软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software [doi: 10.13328/j.cnki.jos.006689]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

基于多策略原型生成的低资源神经机器翻译*

于志强[1,2,3], 余正涛[1,3], 黄于欣[1,3], 郭军军[1,3], 线岩团[1,3]

[1](昆明理工大学 信息工程与自动化学院, 云南 昆明 650500)
[2](云南民族大学 数学与计算机科学学院, 云南 昆明 650500)
[3](云南省人工智能重点实验室 (昆明理工大学), 云南 昆明 650500)
通信作者: 余正涛, E-mail: ztyu@hotmail.com

摘　要: 资源丰富场景下, 利用相似性翻译作为目标端原型序列, 能够有效提升神经机器翻译的性能. 然而在低资源场景下, 由于平行语料资源匮乏, 导致不能匹配得到原型序列或序列质量不佳. 针对此问题, 提出一种基于多种策略进行原型生成的方法. 首先结合利用关键词匹配和分布式表示匹配检索原型序列, 如未能获得匹配, 则利用伪原型生成方法产生可用的伪原型序列. 其次, 为有效地利用原型序列, 对传统的编码器-解码器框架进行改进. 编码端使用额外的编码器接收原型序列输入; 解码端在利用门控机制控制信息流动的同时, 使用改进的损失函数减少低质量原型序列对模型的影响. 多个数据集上的实验结果表明, 相比基线模型, 所提出的方法能够有效提升低资源场景下的机器翻译性能.

关键词: 神经机器翻译; 低资源; 多策略; 原型
中图法分类号: TP18

中文引用格式: 于志强, 余正涛, 黄于欣, 郭军军, 线岩团. 基于多策略原型生成的低资源神经机器翻译. 软件学报. http://www.jos.org.cn/1000-9825/6689.htm
英文引用格式: Yu ZQ, Yu ZT, Huang YX, Guo JJ, Xian YT. Low-resource Neural Machine Translation with Multi-strategy Prototype Generation. Ruan Jian Xue Bao/Journal of Software (in Chinese). http://www.jos.org.cn/1000-9825/6689.htm

Low-resource Neural Machine Translation with Multi-strategy Prototype Generation

YU Zhi-Qiang[1,2,3], YU Zheng-Tao[1,3], HUANG Yu-Xin[1,3], GUO Jun-Jun[1,3], XIAN Yan-Tuan[1,3]

[1](Faculty of Information Engineering and Automation, Kunming University of Science and Technology, Kunming 650500, China)
[2](School of Mathematics and Computer Science, Yunnan Minzu University, Kunming 650500, China)
[3](Key Laboratory of Artificial Intelligence in Yunnan Province (Kunming University of Science and Technology), Kunming 650500, China)

Abstract: In rich-resource scenarios, using similarity translation as the target prototype sequence can improve the performance of neural machine translation. However, in low-resource scenarios, due to the lack of parallel corpus resources, the prototype sequence cannot be matched, or the sequence quality is poor. To address this problem, this study proposes a low-resource neural machine translation approach with multi-strategy prototype generation, and the approach includes two phases. (1) Keyword matching and distributed representation matching are combined to retrieve prototype sequences, and the pseudo prototype generation approach is leveraged to generate available prototype sequences during retrieval failures. (2) The conventional encoder-decoder framework is improved for the effective employment of prototype sequences. The encoder side utilizes additional encoders to receive prototype sequences. The decoder side, while employing a gating mechanism to control information flow, adopts improved loss functions to reduce the negative impact of low-quality prototype sequences on the model. The experimental results on multiple datasets show that the proposed method can effectively improve the translation performance compared with the baseline models.

Key words: neural machine translation (NMT); low-resource; multi-strategy; prototype

---

* 基金项目: 国家重点研发计划 (2019QY1800); 国家自然科学基金 (61732005, 61672271, 61761026, 61762056, 61866020); 云南省重大科技专项 (202002AD080001); 云南省高新技术产业专项 (201606); 云南省自然科学基金 (2018FB104)
收稿时间: 2021-04-14; 修改时间: 2021-06-28, 2022-01-13; 采用时间: 2022-04-18;

1 引 言

近年来, 随着端到端翻译模型和注意力机制的提出[1,2], 神经机器翻译 (neural machine translation, NMT) 取得了长足的发展, 在主流语言对上的翻译性能迅速超过统计机器翻译[3], 逐渐发展为目前主流的机器翻译模式. 为提升神经机器翻译性能, 研究者们提出了各种方法. 其中, 基于原型序列融入的原型方法受到很多关注.

原型序列是存在于翻译记忆库中的目标端句子, 内含目标语言端语义信息. 原型方法通过在翻译进程中引入原型序列来利用目标端语义信息, 使其被隐式地用于指导词对齐和解码约束等过程. 如图 1 所示, 原型方法的主要流程为: 利用检索技术在既有的翻译记忆库中检索与当前输入相似的源端句子, 随后提取该源端句子所对应的目标端句子作为原型序列, 最后将原型序列用于数据增强或直接作为编码器输入融入翻译过程. 近年来, 原型序列检索方法在资源丰富场景下得到了较好的发展[4,5], 原因在于资源丰富场景下存在大规模的翻译记忆库, 因此原型方法可以通过检索记忆库得到较高质量的原型序列, 进而有效地提升翻译性能. 然而在低资源场景下, 受限于平行语料的规模和质量, 传统的原型序列检索方法往往难以检索得到可用的原型, 对下一步翻译任务的效果提升有限. 除此以外, 在对原型序列利用方面, 尤其是将原型序列作为编码输入融入翻译模型的方式上, 研究者们提出了很多改进, 文献 [6] 采用双编码器结构对输入句子和原型序列同时进行编码, 同时在解码端引入门控机制来平衡源句和原型序列间的信息比例. 文献 [7] 针对原型方法中效率和信息利用率不高的问题, 提出了软原型模型, 该模型通过直接建立源语言单词和目标语言单词的直接映射来生成原型序列. 然而, 以上工作重点关注编码过程中的原型序列表示及解码过程中的信息流动控制, 对低资源场景下的解码过程中, 如何在最大化利用优质原型序列的同时减少低质量原型序列对翻译过程的负面影响则讨论较少. 整体而言, 原型序列检索和融入成为低资源场景下应用原型方法的瓶颈, 是值得深入研究的问题.

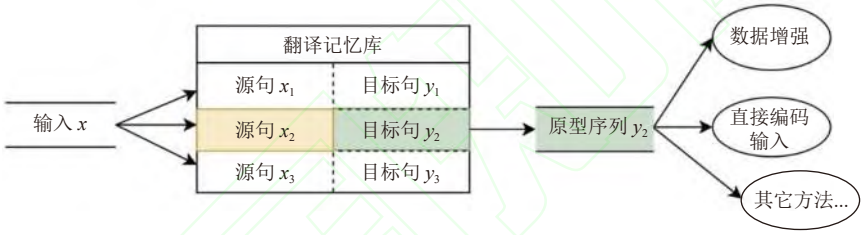

图 1 原型序列检索与利用示例

针对以上问题, 本文提出一种基于多策略原型生成的低资源神经机器翻译方法, 通过改进的原型获取方法和特定的翻译框架结构, 更好地提升低资源神经机器翻译的性能. 本文的主要贡献有如下 3 方面.

(1) 提出一种基于多种策略混合的原型生成方法, 通过联合使用原型序列检索方法和基于词替换的伪原型生成方法来获取原型序列, 从而在保证序列质量的前提下, 最大化提升低资源场景下可用的原型序列数量.

(2) 对编码器-解码器翻译框架进行了改进, 编码端使用额外的编码器接收原型序列输入, 解码端使用门控机制控制信息的比例和流动过程. 此外, 改进损失计算方法, 使模型在尽力利用高质量原型序列所含语义信息的同时, 削弱低质量原型序列对翻译模型的负面影响.

(3) 在低资源场景下的多个翻译数据集上验证了本文方法. 与基线系统和同类原型方法相比, 本文模型在低资源场景下具有更优异的性能, 同时能带来更好的翻译流畅度.

2 相关工作

目前原型方法领域的研究工作主要集中在原型匹配和原型利用两个阶段. 原型匹配主要通过句子相似性来检索相似句子, 代表性的匹配方法有关键词匹配 (lexical matching) 和分布式表示匹配 (distributed representations matching). 原型利用方面, 目前的主流方法主要通过数据增强和直接编码输入的方式使用原型序列.

2.1 原型匹配

关键词匹配是基于词汇的匹配方式, 目标是识别出与给定句子相似的句子, 主要实现方法有模糊匹配和 N-gram 匹配. 模糊匹配主要通过编辑距离进行句子间相似度的计算和度量, N-gram 匹配则是通过最长公共子序列进行相似度计算. 基于关键词匹配的代表性工作中, 文献 [4] 利用 lucene 工具首先检索得到候选源句, 随后利用编辑距离计算输入句子与候选源句的相似度. 在编辑距离计算中, 将无需编辑的词或短语存储为翻译碎片, 作为准确翻译片段指导进一步的翻译过程. 文献 [5] 则针对编辑距离计算时间复杂度较高的问题, 提出了先利用 SetSimilaritySearch 检索得到 top-k 个相似性匹配, 后利用编辑距离进行相似度排序的方法. 此外, 为综合利用模糊匹配和 N-gram 匹配的优势, 文献 [8] 提出一种相关词检测算法, 该算法能够同时被应用到模糊匹配及 N-gram 匹配方法中. 针对编辑距离计算中各单词权重总是平均分配的问题, 一些研究工作[9]在计算相似度之前, 通过句子规范化方法进行预处理, 降低停用词和标点符号等的权重. 也有研究工作[10]利用 IDF 和词干提取技术增加重要单词的权重, 从而减少形态变体的权重.

随着基于神经网络计算的分布式表征的出现, 句子相似度度量的研究取得了巨大进展. 特别是句子向量化技术[11]和预训练技术[12]的出现, 使得句子的表征和检索更为精确和高效. 其中, 文献 [13] 构建检索器, 通过比较句子表征计算句子间的相似性, 为进一步的编辑操作提供基础. 与其不同的是, 文献 [5] 利用向量化检索工具进行相似性检索, 有效地解决了编辑距离计算中时间复杂度较高的问题. 基于文献 [5] 的工作, 文献 [8] 提出了相关词检测算法并将其融入模糊匹配及 N-gram 匹配方法中, 随后结合分布式表示匹配方法进行综合检索, 有效地提升了检索到的原型序列质量. 此外, 文献 [14] 在无监督场景下, 使用 FastText 和 Vecmap 进行无监督的跨语言词嵌入, 通过 SIF 获得基于该词嵌入的跨语言句子嵌入, 进而利用 Marginal-based 评分从两个语料库中检索相似的句子. 其他自然语言处理任务中, 文献 [15] 针对文本生成任务, 将稀疏诱导先验引入原型选择分布: 检索得到的原型序列、原始输入和编辑操作被共同编码为分布式表征后输入到推断网络进行预测, 以期得到语义相似的文本输出.

以上方法有效地提升了原型匹配的效率和准确性, 但较少考虑多种匹配方法的结合使用. 已存在的综合使用方法[8]也未对低资源环境下未能检索到原型序列的情况做进一步处理.

2.2 原型利用

目前, 原型方法主要通过数据增强和直接编码输入的方式使用原型序列. 基于数据增强的方法通过将检索得到的原型序列加入原始输入语料中以实现语料扩充, 多以拼接源句的方式进行. 受文献 [16,17] 所提出的源句与译文拼接和多种源语言间拼接的思想启发, 文献 [5] 在获取到原型序列后, 通过 3 种拼接策略将其拼接到输入的源句之后, 进而与原始输入的目标句组合成新语料后投入训练, 随后验证了新语料与原始语料组合或消融后对翻译性能的影响. 文献 [8] 则提供了 FM#等 5 种不同的拼接方式, 拼接的同时通过相关词标签对基于关键词匹配得到的原型序列进行标记, 因此不仅实现了语料的扩充, 同时也考虑了词共现信息对模型的影响. 文献 [18,19] 则针对只有双语词典的低资源场景, 将目标端单词视为短原型序列, 以拼接到源端单词之后或直接替换的方式扩展语料, 同时对每个单词的来源予以标记, 该方法能够提升低资源场景下的语料规模和翻译性能.

除数据增强外, 很多研究工作通过直接编码输入的方式使用原型序列. 具体而言, 通过改变编码器-解码器框架, 增加额外的原型编码器对原型序列进行编码, 得到序列表征后与输入句子表征一起投入解码过程. 文献 [6] 将原型序列作为额外信号, 利用新的编码器对它进行编码, 进而用来指导解码过程. 同时, 通过引入门控机制来平衡源句和原型序列间的信息比例. 针对原型方法中效率和信息利用率不高的问题, 文献 [7] 提出了软原型模型. 模型首先训练得到基础翻译模型, 据此建立源语言单词和目标语言单词的概率关系. 与文献 [6] 不同的是, 软原型序列通过源与目标间的直接映射得到, 随后被输入到额外的原型编码器中进行编码. 除此以外, 有研究工作通过多次编解码机制[20−22]将第 1 次产生的译文作为原型序列引入二次编解码过程. 也有研究工作通过结合词典或句法结构[23]形成富信息原型序列, 作为额外的编码器输入.

以上方法均带来了翻译性能上的提升, 但是仍然主要面向资源丰富场景, 较少针对低资源场景进行特定的改进. 为此, 本文提出一种基于多策略原型生成的低资源神经机器翻译方法, 通过多种策略混合方式进行原型序列生

成, 同时改进翻译模型结构, 使模型侧重学习较高质量原型序列中的语义信息, 减少低质量原型序列对模型的影响.

3 基于多种策略混合的原型生成方法

为保证原型序列的可用性, 本文利用基于多种策略混合的原型生成方法进行原型生成. 方法的具体思路为: 首先结合使用模糊匹配和分布式表示匹配进行原型检索, 如未检索到原型, 则利用词替换操作对输入句子中的关键词进行替换, 得到伪原型序列. 原型生成的整体流程如图 2 所示.

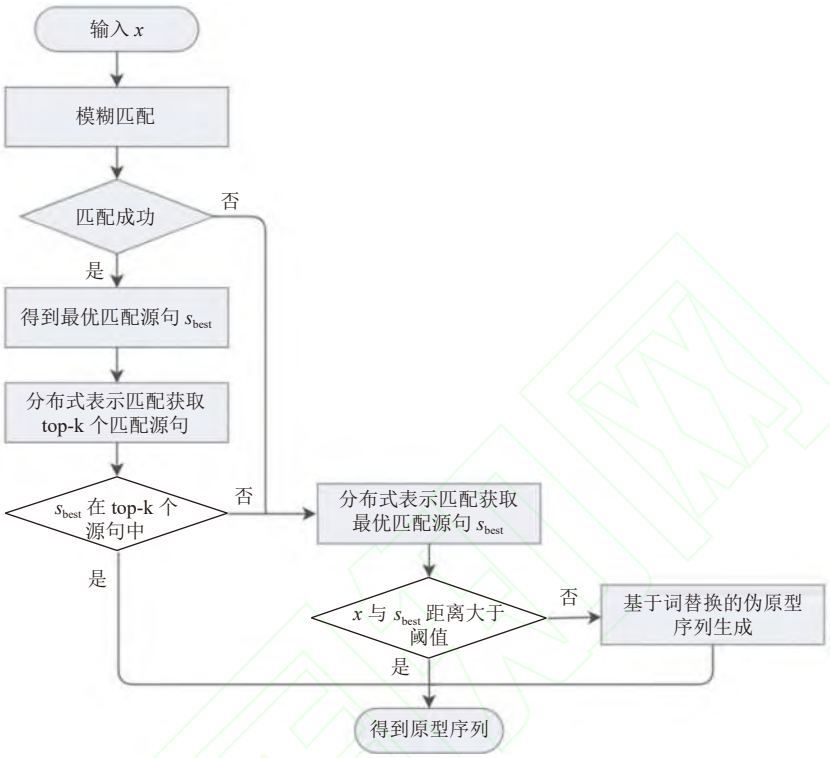

图 2 原型生成流程图

3.1 结合分布式表示匹配的混合式原型匹配

翻译记忆库是由 L 对平行句组成的集合 $\{(s_l,t_l):l=1,\ldots,L\}$, 其中 s_l 为源句, t_l 为目标句. 对给定的输入句子 x, 首先使用关键词匹配于翻译记忆库中进行检索. 低资源环境下, 由于语料中的长句数量较少, 易导致匹配到的片段长度较短, 难以形成有效的相似度度量. 因此本文未采用 N-gram 匹配, 而是采用模糊匹配作为关键词匹配方法, 其定义为:

$$FM(x,s_i)=1-\frac{ED(x,s_i)}{\max(|x|,|s_i|)} \tag{1}$$

其中, $ED(x,s_i)$ 是 x,s_i 间的编辑距离, $|x|$ 为 x 的句长.

与基于关键词的匹配方法不同, 分布式表示匹配根据句子向量表征之间的距离进行检索, 某种程度上是利用语义信息进行相似性检索的手段, 也因此提供了与关键词匹配不同的检索视角. 基于余弦相似度的分布式表示匹配定义为:

$$EM(x,s_i)=\frac{h_x\times h_{s_i}}{\|h_x\|\times\left\|h_{s_i}\right\|} \tag{2}$$

其中, h_x 和 h_{s_i} 分别为 x 和 s_i 的向量表征, $\|h_x\|$ 为向量 h_x 的度量. 为实现快速计算, 本文首先使用多语言预训练模

型 mBERT (multilingual-BERT) 得到句子 x 和 s_i 的向量表征, 随后依据表征, 使用 Faiss 工具 (https://github.com/facebookresearch/faiss) 进行相似性匹配.

本文结合利用两种方法的优势, 如图 2 所示, 当模糊匹配能够得到最优匹配源句 s_{best} 时, 利用分布式表示匹配得到 top-k 个匹配结果的集合 $s' = \{s_1, s_2, \ldots, s_k\}$, 如 $s_{\text{best}} \in s'$, 则选取 s_{best} 对应的目标端句子 t_{best} 作为原型序列. 当模糊匹配未能检索到匹配源句或 $s_{\text{best}} \in s'$ 时, 则通过分布式表示匹配检索出最优匹配源句 s_{best}.

3.2 基于词替换的伪原型生成

本节针对不同数据规模场景下匹配方法的效果进行了对比. 常规来说, 翻译领域将高质量平行句对数达到百万的翻译场景视为资源丰富场景. 因此, 我们采用百万句对级别的英-荷数据集 (European commission (en-nl), 2.4M)[5]和英-德数据集 (WMT14 (en-de), 4.5M) 作为资源丰富语言对, 采用 10 万句对级别的英-越数据集 (IWSLT15 (en-vi), 133k) 和英-德数据集 (IWSLT15 (en-de), 172k) 作为低资源语言对展开分析, 其中 IWSLT15 (en-vi) 数据集的分析结果来源于本文实验. 如表 1 所示, 资源丰富场景下仅基于模糊匹配即可得到大规模的高质量原型序列. 然而在低资源场景下, 模糊匹配检索得到原型序列数量明显不足. 使用结合了模糊匹配和分布式表示匹配的混合式原型匹配后, 可在一定程度上缓解该问题, 然而如表 2 所呈现的, 大规模数据集 WMT14 上, 相比单独使用模糊匹配策略, 结合使用分布式表示匹配能够得到较为理想的匹配结果. 而对于小规模数据集 IWSLT15 而言, 结合模糊匹配和分布式表示匹配后, 能够在关键词匹配基础上附加语义层面的综合考量, 进一步提升原型序列质量. 然而, 为了保证检索到的序列质量, 需对分布式表示匹配设定阈值, 根据输入句子 x 与匹配源句 s_{best} 间的相似性距离对结果进行筛选, 得到符合质量要求的原型序列. 筛选操作会减少匹配结果的数量, 相比资源丰富场景, 该操作对低资源场景的影响更甚, 也因此造成了低资源场景下匹配到的高质量原型序列数量依然远小于资源丰富场景的情况. 因此, 如需在低资源环境下较好地应用原型方法, 合理的原型序列生成机制是必要的. 受文献 [24] 的思想启发, 本文针对此问题提出了基于词替换的伪原型生成方法, 含以下两种替换策略.

表 1 不同数据规模下的模糊匹配效果 (%)

场景	数据规模	模糊匹配区间					
		<50	50–59	60–69	70–79	80–89	90–99
资源丰富	2.4M	41.3	11.4	10.3	8.8	14.2	14.0
低资源	133k	84.7	5.5	4.1	3.7	1.1	0.9

注: 资源丰富场景数据集: European commission (en-nl); 低资源场景数据集: IWSLT15 (en-vi)

表 2 模糊匹配与混合式原型检索效果对比 (%)

场景	匹配策略	匹配区间					
		<50	50–59	60–69	70–79	80–89	90–99
资源丰富	模糊匹配	40.1	12.1	13.5	10.7	12.8	10.8
	模糊匹配+分布式表示匹配 (EM > 0.5)	33.2	13.5	14.7	14	13.4	11.2
低资源	模糊匹配	83.7	4.9	5.2	4.0	1.4	0.8
	模糊匹配+分布式表示匹配 (EM > 0.5)	78.9	6.0	6.5	4.9	2.2	1.5

注: 资源丰富场景数据集: WMT14 (en-de), http://workshop2017.iwslt.org/; 低资源场景数据集: IWSLT15 (en-de), http://statmt.org/wmt14/translation-task.html

- 全局替换: 当输入句子未能检索到匹配时, 基于最大化原则, 利用双语词典对输入句子中的词进行尽力替换, 替换后的句子被称为伪原型序列. 我们期望由目标端词汇构成的伪原型序列能够为编解码过程提供额外指导信息.
- 关键词替换: 从双语词典中抽取重要名词和实体构建关键词词典. 当输入句子未能检索到匹配时, 利用该词典对输入句子中的关键词进行替换, 生成伪原型序列, 替换次数上限小于设定的阈值. 我们期望在共享词表的基础上, 该混合了源端和重要目标端词汇的伪原型序列能够为译文的生成提供指导.

图 3 展示了利用全局替换进行伪原型生成的过程. 给定的英语输入句子“Special education is a noble work.”, 经

查找双语词典后被直接替换为越南语句子“Học hành đặc biệt là cao quý một công việc.”, 随后, 源句和替换后的句子分别作为句子编码器和原型编码器的输入进行编码. 与全局替换不同的是, 关键词替换通过查找关键词词典将输入句子替换为双语混合句子, 随后将两个句子分别输入到句子编码器和原型编码器. 两种方式对性能的不同影响在实验章节中说明.

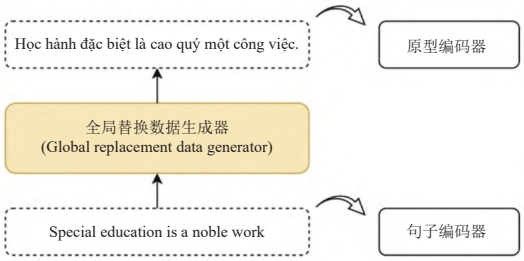

图 3 词替换示例 (英语→越南语)

4 基于原型序列融入的翻译模型

本文期望通过层次化组合使用原型匹配和原型生成方法来提升检索到的原型序列的数量与质量, 然而在原型利用方面, 传统的原型方法在神经网络架构上并未针对性地考虑低资源环境下的适用情况. 针对此问题, 我们对编解码器结构进行了改进并于本节展开介绍. 首先回顾传统基于注意力机制的标准神经机器翻译模型: 给定源语言输入句子 $x=\{x_1,x_2,\ldots,x_m\}$, 传统模型被定义为:

$$P(y|x;\theta)=\prod_{i=1}^{N}P(y_i|y_{<i};x;\theta) \tag{3}$$

其中, θ 为模型参数, $y_{<i}$ 为已经得到的译文. 融入原型序列 $t=\{t_1,t_2,\ldots,t_n\}$ 后, 新的翻译模型表述为:

$$P(y|x;t;\theta)=\prod_{i=1}^{N}P(y_i|y_{<i};x;t;\theta) \tag{4}$$

为了更好地融入原型序列, 本文模型的整体结构设计如下: 编码端采用双编码器结构, 能够同时接收句子输入和原型序列输入, 然后将输入编码为相应的隐状态表示; 解码端融入门控机制, 利用神经网络自学习能力实现句子信息和原型信息间的比例优化, 控制解码过程中的信息流动. 为减少过低质量原型序列对解码过程的影响, 对损失函数进行了优化. 模型整体结构如后文图 4 所示. 句子编码器和原型编码器分别对输入的句子和检索到 (生成) 的原型进行编码. 解码端基于编码器隐状态进行门控操作.

4.1 编码端

为能同时处理句子输入和原型序列输入, 编码端采用双编码器结构. 句子编码器为标准的 Transformer 编码器, 由多层堆叠而成, 其中每层又由 2 个子层构成: 多头自注意力层和前馈神经网络层, 均使用残差连接和层正则化机制. 给定输入句子 $x=(x_1,x_2,\ldots,x_m)$, 句子编码器将其编码为相应的隐状态序列 $h_x=(h_{x1},h_{x2},\ldots,h_{xm})$, 其中 h_{xi} 为 x_i 对应的隐状态. 原型编码器在神经网络结构上与句子编码器一致. 给定原型序列 $t=(t_1,t_2,\ldots,t_n)$, 原型编码器将其编码为相应的隐状态序列 $h_t=(h_{t1},h_{t2},\ldots,h_{tm})$, 其中 h_{ti} 为 t_i 对应的隐状态.

4.2 解码端

解码端通过融入门控机制对编解码注意力层进行了改进, 使其能平衡句子信息和原型信息的比例, 控制解码过程中的信息流动. 改进后的解码器由 3 个子层构成: (1) 自注意力层; (2) 改进的编解码注意力层; (3) 全连接前馈网络层. 其中, 改进的编解码注意力层由句子编解码注意力模块和原型编解码注意力模块构成. 如图 4 所示, 接收到 i 时刻多头自注意力层的输出 s_{self} 和句子编码器的输出 h_x 时, 句子编解码注意力模块进行注意力计算:

$$s_x = MultiheadAtt(s_{\text{self}}, h_x, h_x) \tag{5}$$

其中, $MultiheadAtt(\cdot)$为基于多头的注意力计算. 与此类似, 原型编解码注意力的计算为:

$$s_t = MultiheadAtt(s_{\text{self}}, h_t, h_t) \tag{6}$$

随后, 句子编解码注意力输出 s_x 和原型编解码注意力输出 s_t 被连接, 用于计算比例变量 α:

$$\alpha = Sigmoid(W_\alpha [s_x; s_t] + b_\alpha) \tag{7}$$

其中, W_α 和 b_α 为可训练参数. α 随后被用于计算编解码注意力层的最终输出, 计算公式为:

$$s_{\text{enc\_dec}} = \alpha \times s_x + (1-\alpha) \times s_t \tag{8}$$

如图 4 右上部分所示, $s_{\text{enc\_dec}}$ 作为输入被填充到全连接前馈网络中:

$$s_{\text{ffn}} = f\left(s_{\text{enc\_dec}}\right) \tag{9}$$

其中, $f(x)$ 的定义为: $f(x) = \max(0, xW_1 + b_1)W_2 + b_2$, 其中 W_1, W_2, b_1 和 b_2 均为参数. 最终 i 时刻的译文 y_i 计算如下:

$$P(y_i | y_{<i}; x; t, \theta) = Softmax(\sigma(s_{\text{ffn}})) \tag{10}$$

其中, t 为原型序列, $\sigma(\cdot)$为线性变换函数.

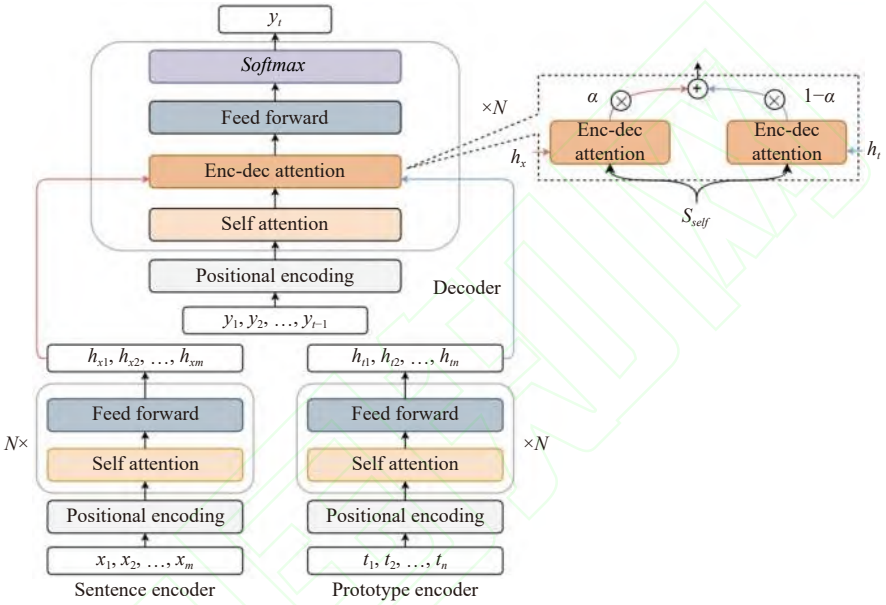

图 4 翻译模型整体结构

4.3 损失计算

给定训练语料集合 D, 传统基于注意力机制的神经机器翻译模型使用最大似然估计 (MLE) 损失函数来更新模型参数:

$$L(\theta') = \sum_{x,y \in D} -\log P(y|x;\theta') \tag{11}$$

其中, θ' 为编码器和解码器的参数. 在通过原型编码器将原型序列 t 引入模型后, 损失计算则变化为:

$$L(\theta'') = \sum_{x,y \in D} -\log P(y|x;t;\theta'') \tag{12}$$

其中, θ'' 为句子编码器、原型编码器和解码器的参数. 传统基于最大似然估计的损失函数促使译文接近真实分布, 引入原型序列的损失函数则考虑了原型序列对译文语义的指导作用. 综合考虑 2 种损失函数特点, 同时受文献 [23] 等工作启发, 我们在训练过程中将两种损失进行了结合, 以比例 β 应用损失函数 $L(\theta')$, 以剩余比例 $(1-\beta)$ 使用损失函数 $L(\theta'')$. 最终的损失函数 $L(\theta)$ 为:

$$L(\theta) = \beta L(\theta') + (1-\beta) L(\theta'') \tag{13}$$

其中, β 为比例参数. 实际应用中, 我们发现通过合理的 β 值设定可控制原型编码器所引入的信息量, 利用神经网络的自学习能力, 使模型更关注于高质量原型序列, 减少低质量的原型序列对训练过程的影响, 提升在低资源环境下的性能. β 比例参数对模型性能的影响于第 5 节中进行了说明.

5 实验与分析

5.1 数据集

本文使用机器翻译领域的通用数据集 IWSLT15 进行模型训练, 翻译任务为英 (en)-越 (vi)、英 (en)-中 (zh) 和英 (en)-德 (de). 表 3 显示了数据集的构成情况. 验证和测试方面, 选择 tst2012 作为验证集进行参数优化和模型选择, 选择 tst2013 作为测试集进行测试评估. 实验使用的全局替换词典源自 PanLex (https://panlex.org/)、维基百科及实验室自建的英汉-东南亚语词典, 其中实体部分主要来源于维基百科和自建词典并附带实体标记. 通过 PanLex 及谷歌翻译接口自动实现了德语词汇与其他语种的对齐. 最终的英 (en)-越 (vi)-中 (zh)-德 (de) 词典包含 61k 词对. 关键词词典通过标记筛选方式自全局词典得到, 筛选过程中保留全部实体. 为避免替换过于集中于某些热点名词, 对名词性词汇于语料中检索并按出现频率进行倒排, 最终形成的关键词词典规模为 7.3k.

表 3 实验数据集

语料类型	数据集	语言对	训练集	验证集	测试集
小规模平行语料	IWSLT15	en↔vi	133k	1 553	1 268
	IWSLT15	en↔zh	209k	887	1 261
	IWSLT15	en↔de	172k	887	1 565
大规模平行语料	WMT14	en↔de	4.5M	3 003	3 000

5.2 参数设置

本文采用 Transformer 模型作为实现平台, 针对低资源场景进行神经网络参数设定. 我们参考了 Sennrich 等人[25]关于低资源环境下优化神经机器翻译效果的推荐设置, 包括层正则化和较为激进的 dropout. 编码器和解码器分别采用默认的 2 层神经网络, 注意力头数设置为 4. 隐状态维度、词嵌入维度及批次大小设置为 256, 每 1 000 次迭代后即进行模型评估. 使用 Adam 优化器进行模型参数优化, dropout 和标签平滑设定为 0.1, 初始学习率设置为 10^{-3}. 测试阶段我们使用集束搜索算法进行解码, 集束宽度为 4, 长度惩罚设置为 1.

5.3 评价指标

选择大小写不敏感的 4-gram BLEU 值[26]作为译文质量评价指标, 评价脚本采用大小写不敏感的 multi-bleu.perl. 为了从更多角度评价译文质量, 另外采用 RIBES (rank-based intuitive bilingual evaluation score) [27]进行辅助评测. 与 BLEU 评测不同, RIBES 评测方法侧重于关注译文的词序是否正确, 适用于低资源环境下的译文流畅度评判. 对最优的 BLEU 值结果, 均利用 bootstrap 重采样方法[28]进行了显著性测试.

5.4 匹配策略

如第 3.1 节所示, 对于模糊匹配, 利用 $FM(x, s_i)$ 进行相似性计算后即可得到最优的模糊匹配序列, 而分布式表示匹配需首先使用多语言预训练模型 mBERT 得到句子的向量表征. 然而, 虽然 mBERT 基于多种语言训练得到, 但在训练过程中没有显式的跨语言对齐信号. 因此, 为了更加适配本文所提的基于替换的伪原型生成方法, 对齐源语言与目标语言词语, 本文使用同样基于替换机制的 code-switching 方法[24]对 mBERT 进行了微调, 使多种语言自动对齐到统一的向量空间中. 分布式表示匹配 $EM(x, s_i)$ 基于微调后的 mBERT 得到句子向量表征, 随后使用 Faiss 工具以 IndexFlatL2 方式构建索引、进行相似度计算. 评估最优模糊匹配序列 s_{best} 是否属于分布式表示匹配得到的 top-k 个结果集合 $s' = \{s_1, s_2, \ldots, s_k\}$ 时, k 设置为 3. 检索最优分布式表示匹配序列时, 阈值设置为 $EM(x, s_i)$>0.5.

在利用关键词替换策略生成伪原型时, 以顺序次序进行词典检索. 同时为避免退化为全局替换, 依据实验结果设置替换操作上限为 3 次.

5.5 基线模型

为了验证本文模型的性能, 将其与一些基线进行了比较.

● GNMT: 传统的 RNN 模型, 选择双向 LSTM 作为编解码器单元. 参考低资源翻译场景[25]进行模型参数设置. 其中神经网络层数设置为 2, 隐状态和词嵌入维度设置为 256. 其余参数参考默认设置 (https://github.com/NVIDIA/DeepLearningExamples/tree/master/PyTorch/Translation/GNMT).

● Transformer[29]: 基于自注意力机制的基础翻译模型. 如第 5.2 节所述, 采用针对低资源场景下的谨慎神经网络参数设定. 本文所提方法基于基础 Transformer 模型实现.

● Term-constraint[18]: 基于外部词典的数据增强方法, 该方法提供了 append 和 replace 两种策略用于产生新数据. Append 策略是将词典中的目标词追加到相应的源端单词之后. Replace 策略是将源句中的词语替换为词汇表中相应的目标词. 我们将本文方法与该工作中性能较优的 append 策略进行了比较.

● NMT-GTM[6]: 一种高效的原型模型. 编码端利用双编码器编码输入句子和原型序列, 解码端引入门控机制平衡输入句子和原型序列间的信息比例. 模型采用 GRU 为神经网络单元, 本文实验中, 参数设置参考 GNMT 模型进行.

● SoftPrototype[7]: 一种高效的原型神经机器翻译模型. 首先训练基础翻译模型, 通过该模型实现源与目标间的直接映射、得到软原型序列. 随后在编码端利用额外的原型编码器对原型序列进行编码, 解码端利用门控机制控制信息流动.

5.6 实验结果

5.6.1 损失函数超参对翻译性能的影响

本文于第 4.3 节中对传统的损失函数进行了改进, 通过引入比例参数 β 来自动调整原始 MLE 损失和原型序列损失的比例. 为了观察所引入的比例参数 β 的不同取值对翻译模型性能产生的影响, 我们设定 β={10%, 20%, 30%, 40%, 50%, 60%, 70%, 80%, 90%}, 在不同的验证集上展开评估. 评估结果表明, 当 β 位于 0.5 至 0.8 区间时, 模型整体表现较好并均于 β=0.6 时达到最优性能. 随后, 我们使用测试集于 3 个翻译任务上进行了验证. 验证结果与评估结果保持一致. 后文图 5 显示了 β=0.6 时模型在越 (vi)→英 (en) 测试集上的性能曲线.

5.6.2 关键词替换阈值对翻译性能的影响

利用关键词替换进行伪原型生成过程中, 我们首先依据经验设置替换阈值, 随后基于验证集性能对阈值进行调整. 评估结果表明, 在顺序遍历词典策略下, 当替换次数阈值设置较小时, 生成的原型序列与原文区别有限, 难以为翻译过程提供有效的指导信息; 而当设置较大时, 则趋向退化为全局替换; 当关键词替换次数设置为 3 时模型表现最优. 测试集上的结果显示了同样的变化趋势, 如图 6 越 (vi)→英 (en) 翻译结果所示.

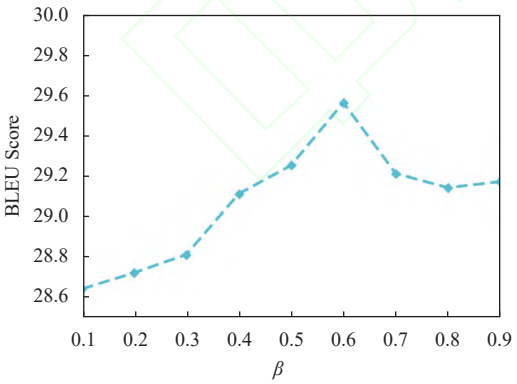

图 5 参数 β 对模型性能的影响

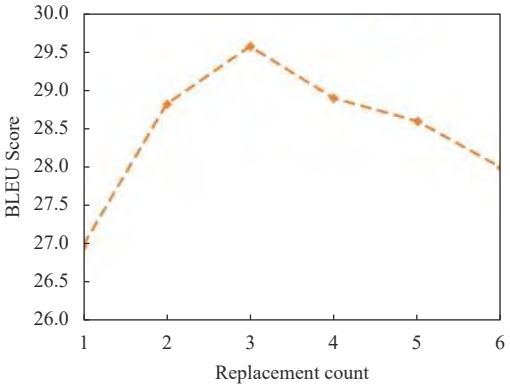

图 6 关键词替换次数对模型性能的影响

5.6.3 BLEU 值评测

首先比较了不同基线系统的性能, 如后文表 4 上半部分所示. 其次为了观察不同的替换策略及改进的损失计算对翻译效果的影响, 我们进行了消融试验并在表 4 中报告了实验结果, 其中默认设置及方法缩写如下.

● 默认: 不进行伪原型生成, 只使用模糊匹配和分布式表示匹配得到原型序列. 使用原型损失函数 (第 4.3 节公式 (12)) 计算损失.

● AR: 利用全局词替换生成伪原型序列, 通过双语词典对输入句子中的单词进行匹配和替换. 未采用最大化匹配原则, 即词的不同形态视为不匹配. 如超出词典范围未能匹配成功, 则保持原词不变.

● KR: 利用关键词替换生成伪原型序列, 通过关键词词典对输入句子中的单词进行最多 3 次匹配和替换.

● L: 使用改进后的损失函数 (第 4.3 节公式 (13)) , 根据第 5.6.1 节的实验结果, 选取了最优的超参设定 β=0.6.

依据表 4 结果可以观察到, 相比未使用原型序列的 Transformer 基线系统, 基于传统检索策略的原型方法在低资源环境下性能提升有限. 以本文默认模型为例, 其在越 (vi)→英 (en) 翻译任务上仅获得 0.4 个 BLEU 值的提升, 英 (en)→越 (vi) 任务上的提升则更不明显. 我们认为造成此现象的一个主要原因是: 虽然利用原型序列所含语义信息对于指导译文生成有促进作用, 但在低资源环境下, 检索到的可用原型序列比例过低, 不能提供足够的语义指导. 除此以外, 消融实验结果表明, 使用 AR 后, 本文模型在所有翻译任务均取得了一定的性能提升, 但幅度较小. 可能原因是全局替换虽产生了更接近目标语言的句子, 但过多的词替换操作也同时放大了一词多义等噪声的影响. 与之相比, 在限定替换上限的情况下, 基于关键词的替换 KR 的效果更佳. 此外, 两种伪原型生成策略在结合使用改进后的损失函数后 (AR+L 和 KR+L) , 性能均得以进一步提升.

5.6.4 RIBES 值评测

依据 RIBES 对本文所提出的方法进行了评测, 结果如表 5 所示. 可以看出相比基线模型 GNMT 和 Transformer 本文方法均获得了显著的提升, 与原型模型 NMT-GTM 和数据增强模型 Term-constraint 相比也取得了改善. 在所有翻译任务上, 本文方法相比 SoftPrototype 均获得了 RIBES 值的提升. 上述结果说明本文方法能够较好地利用原型序列提供的指导信息, 通过对真实句子 (原型序列) 的学习, 有效提升译文的流畅度.

表 4 BLEU 值评测结果 (%)

模型	翻译方向					
	en→vi	vi→en	en→zh	zh→en	en→de	de→en
GNMT	27.34	25.13	22.37	20.29	26.18	29.67
Transformer	30.21	28.20	25.64	23.32	30.55	34.07
Term-constraint	30.68	28.72	25.76	23.49	31.58	34.73
NMT-GTM	30.25	28.44	25.79	23.45	30.71	34.02
SoftPrototype	30.70	28.79	26.40	23.91	31.37	34.45
本文模型	30.54	28.60	26.46	23.98	31.34	34.49
本文模型 (+AR)	30.89	29.03	26.85	24.27	31.95	34.93
本文模型 (+KR)	31.32	29.38	27.07	24.64	32.24	35.41
本文模型 (+AR+L)	31.07	29.35	27.13	24.67	32.18	35.25
本文模型 (+KR+L)	**31.61**	**29.56**	**27.32**	**24.88**	**32.53**	**35.67**

注: 结果通过了显著性检验(bootstrap重采样, $p < 0.05$)

表 5 RIBES 值评测结果 (%)

模型	翻译方向					
	en→vi	vi→en	en→zh	zh→en	en→de	de→en
GNMT	71.82	71.59	70.37	69.39	76.56	76.12
Transformer	73.15	73.00	71.44	70.24	79.03	78.29
Term-constraint	74.90	74.64	72.63	71.67	80.39	79.13
NMT-GTM	74.36	74.28	72.85	71.80	79.78	78.45
SoftPrototype	74.95	74.59	73.25	72.31	80.02	78.77
本文模型 (KR+L)	**75.71**	**75.46**	**73.75**	**72.47**	**80.88**	**79.53**

5.6.5 预训练模型微调对模型的影响

本文使用基于替换机制的 code-switching 方法对多语言预训练模型 mBERT 进行了微调, 将多种语言自动对齐到统一的向量空间. 后文表 6 的结果表明, 相比原始 mBERT, 在基于经过微调的 mBERT 上验证本文方法, 获得了平均 0.51 个 BLEU 值的提升. 出现此情况的可能原因是, 微调使互为替换的词对具有相似的向量表征, 进而在分布式表示匹配过程中获取了更接近原文的表示、提升了原型序列的质量.

5.6.6 语言相似性对伪原型生成的影响

第 5.6.5 节中, 预训练模型微调使互为替换的词对具有相似的向量表征, 进而对原型生成过程产生积极影响.

那么具有天然语言相似性的语言对是否更加适用于本文方法? 针对此问题, 本文选择 ALT 数据集 (http://www2.nict.go.jp/astrec-att/member/mutiyama/ALT/) 的英语 (en)-中文 (zh)-泰语 (th)-老挝语 (lo) 子集部分进行了试验验证. ALT 数据集是由 20k 组平行句构成的小规模多语平行数据集, 其特点为语料均经过人工调整、质量较高. 选取的子集中, 泰语-老挝语为同属于壮侗语系的高度相似性语言[30,31], 英语和中文则与泰老两种语言的差异性较大. 我们使用数据集内置切分工具 ALT-Standard-Split 对子集进行了切分, 切分后的训练集、验证集、测试集分别由 18 088、1 000、1 018 组三语平行句子构成. 采用 4-gram BLEU 值进行译文质量评估.

表 6 微调对模型的影响

预训练模型	翻译方向						平均BLEU
	en→vi	vi→en	en→zh	zh→en	en→de	de→en	
mBERT	31.17	29.05	26.70	24.24	32.12	35.25	29.75
code-switching+mBERT	**31.61**	**29.56**	**27.32**	**24.88**	**32.53**	**35.67**	**30.26**

从表 7 可以观察到, 在英 (en)/中 (zh)→泰 (th)/老 (lo) 4 个翻译任务中, 英→泰任务得到最大的 1.18 个 BLEU 值的提升. 与之相比, 泰 (th)→老 (lo) 任务得到了更为显著的 1.89 个 BLEU 值的提升. 可能原因是对于相似性语言, 词的向量表征更为接近, 同时句法结构的相似性使替换后的句子更接近真实结构. 依据结果我们认为, 本文方法在低资源环境下的相似性语言翻译任务上更具优势.

表 7 语言相似性对伪原型生成的影响

模型	翻译方向				
	en→th	en→lo	zh→th	zh→lo	th→lo
Transformer	7.53	7.38	7.25	7.11	7.27
本文模型	8.71	8.44	8.33	8.2	**9.16**

6 结 论

本文针对目前主流的原型方法在应用于低资源环境时, 存在检索到的原型序列数量不足及低质量原型序列带来的噪声引入问题, 提出了一种基于多策略原型生成的低资源神经机器翻译方法. 通过结合传统检索方法和所提出的伪原型生成方法提升原型序列获取的效率和质量, 同时通过改进的翻译模型结构, 将检索到的原型序列融入传统编解码器框架, 在最大化利用原型序列所含语义信息的同时削弱低质量序列带来的影响. 相比其他原型方法, 在原型序列检索方面, 本文方法能够基于少量平行语料有效地提升检索到的原型序列的数量和质量; 在信息利用方面, 本文方法能够在编码端使用额外的原型编码器学习原型序列表征, 同时在解码端更深入地利用高质量原型序列所含语义信息、减少低质量原型序列引入的噪声信息. 通过实验验证了本文方法在低资源环境下及相似性语言环境下的适用性, 同时也表明本文方法并不局限于特定语言, 具有可迁移性. 下一步将在更多的数据集上实践所提出的方法.

References:

[1] Sutskever I, Vinyals O, Le QV. Sequence to sequence learning with neural networks. In: Proc. of the 27th Int'l Conf. on Neural Information Processing Systems. Montreal: MIT Press, 2014. 3104–3112.

[2] Bahdanau D, Cho K, Bengio Y. Neural machine translation by jointly learning to align and translate. In: Proc. of the 3rd Int'l Conf. on Learning Representations. San Diego: ICLR, 2015. 1–15.

[3] Li YC, Xiong DY, Zhang M. A survey of neural machine translation. Chinese Journal of Computers, 2018, 41(12): 2734–2755 (in Chinese with English abstract). [doi: 10.11897/SP.J.1016.2018.02734]

[4] Zhang JY, Utiyama M, Sumita E, Neubig G, Nakamura S. Guiding neural machine translation with retrieved translation pieces. In: Proc. of the 2018 Conf. of the North American Chapter of the Association for Computational Linguistics. New Orleans: ACL, 2018. 1325–1335. [doi: 10.18653/v1/N18-1120]

[5] Bulté B, Tezcan A. Neural fuzzy repair: Integrating fuzzy matches into neural machine translation. In: Proc. of the 57th Annual Meeting of the Association for Computational Linguistics. Florence: ACL, 2019. 1800–1809. [doi: 10.18653/v1/P19-1175]

[6] Cao Q, Xiong DY. Encoding gated translation memory into neural machine translation. In: Proc. of the 2018 Conf. on Empirical Methods in Natural Language Processing. Brussels: ACL, 2018. 3042–3047. [doi: 10.18653/v1/D18-1340]

[7] Wang YR, Xia YC, Tian F, Gao F, Qin T, Zhai CX, Liu TY. Neural machine translation with soft prototype. In: Proc. of the 33rd Int'l Conf. on Neural Information Processing Systems. Vancouver: Curran Associates Inc., 2019. 567.

[8] Xu JT, Crego J, Senellart J. Boosting neural machine translation with similar translations. In: Proc. of the 58th Annual Meeting of the Association for Computational Linguistics. ACL, 2020. 1580–1590. [doi: 10.18653/v1/2020.acl-main.144]

[9] Vanallemeersch T, Vandeghinste V. Assessing linguistically aware fuzzy matching in translation memories. In: Proc. of the 18th Annual Conf. of the European Association for Machine Translation. Antalya: ACL, 2015. 153–160.

[10] Bloodgood M, Strauss B. Translation memory retrieval methods. In: Proc. of the 14th Conf. of the European Chapter of the Association for Computational Linguistics. Gothenburg: ACL, 2014. 202–210. [doi: 10.3115/v1/E14-1022]

[11] Pagliardini M, Gupta P, Jaggi M. Unsupervised learning of sentence embeddings using compositional n-gram features. In: Proc. of the 2018 Conf. of the North American Chapter of the Association for Computational Linguistics. New Orleans: ACL, 2018. 528–540. [doi: 10.18653/v1/N18-1049]

[12] Zhu JH, Xia YC, Wu LJ, He D, Qin T, Zhou WG, LI HQ, Liu TY. Incorporating BERT into neural machine translation. In: Proc. of the 8th Int'l Conf. on Learning Representations. Addis Ababa: ICLR, 2020. 26–30.

[13] Hashimoto TB, Guu K, Oren Y, Liang P. A retrieve-and-edit framework for predicting structured outputs. In: Proc. of the 32nd Int'l Conf. on Neural Information Processing Systems. Montréal: Curran Associates Inc., 2018. 10073–10083.

[14] Ren S, Wu Y, Liu SJ, Zhou M, Ma S. A retrieve-and-rewrite initialization method for unsupervised machine translation. In: Proc. of the 58th Annual Meeting of the Association for Computational Linguistics. ACL, 2020. 3498–3504. [doi: 10.18653/v1/2020.acl-main.320]

[15] He JX, Berg-Kirkpatrick T, Neubig G. Learning sparse prototypes for text generation. In: Proc. of the 34th Neural Information Processing Systems. 2020.

[16] Hokamp C. Ensembling factored neural machine translation models for automatic post-editing and quality estimation. In: Proc. of the 2nd Conf. on Machine Translation. Copenhagen: ACL, 2017. 647–654. [doi: 10.18653/v1/W17-4775]

[17] Dabre R, Cromieres F, Kurohashi S. Enabling multi-source neural machine translation by concatenating source sentences in multiple languages. arXiv:1702.06135, 2017.

[18] Dinu G, Mathur P, Federico M, Al-Onaizan Y. Training neural machine translation to apply terminology constraints. In: Proc. of the 57th Annual Meeting of the Association for Computational Linguistics. Florence: ACL, 2019. 3063–3068. [doi: 10.18653/v1/P19-1294]

[19] Song K, Zhang Y, Yu H, Luo WH, Wang K, Zhang M. Code-switching for enhancing NMT with pre-specified translation. In: Proc. of the 2019 Conf. of the North American Chapter of the Association for Computational Linguistics. Minneapolis: ACL, 2019. 449–459. [doi: 10.18653/v1/N19-1044]

[20] Xia YC, Tian F, Wu LJ, Lin JX, Qin T, Yu NH, Liu TY. Deliberation networks: Sequence generation beyond one-pass decoding. In: Proc. of the 31st Int'l Conf. on Neural Information Processing Systems. Long Beach: Curran Associates Inc., 2017. 1782–1792.

[21] Zhang XW, Su JS, Qin Y, Liu Y, Ji RR, Wang HJ. Asynchronous bidirectional decoding for neural machine translation. In: Proc. of the 32nd AAAI Conf. on Artificial Intelligence. New Orleans: AAAI, 2018. 5698–5705. [doi: 10.1609/aaai.v32i1.11984]

[22] Zhou L, Zhang JJ, Zong CQ. Synchronous bidirectional neural machine translation. Trans. of the Association for Computational Linguistics, 2019, 7: 91–105. [doi: 10.1162/tacl_a_00256]

[23] Yang J, Ma SM, Zhang DD, Li ZJ, Zhou M. Improving neural machine translation with soft template prediction. In: Proc. of the 58th Annual Meeting of the Association for Computational Linguistics. ACL, 2020. 5979–5989. [doi: 10.18653/v1/2020.acl-main.531]

[24] Qin LB, Ni MH, Zhang Y, Che WX. CoSDA-ML: Multi-lingual code-switching data augmentation for zero-shot cross-lingual NLP. In: Proc. of the 29th Int'l Joint Conf. on Artificial Intelligence. Yokohama: IJCAI.org, 2021. 533.

[25] Sennrich R, Zhang B. Revisiting low-resource neural machine translation: A case study. In: Proc. of the 57th Annual Meeting of the Association for Computational Linguistics. Florence: ACL, 2019. 211–221. [doi: 10.18653/v1/P19-1021]

[26] Papineni K, Roukos S, Ward T, Zhu WJ. BLEU: A method for automatic evaluation of machine translation. In: Proc. of the 40th Annual Meeting of the Association for Computational Linguistics. Philadelphia: ACL, 2002. 311–318. [doi: 10.3115/1073083.1073135]

[27] Isozaki H, Hirao T, Duh K, Sudoh K, Tsukada H. Automatic evaluation of translation quality for distant language pairs. In: Proc. of the 2010 Conf. on Empirical Methods in Natural Language Processing. Cambridge: ACL, 2010. 944–952.

[28] Koehn P. Statistical significance tests for machine translation evaluation. In: Proc. of the 2004 Conf. on Empirical Methods in Natural Language Processing. Barcelona: ACL, 2004. 388–395.

[29] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser Ł, Polosukhin I. Attention is all you need. In: Proc. of the 31st Int'l Conf. on Neural Information Processing Systems. Long Beach: Curran Associates Inc., 2017. 6000–6010.

[30] Ding CC, Utiyama M, Sumita E. Similar southeast Asian languages: Corpus-based case study on Thai-Laotian and Malay-Indonesian. In: Proc. of the 3rd Workshop on Asian Translation. Osaka: The COLING 2016 Organizing Committee, 2016. 149–156.

[31] Singvongsa K, Seresangtakul P. Lao-Thai machine translation using statistical model. In: Proc. of the 13th Int'l Joint Conf. on Computer Science and Software Engineering. Khon Kaen: IEEE, 2016. 1–5. [doi: 10.1109/JCSSE.2016.7748893]

附中文参考文献:

[3] 李亚超, 熊德意, 张民. 神经机器翻译综述. 计算机学报, 2018, 41(12): 2734–2755. [doi: 10.11897/SP.J.1016.2018.02734]

于志强(1983－), 男, 博士, 主要研究领域为自然语言处理, 神经机器翻译.

郭军军(1987－), 男, 博士, 副教授, CCF 专业会员, 主要研究领域为自然语言处理, 神经机器翻译, 多模态机器翻译.

余正涛(1970－), 男, 博士, 教授, 博士生导师, CCF 高级会员, 主要研究领域为自然语言处理, 神经机器翻译, 信息检索.

线岩团(1982－), 男, 副教授, CCF 专业会员, 主要研究领域为自然语言处理, 神经机器翻译.

黄于欣(1983－), 男, 博士, CCF 专业会员, 主要研究领域为自然语言处理, 神经机器翻译, 文本摘要.