文章编号：1003-0077（2011）00-0000-00

基于联合注意力机制的篇章级机器翻译*

李京谕[1,2]，冯 洋[1,2]

（1.中国科学院 计算技术研究所 智能信息处理重点实验室，北京 100190；

2. 中国科学院大学，北京 100049）

摘要：近年来，神经机器翻译（Neural machine translation, NMT）表现出极大的优越性，然而如何在翻译一个文档时考虑篇章上下文信息仍然是一个值得探讨的问题。传统的注意力机制对源端的所有词语进行计算，而在翻译当前句子时篇章中大量的信息中只有小部分是与之相关的。在篇章级机器翻译中，采用传统的注意力机制建模篇章信息存在着信息冗余的问题。该文提出了一种联合注意力机制，结合“硬关注”和“软关注”的机制对篇章上下文的信息进行建模。关键思想是通过“硬关注”筛选出与翻译当前句子相关的源端历史词语，然后采用“软关注”的方法进一步抽取翻译中所需的上下文信息。实验表明，相比于基线系统，该方法能使翻译性能获得明显提升。

关键词：神经机器翻译，注意力机制，篇章级机器翻译

中图分类号：TP391　　**文献标识码：**A

Co-attention Mechanism for Document-level Neural Machine Translation

LI Jingyu[1,2], FENG Yang[1,2]

(1.Keylab of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China;

2. University of Chinese Academy of Sciences, Beijing 100049, China)

Abstract: Neural machine translation (NMT) has achieved remarkable progress in recent years. However, how to model contextual information when translating a document is still a problem worth discussing. Traditional attention mechanism considers all the words in source sentences, whereas translating a sentence relies solely on very sparse tokens in large document. In document-level neural machine translation, this results in the problem of information redundancy when dealing with long-range contextual sentences. For this purpose, we introduce a co-attention mechanism to capture the context which combines hard attention and soft attention. Specifically, the hard approach is used to select the source historical words related to the current sentence to be translated, and then use soft attention mechanism to further extract the context information needed in the current translation. Experiments show that our method leads to strong improvements in translation quality, greatly outperforming the baseline models.

Key words: Neural Machine Translation; Attention Mechanism; Document-level Neural Machine Translation

1 引言

神经机器翻译（Neural Machine Translation, NMT）是目前主流的一种机器翻译建模方法[1][2][3]，利用神经网络搭建翻译模型，并采用端到端（End-to-end）的方式进行优化。神经机器翻译提出至今，主要的关注点都在于句子级的翻译，即给定一个段落，翻译模型逐句进行翻译，句子与句子之间是相互独立的，这忽略了篇章上下文信息在翻译的过程中的影响。一方面，在翻译一个完整的段落时，句子与句子之间需要保持一致和连贯，如果忽略篇章上下

* **收稿日期：**2019-06-15　　**定稿日期：**2019-08-15

作者简介：李京谕（1995—），女，硕士，主要研究方向机器翻译、自然语言处理；冯洋（1982—），女，副研究员，主要研究方向机器翻译、自然语言处理、机器学习。

文的信息，可能造成语义不连贯、语句不通顺的现象。另一方面，篇章上下文信息可以提供给句子一些辅助信息，在翻译的过程中减少句子存在的歧义问题。

神经机器翻译中根据注意力机制对源语言句子中的所有词语生成对齐概率，这种对所有词语进行计算的方式被称之为“软关注”（Soft Attention）。在篇章级别的机器翻译中，篇章中的上下文信息有篇幅长、信息量多的特点，但是在实际情况下，对翻译句子有帮助的篇章信息往往十分有限。在篇章信息存在大量冗余的情况下，采用传统的注意力机制在篇章机器翻译中很难从中提取对翻译有实际帮助的信息，这在篇章机器翻译中是一个不可忽视的问题。针对篇章信息冗余的现象，本文提出一种“硬关注”（Hard Attention）的方式计算注意力，并应用在篇章机器翻译的任务上。“软关注”的方式计算注意力时，每个输入对应的隐状态都参与了权重计算，这种方法便于训练中梯度的反向传播。对应地，我们提出在对篇章信息进行注意力的计算时，只赋予 0 和 1 这两种权重，这也使得模型难以进行梯度更新。因此，我们在篇章翻译中通过强化学习[4]对硬关注模型梯度更新。在这里，引入强化学习有两个目的。其一是由于硬关注的机制，在离散的信号中梯度无法回传的问题。通过强化学习的方法，我们可以获得注意力机制的奖励信号，从而对模型进行梯度更新。第二个问题是由于求解注意力本身是一个无监督、无标签的问题，所以无法采用有监督的任务中神经网络的损失函数对注意力进行建模，而强化学习的方法可以解决这个问题。

本文提出了一种联合注意力机制，将硬关注和软关注两种注意力机制相结合，共同在篇章级机器翻译模型中对篇章上下文建模。联合注意力机制的关键思想是通过“hard”的方法筛选出对翻译当前句子有帮助的部分篇章上下文的相关状态，在翻译的每一步对候选状态进行“soft”的方式进一步提取篇章信息，通过两种注意力模型结合的方式得到每一步的篇章上下文的向量表示。通过在原始的神经机器翻译模型中融入篇章信息，构成一个针对篇章数据结的神经机器翻译模型。在两个不同领域数据集上的实验表明，通过联合注意力机制引入篇章信息的方法对机器翻译模型的性能有明显的提升。

2 相关研究

注意力机制提出以来，被广泛应用到了包括自然语言处理在内的许多研究领域。虽然注意力机制最初是被用在机器翻译上，但随后在各种任务上都占有一席之地，针对注意力的改进也一直是研究的热点。在机器翻译中，许多研究针对注意力机制提出优化和改进。针对上下文信息较长的情况下，注意力机制对齐困难的情况，Luong 等[5]提出了局部注意力（Local Attention）机制，在计算注意力寻找源端对齐信息的时候，局部注意力仅对一个窗口范围内的词进行分布式表示，而不是对整个句子的所有源端表示做加权求和，通过一个位置对齐参数计算当前时刻对应注意力的位置，然后对一个固定大小的窗口范围内所有的隐状态进行权重计算。Vaswani 等[3]提出的自注意力（Self-Attention）和多头注意力（Multi-Head Attention）也是对注意力机制的改进形式。

近年来，篇章级机器翻译逐渐成为机器翻译领域里的一个研究热点。Wang 等[6]首先尝试在神经机器翻译模型中引入篇章信息，在 RNNSearch 模型的基础上利用层次化的循环神经网络建模篇章信息，采用两个级别的 RNN 分别对词向量和句子向量进行编码，得到代表整个篇章信息的向量表征。Jean 等[7]在 RNNSearch 模型中加入一套额外的编码器和注意力机制，将篇章的前一个句子的信息引入神经机器翻译模型中。Tu 等[8]为篇章信息设计了一个类似高速缓存（cache）的结构，Maruf 等[9]采用额外的记忆单元（Memory Networks）存储篇章信息，以扩大对篇章信息的利用范围。在基于自注意力机制的 Transformer 模型的基

础上，Miculicich 等[10]实现了基于多头注意力机制的层次化网络，分别对篇章进行词级别和句子级别的表示。Zhang 等[11]在 Transformer 模型中加入一个额外的篇章信息编码器，对篇章中源端的历史句子进行编码。Xiong 等[12]提出了一种多轮解码方案，将篇章一致性作为强化学习的一个奖励函数来优化模型。

3 基线系统

给定一个平行语料构成的数据集 $\mathrm{S}=\{(\mathbf{x}^{(n)},\mathbf{y}^{(n)})\}_{n=1}^{N}$，翻译模型的优化目标是求解参数 θ，使得 $\arg\max_{\theta} P(\mathbf{y}\mid\mathbf{x})$。$\mathbf{x}=(x_1,x_2,\ldots,x_I)$ 表示源语言句子，$\mathbf{y}=(y_1,y_2,\ldots,y_J)$ 表示目标语言句子，条件概率 $P(\mathbf{y}\mid\mathbf{x})$ 表示一句话被翻译的概率，通过翻译模型对目标端词语进行预测：

$$P(\mathbf{y}\mid\mathbf{x};\theta)=\prod_{j=1}^{J}P(y_j\mid\mathbf{y}_{<j},\mathbf{x};\theta) \tag{1}$$

其中，$\mathbf{y}_{<j}=\{y_1,\ldots,y_{j-1}\}$ 表示目标端的前驱输出序列。本文将 Transformer 模型作为基线系统，Transformer 模型基于编码器-解码器结构，编码器和解码器都由重复的网络栈式地搭建了 N_c 层。

3.1 编码器

给定一个源端序列 $\mathbf{x}=(x_1,x_2,\ldots,x_I)$，$I$ 表示源端句子的长度。由于注意力机制在建模中丢失了时序信息，在 Transformer 中采用了一个 Positional Encoding 层拟合了位置编码函数来模拟词语序列的时序信息：

$$PE_{pos,2i}=\sin\left(\frac{pos}{10000^{2i/d_{\text{model}}}}\right), \tag{2}$$

$$PE_{pos,2i+1}=\cos\left(\frac{pos}{10000^{2i/d_{\text{model}}}}\right), \tag{3}$$

通过词向量和 Positional Encoding 层相加的方式，可以得到源端语言句子 $\mathbf{x}$ 的向量表示 $\boldsymbol{E}_x=[\boldsymbol{E}[x_1];\ldots;\boldsymbol{E}[x_I]]\in\mathbb{R}^{d_{\text{model}}\times I}$，$d_{\text{model}}$ 表示模型的隐层维度。

Transformer 编码器由 N_c 层相同的网络结构构成，每一层都由一个多头注意力子层和一个前向神经网络子层组成。多头注意力模块对源语言输入序列之间的依赖关系建模，捕获源语言句子中的内部结构，编码器中第 n 层计算如下式所示：

$$\boldsymbol{A}^{(n)}=\mathrm{MultiHead}(\boldsymbol{H}^{(n-1)},\boldsymbol{H}^{(n-1)},\boldsymbol{H}^{(n-1)}), \tag{4}$$

这里，$\boldsymbol{A}^{(n)}\in\mathbb{R}^{d_{\text{model}}\times I}$ 表示第 n 层多头注意力模块的隐状态，$\boldsymbol{H}^{(n-1)}$ 表示第 $n-1$ 层编码器的隐状态。当 n=1 时，第一层编码器的输入为 $\boldsymbol{H}^{(0)}=\boldsymbol{E}_x$。$\mathrm{MultiHead}(\mathbf{Q},\mathbf{K},\mathbf{V})$ 表示多头注意力函数，将输入映射到 h 个不同的子空间进行注意力的计算，这里 $\mathbf{Q}$=$\mathbf{K}$=$\mathbf{V}$ 用作计算自注意力。由于 Transformer 模型使用深层次的网络结构，在每两个子层之间都采用残差结构（Add）[13]和对层的规范化（LayerNorm）[14]来提升模型的能力，为简便起见，后文在书写中省略这一过程：

$$\boldsymbol{A}^{(n)}=\mathrm{LayerNorm}(\boldsymbol{A}^{(n)}+\boldsymbol{H}^{(n-1)}). \tag{5}$$

第二个子层是前馈网络层 $\mathrm{FFN}(\cdot)$，加强编码器每个位置的表示能力。形式化如下：

$$\boldsymbol{H}^{(n)}=[\mathrm{FFN}(\boldsymbol{A}^{(n)}_{,1});\ldots;\mathrm{FFN}(\boldsymbol{A}^{(n)}_{,I})], \tag{6}$$

其中，$\boldsymbol{H}^{(n)}\in\mathbb{R}^{d_{\text{model}}\times I}$ 表示第 n 层编码器对源语言序列的向量表示，$\boldsymbol{A}^{(n)}_{,t}$ 是第 n 子层对第 t 个词的表示。$\mathrm{FFN}(\cdot)$ 表示前馈神经网络，由两层全连接网络构成，ReLU 函数作为两

层网络之间的激活函数，

$$\mathrm{FFN}(\boldsymbol{x})=\max(0,\boldsymbol{x}\boldsymbol{W}_1+\boldsymbol{b}_1)\boldsymbol{W}_2+\boldsymbol{b}_2, \tag{7}$$

其中，$\boldsymbol{W}_1\in\mathbb{R}^{d_{\mathrm{model}}\times d_{\mathrm{ff}}}$、$\boldsymbol{W}_2\in\mathbb{R}^{d_{\mathrm{ff}}\times d_{\mathrm{model}}}$ 分别表示可训练的参数矩阵。

3．2 解码器

对目标端序列 $\mathbf{y}=(y_1,y_2,\ldots,y_J)$，$J$ 表示目标端句子的长度，通过目标端的词向量和 Positional Encoding 层，得到 $\mathbf{y}$ 的向量表示 $\boldsymbol{E}_y=[\boldsymbol{E}[y_1],\ldots,\boldsymbol{E}[y_J]]\in\mathbb{R}^{d_{\mathrm{model}}\times J}$。

解码器由 N_c 层相同的网络结构构成，每一层都由两个多头注意力子层和一个前馈网络子层组成：

$$\boldsymbol{F}^{(n)}=\mathrm{MultiHead}(\boldsymbol{S}^{(n-1)},\boldsymbol{S}^{(n-1)},\boldsymbol{S}^{(n-1)}), \tag{8}$$

$$\boldsymbol{G}^{(n)}=\mathrm{MultiHead}(\boldsymbol{F}^{(n-1)},\boldsymbol{H}^{(N_c)},\boldsymbol{H}^{(N_c)}), \tag{9}$$

第一个多头注意力子层计算自注意力，对目标端序列之间的依赖关系建模。解码器的第一层输入为 $\boldsymbol{S}^{(0)}=\boldsymbol{E}_y$。第二个多头注意力子层是用于建模目标语言与源端语言之间的依赖关系，多头注意力的输入为编码器最顶层的隐状态 $(\mathbf{K}=\mathbf{V})$。最后通过前馈网络子层，加强解码器的表示能力，

$$\boldsymbol{S}^{(n)}=[\mathrm{FFN}(\boldsymbol{G}_{:,1}^{(n)});\ldots;\mathrm{FFN}(\boldsymbol{G}_{:,J}^{(n)})],. \tag{10}$$

这里，$\boldsymbol{S}^{(n)}\in\mathbb{R}^{d_{\mathrm{model}}\times J}$ 为第 n 层解码器的隐状态 $(n=1,\ldots,N_c)$。

将解码器顶层的隐状态 $\boldsymbol{S}^{(N_c)}=(\boldsymbol{s}_1,\ldots,\boldsymbol{s}_J)$ 映射到目标端词表的空间，经过 *softmax* 函数计算得到对目标端的概率分布，对第 j 个位置：

$$P(y_j\mid\mathbf{y}_{<j},\mathbf{x};\theta)\propto\exp(\mathbf{W}_O\mathbf{s}_j), \tag{11}$$

其中，$\mathbf{W}_O\in\mathbb{R}^{|\mathrm{V}_y|\times d_{\mathrm{model}}}$ 是一个参数矩阵，V_y 表示目标端词表，$\boldsymbol{s}_j\in\mathbb{R}^{d_{\mathrm{model}}\times 1}$ 表示解码器顶层 $\boldsymbol{S}^{(N_c)}$ 第 j 个位置的隐状态。

4 基于联合注意力的篇章机器翻译

4．1 问题定义

具体地，对篇章级机器翻译数据集 $\mathrm{S}=\{(\mathbf{x}^{(n)},\mathbf{y}^{(n)})\}_{n=1}^{N}$，其中任意一组数据 $(\mathbf{x}^{(K)},\mathbf{y}^{(K)})$，我们可以获得对应的篇章上文 $\{(\mathbf{x}^{(1)},\mathbf{y}^{(1)}),\ldots,(\mathbf{x}^{(K-1)},\mathbf{y}^{(K-1)})\}$，简便起见，将 $(\mathbf{X}_{<K},\mathbf{Y}_{<K})$ 作为篇章中上文的 $K-1$ 个句对的表示。篇章级别的翻译概率定义为：

$$\begin{aligned}&P(\mathbf{y}^{(1)},\ldots,\mathbf{y}^{(K)}\mid\mathbf{x}^{(1)},\ldots,\mathbf{x}^{(K)};\boldsymbol{\theta})\\&=\prod_{k=1}^{K}P(\mathbf{y}^{(k)}\mid\mathbf{X}_{<k},\mathbf{Y}_{<k},\mathbf{x}^{(k)};\boldsymbol{\theta})\end{aligned} \tag{12}$$

其中，$\mathbf{X}_{<k}=(\mathbf{x}^{(1)},\ldots,\mathbf{x}^{(k-1)})$ 表示篇章中前 $k-1$ 个句子的源端序列，$\mathbf{Y}_{<k}=(\mathbf{y}^{(1)},\ldots,\mathbf{y}^{(k-1)})$ 表示篇章中前 $k-1$ 个句子的输出序列。

前人的研究工作中[6][7][11]，由于目标端依次解码带来的误差累积问题，目标端的篇章信息 $\mathbf{Y}_{<k}$ 通常在建模中省略，翻译概率用以下形式计算：

$$
\begin{aligned}
& P(\mathbf{y}^{(1)},\ldots,\mathbf{y}^{(K)} \mid \mathbf{x}^{(1)},\ldots,\mathbf{x}^{(K)};\boldsymbol{\theta}) \\
\approx & \prod_{k=1}^{K} P(\mathbf{y}^{(k)} \mid \mathbf{X}_{<k},\mathbf{x}^{(k)};\boldsymbol{\theta}), \\
= & \prod_{k=1}^{K}\prod_{j=1}^{J_k} P(y_j^{(k)} \mid \mathbf{X}_{<k},\mathbf{x}^{(k)},\mathbf{y}_{<j}^{(k)};\boldsymbol{\theta}),
\end{aligned}
\tag{13}
$$

其中，$y_j^{(k)}$表示第k个句子中的第j个目标端词，$\mathbf{y}_{<j}^{(k)} = y_1^{(k)},\ldots,y_{j-1}^{(k)}$表示第$k$个句子已经生成的目标端序列。由上式可知，篇章级翻译模型与句子级翻译模型最大的区别在于对篇章信息$\{\mathbf{x}_1,\ldots,\mathbf{x}_K\}$进行建模。在本文中我们考虑篇章中源端的上文信息，即$\mathbf{X}_{<K}$。

我们提出的模型如图 1 所示，这里只示意了模型的编码器结构，模型的解码器与原始 Transformer 模型中相同，在图中没有给出具体的细节描述。联合注意力机制分为硬关注和软关注两个部分。其中，硬关注是对篇章上下文和源端句子之间的关系建模，相当于是从上下文中选择一个子集，能够为翻译当前句子筛选出所有需要的篇章上下文的部分。软关注在源端句子的每一个位置对硬关注选择的集合进行更进一步的计算，从而动态地获得翻译不同位置时篇章信息的向量表示。

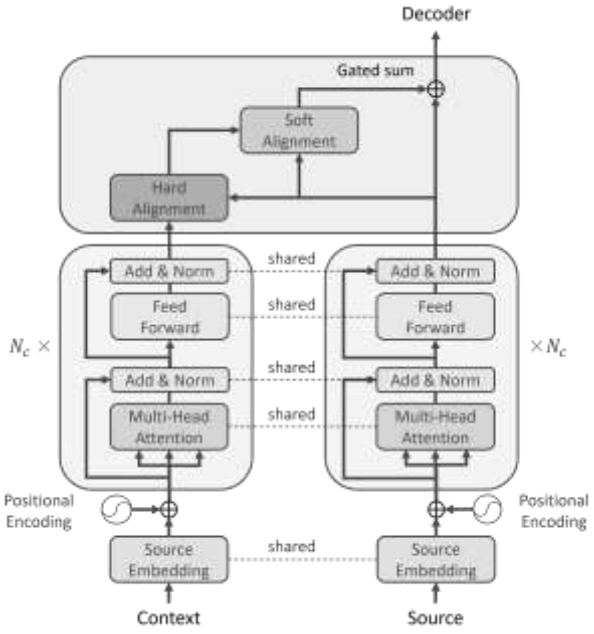

图 1　基于联合注意力机制的编码器结构

对于一个给定的源语言句子$\mathbf{x}^{(K)} = (x_1^{(K)},\ldots,x_I^{(K)})$，通过词向量和 Positional Encoding 层相加得到其向量表示，

$$
\boldsymbol{E}_x = [\boldsymbol{E}[x_1^{(K)}];\ldots;\boldsymbol{E}[x_I^{(K)}]] \tag{14}
$$

其中，$\boldsymbol{E}_x \in \mathbb{R}^{d_{\text{model}} \times I}$，$I$表示句子的长度。然后通过 Transformer 编码器对源语言进行编码表示，当输入$\boldsymbol{H}^{(0)} = \boldsymbol{E}_x$时，得到编码器最顶层隐状态$\boldsymbol{H}^{(N_c)}$为编码器对源语言句子$\boldsymbol{E}_x$的向量表示，这里用$\boldsymbol{H}^{cur} \in \mathbb{R}^{d_{\text{model}} \times I}$表示编码器最顶层的隐状态。对$\mathbf{x}^{(K)}$的篇章上文$\mathbf{X}_{<K}$中每个句子分别进行上述操作，得到所有篇章上下文句子的编码表示$(\boldsymbol{H}_1^{doc},\ldots,\boldsymbol{H}_{K-1}^{doc})$，将所有句子的表示拼接起来：

$$
\boldsymbol{H}^{doc} = [\boldsymbol{H}_1^{doc};\ldots;\boldsymbol{H}_{K-1}^{doc}], \tag{15}
$$

其中，$\boldsymbol{H}^{(doc)} \in \mathbb{R}^{d_{model} \times L}$.为篇章上下文的向量表示，$L$ 表示篇章信息的长度。

联合注意力模型中，根据源语言句子和篇章上下文的向量表示计算注意力，硬关注将注意力作为采样的概率来选择上下文的部分区域，可以用 Attention_{hard} 函数表示：

$$\boldsymbol{C} = \text{Attention}_{hard}(\boldsymbol{H}^{doc}, \boldsymbol{H}^{cur}), \tag{16}$$

其中，$\boldsymbol{C} = [\boldsymbol{H}^{doc}_{\cdot,1}; \ldots; \boldsymbol{H}^{doc}_{\cdot,K}]$ 本质上是一个候选集合，通过硬关注的计算将特定位置上的 $\boldsymbol{H}^{doc}$ 向量拼接起来得到的。软关注与传统的注意力模式相同，对源端语言的表示 H_x 和候选集合 $\boldsymbol{C}$ 进行注意力的计算。这里用 Attention_{soft} 函数表示：

$$\boldsymbol{D} = \text{Attention}_{soft}(\boldsymbol{C}, \boldsymbol{H}^{cur}), \tag{17}$$

其中，$\boldsymbol{D}$ 为联合注意力模型得到的篇章上下文的隐状态表示。最后将篇章信息和源语言编码融合起来作为编码器的最终表示：

$$\boldsymbol{H} = \text{Gate}(\boldsymbol{D}, \boldsymbol{H}^{cur}), \tag{18}$$

其中，$\text{Gate}(\cdot)$ 表示一个门控单元函数，$\boldsymbol{H}$ 为源语言融合篇章信息的编码表示。翻译模型的解码器结构与原始 Transformer 相同，如 2.2 节中所述，这里不再赘述。

4.2 基于强化学习的硬关注模块

硬关注模块，即公式**错误!未找到引用源。**，可以以二分类任务的形式进行定义。如果一个上下文向量被分为了有关联的类别，就将这个上下文的隐状态加入候选集合 $\boldsymbol{C}$，提供给翻译的过程使用，如果没有被选中，则将其舍去。对源语言和上下文的表示 $\boldsymbol{H}^{cur}$、$\boldsymbol{H}^{doc}$，分类器生成与篇章上下文词语个数 L 等长的二进制序列 $\boldsymbol{z} = (z_1, \ldots, z_L)$，其中，$z_l = 1$ 表示选择 $z_l = 0$ 表示 $\boldsymbol{H}^{doc}_{\cdot,l}$ 对当前待翻译句而言是冗余信息。

对第 l 个位置的上下文，将对应的隐状态 $\boldsymbol{H}^{doc}_{\cdot,l}$，与序列 $\boldsymbol{H}^{cur}$ 一起输入分类器，得到上下文与源语言句子有依赖关系的概率：

$$p(z_l \mid \boldsymbol{H}^{cur}, \boldsymbol{H}^{doc}; \theta_r) \propto \exp(f(\boldsymbol{H}^{cur}, \boldsymbol{H}^{doc}_{\cdot,l})), \tag{19}$$

其中，函数 $f(\cdot)$ 计算 $\boldsymbol{H}^{cur}, \boldsymbol{H}^{doc}_{\cdot,l}$ 之间的依赖关系，其输出分布在 0 到 1 之间，作为篇章上下文 $\boldsymbol{H}^{doc}$ 第 l 个位置上的隐状态与源语言句子相关的概率。进一步地，由于生成 $\boldsymbol{z} = (z_1, \ldots, z_L)$ 是一个序列决策的过程，所以这里采用循环神经网络建模函数 $f(\cdot)$。

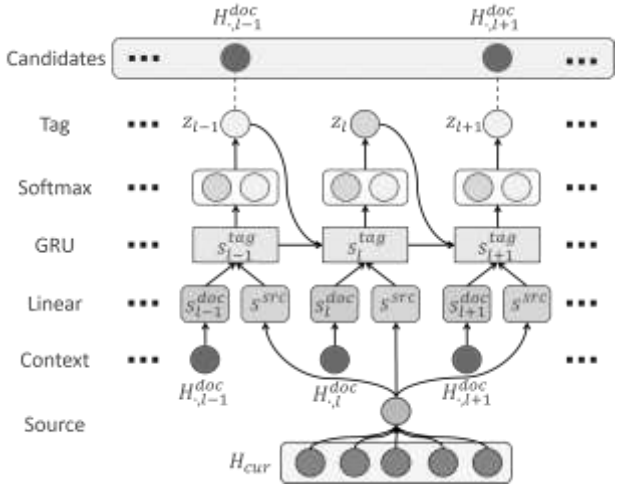

图 2　联合注意力中的硬关注模块

如图 2 所示，采用两个前向网络分别将 $\boldsymbol{H}^{cur}$ 和 $\boldsymbol{H}^{doc}$ 映射到不同的空间内，

$$\boldsymbol{s}^{src} = \tanh(\boldsymbol{W}_h \text{pooling}(\boldsymbol{H}^{cur}) + \boldsymbol{b}_h), \tag{20}$$

$$\boldsymbol{s}_l^{doc} = \tanh(\boldsymbol{W}_c \boldsymbol{H}_{\cdot,l}^{doc} + \boldsymbol{b}_c), \tag{21}$$

其中，$\boldsymbol{s}^{src}$ 代表硬关注模块对源端句子的表示，$\boldsymbol{s}_l^{doc}$ 代表硬关注模块对第 l 个位置上篇章信息的表示。$\boldsymbol{W}_h \in \mathbb{R}^{d_{\text{model}} \times d_{\text{model}}}, \boldsymbol{W}_c \in \mathbb{R}^{d_{\text{model}} \times d_{\text{model}}}$ 为参数矩阵，$\text{pooling}(\cdot)$ 表示池化操作，也就是对 $\boldsymbol{H}^{cur}$ 在序列长度的维度上取平均值。通过循环神经网络预测生成 $\boldsymbol{z}$ 的概率。在第 l 个位置：

$$\boldsymbol{s}_l^{tag} = f(\boldsymbol{s}^{src}, [\boldsymbol{s}_l^{doc}; \boldsymbol{z}_{l-1}]), \tag{22}$$

其中，$\boldsymbol{z}_{l-1}$ 为前一时刻标签的向量表示。$f(\cdot)$ 表示单层 GRU 函数。将 $\boldsymbol{s}_l^{tag}$ 映射为分类标签上的概率分布：

$$P_z(z_l \mid z_{<l}, \boldsymbol{H}^{cur}, \boldsymbol{H}_{\cdot,l}^{doc}; \theta_r) = \text{softmax}(\boldsymbol{W}_p \boldsymbol{s}_l^{tag}), \tag{23}$$

其中，$\boldsymbol{W}_p \in \mathbb{R}^{2 \times d_{\text{model}}}$ 为参数矩阵，$P_z(z_l)$ 表示对 z_l 的概率分布。由此，可以求解整个标签序列 $\boldsymbol{z} = z_1, \ldots, z_L$ 的概率：

$$\begin{aligned} & P_z(\boldsymbol{z} \mid \boldsymbol{H}^{cur}, \boldsymbol{H}^{doc}; \theta_r) \\ = & \prod_{l=1}^{L} P_z(z_l \mid z_{<l}, \boldsymbol{H}^{cur}, \boldsymbol{H}_{\cdot,l}^{doc}; \theta_r), \end{aligned} \tag{24}$$

通过生成的决策序列 $\boldsymbol{z} = (z_1, \ldots, z_L)$，将所有 $z_l = 1$ 对应位置中上下文的隐状态进行拼接，得到硬关注模块的输出 $\boldsymbol{C} = [\boldsymbol{H}_{\cdot,1}^{doc}; \ldots; \boldsymbol{H}_{\cdot,L'}^{doc}]$。由于硬关注是一个没有标签的任务，并且 $\boldsymbol{z}$ 是一个离散变量，在反向传播的过程中是一个不可微的问题。因此，我们将学习硬关注的参数 θ_r 作为一个强化学习的问题，并应用策略梯度算法[15]进行梯度更新。

4. 3 基于多头注意力的软关注模块

硬关注模块筛选出与翻译当前句子相关的向量集合，然而，不同的篇章上下文向量在翻译的每一个时间步的作用都占有不同的权重。因此，我们设计了软关注模块，如图 3，计算上下文表示 $\boldsymbol{C}$ 和 $\boldsymbol{H}^{cur}$ 之间的依赖关系，从而得到翻译的每一个时间步中相关的篇章信息向量。

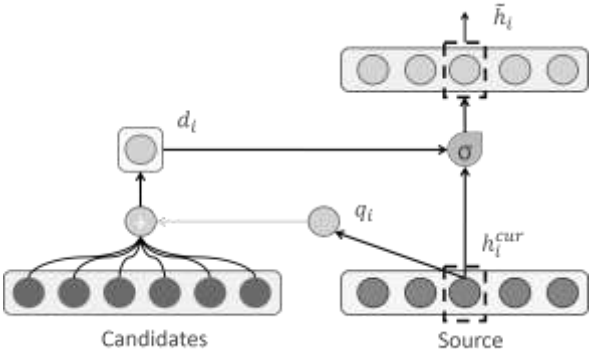

图 3　联合注意力中的软关注模块和上下文门控单元

我们采用多头注意力模块实现软关注。对 $\boldsymbol{H}^{cur} = [\boldsymbol{h}_1^{cur}; \ldots; \boldsymbol{h}_I^{cur}]$，源端待翻译句的第 i 个位置，将编码器的隐状态 $\boldsymbol{h}_i^{cur}$ 映射到另一个空间内作为计算注意力的 $\mathbf{Q}$，硬关注的输出 $\boldsymbol{C}$ 作为注意力的 $\mathbf{K}$ 和 $\mathbf{V}$（$\mathbf{K}$=$\mathbf{V}$），从而得到该位置的篇章信息表示 $\boldsymbol{d}_i$。形式化为：

$$\boldsymbol{q}_i = f_q(\boldsymbol{h}_i^{cur}), \tag{25}$$

$$\boldsymbol{d}_i = \mathrm{FFN}(\mathrm{MultiHead}(\boldsymbol{q}_i, \boldsymbol{C}, \boldsymbol{C})), \tag{26}$$

这里，$f_q(\cdot)$是一个线性变换函数，$\boldsymbol{q}_i$是用于计算多头注意力的$\mathbf{Q}$。$\mathrm{MultiHead}(\cdot)$表示多头注意力函数，$\mathrm{FFN}(\cdot)$表示一个前馈网络。

4. 4 上下文门控单元

在翻译当前句的每个位置时，对篇章信息的依赖程度都不相同。因此，获得篇章上下文的向量表示后，通过一个门控单元（gate）[16]学习句子与和篇章信息之间的关联，动态地控制句子信息和篇章信息对翻译的影响，如图 3 所示。

在待翻译句的第i个位置，门控单元通过编码器的隐状态$\boldsymbol{h}_i^{cur}$和篇章向量$\boldsymbol{d}_i$决定在翻译中所占的比率：

$$\lambda_i = \sigma(\boldsymbol{W}_h \boldsymbol{h}_i^{cur} + \boldsymbol{W}_d \boldsymbol{d}_i), \tag{27}$$

$$\boldsymbol{h}_i = \lambda_i \boldsymbol{h}_i^{cur} + (1-\lambda_i)\boldsymbol{d}_i, \tag{28}$$

其中，$\mathbf{W}_h$、$\mathbf{W}_d$分别表示参数矩阵。λ_i在经过 *sigmoid* 函数计算后输出在 0 到 1 之间，定义了篇章信息通过的程度。

将门控单元输出的隐状态$\boldsymbol{h}_i$代替原始状态$\boldsymbol{h}_i^{cur}$，则编码器对输入源语言$\mathbf{x}$的最终表示为$\boldsymbol{H}^{mix} = [\boldsymbol{h}_1; \ldots; \boldsymbol{h}_I]$，代替原始 Transformer 模型中第$N_c$层的隐状态$\boldsymbol{H}^{(N_c)}$输入到解码器中。篇章翻译模型的解码器与原始的 Transformer 模型相同。

4. 5 模型的训练

在本文提出的篇章翻译模型中，参数可以分为两个部分：硬关注模块的θ_r和其余所有部分的θ_s。

由于硬关注模块中存在离散的过程，其目标函数是不可微分的，对参数θ_r的优化采用策略梯度算法。可以认为硬关注模型为强化学习中的智能体，实际上是一个策略网络，其动作空间的大小为 2，每一步可以执行选中或者未选中两种动作。智能体根据全局信号做出决策，在智能体执行一系列动作之后，收到一个环境中得到的反馈信号。这里，我们将翻译模型生成目标译文的翻译概率作为奖励，对待翻译的数据对$(\mathbf{x}, \mathbf{y})$和对应的篇章上文$\mathbf{X}_{<K}$：

$$\begin{aligned} R(z_{1:L}) &= P(\mathbf{y} \mid \mathbf{X}_{<K}, \mathbf{x}; \theta_s) \\ &= \prod_{j=1}^{J} P(y_j \mid \mathbf{X}_{<K}, \mathbf{x}, \mathbf{y}_{<j}; \theta_s), \end{aligned} \tag{29}$$

参数θ_r的优化学习被建模为通过策略梯度方法（即 REINFORCE 算法）解决的强化学习问题。训练硬关注的总体目标是选择部分上下文的同时保留与翻译相关的信息，其目标函数是得到奖励函数的最大期望，通过翻译概率得到：

$$J_r(\theta_r) = \mathrm{E}_{[z_{1:L} \sim p_{\theta_r}]} R(z_{1:L}), \tag{30}$$

对其余参数θ_s，采用原始的端到端的方式直接通过反向传播来进行优化。使用交叉熵作为损失函数，其优化目标为：

$$J_s(\boldsymbol{\theta}_s) = \sum_{n=1}^{N} \log P(\mathbf{y}^{(n)} \mid \mathbf{X}_{<K}^{(n)}, \mathbf{x}^{(n)}; \boldsymbol{\theta}_s), \tag{31}$$

如式**错误!未找到引用源。**，硬关注模块的训练过程中将翻译模型输出的概率作为其奖励函数，而在翻译模型的其余参数θ_s的训练中，同样依赖于硬关注模块的输出。因此，参数θ_r和θ_s两个部分的优化是彼此依赖的，如何进行有效的训练是本文中的一大难点。

在实际的训练过程中，为了降低训练难度，采用预训练（pre-train）和交替训练（cross-trian）的方式。首先最大似然估计的标准训练 Transformer 模型直至收敛：

$$J(\boldsymbol{\theta}) = \sum_{n=1}^{N} \log P(\mathbf{y}^{(n)} \mid \mathbf{x}^{(n)}; \boldsymbol{\theta}), \tag{32}$$

这样，模型能具有相对完整的表示能力。在此基础上，对我们提出的模型采用交替训练的方式，对两个部分的参数分别进行训练，训练中需要把另一部分参数固定。对于 θ_s 的优化，硬关注模型生成完整的序列之后，才能得到在翻译中使用的篇章信息，从而通过翻译概率计算其奖励，因此在采用式更新中，通过蒙特卡洛搜索（Monte-Carlo Search）的方式得到选择动作序列。对参数 θ_s 进行优化时，也需要通过硬关注生成完整的选择序列，采用柱搜索（Beam Search）的方法得到完整序列，与机器翻译中柱搜索方式略微不同之处在于，这里生成一个定长的决策序列 $\boldsymbol{z} = (z_1, \ldots, z_L)$，其长度与篇章上下文相同。由于硬关注模型的动作空间非常小（选择/不选择），这里柱搜索的窗口设置为 2。

5 实验与分析

5．1 实验数据和指标

我们在中英机器翻译任务上验证提出的模型，分别使用了两个不同领域的篇章机器翻译语料训练模型，统计信息如表 1 所示。其中，TED 演讲数据集（TED Talks），来自于 IWSLT（International Workshop on Spoken Language Translation）国际口语机器翻译评测大赛 2014 年的评测活动[1]，数据集收录了 TED 大会中的演讲稿。我们使用 dev2010 作为开发集验证模型，tst2010-2013 作为测试集检验模型最终的效果。第二个数据集是中英字幕数据集（TVSUB）[17]，适用于研究多轮对话、篇章翻译等具有篇章上下文的任务，由影视剧的字幕汇编而成[2]。在测试集中，每一句原文都有三句对应的参考译文。

对于译文的质量评估，使用大小写不敏感的 BLEU-4[18]作为评价指标，本文采用 multi-bleu 脚本进行计算。

表 1 数据集统计信息

Data	**TED Talks**			**Subtitles**		
	|S|	**|W|**		**|S|**	**|W|**	
		Zh	**En**		**Zh**	**En**
Train	0.2M	4.2M	4.5M	2.2M	12.1M	16.6M
Development	0.8K	22.3K	21.1K	1.1K	6.7K	9.3K
Test	5.5K	0.1M	0.1M	1.2K	6.7K	9.4K

5．2 实验细节

对训练数据，在汉语端使用 ICTCLAS[3]进行了分词预处理，英语端进行了词法预处理和小写化。针对神经翻译面对的词汇数据稀疏问题，在实验中分别采用词和字节对编码[19]（Byte Pair Encoding，BPE）作为基本的翻译单元。以词为翻译单元的实验，汉语端和英文端都采用 30K 的词表大小，在 TED 演讲数据集上分别覆盖 98.6%和 99.3%的源端和目标端文本，在 TVSUB 字幕数据集上分别覆盖 98.2%和 99.4%的源端和目标端文本。

[1] https://wit3.fbk.eu
[2] 数据来源于以下两个字幕网站：http://www.zimuzu.tv, http://assrt.net 。
[3] http://ictclas.nlpir.org

以字节对编码为翻译单元的实验，BPE 单元的词汇表设置为 16K，在两个数据集上的训练数据中文本覆盖率均为 100%。

我们采用了 Transformer Base 模型的配置，词向量维度和注意力计算的隐层单元个数都是 512，前馈网络的中间层单元为 2048，多头注意力的层数为 8。为了避免过拟合，我们在模型中使用了 Dropout 方法，分别在编码器和解码器每一层的前馈网络上加入，Dropout 的比率设置为 0.1。参数的优化使用了批量的随机梯度下降方法，根据数据的长度对数据进行批处理，批处理的大小限制为 1024 个 token 以内，学习率采用自适应的 Adam 算法[19]进行调整学习（$\grave{o}=10^{-9},\beta_1=0.9,\beta_2=0.98$）。 我们使用与基线系统相同的预热（warm up）和衰减（decay）策略，设置 warm-up 的步长为 4000。

在模型测试的过程中，柱搜索的窗口大小设置为 10，为了减小候选译文的长度对序列得分的影响，设置长度惩罚（length penalty）的系数 $\alpha=0.6$ 对候选译文进行重排序，选择得分最大的作为最终输出的译文。

5. 3 主要实验结果

实验结果参照表 2，展示了 TED Talks 和 TVSUB 两个中英数据集上的 BLEU 结果。

表 2 不同数据集上的 BLEU-4 指标

Systems	**TED Talks**	**Δ**	**TVSUB**	**Δ**
Transformer	18.86		29.89	
Our method	19.66	+0.80	30.38	+0.49
Transformer + BPE	20.25		30.58	
Our method + BPE	20.72	+0.47	30.86	+0.36

对比基于联合注意力机制的篇章机器翻译模型和基线系统 Transformer 模型，可以看出本章提出的模型在 TED Talks 数据集上 BLEU 值提升了+0.8，在 TVSUB 数据集上提升了+0.49 个 BLEU。对数据分别采用字节对编码进行处理后，我们的模型在 TED Talks 数据集和 TVSUB 数据集上比 Transformer 系统分别提升了+0.47 和+0.36 个 BLEU，表明基于联合注意力机制的篇章机器翻译模型在针对不同粒度的语言表示方法上都能够提升机器翻译模型的效果。

本文提出的模型和 Transformer 模型中采用字节对编码（BPE）的方式对语料进行处理，翻译模型的性能均有提升。这说明字节对编码可以增加词汇表覆盖率，改善神经机器翻译中数据稀疏的问题。在模型中采用字节对编码处理后，本章提出的模型提升没有对原始 Transformer 模型的提升明显，表明模型融合篇章信息后提升了原始 Transformer 的表达能力，与字节对编码缓解词汇数据稀疏问题有所重合。

5. 4 与现有工作的对比

如表 3 所示，列出的实验结果为 TED 数据集上各系统的 BLEU-4 指标。从中可以看出基于 RNNSearch 的三个篇章级机器翻译系统[6][7][8]BLEU 值明显低于基于 Transformer 模型改进的篇章级翻译系统[10]。

表 3 与前人工作的实验结果对比

Systems	**Base Model**	**TED Talks**
Existing document-level NMT systems		
Hierarchical RNN (Wang 等[6])	RNNSearch	12.43
Context Encoder (Jean 等[7])	RNNSearch	12.46
Cache Model (Tu 等[8])	RNNSearch	12.68
Hierarchical Attention (Miculicich 等[10])	Transformer	17.79
Our document-level NMT systems		
Our method	Transformer	19.66

Our method+BPE	Transformer	20.72

我们的模型在 TED 数据集上输出的译文 BLEU 值最高，在引入篇章信息的机器翻译模型中超过了所有现有系统的表现，达到了最优效果。在这里，通过与这些篇章级机器翻译系统的性能对比，进一步证明了我们提出的基于联合注意力机制的篇章机器翻译模型的有效性。

5. 5 篇章长度的影响

在实验所用的 TED 和 TVSUB 数据集中，一个完整的篇章通常由上百个句子构成，将所有的句子都作为篇章信息在建模的过程中是对计算能力的要求非常高，而且在翻译当前句子的过程中考虑所有的篇章信息这一点也不符合直觉。因此，我们需要探究篇章信息的长度对实验的影响。为了探索我们的模型为了探索我们的模型对篇章信息长度的影响，我们对不同长度的篇章信息做对比实验。篇章的长度这里指加入篇章上下文句子的数量。

如表 4 所示，列出结果为 TED 数据集上不同篇章信息长度下翻译模型的 BLEU-4。可以看出，在 Transformer 模型中引入篇章信息后，翻译模型的性能均有提升。当加入篇章句子数量为 1 时，模型的性能明显低于篇章长度大的系统。随着篇章长度的增加，翻译的性能基本稳定，加入句子数量为 3 和 5 的两个系统基本没有差异。当句子数量为 7 时，系统的性能相比句子数量为 3 和 5 的系统略微有所下降。总体上，我们的模型可以在篇章信息较长的情况下依然维持比较好的性能，证明了我们的方法处理大量冗余信息的有效性。

表 4 不同篇章长度下的实验结果

Context Size	**TED Talks**
1	19.43
3	19.66
5	19.64
7	19.58

6 总结与展望

针对篇章级机器翻译中篇幅长、信息冗余的问题，本文提出了一种将硬关注和软关注两种注意力机制相结合的联合注意力对篇章信息建模，并将其应用在篇章级别的神经机器翻译模型中。在两个不同领域的公开数据集上的结果表明，基于联合注意力机制引入篇章信息的方法对机器翻译模型的翻译效果均有明显的提升。不同篇章长度下的实验结果表明，本文提出的模型在处理长篇章时依然能维持其翻译的性能，证明本文提出的联合注意力机制可以有效的处理冗余的篇章信息。

在篇章结构的数据中，远距离的语义依赖关系是广泛存在的。如何挖掘篇章中更多潜在语义信息以及语句之间的递进关系，为篇章翻译提供更加准确的语境信息是一个值得我们未来的工作中需要进一步探索的问题。

参考文献

[1] Bahdanau D , Cho K , Bengio Y . Neural Machine Translation by Jointly Learning to Align and Translate[C]//Advances in Neural Information Processing Systems. 2015.

[2] Wu Y, Schuster M, Chen Z, et al. Google’s neural machine translation system: Bridging the gap between human and machine translation[J]. CoRR, 2016, abs/1609.08144.

[3] Vaswani A , Shazeer N , Parmar N , et al. Attention is all you need[C]//Advances in Neural Information Processing Systems. 2017: 5998-6008.

[4] Sutton R S, Barto G. Reinforcement learning: An introduction[M]. MIT Press, 1998.

[5] Luong M T, Pham H, Manning C D. Effective approaches to attention-based neural machine

translation[C]//Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. 2015:1412—1421.
[6] Wang L, Tu Z, Way A, et al. Exploiting cross- sentence context for neural machine translation[C]//Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, 2017:2826—2831.
[7] Jean S, Lauly S, Firat O, et al. Does neural machine translation benefit from larger context?[J]. CoRR, 2017, abs/1704.05135.
[8] Tu Z, Liu Y, Shi S, et al. Learning to remember translation history with a continuous cache[J]. Transactions of the Association of Computational Linguistics, 2018, 6:407-420.
[9] Maruf S, Haffari G. Document context neural machine translation with memory networks [C]//Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics. 2018:1275-1284.
[10] Miculicich L, Ram D, Pappas N, et al. Document-level neural machine translation with hierarchical attention networks[J]. CoRR, 2018, abs/1809.01576.
[11] Zhang J, Luan H, Sun M, et al. Improving the transformer translation model with document-level context[J]. CoRR, 2018, abs/1810. 03581.
[12] Xiong H, He Z, Wu H, et al. Modeling coherence for discourse neural machine translation[J]. CoRR, 2018, abs/1811.05683.
[13] HE K, ZHANG X, REN S, et al. Deep residual learning for image recognition[J]. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016:770-778.
[14] Ba L J, Kiros R, Hinton G E. Layer normalization[J]. CoRR, 2016, abs/1607.06450.
[15] Sutton R, Mcallester D, Singh S, et al. Policy gradient methods for reinforcement learning with function approximation[C]// Advances in Neural Information Processing Systems. 2000.
[16] Tu Z, Liu Y, Lu Z, et al. Context gates for neural machine translation. [J]. CoRR, 2016, abs/1608.06043.
[17] Wang L, Tu Z, Shi S, et al. Translating pro-drop languages with reconstruction models[C]//Thirty-Second AAAI Conference on Artificial Intelligence. 2018.
[18] Papineni K, Roukos S, Ward T, et al. Bleu: a method for automatic evaluation of machine translation[R]. IBM Research Report, 2001.
[19] Sennrich R, Haddow B, Birch A. Neural machine translation of rare words with subword units[C]// Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics.2015: 1715–1725.
[20] Kingma D P, Ba J. Adam: A method for stochastic optimization[C]//BENGIO Y, LECUN Y. 3rd International Conference on Learning Representations, 2015, Conference Track Proceedings. 2015.

作者联系方式：李京谕　上海市浦东新区云雅路 555 弄 201201　17610870021
lvy.li.1995@outlook.com