Lecture08 Machine Translation, Sequence-to-sequence and Attention

---

MT任务：原句x翻译到目的句y

始于1950s，最早用于俄语译为英语

基于规则，使用双语词典来映射

1990s-2010s，统计机器学习SMT

从数据中学习概率模型
e.g. 法译英：给定法语句子x，寻找最佳英语句子y

$$\text{argmax}_y P(y|x)$$

使用贝叶斯分解：（为什么分解？单独的式子需要一次性理解如何翻译、如何译出好句子以及理解句子结构等）

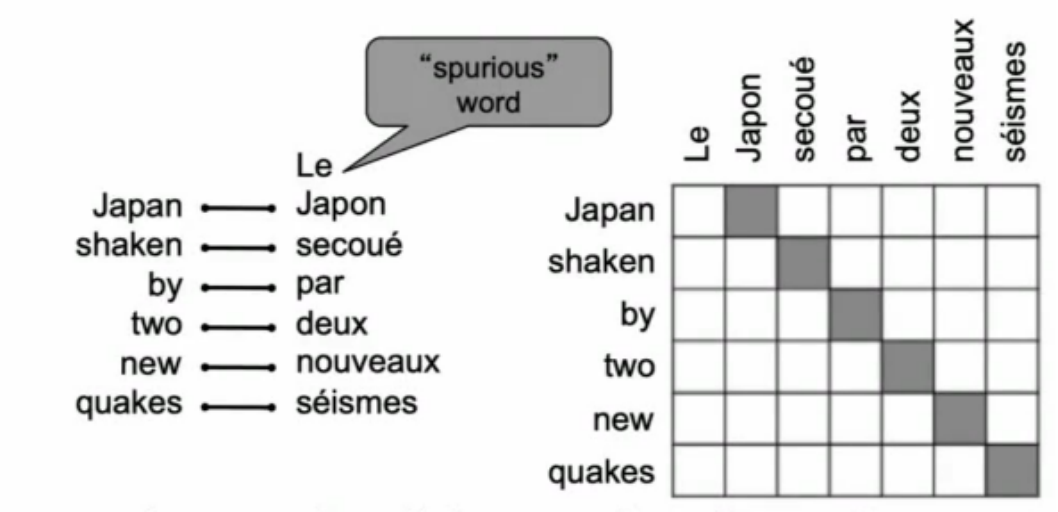
$$= \text{argmax}_y P(x|y)P(y)$$

其中：P(x|y)为翻译模型，从并行数据中学习，单词短语应如何被翻译（精确）
P(y)为语言模型，从单语数据中学习，如何写出好的英语句子（流畅）

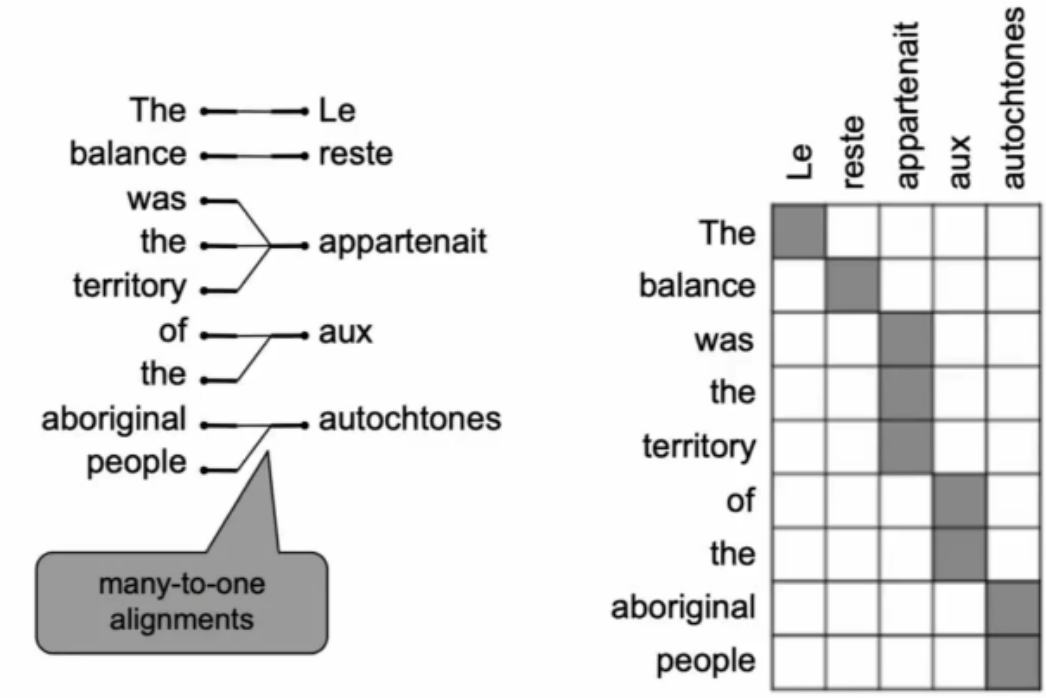
- 如何学习翻译模型P(x|y)？
  并行数据：法语英语句子对

  $$P(x, a|y)$$

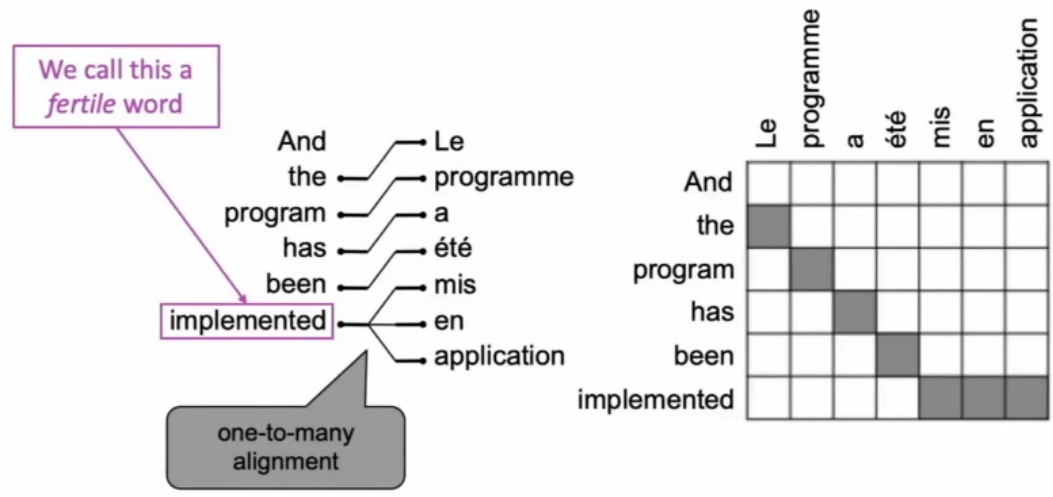
  其中a为对齐，即法语句子x与英语句子y之间特定词语之间的对应关系



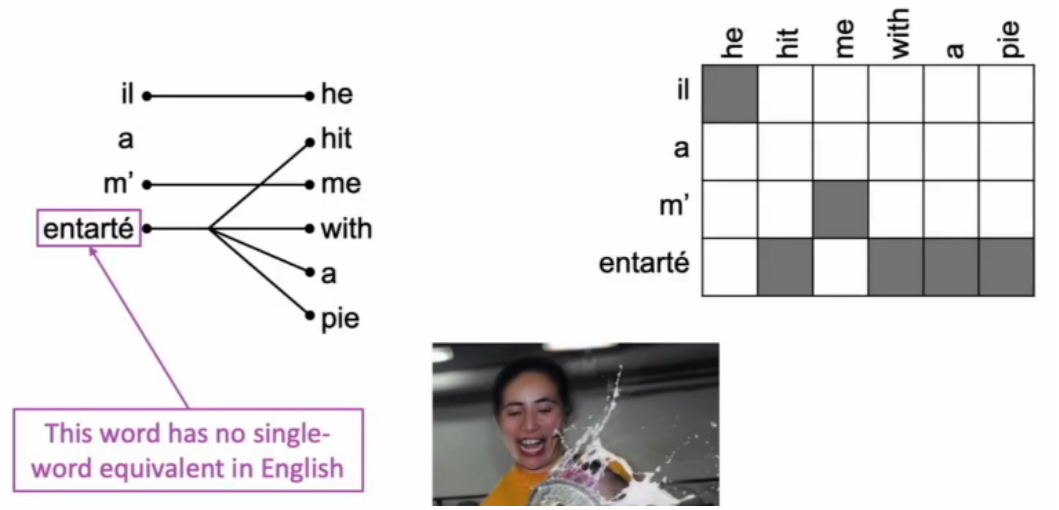
对齐也可以是多对一



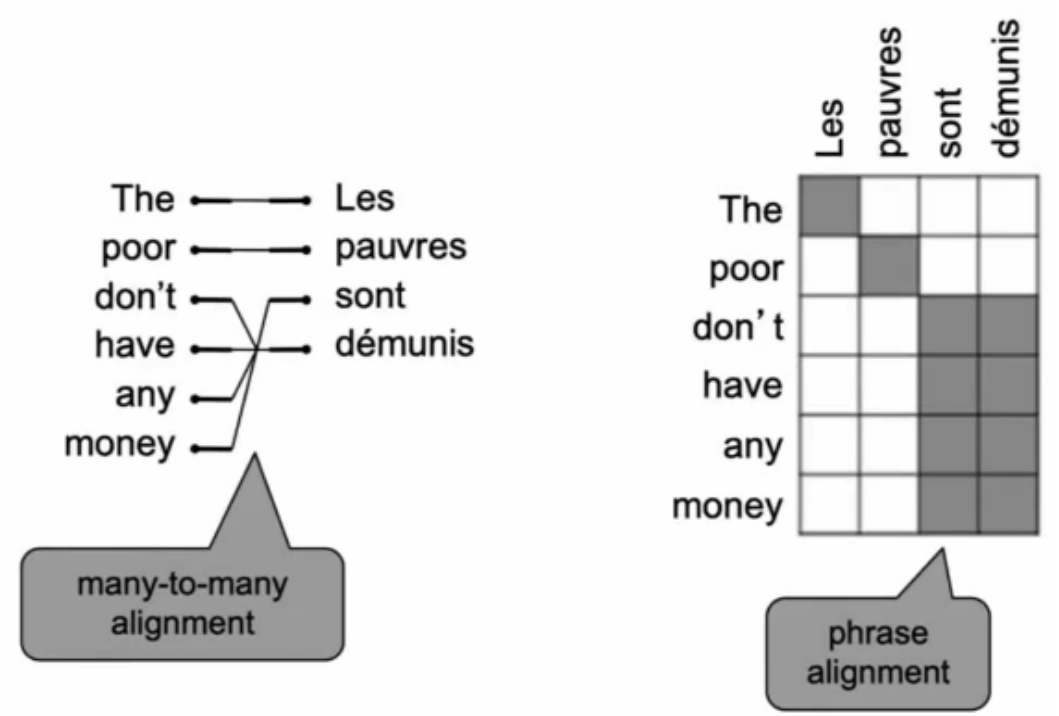
对齐也可以是一对多



有些词在英文中没有对应



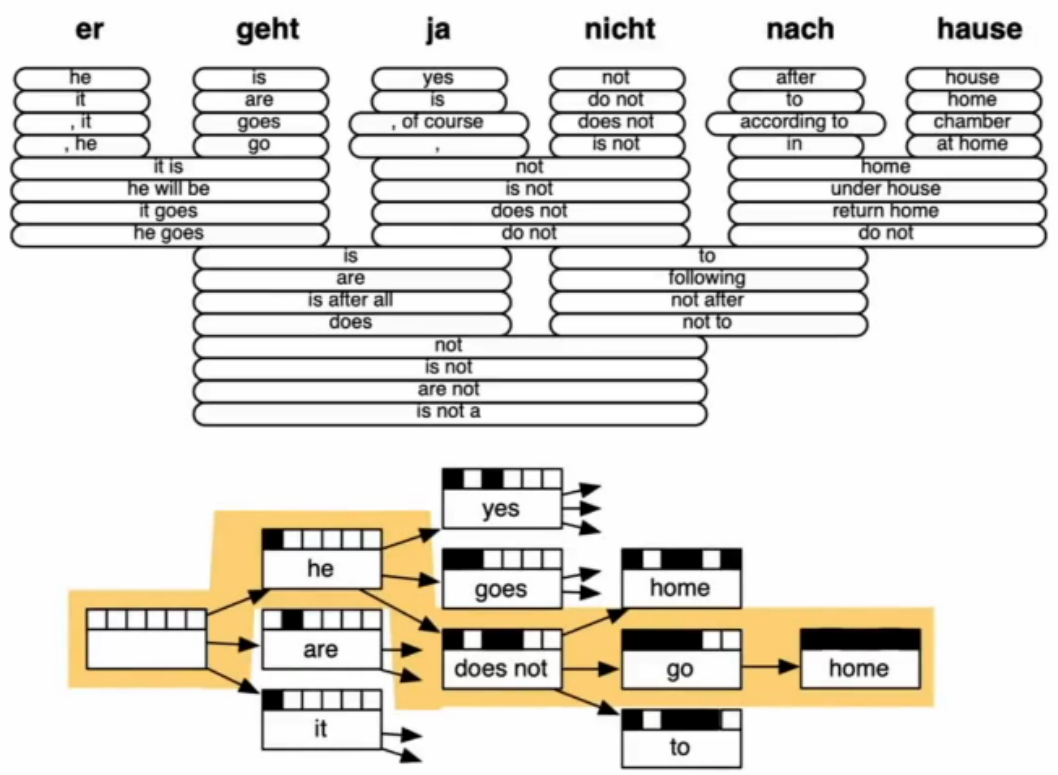
多对多对齐



P(x,a|y)与特定单词对齐的概率(也取决于发送中的位置)和特定词具有特定生育能力的概率(对应词的数量)等有关。

- 如何计算argmax？
  列举出每一个可能的y并计算出概率?>太贵了!
  答:使用启发式搜索算法来搜索最佳翻译，放弃概率太低的假设

————这个过程称为decoding



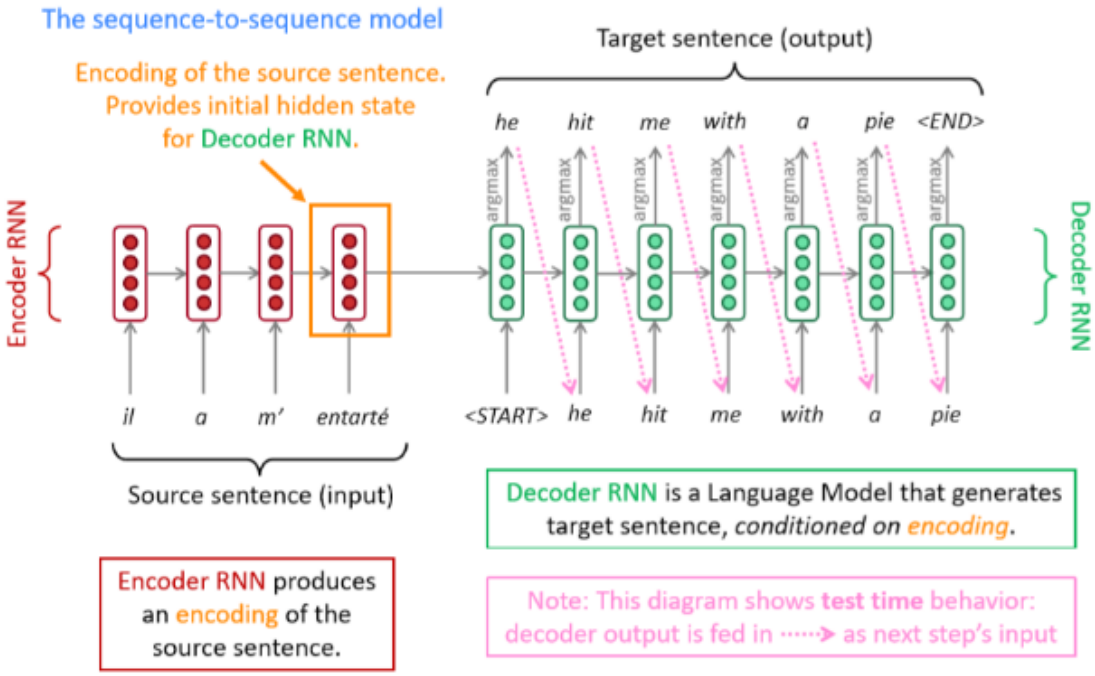
SMT总结

许多单独设计的子组件
大量的特征工程，需要设计特性来捕获特定的语言现象
需要编译和维护额外的资源

2014，Neural Machine Translation

利用单个神经网络（seq2seq）进行机器翻译

seq2seq



很多NLP任务都能使用：

- 摘要(长文本 → 短文本)
- 对话(前一句话 → 下一句话)
- 句法分析(输入文本 → 输出解析为序列)
- 代码生成(自然语言 → Python代码)

seq2seq模型是条件语言模型的一个例子。

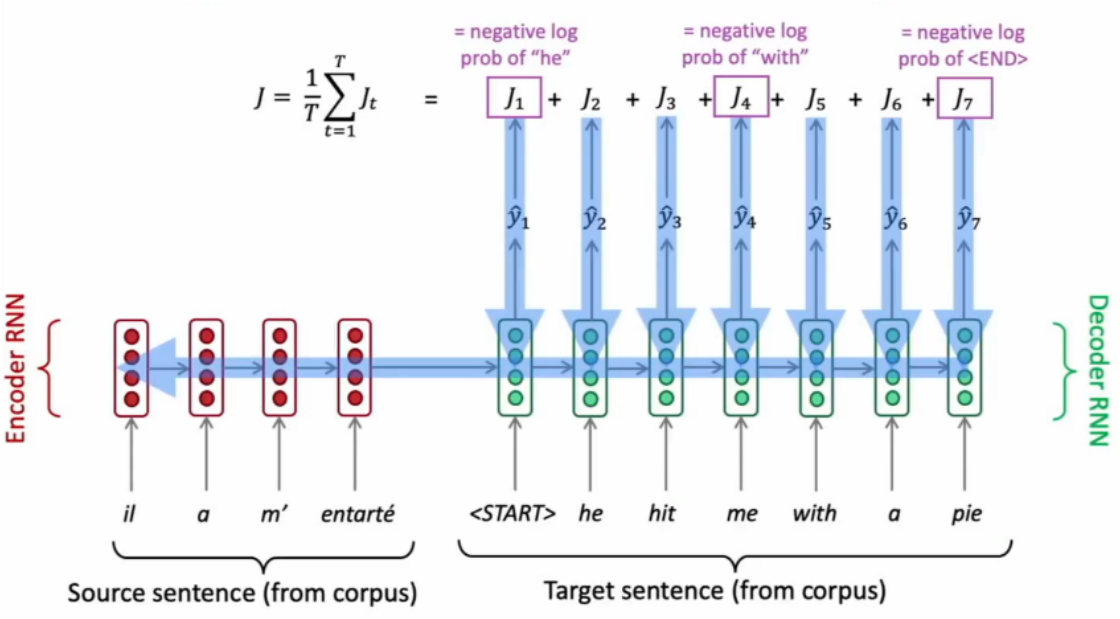
- 语言模型：因为解码器正在预测目标句子的下一个单词y
- 有条件的：因为它的预测也取决于源语句

NMT directly calculates $P(y|x)$:

$$P(y|x) = P(y_1|x)\,P(y_2|y_1,x)\,P(y_3|y_1,y_2,x)\ldots\underbrace{P(y_T|y_1,\ldots,y_{T-1},x)}$$

Probability of next target word, given target words so far and source sentence x

如何训练



Greedy decoding

每一步都取最可能的单词，但不能回退

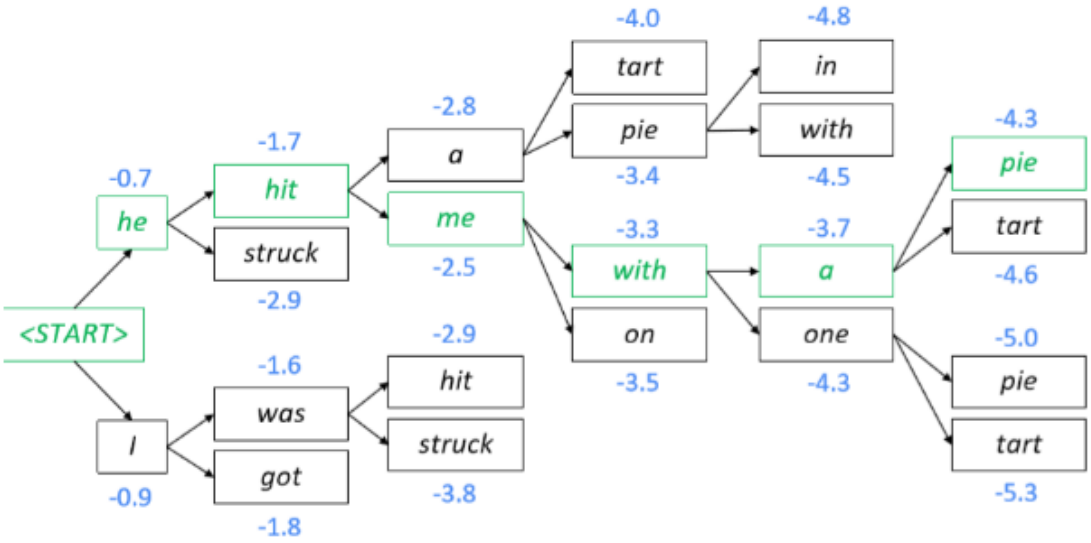
Exhaustive search decoding

理想情况下想要找到译句y使概率最大
需要计算所有可能y的概率，在大小为V的词表上每一时间步t，都要记录V^t次可能的部分翻译，花费大。

Beam search decoding

在解码器的每一步，跟踪 k 个最可能的部分翻译
k是Beam的大小(实际中大约是5到10)
不一定找到解，但是效率高

$$\text{score}(y_1, \ldots, y_t) = \sum_{i=1}^{t} \log P_{\text{LM}}(y_i | y_1, \ldots, y_{i-1}, x)$$



stop criterion：

- 在贪婪解码中，遇到<END>时结束解码

  $$\texttt{<START> he hit me with a pie <END>}$$

- beam search decoding中，不同的假设可能在不同的时间步长上产生 <END>
  - 当一个假设生成了 <END><END> 令牌，该假设完成把它放在一边，通过 Beam Search 继续探索其他假设。直到：
    - 到达时间步长 T (其中 T 是预定义截止点)
    - 至少有 n 个已完成的假设(其中 n 是预定义截止点)

    $$\text{score}(y_1, \ldots, y_t) = \sum_{i=1}^{t} \log P_{\text{LM}}\left(y_i | y_1, \ldots, y_{i-1}, x\right)$$

    越长的假设得分越低，除以句长来归一化，最后选择最高得分的假设。

优点：

- 性能好。流畅、更好地利用上下文、利用短语相似性
- 端到端。没有子组件需要单独优化
- 需要更少的人力工作

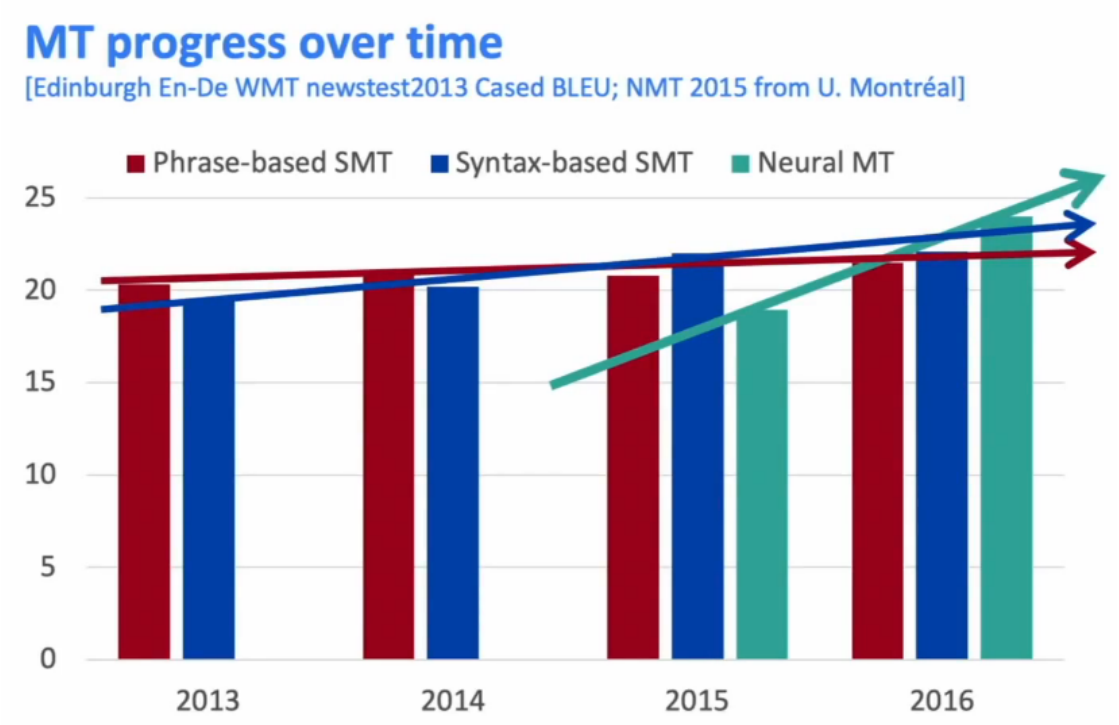
缺点：

- 可解释性差，难以调试
- 难以控制，不能轻易制定规则 & 安全问题

MT评价——BLEU(Bilingual Evaluation Understudy)

- BLEU将机器翻译与一个或多个人工翻译进行比较，并基于以下因素计算相似度评分:

    - n-gram精度(通常为1、2、3和4-gram)
    - 对于过短的翻译加上惩罚
- BLEU很有用,但不完美
    - 一个好的翻译可以得到一个糟糕的BLEU score，因为它与人工翻译的n-gram重叠较低



NMT: the biggest success story of NLP Deep Learning

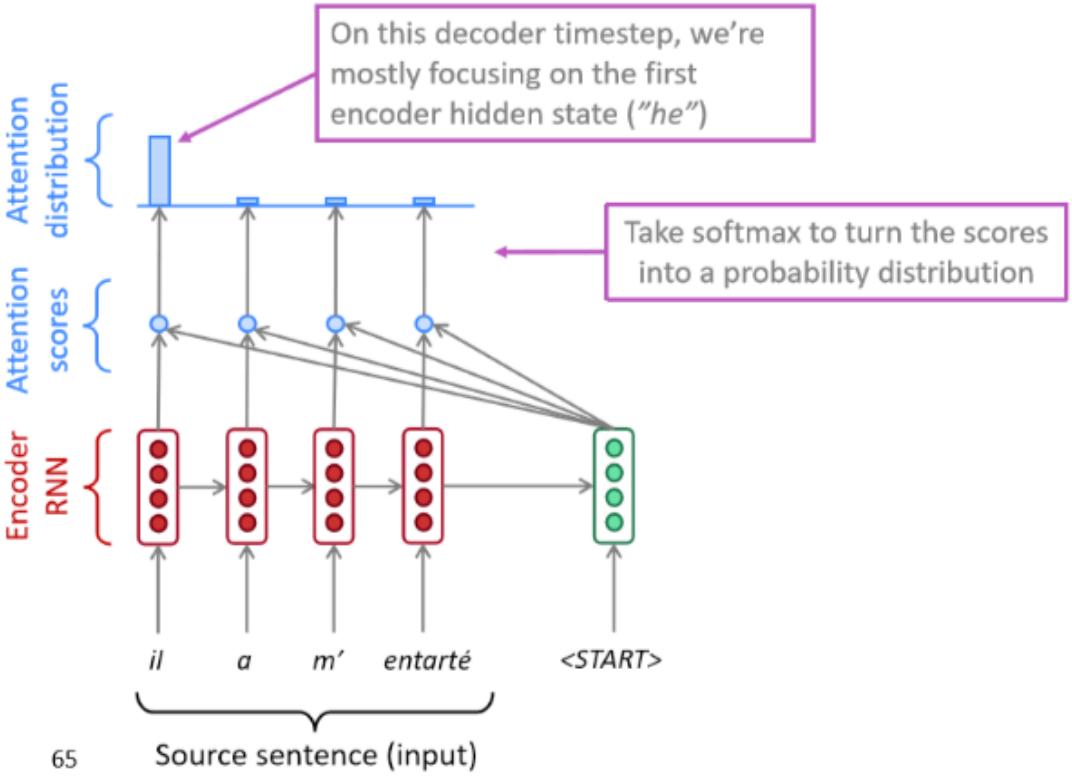
仍然存在的问题：

- 词表外的单词
- 处理训练和测试数据之间的领域不匹配
- 在较长文本上维护上下文
- 资源较低的语言对
- 在训练数据中发现偏见
- 翻译常识词性能不好
- weird things ：p

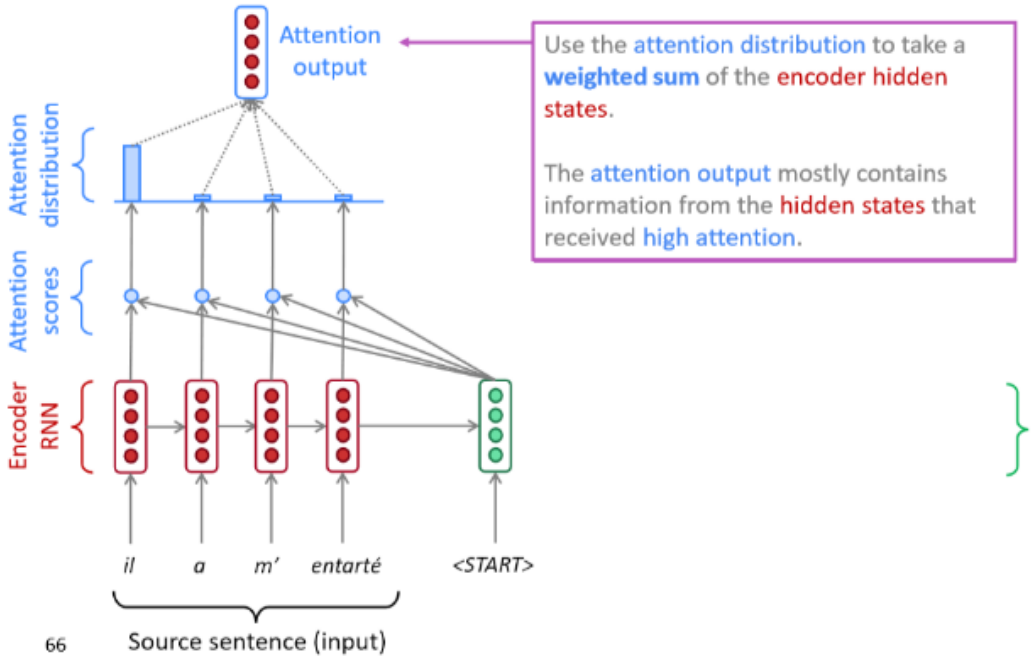
2019年：NMT研究将继续蓬勃发展。研究人员发现，对于我们今天介绍的普通seq2seq NMT系统，有很多、很多的改进。但有一个改进是如此不可或缺

Attention

Sequence-to-sequence的瓶颈问题：解码器输出到定长向量，可能会有信息丢失。
解决办法——**ATTENTION**

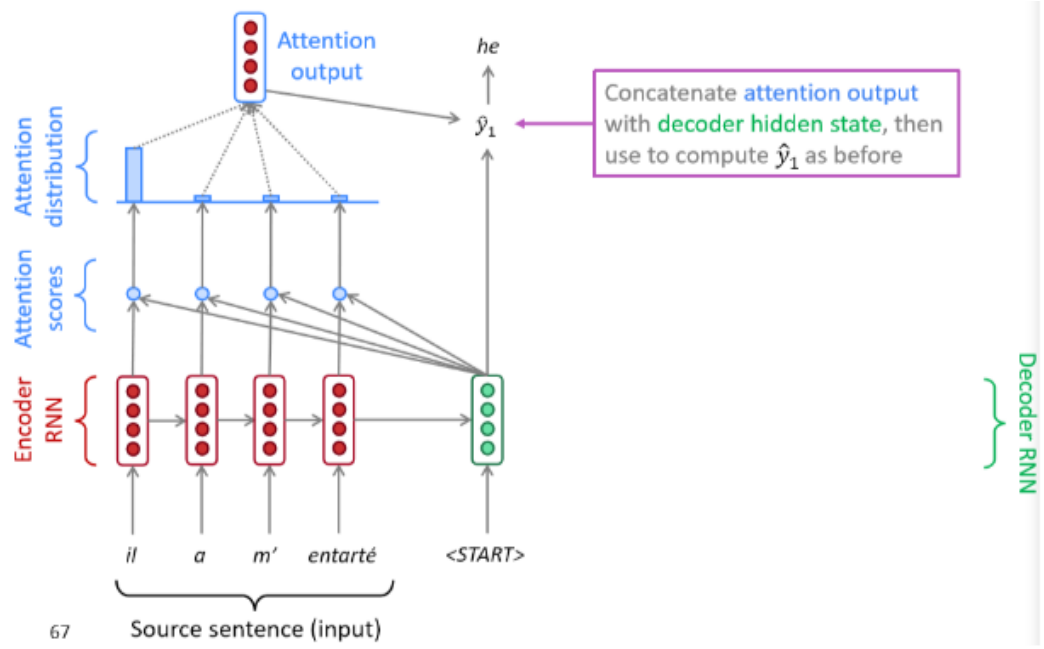


- 将解码器部分的第一个token <START>与源语句中的每一个时间步的隐藏状态进行Dot Product 得到每一时间步的分数
- 通过softmax将分数转化为概率分布

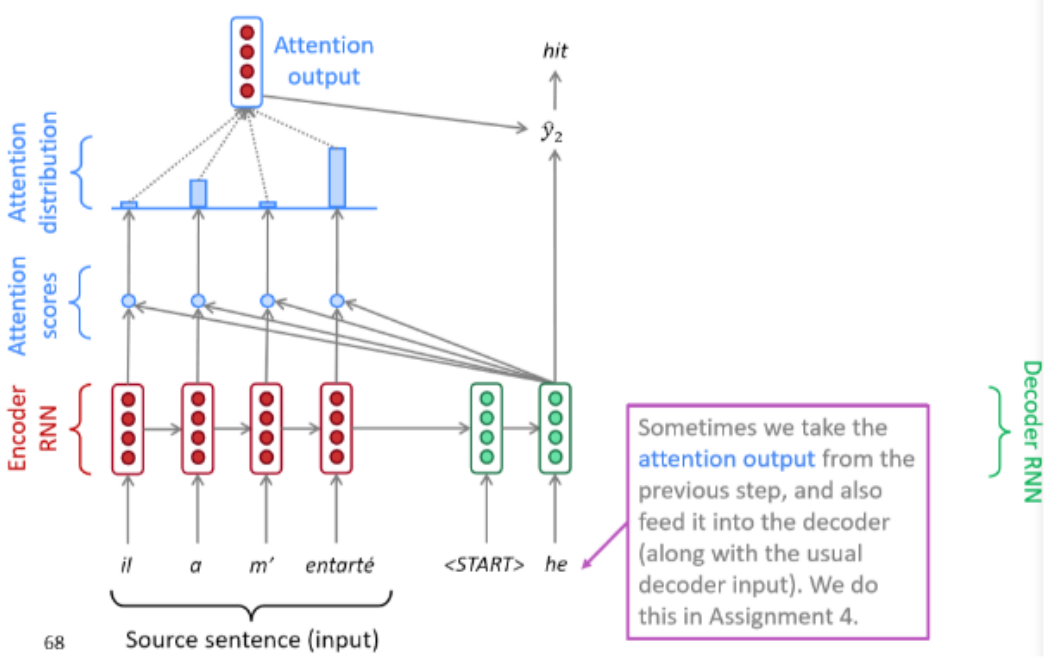


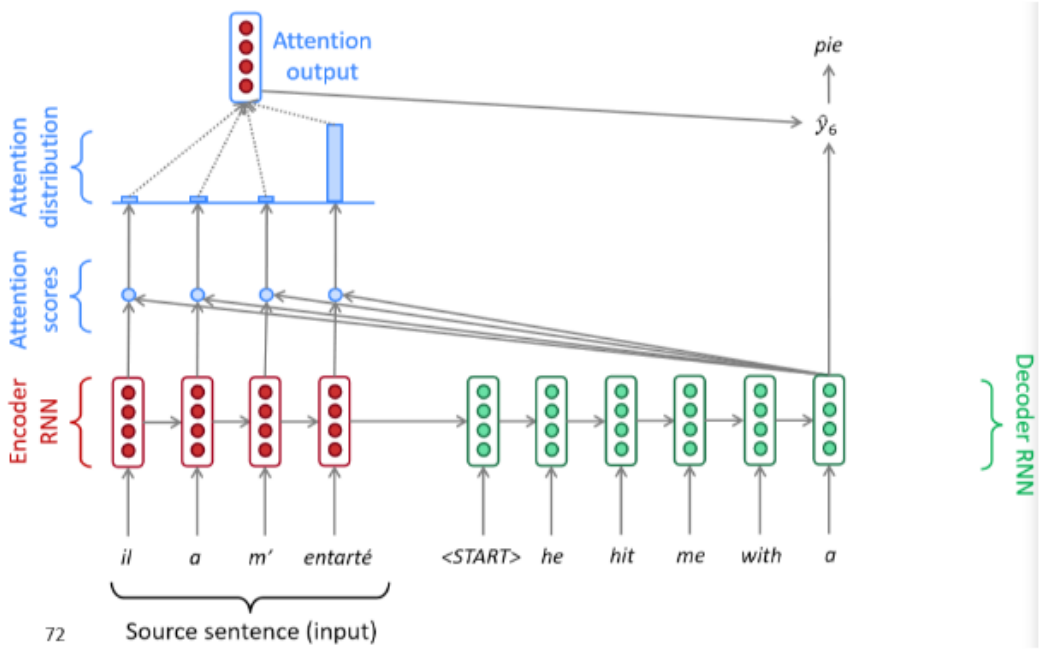
- 利用注意力分布对编码器的隐藏状态进行加权求和

- 注意力输出主要包含来自于受到高度关注的隐藏状态的信息



- 连接的 注意力输出 与 解码器隐藏状态 ，然后用来计算 y1





公式

- We have encoder hidden states $h_1, \ldots, h_N \in \mathbb{R}^h$
- On timestep *t*, we have decoder hidden state $s_t \in \mathbb{R}^h$
- We get the attention scores e^t for this step:

$$\boldsymbol{e}^t = [\boldsymbol{s}_t^T \boldsymbol{h}_1, \ldots, \boldsymbol{s}_t^T \boldsymbol{h}_N] \in \mathbb{R}^N$$

- We take softmax to get the attention distribution α^t for this step (this is a probability distribution and sums to 1)

$$\alpha^t = \text{softmax}(\boldsymbol{e}^t) \in \mathbb{R}^N$$

- We use α^t to take a weighted sum of the encoder hidden states to get the attention output $\boldsymbol{a}_t$

$$\boldsymbol{a}_t = \sum_{i=1}^{N} \alpha_i^t \boldsymbol{h}_i \in \mathbb{R}^h$$

- Finally we concatenate the attention output $\boldsymbol{a}_t$ with the decoder hidden state s_t and proceed as in the non-attention seq2seq model

$$[\boldsymbol{a}_t; \boldsymbol{s}_t] \in \mathbb{R}^{2h}$$

73

优点

- 显著提高NMT性能
- 解决了瓶颈问题
  - 注意力允许解码器直接查看源语句
- 帮助梯度消失问题
- 可解释性

- 通过检查注意力的分布，我们可以看到解码器在关注什么
- 得到(软)对齐

注意力的一般定义： 给定一组向量值和一个向量查询 ，注意力是一种根据查询，计算值的加权和的技术