小组组成：

PB21111712 王亭鉴

PB21111722 王艺洲

PB21111733 牛庆源

分工：

PB21111722 王艺洲完成了背景介绍，现有方法，以及实验结果分析部分；

PB21111733 牛庆源完成了论文方法部分以及对原先方法的改进；

PB21111712 王亭鉴完成了局限性和扩展方向部分，提出了两种扩展。

校对：

PB21111722 王艺洲，PB21111712王亭鉴完成了对报告错字的修改。

PB21111733 牛庆源完成了对报告格式方面的修改。

自然语言处理-文献T7.1阅读报告

背景介绍

神经机器翻译是最近提出的一种机器翻译方法。与传统的基于短语的翻译系统由许多单独调整的小子组件组成不同， 神经机器翻译尝试构建和训练单个大型神经网络来读取句子并输出正确的翻译。最近提出的神经机器翻译模型通常属于编码器-解码器架构，即将原句子编码为固定长度的向量，解码器从中生成翻译。

从概率的角度来看，翻译相当于找到一个目标句子 y ，在给定源句子 x 的情况下最大化 y 的条件概率，所以我们可以使用神经机器翻译中的并行训练语料库拟合参数化模型来最大化句子对的条件概率。目前，有许多论文提出使用神经网络来直接学习这种条件分布，这种神经网络翻译方法通常由两个组件组成，第一个组件对原句子 x 进行编码，第二个组件对目标句子进行解码。这种神经网络翻译方法得到的结果已经十分不错。

现有方法

在编码器-解码器架构中，编码器将输入句子$x = (x_1, x_2, \ldots, x_{T_x})$读取到向量$c$[2]中。这一过程最常见的方法为使用循环神经网络（ RNN ），有：

$$\begin{aligned} h_t &= f(x_t, h_{t-1}) \\ c &= q(h_1, h_2, \ldots, h_{T_x}) \end{aligned} \tag{1}$$

其中$h_t \in R_n$是隐藏层 t 时刻的状态， c 是从隐藏状态序列生成的向量， f 和 q 是一些非线性函数。

而解码器的作用为在给定上下文向量 c 和所有先前预测的单词$y_1, y_2, \ldots, y_{t-1}$的情况下预测下一个单词 y_t，即

$$p(y) = \prod_{t=1}^{T} p(y_t | y_1, y_2, \ldots, y_{t-1}, c) \tag{2}$$

其中$y = (y_1, y_2, \ldots, y_{T_y})$。

使用 RNN 方法，每个状态都可以写为

$$p(y_t | y_1, y_2, \ldots, y_{t-1}, c) = g(y_{t-1}, s_t, c) \tag{3}$$

其中 g 是一个非线性，可能是多层的函数，输出 y_t 的概率。而 s_t 为RNN的隐藏状态。

RNN 的详细工作流程如下：

1. 输入表示：首先，输入序列会被转化为向量表示。在自然语言处理任务中，输入通常是文本序列，每个单词会被映射到一个向量表示。这个向量表示可以是预训练的词向量，也可以是通过神经网络学习到的。
2. 时间步处理：在 RNN 中，序列中的每个时间步都会被依次输入到 RNN 网络中。每个时间步的输入包括当前时间步的输入向量和上一个时间步的隐藏状态。 RNN 的核心思想是通过不断迭代更新隐藏状态，捕捉序列中的长期依赖关系。
3. 隐藏状态更新：在每个时间步， RNN 都会根据当前输入和上一个时间步的隐藏状态计算出一个新的隐藏状态。这个隐藏状态会蕴含序列中之前时间步的信息，并且会传递给下一个时间步。
4. 输出计算：在 RNN 中，每个时间步都会有一个输出。这个输出可以用于进一步的处理或者预测。在文本生成任务中， RNN 的输出可以被用来生成下一个单词；在时间序列预测任务中， RNN 的输出可以被用来预测下一个时间步的数值。
5. 反向传播：在训练阶段， RNN 会通过反向传播算法来更新网络中的参数，使得模型的预测结果更加准确。反向传播的过程中，会根据损失函数的值来计算梯度，并且根据梯度来更新网络中的权重和偏置。

总的来说， RNN 通过不断迭代的方式来处理序列数据，通过更新隐藏状态来捕捉序列中的长期依赖关系。

现有方法局限性

现有方法中，一个潜在问题为神经网络需要将原句子的所有必要信息压缩成固定长度的向量，这可能会使神经网络难以处理长句子，尤其是比语料库中的句子更长的句子。研究表明，随着输入句子长度的增加，基本编码器-解码器的性能确实会迅速恶化。

论文方法

新的模型架构

定义公式(2)中的条件概率为

$$p(y_i|y_1,\ldots,y_{i-1},x) = g(y_{i-1},s_i,c_i) \tag{4}$$

其中 s_i 是时间 i 的 RNN 隐藏状态，公式为：

$$s_i = f(s_{i-1},y_{i-1},c_i)$$

注意这里的概率以每个目标词 y_i 的不同上下文向量 c_i 为条件，与之前给出的 RNN 方法不同。

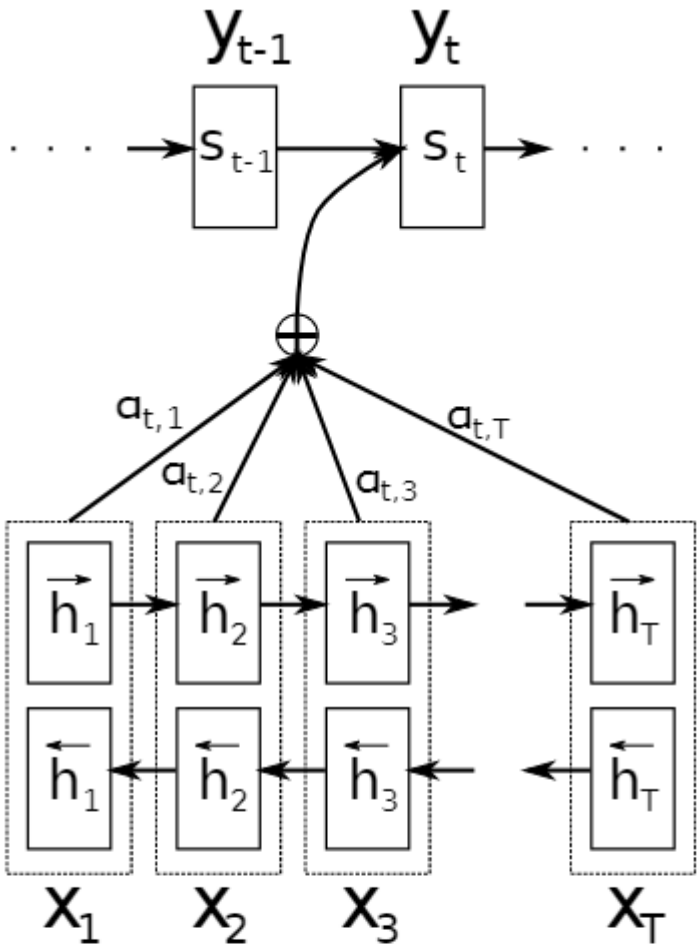


Figure 1: The graphical illustration of the proposed model trying to generate the t-th target word y_t given a source sentence $(x_1, x_2, \ldots, x_T)$.

上下文向量 c_i 取决于编码器将输入句子映射到的注释序列($h_1, \ldots, h_{T_x}$)。每个注释 h_i 包含有关整个输入序列的信息，重点关注输入序列第 i 个单词周围的部分。上下文向量 c_i 为这些注释 h_i 的加权和：

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j \tag{5}$$

(5)中每个注释 h_j 的权重 α_{ij} 计算为：

$$\alpha_{ij} = \frac{exp(e_{ij})}{\sum_{j=1}^{T_x} exp(e_{ik})} \tag{6}$$

其中

$$e_{ij} = a(s_{i-1}, h_j)$$

是一个对齐模型，用于对位置 j 周围的输入与位置 i 的输出匹配程度进行评分。分数基于 RNN 隐藏状态 s_{i-1} （公式(4)得到的 y_i 出现之前）和输入句子的第 j 个注释 h_j 。

将对齐模型 a 参数化为前馈神经网络，该网络与所提出系统的所有其他组件联合训练。与传统的机器翻译不同，对齐不被视为潜在变量。相反，对齐模型直接计算软对齐，这允许反向传播成本函数的梯度。该梯度可用于联合训练对齐模型以及整个翻译模型。

我们可以将所有注释的加权和的方法理解为计算预期注释，其中期望超过可能得对齐。令 α_{ij} 为目标词 y_i 与源词 x_j 对齐或翻译自源词 x_j 的概率。然后第 i 个上下文向量 c_i 是所有概率为 α_{ij} 的注释的预期注释。

概率 αij 或其相关能量 e_{ij} 反映了注释 h_j 相对于先前隐藏状态 s_{i-1} 在决定下一个状态 s_i 和生成 y_i 时的重要性。

模型需要使用的双向RNN

下面将构造用于注释序列的双向 RNN 。

通常的 RNN 如公式(1)，按照从第一个符号 x_1 到最后一个符号 x_{T_x} 的顺序读入输入序列 x 。然而，在刚刚提出的方案中，需要每个单词的注释不仅可以总结前面的单词，还可以总结后面的单词。因此需要使用双向 RNN （ $BiRNN$ ）。

$BiRNN$ 由前向和后向 RNN 组成。前向 RNN $\overrightarrow{f}$ 按顺序读取输入序列（从 x_1 到 x_{T_x} ）并计算前向隐藏状态序列 $(\overrightarrow{h_1}, \ldots, \overrightarrow{h_{T_x}})$ 。后向 RNN $\overleftarrow{f}$ 以相反的顺序读取序列（从 x_{T_x} 到 x_1 ）从而产生后向隐藏状态序列 $(\overleftarrow{h_1}, \ldots, \overleftarrow{h_{T_x}})$ 。

通过连接前向隐藏状态和后向隐藏状态 $h_j = [\overrightarrow{h_j^T}; \overleftarrow{h_j^T}]^T$ 为每一个词 x_j 获得一个注释。通过这样的处理方式，注释 h_j 就包含了前面单词和后面单词的摘要。由于 RNN 倾向于更好地表示最近的输入，因此注释 h_j 将集中在 x_j 周围的单词上。解码器和对齐模型随后使用该注释序列来计算上下文向量（公式(5)-(6)）。

$Fig.1$ 为模型示意图。

相比于原先方法的改进：

1. 不同于RNN，这里的 c_i 包含了 $BiRNN$ 生成的对于词 x_i 的注释向量，由前向和后向隐藏状态组成，比原来的方法只使用单一方向的 RNN 要更加全面，能更加精确地表示单词 x_i 的注释。
2. 除此之外，对每一个复合隐藏状态表示的注释 h_i ，使用基于上一个单词的注释和隐藏状态生成的对齐模型为其分配权重，这样处理可以使当前单词的翻译更加依赖于周围单词的权重，而周围单词也同样地收到他们周围的印象，形成一个影响链，实现了对每个位置单词翻译准确度的较大提升。
3. 简单来说，这在解码器中实现了注意力机制。解码器决定源语句中要注意的部分。通过让解码器具有注意力机制，减轻了编码器将源句子中的所有信息编码为固定长度向量的负担。通过这种新方法，信息可以分布在整个注释序列中，解码器可以相应地选择性地检索这些注释。

论文方法的优越性：

论文方法中对编码器-解码器模型进行了扩展，通过学习联合对齐和翻译，在翻译生成每一个单词时，模型都会（软）搜索原句子中最相关信息中的一组位置，然后根据这些与原位置相关联的上下文向量以及所有先前生成的目标单词来预测当前目标单词。

这种方法不会尝试将整个输入句子编码为单个固定长度的向量，而是将输入句子编码为向量序列，并在解码翻译时自适应地选择这些向量的子集。这使得神经翻译模型不必将源句子的所有信息压缩为固定长度的向量，所以能够更好地处理长句子。经定性分析，这种方法可行并且有效。下面这一部分为分析过程与结果：

实验结果与分析（数据集、对比模型、数据分析等）

数据集

这篇论文使用`news-test-2012`和`news-test-2012`来制作训练集合，之后使用缩小后的`WMT'14`作为测试集来评估模型。所选择的`WMT'14`为包含`Europarl（61M）`，`新闻评论（5.5M）`，`UN(421M)`以及两个语料库中的850M单词，之后使用Axelrod等人的数据选择方法将组合语料库的大小减少到348M个单词，并且不使用任何单语数据来预训练编码器。这样一来，该测试集中包含训练模型中不存在的3003个句子。

对比模型

论文中训练了两种模型，一个是已有的RNN编码器-解码器模型，称为 $RNNencdec$ 模型，另一个是论文中所提出的模型，称为 $RNNsearch$ 模型。之后对每个模型进行两次训练，第一次使用长度最多为30个单词的句子训练，称为 $RNNencdec-30$ 和 $RNNsearch-30$；第二次使用长度最多为50个单词的句子训练，称为 $RNNencdec-50$ 和 $RNNsearch-50$，共计4个模型。

其中，$RNNencdec$ 的编码器和解码器各有 1000 个隐藏单元。$RNNsearch$ 的编码器由前向和后向循环神经网络组成，每个网络都有 1000 个隐藏单元共计2000个隐藏单元。而它的解码器和 $RNNencdec$ 一样有 1000 个隐藏单元。在这两种情况下，实验都使用具有单个 $maxout$ 隐藏层的多层网络来计算每个目标单词的条件概率。

关于模型训练，实验使用小批量随机梯度下降 (SGD) 算法和 $Adadelta$ 来训练每个模型。每个 SGD 更新方向都是使用 80 个句子的小批量计算的。每个模型都进行了大约 5 天的训练。模型训练完成后，实验使用集束搜索来找到近似最大化条件概率的翻译，从神经网络翻译机器翻译模型生成翻译。

数据分析

定量分析

表一中列出了以BLUE分数（取值范围为 0～1，分数越接近1，说明翻译的质量越高）衡量的翻译性能。从表中可知，在所有情况下论文提出的 $RNNsearch$ 方法都优于传统的 $RNNencdec$ 方法。进一步来说，当只考虑由已知单词组成的句子时，$RNNsearch$ 搜索的性能与传统的基于短语的翻译系统Moses一样高（Moses使用的是单独语料库，更加说明 $RNNsearch$ 搜索性能之高）。

图二中显示了测试集上生成的翻译相对与句子长度的BLUE分数。从图中可以看出，随着句子长度的增加，$RNNencdec$ 的性能急剧下降。而对于 $RNNsearch$ 来说，当训练句子长度为30时，在面对长句子翻译时性能也会有所下降，但当训练句子长度为50时，即使翻译句子长度为50或者更长，性能也不会下降。并且 $RNNsearch-30$ 甚至优于 $RNNencdec-50$,这一事实进一步证实了所提出的模型相对于基本编码器-解码器的优越性。

Model	All	No UNK°
RNNencdec-30	13.93	24.19
RNNsearch-30	21.50	31.44
RNNencdec-50	17.82	26.71
RNNsearch-50	26.75	34.16
RNNsearch-50⋆	28.45	36.15
Moses	33.30	35.63

Table 1: BLEU scores of the trained models computed on the test set. The second and third columns show respectively the scores on all the sentences and, on the sentences without any unknown word in themselves and in the reference translations. Note that RNNsearch-50⋆ was trained much longer until the performance on the development set stopped improving. (○) We disallowed the models to generate [UNK] tokens when only the sentences having no unknown words were evaluated (last column).

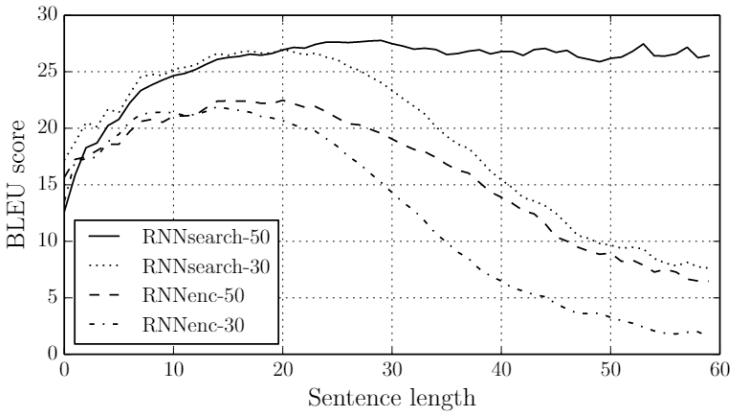


Figure 2: The BLEU scores of the generated translations on the test set with respect to the lengths of the sentences. The results are on the full test set which includes sentences having unknown words to the models.

定性分析

对齐

实验通过可视化等式中的注释权重 α_{ij} 来直观地检查生成地翻译中的单词与原句子之间的（软）对齐。如图三所示，矩阵中的每一行表示与注释相关的权重，从中可以直观地看出在生成目标单词时原句子中的哪些位置更加重要。

从图中可以看出英语和法语之间的单词对齐基本上是单调的，每个对角线都有很强的权重。然而也存在很多重要的非单调的排列。法语和英语中形容词和名词的顺序通常不同，图3(a)中我们可以看到模型正确地将短语 `[European Economic Area]` 翻译为 `[zoneéconomique européen]`。本次实验提出的 $RNNsearch$ 能够正确地将 `[zone]` 与 `[Area]` 对齐，跳过这两个单词 （`[European]` 和 `[Economic]`）之后回看一个单词以完成整个短语 `[zone économique européenne]`。

软对齐有两个好处。其一为翻译强度相比与硬对齐更高。如图三中3(d)中的短语 `[the man]`，它被翻译成 `[l' homme]`。任何硬对齐都会将 `[the]` 映射到 `[l']`， 将 `[man]` 映射到 `[homme]`。这对翻译没有帮助，因为必须考虑 `[the]` 后面的单词来确定是否应该翻译为 `[le]` 、`[la]` 、`[les]` 或 `[l']` 。我们的软对齐通过让模型同时查看 `[the]` 和 `[man]` 自然地解决了这个问题，在这个例子中，我们看到模型能够正确地将 `[the]` 翻译为 `[l']`。对于图3中的其他矩阵，我们同样可以发现这一点。

长句子

从图2中可以清楚地看出，所提出的模型（ $RNNsearch$ ）在翻译长句子方面比传统模型（ $RNNencdec$ ）要好得多。这可能是因为 RNN 搜索不需要将长句子完美地编码为固定长度的向量，而只需要准确地编码输入句子中围绕特定单词的部分。考虑测试集中的一个句子：

```
This kind of experience is part of Disney's efforts to "extend the lifetime of its series and build new relationships with audiences via digital platforms that are becoming ever more important," he added.
```

$RNNencdec$ 在生成大约30个单词后开始偏离源句子的实际含义，之后翻译质量下降并出现基本错误。但是 $RNNsearch$ 可以正确地翻译该句子。结合定量分析，定性分析证实了 $RNNsearch$ 能够比 $RNNencdec$ 更可靠地翻译长句子。

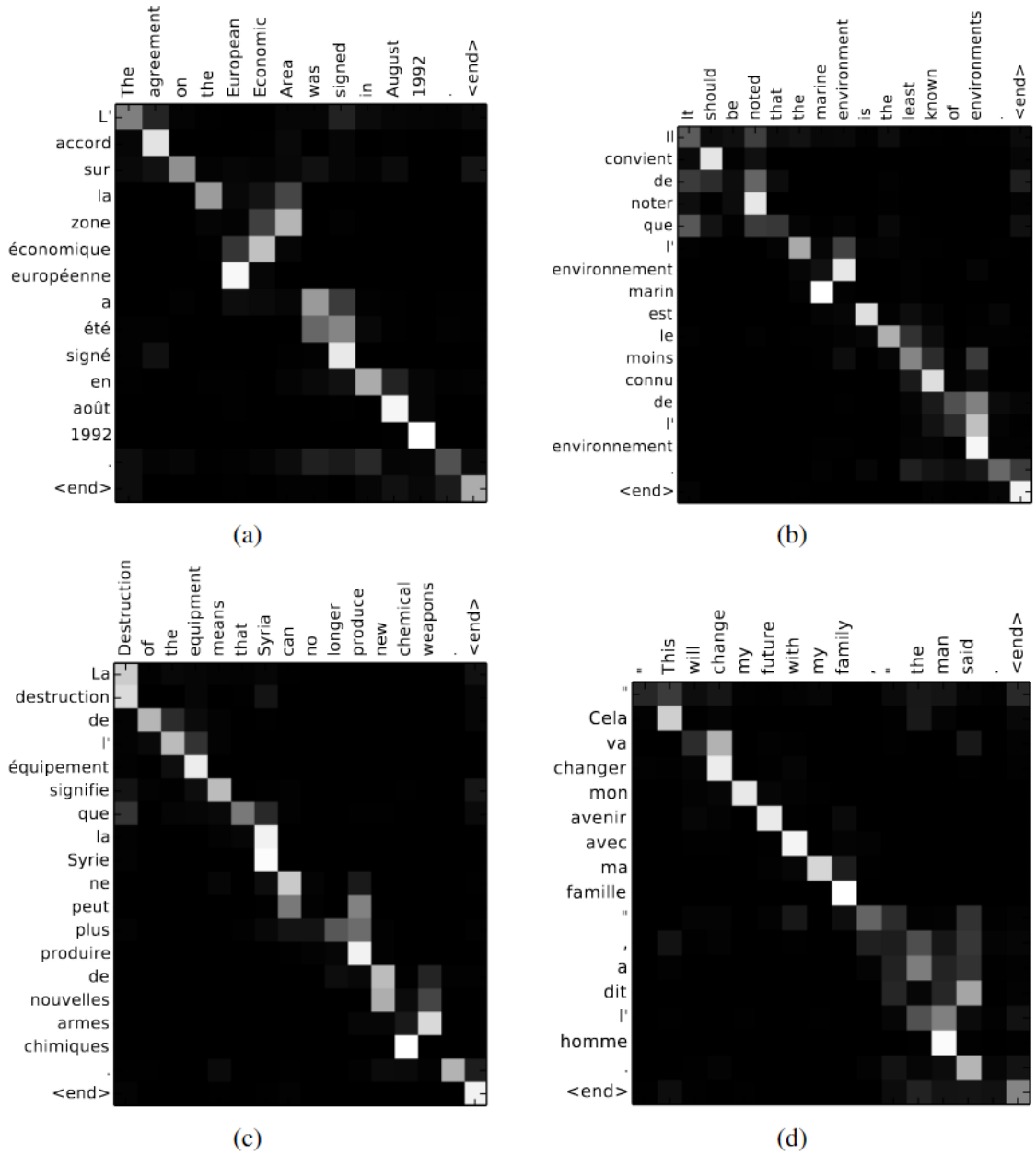


Figure 3: Four sample alignments found by RNNsearch-50. The x-axis and y-axis of each plot correspond to the words in the source sentence (English) and the generated translation (French), respectively. Each pixel shows the weight α_{ij} of the annotation of the j-th source word for the i-th target word (see Eq. (6)), in grayscale (0: black, 1: white). (a) an arbitrary sentence. (b–d) three randomly selected samples among the sentences without any unknown words and of length between 10 and 20 words from the test set.

局限性与可能扩展方向

问题

$BiRNN$ 在处理长序列时未能摆脱 RNN 的局限性，难以捕捉到长期依赖关系，只能有效利用较短的上下文信息。

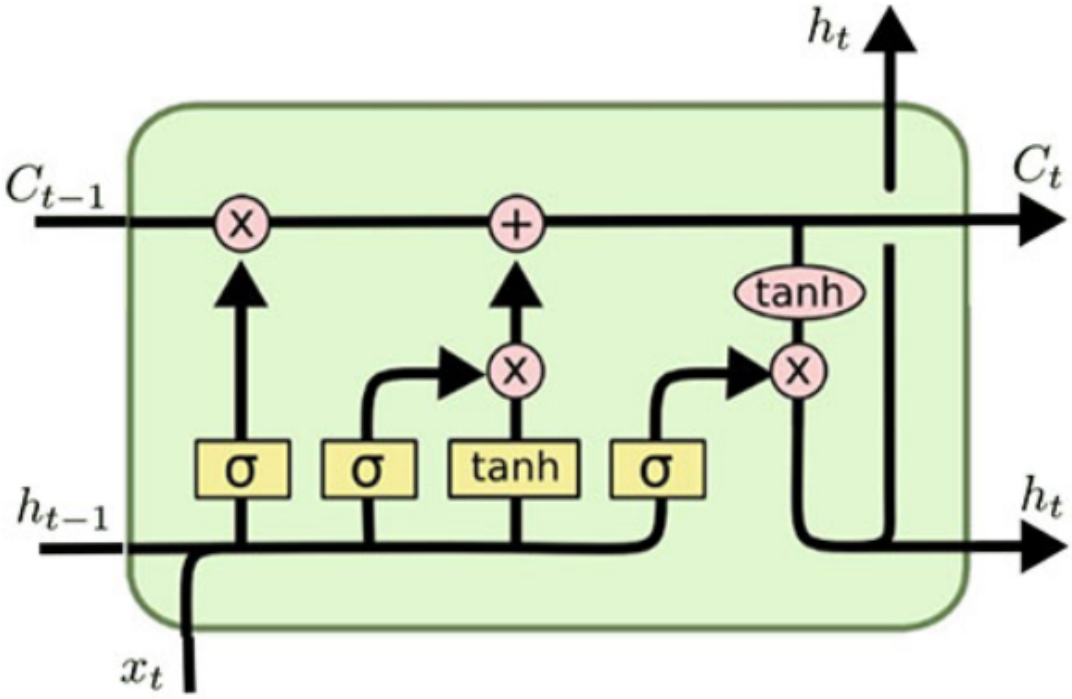
该方法需要计算翻译中每个单词的源句子中每个单词的注释权重，单词较多、翻译任务大时，效率低。

需要一个较大的文本语料库。

扩展方向

双向LSTM（Long Short-Term Memory）

$LSTM$ 设计了一种能够灵活控制信息流的细胞状态。该细胞状态贯穿整个序列，允许信息的长期保存或遗忘。 $LSTM$ 由三个关键的门控单元构成：输入门、遗忘门和输出门，它们共同决定细胞状态的更新和最终的隐藏状态输出。这些门控单元通过 $sigmoid$ 函数产生介于0到1之间的值，分别代表对新信息的接纳程度、对旧信息的遗忘程度、对细胞状态暴露给输出的程度



$sigmoid$ 函数的值为0-1之间，当为0时，不传送，当为1时，全部传送。

遗忘门：决定前一时刻细胞状态中哪些信息需要被遗忘
W_f 为权重矩阵, h_{t-1} 是上一时刻的隐藏状态, x_t 是当前输入, b_f 为偏置项

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{7}$$

输入门：决定当前时刻输入中哪些信息应被加入到细胞状态
其中 i_t 为信息的接纳权重, C_t 为候选状态

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$
$$\tilde{C}_t = tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$$

细胞状态更新：结合遗忘门和输入门的结果，更新细胞状态 C_t

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \tag{8}$$

输出门：决定细胞状态中哪些信息应被传递到下一时刻的隐藏状态或作为当前时刻的模型输出。输出门的激活值o_t由当前输入x_t和前一时刻隐藏状态h_{t-1}通过一个带有sigmoid激活函数的全连接层计算得到，然后与细胞状态经过tanh函数后的值按元素乘积得到最终的隐藏状态h_t

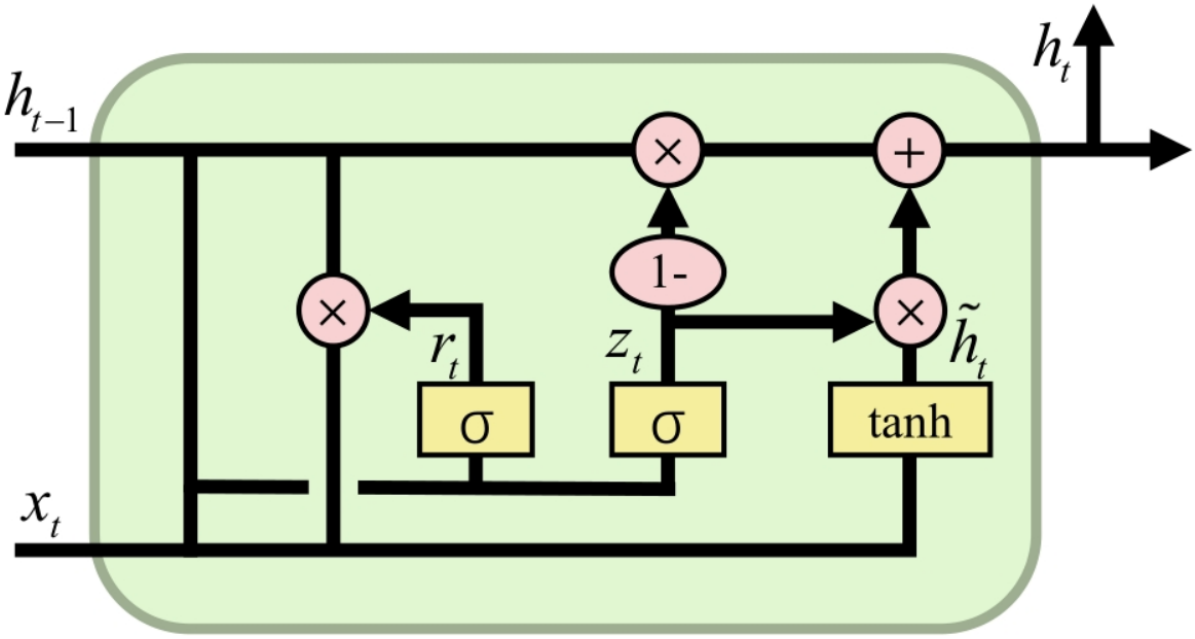
$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$$
$$h_t = o_t * tanh(C_t)$$

遗忘门允许模型根据当前输入选择性地"遗忘"过去细胞状态中的信息；输入门负责筛选当前时刻输入的信息，将其整合到新的候选状态中；输出门决定细胞状态中哪些信息应作为隐藏状态输出，并传递到后续层或作为模型输出。这种设计使得 $LSTM$ 能够有效地捕捉和建模长期依赖关系。使它在处理序列数据时比传统的 RNN 模型更具优势。一定程度上解决了梯度消失或梯度爆炸的问题，使得网络能够更稳定地进行训练

双向 $LSTM$ 的结构比 $BiRNN$ 更复杂，在计算和存储上需求都更高。 $LSTM$ 中的门控单元和记忆单元增加了网络的参数量，特别是当网络层数较多时，参数量会进一步增加。门控结构和复杂的记忆单元使得网络的决策过程相对难以解释和理解。

Gated Recurrent Unit（GRU）

$GRU(Gatedrecurrentunit)$ 是 $LSTM$ 网络的一种效果很好的变体，只有更新门和重置门。
GRU 的门控机制允许选择性的信息保留和遗忘，比传统 RNN 更擅长捕获长期依赖性。且结构相对简单，参数较少，与其他类型的循环神经网络相比需要更少的训练时间，且不易过拟合。能够一定程度解决长期依赖和反向传播中的梯度消失或梯度爆炸问题。



重置门：

$$r_t = \sigma(W_r x_t + U_r h_{r-1} + b_r) \tag{9}$$

更新门：

$$z_t = \sigma(W_z x_t + U_z h_{r-1} + b_z) \tag{10}$$

更新表达式：门控信号 z_t 的范围为0~1。门控信号越接近1，表示”记忆“的数据越多；越接近0则表示”遗忘“的数据越多。

$$\tilde{h_t} = tanh(W_h x_t + U_h(r_t \odot h_{t-1}) + b_h)$$
$$h_t = z_t \odot h_{t-1} + (1 - z_t) \odot \tilde{h}_t$$

GRU 使用了同一个门控 z_t ，即可同时可以进行遗忘和记忆。参数比 $LSTM$ 少，在某些较小的数据集上， GRU 相比于 $LSTM$ 能表现出更好的性能。但在需要对复杂的顺序依赖关系进行建模的任务中，它的表现可能不如 $LSTM$ 。