下，就会面临一个数据稀疏的问题。同时，也面临另外一个问题，其翻译知识来自大数据的自动训练，那么如何加入专家知识？这也是目前机器翻译方法所面临的一个比较大挑战。

翻译知识主要来自两类训练数据：平行语料，一句中文一句英文，并且这句中文和英文，是互为对应关系的，也叫双语语料；单语语料，比如说只有英文我们叫单语语料。

从平行语料中能学到什么呢？翻译模型能学到类似于词典这样的一个表，一般称为『短语表』。比如说『在周日』可以翻译成『on Sunday』。后面还有一个概率，衡量两个词或者短语对应的可能性。这样，『短语表』就建立起两种语言之间的一种桥梁关系。

那么我们能够用单语语料来做什么呢？我们用单语语料来训练语言模型。语言模型是做什么事情的呢？就是衡量一个句子在目标语言中是不是地道，是不是流利。比如这里说『read a book』，这个表述是没有问题的，『read a』后面跟一个『book』这个词的概率可能是0.5，那么如果说『read a TV』呢？可能性就很低。因为这不符合目标语言的语法。

所以，翻译模型建立起两种语言的桥梁，语言模型是衡量一个句子在目标语言中是不是流利和地道。这两种模型结合起来，加上其他的一些特征，就组成了一个统计机器翻译这样的公式。

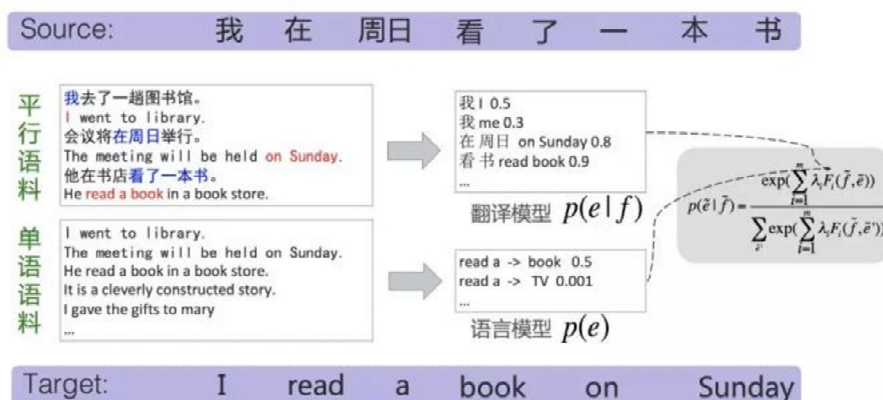
## 统计机器翻译

Source: 我 在 周 日 看 了 一 本 书



Target: I read a book on Sunday

## 统计机器翻译

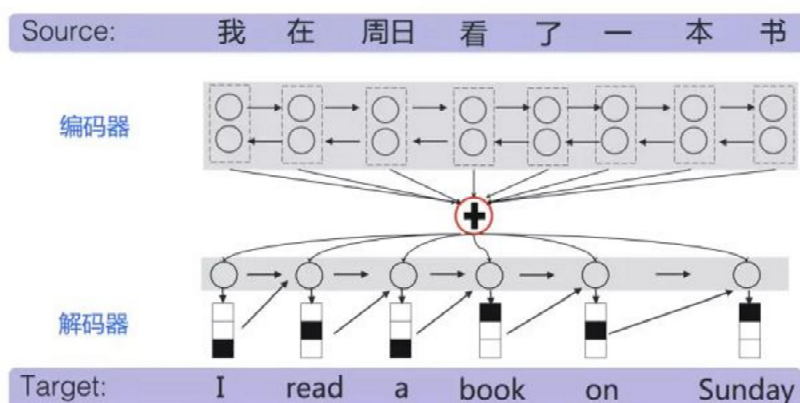


### ➤ 神经网络

神经网络翻译近年来迅速崛起。相比统计机器翻译而言，神经网络翻译从模型上来说相对简单，它主要包含两个部分，一个是编码器，一个是解码器。编码器是把源语言经过一系列的神经网络的变换之后，表示成一个高维的向量。解码器负责把这个高维向量再重新解码（翻译）成目标语言。

随着深度学习技术的发展，大约从2014年神经网络翻译方法开始兴起。2015年百度发布了全球首个互联网神经网络翻译系统。短短3、4年的时间，神经网络翻译系统在大部分的语言上已经超过了基于统计的方法。

## 神经网络机器翻译



- Linguistics
- Character-level model
- Subword-models
- Hybrid character and word level models
- Fasttext
- BLEU (bilingual evaluation understudy) 即双语互译质量评估辅助工具，用来评估机

器翻译质量的工具。

## 翻译质量评价 – 自动评价

基于n-gram，计算机器译文和人工译文（参考译文）的匹配程度

Source: 我在周日读了一本书。  
Reference: I read a book on Sunday.  
System1: On Sunday, I read book.  
System2: I read a book on Sunday.

Bilingual Evaluation Understudy

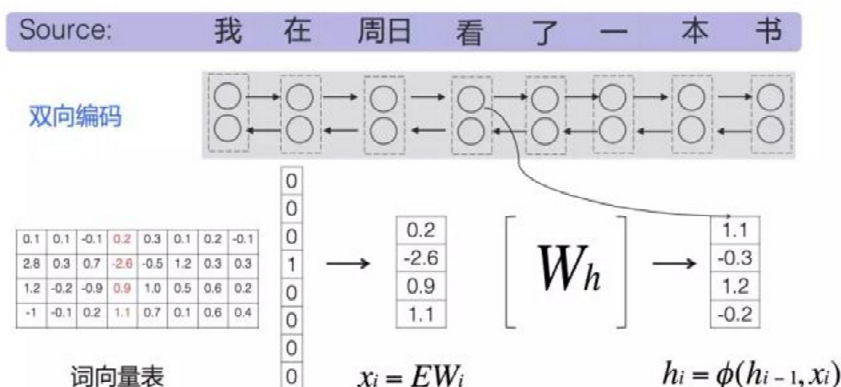
$$\text{BLEU} = \text{BP} \cdot \exp\left(\sum_{n=1}^N w_n \log p_n\right) \quad \text{BP} = \begin{cases} 1 & \text{if } c > r \\ e^{(1-r/c)} & \text{if } c \leq r \end{cases}$$

(Papineni et al., 2002)

- 神经网络翻译基本原理

解决了在统计机器翻译时非常难做的事情。比如：中文句子中的【尽快】，在英语中，【as soon as possible】换到后面了，进行了比较长距离的调序

## 神经网络机器翻译 – 基本原理



一个神经网络模型包含了编码器和解码器。先进行一个双向的编码，双向的编码干了什么事情？就是把词用词向量来表示，那是如何做到的呢？首先需要有个词向量表，通过神经网络训练出来。源语言句子中的词，可以用一个one hot的向量表示。所谓onehot就是，比如上例子中的中文句子有8个词，哪个词出现了，就把这个词标记为1，其他词标为0。比如第四个词“看”这个词1，那么其他的都是0，这两个矩阵乘起来，相当于查表操作。就把其中这个词向量表的一列取出来了，那么这一列的向量就代表了这个词。神经网络里所有的词都会用向量来表示。得到的词向量表示后，在经过一个循环神经网络的变换，得到另一个向量，称为隐状态（Hidden State）。

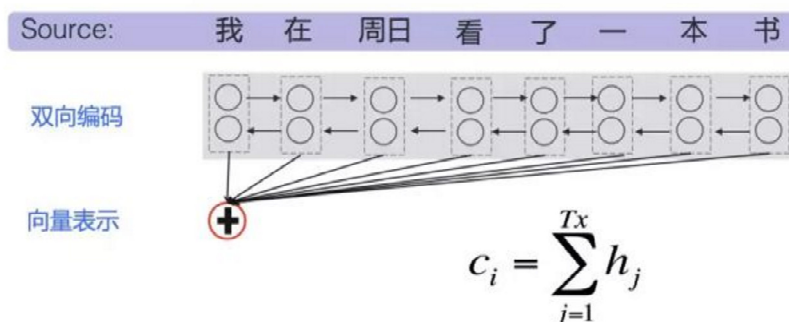
为什么做了一个双向的编码呢？为了充分利用上下文信息。比如：如果只是从左往右编码，“我再周日看”，看的是什么呢？“看”后面的你不知道，因为你只得到了“看”前面的信息。那么怎么知道后面的信息的呢？这时候我们想能不能从后面到前面进行一个编码，那就是“书本一了看”，从后面往前的编码，这时候“看”既有前面的信息，



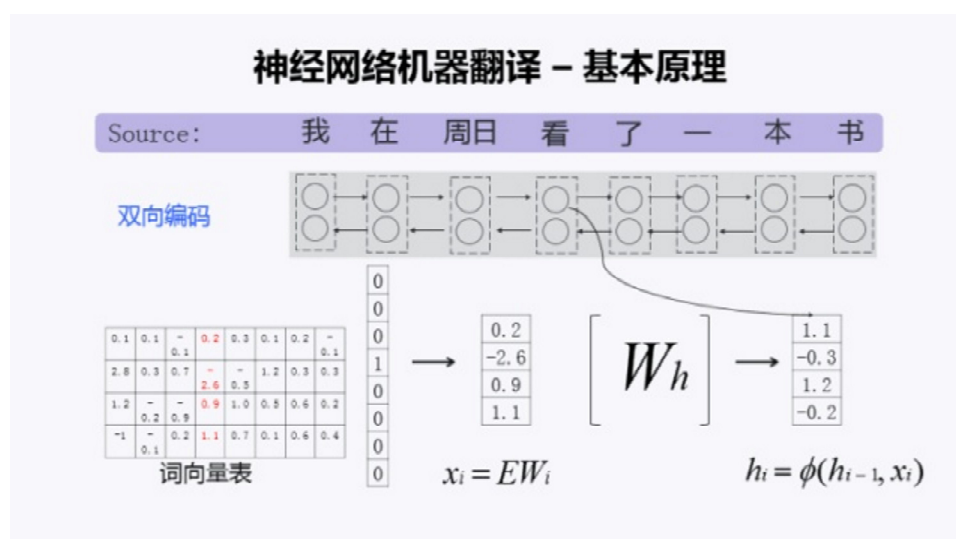
也有后面的信息，所以它有了一个上下文的信息，可以提高译文的质量。

每个词经过一系列变换，映射为一个向量表示，如果将双向编码的向量结合起来？现在一般采用一个非常简单的方法，将两个向量进行拼接。比如两个256维向量，拼接完后得到一个512维的向量，用来表示一个词。

## 神经网络机器翻译 – 基本原理

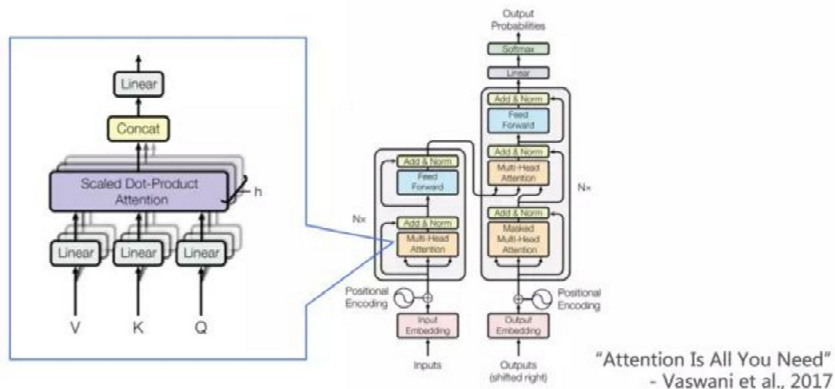


编码完成以后，需要把这个源语言的句子压缩到一个向量里去。这一步是怎么做的？一个最简单的方式是把这所有的向量加起来。但是后来大家发现这样其实不太合理。为什么不太合理，因为每一个词都是被作为相同的权重去对待的，那显然是不合理的，这时候就提出了一个**注意力机制，叫Attention**。这里用不同深度颜色的线去表示Attention的能量强弱，用以衡量产生目标词时，它所对应的源语言词的贡献大小。所以呢h前面又加一个 $\alpha$ ， $\alpha$ 就表示它的一个权重。



有了句子的向量表示后，就掌握了整个源语言句子的所有的信息。解码器就开始从左到右一个词一个词的产生目标句子。在产生某个词的时候，考虑了历史状态。第一个词产生以后，再产生第二个词，直到产生句子结束符EOS(End of Sentence)，这个句子就生成完毕了。

## 神经网络机器翻译 – SOTA

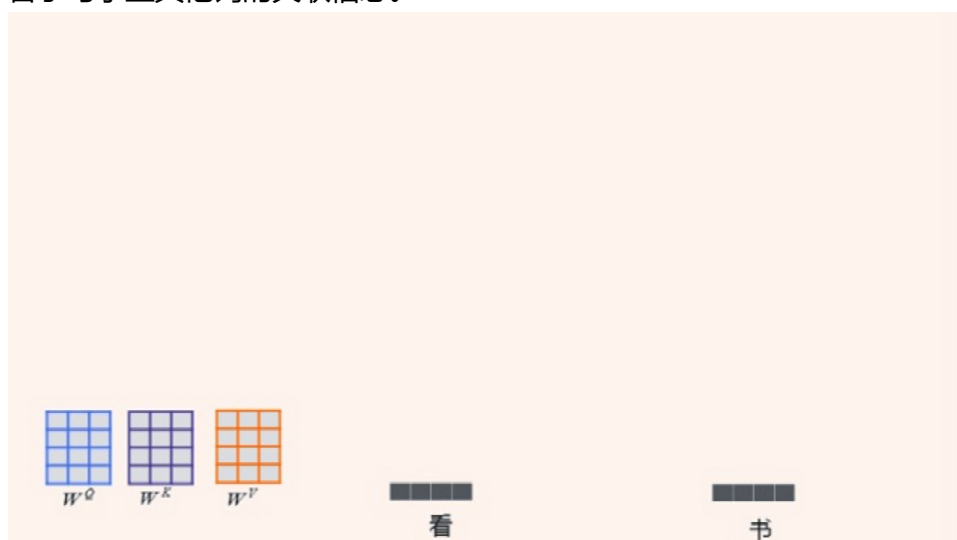


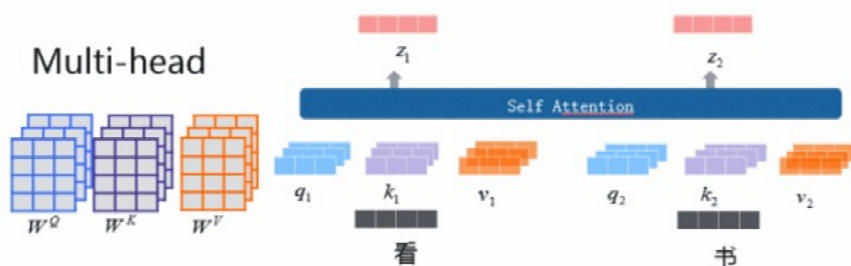
那么它是怎么做来的呢？它其实也有一个编码器和一个解码器，这个是架构是没有变的。其中编码器和解码器都有多层。下面我们通过一个具体例子，来简单解释一下其原理。

我们这个句子就包含两个词『看书』。

论文中，把每一个词都用三个向量表示，一个叫Query (Q)，一个叫Key (K)，另外一个Value (V)。那怎么得到一个词的Query、Key和Value呢？左边有三个矩阵， $W^Q$ 、 $W^K$ 和 $W^V$ ，只要跟每一词向量相乘，就能够把这个词转换成三个向量表示。那么目标是什么，我们想把『看』这样一个词，通过一系列的网路变换，抽象到高维的向量表示。

通过Q和K进行点积，并通过softmax得到每个词的一个attention权重，在句子内部做了一个attention，称作Self Attention。Self Attention可以刻画句子内部各成分之间的联系，比如说“看”跟“书”之间就建立了联系。这样，每个词的向量表示 (Z) 就包含了句子里其他词的关联信息。





作者认为只有这一个QKV不太够，需要从多个角度去刻画。如何做呢？提出了“Multi-head”。在里面论文里面定义了8组QKV的矩阵，当然也可以定义16个，这个数值可以自定义。在通过一系列变换，最终得到了每个词的向量表示。这只是encoder一层。那么这一层的输出做为下一层的输入，再来一轮这样的表示，就是Encoder-2，那么再来一轮就是第三层，如此一直到第N层。Decoder也是类似，不再解释。感兴趣的可以阅读原文。

- 神经机器翻译的挑战

- 漏译
- 数据稀疏
- 引入知识
- 可解释性
- 语篇翻译

## ##02\_模型

- 词级别下的模型

- character-level model

(1) 文本分类

(2) NMT

- Sub-word model

一些概念：

(1) Byte Pair Encoding

(2) word segmentation

- Wordpiece/sentencepiece model

- FastText







## ##0 \_实验

参考文献10

## ##04\_总结

### 参考文献

1	Luong M T , Manning C D . Achieving Open Vocabulary Neural Machine Translation with Hybrid Word-Character Models[J]. 2016.	 Achieving Open...	
2		 Revisiting Characte...	
3	BLEU: a Method for Automatic Evaluation of Machine Translation	 BLEU a method...	
4	<a href="https://www.zhihu.com/question/24588198">https://www.zhihu.com/question/24588198</a>		
5	<a href="https://blog.csdn.net/feilong_csdn/article/details/88655927">https://blog.csdn.net/feilong_csdn/article/details/88655927</a>	Fasttext使用	
6	Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2016). Enriching word vectors with subword information. arXiv preprint arXiv:1607.04606.	 Enriching word...	
7	<a href="https://zhuanlan.zhihu.com/p/32965521?from_voters_page=true">https://zhuanlan.zhihu.com/p/32965521?from_voters_page=true</a>		
8	<a href="https://github.com/facebookresearch/fastText">https://github.com/facebookresearch/fastText</a>		
9	<a href="http://zh.d2l.ai/chapter_natural-language-processing/fasttext.html">http://zh.d2l.ai/chapter_natural-language-processing/fasttext.html</a>		
10	<a href="https://daiwk.github.io/posts/nlp-fasttext.html">https://daiwk.github.io/posts/nlp-fasttext.html</a>		