中国科学院大学

University of Chinese Academy of Sciences

博士学位论文

非自回归神经机器翻译的训练方法研究

作者姓名：邵晨泽

指导教师：冯洋 研究员

中国科学院计算技术研究所

学位类别：工学博士

学科专业：计算机应用技术

培养单位：中国科学院计算技术研究所

2023 年 6 月

Training Methods for Non-Autoregressive Neural Machine Translation

A dissertation submitted to

University of Chinese Academy of Sciences

in partial fulfillment of the requirement

for the degree of

Doctor of Philosophy

in Computer Applied Technology

By

Shao Chenze

Supervisor: Professor Feng Yang

Institute of Computing Technology, Chinese Academy of Sciences

June, 2023

中国科学院大学
学位论文原创性声明

本人郑重声明：所呈交的学位论文是本人在导师的指导下独立进行研究工作所取得的成果。承诺除文中已经注明引用的内容外，本论文不包含任何其他个人或集体享有著作权的研究成果，未在以往任何学位申请中全部或部分提交。对本论文所涉及的研究工作做出贡献的其他个人或集体，均已在文中以明确方式标明或致谢。本人完全意识到本声明的法律结果由本人承担。

作者签名：

日　　期：

中国科学院大学
学位论文授权使用声明

本人完全了解并同意遵守中国科学院大学有关收集、保存和使用学位论文的规定，即中国科学院大学有权按照学术研究公开原则和保护知识产权的原则，保留并向国家指定或中国科学院指定机构送交学位论文的电子版和印刷版文件，且电子版与印刷版内容应完全相同，允许该论文被检索、查阅和借阅，公布本学位论文的全部或部分内容，可以采用扫描、影印、缩印等复制手段以及其他法律许可的方式保存、汇编本学位论文。

涉密及延迟公开的学位论文在解密或延迟期后适用本声明。

作者签名：　　导师签名：

日　　期：　　日　　期：

摘 要

机器翻译是指通过计算机将源语言句子翻译到与之语义等价的目标语言句子的过程，是自然语言处理领域的一个重要研究方向。神经机器翻译仅使用神经网络就能实现从源语言到目标语言的端到端翻译，其性能逐渐实现了对统计机器翻译的超越，目前已成为机器翻译研究的主流范式。神经机器翻译模型目前主要以自回归的方式对从源端到目标端的翻译概率进行建模，每一步的译文单词的生成都依赖于之前的翻译结果，因此模型只能逐词生成译文，无法充分利用硬件的并行计算能力，导致模型的解码速度较慢。

非自回归神经机器翻译是一种新兴的翻译模型，它对目标端的概率分布做了条件独立性假设，仅依赖于源端文本来生成所有译文单词，因此能并行生成整句译文，在解码速度上相较于自回归模型有显著优势。然而，在实际应用中，非自回归模型的性能与自回归模型有较大差距，具体表现为译文中包含较多的重复词，并会遗漏原文中的一些信息。造成这种性能差距的主要原因为极大似然估计的训练目标与非自回归模型表达能力的不匹配。类似于多数生成模型，非自回归模型也采用极大似然估计的方式进行训练，希望模型能够拟合训练数据的分布。然而，机器翻译任务并不满足条件独立性假设，同一原文可能存在多种正确的译文，译文的前后词之间存在较强的依赖关系，而这种目标端依赖恰好是非自回归模型表达能力缺失的部分。因此，非自回归模型在理论上无法通过参数估计拟合机器翻译训练数据的分布，这导致极大似然估计无法使模型学习到正确的翻译方式。针对这一挑战，本文不局限于极大似然估计的训练范式，从不同角度改进非自回归神经机器翻译模型的训练方法，详细研究内容如下：

1. 基于强化学习的序列级训练方法

针对词级交叉熵损失不准确的问题，本文提出基于强化学习的序列级训练方法，在序列层面上训练非自回归模型。非自回归模型无法通过极大似然估计学习到正确的翻译方式，这在实践上表现为基于极大似然估计的词级交叉熵损失独立评估每个位置的翻译质量，忽略了译文前后间的依赖关系，导致非自回归模型关注译文的局部正确性而忽略了整体翻译质量。对此，一个直接的改进思路是从整体评估模型的输出结果，令非自回归模型直接优化序列级的评估指标。本文指出了非自回归模型在训练目标上的缺陷，提出了基于强化学习的序列级训练方法，并针对非自回归模型改进强化学习算法来降低梯度估计的方差。实验结果表明，序列级训练能显著改善非自回归模型的翻译质量，使其在性能接近自回归模型的同时仍能保持着相对自回归模型十倍以上的解码加速。

2. 基于 n 元组匹配的训练方法

针对强化学习方法误差较高的问题，本文提出基于 *n* 元组匹配来设计的训练目标，能够高效、准确地训练非自回归模型。机器翻译的评估指标主要基于 n 元组的匹配准确率来评估模型的翻译质量。因此，若能直接基于 *n* 元组的匹配来

设计训练目标，就能更准确、高效地训练非自回归模型。本文将这类训练目标形式化为最小化 n 元组词袋间的距离，并针对非自回归模型设计了优化其一阶距离的高效算法。进一步地，本文将 n 元组匹配方法扩展到了基于 CTC 的非自回归模型架构中，针对模型只能建模单调对齐的局限性，基于 n 元组匹配建模模型输出与参考译文的非单调对齐。在基于 n 元组匹配的训练目标优化下，非自回归模型能够达到与自回归模型相似的性能水平。

3. 基于动态参考译文的训练方法

针对参考译文可能与模型输出不匹配的问题，本文提出基于动态参考译文的训练方法，将参考译文动态调整为合适的形式来训练模型。在非自回归模型的训练中，当模型没有按照参考译文的形式输出结果时，交叉熵损失就无法正确地评估模型输出的好坏。本文提出了基于动态参考译文的解决方案，不再使用静态的参考译文，而是将参考译文动态调整为合适的形式来训练模型。动态调整后的参考译文应既能保持自身的正确性，又与模型的输出相匹配。本文将其建模为对参考译文的优化问题，并给出了基于多样蒸馏和改写器的两种优化方法。实验结果表明，动态参考译文方法能有效地与损失函数方面的改进结合，使非自回归模型达到甚至超越自回归模型的翻译质量。

关键词：神经机器翻译，非自回归，序列级训练，n 元组匹配，动态参考译文

Abstract

Machine translation refers to the process of translating source language sentences into semantically equivalent target language sentences through the use of computers. It is an important research direction in the field of natural language processing. Neural machine translation enables end-to-end translation from source to target language solely using neural networks, which has gradually surpassed statistical machine translation in terms of translation performance and has now become the dominant paradigm in machine translation research. Neural machine translation models mainly represent the translation probability from source to target in an autoregressive way, where the translation probability at each step depends on the previous translation results. As a result, the model can only generate target words step by step and is unable to fully utilize the hardware's parallel computing capability, leading to a large inference latency.

Non-autoregressive neural machine translation is a novel kind of translation models that assume conditional independence of the target-side distribution. This leads to the parallel generation of the entire translation and results in a significant advantage in terms of decoding speed compared to autoregressive models. However, in practical applications, there is a large performance gap between non-autoregressive and autoregressive models, specifically manifested in over-translation and under-translation errors in the model output. The major reason of this performance gap is the mismatch between the training method of maximum likelihood estimation and the representation power of non-autoregressive models. Like most generative models, non-autoregressive models also employ the maximum likelihood estimation method for the training, with the underlying goal of fitting the distribution of the training data. However, the task of machine translation does not satisfy the conditional independence assumption. As a source sentence may have multiple correct translations, there is a strong sequential dependency in the target sentence, which is beyond the representation power of non-autoregressive models. Therefore, in machine translation, non-autoregressive models are theoretically unable to fit the distribution of training data through parameter estimation, so they are unable to correctly learn to translate through maximum likelihood estimation. In response to this challenge, this thesis is not limited to the training paradigm of maximum likelihood estimation and improves the training methods of non-autoregressive neural machine translation models from different perspectives. The detailed research content is as follows:

1. Sequence-Level Training Methods based on Reinforcement Learning

To address the issue of inaccurate word-level cross-entropy loss, this thesis proposes a sequence-level training method based on reinforcement learning to train

non-autoregressive models at the sequence level. Non-autoregressive models cannot correctly learn to translate through maximum likelihood estimation. It is practically demonstrated as the inaccuracy of the word-level cross-entropy loss based on maximum likelihood estimation, which independently evaluates the translation quality of each step and ignores the dependency between target words. As a result, non-autoregressive models only focus on local correctness and neglect the overall quality of the translation. To address this issue, a straightforward solution is to evaluate the model output as a whole and directly train non-autoregressive models with sequence-level evaluation metrics. This thesis points out the defects of non-autoregressive models in the training and proposes a sequence-level training solution based on reinforcement learning. The reinforcement learning algorithm is customized for non-autoregressive models to reduce the variance of gradient estimation. Experimental results show that sequence-level training significantly improves the translation quality of non-autoregressive models, reducing the performance gap with autoregressive models while still maintaining a decoding acceleration of over ten times.

2. Training Methods based on N-Gram Matching

To address the issue of high variance in reinforcement learning, this thesis proposes training objectives designed based on n-gram matching, which enables efficient and accurate training of non-autoregressive models. Evaluation metrics for machine translation mainly assess the accuracy of n-gram matching to evaluate the translation quality. Therefore, directly designing loss functions based on n-gram matching can enable more accurate and efficient training of non-autoregressive models. This thesis formalizes these training objectives as minimizing the distance between bag-of-ngrams and proposes an efficient algorithm to minimize the first-order distance of bag-of-ngrams for non-autoregressive models. Furthermore, this thesis extends n-gram matching to CTC-based model architecture to model non-monotonic alignments between the model output and reference, which overcomes the limitation that CTC only considers monotonic alignments. With the optimization of n-gram level training objectives, non-autoregressive models can achieve comparable performance to autoregressive models.

3. Training Methods based on Dynamic Reference

To address the issue of mismatch between model output and reference, this thesis proposes training methods based on dynamic reference, which dynamically adapted the reference to an appropriate form for training the model. In the training of non-autoregressive models, the model output will not be correctly evaluated by the cross-entropy loss if it does not follow the format of the reference. This thesis presents a solution based on dynamic reference, which does not use the static reference but dynamically adapted the reference for the training. The dynamic reference should match

the model output while keeping the original semantics of the reference. This thesis represents it as an optimization problem for the reference, and gives two solutions based on diverse distillation and rephraser. Experimental results show that the dynamic reference can effectively work together with improved loss functions and help non-autoregressive models achieve the same and even better translation quality than autoregressive models.

Key Words: Neural Machine Translation, Non-Autoregressive, Sequence-Level Training, N-Gram Matching, Dynamic Reference

目 录

图目录

表目录

b

第 1 章　引言

1.1　研究背景

翻译是连接不同文化和语言的桥梁，能够帮助人们跨越语言和文化障碍，促进跨国贸易、国际合作和文化交流。随着全球化的不断发展，越来越多的人需要进行跨语言的交流和合作，人工翻译已经无法满足日益增长的翻译需求。借助计算机将一种语言转换为与之语义等价的另一种语言的过程被称为机器翻译，它是自然语言处理领域中的一个重要研究课题，能使跨语言交流变得更加便捷和高效，自计算机诞生以来就受到研究者们的广泛关注。使用计算机进行翻译的思想可以追溯到 20 世纪 30 年代。当时，法国科学家 Georges Artsrouni 提出使用机器来进行翻译的想法，苏联科学家 Peter Troyanskii 设计了把一种语言翻译为另一种语言的机器，但这些想法受技术水平限制均没有成功。1946 年，第一台通用计算机 ENIAC 在美国诞生，随后信息论先驱 Warren Weaver 于 1949 年在《翻译备忘录》[2] 中正式提出机器翻译的概念，机器翻译研究从此开始蓬勃发展。

在机器翻译研究初期，人们通常采用基于规则的翻译方法，基于词典和手工书写的翻译规则来将原文替换成目标语言的对应译文。这种方式可以较好地保持原文结构，产生的译文与原文关系密切，在格式较固定的句子上有较高的翻译精度。然而，在口语等非规范的语言场景中，基于规则的方法难以覆盖所有的翻译现象，并且容易出现不同规则之间的冲突，系统健壮性差。1990 年，IBM 的 P.F.Brown 等人在《计算语言学》期刊上发表了论文《统计机器翻译方法》[3]，将机器翻译建模为一种概率模型，并通过统计方法利用大规模的平行语料来优化模型。1993 年，IBM 的研究人员进一步提出了基于噪声信道模型的 5 种 IBM 模型[4]，奠定了统计机器翻译的理论基础，使其逐渐成为机器翻译的主流方法。虽然不再需要手工书写规则，但统计机器翻译的模型仍比较复杂，包含多个子模块，每个模块的错误都会影响最终的翻译质量。

进入 21 世纪，随着计算机算力的发展，深度学习的方法逐渐成熟，并开始被应用在自然语言处理领域。2014 年，Cho 等[5] 提出了基于循环神经网络的编码器-解码器框架，将机器翻译建模为序列到序列的转换问题，并将其集成到统计机器翻译模型中。随后，Sutskever 等[6] 进一步引入多层长-短时记忆网络进行编码、解码，仅使用神经网络进行端到端的机器翻译。同年，Bahdanau 等[7] 在编码器-解码器的框架中引入注意力机制，使模型在解码时能注意到源端对应词的信息，显著提升了模型的性能，引发了研究者们对神经机器翻译的广泛关注。2017 年，Vaswani 等[1] 提出了完全基于注意力机制的 Transformer 模型，不仅提高了模型的训练效率，还将神经机器翻译的性能提升到了一个新的台阶。相比于统计机器翻译，神经机器翻译具有模型上的简洁性和性能上的优越性，因此迅速成为了学术研究和工业应用的主流范式。

神经机器翻译（Neural Machine Translation，NMT）相较于基于规则和统计

的方法具有更好的翻译性能，但其代价是需要基于深度神经网络做大量的计算。基于规则和统计的模型只需要使用 CPU 就能满足训练和测试的计算需求，而神经机器翻译模型需要依赖 GPU 等设备进行大规模的并行计算。目前，神经机器翻译模型主要是以自回归的方式对从源端到目标端的翻译概率进行建模，每一步的译文单词生成都依赖于之前的翻译结果。因此，模型在解码时只能逐词生成译文，无法充分利用硬件的并行计算能力，导致模型的解码速度较慢，尤其是在生成长句时具有较高的延迟。这一缺陷使得神经机器翻译模型在解码时需要消耗较多的计算资源，并且影响了它在需要模型快速响应的实时翻译场景下的应用效果。

随着 Vaswani 等[1] 提出的 Transformer 结构使模型能够在序列层面上并行训练，模型的并行解码能力也开始受到研究者们的关注。2017 年末，Gu 等[8] 提出非自回归神经机器翻译模型（Non-Autoregressive Neural Machine Translation，NAT），它对目标端的概率分布做了条件独立性假设，仅依赖于源端文本来独立地生成所有译文单词。因此，非自回归模型在解码时能并行生成整句译文，在解码速度上相较于自回归模型有显著优势。然而，非自回归模型的翻译质量与自回归模型有较大差距，生成的译文类似于多种翻译结果的混合，其中包含较多的重复词，并会遗漏原文中的一些信息，这使得非自回归模型难以被投入到实际应用中。

Gu 等[8] 将非自回归模型在机器翻译任务上性能较差的现象归因为多峰性问题，即同一原文可能存在多种正确的译文，译文的前后词之间存在较强的依赖关系，而这种目标端依赖恰好是非自回归模型表达能力缺失的部分。因此，多峰性问题使得非自回归模型在理论上无法通过参数估计拟合机器翻译训练数据的分布，造成了极大似然估计的训练目标与非自回归模型表达能力的不匹配，导致模型无法通过极大似然估计学习到正确的翻译方式。针对这一挑战，本文不局限于极大似然估计的训练范式，从不同角度改进非自回归神经机器翻译模型的训练目标，旨在降低非自回归模型的学习难度，使非自回归模型在保持解码速度优势的同时达到甚至超越自回归模型的翻译质量。

1.2 研究现状

本文的研究对象为非自回归神经机器翻译模型，本节将会对相关研究现状进行介绍，包括神经机器翻译模型的主流结构、非自回归神经机器翻译模型的实现方式以及其面临的多峰性问题。

1.2.1 神经机器翻译

神经机器翻译模型使用神经网络来学习源语言和目标语言之间的语义映射关系，相较于基于规则和统计的方法具有更好的翻译性能，目前已成为机器翻译的主流方法。神经机器翻译模型通常采用编码器-解码器框架，编码器网络将源句编码为向量表示，解码器网络再从源端编码的表示中解码出目标端句子。目

前，神经机器翻译模型主要是以自回归的方式对从源端到目标端的翻译概率进行建模。给定源端输入序列 X 与目标端译文 $Y = \{y_1, y_2, ..., y_T\}$，自回归神经机器翻译模型将翻译概率链式分解，按下式进行建模：

$$p(Y|X,\theta) = \prod_{t=1}^{T} p(y_t|X, y_{<t}, \theta). \tag{1-1}$$

神经机器翻译模型通常采用极大似然估计的方式进行训练，损失函数如下所示：

$$\mathcal{L}(\theta) = \sum_{t=1}^{T} \log(p(y_t|X, y_{<t}, \theta)). \tag{1-2}$$

在解码时，自回归神经机器翻译模型通常采用贪心搜索（Greedy Search）或束搜索（Beam Search）算法来生成译文，贪心搜索在每个时间步选择概率最高的单词作为预测结果，束搜索算法则是维护一个大小为 k 的集束来保存候选的译文前缀，并在解码过程中根据翻译概率大小动态更新集束中的候选译文。受限于自回归的概率建模机制，这类解码方法每次均只能生成一个词，生成长度为 T 的译文就需要至少运行 T 次解码器，因此在生成长句时具有较高的延迟。

在编码器-解码器框架下，两种主要的神经机器翻译模型架构分别为 RNNSearch[7] 和 Transformer[1]。基于循环神经网络和注意力机制的 RNNSearch 模型在很长一段时间内都是神经机器翻译的主流模型。该模型引入了注意力机制，使得解码器在每个解码时间步都可以将注意力集中在源端最相关的几个词语上，从中获取有效的信息来预测当前译文词。注意力机制解决了信息在循环神经网络的长距离传输中容易被丢失、遗忘的问题，使模型能更好地处理长距离的依赖关系。

RNNSearch 模型使用双向门控循环单元（GRU）将源语句编码为一组稠密的语义向量。假设输入序列为 $X = \{x_1, x_2, ..., x_L\}$，则编码器的输出如下式所示：

$$\overrightarrow{h_i} = \overrightarrow{\text{GRU}}(x_i, \overrightarrow{h_{i-1}}), \quad \overleftarrow{h_i} = \overleftarrow{\text{GRU}}(x_i, \overleftarrow{h_{i+1}}), \quad h_i = [\overrightarrow{h_i}, \overleftarrow{h_i}]. \tag{1-3}$$

模型的解码器由前向门控循环单元和注意力模块组成。在时间步 j 时，先计算解码器隐状态 s_{j-1} 对源端每个状态 h_i 的注意力，再根据注意力对源端隐状态加权求和，得到注意力机制的输出 a_j：

$$e_{ij} = v_a^T \tanh(W_a s_{j-1} + U_a h_i), \quad \alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{i'=1}^{L} \exp(e_{i'j})}, \quad a_j = \sum_{i=1}^{L} \alpha_{ij} h_i. \tag{1-4}$$

其中，v_a、W_a、U_a 为模型参数。随后，前向门控单元将解码器输入 y_{j-1}、隐状态 s_{j-1}、注意力输出 a_j 作为输入来计算当前步的隐状态：

$$s_j = \text{GRU}(y_{j-1}, s_{j-1}, a_j). \tag{1-5}$$

最后，模型根据隐状态 s_j、解码器输入 y_{j-1}、注意力输出 a_j 来计算当前步的概率分布：

$$p(y_j|X, y_{<j}, \theta) \propto \exp(W_{y_j} \cdot g(s_j, y_{j-1}, a_j)), \tag{1-6}$$

其中，g 为基于前馈神经网络的非线性映射，W_{y_j} 为输出层中对应词 y_j 的参数。

基于循环神经网络的 RNNSearch 模型的主要缺陷为训练效率较低：由于循环神经网络的限制，不同时间步的隐状态之间存在顺序依赖关系，导致模型无法在序列层面上实现并行化，在训练时对 GPU 算力的利用程度较低。2017 年，Vaswani 等[1] 提出了完全基于注意力机制的 Transformer 模型，摒弃了序列建模中的循环结构，使模型能够在序列层面上并行训练，模型的训练效率得以大幅提升。Transformer 的模型架构如图1-1所示。

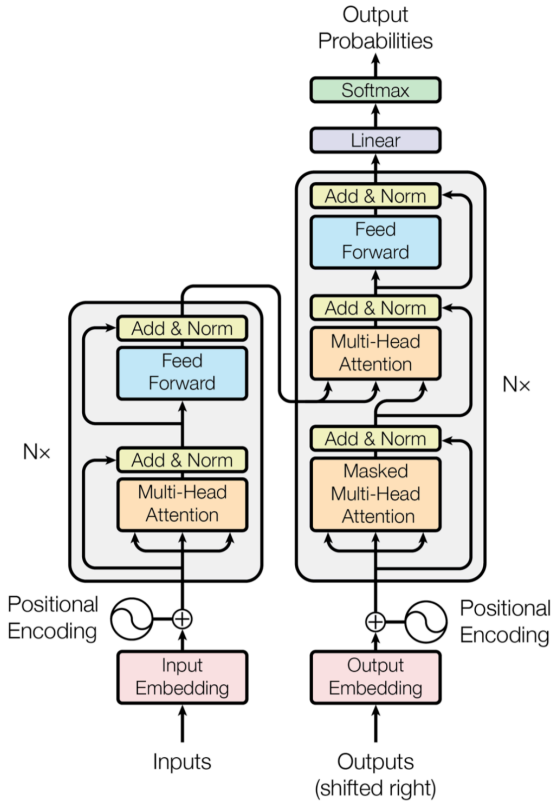


图 1-1 Transformer 模型架构，图片引用自 Vaswani 等[1]。

Figure 1-1 The model architecture of Transformer. The figure is cited from Vaswani 等[1].

Transformer 模型的并行特性导致其没有对序列顺序的显式利用，因此模型需要引入位置编码，以表示序列中不同词的位置关系：

$$\begin{aligned} \mathrm{PE}_{(\mathrm{pos},2i)} &= \sin(\mathrm{pos}/10000^{2i/d_{\mathrm{model}}}), \\ \mathrm{PE}_{(\mathrm{pos},2i+1)} &= \cos(\mathrm{pos}/10000^{2i/d_{\mathrm{model}}}), \end{aligned} \tag{1-7}$$

其中，pos 表示词在句子中的位置，d_{model} 表示模型的维度。模型编码器的输入为词嵌入与位置编码的加和。编码器由 n 个结构相同的层组成，每层主要包含多头自注意力和前馈神经网络两个模块。多头自注意力模块以点乘注意力为基础，将注意力建模为三个序列输入 Q、K、V 的运算，如下式所示：

$$\mathrm{Attention}(Q,K,V) = \mathrm{softmax}(\frac{QK^T}{\sqrt{d_k}})V, \tag{1-8}$$

其中，d_k 为输入 K 的维度。在编码器的自注意力模块中，令模块输入同时作为 Q、K、V，即输入序列对自身做注意力运算。此外，作者还提出多头注意力的机制，即将注意力模块的输入均匀地分成多份，每份独立地进行注意力运算，最后将不同头的输出合并。在经过自注意力计算后，再用包含一个隐层的前馈神经网络对模块输入做变换，如下式所示：

$$\mathrm{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2. \tag{1-9}$$

Transformer 的解码器架构与编码器类似，主要不同点为解码器额外包含一个交叉注意力模块，令解码器可以将注意力集中在源端最相关的几个词语上。在交叉注意力模块中，模块的输入作为式1-8中的 Q，编码器的输出作为式1-8中的 K 和 V，进行解码器与编码器之间的注意力运算。另外，由于译文在解码时是单向生成的，前面的解码时间步无法看到后面的译文，所以解码器的自注意力模块引入了一个对角的掩码矩阵，使从前往后的注意力大小被固定为 0。解码器的输出最终被映射到词表大小，通过 softmax 层计算译文的概率 $p(y_j|X, y_{<j}, \theta)$。除上述主要模块，Transformer 模型中还应用了残差连接[9]、层归一化[10] 等技术来辅助模型训练。

1.2.2　非自回归神经机器翻译

随着 Vaswani 等[1] 提出的 Transformer 使模型能够在序列层面上实现并行训练，模型的并行解码能力也开始受到研究者们的关注。Gu 等[8] 提出非自回归神经机器翻译模型，对译文中所有词的生成进行独立的概率建模，并基于 Transformer 实现了译文的并行解码，显著提升了模型的翻译速度。非自回归 Transformer 的模型架构如图1-2所示。

给定源端输入序列 X 与目标端译文 $Y = \{y_1, y_2, ..., y_T\}$，非自回归模型将从 X 到 Y 的翻译概率建模为：

$$p(Y|X, \theta) = \prod_{t=1}^{T} p_t(y_t|X, \theta), \tag{1-10}$$

其中，θ 表示模型的参数，$p_t(y_t|X, \theta)$ 表示模型在位置 t 对目标端单词 y_t 的翻译概率。基础的非自回归模型同样使用词级别的交叉熵损失函数训练模型：

$$\mathcal{L}(\theta) = \sum_{t=1}^{T} \log(p_t(y_t|X, \theta)). \tag{1-11}$$

基础的非自回归 Transformer 模型与自回归 Transformer 模型的架构基本相同，仅存在几处差异。由于非自回归模型在预测 y_t 时不依赖于其他目标端单词，因此不能像自回归模型一样将目标端翻译历史作为解码器输入。对此，最简单的应对方法是给解码器输入一串空字符，Gu 等[8] 也提出了两种基于拷贝的方法。第一种方法是平均拷贝，即将源端输入的词嵌入均匀地拷贝到目标端，作为解码

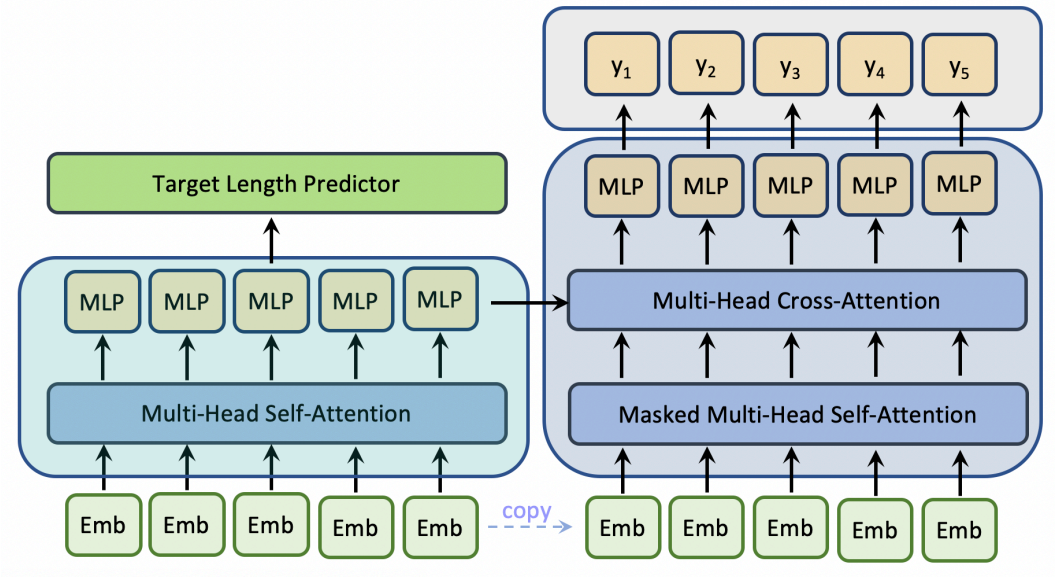


图 1-2 非自回归 Transformer 模型架构。

Figure 1-2 The model architecture of non-autoregressive Transformer.

器的输入。第二种方法是基于繁衍率的拷贝，繁衍率即源端单词对应目标端译文单词的数目，模型先获取每个源端单词的繁衍率，再根据繁衍率将每个源端单词拷贝指定次数到目标端。另外，由于解码器的输入不包含目标端译文信息，所以也不需要在自注意力模块引入对角的掩码矩阵。

基础的非自回归模型需要预先知道译文的长度，再去生成指定长度的译文。因此，模型还包含一个长度预测模块，以模型对源端句子的编码结果作为输入，通过前馈神经网络的计算预测译文长度。在训练时，模型使用参考译文的长度进行解码，并用其训练长度预测模块。在测试时，模型先预测出译文长度，再使用 argmax 解码方法得到概率最高的译文：

$$\hat{y}_t = \text{argmax}_{y_t} p_t(y_t|X,\theta). \tag{1-12}$$

尽管非自回归模型在解码速度上拥有巨大优势，但基础的非自回归模型的性能与自回归模型有很大差距，使其难以投入到实际应用中。具体来说，非自回归模型的译文中通常包含较多的重复翻译，并容易遗漏原文中的一些信息。如表1-1所示，非自回归模型的输出中缺少了“that”、“and”等连接词，并将“cancers”、“aggressive”重复输出了一次，这分别是非自回归模型中典型的漏翻译、重复翻译现象。

1.2.3 非自回归模型的多峰性问题

非自回归模型在机器翻译任务上表现不佳的主要原因为多峰性问题[8,11]。在机器翻译任务中，同一原文通常有多种正确的译文，例如 ‘‘我今天下午吃了披萨’’ 可以被翻译为 “I ate pizza this afternoon’’ 或 “this afternoon I ate pizza’’。因此，

表 1-1 非自回归模型翻译案例。

Table 1-1 A translation case of non-autoregressive model.

原文	Es gibt Krebsarten , die aggressiv und andere , die indolent sind .
参考译文	There are aggressive cancers and others that are indolent .
自回归	There are cancers that are aggressive and others that are indolent .
非自回归	There are cancers cancer aggressive aggressive others are indindent .

目标端的概率分布可能存在多个高峰，而这种多峰性使得我们难以正确地训练非自回归模型，这就是非自回归神经机器翻译中的多峰性问题。

在理论层面上，非自回归模型没有建模多峰性分布的能力，因此无法通过参数估计拟合机器翻译训练数据的分布。类似于多数生成模型，非自回归模型也采用极大似然估计的方式进行训练，希望模型能够拟合训练数据的分布。假设在训练数据中，‘‘我今天下午吃了披萨’’ 的两种译文各占一半。考虑前两个词的联合分布，则有 “I ate’’ 与 “this afternoon’’ 各占 50% 的概率。然而，非自回归模型无法在条件独立性假设下建模这一联合分布。在基于极大似然估计的训练下，模型最多只能学习到每个位置的边缘分布，即第一个位置下 ‘I’ 与 ‘this’ 各占 50% 的概率，第二个位置下 ‘ate’ 与 ‘afternoon’ 各占 50% 的概率，所得联合分布如图1-3所示。可见，由于非自回归模型在理论上没有建模多峰性分布的能力，其无法通过极大似然估计学习到正确的翻译方式，生成的译文中很可能包含如 “I afternoon’’、“this ate’’ 等前后不连贯的片段。

I ate	this afternoon	I afternoon	this ate
50%→25%	50%→25%	0%→25%	0%→25%

图 1-3 非自回归模型无法建模多峰性分布的案例。

Figure 1-3 A case that non-autoregressive model cannot capture multi-modal distribution.

在实践层面上，多峰性问题导致非自回归模型的损失函数不准确，从而影响了最终的翻译质量。基于极大似然估计的交叉熵损失以词为级别，要求模型在每个位置输出参考译文对应位置的词语。即使模型在训练时生成了正确的译文，一旦该译文与参考译文不一致，词级交叉熵损失就会惩罚所有与参考译文错位的模型输出。如图1-4所示，当模型输出为 “this afternoon I ate pizza’’ 而参考译文为 “I ate pizza this afternoon’’ 时，模型输出了正确的翻译结果，但在交叉熵损失的评估下，所有位置的输出都应被更正为参考译文中的对应词。在这样的训练下，非自回归模型就难以生成正确的译文，反而更倾向于生成多种译文的混合，导致了重复词、信息遗漏等非自回归模型中常见的翻译错误。

由上述分析可见，多峰性问题在理论上导致了模型无法拟合训练数据的分布，在实践上导致了模型训练时的损失函数不准确，这两方面缺陷都与极大似然

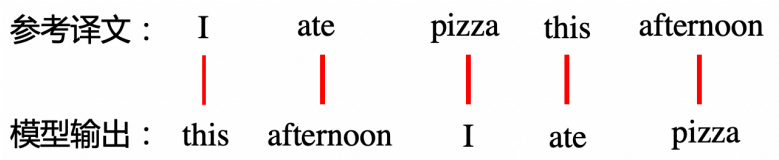


图 1-4 交叉熵损失无法准确评估非自回归模型输出结果的案例。

Figure 1-4 A case that the cross-entropy loss cannot correctly evaluate the output of NAT.

估计有着密切联系。如果不局限于极大似然估计的训练范式，不要求非自回归模型拟合训练数据的分布，而是设计更准确、更适用于非自回归模型的训练方法，就能有效地克服多峰性问题带来的缺陷，改善非自回归模型的翻译质量。

1.3 研究内容

前已述及，非自回归机器翻译中的多峰性问题在理论上导致了模型无法拟合训练数据的分布，在实践上导致了模型训练时的损失函数不准确，这两方面缺陷都与极大似然估计有着密切联系。本文不局限于极大似然估计的训练范式，从不同角度改进非自回归神经机器翻译模型的训练方法，主要研究内容如下：

1.3.1 基于强化学习的序列级训练方法

在第二章中，针对词级交叉熵损失无法准确评估模型输出的问题，本文提出对非自回归模型做序列级训练，从序列层面整体评估模型的输出结果并训练模型。在机器翻译中存在许多对翻译质量的序列级评估指标，但这类指标通常都是离散的，无法直接用于训练模型。因此，本文采用基于强化学习的训练方式，将待优化的指标作为奖赏，令模型以最大化期望奖赏为训练目标，并应用强化学习算法来估计训练目标的梯度。这种训练方式的主要缺陷是梯度估计的方差较高，导致模型在训练时的噪声较大，训练过程不稳定。本文利用非自回归模型并行生成的特性，再结合机器翻译评估指标的性质，针对性地改良了强化学习算法，使非自回归模型能够准确、稳定地优化序列级评估指标。

1.3.2 基于 n 元组匹配的训练方法

在第三、四章中，针对强化学习方法误差较高的问题，本文提出基于 n 元组匹配来设计的训练目标，能够高效、准确地训练非自回归模型。机器翻译的评估指标（如 BLEU 值）主要基于 n 元组的匹配准确率来评估模型的翻译质量。因此，我们进一步考虑直接基于 n 元组的匹配来设计损失函数，从而能避免进行梯度估计，更准确、高效地训练非自回归模型。本文将对 n 元组匹配的优化形式化为最小化 n 元组词袋之间的距离，并利用非自回归模型并行生成的特性设计了能准确计算一阶距离的高效算法，使非自回归模型能直接以最小化 n 元组词袋距离为训练目标。本文进一步将 n 元组匹配的思路扩展到了基于 CTC 的模型架构中，它最早是为语音识别任务设计的结构，只能建模参考译文与模型输出间

的单调对齐，无法处理机器翻译中普遍存在的非单调对齐现象。针对 CTC 无法处理非单调对齐的问题，本文基于 n 元组匹配对 CTC 中的非单调对齐进行建模，弥补了其损失函数的缺陷，使模型的训练更加准确。

1.3.3　基于动态参考译文的训练方法

在第五、六章中，针对参考译文可能与模型输出不匹配的问题，本文提出基于动态参考译文的训练方法，将参考译文动态调整为合适的形式来训练模型。交叉熵损失函数要求模型输出与参考译文严格对齐，因此在多数情况下无法准确地评估模型的输出结果。如果参考译文能按模型输出的形式动态调整，交叉熵损失的准确性就能得到显著改善。基于这个思路，本文提出了基于动态参考译文的解决方案，将参考译文动态调整后再训练模型。本方案的主要难点是如何对参考译文做动态调整，使其既能保持自身的正确性，又与模型的输出相匹配。本文给出了基于多样蒸馏和改写器的两种获取动态参考译文的方法。多样蒸馏方法直接向模型提供多个参考译文，并选择与模型输出最匹配的参考译文来训练模型。改写器方法则是引入了改写器结构，根据模型输出对参考译文做动态改写，并构建奖赏函数来评估改写器的输出是否满足要求，以强化学习的方式训练改写器。这两种方法都能基于模型输出给出更匹配的参考译文，基于这种动态参考译文计算的损失函数就能更准确地训练模型。此外，这两种基于动态参考译文的训练方法也可以与损失函数方面的改进相结合，使模型性能得到进一步提升。

1.4　章节组织

本文的章节组织结构如图1-5所示，具体描述如下：

第一章介绍了机器翻译的历史和目前主流的神经机器翻译模型，以及自回归模型的缺陷和非自回归神经机器翻译的研究意义。本章也介绍了非自回归神经机器翻译中存在的问题和目前的改进方法，并确定了本文的主要研究内容。

第二章介绍了非自回归机器翻译的研究现状，包括知识蒸馏、增强模型表达能力、引入隐变量、设计解码策略、改进训练方法等主流方向。

第三章提出为非自回归模型提出了基于强化学习的序列级训练方法，并针对模型并行生成的特性改良了强化学习算法，使非自回归模型能够准确、稳定地优化序列级评估指标。

第四章基于 n 元组匹配提出了最小化 n 元组词袋距离的训练目标，并针对非自回归模型给出了优化这一训练目标的高效算法。

第五章进一步将 n 元组匹配方法扩展到了基于 CTC 的模型架构中，为 CTC 模型提供了建模非单调对齐的能力，克服了 CTC 模型无法处理非单调对齐的缺陷。

第六章提出多样蒸馏与译文选择结合的方法来提供动态参考译文，多样蒸馏为每个原文提供多个可选的参考译文，译文选择从多个参考译文中选择一个来训练模型。

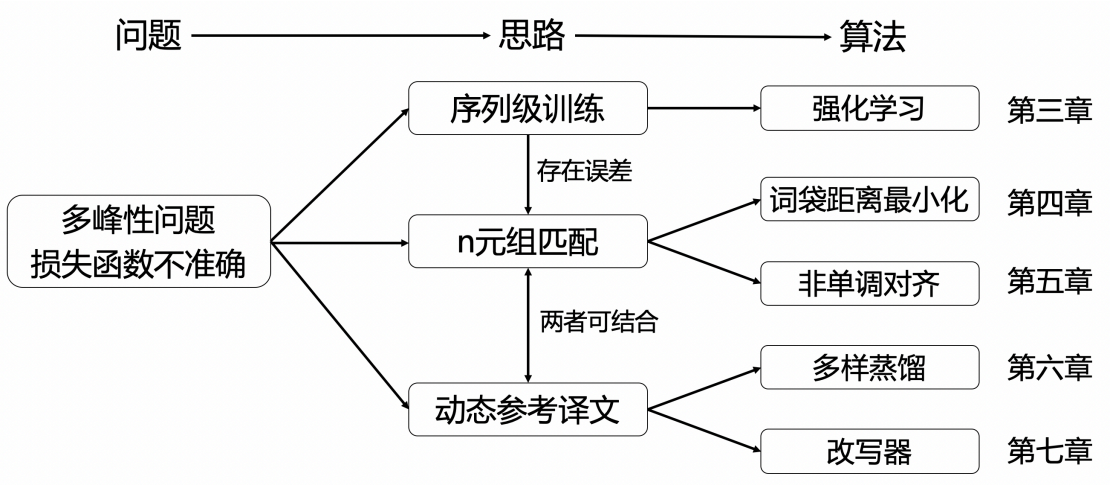


图 1-5 本文章节组织结构图。

Figure 1-5 Chapter organization of this paper.

第七章提出改写器结构来根据模型的输出动态改写参考译文，并将对改写器的要求量化为奖赏函数，通过强化学习方法来优化改写器。

第八章是对本文的总结。

第 2 章　非自回归机器翻译研究现状

多峰性问题在理论上导致了模型无法拟合训练数据的分布，在实践上导致了模型训练时的损失函数不准确，使模型难以学习到正确的翻译方式，翻译结果中会包含很多重复词并且遗漏一些信息。为了提升非自回归模型的翻译性能，研究者们做了许多尝试来减小多峰性问题带来的影响，主要可以分为知识蒸馏、增强表达能力、引入隐变量、设计解码策略、改进训练方法五类。在本章的后续部分，我们将对这五类方法的具体研究现状做详细介绍。

2.1　知识蒸馏

Gu 等[8] 引入的序列级知识蒸馏技术[12] 是非自回归模型的关键技术之一。知识蒸馏通常是指将一个大型神经网络模型的知识转移给另一个小型神经网络模型的过程，大模型通常被称为 ‘‘教师模型’’，小模型被称为 ‘‘学生模型’’。知识蒸馏的目的是让学生模型具有与教师模型相近的性能，在资源受限的场景下有较多应用。传统的知识蒸馏方法一般应用在分类任务上，假设类别数为 K，教师模型预测的分布为 $q(y|x)$，学生模型的参数为 θ，预测结果为 $p(y|x,\theta)$，则知识蒸馏方法要求学生模型去拟合教师模型的分布，损失函数如下：

$$\mathcal{L}_{\text{KD}}(\theta) = -\sum_{k=1}^{K} q(y=k|x) \cdot \log(p(y=k|x,\theta)). \tag{2-1}$$

Kim 等[13] 首先将知识蒸馏方法应用到了神经机器翻译中，具体分为词级知识蒸馏和序列级知识蒸馏。词级知识蒸馏方法直接令学生模型在每个时间步都去学习教师模型的分布，假设词表大小为 $|V|$，则损失函数为：

$$\mathcal{L}_{\text{Word-KD}}(\theta) = -\sum_{t=1}^{T}\sum_{k=1}^{|V|} q(y_t=k|X,y_{<t}) \cdot \log(p(y_t=k|X,y_{<t},\theta)). \tag{2-2}$$

序列级知识蒸馏则是令学生模型直接在序列层面上拟合教师模型的分布，理想化的损失函数如下：

$$\mathcal{L}_{\text{Seq-KD}}(\theta) = -\sum_{Y} q(Y|X) \cdot \log(p(Y|X,\theta)). \tag{2-3}$$

上式需要遍历所有可能的译文，搜索空间是指数级的，因此无法直接计算。Kim 等[13] 提出用一个零一分布来近似教师的分布，其中教师模型的译文概率为 1，其他译文概率均为 0。基于此，上式损失函数的近似形式为：

$$\mathcal{L}_{\text{Seq-KD}}(\theta) \approx -\sum_{Y} 1\{Y=\hat{Y}\} \log(p(Y|X,\theta)) = -\log(p(\hat{Y}|X,\theta)), \tag{2-4}$$

其中，$\hat{Y}$ 为教师模型通过束搜索得到的译文。总而言之，序列级知识蒸馏的具体流程为：（1）训练教师模型，（2）令教师模型对训练集的源端做束搜索解码，（3）在源端句子与教师译文组成的句对上训练学生模型。

在非自回归机器翻译中，通常令自回归模型为教师模型，非自回归模型为学生模型，采用序列级知识蒸馏技术[13] 进行训练。序列级知识蒸馏技术直接将训练数据中的参考译文替换为自回归模型的翻译结果，增强了目标端译文的确定性，使非自回归模型更容易拟合目标端的概率分布，降低了多峰性带来的影响。目前，序列级知识蒸馏已经成为了非自回归机器翻译的一种标准设置，被大部分非自回归模型采用。

为了探究序列级知识蒸馏给训练数据带来的影响，Zhou 等[14] 提出了两种指标来分别表示数据集的可信度和复杂度，可信度越高表示数据集的目标端与真实分布的差异越小，复杂度越高表示目标端译文的不确定性越强，即多峰性问题越严重。据此，Zhou 等[14] 对序列级知识蒸馏所起的作用做了定量分析，发现它以牺牲目标端数据可信度的代价降低了数据的复杂度，教师模型越强，则蒸馏数据集的可信度和复杂度越高，与原始的训练数据越接近。因此，教师模型也不是越强越好，非自回归的学生模型与自回归的教师模型能力较匹配时才能发挥最好的蒸馏效果。

Sun 等[15] 进一步对序列级知识蒸馏方法做改进，在知识蒸馏的过程中应用 EM 算法[16,17] 来迭代地优化教师模型与学生模型。在每一轮的优化中，首先用自回归模型构建蒸馏数据集，对非自回归模型进行训练，再根据非自回归模型的状态构建数据集来训练自回归模型。在这一过程中，教师与学生的匹配程度逐渐提升，从而能更好地训练学生模型。

Ding 等[18] 指出知识蒸馏存在错误传播的缺陷，自回归教师模型预测译文中的错误会随着知识蒸馏传播到非自回归学生模型中。具体地，他们发现自回归模型在低频词的生成上存在一定缺陷，使蒸馏数据中高低频词汇的比例与原始数据有较大偏差。对此，Ding 等[18] 将原始数据的分布作为先验补充到损失函数中，令学生模型在不过于偏离原始分布的前提下从教师模型中学习。

基于序列级知识蒸馏的这类方法能有效地减小多峰性问题带来的影响，大幅提升了非自回归模型的翻译性能。然而，知识蒸馏方法依赖于从自回归教师模型中提取的知识，这使得训练过程更加复杂，也使非自回归模型的性能上限由自回归教师模型决定，制约了非自回归模型进一步发展的潜力，这是知识蒸馏方法的主要缺陷。

2.2 增强表达能力

为了能并行生成整句译文，非自回归模型对目标端的概率分布做了条件独立性假设，仅依赖于源端文本来生成所有译文单词。然而，如1.2.3节所示，机器翻译任务的目标端分布存在多峰性，译文的前后词之间存在较强的依赖关系，并不满足条件独立性假设。非自回归模型无法捕捉到目标端各部分之间的依赖关

系，导致模型在理论上无法通过参数估计拟合机器翻译训练数据的分布，难以生成正确的译文。如果能增强非自回归模型的表达能力，使其能够在保留并行生成能力的同时建模目标端各部分之间的依赖关系，就能让模型更好地拟合多峰的数据分布，提升模型的翻译性能。这方面的工作主要包括基于条件随机场[19]、CTC[20]、有向无环图[21] 增强的非自回归模型。

Sun 等[19] 在建模每步翻译概率的同时引入了条件随机场[22] 来建模译文中相邻词之间的依赖关系。相比于基础的非自回归模型，条件随机场模型具备更强的表达能力，能够生成更加流畅的译文。相比于自回归模型，条件随机场模型的优势是能进行精确的解码，通过维特比算法[23] 找出概率最大的译文，并且在解码速度上也大幅优于自回归解码。

CTC[20] 最早是语音识别任务中的技术，要求模型在每个输入的位置都预测输出结果。由于输入的语音序列通常远比目标的文本序列长，CTC 允许模型生成空白词与重复词，并删除空白与重复来得到最终解码结果。Libovický 等[24]，Saharia 等[25] 在非自回归机器翻译中引入 CTC 方法，通过增大解码器长度来提升模型的表达能力，并允许模型生成空白词与重复词，使其能在解码时动态调整译文长度。CTC 方法在仅进行单轮解码时就展现出了优越的性能，吸引了众多研究者的关注[26–29]，目前已成为非自回归模型的主流结构之一。

不久前，Huang 等[21] 提出了基于有向无环图的非自回归模型结构，称为 DA-Transformer。与 CTC 类似，DA-Transformer 通过增大解码器长度来提升模型的表达能力，并从解码器中选择一条解码路径来生成最终译文。不同点在于 CTC 通过空白词和重复词的概率来隐式地建模解码路径，而 DA-Transformer 显式地将译文的概率分解为路径概率和发射概率，并使用有向无环图建模翻译路径的概率，使模型能通过多条翻译路径表达多峰的目标端分布。这种结构不需使用知识蒸馏就能达到接近自回归模型的翻译性能，吸引了众多研究者的关注[30–32]。

上述方法均能增强非自回归模型的表达能力，但也有一定的缺陷。为了增强表达能力，这类方法都需要模型输出更多的信息来更好地表示概率分布，例如条件随机场需要建模相邻词之间的转移矩阵，CTC 与 DA-Transformer 需要增大解码器的长度，但这也增大了模型的计算代价。另外，尽管模型的表达能力能通过输出更多信息得到提升，但仍与译文空间的指数级大小有很大的差距，难以通过这种方式来追平自回归模型的表达能力。

除此之外，另一种增强非自回归模型表达能力的方式是进行大规模的预训练。Qi 等[33] 将自回归和非自回归生成统一到同一模型框架下并进行大规模的预训练，从而增强非自回归模型在各种任务下的生成能力。Jiang 等[34] 采用混合训练的方式来对掩码生成模型[35] 进行大规模预训练，减小了其与自回归预训练模型的性能差距。Huang 等[36] 对基于有向无环图的 DA-Transformer 结构进行了大规模预训练，在各个任务上都达到了媲美自回归预训练模型的水平。

2.3 引入隐变量

除上述对模型结构的直接改进外，在非自回归模型中引入隐变量也可以增强模型的表达能力。隐变量代表一种不能直接被观测或测量到的变量，可以在模型的生成过程中作为中间状态来帮助模型生成更加连贯和准确的结果。在机器翻译中，有些词和短语可能有多种翻译方式，翻译的语序也不是固定不变的，整体有较强的不确定性，导致了目标分布的多峰性。理想情况下，隐变量应该能消除翻译过程中的不确定性，不同隐变量对应不同的翻译结果，给定隐变量后的目标分布是确定性较强的单峰分布。在非自回归模型中引入隐变量的方法主要可以分为两种，一种是基于成熟的深度学习算法建模隐变量，一种是基于机器翻译的任务特征人工设计隐变量。

在机器翻译任务中，非自回归模型主要使用变分自编码器技术（VAE[37]）和矢量量化技术（VQ-VAE[38]）来分别建模连续和离散的隐变量。Gu 等[26], Shu 等[39] 使用变分自编码器将输入数据映射到低维的连续隐变量空间中，并用解码器将隐变量映射到译文空间中。在训练时，模型从后验分布中采样隐变量，因此模型的输出较靠近参考译文，使损失函数更准确。在解码时，模型则从先验分布中采样隐变量并生成对应译文。此外，Ma 等[40] 将流模型技术[41] 与变分自编码器结合，使模型能够建模更复杂的隐变量先验。Kaiser 等[42], Roy 等[43], Bao 等[44,45] 使用矢量量化技术将输入数据映射到低维的离散隐变量空间中。这类方法维护了一个固定大小的隐变量表，每次从表中找出与参考译文的表示最接近的隐变量，并据此来动态更新隐变量表。模型需要能根据源端信息预测隐变量，并从隐变量中重建出译文。

另外一种引入隐变量的方式是基于机器翻译的任务特征人工设计隐变量。在最早的非自回归机器翻译模型中，Gu 等[8] 就为机器翻译任务设计了基于繁衍率的隐变量，繁衍率代表了每个源端词对应的目标端词数。在训练时，首先使用词对齐工具得到原文对参考译文的繁衍率，解码器再根据繁衍率生成翻译结果。在测试时，模型则直接预测繁衍率并生成译文。类似地，Ran 等[46], Song 等[47] 将源端与目标端之间的词对齐作为隐变量，词对齐信息可以理解为是比繁衍率更强的隐变量，不仅考虑了每个源端词对应的目标端词数，还考虑了词语间相对位置的变化。模型在训练时利用词对齐信息对源端表示做重排序，只需生成与源端词序较一致的译文。在测试时，模型则需要先预测词对齐信息，再根据词对齐生成对应的译文。另外，Akoury 等[48], Liu 等[49] 也分别设计了句法结构信息和词法结构信息的隐变量，

隐变量方法的一个难点是控制隐变量所携带信息的尺度。如果隐变量携带的信息较少，则它难以有效消除翻译过程中的不确定性，损失函数仍然是较不准确的。例如，在给定繁衍率信息后，源句 ‘‘我今天下午吃了披萨’’ 仍有 “I ate pizza this afternoon’’ 与 “this afternoon I ate pizza’’ 两种可能的翻译结果，多峰性问题仍然存在。如果隐变量携带的信息较多，能有效地消除多峰性，则模型训练时的损失函数会变得较为准确。然而，模型在测试时只能根据源端输入来预测先

验隐变量，如果隐变量携带的信息过多，则模型在测试时难以准确预测隐变量，导致训练与测试之间存在较大的差异，同样会影响模型性能。一个极端的例子是将完整的参考译文作为隐变量，此时对隐变量的预测就等价于原来对译文的预测，并没有减小训练的难度。对于这个难点，Qian 等[50] 提出一种预解码的解决方案来控制隐变量携带信息的多少，能一定程度上减轻隐变量方法的这一缺陷。Qian 等[50] 直接对参考译文施加一定的掩码来作为解码器的输入，未被掩盖的部分就可以视为隐变量。在正式训练前，模型先进行一次预解码，观察输出结果与参考译文之间的差异，从而确定当前样本的多峰性强弱。若差异较大，说明样本的多峰性较强，则使用较小的掩码比例，令隐变量携带较多的信息来减小多峰性，反之使用较大的掩码比例进行训练。

2.4　设计解码策略

非自回归模型的条件独立性假设使其能够并行解码整句译文，大幅提升了解码速度，但也牺牲了一部分翻译质量。因此，一些研究者开始探索速度稍慢但能提高翻译质量的解码策略，包括半自回归解码、推测式解码、插入式解码、迭代式解码等。

自回归模型一次只生成一个词，非自回归模型一次生成整句译文，介于两者之间的就是半自回归神经机器翻译模型[51]。半自回归模型每次生成固定数目的一组单词，因此解码速度介于自回归和非自回归模型之间，翻译质量显著优于最初的非自回归模型。类似地，Ran 等[52] 提出了多线程的解码策略，将整个目标端序列均匀切分为多个子序列，每个子序列同时进行解码。Wang 等[53] 提出了混合自回归与非自回归的解码策略，首先由自回归模型生成目标序列的部分词语，再由非自回归模型填充剩余的词。这类模型每次能生成固定数目的单词，能将解码速度提升常数倍，但解码时间仍然与序列长度线性相关。

Stern 等[54], Xia 等[55] 提出了推测式解码策略，结合自回归模型与非自回归模型来解码译文，将每步的解码分为预测、验证、接受三个阶段，首先令非自回归模型解码出多个后续的单词，再用自回归模型验证这些词是否符合贪心搜索的规则，最后将符合贪心规则的部分作为当前步的解码结果。这种解码方式能够保证最终的译文与自回归模型的贪心搜索相同，同时所需的解码步数远小于自回归模型。

Stern 等[56], Chan 等[57] 提出了插入式解码策略，不是像自回归模型一样顺序地从左到右生成译文，而是每次选择多个合适的插入位置，并行地在这些位置上插入单词。译文长度为 T 时，插入式解码需要的解码轮数约为 $\log_2 T$，将解码的时间复杂度从线性降低到了对数。

研究最深入的解码策略是迭代式的非自回归解码，由 Lee 等[58] 最早提出，解码器的输出在下一轮迭代中作为输入反馈到解码器的输入中，通过多轮迭代优化模型输出，得到最终的翻译结果。随后，Ghazvininejad 等[35] 提出了基于掩码-预测的解码方式，在训练时按比例掩盖一部分的参考译文，模型能够观察到

未被掩盖的部分，并对掩码部分进行预测。在解码时，模型每次预测完译文后按比例掩盖掉模型不够自信的预测结果，在下一轮重新预测。这种解码方式使非自回归模型首次达到了与自回归模型相当的翻译性能，引发了后续研究者的广泛关注[59–61]。Gu 等[62] 提出了 Levenshtein Transformer 模型，通过多轮的插入和删除操作对翻译结果进行迭代优化。Geng 等[63] 提出基于重写的迭代方式，每轮迭代先检测出当前翻译结果的出错位置，再对出错位置进行重写。Qian 等[31], Gao 等[64] 引入了扩散模型的技术，通过对模型输出做多步去噪来得到最终译文。

2.5 改进训练方法

在非自回归模型研究的早期阶段，研究人员对非自回归模型的训练方法关注较少，仅有 Wang 等[65] 针对非自回归模型的重复翻译和漏翻译问题改进训练方法，引入基于相邻位置表示相似度和重建源端文本的正则化项来辅助模型训练。在我们指出非自回归模型损失函数不准确的问题之后[66–68]，研究者们关注到了非自回归模型在训练方法上的缺陷，并开始重视对模型训练方法的改进。

针对模型输出与参考译文可能不对齐的问题，Ghazvininejad 等[69] 提出对齐的交叉熵损失，应用动态规划算法找出模型输出与参考译文间的最优单调对齐，基于最优的对齐来计算损失函数并训练模型。Du 等[70] 进一步扩大了对齐的搜索空间，应用匈牙利算法找出模型输出与参考译文之间的最优非单调对齐，根据找出的对齐调整参考译文的词序后训练模型。

近期，Huang 等[11] 对非自回归模型的训练做了理论上的分析，指出非自回归模型的表达能力存在理论上界，无法完美拟合训练数据的分布。基于此，Huang 等[11] 提出了基于代理分布的训练框架，将模型的损失函数分成两部分，第一部分为预测分布与代理分布的 KL 距离，即令非自回归模型去拟合代理分布，第二部分为代理分布与数据分布之间的差异。Huang 等[11] 进一步将非自回归模型过去的一些训练方法统一到了这一框架中，例如上述两种基于对齐的交叉熵损失。

我们在非自回归模型训练方法上的工作主要包括序列级训练[66]、n 元组匹配[67,71]、动态参考译文[72,73] 三部分。在序列级训练方面，后续有研究者对序列级评估指标的优化做了进一步研究。Guo 等[74] 将非自回归模型的训练看作是多智能体的强化学习问题，并应用相关算法来使模型优化序列级评估指标。Li 等[75] 提出了多粒度优化算法，收集不同掩码粒度下的模型行为并计算序列级评估指标，集成不同粒度下的反馈信号来训练模型。

在 n 元组匹配方面，近期也有研究者基于这一准则来设计训练目标。Liu 等[76] 通过卷积操作来高效计算 n 元组的匹配数目。并基于 n 元组匹配设计了不受编辑影响的序列损失。Du 等[77] 将词级别的非单调对齐扩展到 n 元组级别，在 n 元组层面上应用匈牙利算法搜索模型输出与参考译文之间的最优匹配，并基于最优的 n 元组匹配来计算损失函数。Ma 等[32] 考虑基于有向无环图的非自回归模型结构 DA-Transformer，利用翻译路径的马尔可夫性推导出了计算 n 元组数目的高效算法，从而训练 DA-Transformer 直接优化 n 元组匹配的准确率。

2.6　本章小结

在本章中，我们对非自回归神经机器翻译的研究现状做了详细介绍，包括知识蒸馏、增强表达能力、引入隐变量、设计解码策略、改进训练方法这五种改进思路。知识蒸馏方法能简化目标端的数据分布，在数据层面上减小多峰性，但也会限制模型的性能上界。我们可以通过令模型输出更多信息来增强非自回归模型的表达能力，使其能够更好地拟合多峰的数据分布，但这种方式理论上仍难以拟合指数级大小下的译文空间概率分布。在非自回归模型中引入隐变量可以减小翻译过程中的不确定性，使目标分布变为确定性较强的单峰分布，但会使得模型的训练与测试之间存在较大的差异。另外，也可以通过设计其他的解码策略来提高模型的翻译质量，但要以在一定程度上降低解码速度为代价。最后，我们可以不局限于极大似然估计的训练范式，设计更准确、更适用于非自回归模型的训练方法来克服多峰性问题带来的缺陷。在表2-1中，我们简单地总结了当前对非自回归机器翻译模型的主流改进方法。

表 2-1 非自回归机器翻译的主流改进方法。

Table 2-1 Mainstream methods of improving non-autoregressive machine translation.

知识蒸馏	通过知识蒸馏减小训练数据的多峰性
增强表达能力	设计表达能力更强的非自回归模型结构
引入隐变量	使用隐变量来消除翻译过程中的不确定性
设计解码策略	设计其他解码策略，通过多轮解码改善翻译性能
改进训练方法	将极大似然估计更换为其他训练方法

第 3 章　基于强化学习的序列级训练

3.1　引言

机器翻译是自然语言处理领域的一个重要研究课题，自计算机诞生以来就受到研究者们的广泛关注。随着深度学习的方法逐渐成熟，基于深度神经网络的模型在机器翻译上取得了显著的成绩[1,5–7,78]。神经机器翻译模型通常采用编码器-解码器框架，编码器网络将源句编码为向量表示，解码器网络再从源端编码的表示中解码出目标端句子。目前，神经机器翻译模型主要是以自回归的方式对从源端到目标端的翻译概率进行建模，每一步的译文单词生成都依赖于之前的翻译结果。因此，模型预测的单词要反过来作为输入反馈给解码器，这使得模型只能逐词生成译文。这种逐词生成的解码方式使得神经机器翻译模型的解码速度较慢，尤其是在生成长句时具有较高的延迟。

Gu 等[8] 提出非自回归神经机器翻译模型（Non-Autoregressive Neural Machine Translation，NAT）来降低翻译的延迟。非自回归模型对目标端的概率分布做了条件独立性假设，能够仅根据源端文本并行地生成所有译文单词，因此在解码速度上相对自回归模型有显著优势。然而，非自回归模型仍然使用词级交叉熵作为损失函数，而词级损失无法准确地评估非自回归模型的输出，这对非自回归模型的翻译质量造成了较大影响。在机器翻译任务中，同一原文可能有多种不同的译文，这些译文在用词、词序等方面有差异，但能表达相同的语义。然而，交叉熵损失没有考虑这种现象，仅在模型输出与参考译文严格对齐时才能正确评估模型的输出。一旦非自回归模型没有按参考译文的形式输出翻译结果，交叉熵损失就会将没有对齐的位置都看作是错误的预测结果，给出较高的损失值。因此，交叉熵损失与实际的翻译质量之间的相关性较弱，这给非自回归模型的训练带来了较高噪声，对模型性能有较强的负面影响。

如图3-1所示，尽管模型的输出 “I have to get up and start working.’’ 与参考译文 “I must get up and start working now.’’ 有着相似的语义，词级交叉熵损失还是会因为它没有与参考译文严格对齐而给出较高的惩罚。在交叉熵损失的监督下，模型输出的译文可能会被纠正为 “I have to up up start start working.’’，纠正后的模型输出有更小的损失，但实际的翻译质量反而更差。造成这种现象的本质原因是交叉熵损失独立地评估每个位置的生成质量，没有考虑序列内的依赖关系，使非自回归模型仅关注局部的正确性而忽视了整体的翻译质量。

针对词级交叉熵损失无法准确评估模型输出的问题，我们提出对非自回归模型做序列级训练，从序列层面整体评估模型的输出结果并训练模型。具体地，我们令模型直接来优化机器翻译中翻译质量的评估指标（例如 BLEU[79]、GLEU[78]、ROUGE[80] 等）。这类指标通常都是离散的，无法直接用于训练模型。因此，我们采用基于强化学习的训练方式，将待优化的指标作为奖赏，令模型以最大化期望奖赏为训练目标，并应用强化学习算法来估计训练目标的梯度。这种训练方式

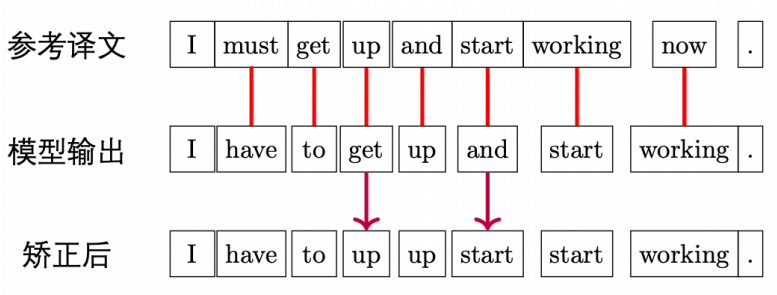


图 3-1 非自回归模型输出与参考译文没有对齐的例子。

Figure 3-1 An example of NAT output that is not aligned with the reference sentence.

的主要缺陷是梯度估计的方差较高，导致模型在训练时的噪声较大，训练过程不稳定。我们利用非自回归模型并行生成的特性，再结合机器翻译评估指标的性质，针对性地改良了强化学习算法，降低了梯度估计的方差，使非自回归模型能够准确、稳定地优化序列级评估指标。

我们在多个机器翻译数据集（WMT14 En↔De，WMT16 En↔Ro）上进行了实验来验证所提方法的有效性。实验结果表明，序列级训练能显著改善非自回归模型的翻译质量，使其在性能接近自回归模型的同时仍能保持着相对自回归模型十倍以上的解码加速。

3.2 自回归模型的序列级训练

用序列级评估指标训练神经机器翻译模型的技术在自回归模型中已经有了广泛研究[78,81–84]。机器翻译的评估指标通常都是离散的，无法直接用于训练模型。因此，研究者们一般采用强化学习的训练框架，将待优化的指标作为奖赏，令模型以最大化期望奖赏为训练目标：

$$\mathcal{L}(\theta) = -\sum_{Y} p(Y|X,\theta) \cdot r(Y), \tag{3-1}$$

其中 Y 表示目标端的译文序列，$r(Y)$ 表示该译文的奖赏，通常使用的奖赏是译文 Y 与参考译文间的 BLEU 值。由于译文空间的大小是指数级的，我们无法通过直接遍历所有译文来计算上式，通常是运用 REINFORCE 算法[85] 来估计上式的梯度：

$$\begin{aligned} \nabla_\theta \mathcal{L}(\theta) &= -\sum_{Y} p(Y|X,\theta) \cdot \frac{\nabla_\theta p(Y|X,\theta)}{p(Y|X,\theta)} \cdot r(Y) \\ &= -\mathop{\mathbb{E}}_{Y}[\nabla_\theta \log(p(Y|X,\theta)) \cdot r(Y)] \\ &= -\mathop{\mathbb{E}}_{Y}[\sum_{t=1}^{T} \nabla_\theta \log(p(y_t|X,y_{<t},\theta)) \cdot r(Y)]. \end{aligned} \tag{3-2}$$

上式指出，我们可以按以下步骤对损失函数的梯度做无偏估计：

- 给定原文 X，从模型预测的概率分布 $p(Y|X,\theta)$ 中采样出一句译文 Y。

- 计算译文 Y 的奖赏值 $r(Y)$。
- 求得对损失函数梯度的无偏估计 $\nabla_\theta \log(p(Y|X,\theta)) \cdot r(Y)$，用估计出的梯度训练模型。

直观上看，上述步骤用采样出译文的对数似然损失来训练模型，并用它的奖赏来作为损失的权重。奖赏较高表示译文的翻译质量较好，因此给损失较高的权重，反之则降低损失的权重。这种方法的优点是它对梯度的估计是无偏的，但之前的研究也显示这样的梯度估计方法有较高的估计方差，使得这类基于强化学习的训练过程有较大的噪音。

这种方法的估计方差主要来源于它的两个缺陷。第一，译文通常由数十个词构成，现在的方法对所有词一视同仁，均用整句的奖赏来评估每个词的好坏。然而，译文翻译质量较低时，通常是有少数几个位置上的翻译出了错，而一视同仁的训练方式会把这些错误牵连到其他输出正确结果的位置，使得训练中的噪声较大。第二，机器翻译中的评估指标通常都被定义为正数，因此当前的算法总是倾向于提高采样出译文的概率，不会对低质量的译文施加惩罚，导致训练较为低效。目前也有一些解决方案来降低梯度估计的方差，如对奖赏函数进行调整[86]和引入基线奖赏值[87]。然而，这类方法通常都需要让模型采样多组结果来做对比，而自回归模型的逐词解码导致其采样代价非常高，使这类技术的应用存在困难。

3.3　非自回归模型的序列级训练

本节将详细介绍对非自回归模型的序列级训练方法。由于词级交叉熵损失无法正确评估模型的输出，我们希望能让模型直接优化序列级的训练目标，如 BLEU[79]、GLEU[78]、ROUGE[80] 等机器翻译的评估指标。这类指标通常都是离散的，无法直接用于训练模型，因此我们也效仿自回归模型中的序列级训练方法，令模型以最大化期望奖赏为训练目标：

$$\mathcal{L}(\theta) = -\sum_{Y=y_{1:T}} p(Y|X,\theta) \cdot r(Y). \tag{3-3}$$

上式中，$r(Y)$ 表示机器翻译的评价指标，例如 BLEU 值。这类指标通常是基于参考译文 $\hat{Y}$ 和模型输出 Y 来评估翻译质量的，完整的写法是 $r(\hat{Y},Y)$，在参考译文确定的情况下，我们可以简写为 $r(Y)$。虽然训练目标从词级变到了序列级，但这种训练目标不要求模型去拟合数据分布，反而降低了非自回归模型的学习难度。从理论上看，下面的定理保证了最优的模型分布为单峰的 0-1 分布，因此非自回归模型有能力去最大化上式的期望奖赏。

定理 3.1. *在最小化期望损失的意义下，模型的最优分布为满足 $p(Y^*|X,\theta)=1$ 的 0-1 分布，其中 $Y^* = argmax_Y \mathbb{E}_{\hat{Y}\sim p_{data}} r(\hat{Y},Y)$，$p_{data}$ 表示训练数据中译文的分布。*

证明.

$$\begin{aligned}
\mathop{\mathbb{E}}_{\hat{Y}\sim p_{data}}[\mathcal{L}(\theta)] &= -\sum_{\hat{Y}} p_{data}(\hat{Y}|X)\sum_{Y} p(Y|X,\theta)\cdot r(\hat{Y},Y) \\
&= -\sum_{Y} p(Y|X,\theta)\sum_{\hat{Y}} p_{data}(\hat{Y}|X)\cdot r(\hat{Y},Y) \\
&\geq -\operatorname{argmax}_{Y}\sum_{\hat{Y}} p_{data}(\hat{Y}|X)\cdot r(\hat{Y},Y) \\
&= -\operatorname{argmax}_{Y}\mathop{\mathbb{E}}_{\hat{Y}\sim p_{data}} r(\hat{Y},Y).
\end{aligned}$$

□

上述定理保证了模型有足够能力去优化式3-3的损失函数。然而，译文空间的大小是指数级的，我们无法通过直接遍历所有译文来计算这一损失。对此，我们采用强化学习的训练框架，尝试找出对如下梯度的无偏估计：

$$\begin{aligned}
\nabla_{\theta}\mathcal{L}(\theta) &= -\sum_{Y}\nabla_{\theta} p(Y|X,\theta)\cdot r(Y) \\
&= -\sum_{Y}\nabla_{\theta}\prod_{t=1}^{T} p_t(y_t|X,\theta)\cdot r(Y).
\end{aligned} \tag{3-4}$$

在下面的内容中，将首先介绍基础的训练方法，直接运用 REINFORCE 算法[85] 来估计上式，命名为 Reinforce-Base。随后，我们对奖赏函数进行调整，不使用整句的奖赏来训练模型，而是使不同位置的预测结果能有独立的词级奖赏，该方法命名为 Reinforce-Step。基于此，我们对每步高概率词汇的梯度做更精确的计算，从而进一步降低估计的方差，该方法命名为 Reinforce-Topk。最后，我们利用机器翻译中评估指标的性质，对奖赏函数做了合理的假设，提出 Traverse-Ref 算法来通过对参考译文的遍历更准确地计算梯度。我们也分析了这一系列算法的时间复杂度。

3.3.1 Reinforce-Base 算法

首先，我们可以不考虑非自回归模型的特性，直接运用 REINFORCE 算法来估计式3-4：

$$\nabla_{\theta}\mathcal{L}(\theta) = -\mathop{\mathbb{E}}_{Y}[\sum_{t=1}^{T}\nabla_{\theta}\log(p_t(y_t|X,\theta))\cdot r(Y)]. \tag{3-5}$$

根据上式，模型首先对原文进行编码并生成目标端的概率分布，并从分布中采样一个句子 Y。随后，我们计算译文 Y 的对数似然损失和奖赏值，将奖赏作为损失函数的权重，求得对损失函数梯度的无偏估计 $\nabla_{\theta}\log(p(Y|X,\theta))\cdot r(Y)$。算法1描述了 Reinforce-Base 方法的具体训练过程。

3.3.2 Reinforce-Step 算法

上面的方法通过采样一句译文来求得对整体梯度的无偏估计，但由于译文的搜索空间是指数级的，这种估计方法的缺陷是有较高的估计方差。直观上看，

算法 1 Reinforce-Base

Input: probability distribution $p(\cdot|X,\theta)$), length T

Output: the estimate of $\nabla_\theta \mathcal{L}(\theta)$

$\nabla_\theta \mathcal{L}(\theta) = 0$
for $t = 1$ **to** T **do**
    sample y_t from $p_t(\cdot|X,\theta)$)
    $\nabla_\theta \mathcal{L}(\theta)$ += $-\nabla_\theta \log(p_t(y_t|X,\theta))$
end for
construct $Y = \{y_1, ..., y_T\}$, calculate $r(Y)$
$\nabla_\theta \mathcal{L}(\theta) = r(Y) \cdot \nabla_\theta \mathcal{L}(\theta)$
return $\nabla_\theta \mathcal{L}(\theta)$

非自回归模型不同时间步的预测结果是相互独立的，每步的奖赏也应只取决于当前的预测结果 y_t，而非与其他步共享整句奖赏。如果能独立考虑每个时间步，对每步的梯度做独立的估计，搜索空间也就能从指数级降低到词表大小，从而减小梯度估计的方差。为实现这一设想，我们利用模型并行生成的特性，将式3-4的整句梯度散布到每个时间步上，每步的预测结果都有其独立的词级奖赏。具体地，我们证明了如下定理：

定理 3.2.

$$\nabla_\theta \mathcal{L}(\theta) = -\sum_{t=1}^{T} \sum_{y_t} \nabla_\theta p_t(y_t|X,\theta) \cdot r(y_t), \tag{3-6}$$

$$\text{其中} \quad r(y_t) = \mathop{\mathbb{E}}_{y_{1:t-1}} \mathop{\mathbb{E}}_{y_{t+1:T}} r(Y). \tag{3-7}$$

证明.

$$\begin{aligned}
\nabla_\theta \mathcal{L}(\theta) &= -\sum_{Y} \nabla_\theta \prod_{t=1}^{T} p_t(y_t|X,\theta) \cdot r(Y) \\
&= -\sum_{Y} \sum_{t=1}^{T} \nabla_\theta p_t(y_t|X,\theta) \cdot \prod_{i=1}^{t-1} p_i(y_i|X,\theta) \cdot \prod_{j=t+1}^{T} p_j(y_j|X,\theta) \cdot r(Y) \\
&= -\sum_{t=1}^{T} \sum_{Y} \nabla_\theta p_t(y_t|X,\theta) \cdot \prod_{i=1}^{t-1} p_i(y_i|X,\theta) \cdot \prod_{j=t+1}^{T} p_j(y_j|X,\theta) \cdot r(Y) \\
&= -\sum_{t=1}^{T} \sum_{y_t} \nabla_\theta p_t(y_t|X,\theta) \cdot \sum_{y_{1:t-1}} \sum_{y_{t+1:T}} \prod_{i=1}^{t-1} p_i(y_i|X,\theta) \cdot \prod_{j=t+1}^{T} p_j(y_j|X,\theta) \cdot r(Y) \\
&= -\sum_{t=1}^{T} \sum_{y_t} \nabla_\theta p_t(y_t|X,\theta) \cdot \mathop{\mathbb{E}}_{y_{1:t-1}} \mathop{\mathbb{E}}_{y_{t+1:T}} r(Y) \\
&= -\sum_{t=1}^{T} \sum_{y_t} \nabla_\theta p_t(y_t|X,\theta) \cdot r(y_t).
\end{aligned}$$

□

算法 2 Estimation of step reward $r(y_t)$

Input: probability distribution $p(\cdot|X,\theta)$), step t, word y_t, length T, sampling times n

Output: the estimate of $r(y_t)$

$r = 0$
for $i = 1$ **to** n **do**
    sample $y_{1:t-1}$, $y_{t+1:T}$ from $p(\cdot|X,\theta)$)
    construct $Y = \{y_{1:t-1}, y_t, y_{t+1:T}\}$, calculate $r(Y)$
    r += $r(Y)$
end for
$r = r/n$
return r

算法 3 Reinforce-Step

Input: probability distribution $p(\cdot|X,\theta)$), length T, sampling times n

Output: the estimate of $\nabla_\theta \mathcal{L}(\theta)$

$\nabla_\theta \mathcal{L}(\theta) = 0$
for $t = 1$ **to** T **do**
    sample y_t from $p_t(\cdot|X,\theta)$)
    estimate $r(y_t)$ by algorithm 2 with sampling times n
    $\nabla_\theta \mathcal{L}(\theta)$ += $-r(y_t) \cdot \nabla_\theta \log(p_t(y_t|X,\theta))$
end for
return $\nabla_\theta \mathcal{L}(\theta)$

如上所示，我们将整句梯度分解为每个时间步的梯度，每步的梯度为词表上所有词的概率梯度与其词级奖赏的加权求和，词级奖赏定义为固定当前步预测结果 y_t 后整句奖赏的期望。随后，我们仍能运用 REINFORCE 算法来估计上式：

$$\begin{aligned}
\nabla_\theta \mathcal{L}(\theta) &= -\sum_{t=1}^{T}\sum_{y_t} \nabla_\theta p_t(y_t|X,\theta) \cdot r(y_t) \\
&= -\sum_{t=1}^{T}\sum_{y_t} p_t(y_t|X,\theta) \cdot \frac{\nabla_\theta p_t(y_t|X,\theta)}{p_t(y_t|X,\theta)} \cdot r(y_t) \\
&= -\sum_{t=1}^{T}\mathop{\mathbb{E}}_{y_t}[\nabla_\theta \log(p_t(y_t|X,\theta)) \cdot r(y_t)].
\end{aligned} \tag{3-8}$$

上式与式3-5的唯一区别是句级奖赏 $r(Y)$ 变成了词级奖赏 $r(y_t)$，我们将这种估计方法称为 Reinforce-Step。词级奖赏是按期望形式定义的，因此可以用蒙特卡洛采样法估计，首先固定住时间步 t 的预测结果为 y_t，再从其他位置的概率分布中进行采样，计算采样出的句子的奖赏。重复该过程 n 次后，取 n 次所得奖赏的平均值，就是对词级奖赏 $r(y_t)$ 的估计结果。算法2具体描述了对词级奖赏的估计方法。基于此，算法3描述了 Reinforce-Step 方法的训练过程。

3.3.3　Reinforce-Topk 算法

Reinforce-Step 方法将整句梯度分解到每个时间步上，每步的梯度为词表上所有词的概率梯度与其词级奖赏的加权求和，随后运用 REINFORCE 算法采样一个词来估计整个词表上的梯度。然而，机器翻译中词表通常是数万词的大小，仅采样一个词的估计方式仍会带来较大的估计方差。如果我们可以遍历整个词表直接计算式3-6，就能最大程度下减小估计的方差。然而，由于词表大小为数万的量级，这会使算法的计算成本大幅上升。因此，我们考虑一个折中的方案，仅对词表中的一个子集做精确的遍历，再通过采样来估计剩余部分的梯度。对于这个要遍历的子集，它应该仅包含梯度较高的、对梯度估计较重要的词，从而能在保持较低训练成本的前提下有效减小梯度估计的方差。

在机器翻译中，每一步的概率分布通常都是较集中的分布，几个译文单词占据概率分布的绝大部分，其他词的概率合起来一般也不会太高。另外，softmax 函数的性质保证了概率很小的词对应的概率梯度也很小。据此，我们认为在机器翻译中，概率高的词就是对梯度估计较重要的词。将每个时间步 t 的概率分布按概率大小排序后，我们可以取出概率大小排前 k 的词，将该集合称为 T_k^t。如下所示，我们令 P_k^t 表示集合 T_k^t 中所有词的概率和，令 $\tilde{p}$ 表示在移除这些词后，剩余词归一化后的概率分布。

$$P_k^t = \sum_{y_t \in T_k^t} p_t(y_t|X,\theta),\ \ \tilde{p}_t(y_t|X,\theta) = \begin{cases} 0, & y \in T_k^t \\ \frac{p_t(y_t|X,\theta)}{1-P_k^t}, & y \notin T_k^t \end{cases}. \tag{3-9}$$

我们将词表分为两部分来计算式3-6。对于 T_k^T 中的词，我们进行遍历并累积它们的梯度。对剩余的词，我们从归一化分布 $\tilde{p}$ 中进行一次采样来估计它们的梯度。算法4描述了 Reinforce-Topk 方法的训练过程。下面的定理表明，这种方式仍能得到对梯度的无偏估计：

定理 3.3.

$$\nabla_\theta \mathcal{L}(\theta) = -\sum_{t=1}^{T}\Big(\sum_{y_t \in T_k^t} \nabla_\theta p_t(y_t|X,\theta)\cdot r(y_t) + (1-P_k^t)\cdot \mathop{\mathbb{E}}_{y_t\sim\tilde{p}}[\nabla_\theta \log(p_t(y_t|X,\theta))\cdot r(y_t)]\Big). \tag{3-10}$$

算法 4 Reinforce-Topk

Input: probability distribution $p(\cdot|X,\theta)$, topk size k, length T, sampling times n

Output: the estimate of $\nabla_\theta \mathcal{L}(\theta)$

$\nabla_\theta \mathcal{L}(\theta) = 0$
for $t = 1$ **to** T **do**
  T_k^t = {words ranking top-k in $p_t(\cdot|X,\theta)$}
  $\tilde{p}_t = p_t$, $P_k^t = 0$
  for y_t in T_k^t **do**
    estimate $r(y_t)$ by algorithm 2 with sampling times n
    $\nabla_\theta \mathcal{L}(\theta)$ += $-\nabla_\theta p_t(y_t|X,\theta) \cdot r(y_t)$
    $\tilde{p}_t(y_t|X,\theta) = 0$, P_k += $p_t(y_t|X,\theta)$
  end for
  $\tilde{p}_t = \tilde{p}_t/(1-P_k)$, sample y_t from $\tilde{p}_t(\cdot|X,\theta)$
  estimate $r(y_t)$ by algorithm 2 with sampling times n
  $\nabla_\theta \mathcal{L}(\theta)$ += $-(1-P_k) \cdot \nabla_\theta \log(p_t(y_t|X,\theta)) \cdot r(y_t)$
end for
return $\nabla_\theta \mathcal{L}(\theta)$

证明.

$$
\begin{aligned}
\nabla_\theta L_\theta &= -\sum_{t=1}^{T}\sum_{y_t} \nabla_\theta p_t(y_t|X,\theta)\cdot r(y_t) \\
&= -\sum_{t=1}^{T}\Big(\sum_{y_t\in T_k^t} \nabla_\theta p_t(y_t|X,\theta)\cdot r(y_t) + \sum_{y_t\notin T_k^t} \nabla_\theta p_t(y_t|X,\theta)\cdot r(y_t)\Big) \\
&= -\sum_{t=1}^{T}\Big(\sum_{y_t\in T_k^t} \nabla_\theta p_t(y_t|X,\theta)\cdot r(y_t) + (1-P_k^t)\cdot \sum_{y_t\notin T_k^t} \frac{p_t(y_t|X,\theta)}{1-P_k^t} \nabla_\theta \log(p_t(y_t|X,\theta))\cdot r(y_t)\Big) \\
&= -\sum_{t=1}^{T}\Big(\sum_{y_t\in T_k^t} \nabla_\theta p_t(y_t|X,\theta)\cdot r(y_t) + (1-P_k^t)\cdot \mathop{\mathbb{E}}_{y_t\sim\tilde{p}}[\nabla_\theta \log(p_t(y_t|X,\theta))\cdot r(y_t)]\Big).
\end{aligned}
$$

□

3.3.4 Traverse-Ref 算法

由于词表较大，上述 Reinforce-Topk 方法只能选择词表的一个子集来遍历，剩余部分仍需通过采样来估计。对一般的奖赏函数，我们只能用这种方法来减小估计的误差。精确到机器翻译的任务上，结合机器翻译的主要评估指标的性质，我们发现可以在较小代价下遍历完整的词表来计算式3-6。观察机器翻译的主要评估指标（例如 BLEU[79]、GLEU[78]、ROUGE[80] 等），这些指标主要通过衡量模型输出与参考译文的匹配程度来评估翻译质量，若译文中的词或 n 元组与参考译文有较高的重合度，则认为翻译质量较高。因此，若模型输出的词不在参考

算法 5 Traverse-Ref

Input: probability distribution $p(\cdot|X,\theta)$), length T, sampling times n, target vocabulary V, reference sentence $\hat{Y}$

Output: the estimate of $\nabla_\theta \mathcal{L}(\theta)$

$\nabla_\theta \mathcal{L}(\theta) = 0$
$V_{\hat{Y}}$ = {words in the reference sentence $\hat{Y}$}
randomly choose a word w from $V \setminus V_{\hat{Y}}$
for $t = 1$ **to** T **do**
    for y_t in $V_{\hat{Y}}$ **do**
        estimate $r(y_t)$ by algorithm 2 with sampling times n
    end for
    estimate $r(w)$ by algorithm 2 with sampling times n
    for y_t in $V \setminus V_{\hat{Y}}$ **do**
        $r(y_t) = r(w)$
    end for
    for y_t in V **do**
        $\nabla_\theta \mathcal{L}(\theta)$ += $-\nabla_\theta p_t(y_t|X,\theta) \cdot r(y_t)$
    end for
end for
return $\nabla_\theta \mathcal{L}(\theta)$

译文中，则这些词一定无法与参考译文匹配上，将其替换为其他不在参考译文中的词也不会对奖赏值造成影响。我们将具备这种性质的奖赏称为匹配型奖赏，具体定义如下：

定义 3.1. 我们称奖赏函数 $r(Y)$ 为匹配型奖赏，若它满足性质：将译文 Y 中的任意不在参考译文中的词替换为其他不在参考译文中的词时，$r(Y)$ 的值保持不变。

按上述分析，机器翻译中的大多数评估指标均为匹配型奖赏。词级奖赏 $r(y_t)$ 表示固定位置 t 的预测结果为 y_t 时的期望奖赏。按匹配型奖赏的定义，对于不在参考译文中的词语 y_t，它们的词级奖赏 $r(y_t)$ 的值一定是相同的。因此，当奖赏函数是匹配型奖赏时，我们就能根据词语是否在参考译文中来将词表分为两部分。对于在参考译文内的词，我们可以遍历它们来计算式3-6。对于不在参考译文内的词，由于它们的词级奖赏是相同的，我们仅需要从中任选一个词来计算它们的词级奖赏。在机器翻译任务中，参考译文的长度一般在一百以内，远小于词表大小的上万的量级，这使得我们能将完整计算式3-6的代价减小上百倍，使计算代价降低到可承担的范围内。算法5描述了 Traverse-Ref 方法的具体流程。

3.3.5 训练流程

本章提出的训练方法需要对译文空间采样，而译文空间的大小是指数级别的。如果模型是随机初始化的，从这样的模型中很难采样出有意义的语句，奖赏值也一般没有波动，使模型难以进行有效的训练。因此，我们采用预训练再微调的训练流程，先用交叉熵损失预训练模型，再用本章提出的强化学习方法微调模型。

3.3.6 时间复杂度分析

由于非自回归模型可以并行生成整个概率分布，对分布进行采样不需要重新运行一次模型，因此本章所提的序列级训练方法不会影响模型在 GPU 上的计算代价。然而，这些方法要求我们多次计算奖赏值来更好地估计梯度，因此在 CPU 上会有一定的计算成本。我们将奖赏函数的计算次数作为时间单位，用 T 代表参考译文的长度，n 代表估计词级奖赏时的采样次数，k 代表 Reinforce-Topk 方法中的参数，在下表给出了这一系列方法的时间复杂度。一般来说，估计得更精确、估计方差越小的方法，时间复杂度也越高。

表 3-1 本章所提方法的时间复杂度。

Table 3-1 Time complexity of the proposed methods.

	Reinforce-Base	Reinforce-Step	Reinforce-Topk	Traverse-Ref
Time Complexity	$\mathcal{O}(1)$	$\mathcal{O}(nT)$	$\mathcal{O}(nkT)$	$\mathcal{O}(nT^2)$

3.4 相关工作

自回归的神经机器翻译模型通常使用 teacher forcing 算法[88] 进行训练，这种训练方法受曝光偏差问题影响[81]，即模型在训练和测试时接触到的是不同的数据分布。训练时，模型以参考译文的前序词为输入来预测后面的词，而在测试时，模型基于自身之前的翻译结果来预测下一个词。为减小曝光偏差，计划采样方法[89–91] 将模型预测结果和参考译文混合后作为模型的输入，使模型在训练时也能接触到自身的预测结果。然而，这会造成模型的输出无法与参考译文对齐，降低了词级交叉熵损失的准确性。在这种情况下，序列级评估指标能够更好地评估模型的输出结果，因此序列级训练是对曝光偏差的一个较合理的解决方案。

由于序列级的训练目标通常是不可导的，强化学习方法[85,92] 被广泛用于自回归机器翻译模型的序列级训练中。Ranzato 等[81] 首先指出了神经机器翻译中的曝光偏差问题，并提出 MIXER 算法来结合词级交叉熵损失和序列级训练目标，从而缓解曝光偏差问题。随后，Bahdanau 等[82] 提出用强化学习中的 actor-critic 方法来对神经机器翻译模型做序列级训练。He 等[9] 提出对偶学习的训练目标，将反向模型的翻译概率提供给正向模型作为奖赏。Wu 等[78] 提出了一种

更适用于句级翻译质量评价的 GLEU 指标来指导模型训练。Yu 等[93] 提出了名为 SeqGAN 的序列生成框架，将模型本身看作生成器，结合强化学习方法令其优化源于判别器的奖赏。随后，Yang 等[83], Wu 等[94] 将这种技术应用到了神经机器翻译中。Wu 等[84] 对强化学习方法在神经机器翻译中的应用做了系统的分析和研究。

除基于强化学习的方法外，也有一些其他方法能在序列层面训练神经机器翻译模型。Shen 等[95] 从最小风险训练的角度优化模型，Norouzi 等[96] 将序列级的奖赏集成到了极大似然估计的框架中，Edunov 等[97] 研究了一系列针对序列生成模型的训练目标，Ma 等[98] 提出基于词袋的训练目标来优化模型，Shao 等[99] 提出概率化 n 元组技术来将离散的序列级训练目标转化为连续形式。

如上述分析所示，自回归模型的序列级训练技术受到了研究者的广泛关注，目前已经有了较深入的研究。在非自回归模型上，这方面的研究工作仍然比较欠缺。

3.5　实验结果与分析

3.5.1　实验设置

数据集　我们在四个翻译任务上验证了所提的方法：WMT14 英语 ↔ 德语（En↔De，约 450 万个句对）、WMT16 英语 ↔ 罗马尼亚语（En↔Ro，约 61 万个句对）。对于 WMT14 英德数据集，我们用大小写敏感的 BLEU 值来评估模型的翻译质量，分别将 *newstest2013* 和 *newstest2014* 作为验证集和测试集。对于 WMT16 英罗数据集，我们用大小写不敏感的 BLEU 值来评估模型的翻译质量，分别将 *newsdev-2016* 和 *newstest-2016* 作为验证集和测试集。我们用 32K 次操作的联合 BPE 模型[100] 来切分源端和目标端词汇，并在源端和目标端共享词表。

知识蒸馏　序列级知识蒸馏[12,13] 是非自回归模型中的一项关键技术。在所有翻译任务中，我们都应用序列级知识蒸馏技术，首先训练一个自回归模型作为教师，再令它翻译一遍训练数据集的源端，将原文与译文组成句对来构建蒸馏数据集。最后，我们使用蒸馏数据集来训练非自回归模型。

基线模型　我们将基础设置的 Transformer 模型[1] 作为自回归的基线模型。非自回归的基线模型也采用 Transformer 的模型结构，我们仅对解码器的注意力掩码矩阵做修改，使解码器的每个位置都能看到所有其他位置的信息。我们采用平均拷贝的方法[8] 来构建解码器的输入。我们在编码器顶端接入一个长度预测器，以编码器隐状态为输入来预测译文的长度。在训练时，解码器的长度为参考译文长度。在测试时，解码器的长度由长度预测器决定。

参数设置　在实验中，除非特别说明，我们将采样次数 n 设为 10，Reinforce-Topk 的参数 k 设为 5，用 ROUGE-2 指标作为奖赏函数。在预训练阶段，WMT14 英德数据集的训练步数为 30 万步，WMT16 英罗数据集的训练步数为 15 万步。在微调阶段，训练步数均为 3000 步。在预训练阶段，batch 大小为 12.8 万词，在微

表 3-2 本章所提方法在 WMT14 英语 ↔ 德语、WMT16 英语 ↔ 罗马尼亚语数据集上的性能。

Table 3-2 The performance of our methods on WMT14 En↔De and WMT16 En↔Ro.

Model	WMT14		WMT16		Speedup
	EN-DE	DE-EN	EN-RO	RO-EN	
Transformer	27.42	31.63	34.18	33.72	1.0×
Vanilla NAT	19.51	24.47	28.89	29.35	15.6×
NAT-FT[8]	17.69	21.47	27.29	29.06	15.6×
LT[42]	19.80	—	—	—	3.8×
CTC[24]	17.68	19.80	19.93	24.71	—
INAT[58]	21.54	25.43	29.66	30.30	2.4×
ENAT[102]	20.65	23.02	30.08	—	25.3×
NAT-REG[65]	20.65	24.77	—	—	27.6×
NAT-Hints[103]	21.11	25.24	—	—	30.8×
Reinforce-Base	23.23	27.59	29.67	29.85	15.6×
Reinforce-Step	23.76	28.08	30.14	30.31	15.6×
Reinforce-Topk	24.68	29.05	30.73	30.88	15.6×
Traverse-Ref	**25.15**	**29.50**	**31.12**	**31.34**	15.6×

调阶段，batch 大小为 51.2 万词。对于 WMT14 英德数据集，dropout 比例在预训练时设为 0.3，在微调时设为 0.1。对于 WMT16 英罗数据集，dropout 比例均设为 0.3。我们使用 0.01 L_2 权重衰减和标签平滑技术（$\epsilon = 0.1$）来对模型做正则化。我们用 Adam 优化器[101] 来优化模型，其参数为 $\beta = (0.9, 0.98)$、$\epsilon = 10^{-8}$。学习率在 1 万步内线性提升至 $5 \cdot 10^{-4}$，之后按倒数平方根规则衰减。我们用 8 张 GeForce RTX 3090 GPU 卡来训练模型，用单张卡在 batch 大小为 1 的设置下测量解码速度。

3.5.2 主要实验结果

我们在表3-2中给出了本章所提方法（Reinforce-Base、Reinforce-Step、Reinforce-Topk、Traverse-Ref）的主要实验结果。其中，Vanilla NAT 为非自回归的基线模型，我们的其他模型均从 Vanilla NAT 微调而来。可以看到，这些方法均能显著提升非自回归模型的翻译质量，即便是最基础的 Reinforce-Base 方法也能通过短时间的微调将 Vanilla NAT 模型的性能在 WMT14 英德数据集上提升 3 BLEU 值以上。这显示了本章所提方法的有效性，表明序列级训练相比于词级训练更适合非自回归模型。

对比四种序列级训练方法，可以看到我们提出的减小估计方差的技术可以进一步提升非自回归模型的翻译质量。与基础的 Reinforce-Base 方法相比，

Reinforce-Step 将句级奖赏改进为词级奖赏，使翻译质量提升了约 0.5 BLEU。Reinforce-Topk 对重要的词做遍历来减小估计方差，使翻译质量进一步提升了约 0.8 BLEU。最后，Traverse-Ref 给出了一种遍历整个词表的方法，进一步改善了翻译质量。相比于自回归模型，Traverse-Ref 方法仅有约 3 BLEU 值的性能差距，同时保持着 15.6 倍的解码加速比。相比于同时代的其他工作，序列级训练方法解决了非自回归模型的关键问题，有着显著的性能优势。

3.5.3　消融实验

我们在 WMT14 英德数据的开发集上开展了消融实验，以验证本章方法中一些超参数的影响，包括奖赏函数的选取和 Reinforce-Topk 方法中 k 的大小。

奖赏函数　本章提出的序列级训练方法将序列级评估指标看作奖赏函数，以最大化奖赏函数为目标来训练非自回归模型。因此，奖赏函数的选择对序列级训练方法的最终性能有决定性影响。本章提出的方法对奖赏函数几乎没有限制，只有 Traverse-Ref 方法要求奖赏函数为匹配型奖赏。因此，我们将三种广泛使用的评估指标 BLEU[79]、GLEU[78] 和 ROUGE-2[80] 作为候选，并通过实验选出最合适的一种作为奖赏函数。一方面，我们直接用这三种指标微调非自回归模型，比较它们的性能水平。另一方面，我们使用 WMT16 DAseg De-En 数据集直接评估这些指标与真实翻译质量的相关性。这一数据集由 560 组数据组成，每组包含原文、模型译文、参考译文和人类评分，我们对每种自动评估指标计算它的打分与人类评分的皮尔逊相关系数，以此评估这些指标的好坏。表3-3给出了我们的实验结果。可以看到，用这三种指标训练模型后的 BLEU 性能没有显着差异。就与人类评分的相关性而言，BLEU 显著弱于其他两种指标，这验证了 BLEU 在句级评估上的不稳定性。由于 ROUGE-2 与 GLEU 的性能差异不大，但 ROUGE-2 只在 2 元组层面上计算与参考译文的匹配，计算成本较低，我们最终使用 ROUGE-2 作为序列级训练中的奖赏函数。

表 3-3 使用不同奖赏函数时在 WMT14 英德验证集上的 BLEU 值，以及这些指标与人类评分的皮尔森相关系数。

Table 3-3 BLEU scores on WMT14 En-De validation set when using different reward functions, and the pearson correlation coefficient of these rewards.

	Correlation	Reinforce-Base	Reinforce-Step	Traverse-Ref	Average
BLEU	0.389	22.58	23.01	24.12	23.24
GLEU	0.482	22.56	23.17	24.16	23.30
ROUGE-2	0.483	22.39	23.14	24.25	23.26

遍历数目　Reinforce-Topk 方法通过对词表中最重要的一部分词做遍历来减小估计方差，其中遍历数目 k 是一个重要的超参数。当 k 为 0 时，Reinforce-Topk 方法就退化为了 Reinforce-Step。当 k 为词表大小 $|V|$ 时，Reinforce-Topk 方法就能

达到与 Traverse-Ref 相同的性能，但使用这么大的 k 也会使训练速度变得极为缓慢。因此，我们需要选择一个合适的 k 值来平衡模型性能和训练代价。我们在 WMT14 英德验证集上进行了实验，将模型性能随遍历数目 k 变化的曲线画在了图3-2中。

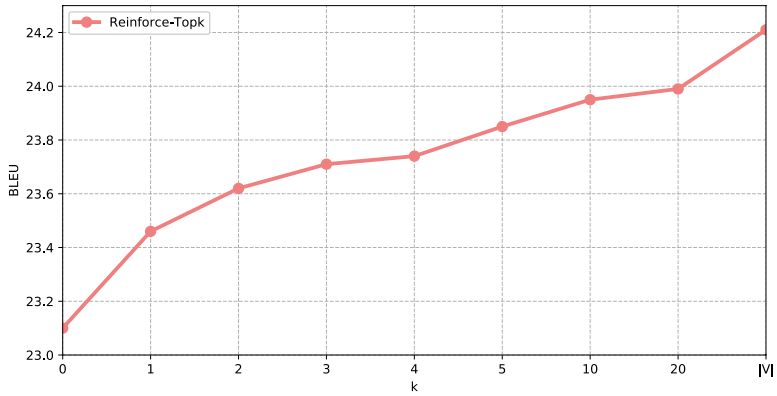


图 3-2 Reinforce-Topk 方法的性能随 k 变化的曲线。

Figure 3-2 The performance curve of Reinforce-Topk as a function of k.

从图中可以看到，随着遍历数目 k 从 0 增长到 5，模型的性能有着显著增长。遍历数目 k 从 5 增长到 10 时，模型的性能也有一定提升。当 k 从 10 增长到 20 时，性能的提升已经很轻微了，但我们还要为之付出双倍的计算成本。因此，k 的取值应该在 5 到 10 之间较为合适。另外，我们用 Traverse-Ref 方法展现了 $k = |V|$ 时的性能，相比于较小的 k 还是能有显著的性能提升。

3.5.4 训练速度

在表3-1中，我们对各种方法的时间复杂度做了理论分析。在这里，我们进一步通过实验来测量它们实际要花费的时间。我们用 CE 表示交叉熵损失，将 RF 作为对 Reinforce 的缩写，在表3-4给出了交叉熵损失和本章提出的四种序列级训练方法使用单张 GeForce RTX 3090 处理一个大小为 6.4 万词的 batch 所需的时间。如表所示，交叉熵损失的训练速度是最快的，仅需要 1.2 秒就能处理完 6.4 万词。Traverse-Ref 方法速度最慢，需要 63.2 秒。尽管低方差的序列级训练方法速度较慢，但整体上也还在可承担的范围内。我们采用的训练策略是先用交叉熵损失预训练模型三十万步，再使用序列级训练微调模型三千步，因此序列级训练的代价还是在可控范围内。

表 3-4 不同训练方法的训练速度对比。

Table 3-4 The comparison of training speeds for different training methods.

Method	CE	RF-Base	RF-Step	RF-Topk	Traverse-Ref
Time	1.2s	2.2s	16.9s	31.3s	63.2s

3.5.5　解码速度

非自回归模型能并行生成整句译文，因此在逐句翻译时的延迟很低。如果增大解码的 batch 大小，令模型同时解码多个句子，自回归模型也能一定程度上发挥硬件的并行计算能力，这会一定程度上减弱非自回归模型的速度优势。为了探究非自回归模型在这种场景下的解码加速效果，我们在不同 batch 大小下分别测量自回归和非自回归模型的翻译延迟，在 WMT14 英德测试集上进行实验，结果如图3-3所示。

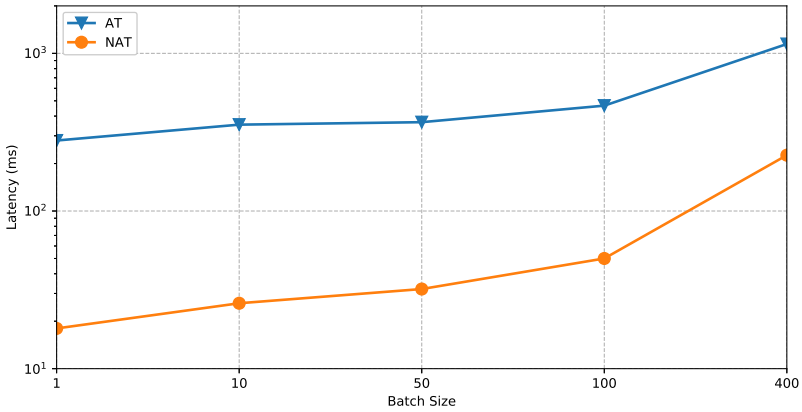


图 3-3 自回归和非自回归模型在不同 batch 大小下的翻译延迟。

Figure 3-3 The translation latency of autoregressive and non-autoregressive models under different batch sizes.

从图中可以看到，随着解码 batch 大小的增加，非自回归和自回归模型的解码延迟均会增大，其中非自回归模型解码延迟的上升速度更快一些。受显存大小的限制，硬件支持的最大 batch 大小为 400，此时非自回归模型的解码速度大约是自回归模型的 5 倍左右。这说明了非自回归模型在各种 batch 大小的场景下都相对自回归模型有显著的速度优势。

3.6　本章小结

针对词级交叉熵损失无法准确评估模型输出的问题，本章提出对非自回归模型做序列级训练，直接令模型优化序列级的评估指标。我们将待优化的指标看作强化学习中的奖赏函数，令模型以最大化期望奖赏为训练目标，使用强化学习算法来估计损失函数的梯度。实验结果表明，最基础的序列级训练方法就能显著提升非自回归模型的翻译质量，表明序列级训练相比于词级训练更适合非自回归模型。随后，我们利用非自回归模型并行生成的特性，再结合机器翻译评估指标的性质，提出了对基础算法的一系列改进，可以有效地降低梯度估计的方差，进一步提升了非自回归模型的翻译质量。

第 4 章　基于 n 元组匹配的词袋距离最小化

4.1　引言

神经机器翻译模型使用编码器-解码器框架实现了端到端的机器翻译，并取得了令人瞩目的成绩[5–7]。近年来，基于注意力机制的 Transformer 模型[1] 在模型的训练速度和翻译质量上都取得了大幅提升，成为了神经机器翻译的主流模型。Transformer 模型摒弃了序列建模中的循环结构，使模型能够在序列层面上并行计算，训练效率得以大幅提升。然而，目前的神经机器翻译模型主要使用自回归机制来建模序列的概率，模型在解码下一个词时必须以上个词的预测结果作为输入，因此在解码时无法利用 Transformer 模型并行计算的特性，使模型的解码速度较慢。

Gu 等[8] 提出非自回归神经机器翻译（Non-Autoregressive Neural Machine Translation，NAT）来提升模型的解码速度。非自回归模型的模型结构与自回归模型基本相同，也是基于编码器-解码器架构的 Transformer 模型。主要的不同点在于非自回归模型对目标端各位置的概率分布做了独立的建模，因此解码器不需要以之前的预测结果作为输入，而是直接将编码器的词向量输入均匀拷贝过来作为解码器的输入信号。基于这种结构，非自回归模型可以并行地预测所有位置的概率分布，并采用 argmax 解码并行生成译文，大幅提升了模型的解码速度。

然而，对非自回归模型来说，基于极大似然估计的交叉熵损失无法建模目标端的序列依赖关系，导致损失函数与翻译质量的相关性较弱，无法准确地评估模型的输出结果。交叉熵损失仅鼓励模型在每个位置生成参考译文的对应词，不考虑译文的全局正确性。例如，在图4-1中，尽管模型的输出 “I have to get up and start working’’ 与参考译文的语义比较接近，但由于它与参考译文没有按位置严格对齐，这种输出反而会有较高的损失值。在交叉熵损失的训练下，译文可能会被纠正为 “I have to up start start working’’，这种译文的损失值更低，但实际的翻译质量也更差。交叉熵损失的这种缺陷导致了非自回归模型不擅长建模目标端的序列依赖关系，使翻译结果往往会出现过翻译和漏翻译的错误[65]。另外，由于交叉熵损失和翻译质量的相关性较弱，用交叉熵损失训练的非自回归模型通常需要较长时间才能训练到收敛。

机器翻译的评估指标（如 BLEU 值）主要基于 n 元组的匹配准确率来评估模型的翻译质量。因此，若能直接基于 n 元组的匹配来设计损失函数，就能更准确、高效地训练非自回归模型。在本章中，我们将 n 元组匹配形式化为模型输出与参考译文之间的 n 元组词袋（Bag-of-Ngrams, BoN）距离，针对非自回归模型提出最小化一阶 n 元组词袋距离的训练目标。由于模型输出与参考译文不一定是对齐的，我们在训练目标中放松对齐的限制，而是直接最小化 n 元组词袋的差异。如图4-1所示，n 元组词袋距离适用于未对齐的场景，对翻译质量的评估更加准确。由于目标端译文的搜索空间是指数级的，要直接优化这样的训练目标通常

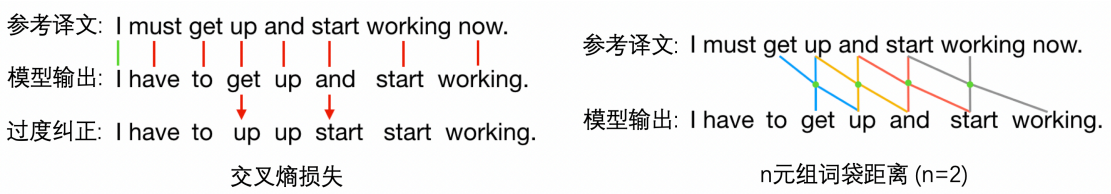


图 4-1 交叉熵损失与 2 元组词袋距离的示意图。

Figure 4-1 The illustration of cross-entropy loss and bag-of-2grams difference.

都存在较大困难。过去的工作[81,82,84] 通常会应用强化学习算法求得对梯度的无偏估计，但梯度估计的方差往往较大。在上一章的序列级训练方法中，我们针对非自回归模型提出了一系列改进方法来降低估计方差，但仍无法完全消除估计方差，并且也使得训练代价较大。在本章中，我们利用非自回归模型并行生成的特性设计了能准确计算一阶距离的高效算法，使非自回归模型能直接以最小化 n 元组词袋距离为训练目标。这一训练目标有如下优点：

- n 元组词袋距离能够建模目标端的序列依赖关系，与翻译质量的相关性强。
- 我们不需要做任何估计，可以对 n 元组词袋距离做准确计算。
- n 元组词袋距离是可导的，计算代价也很小，能够直接用来训练模型。

我们在多个翻译任务（WMT14 En↔De，WMT16 En↔Ro）上对所提方法进行了实验验证。结果表明，n 元组词袋距离与翻译质量有较强的相关性，最小化这一训练目标能显著提升非自回归模型的翻译质量。结合 n 元组词袋距离与上一章的强化学习方法后，翻译质量能得到进一步提升，达到了与自回归模型相近的性能水平。

4.2 损失函数改进：n 元组词袋的一阶距离

词袋（Bag-of-Words, BoW）[104] 是表示文本信息的一种常用特征，在自然语言处理任务中有广泛应用。词袋将文本看作一组词的无序集合，用一个向量表示每个词在文本中出现的次数，但也忽略了词的顺序和上下文关系，因此可能会忽略文本中的一些重要信息。n 元组词袋（Bag-of-Ngrams, BoN）[105–107] 是词袋模型的一种扩展形式，可以捕捉到文本中的局部结构和上下文关系，表示能力比词袋更强。此外，n 元组词袋在机器翻译的评估中也起到了重要作用。基于统计方法的机器翻译评估指标主要基于 n 元组匹配的准确率来评估翻译质量，而匹配准确率基本取决于 n 元组词袋间的差异，即 n 元组词袋间的距离大小。

我们从 n 元组词袋出发设计非自回归模型的损失函数，首先对模型和参考译文的 n 元组词袋做形式化定义，再针对非自回归模型设计高效计算 n 元组词袋的算法，最后给出计算模型输出与参考译文之间的一阶 n 元组词袋距离的方法，以最小化 n 元组词袋距离为目标训练模型。此外，我们也给出了结合这一训练目标和强化学习方法的三阶段训练策略。

4.2.1　定义

我们将 n 元组词袋定义为一个长度为 $|V|^n$ 的向量，其中 V 表示词表的大小。向量的每一维都对应一个 n 元组 $g = (g_1, \ldots, g_n)$，这一维的值代表 n 元组 g 的出现次数。对于一个目标端译文 $Y = \{y_1, ..., y_T\}$，我们用 BoN_Y 来表示译文 Y 的 n 元组词袋向量，用 $\mathrm{BoN}_Y(g)$ 表示 n 元组 g 对应维度的值，即 g 在译文 Y 中的出现次数，形式化定义如下：

$$\mathrm{BoN}_Y(g) = \sum_{t=0}^{T-n} 1(y_{t+1:t+n} = g), \tag{4-1}$$

其中，$1(\cdot)$ 为指示函数，若括号内的条件满足，则其取值为 1，否则值为 0。

在上面，我们给出了对单个句子的 n 元组词袋的定义，这种定义方式与过去的工作也较为一致。对于像神经机器翻译这种建模目标端概率分布的生成模型，过去并没有对这种模型定义 n 元组词袋的案例。由于所有译文都有概率被模型生成，我们采用期望的形式定义模型的 n 元组词袋。令模型的参数为 θ，我们用 BoN_θ 表示模型生成的 n 元组词袋，用 $\mathrm{BoN}_\theta(g)$ 表示 n 元组 g 对应维度的值，形式化定义如下：

$$\mathrm{BoN}_\theta(g) = \sum_{Y} p(Y|X,\theta) \cdot \mathrm{BoN}_Y(g). \tag{4-2}$$

4.2.2　高效计算

由于译文的搜索空间是指数级的，我们无法直接根据定义去计算 n 元组 g 的期望出现次数 $\mathrm{BoN}_\theta(g)$。对自回归模型来说，由于每个位置的翻译概率与其他位置的翻译结果耦合在了一起，我们很难对式4-2做变换来简化其计算。相反地，非自回归模型独立地对每个位置的翻译概率进行建模，使我们能够将目标端序列划分为若干个子片段，独立地分析每个片段下的 n 元组词袋。具体地，对于非自回归模型，我们可以将式4-2转换为以下形式：

定理 4.1.

$$BoN_\theta(g) = \sum_{t=0}^{T-n} \prod_{i=1}^{n} p_{t+i}(y_{t+i} = g_i|X,\theta). \tag{4-3}$$

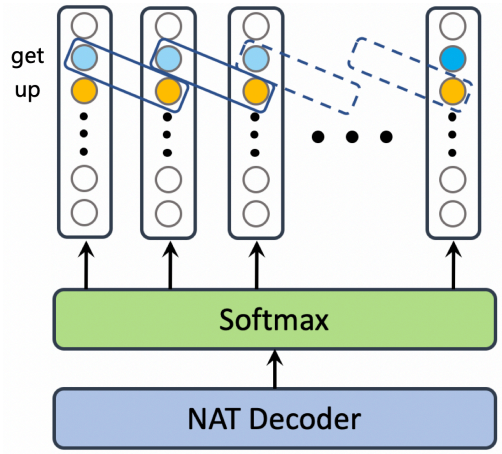


图 4-2 根据式4-3计算 BoN_θ(“get up”) 的示例。

Figure 4-2 The illustration of calculating BoN_θ(“get up”) according to equation 4-3.

证明.

$$
\begin{aligned}
\mathrm{BoN}_\theta(g) &= \sum_{Y} p(Y|X,\theta) \cdot \sum_{t=0}^{T-n} 1(y_{t+1:t+n} = g) \\
&= \sum_{t=0}^{T-n} \sum_{Y} p(Y|X,\theta) \cdot 1(y_{t+1:t+n} = g) \\
&= \sum_{t=0}^{T-n} \sum_{Y_{1:t}} p(Y_{1:t}|X,\theta) \sum_{Y_{t+n+1:T}} p(Y_{t+n+1:T}|X,\theta) \sum_{Y_{t+1:t+n}} p(Y_{t+1:t+n}|X,\theta) \cdot 1(y_{t+1:t+n} = g) \\
&= \sum_{t=0}^{T-n} \sum_{Y_{t+1:t+n}} p(Y_{t+1:t+n}|X,\theta) \cdot 1(y_{t+1:t+n} = g) \\
&= \sum_{t=0}^{T-n} \prod_{i=1}^{n} p_{t+i}(y_{t+i} = g_i|X,\theta).
\end{aligned}
$$

□

上式给出了计算模型 n 元组词袋 $\mathrm{BoN}_\theta(g)$ 的一种高效算法。我们仅需在非自回归模型预测的概率分布上滑动一个大小为 n 的窗口，在窗口的每个位置计算 n 元组 g 的数目，最后将所有窗口的值求和就计算出了 n 元组 g 的期望出现次数 $\mathrm{BoN}_\theta(g)$。这种方法不需要做任何估计，能够准确、高效地计算出 n 元组词袋 BoN_θ 中任意维度的值，但上述推导仅对非自回归模型成立，无法推广到自回归模型中。图4-2给出了按这种方法计算二元组 “get up’’ 期望数目的示例。

4.2.3 最小化词袋距离

我们的训练目标为最小化非自回归模型的输出结果和参考译文之间的 n 元组词袋距离。我们不难得到参考译文的 n 元组词袋向量，只需考虑参考译文中的每个 n 元组 g，将向量中对应位置设为 g 的出现次数，其他位置均设为 0。然而，

我们很难完整计算模型的 n 元组词袋向量 BoN_θ。即便我们能高效地计算出向量中每一维的值，计算总长度为 $|V|^n$ 的完整向量的代价仍然是无法承受的。唯一的例外是 $n = 1$ 的情况，这时 n 元组词袋退化为了简单的词袋（Bag-of-Words，BoW）。在这种情况下，我们只需要对所有位置的概率分布求和就能得到 BoW_θ。对两个向量的距离有多种度量指标，例如 L_1 距离、L_2 距离、余弦距离等，我们可以直接应用这些指标来度量两个词袋向量的距离，将所得的 BoW-L_1、BoW-L_2 和 BoW-*Cos* 作为损失函数。

4.2.4　最小化 n 元组词袋距离

对于 $n > 1$ 的情况，我们无法计算出完整的 n 元组词袋向量 BoN_θ，因此无法使用 L_2 距离、余弦距离等距离度量作为损失。幸运的是，我们发现两个向量之间的 L_1 距离，简写为 BoN-L_1，可以被高效、准确地计算出来。模型的向量 BoN_θ 是非常稠密的，每一维都是非零值。相反地，假设参考译文为 $\hat{Y}$，它的 n 元组词袋向量 $\text{BoN}_{\hat{Y}}$ 是非常稀疏的，仅少数位置是非零值。利用这个性质，我们可以将 BoN-L_1 写成如下形式来减小计算代价：

定理 4.2.

$$\textit{BoN-}L_1 = 2(T - n + 1 - \sum_g \min(\textit{BoN}_\theta(g), \textit{BoN}_{\hat{Y}}(g))). \tag{4-4}$$

证明. 首先，我们证明向量 $\text{BoN}_{\hat{Y}}$ 和 BoN_θ 的 L_1 范数均为 $T - n + 1$:

$$\begin{aligned}
\sum_g \text{BoN}_{\hat{Y}}(g) &= \sum_{t=0}^{T-n} \sum_g 1(y_{t+1:t+n} = g) = T - n + 1. \\
\sum_g \text{BoN}_\theta(g) &= \sum_g \sum_Y p(Y|X,\theta) \cdot \text{BoN}_Y(g) \\
&= \sum_Y p(Y|X,\theta) \cdot \sum_g \text{BoN}_Y(g) = T - n + 1.
\end{aligned}$$

基于此，我们能将一阶距离 BoN-L_1 转换为所需形式：

$$\begin{aligned}
\text{BoN-}L_1 &= \sum_g |\text{BoN}_\theta(g) - \text{BoN}_{\hat{Y}}(g)| \\
&= \sum_g (\text{BoN}_\theta(g) + \text{BoN}_{\hat{Y}}(g) - 2\min(\text{BoN}_\theta(g), \text{BoN}_{\hat{Y}}(g)) \\
&= 2(T - n + 1 - \sum_g \min(\text{BoN}_\theta(g), \text{BoN}_{\hat{Y}}(g))).
\end{aligned}$$

□

式4-4中，$\text{BoN}_\theta(g)$ 和 $\text{BoN}_{\hat{Y}}(g)$ 之间的最小值可以理解为是 n 元组 g 的匹配数目，所得的 L_1 距离衡量的就是模型预测结果与参考译文间未匹配的 n 元组数目。注意到，向量 $\text{BoN}_{\hat{Y}}$ 是高度稀疏的，式4-4中的最小值仅当 n 元组 g 在参考译文中时才是非零的。因此，我们只需要对参考译文中的 n 元组做求和，不需

算法 6 BoN-L_1

Input: probability distribution $p(\cdot|X,\theta)$), reference sentence $\hat{Y}$, prediction length T, n
Output: BoN distance BoN-L_1

construct the bag-of-ngrams $\text{BoN}_{\hat{Y}}$ for the reference sentence
ref-ngrams=$\{g|\text{BoN}_{\hat{Y}}(g)$!= $0\}$
match = 0
for g in ref-ngrams **do**
    calculate $\text{BoN}_{\theta}(g)$ according to Equation (4-3)
    match += $\min(\text{BoN}_{\theta}(g), \text{BoN}_{\hat{Y}}(g))$
end for
BoN-L_1 = $2 \cdot$ (T-n+1-match)
return BoN-L_1

要计算完整的向量 BoN_{θ}，这大幅降低了计算代价，使我们能高效、准确地计算 BoN-L_1。算法6给出了一阶距离 BoN-L_1 的具体计算过程。

我们将距离归一化到 $[0,1]$ 区间内来保持损失函数在尺度上的稳定性。对于词袋距离，我们保持余弦距离 BoW-Cos 不变，将 BoW-L_1 和 BoW-L_2 除以常数 $2T$ 来作为损失函数。

$$\mathcal{L}_{\text{BoW-}L_1}(\theta) = \frac{\text{BoW-}L_1}{2T}, \qquad \mathcal{L}_{\text{BoW-}L_2}(\theta) = \frac{\text{BoW-}L_2}{2T}. \tag{4-5}$$

对于 n 元组词袋损失，我们将一阶距离 BoN-L_1 除以常数 $2(T-n+1)$ 来作为损失函数。

$$\mathcal{L}_{\text{BoN-}L_1}(\theta) = \frac{\text{BoN-}L_1}{2(T-n+1)}. \tag{4-6}$$

4.2.5 训练流程

我们采用预训练再微调的训练流程，先用交叉熵损失预训练模型，再用本章提出的 n 元组损失来微调模型。另外，我们也考虑结合本章的损失函数和第二章的强化学习方法，使用一种三阶段训练策略来训练模型。词级交叉熵损失无法准确地评估非自回归模型的输出，但它也具有训练速度快的优势，适合用来学习对文本的表示。因此，我们在第一阶段使用交叉熵损失来预训练非自回归模型。本章提出的 n 元组损失可以准确、高效地评估模型输出与参考译文的 n 元组差异，局限性是只能计算一阶距离，我们用它来在第二阶段微调非自回归模型。强化学习方法的训练速度较慢，需要在 CPU 上进行大量计算以减少梯度估计的方差，因此我们在第三阶段用第二章中的 Traverse-Ref 方法微调非自回归模型，这一阶段的训练步数是最少的。这种三阶段训练策略能发挥各个损失函数的优势来提升模型的翻译质量，并将训练成本控制在较低范围内。

4.3　实验结果与分析

在本节中，我们将通过实验来验证序列级训练方法的实际效果，首先介绍相关的实验设置，再给出主要的实验结果，最后对所得结果进行分析。

4.3.1　实验设置

数据集　我们在四个翻译任务上验证了所提的方法：WMT14 英语 ↔ 德语（En↔De，约 450 万个句对）、WMT16 英语 ↔ 罗马尼亚语（En↔Ro，约 61 万个句对）。对于 WMT14 英德数据集，我们用大小写敏感的 BLEU 值来评估模型的翻译质量，分别将 *newstest2013* 和 *newstest2014* 作为验证集和测试集。对于 WMT16 英罗数据集，我们用大小写不敏感的 BLEU 值来评估模型的翻译质量，分别将 *newsdev-2016* 和 *newstest-2016* 作为验证集和测试集。我们用 32K 次操作的联合 BPE 模型[100] 来切分源端和目标端词汇，并在源端和目标端共享词表。

知识蒸馏　序列级知识蒸馏[12,13] 是非自回归模型中的一项关键技术。在所有翻译任务中，我们都应用序列级知识蒸馏技术，首先训练一个自回归模型作为教师，再令它翻译一遍训练数据集的源端，将原文与译文组成句对来构建蒸馏数据集。最后，我们使用蒸馏数据集来训练非自回归模型。

基线模型　我们将基础设置的 Transformer 模型[1] 作为自回归的基线模型。非自回归的基线模型也采用 Transformer 的模型结构，我们仅对解码器的注意力掩码矩阵做修改，使解码器的每个位置都能看到所有其他位置的信息。我们采用平均拷贝的方法[8] 来构建解码器的输入。我们在编码器顶端接入一个长度预测器，以编码器隐状态为输入来预测译文的长度。在训练时，解码器的长度为参考译文长度。在测试时，解码器的长度由长度预测器决定。

重排序　在解码时，除了直接选取每个位置最高概率的单词组成译文，我们也采用另一种重排解码方法[8]，首先并行生成多个译文，再对这些译文重排序后输出最佳的译文。长度预测器给出的译文长度为 T 时，我们将句长区间设置为 $[T-B, T+B]$，令模型同时生成 $2B+1$ 个不同长度的译文。随后，我们用自回归模型计算这些译文的翻译概率，根据翻译概率对译文打分，将分数最高的译文作为翻译结果。

参数设置　在实验中，除非特别说明，我们将 n 元组的大小设置为 $n=2$。在预训练阶段，WMT14 英德数据集的训练步数为 30 万步，WMT16 英罗数据集的训练步数为 15 万步。在第二阶段的微调中，训练步数均为 3000 步。在第三阶段的微调中，训练步数均为 300 步。在预训练阶段，batch 大小为 12.8 万词，在微调阶段，batch 大小为 51.2 万词。对于 WMT14 英德数据集，dropout 比例在预训练时设为 0.3，在微调时设为 0.1。对于 WMT16 英罗数据集，dropout 比例均设为 0.3。我们使用 0.01 L_2 权重衰减和标签平滑技术（$\epsilon=0.1$）来对模型做正则化。我们用 Adam 优化器[101] 来优化模型，其参数为 $\beta=(0.9, 0.98)$、$\epsilon=10^{-8}$。学习率在 1 万步内线性提升至 $5\cdot10^{-4}$，之后按倒数平方根规则衰减。我们用 8

表 4-1 本章所提方法在 WMT14 英语 ↔ 德语、WMT16 英语 ↔ 罗马尼亚语数据集上的性能。

Table 4-1 The performance of our methods on WMT14 En↔De and WMT16 En↔Ro.

	Model	WMT14		WMT16		Speedup
		EN-DE	DE-EN	EN-RO	RO-EN	
Base	Transformer	27.42	31.63	34.18	33.72	1.0×
	Vanilla NAT	19.51	24.47	28.89	29.35	15.6×
RL	Reinforce-Base	23.23	27.59	29.67	29.85	15.6×
	Reinforce-Step	23.76	28.08	30.14	30.31	15.6×
	Reinforce-Topk	24.68	29.05	30.73	30.88	15.6×
	Traverse-Ref	25.15	29.50	31.12	31.34	15.6×
BoN	BoW-Cos	23.90	28.11	29.72	29.69	15.6×
	BoW-L_2	24.22	29.03	30.08	29.93	15.6×
	BoW-L_1	24.75	29.41	31.01	31.19	15.6×
	BoN-L_1 (N=2)	25.28	29.66	31.37	31.51	15.6×
	BoN-L_1 (N=3)	25.13	29.47	31.29	31.40	15.6×
3-Stage	RL+BoN	**25.54**	**29.91**	**31.69**	**31.78**	15.6×

张 GeForce RTX 3090 GPU 卡来训练模型，用单张卡在 batch 大小为 1 的设置下测量解码速度。

4.3.2 主要实验结果

我们在表4-1中给出了用交叉熵损失训练的自回归模型 Transformer 和非自回归模型 Vanilla NAT 的性能水平，以及分别用第二章的强化学习方法和本章 n 元组方法微调 Vanilla NAT 的效果。另外，我们也尝试使用三阶段训练策略来结合以上方法（交叉熵损失，Traverse-Ref，BoN-L_1 (N=2)）。从表中，我们有以下几个发现：

1. 在基于词袋（BoW）的训练目标中，用 L_1 距离训练模型的效果是最好的。比较 BoW-Cos、BoW-L_1、BoW-L_2 的性能，BoW-L_1 的 BLEU 值最高并显著优于其他两种损失，说明 L_1 距离比较适合作为损失函数来训练非自回归模型。对于 n 元组词袋（$n > 1$）来说，虽然能够计算 L_1 距离，但它的局限性是无法将 L_2 距离、余弦距离等作为损失。基于这个发现，L_1 距离已经是较好的损失函数了，这个局限性的影响就小了很多。

2. 在基于 n 元组词袋（BoN）的训练目标中，$N = 2$ 设置下的效果最好，并且优于词袋目标（BoW）和强化学习方法。相比于完全不考虑词序的词袋损失，n 元组词袋能够建模序列前后的依赖关系，更好地评估输出的准确性，BLEU 值也显

著高于词袋损失 BoW-L_1。相比于强化学习方法 Traverse-Ref，n 元组词袋方法不需要做梯度估计，训练过程中的噪音更小，在 BLEU 性能上也有一定优势。

3. 三阶段训练策略能有效地结合强化学习方法和 n 元组方法。结合这两个类别下最好的方法（即 Traverse-Ref 和 BoN-L_1 (N=2)）进行三阶段训练后，模型的性能达到了最优水平，将基线模型 Vanilla NAT 的性能最高提升了 6 个 BLEU 值以上，大幅缩小了非自回归模型与自回归模型的性能差距。

在表4-2中，我们对本章所提方法与之前其他工作的性能做了对比，其中 n 表示重排序译文的个数。结果显示，我们的方法与自回归模型之间的性能差距仅为 2 BLEU 左右，优于当时大多数的非自回归模型。对 9 个候选译文进行重排序后，翻译质量进一步得到提升，与自回归模型的性能差距仅为 0.8 BLEU 左右，并且仍能保持 9.0× 的加速比。

在上述实验中，我们验证了 n 元组方法对单轮非自回归模型的有效性。进一步地，我们也尝试将其应用在迭代式非自回归模型中。我们使用掩码预测模型（CMLM[35]）作为基线模型，并用本章方法对其进行微调，在 WMT14 英德测试集上的实验结果如图4-3所示。可见，n 元组方法在各种迭代次数下都有稳定的提升，尤其是在迭代次数较低时效果很明显，仅用 2 轮迭代就能达到原模型在 4 轮迭代时的性能。在迭代次数较高时，我们的方法也有约 0.4 BLEU 的稳定提升，展现了其在迭代式非自回归模型上的有效性。

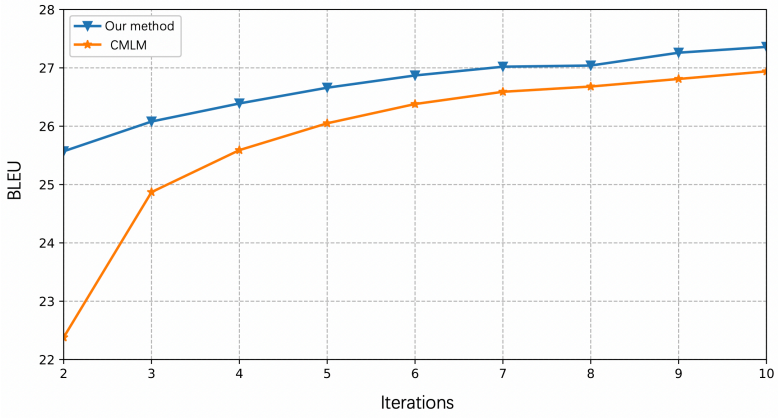


图 4-3 迭代式非自回归模型 CMLM 在微调前后的效果对比。

Figure 4-3 Performance comparison between the iterative CMLM model and the finetuned model.

4.3.3　训练策略的影响

我们使用三阶段训练策略来结合交叉熵损失、强化学习方法以及本章所提的 n 元组方法。在本节，我们对这种训练策略带来的影响进行分析。我们用 CE 表示交叉熵损失，将 RF 作为对 Reinforce 的缩写，在表4-3给出了每种方法使用单张 GeForce RTX 3090 处理一个大小为 6.4 万词的 batch 所需的时间。可见，交

表 4-2 本章方法与之前工作的性能对比。

Table 4-2 Performance comparison between our method and existing methods.

Model	WMT14		WMT16		Speedup
	EN-DE	DE-EN	EN-RO	RO-EN	
Autoregressive					
Transformer	27.42	31.63	34.18	33.72	1.0×
Non-Autoregressive w/o Rescoring					
NAT-FT[8]	17.69	21.47	27.29	29.06	15.6×
ENAT[102]	20.65	23.02	30.08	—	25.3×
NAT-REG[65]	20.65	24.77	—	—	27.6×
NAT-Hints[103]	21.11	25.24	—	—	30.8×
imitate-NAT[108]	22.44	25.67	28.61	28.90	18.6×
FlowSeq[40]	23.72	28.39	29.73	30.72	—
NART-DCRF[19]	23.44	27.22	—	—	10.4×
ReorderNAT[46]	22.79	27.28	29.30	29.50	16.1×
PNAT[109]	23.05	27.18	—	—	7.3×
FCL-NAT[110]	21.70	25.32	—	—	28.9×
AXE[69]	23.53	27.90	30.75	31.54	—
EM[15]	24.54	27.93	—	—	16.4×
Imputer[25]	**25.80**	28.40	**32.30**	31.70	—
Our method	25.54	**29.91**	31.69	**31.78**	15.6×
Non-Autoregressive w/ Rescoring					
NAT-FT (n=100)[8]	19.17	23.20	29.79	31.44	2.36×
ENAT (n=9)[102]	24.28	26.10	**34.51**	—	12.4×
NAT-REG (n=9)[65]	24.61	28.90	—	—	15.1×
NAT-Hints (n=9)[103]	25.20	29.52	—	—	17.8×
imitate-NAT (n=7)[108]	24.15	27.28	31.45	31.81	9.70×
FlowSeq (n=15)[40]	25.03	30.48	31.89	32.43	—
NART-DCRF (n=9)[19]	26.07	29.68	—	—	6.14×
PNAT (n=7)[109]	—	27.90	—	—	3.7×
FCL-NAT (n=9)[110]	25.75	29.50	—	—	16.0×
EM (n=9)[15]	25.75	29.29	—	—	9.14×
Our method (n=9)	**26.35**	**30.70**	33.21	**33.28**	9.0×

叉熵损失的训练速度最快，n 元组方法居中，Traverse-Ref 方法最慢。因此，在制定训练策略时，应避免在 Traverse-Ref 方法上的大量计算。

表 4-3 不同训练方法的训练速度对比。

Table 4-3 The comparison of training speeds for different training methods.

Method	CE	BoW	BoN (n=2)	RF-Base	RF-Step	RF-Topk	Traverse-Ref
Time	1.2s	1.5s	7.1s	2.2s	16.9s	31.3s	63.2s

基于上面的训练速度分析，我们主要考虑以下四种训练策略。首先，我们可以对 Traverse-Ref 方法和 n 元组方法的损失进行加权，用两者的加权和来微调模型。其次，我们考虑将交叉熵损失和 n 元组方法混合来预训练模型，再依次用 n 元组方法和 Traverse-Ref 方法微调。第三种策略为进行三阶段的训练，先使用 Traverse-Ref 方法微调模型，最后再使用 n 元组方法。最后为本章使用的策略，先使用 n 元组方法微调，再使用 Traverse-Ref 方法。我们用 CE 表示交叉熵损失、TR 表示 Traverse-Ref 方法、BoN 表示 n 元组方法，在表4-4中给出了这四种策略的 BLEU 值以及训练时间。

表 4-4 使用 8 张 GeForce RTX 3090 训练时，不同训练策略的训练时间以及在 WMT14 英德验证集上的 BLEU 值。

Table 4-4 Validation BLEU scores and training time of different training strategies on WMT14 En-De. The training time is measured on 8 GeForce RTX 3090 GPUs.

Strategy	BLEU	Time
1. CE—BoN+TR	24.56	65.6h
2. CE+BoN—BoN—TR	24.82	93.1h
3. CE—TR—BoN	24.63	63.3h
4. CE—BoN—TR	24.69	37.5h

上表显示，第二种训练策略的 BLEU 值最高，但训练成本也很高。第四种策略更经济，BLEU 值仅略低于第二种策略，但训练时间大幅缩短。由于我们仅在最后阶段用 Traverse-Ref 方法做短时间的微调，在训练时间上也明显优于第一和第三种方法。以上分析说明了本章使用的这种训练策略的合理性。

4.3.4　与翻译质量的相关性

直观上看，基于 n 元组词袋距离的损失函数能更好地评估模型的输出，与翻译质量的相关性比交叉熵损失更好。在本节中，我们对损失函数与翻译质量之间的相关性进行定量分析。我们使用 GLEU 值来表示翻译质量，因为它在句级翻译质量评估中比 BLEU 更为准确[78]。我们在 WMT14 英德验证集上进行实验，验证集中包含 3000 个句对。我们使用 Vanilla NAT 模型对验证集的源句预测其

概率分布，计算交叉熵损失和 n 元组损失，并进行解码来计算每句译文的 GLEU 值。最后，我们计算损失函数与 GLEU 值之间的皮尔逊相关系数。

对于交叉熵损失，我们用译文长度对其进行归一化。对于 n 元组词袋距离，损失函数是由 $2(T-n+1)$ 归一化的 L_1 距离，我们分别将 n 设为 2、3、4 来计算不同 n 元组大小下的相关性。我们用 CE 表示交叉熵损失，用 $n=k$ 表示 k 元组词袋，实验结果如表4-5所示。

表 4-5 损失函数与翻译质量之间的皮尔森相关系数。

Table 4-5 The pearson correlation coefficient bewteen loss functions and the translation quality.

Loss function	CE	$n=2$	$n=3$	$n=4$
Correlation	0.56	0.87	0.84	0.79

上表结果显示，n 元组词袋距离与翻译质量的相关性远远优于交叉熵损失。$n=2$ 时的相关性结果最好，达到了 0.87。注意到，这也与表4-1中 $n=2$ 的 BLEU 值略高于 $n=3$ 的实验结果一致。进一步地，我们对不同句长下的相关性结果做更细致的分析。我们根据源句长度将验证集分为两部分，第一部分由 1500 个短句组成，第二部分由 1500 个长句组成。我们分别测量这两个部分的皮尔逊相关系数，结果如表4-6所示。

表 4-6 损失函数与翻译质量分别在短句和长句下的皮尔森相关系数。

Table 4-6 The pearson correlation coefficient bewteen loss functions and translation quality on short sentences and long sentences.

	all	short	long
Cross-Entropy	0.56	0.68	0.44
BoN (n=2)	0.87	0.89	0.86

可以看到，交叉熵损失的相关系数在长句下较低，而 n 元组词袋损失对长句仍有较好的相关性。其中的原因不难解释。交叉熵损失要求模型译文与参考译文严格对齐，但随着句长的增大，这一对齐的要求变得更加困难，大多时候都无法满足，这导致交叉熵损失与翻译质量之间的相关系数降低。相比之下，n 元组词袋损失更加鲁棒，在未对齐的情况下也能准确评估模型的输出，因此能在长句下保持较高的相关系数。

4.3.5 句长测试

在上节中，我们分析了不同句长下损失函数与翻译质量之间的相关性，发现 n 元组词袋损失在评估长句时远优于词级交叉熵损失。在本节中，我们进一步计算模型在不同句长下的 BLEU 值，以确认这种损失函数与翻译质量相关性上的优势是否能转换为在模型性能上的优势。在 WMT14 英德验证集上，我们根据源

端句子的长度将所有句子分为不同句长的子集，并计算自回归模型、非自回归基线模型和我们的方法在这些不同句长子集上的 BLEU 值，结果如图4-4 所示。

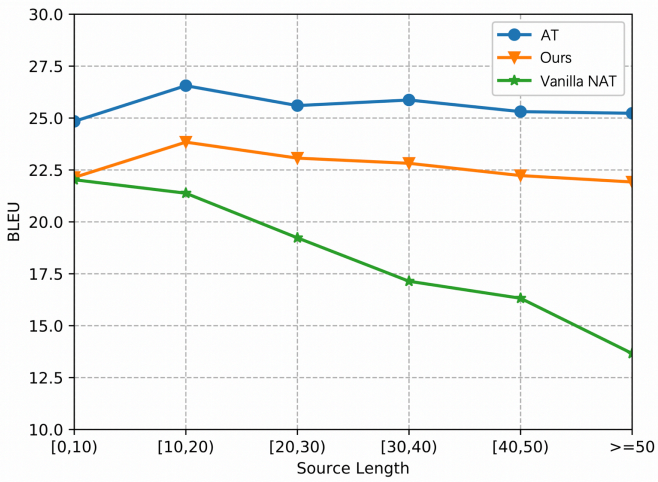


图 4-4 自回归模型、非自回归基线模型和我们的方法在不同句长下的 BLEU 值。
Figure 4-4 BLEU scores of autoregressivee Transformer, Vanilla NAT and our method under different length buckets.

如图可见，在翻译短句时，非自回归基线模型和我们的方法性能很接近，与自回归模型也没有较大差距。随着句长的增加，非自回归基线模型的翻译质量迅速下降，而我们的方法与自回归模型在不同句长上均有较稳定的性能，这与上节的相关性结果非常吻合。随着句长的增加，交叉熵损失与翻译质量之间的相关性下降，导致非自回归基线模型在翻译长句时性能较弱。相反地，n 元组词袋损失在长句上也与翻译质量高度相关，因此我们的模型在长句上也能保持稳定的翻译质量。

4.3.6　翻译案例

在本节中，我们通过表4-7中的两个翻译案例来展示非自回归模型的译文存在的问题，以及我们的方法如何提升非自回归模型的翻译质量。其中，Source 表示原文，Target 表示参考译文，AT 是自回归模型的译文，Vanilla NAT 是非自回归基线模型的译文，Ours 是我们方法的译文。可以看到，非自回归基线模型 Vanilla NAT 存在过翻译和漏翻译的错误，尤其是在翻译长句子时译文会有很多重复词，如 “without without’’、“shadow shadow’’ 等。另外，译文的结构也是不完整的，缺少许多信息。这是由于词级交叉熵损失独立评估每个位置的准确性，没有对目标端的序列依赖进行建模，导致非自回归模型只关注局部的正确性而忽略了整体的翻译质量。相反，我们的方法能够建模目标端的序列依赖关系，损失函数与翻译质量的相关性强，训练得到的模型中译文中过翻译和漏翻译的错误明显减少，生成的译文更加流畅，翻译质量有显著改善。

4.4 本章小结

非自回归模型使用的交叉熵损失函数无法建模目标端的序列依赖关系，与翻译质量的相关性较弱。本章提出一种基于 n 元组词袋距离的损失函数，与翻译质量的相关性强，能够准确、高效地训练非自回归模型。实验结果显示，我们的方法能大幅减少译文中过翻译和漏翻译的错误，显著改善了模型的翻译质量。

表 4-7 在 WMT14 德英验证集上的两个翻译案例。

Table 4-7 Two translation cases in the validation set of WMT14 De-En.

Source	Es gibt Krebsarten , die aggressiv und andere , die indolent sind .
Target	There are aggressive cancers and others that are indolent .
AT	There are cancers that are aggressive and others that are indolent .
Vanilla NAT	There are cancers cancer aggressive aggressive others are indindent .
Ours	There are cancers that are aggressive and others that indolent .
Source	Wir wissen ohne den Schatten eines Zweifels , dass wir ein echtes neues Teilchen haben , und dass es dem vom Standardmodell vorausgesagten Higgs-Boson stark ähnelt .
Target	We know without a shadow of a doubt that it is a new authentic particle , and greatly resembles the Higgs boson predicted by the Standard Model .
AT	We know without the shadow of a doubt that we have a real new particle , and that it is very similar to the Higgs Boson predicted by the standard model .
Vanilla NAT	We know without without shadow shadow of doubt doubt that we have a new particle le that it is very similar similar to HiggsgsBoson predicted by the standard model .
Ours	We know without the shadow of a doubt that we have a real new particle and that it is very similar to the Higgs-Boson predicted by the standard model .

第 5 章　基于 n 元组匹配的非单调对齐建模

5.1　引言

非自回归神经机器翻译[8]（Non-Autoregressive Neural Machine Translation，NAT）能够并行生成所有译文单词，显著提升了神经机器翻译[1,7]的解码速度。然而，由于缺乏合适的训练目标，非自回归模型对解码速度的提升通常是以降低翻译质量为代价的。非自回归模型一般采用极大似然估计的训练方式，使用交叉熵作为损失函数。交叉熵损失要求模型输出与参考译文严格对齐，在位置上的轻微偏移就会导致损失函数大幅上升。然而，在机器翻译中，同一原文可能有多种正确的译文，这种多峰性问题导致模型输出难以与参考译文显式对齐，使交叉熵损失无法准确地评估非自回归模型的输出结果。

由于交叉熵损失显式严格对齐的要求难以满足，一些工作尝试放松这一要求来改进训练目标[25,69,70]。其中，基于联结主义时间分类器[25]（Connectionist Temporal Classification，CTC）的隐式对齐模型能够考虑所有与参考译文的单调对齐来建模翻译概率，并具有生成变长译文的能力，是目前备受关注的一种非自回归模型结构[24–27,50,111]。隐式对齐模型通常会对模型输出与参考译文之间的对齐做单调性假设，无法建模非单调的对齐关系。如图5-1所示，单调性假设在 CTC 经典的语音识别（Automatic Speech Recognition，ASR）应用场景中是适用的，因为语音输入和文本输出之间存在天然的单调映射。然而，在机器翻译任务中，单词之间是可以调整语序的，因此非单调对齐是广泛存在的现象。如图5-1所示，当参考译文为 “I ate pizza this afternoon’’ 但模型的输出结果是 “this afternoon I ate pizza’’ 时，尽管模型输出了语义正确的结果，但 CTC 无法处理这种非单调的对齐，会错误地惩罚这一输出结果。

针对这一问题，本章提出了为基于 CTC 的非自回归机器翻译模型建模非单调隐式对齐的方法。由于 CTC 的复杂对齐结构使我们难以直接建模非单调对齐，我们首先提出一种简化版的 CTC 模型，称其为 SCTC，再将其对齐空间扩展到非单调对齐中，以适应机器翻译中的词序调整现象。不基于单调对齐假设后，我们难以再通过动态规划算法来对所有对齐求和，只能根据匈牙利算法找到的最优对齐来优化模型[70,112–114]。如果不要求模型输出与参考译文之间完全对齐，而是希望它们之间对齐的比例较高，就可以用更灵活的方式建模非单调对齐。具体地，我们进一步扩展对齐空间，考虑所有与参考译文有交集的对齐，对其中的 n 元组与参考译文的 n 元组做非单调的匹配，训练 SCTC 模型来最大化 n 元组匹配的 F1 值。最后，我们回到 CTC 模型中，分析 SCTC 与 CTC 之间的差异，在 $n \leq 2$ 的情形下将在 SCTC 模型上的 n 元组匹配方法扩展到了 CTC 中。通过这种方式，我们能在表达能力更强的 CTC 模型上建模非单调隐式对齐，进一步提升了模型的能力。

我们在多个翻译任务（WMT14 En↔De，WMT16 En↔Ro）上对所提方法进

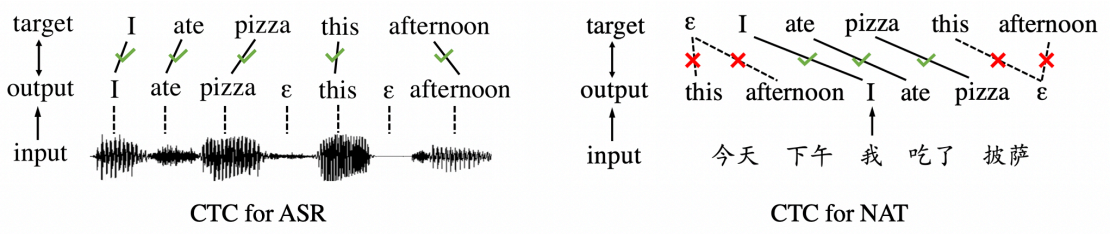


图 5-1 CTC 中单调对齐假设的示例：左图是 CTC 在语音识别任务上的应用，语音输入与文本输出之间存在天然的单调映射，右图是 CTC 在机器翻译任务上的应用，存在由语序调整导致的非单调对齐。ϵ 为空白词，表示什么都不输出。

Figure 5-1 Illustration of the monotonic alignment assumption of CTC: (left) CTC for ASR where there is a natural monotonic mapping between the speech input and ASR target, (right) CTC for NAT where there exists global word reordering that induces non-monotonic alignment. ϵ is the blank token that means 'output nothing'.

行了实验验证。结果表明，我们的方法在 SCTC 模型和 CTC 模型上都效果显著。在基于 n 元组匹配的训练目标优化下，CTC 模型仅进行单轮解码就达到甚至超越了自回归模型的性能水平，并且仍相对自回归模型有可观的解码加速。

5.2 研究背景

本章工作是围绕 CTC 翻译模型展开的。在本节中，我们将对相关研究背景做介绍，包括基础的非自回归机器翻译模型和基于 CTC 的非自回归机器翻译模型。

5.2.1 非自回归机器翻译

非自回归神经机器翻译模型[8] 能够并行生成所有译文单词，显著提升了模型的翻译速度。非自回归模型对目标端的概率分布做独立的建模，如下所示：

$$p(Y|X,\theta) = \prod_{t=1}^{T} p_t(y_t|X,\theta), \tag{5-1}$$

其中，X 为源端的句子，$Y = \{y_1, ..., y_T\}$ 是目标端的句子，$p_t(y_t|X,\theta)$ 表示在位置 t 的翻译概率。在训练时，模型解码器长度与参考译文一致，使用交叉熵损失训练模型，要求模型在位置 t 生成参考译文的对应词 y_t：

$$\mathcal{L}_{CE}(\theta) = -\sum_{t=1}^{T} \log(p_t(y_t|X,\theta)). \tag{5-2}$$

在解码时，基础的非自回归模型会使用一个长度预测器，以编码器的隐状态为输入来预测译文的句长。随后，模型再构建对应长度的解码器，预测每个位置的概率分布，通过 argmax 解码方法得到概率最高的译文：

$$\hat{y}_t = \text{argmax}_{y_t} p_t(y_t|X,\theta). \tag{5-3}$$

5.2.2　基于 CTC 的非自回归机器翻译

基础的非自回归模型有两个主要缺陷。第一个缺陷是交叉熵损失要求模型输出与参考译文严格对齐，但机器翻译中的多峰性导致模型输出难以与参考译文对齐，使交叉熵损失无法准确地评估模型的输出结果。第二个缺陷是模型需要先预测译文长度，再生成指定长度的译文。长度预测结果可能不准确，但模型也只能按这个长度生成译文，无法在解码时动态修改译文长度。

基于 CTC[20] 的非自回归机器翻译模型能够根据模型输出与参考译文的隐式对齐来建模翻译概率，并具有生成变长译文的能力，是目前备受关注的一种非自回归模型结构。CTC 一般会假设模型输出的序列比目标序列要长，并允许输出序列中包含空白词 ϵ。在非自回归机器翻译中，一般会在目标端的词表中加入一个空白词 ϵ，并移除长度预测器，直接令解码器的长度固定为源端长度的 λ 倍，其中 λ 为大于 1 的超参数。从形式上看，引入空白词 ϵ 后，模型的输出从译文空间 $\mathcal{Y}$ 扩展到了 $\mathcal{Y}^*$，从 $\mathcal{Y}^*$ 到 $\mathcal{Y}$ 有一种多对一的映射关系。其中，$\mathcal{Y}^*$ 一般被称为对齐空间[25,115]，因为其中的元素可以理解为是译文与模型输出之间的一种对齐。假设对齐长度为 T_A，我们定义函数 $\beta(Y)$ 为 $\mathcal{Y}^*$ 的子集，包含译文 Y 的所有长为 T_A 的对齐，称其为译文 Y 的对齐空间。它的反函数 $\beta^{-1}: \mathcal{Y}^* \mapsto \mathcal{Y}$ 为从对齐 A 到译文 Y 的映射，其中包含两个步骤，首先去除对齐 A 中的所有连续的重复词，再去除其中的所有空白词后得到译文 Y。如图5-2所示，对齐 $A = (\epsilon, a, a, \epsilon, a, b, b, c, \epsilon, d)$ 对应的译文为 $Y = \beta^{-1}(A) = (a, a, b, c, d)$，其中对齐 A 中的词会被单调地映射到译文 Y 中。CTC 模型同样采样极大似然估计的训练方式，通过动态规划算法对所有对齐的概率求和，并最大化对数似然损失：

$$\log p(Y|X,\theta) = \log \sum_{A \in \beta(Y)} p(A|X,\theta), \tag{5-4}$$

其中，对齐的概率 $p(A|X,\theta)$ 由非自回归 Transformer 模型建模：

$$p(A|X,\theta) = \prod_{t=1}^{T_A} p_t(a_t|X,\theta). \tag{5-5}$$

在解码时，一般直接使用 argmax 解码方法，在每一步预测概率最大的 a_t，并输出该对齐对应的译文 Y：

$$\hat{a}_t = \text{argmax}_{a_t} p_t(a_t|X,\theta), \quad Y = \beta^{-1}(A). \tag{5-6}$$

另外，也可以采用束搜索的方法，直接搜索概率较高的译文 Y。在束搜索时，也可以引入自回归的统计语言模型对译文打分[111,116]，束搜索的优化目标如下：

$$\log p(Y|X,\theta) + \alpha \cdot \log p_{LM}(Y) + \beta \cdot \log(|Y|), \tag{5-7}$$

其中，α 和 β 是对语言模型评分和长度惩罚项的权重。束搜索是串行的解码方法，因此会对模型的解码速度有一定影响，但由于在搜索过程中不需要做任何神经网络的计算，因此解码速度相较自回归模型仍有很大优势。

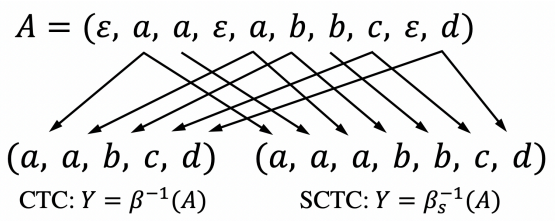


图 5-2 从对齐 A 映射到译文 Y 的示意图。CTC 的映射 β^{-1} 先移除对齐中的所有连续的重复词，再去除其中的所有空白词。SCTC 的映射 β_s^{-1} 仅移去对齐中的空白词 ϵ。

Figure 5-2 An example of target sentences obtained from the alignment A. The collapsing function of CTC removes repetitions first and then removes blanks. The collapsing function of SCTC β_s^{-1} only removes blanks in the alignment.

5.3 基于 CTC 模型的非单调对齐

在本节中，我们对基于 CTC 模型的非单调对齐建模方法做了深入研究，以弥补其仅支持单调对齐的缺陷。首先，我们在5.3.1节提出一种简化版的 SCTC 模型以简化分析。基于 SCTC 模型，我们在5.3.2节中提出了二部图匹配方法和 n 元组匹配方法来建模非单调对齐。随后，我们在5.3.3节回到 CTC 模型中，分析 SCTC 与 CTC 之间的差异，将在 SCTC 模型上的 n 元组匹配方法扩展到了 CTC 模型中。最后，我们在5.3.4节介绍了结合单调对齐与非单调对齐的训练策略。

5.3.1 SCTC 模型

在 CTC 中，函数 β^{-1} 通过两个操作将对齐 A 映射到译文 Y：（1）去除对齐 A 中的所有连续的重复词，（2）去除所有空白词 ϵ。这两个操作结合在一起，使得译文的对齐空间 $\beta(Y)$ 变得比较复杂。因此，我们首先提出一种简化版的 CTC 模型，称其为 SCTC（Simplified CTC），它的对齐空间结构更简单，有助于我们分析、推导出非单调隐式对齐的建模方法。另外，在 SCTC 上得出的结果也对我们最后在 CTC 模型上的分析有很大帮助。

为了简化对齐空间的结构，我们考虑仅使用一种操作来将对齐 A 映射到译文 Y。一种方式是仅去除对齐 A 中的所有连续的重复词，它的缺陷是会限制模型的表达能力。例如，由于译文 $Y = (a, a, b, c, d)$ 中包含连续的重复词，所有对齐都无法被映射到 Y，它的翻译概率将始终是 0。另一种方式是仅去除对齐 A 中的空白词，这种方式更为合理，模型的表达能力保持不变。另外，对齐空间也变得更加简单，译文 Y 的对齐中包含的词是确定的，一定是 Y 中所有词与 $T_A - T$ 个空白词的组合。

我们将这种简化后的模型称为 SCTC，并在图5-2中展示了 CTC 与 SCTC 之间的差异。在 SCTC 的背景下，我们将译文 Y 的对齐空间写作 $\beta_s(Y)$，从对齐 A 到译文 Y 的映射写作 β_s^{-1}。类似地，我们可以通过动态规划算法求得翻译概率 $p(Y|X, \theta)$。我们将前向变量 $\alpha_t(s)$ 定义为从前 t 个对齐位置 $a_{1:t}$ 到前 s 个译文单

词 $y_{1:s}$ 的翻译概率，它可以由 $\alpha_{t-1}(s)$ 和 $\alpha_{t-1}(s-1)$ 迭代地计算而来：

$$\alpha_t(s) = \alpha_{t-1}(s-1)p_t(y_s|X,\theta) + \alpha_{t-1}(s)p_t(\epsilon|X,\theta). \tag{5-8}$$

通过动态规划算法得到的前向变量 $\alpha_{T_A}(T)$ 就是所需的翻译概率 $p(Y|X,\theta)$。因此，我们同样能使用交叉熵损失训练 SCTC 模型：

$$\mathcal{L}_{SCTC}(\theta) = -\log \alpha_{T_A}(T). \tag{5-9}$$

5.3.2　基于 SCTC 的非单调对齐

我们首先在 SCTC 下探索非单调隐式对齐的建模方法，具体包括二部图匹配方法和 n 元组匹配方法。

5.3.2.1　二部图匹配

为了建模非单调对齐，我们将单调的对齐空间 $\beta_s(Y)$ 扩展为非单调的对齐空间 $\gamma_s(Y)$，定义如下：

$$\gamma_s(Y) = \bigcup_{Y' \in \Pi(Y)} \beta_s(Y'), \tag{5-10}$$

其中，$\Pi(Y)$ 表示译文 Y 中词语的所有排列方式的集合，例如译文 (A,B) 的所有排列为 $\Pi((A,B)) = \{(A,B),(B,A)\}$。通过枚举 $\Pi(Y)$，我们考虑了译文的所有词序调整的可能性来构建非单调对齐空间 $\gamma_s(Y)$。理想情况下，我们希望能遍历 $\gamma_s(Y)$ 中的所有对齐来计算如下损失函数：

$$\mathcal{L}_{sum}(\theta) = -\log \sum_{A \in \gamma_s(Y)} p(A|X,\theta). \tag{5-11}$$

然而，由于对齐空间不再具有单调性，我们无法再应用动态规划算法来对所有对齐求和，很难准确地计算上式的损失函数。因此，我们转而去最小化上式的一个上界，以最优对齐的对数概率作为损失：

$$\mathcal{L}_{max}(\theta) = -\log \max_{A \in \gamma_s(Y)} p(A|X,\theta). \tag{5-12}$$

尽管对所有对齐的概率求和存在较大困难，但找出最优对齐是可以在多项式时间内实现的。参考过去的一些工作[70,112,114]，我们将寻找最优对齐的问题转换为二部图匹配问题，并应用匈牙利算法[113]来在 $\mathcal{O}(T^3)$ 时间内求解。具体地，我们不难发现非单调的对齐空间 $\gamma_s(Y)$ 的结构较为简单，是由 Y 中所有词与 $T_A - T$ 个空白词的所有排列组成的对齐空间。因此，我们可以将这些词与模型的预测结果看作两组数目为 T_A 的节点，两组节点之间通过边来连接，边的权重为预测概率的对数值。例如，第 t 个预测位置与词 w 之间的边的权重为 $\log p_t(w|X,\theta)$。寻找最优对齐等价于找到两组节点之间的最优匹配，最优匹配中的边的权重和就是式5-12的值。

5.3.2.2 n 元组匹配

由于对齐空间不再具有单调性，我们无法再应用动态规划算法来对所有对齐求和，很难准确地计算式5-11。另外，这种困难也是因为我们要求对齐与译文严格匹配，导致难以对求和做变换来简化计算。在实践中，由于机器翻译中存在多峰性，严格匹配是较难满足的要求。如果不要求模型输出与参考译文之间完全对齐，而是希望它们之间对齐的比例较高，就可以用更灵活的方式建模非单调对齐。具体地，我们进一步扩展对齐空间，考虑所有与参考译文有交集的对齐。参考机器翻译评估指标以 n 元组匹配准确率评价翻译质量的方式，我们将对齐与参考译文中的 n 元组做非单调的匹配，训练基于隐式对齐的 SCTC 模型来最大化 n 元组匹配的准确率。参照概率化匹配的思想[99]，我们引入概率化 n 元组来使训练目标变得可导。

我们用 $C_g(Y)$ 来表示 n 元组 $g=(g_1,...,g_n)$ 在译文 Y 中的出现次数，用 $C_g^s(\theta)$ 表示源端输入为 X、参数为 θ 的模型中的概率化 n 元组计数，符号 $C_g^s(\theta)$ 中省略了源句 X 以简化表示，上标 s 表示 SCTC。概率化 n 元组计数定义为所有译文中 n 元组 g 数目的加权求和，如下所示：

$$C_g^s(\theta)=\sum_{Y\in\mathcal{Y}}p(Y|X,\theta)C_g(Y)=\sum_{A\in\mathcal{Y}^*}p(A|X,\theta)C_g(\beta_s^{-1}(A)). \tag{5-13}$$

$C_g^s(\theta)$ 的计算方法是本节的核心部分，将在后面详细介绍。我们使用 $M_g^s(\theta)$ 来表示模型与参考译文之间 n 元组 g 的匹配数（同样省略了 Y 以简化表示），定义如下：

$$M_g^s(\theta)=\min(C_g(Y),C_g^s(\theta)). \tag{5-14}$$

我们的训练目标是最大化模型输出与参考译文之间 n 元组的非单调匹配，因此我们用 G_n 来表示所有 n 元组的集合，以 n 元组匹配的准确率或召回率来训练模型：

$$P_n^s(\theta)=\frac{\sum_{g\in G_n}M_g^s(\theta)}{\sum_{g\in G_n}C_g^s(\theta)},\quad R_n^s(\theta)=\frac{\sum_{g\in G_n}M_g^s(\theta)}{\sum_{g\in G_n}C_g(Y)}. \tag{5-15}$$

然而，隐式对齐模型具有动态控制译文长度的能力，减小译文长度一般就能获得更高的准确率，增大译文长度就有更高的召回率。因此，无论是上式中的准确率还是召回率，都无法准确地反映翻译质量。因此，我们结合两者，令训练目标为 n 元组匹配的 F1 值：

$$\mathcal{L}_n^s(\theta)=-\mathrm{F1}_n^s(\theta)=-\frac{2\cdot\sum_{g\in G_n}M_g^s(\theta)}{\sum_{g\in G_n}(C_g(Y)+C_g^s(\theta))}. \tag{5-16}$$

在上式的分子中，$M_g^s(\theta)$ 为 n 元组 g 的匹配数，仅当 $C_g(Y)$ 非零，即 n 元组 g 出现在参考译文中时取非零值。因此，我们只需考虑参考译文中的 n 元组 g，只要能有计算 $C_g^s(\theta)$ 的高效算法，就能计算出上式的分子。在分母中，$\sum_{g\in G_n}C_g(Y)$ 的值为常数 $T-n+1$，我们只需找到另一项 $\sum_{g\in G_n}C_g^s(\theta)$ 的计算方法。总的来说，

我们需要找出计算 $C_g^s(\theta)$ 和 $\sum_{g\in G_n} C_g^s(\theta)$ 的高效算法，才能将式5-16作为模型的损失函数。

分子计算　为了计算式5-16中的分子，我们需要找出计算 $C_g^s(\theta)$ 的高效算法。$C_g^s(\theta)$ 的定义为 n 元组 g 的期望出现次数，需要对所有译文遍历求和，搜索空间是指数级的。因此，我们首先需要对式5-13做变换来减小其搜索空间。具体地，我们证明了如下定理成立：

定理 5.1.

$$C_g^s(\theta) = \sum_{i_1=1}^{T_A}\sum_{i_2=1}^{T_A}\dots\sum_{i_n=1}^{T_A} p_{i_1}(g_1|X,\theta) \times \prod_{k=2}^{n} E_{i_{k-1},i_k} p_{i_k}(g_k|X,\theta),\ g\in G_n, \tag{5-17}$$

其中 E 是大小为 $T_A \times T_A$ 的矩阵，E_{ij} 表示从位置 i 转移到位置 j 的概率，定义如下：

$$E_{i,j} = \begin{cases} 0, & j < i+1 \\ 1, & j = i+1 \\ \prod_{t=i+1}^{j-1} p_t(\epsilon|X,\theta), & j > i+1 \end{cases}. \tag{5-18}$$

证明. 在式5-13中，我们对 $C_g^s(\theta)$ 做了如下定义：

$$C_g^s(\theta) = \sum_{A\in\mathcal{Y}^*} p(A|X,\theta) C_g(\beta_s^{-1}(A)).$$

在上式中，$C_g(\beta_s^{-1}(A))$ 为 n 元组 g 在译文 $\beta_s^{-1}(A)$ 中的出现次数。我们用 $i_{1:n} = \{i_1,\dots,i_n\}$ 来表示 n 元组 $g = \{g_1,\dots,g_n\} \in G_n$ 在对齐中可能的出现位置。在位置 $i_{1:n}$ 上有一个 n 元组 g 等价于 $a_{i_k} = g_k$ 对所有 $k \in \{1,\dots,n\}$ 都成立，且这之间的所有其他词均为空白词 ϵ。通过遍历所有可能的 $i_{1:n}$，我们可以将 $C_g(\beta_s^{-1}(A))$ 写作如下形式：

$$C_g(\beta_s^{-1}(A)) = \sum_{i_1=1}^{T_A}\sum_{i_2=1}^{T_A}\dots\sum_{i_n=1}^{T_A} 1(a_{i_1}=g_1) \times \prod_{k=2}^{n} 1(i_{k-1}<i_k, a_{i_k}=g_k, a_{i_{k-1}+1:i_k-1}=\epsilon), g\in G_n.$$

其中，$1(\cdot)$ 是指示函数，若括号内的条件满足，则其取值为 1，否则值为 0。回顾我们对转移矩阵 E 的定义，E_{ij} 表示从位置 i 转移到位置 j 的概率，满足如下条件：

$$E_{i_{k-1},i_k} = p(i_{k-1}<i_k, a_{i_{k-1}+1:i_k-1}=\epsilon|X,\theta).$$

根据以上分析，我们可以对 $C_g^s(\theta)$ 做如下变换：

$$\begin{aligned}
C_g^s(\theta) &= \sum_A p(A|X,\theta) C_g(\beta_s^{-1}(A)) \\
&= \sum_A p(A|X,\theta) \sum_{i_1=1}^{T_A} \sum_{i_2=1}^{T_A} \dots \sum_{i_n=1}^{T_A} 1(a_{i_1} = g_1) \times \prod_{k=2}^{n} 1(i_{k-1} < i_k, a_{i_k} = g_k, a_{i_{k-1}+1:i_k-1} = \epsilon) \\
&= \sum_{i_1=1}^{T_A} \sum_{i_2=1}^{T_A} \dots \sum_{i_n=1}^{T_A} \sum_A p(A|X,\theta) 1(a_{i_1} = g_1) \times \prod_{k=2}^{n} 1(i_{k-1} < i_k, a_{i_k} = g_k, a_{i_{k-1}+1:i_k-1} = \epsilon) \\
&= \sum_{i_1=1}^{T_A} \sum_{i_2=1}^{T_A} \dots \sum_{i_n=1}^{T_A} p(a_{i_1} = g_1|X,\theta) \times \prod_{k=2}^{n} p(a_{i_k} = g_k|X,\theta) p(i_{k-1} < i_k, a_{i_{k-1}+1:i_k-1} = \epsilon) \\
&= \sum_{i_1=1}^{T_A} \sum_{i_2=1}^{T_A} \dots \sum_{i_n=1}^{T_A} p_{i_1}(g_1|X,\theta) \times \prod_{k=2}^{n} E_{i_{k-1},i_k} p_{i_k}(g_k|X,\theta), g \in G_n.
\end{aligned}$$

□

根据定理5.1，我们可以将计算 $C_g^s(\theta)$ 的指数级时间复杂度降低到 $\mathcal{O}(nT_A^n)$ 的多项式复杂度，在 n 较小时能够高效地计算 $C_g^s(\theta)$。下面，我们给出其在 n 的取值为 1 和 2 时的计算方法：

$$C_g^s(\theta) = \sum_{t=1}^{T_A} p_t(g_1|X,\theta), \quad g \in G_1. \tag{5-19}$$

$$C_g^s(\theta) = \sum_{i=1}^{T_A} \sum_{j=1}^{T_A} p_i(g_1|X,\theta) E_{i,j} p_j(g_2|X,\theta), \quad g \in G_2. \tag{5-20}$$

然而，对于 $n > 2$ 的情况，计算 $C_g^s(\theta)$ 的时间复杂度仍较高，是训练速度的瓶颈所在。在这里，我们进一步给出将复杂度从 $\mathcal{O}(nT_A^n)$ 降低到 $\mathcal{O}(nT_A^2)$ 的方法，使我们对任意 n 都能高效地计算出其概率化 n 元组计数 $C_g^s(\theta)$。观察式5-17的形式，可以发现它与马尔可夫过程的 n 步转移概率比较相似，E 即为马尔可夫转移矩阵，不同点在于每步转移后还要乘以一个当前状态的权重 $p_{i_k}(g_k|X,\theta)$。因此，我们也可以调整计算 n 步转移概率的方法来简化式5-17的计算。具体地，我们引入一系列状态向量 $s_{1:n}$，每个向量 s_k 的维度为 T_A。我们令每个 s_k 由 s_{k-1} 迭代地计算而来，如下所示：

$$s_k(i_k) = \begin{cases} p_{i_1}(g_1|X,\theta), & k = 1 \\ \sum_{i_{k-1}=1}^{T_A} s_{k-1}(i_{k-1}) E_{i_{k-1},i_k} p_{i_k}(g_k|X,\theta), & k > 1 \end{cases}. \tag{5-21}$$

根据以上定义从 s_{k-1} 计算 s_k 的时间复杂度为 $\mathcal{O}(T_A^2)$，因此计算 s_n 的时间复杂度为 $\mathcal{O}(nT_A^2)$。在下面的定理中，我们证明了式5-17即为向量 s_n 中所有维度的求和，因此计算式5-17的时间复杂度可以降低为 $\mathcal{O}(nT_A^2)$。

定理 5.2.

$$C_g^s(\theta) = \sum_{i_n=1}^{T_A} s_n(i_n), g \in G_n. \tag{5-22}$$

证明.

$$
\begin{aligned}
C_g^s(\theta) &= \sum_{i_1=1}^{T_A}\sum_{i_2=1}^{T_A}\cdots\sum_{i_n=1}^{T_A} p_{i_1}(g_1|X,\theta)\times\prod_{k=2}^{n} E_{i_{k-1},i_k}p_{i_k}(g_k|X,\theta) \\
&= \sum_{i_n=1}^{T_A}\cdots\sum_{i_2=1}^{T_A}\sum_{i_1=1}^{T_A} s_1(i_1)\prod_{k=2}^{n} E_{i_{k-1},i_k}p_{i_k}(g_k|X,\theta) \\
&= \sum_{i_n=1}^{T_A}\cdots\sum_{i_3=1}^{T_A}\sum_{i_2=1}^{T_A} s_2(i_2)\prod_{k=3}^{n} E_{i_{k-1},i_k}p_{i_k}(g_k|X,\theta) \\
&= \cdots\cdots \\
&= \sum_{i_n=1}^{T_A}\sum_{i_{n-1}=1}^{T_A} s_{n-1}(i_{n-1})\times E_{i_{n-1},i_n}p_{i_n}(g_n|X,\theta) \\
&= \sum_{i_n=1}^{T_A} s_n(i_n), g\in G_n.
\end{aligned}
\tag{5-23}
$$

□

分母计算　在上面，我们给出了计算式5-16中分子的高效算法。在这里，我们对分母的计算方法进行分析，即如何高效地计算 $\sum_{g\in G_n} C_g^s(\theta)$。我们首先考虑 $n=1$ 的简单情况，此时的求和可以直接写作如下形式，即对每个位置预测的概率分布求和：

$$
\sum_g C_g^s(\theta) = \sum_{t=1}^{T_A}\sum_g p_t(g_1|X,\theta), g\in G_1. \tag{5-24}
$$

对于 $n>1$ 的情况，直接求和所需的计算代价就太大了，但我们也可以采用间接的计算方式。由于 $C_g^s(\theta)$ 定义为 n 元组 g 的期望出现次数，$\sum_g C_g^s(\theta)$ 就是句子中 n 元组的期望出现次数。每个句子中 n 元组的数目总是比 1 元组数目小 $n-1$，因此我们可以基于 $n=1$ 的结果直接得到对 $n>1$ 情形的求和结果：

$$
\sum_{g\in G_n} C_g^s(\theta) = \sum_{g\in G_1} C_g^s(\theta) - n + 1. \tag{5-25}
$$

5.3.3　基于 CTC 的非单调对齐

在上一节中，我们为基于隐式对齐的 SCTC 模型提出了两种建模非单调隐式对齐的方法，分别为二部图匹配和 n 元组匹配。在本节中，我们尝试将这些方法扩展到 CTC 模型中，使 CTC 模型也具有建模非单调对齐的能力。

在 SCTC 模型中，非单调对齐空间 $\gamma_s(Y)$ 有比较简单的结构，是由 Y 中所有词与 T_A-T 个空白词的所有排列组成的对齐空间，所以能将寻找最优对齐的问题转换为二部图匹配问题，并应用匈牙利算法来求解该问题。然而，在 CTC 模型中，对齐内连续的重复词也会被去除，这导致对齐空间 $\gamma_s(Y)$ 中对齐包含的词可能不同，这导致我们无法应用匈牙利算法来求解最优对齐。因此，二部图匹配方法在 CTC 模型中的应用存在较大的阻碍。

幸运的是，我们在 SCTC 模型上的 n 元组匹配方法能够帮助我们推导出 CTC 模型下的 n 元组匹配方法。CTC 与 SCTC 的差异仅为它们将同一个对齐 A 映射为了不同的译文 Y。相较于 CTC 的译文 $\beta^{-1}(A)$，SCTC 的译文 $\beta_s^{-1}(A)$ 中只是包含了更多的重复词。我们可以先确定这些重复词的影响有多大，然后从 SCTC 的结果中去除这种影响。首先，我们对 CTC 下的概率化 n 元组计数做正式定义：

$$C_g(\theta) = \sum_{A \in \mathcal{Y}^*} p(A|X,\theta) C_g(\beta^{-1}(A)). \tag{5-26}$$

我们定义 $R_g(\theta)$ 为 $n=1$ 时 $C_g(\theta)$ 和 $C_g^s(\theta)$ 之间的差异，表示 CTC 中多删除的重复词数量。下面的定理给出了 $R_g(\theta)$ 的计算方法：

定理 5.3.

$$R_g(\theta) = C_g^s(\theta) - C_g(\theta) = \sum_{t=1}^{T_A-1} p_t(g_1|X,\theta) p_{t+1}(g_1|X,\theta), g \in G_1. \tag{5-27}$$

证明. 证明上述定理等价于证明以下式子成立：

$$C_g(\theta) = p_1(g_1|X,\theta) + \sum_{t=2}^{T_a} p_t(g_1|X,\theta)(1 - p_{t-1}(g_1|X,\theta)), g \in G_1.$$

$C_g(\theta)$ 的定义为 $C_g(\beta^{-1}(A))$ 的期望值，其中 β^{-1} 为从对齐到译文的映射，首先移除对齐中连续的重复词，再移除所有的空白词。因此，若对齐中一个位置的预测结果为 g_1，且它前一个位置不为 g_1，它就不会被映射 β^{-1} 移除。因此，我们将其称为有效词，并将 $C_g(\beta^{-1}(A))$ 写作有效词的总数：

$$C_g(\beta^{-1}(A)) = 1(a_1 = g_1) + \sum_{t=2}^{T_A} 1(a_t = g_1, a_{t-1} \neq g_1), g \in G_1.$$

基于此，我们可以将 $C_g(\theta)$ 写作如下形式，即证明了本定理：

$$\begin{aligned} C_g(\theta) &= \sum_A p(A|X,\theta) C_g(\beta^{-1}(A)) = p(a_1 = g_1|X,\theta) + \sum_{t=2}^{T_A} p(a_t = g_1, a_{t-1} \neq g_1|X,\theta) \\ &= p_1(g_1|X,\theta) + \sum_{t=2}^{T_A} p_t(g_1|X,\theta)(1 - p_{t-1}(g_1|X,\theta)), g \in G_1. \end{aligned}$$

□

上面的定理已经解决了 $n=1$ 时的情况，使我们能高效地计算 $C_g(\theta)$ 和 $\sum_g C_g(\theta)$。对于 $n=2$ 的情况，我们将二元组 $g=(g_1,g_2)$ 分为 $g_1=g_2$ 和 $g_1 \neq g_2$ 两类。对连续重复词的去除只会减少第一类二元组 $g_1=g_2$ 的数目，每个被移除的词对应一个被移除的二元组，因此差值为 $R_g(\theta)$。第二类二元组 $g_1 \neq g_2$ 的数目是保持不变的，以图5-2为例，对同一个对齐 $A=(\epsilon,a,a,\epsilon,a,b,b,c,\epsilon,d)$，SCTC 的输出仅比 CTC 的输出多了两个第一类的二元组 $\{(a,a),(b,b)\}$，第二类二元组

的数目完全相同。因此，我们可以直接对 $n = 2$ 的情况写出 $C_g(\theta)$ 的值，如下所示：

$$C_g(\theta) = \begin{cases} C_g^s(\theta) - R_g(\theta), & g_1 = g_2 \\ C_g^s(\theta), & g_1 \neq g_2 \end{cases}, g \in G_2. \tag{5-28}$$

对于 $n > 2$ 的情况，$C_g(\theta)$ 与 $C_g^s(\theta)$ 之间就没有这么简单的关系了。即便我们直接像定理5.1一样对其作变换，它的形式也相较于 SCTC 更加复杂，并且定理5.2中 $\mathcal{O}(nT_A^2)$ 复杂度的算法也不再适用。因此，在 CTC 模型下，我们不再考虑 $n > 2$ 情形下的 n 元组匹配。最后，我们像式5-14一样定义 CTC 模型的匹配数 $M_g(\theta)$，并将 n 元组匹配的 F1 值作为模型的训练目标：

$$\mathcal{L}_n(\theta) = -\frac{2 \cdot \sum_{g \in G_n} M_g(\theta)}{\sum_{g \in G_n} (C_g(Y) + C_g(\theta))}. \tag{5-29}$$

5.3.4　训练策略

在前面几节中，我们提出了简化版的 SCTC 模型，并在 CTC 模型和 SCTC 模型上都探究了建模非单调对齐的方法。这些方法也需要与基于单调对齐的交叉熵损失结合，否则模型可能会生成例如 “pizza ate I this afternoon’’ 这样的不考虑词序的译文。因此，我们首先用交叉熵损失预训练 SCTC 或 CTC 模型，再对模型进行微调以建模非单调对齐。SCTC 模型可以用二部图匹配和 n 元组匹配两种方法来微调，CTC 模型只能用 n 元组匹配方法来微调，并且 n 的取值范围为 $\{1, 2\}$。

5.4　实验结果与分析

在本节中，我们将通过实验来验证非单调对齐方法的实际效果，首先介绍相关的实验设置，再给出主要的实验结果，最后对所得结果进行分析。

5.4.1　实验设置

数据集　我们在非自回归机器翻译最常用的四个公共数据集上验证了所提的方法：WMT14 英语 ↔ 德语（En↔De，约 450 万个句对）、WMT16 英语 ↔ 罗马尼亚语（En↔Ro，约 61 万个句对）。对于 WMT14 英德数据集，我们将 *newstest2013* 和 *newstest2014* 作为验证集和测试集。对于 WMT16 英罗数据集，我们将 *newsdev-2016* 和 *newstest-2016* 作为验证集和测试集。我们用 32K 次操作的联合 BPE 模型[100] 来切分源端和目标端词汇，并在源端和目标端共享词表。我们使用 BLEU 值[79] 来评估模型的翻译质量。

知识蒸馏　序列级知识蒸馏[12,13] 是非自回归模型中的一项关键技术。在所有翻译任务中，我们都应用序列级知识蒸馏技术，首先训练一个自回归模型作为教师，再令它翻译一遍训练数据集的源端，将原文与译文组成句对来构建蒸馏数据集。最后，我们使用蒸馏数据集来训练非自回归模型。

实现细节 我们将基础设置的 Transformer 模型[1] 作为自回归的基线模型。CTC 模型也采用 Transformer 的模型结构，并用平均拷贝的方法[8] 来构建解码器的输入。我们将 CTC 中的过采样比例 λ 设置为 3，即解码器的长度是编码器的三倍。对于 WMT14 英德数据集，dropout 比例在预训练时设为 0.2，在微调时设为 0.1。对于 WMT16 英罗数据集，dropout 比例均设为 0.3。在预训练阶段，WMT14 英德数据集的训练步数为 30 万步，WMT16 英罗数据集的训练步数为 15 万步。在微调阶段，训练步数均为 6000 步。在预训练阶段，batch 大小为 6.4 万词，在微调阶段，batch 大小为 25.6 万词。我们用 Adam 优化器[101] 来优化模型，其参数为 $\beta = (0.9, 0.98)$、$\epsilon = 10^{-8}$。学习率在 1 万步内线性提升至 $5 \cdot 10^{-4}$，之后按倒数平方根规则衰减。我们用 8 张 GeForce RTX 3090 GPU 卡来训练模型，用单张卡在 batch 大小为 1 的设置下测量解码速度。我们在开源代码框架 `fairseq`[117] 上实现我们的模型。

解码 对于自回归模型，我们使用束搜索进行解码，束大小设置为 5。对于基础的非自回归模型，我们用 argmax 解码方法来生成译文。对于 CTC 模型，我们可以使用 argmax 解码方法，预测概率最大的对齐，再生成对应译文。另外，我们也采用束搜索的方法，束大小设置为 20，直接搜索概率较高的译文 Y，并引入自回归的统计语言模型对译文打分[111,116]。束搜索过程中不需要做任何神经网络的计算，已有基于 C++ 的高效实现[1]，因此解码速度相较自回归模型仍有很大优势。

5.4.2 主要实验结果

我们首先在 WMT14 英德数据集上进行实验，比较基于非单调隐式对齐模型的不同训练目标的性能。我们使用的基线模型包括自回归 Transformer 模型、基础非自回归模型 Aanilla-NAT、SCTC 模型和 CTC 模型。我们用二部图匹配方法和 n 元组匹配方法微调 SCTC 模型，用 n 元组匹配方法微调 CTC 模型。在表5-1中，我们给出了这些方法在 WMT14 英德测试集上的 BLEU 值，以及它们相对于自回归模型的加速比。

从表5-1的实验结果中，我们有以下的主要发现：（1）基于非单调对齐的训练目标能有效提升隐式对齐模型的性能，在 SCTC 模型上最多提升了 1.53 BLEU，在 CTC 模型上也有 1.23 BLEU 的改进，说明了建模非单调对齐的重要性。（2）SCTC 模型在非自回归机器翻译任务上的表现不如 CTC 模型。我们推测这是由对齐空间分布的差异导致的，由于大部分译文中不包含重复词，在这种情况下，SCTC 的对齐空间 $\beta_s^{-1}(Y)$ 是 $\beta^{-1}(Y)$ 的子集，这可能导致 SCTC 在机器翻译任务上的建模能力弱于 CTC。（3）在 SCTC 上，n 元组匹配方法在 $n = 2$ 时达到了最优的性能。回顾 n 元组匹配方法在 CTC 上无法处理 $n > 2$ 情形的局限性，这一发现表明我们不需要用较大的 n 来训练模型，也降低了这一局限性的影响。因此，我们在后面的实验中可以继续使用 BLEU 来评估翻译质量。

[1]https://github.com/parlance/ctcdecode

表 5-1 我们的方法与基线模型在 WMT14 英德测试集上的性能对比。
Table 5-1 Performance comparison between baselines and our methods on WMT14 En-De test set.

	Model	BLEU	Speed
Base	Transformer	27.54	1.0×
	Vanilla-NAT	19.32	15.5×
SCTC	SCTC	25.26	14.7×
	+bipartite	26.13	
	+1-gram	26.22	
	+2-gram	**26.79**	
	+3-gram	26.66	
	+4-gram	26.62	
CTC	CTC	26.34	14.7×
	+1-gram	27.16	
	+2-gram	**27.57**	

我们将表5-1中效果最好的方法二元组微调方法简写为NMLA（Non-Monotonic Latent Alignments），用 AT 表示自回归模型，**Iter.** 表示模型迭代的轮数，‘12-1’ 表示 12 层编码器和 1 层解码器的模型结构[118]，KD 表示知识蒸馏，在表5-2中将 NMLA 方法的性能与之前的模型进行了对比。可以看到，CTC 模型本身就具有较好的翻译质量，仅进行单轮解码就接近了其他迭代式模型的性能。这主要是源于 CTC 模型对单调对齐的建模，以及它能够动态决定译文长度的灵活性。通过建模非单调隐式对齐，NMLA 进一步将 CTC 模型的性能提升了约 1 BLEU 值，在各个数据集上都达到了与自回归 Transformer 模型相当的水平，并仍然有十倍以上的加速比。结合束搜索和语言模型后，NMLA 仅做一次解码的性能就能超越了自回归 Transformer 模型与其他的迭代式非自回归模型，表明了这一方法的有效性。

5.4.3　训练速度

相比于基础的 CTC 模型，NMLA 方法引入的额外训练代价为基于 n 元组匹配的模型微调。我们已经在定理5.1中对 NMLA 方法的时间复杂度做了理论分析。在这里，我们通过实验来测量它实际的训练速度。我们基于 8 张 GeForce RTX 3090 GPU 的设置来在 WMT14 英德数据集上测量 CTC 和 NMLA 的训练速度，用 ‘WPS’ 表示每秒处理的词数，‘Step’ 表示训练步数，‘Batch’ 表示训练的 batch 大小，‘Time’ 表示需要的训练时间，实验结果如表5-3所示。可以看到，NMLA 的 WPS 约为 CTC 的一半，这是因为 NMLA 的损失函数计算更加耗时。然而，由于我们只进行小步数的微调，NMLA 方法的整体训练时间并不高，不

表 5-2 本章方法与之前工作的性能对比。

Table 5-2 Performance comparison between our models and existing methods.

	Models	Iter.	Speed	WMT14		WMT16	
				EN-DE	DE-EN	EN-RO	RO-EN
AT	Transformer (teacher)	T	1.0×	27.54	**31.57**	**34.26**	**33.87**
	Transformer (12-1)	T	2.6×	26.09	30.30	32.76	32.39
	+ KD	T	2.7×	**27.61**	31.48	33.43	33.50
Non-CTC NAT	NAT-FT[8]	1	15.6×	17.69	21.47	27.29	29.06
	NAT-REG[65]	1	27.6×	20.65	24.77	–	–
	AXE[69]	1	–	23.53	27.90	30.75	31.54
	GLAT[50]	1	15.3×	25.21	29.84	31.19	32.04
	AligNART[47]	1	13.4×	26.40	30.40	32.50	33.10
	CMLM[35]	10	–	27.03	30.53	33.08	33.31
	LevT[62]	2.05	4.0×	27.27	–	–	33.26
	JM-NAT[59]	10	–	27.69	**32.24**	33.52	33.72
	RewriteNAT[63]	2.70	–	**27.83**	31.52	**33.63**	**34.09**
	CMLMC[119]	10	–	28.37	31.41	34.57	34.13
CTC-based NAT	CTC[24]	1	–	16.56	18.64	19.54	24.67
	Imputer[25]	1	–	25.80	28.40	32.30	31.70
	GLAT+CTC[50]	1	14.6×	26.39	29.54	32.79	33.84
	REDER[27]	1	15.5×	26.70	30.68	33.10	33.23
	+ beam&reranking	1	5.5×	27.36	31.10	33.60	34.03
	DSLP[28]	1	14.8×	27.02	31.61	34.17	**34.60**
	Fully-NAT[26]	1	16.8×	27.20	31.39	33.71	34.16
	Imputer[25]	8	–	**28.20**	**31.80**	**34.40**	34.10
Our work	CTC w/o finetune	1	14.7×	26.34	29.58	33.45	33.32
	CTC w/ NMLA	1	14.7×	27.57	31.28	33.86	33.94
	+ beam&lm	1	5.0×	**28.35**	**32.27**	**34.72**	**34.95**

到预训练阶段的 0.2 倍。相比 NMLA 方法带来的可观性能提升，这种在训练时间上的小幅增加是可以承受的。

表 5-3 CTC 预训练和 NMLA 微调之间的训练时间比较。

Table 5-3 The comparison of training time between CTC pretraining and NMLA finetuning.

	WPS	Step	Batch	Time
CTC pretraining	141K	300K	64K	26.4h
NMLA finetuning	60K	6K	256K	5.0h

5.4.4　解码速度

基于 CTC 的模型普遍用过采样方法来构建解码器，因此解码器的长度会比基础的非自回归模型更长，需要的计算量更大，这是这类模型的一个缺陷。在逐句解码时，由于硬件设备的并行计算能力较强，这种缺陷并不明显。但当解码时的 batch 大小较大时，这种缺陷可能会降低解码速度，对模型的加速比造成影响。针对这一问题，我们测量了不同模型解码 WMT14 英德测试集时，在不同 batch 大小下的解码速度，并在图5-3中展示了这一结果。由于显存限制，我们使用的最大 batch 大小为 400。可以看到，NMLA 方法在最大的 batch 大小下仍相对自回归模型有 2.4 倍的解码加速，并且翻译质量与自回归模型相当，NMLA+beam&lm 方法在解码速度和翻译质量上都优于自回归模型。这说明过采样带来的计算量更高的缺陷确实有一定影响，但仍不改变这类模型在解码速度上相对于自回归模型的优越性。

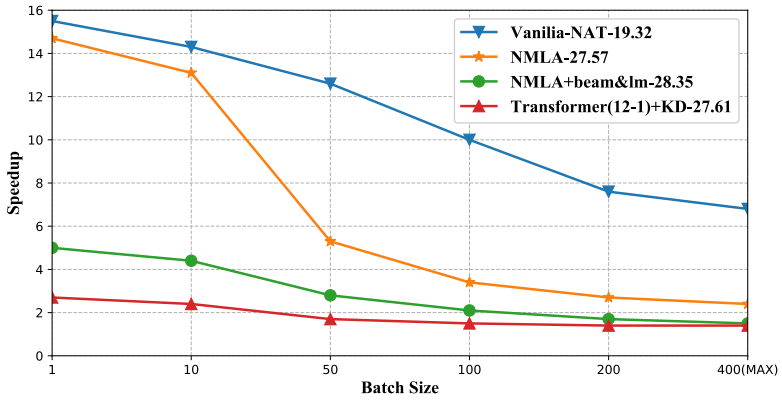


图 5-3 各类非自回归与自回归模型的 BLEU 值以及在不同 batch 大小下的解码加速比。

Figure 5-3 BLEU and speedup of NAT and AT models measured under different batch sizes.

5.4.5　结果分析

在本节中，我们对模型生成的结果做定量分析，以更好地理解 NMLA 方法给模型带来的改变。在以下实验中，我们从不同方面分析了模型在 WMT14 英德

测试集上的输出结果。

NMLA 能更好地翻译长句子。我们首先研究各个模型在不同句长上的翻译质量。根据参考译文的长度，我们将测试集分成了多个互不相交的子集，每个子集包含一个长度区间内的数据。我们用 N 表示目标端的句长，AT 表示自回归 Transformer 模型，分别计算各个模型在不同句长上的 BLEU 值，结果如表5-4所示。首先，我们可以看到基础的非自回归模型 Vanilla-NAT 的性能随着句长增加迅速下降。这是非自回归模型的一个特点，因为句长变长后，模型输出与参考译文不对齐的可能性更高，会使得损失函数更不准确。在比较 CTC 和 NMLA 时，我们也有类似的观察结果。CTC 模型在短句上的翻译质量与 NMLA 接近，但性能差距随着句长的增加而增大。这也可以用类似的方式来解释，句长越长时，单调性假设越有可能无法成立，导致损失函数的不准确性，从而损害模型的翻译质量。NMLA 方法对非单调对齐进行建模，因此弥补了这一缺陷，在长句上也有很好的翻译质量，甚至比自回归模型都高出 2BLEU 以上。

表 5-4 自回归和非自回归模型在不同句长上的性能。

Table 5-4 The performance of AT and NAT models with respect to the sequence length.

Length	Vanilla-NAT	CTC	AT	NMLA
$1 \leq N < 20$	21.27	24.83	25.68	25.55
$20 \leq N < 40$	19.49	26.81	28.05	27.82
$40 \leq N < 60$	16.21	26.54	26.71	27.74
$60 \leq N$	10.69	26.69	26.44	28.89
All	19.32	26.34	27.54	27.57

NMLA 提升了模型的自信度。受机器翻译中的多峰性影响，模型预测的概率分布可能会是多种译文的混合，导致模型在生成译文时的自信度较低。NMLA 方法直接训练模型优化 n 元组匹配的 F1 值，理论上能鼓励模型生成最优的译文，而非多种译文的混合，从而提升模型的自信度。为了验证这一猜想，我们直接观察模型在解码时预测的概率分布，对模型的自信度做定量分析。我们用信息熵来衡量模型的自信度，信息熵的计算公式为 $H(X) = -\sum_{x \in \mathcal{X}} p(x) \log p(x)$，熵越低表示模型预测的概率分布越尖锐，即自信度越高。我们在 WMT14 测试集上测量了每个时间步的熵，并对所有时间步取平均来表示模型的自信度，实验结果如表5-5所示。可以看到，基础的非自回归模型 Vanilla-NAT 的自信度较低，CTC 的自信度相比 Vanilla-NAT 高了很多，而 NMLA 方法进一步把熵降低到了 0.061，说明它能使模型在预测时有很高的自信度，有效地缓解了多峰性问题的影响。

NMLA 提升了译文的流畅性。 由于非自回归模型独立地预测各个位置的词语，译文通常较不流畅，这是非自回归模型生成译文的一个主要缺陷。NMLA 方法对 n 元组匹配进行建模，要求模型生成与参考译文一致的 n 元组，理论上应能

表 5-5 非自回归模型在预测时的平均信息熵。

Table 5-5 The average prediction entropy of NAT models.

	Vanilla-NAT	CTC	NMLA
Entropy	2.098	0.260	0.061

提升译文的流畅性。为了验证这一点，我们对各个模型在 WMT14 英德测试集上的译文的流畅性做了定量研究。我们在 WMT14 英德训练集上训练了一个 n 元组语言模型[116]，并用它来对译文进行评分，以语言模型的困惑度来评估各种模型的译文以及用 'Gold' 表示的参考译文的流畅性，实验结果如表5-6所示。可以看到，基础的模型 Vanilla-NAT 有最高的困惑度，即译文最不流畅。CTC 的困惑度较低，但仍与自回归模型有较大差距。NMLA 方法进一步提升了译文的流畅性，仅比参考译文的困惑度高 42.6。自回归模型的困惑度甚至比参考译文还低，这个发现与之前的工作一致，即自回归模型倾向于生成流畅的译文，但可能会以牺牲译文的忠实度为代价[120,121]。

表 5-6 模型在 WMT14 英德测试集上的译文的困惑度。

Table 5-6 The perplexity scores of translation results on WMT14 En-De test set.

	Vanilla-NAT	CTC	NMLA	AT	Gold
PPL	597.0	315.5	258.8	197.0	216.2

5.5　本章小结

基于隐式对齐的 CTC 模型能够考虑所有与参考译文的单调对齐来建模翻译概率，并具有生成变长译文的能力，是目前备受关注的一种非自回归模型结构。然而，在机器翻译任务中，非单调对齐是广泛存在的现象，而 CTC 模型没有建模非单调对齐的能力。本章提出为基于 CTC 的非自回归机器翻译模型建模非单调隐式对齐，并设计了基于二部图匹配和 n 元组匹配的两种非单调匹配方法。实验结果显示，我们的方法能显著改善模型的翻译质量，仅进行单轮解码就达到甚至超越了自回归模型的性能水平，并且仍相对自回归模型有可观的解码加速。

第 6 章　基于多样蒸馏的动态参考译文

6.1　引言

非自回归神经机器翻译[8]（Non-Autoregressive Neural Machine Translation，NAT）并行解码的特性使其在神经机器翻译领域受到了广泛关注。然而，非自回归模型的翻译质量与自回归模型有较大差距，其中主要的原因是多峰性问题，即同一原文可能有多种正确的译文，而参考译文只是正确译文中的一种。受多峰性问题影响，模型输出的结果可能会与参考译文不一致，此时即便是正确的模型输出也会有较高的损失值。多峰性问题导致了损失函数无法正确评估模型的输出，使得非自回归模型无法从损失函数中学习到正确的翻译方式。

如何克服多峰性问题一直以来都是非自回归机器翻译的研究重点。一种常用的解决方案是序列级知识蒸馏[13]，通过用自回归模型的译文替换训练集的目标端句子来减小数据中的多峰性。知识蒸馏能使数据集复杂度更低，确定性更强[14]，因此能更好地训练非自回归模型，目前已成为了非自回归模型的一种标准做法。然而，知识蒸馏也不能完全消除训练数据中的多峰性，并且需要非自回归模型模仿自回归模型的输出结果，固定了非自回归模型的能力上限，使我们难以进一步发掘非自回归模型的潜力。

由于同一原文可能有多种正确的译文，我们认为不应使用静态的参考译文训练模型，而应令参考译文根据模型的输出动态调整，使参考译文能够与模型输出匹配。具体地，我们提出多样蒸馏与译文选择（Diverse Distillation with Reference Selection）结合的解决方案，多样蒸馏模块为每个原文提供多个可选的参考译文，译文选择模块根据模型输出的情况选择最适合的参考译文来训练模型。如图6-1所示，多样蒸馏模块提供两个候选的参考译文 “I must leave tomorrow’’ 和 “Tomorrow I must leave’’，译文选择模块发现模型输出的概率分布更接近 “I must leave tomorrow’’，因此选择它来训练模型。通过多样蒸馏与译文选择，模型能根据合适的参考译文计算损失函数，因此训练变得更加准确。另外，这样训练的非自回归模型不会去模仿一个特定的教师模型，而是从多个参考译文中选择性地去学习，这提升了非自回归模型的能力上限，使非自回归模型有可能在翻译质量上超越自回归模型。

多样蒸馏的目标为生成多样并且高质量的参考译文，与多样机器翻译的任务目标一致。我们对此提出了一种简单有效的方法，称为 SeedDiv，直接利用模型在训练时的随机性，使用不同随机种子训练出多个自回归模型来生成多样的译文。对于译文选择，我们将模型输出的概率分布与每个参考译文进行比较，选择与模型输出最接近的参考译文来训练模型，这个过程不需要进行额外的神经网络计算，带来的额外开销很小。另外，我们也对译文选择的方法进行扩展，令模型在训练早期不加区分地从所有参考译文中学习，之后再逐渐聚焦到所选的参考译文。

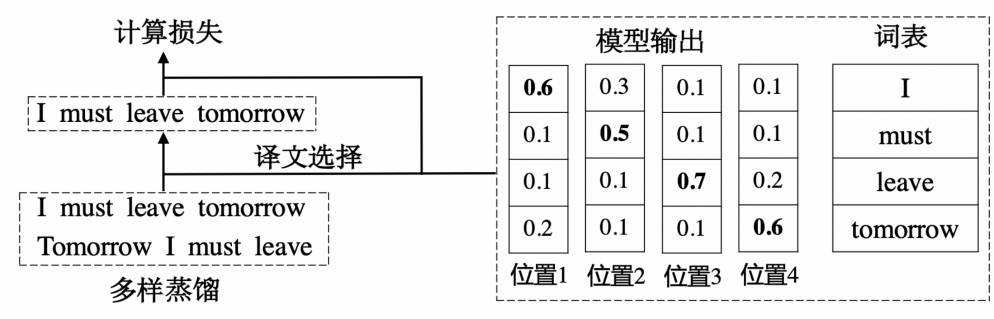


图 6-1 多样蒸馏与译文选择的示意图。

Figure 6-1 Illustration of diverse distillation and reference selection.

我们在非自回归机器翻译最常用的四个公共数据集（WMT14 En↔De，WMT16 En↔Ro）上进行了实验。结果显示，DDRS 方法能有效地提升 CTC 基线模型的性能，并能与上章的 NMLA 方法结合，只进行单轮解码就能达到在 WMT14 英德数据集上达到 28.63 BLEU。增加模型大小后，DDRS 方法甚至能以单轮解码达到 30.06 BLEU，比非自回归模型当时最先进的水平还要高 1 BLEU 值以上。

6.2 研究背景

本章提出多样蒸馏与译文选择结合的方法，使用包含多个参考译文的数据集来训练模型，其中多样蒸馏模块为每个原文提供多个可选的参考译文，任务目标与多样机器翻译相似，可以通过对序列级知识蒸馏方法进行扩展来实现。在本节中，我们对序列级知识蒸馏和多样机器翻译的研究背景做简单介绍。

6.2.1 序列级知识蒸馏

序列级知识蒸馏（Sequence-Level Knowledge Distillation，SeqKD）是神经机器翻译中广泛使用的知识蒸馏方法，一般用来训练能力较弱的学生模型去模仿教师模型的输出结果。给定学生模型预测的分布 p 和教师模型预测的分布 q，知识蒸馏的损失函数为：

$$\mathcal{L}_{SeqKD}(\theta) = -\sum_{Y} q(Y|X) \log p(Y|X,\theta) \approx -\log p(\hat{Y}|X,\theta), \tag{6-1}$$

其中，θ 为学生模型的参数，$\hat{Y}$ 为教师模型通过束搜索得到的译文。由于上式的搜索空间是指数级的，序列级知识蒸馏方法用教师模型的译文来近似教师的分布，因此直接训练学生拟合教师的译文 $\hat{Y}$。

序列级知识蒸馏的具体流程为：（1）训练教师模型，（2）令教师模型对训练集的源端做束搜索解码，（3）在源端句子与教师译文组成的句对上训练学生模型。蒸馏后的数据集一般复杂度更低，确定性更强[14]，因此能更好地训练非自回归模型，目前已成为了非自回归模型的一种标准设置。

6.2.2　多样机器翻译

多样机器翻译任务的目标令模型为生成多样并且高质量的翻译结果。假设参考译文为 Y，模型生成的一组译文为 $\{Y_1, ..., Y_k\}$，则模型的翻译质量由所有译文与参考译文之间的平均 BLEU 值来衡量，记作 rfb（reference BLEU）：

$$\text{rfb} = \frac{1}{k}\sum_{i=1}^{k}\text{BLEU}(Y, Y_i). \tag{6-2}$$

模型的翻译多样性由所有译文两两之间的平均 BLEU 值来衡量，记作 pwb（pairwise BLEU）：

$$\text{pwb} = \frac{1}{(k-1)k}\sum_{i=1}^{k}\sum_{j\neq i}\text{BLEU}(Y_i, Y_j). \tag{6-3}$$

译文与参考译文间的 BLEU 值 rfb 越高，表示翻译质量越高。译文两两之间的 BLEU 值 pwb 越低，表示翻译多样性越好。一般来说，翻译质量与多样性是无法兼得的，现有方法都要以牺牲一定翻译质量为代价来实现翻译多样性。Li 等[122], Vijayakumar 等[123] 引入正则化项来调整束搜索算法，鼓励模型在搜索过程中选择与之前译文不同的路径。He 等[124], Shen 等[125] 通过混合专家方法在翻译模型中引入离散隐变量，基于不同的隐变量生成不同的翻译结果。Sun 等[126] 通过对 Transformer 模型中注意力头的采样来实现翻译的多样性。Wu 等[127] 对模型参数的分布做一定假设，从后验分布中采样模型参数来生成不同的译文。Li 等[128] 将源端句子与训练集中的其他句子混合后作为模型输入，在不同的混合方式下使模型生成不同的翻译结果。

6.3　多样蒸馏与译文选择

在本节中，我们对多样蒸馏与译文选择方法做详细介绍，多样蒸馏模块为每个原文提供多个可选的参考译文，译文选择模块根据模型输出的情况选择最适合的参考译文来训练模型。

6.3.1　多样蒸馏

多样蒸馏的目标是为训练集中的每句原文提供多个多样且高质量的参考译文，这与多样机器翻译的任务目标一致。因此，我们可以直接利用现有的多样机器翻译方法来进行多样蒸馏[122–128]。然而，在现有的多样机器翻译方法中，翻译质量与多样性往往是无法兼得的，一般都要以牺牲一定翻译质量为代价来实现翻译多样性，而参考译文质量的下降难免会对非自回归模型的翻译质量造成影响。

为了在不影响翻译质量的前提下实现翻译多样性，我们提出了一种简单而有效的方法，称为 SeedDiv，能够直接利用随机种子控制模型训练中的随机性，从而训练出多个自回归模型来生成多样的译文。具体地，给定需要的译文数量

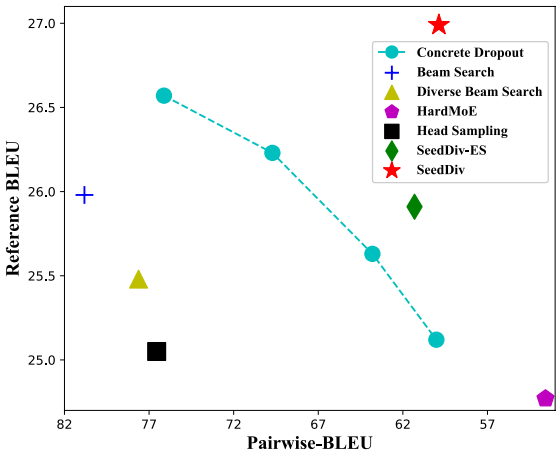


图 6-2 各种多样机器翻译方法在 WMT14 英德测试集上的翻译质量与多样性。

Figure 6-2 Translation quality and diversity of diverse translation methods on the test set of WMT14 En-De.

k，我们直接用 k 个不同的随机种子来训练得到 k 个不同的自回归 Transformer 模型。其中，随机种子控制了模型训练过程的一些随机因素，例如参数初始化、训练数据的顺序、dropout 等等。得到 k 个模型后，每个模型都用束搜索方法对训练集源端进行解码，得到 k 个不同的译文。相对于现有的方法，SeedDiv 的一个明显优势是不需要牺牲翻译质量来实现翻译多样性，因为随机种子的选取不会影响模型的期望性能。

我们在 WMT14 英德数据集上进行实验以评估 SeedDiv 方法的翻译多样性。我们使用 Transformer 模型的基本设置并将训练步数设为 15 万步，将待生成的译文数量 k 设为 3，其他详细参数设置见6.4.1节。由于 SeedDiv 方法需要训练 k 个模型，训练代价变为了 k 倍，我们也实现了一个弱化版本 SeedDiv-ES，每个模型的训练步数都降低为原来的 $\frac{1}{k}$，即 5 万步。另外，我们也实现了几种现有的方法来与 SeedDiv 方法对比，包括 Beam Search（束搜索）、Diverse Beam Search[123]、HardMoE[125]、Head Sampling[126] 和 Concrete Dropout[127]。图6-2展示了这些方法的翻译质量与多样性。翻译质量用参考译文 BLEU 值 rfb 表示，rfb 越高表示翻译质量越好。多样性用两两之间的 BLEU 值 pwb 表示，pwb 越低表示翻译多样性越好。

图中结果显示，SeedDiv 不仅在翻译质量上有很大优势，翻译多样性也优于大多数现有方法。只有 HardMoe 方法的多样性更好，但它的翻译质量有大幅的下滑。SeedDiv 方法唯一的缺陷是训练代价较高，因此我们也考虑弱化版的 SeedDiv-ES 方法，它的翻译质量因提前结束了训练过程而有所下降，但在翻译质量与多样性的平衡上仍然优于大多数现有方法。这说明基于随机种子的 SeedDiv 方法较适合用来生成多样化的翻译结果，因此我们使用 SeedDiv 方法来进行多样蒸馏，生成具有多样并且高质量参考译文的数据集。

6.3.2　译文选择

在多样蒸馏后，我们需要在训练时选择合适的参考译文来训练模型。下面，我们对译文选择做详细介绍，包括多参考译文场景下的损失函数设计、课程学习方法以及高效计算损失的方法。

6.3.2.1　损失函数设计

在多样蒸馏后，我们的训练集中每个源句 X 都有 k 个参考译文 $Y_{1:k}$。在神经机器翻译传统的数据增强方法中[129–132]，一般会对每一个参考译文都计算一次交叉熵损失，并用这些损失之和来训练模型：

$$\mathcal{L}_{sum}(\theta) = -\frac{1}{k}\sum_{i=1}^{k}\log p(Y_i|X,\theta). \tag{6-4}$$

然而，这种损失函数对非自回归模型来说并不准确。模型的输出最多与其中一个参考译文比较接近，在其他参考译文上计算的损失都会包含较多噪声。因此，这样的训练方式不会让模型学习到正确的翻译方式，反而会导致模型输出的模式为多种译文的混合。

我们认为不应按照传统的模式计算多参考译文下的损失函数，而应从多个参考译文中选择最合适的一个来训练模型。所选的参考译文应与模型输出较匹配，否则损失函数将会带有一定噪音，具体表现为对参考译文的预测概率较小，损失值偏高。因此，我们一一计算模型对每个参考译文的翻译概率，认为翻译概率最大的就是与模型最匹配的参考译文，根据这一参考译文计算损失函数：

$$\mathcal{L}_{max}(\theta) = -\log \max_{1\le i\le k} p(Y_i|X,\theta). \tag{6-5}$$

以上损失函数可以避免模型去拟合所有参考译文，而是鼓励它生成其中最接近的一个，这是一个更容易学习也更适合模型的训练目标。此外，这样训练的学生模型不会去模仿一个特定的教师模型，而是能从多个参考译文中选择性地去学习，这提升了非自回归模型的能力上限，使其有可能超越自回归教师模型的性能。

上述两种损失分别使用所有参考译文和单个参考译文来计算损失。除此之外，我们也有一种折衷选项，即最大化所有参考译文的概率之和，如下所示：

$$\mathcal{L}_{mid}(\theta) = -\log \sum_{i=1}^{k} p(Y_i|X,\theta). \tag{6-6}$$

上式等价于令每个参考译文 Y_i 的损失权重为 $\frac{p(Y_i|X,\theta)}{\sum_i p(Y_i|X,\theta)}$，概率较高的参考译文对应的权重也高，但其他参考译文也会有一定权重。

6.3.2.2　课程学习

在上一节中，我们给出了三种多参考译文场景下的损失函数，其中基于译文选择的损失 $\mathcal{L}_{max}(\theta)$ 能最准确地评估模型的输出，其他两种损失能使模型更广

泛地各个参考译文中学习。为了结合它们的优势，我们用一种两阶段的课程学习方式来训练模型。在第一阶段，我们从损失 $\mathcal{L}_{sum}(\theta)$ 线性地转换为折中的损失 $\mathcal{L}_{mid}(\theta)$。在第二阶段，我们再逐渐切换为基于译文选择的损失 $\mathcal{L}_{max}(\theta)$。我们用 t 和 T 来表示当前的训练时间步和总训练步数，用超参数 λ 表示第一阶段的长度，最终的损失函数为：

$$\mathcal{L}(\theta)=\begin{cases} T_1\mathcal{L}_{mid}(\theta)+(1-T_1)\mathcal{L}_{sum}(\theta), & t\le\lambda T \\ T_2\mathcal{L}_{max}(\theta)+(1-T_2)\mathcal{L}_{mid}(\theta), & t>\lambda T \end{cases}. \tag{6-7}$$

其中，T_1 和 T_2 定义如下：

$$T_1=\frac{t}{\lambda T},\quad T_2=\frac{t-\lambda T}{T-\lambda T}. \tag{6-8}$$

第一阶段类似于预训练，模型不加区分地从所有参考译文中学习，这可以给模型提供更全面的知识，使模型学习到更好的表示。第二阶段类似于微调，模型逐渐开始从最匹配的参考译文中学习，这能提供更准确的训练信号，使模型学习到正确的翻译方式。

6.3.2.3 高效计算

基础的非自回归模型必须要将解码器长度设置为译文 Y 的长度才能计算翻译概率 $p(Y|X,\theta)$。因此，要计算所有 k 个参考译文的翻译概率，就需要对解码器做最多 k 次的前向计算，这会使训练成本大幅上升。幸运的是，基于 CTC 的非自回归模型的解码器长度仅由源端句子决定，因此我们只需要运行一次模型，通过动态规划算法就能计算出每个参考译文的概率。在表6-1中，我们以模型所需进行的前向计算和反向计算次数为单位，用 'Forward' 表示前向计算，'Backward' 表示反向计算，给出了不同模型在计算损失 $\mathcal{L}_{max}(\theta)$ 和 $\mathcal{L}_{sum}(\theta)$ 时的计算代价。CTC 模型在计算这两个损失时都只需做一次前向和后向计算，在训练效率上有很大的优势，因此我们将 CTC 模型作为基线模型。

表 6-1 损失函数 $\mathcal{L}_{max}(\theta)$ 和 $\mathcal{L}_{sum}(\theta)$ 的计算代价。

Table 6-1 The calculation cost of $\mathcal{L}_{max}(\theta)$ and $\mathcal{L}_{sum}(\theta)$ for different models.

Models	$\mathcal{L}_{max}(\theta)$		$\mathcal{L}_{sum}(\theta)$	
	Forward	Backward	Forward	Backward
AT	$k\times$	$1\times$	$k\times$	$k\times$
Vanilla-NAT	$k\times$	$1\times$	$k\times$	$k\times$
CTC	$1\times$	$1\times$	$1\times$	$1\times$

6.4 实验结果与分析

在本节中，我们将通过实验来验证多样蒸馏与译文选择方法的实际效果，首先介绍相关的实验设置，再给出主要的实验结果，最后对所得结果进行分析。

6.4.1　实验设置

数据集　我们在非自回归机器翻译最常用的四个公共数据集上验证了所提的方法：WMT14 英语 ↔ 德语（En↔De，约 450 万个句对）、WMT16 英语 ↔ 罗马尼亚语（En↔Ro，约 61 万个句对）。对于 WMT14 英德数据集，我们将 *newstest2013* 和 *newstest2014* 分别作为验证集和测试集。对于 WMT16 英罗数据集，我们将 *newsdev-2016* 和 *newstest-2016* 分别作为验证集和测试集。我们用 32K 次操作的联合 BPE 模型[100] 来切分源端和目标端词汇，并在源端和目标端共享词表。

知识蒸馏　序列级知识蒸馏[12,13] 是非自回归模型中的一项关键技术。在基线的非自回归模型中，我们应用序列级知识蒸馏技术来简化目标端的数据分布。在我们的 DDRS 模型中，我们将多样蒸馏的参考译文数 k 设为 3，等价于使用 3 个教师模型进行序列级知识蒸馏。在训练第 i 个教师模型时，我们将随机种子的值也设为 i。

参数设置　我们将课程学习中的 λ 值设置为 $2/3$，即第一阶段的训练占所有训练步数的 $2/3$。我们将基础设置的 Transformer 模型[1] 作为自回归的基线模型，CTC 模型也采用 Transformer 的模型结构，并用平均拷贝的方法[8] 来构建解码器的输入。我们将 CTC 中的过采样比例设置为 3，即解码器的长度是编码器的三倍。我们用 Adam 优化器[101] 来优化模型，其参数为 $\beta = (0.9, 0.98)$、$\epsilon = 10^{-8}$。我们将训练时的 batch 大小设为 3.2 万词。对于 WMT14 英德数据集，我们将训练步数设为 30 万步，dropout 比例设为 0.2。对于 WMT16 英罗数据集，我们将训练步数设为 15 万步，dropout 比例设为 0.3。学习率在 1 万步内线性提升至 $5 \cdot 10^{-4}$，之后按倒数平方根规则衰减。我们用 8 张 GeForce RTX 3090 GPU 卡来训练模型，用单张卡在 batch 大小为 1 的设置下测量解码速度。我们在开源代码框架 `fairseq`[117] 上实现了我们的模型。

解码　对于自回归模型，我们使用束搜索进行解码，束大小设置为 5。对于 CTC 模型，我们可以使用 argmax 解码方法，预测概率最大的对齐，再生成对应译文。另外，我们也采用束搜索的方法，束大小设置为 20，直接搜索概率较高的译文 Y，并引入自回归的统计语言模型对译文打分[111,116]。束搜索过程中不需要做任何神经网络的计算，已有基于 C++ 的高效实现[1]，因此解码速度相较自回归模型仍有很大优势。

6.4.2　主要实验结果

在表6-2中，我们比较了本章方法 DDRS 与基线模型和现有方法的性能。其中，T 表示译文的长度，$k = 3$ 表示从三个自回归模型的集成中进行知识蒸馏[133]，'Iter' 表示迭代式模型的迭代轮数。在强基线模型 CTC 上，DDRS 方法的平均提升在 1 BLEU 值以上，说明了 DDRS 方法的有效性。注意到，将三个教师模型集成来进行知识蒸馏[133] 的方法对 CTC 模型的提升非常微弱，这说明我们无法

[1]https://github.com/parlance/ctcdecode

表 6-2 我们的方法与现有方法的性能对比。

Table 6-2 Performance comparison between our models and existing methods.

Models		Iter	Speedup	WMT14		WMT16	
				EN-DE	DE-EN	EN-RO	RO-EN
AT	Transformer[1]	T	1.0×	27.51	31.52	34.39	33.76
	+ distillation (k=3)	T	1.0×	**28.04**	**32.17**	**35.10**	**34.83**
One-pass NAT	NAT-FT[8]	1	15.6×	17.69	21.47	27.29	29.06
	CTC[24]	1	–	16.56	18.64	19.54	24.67
	NAT-REG[65]	1	27.6×	20.65	24.77	–	–
	AXE[69]	1	–	23.53	27.90	30.75	31.54
	SNAT[49]	1	22.6×	24.64	28.42	32.87	32.21
	GLAT[50]	1	15.3×	25.21	29.84	31.19	32.04
	CNAT[44]	1	10.3×	25.56	29.36	–	–
	Imputer[25]	1	–	25.80	28.40	32.30	31.70
	OAXE[70]	1	–	26.10	30.20	32.40	33.30
	AligNART[47]	1	13.4×	26.40	30.40	32.50	33.10
	REDER[27]	1	15.5×	26.70	30.68	33.10	33.23
	CTC w/ DSLP&MT[28]	1	14.8×	27.02	31.61	34.17	34.60
	Fully-NAT[26]	1	16.8×	27.20	**31.39**	**33.71**	**34.16**
	REDER + beam&rerank	1	5.5×	**27.36**	31.10	33.60	34.03
Iterative NAT	iNAT[58]	10	2.0×	21.61	25.48	29.32	30.19
	CMLM[35]	10	–	27.03	30.53	33.08	33.31
	RecoverSAT[52]	N/2	2.1×	27.11	31.67	32.92	33.19
	LevT[62]	2.05	4.0×	27.27	–	–	33.26
	DisCO[60]	4.82	–	27.34	31.31	33.22	33.25
	JM-NAT[59]	10	–	27.69	**32.24**	33.52	33.72
	RewriteNAT[63]	2.70	–	27.83	31.52	33.63	34.09
	Imputer[25]	8	–	**28.20**	31.80	**34.40**	**34.10**
Our work	CTC	1	14.7×	26.09	29.50	33.55	32.98
	CTC + distillation (k=3)	1	14.7×	26.35	29.73	33.51	32.82
	DDRS w/o finetune	1	14.7×	27.18	30.91	34.42	34.31
	DDRS w/ NMLA	1	14.7×	28.02	31.80	34.73	34.76
	+ beam&lm	1	5.0×	**28.63**	**32.65**	**35.51**	**35.85**

简单地通过增加教师模型的数量来改善非自回归模型的翻译质量，而应使用多样蒸馏与译文选择结合的方式来训练模型。与现有方法相比，DDRS 的翻译性能与最先进的单轮模型相当，仅略微弱于自回归 Transformer 模型。另外，我们也发现 DDRS 方法与上章的 NMLA 训练目标有很好的互补性。用 NMLA 方法对 DDRS 模型进行微调后，模型的性能得到了进一步的大幅提升，在 BLEU 值上全面超越了自回归的 Transformer 模型，与当时最先进的多轮模型水平相似。最后，在做束搜索并集成统计语言模型后，所得模型全面超越了现有的方法，并仍然具有相对自回归模型 5 倍的翻译加速比。

我们进一步在更大的模型尺寸和更丰富的教师模型下进行实验，以探索 DDRS 方法能达到的性能上限。我们使用 Transformer-big 设置[1] 作为教师模型和学生模型的结构，并进一步增加 3 个从右到左翻译的（Right-to-Left, R2L）教师来使参考译文更丰富，实验结果如表6-3所示。DDRS-big 的 BLEU 值为 28.24，与自回归模型的提升幅度相似。在结合 NMLA 后，模型性能达到了 29.11BLEU，在翻译质量和翻译速度上都大幅超越了同设置的自回归模型。最后，模型在束搜索下达到了 30.06 BLEU，将非自回归模型在这一数据集上的最高性能提升了近 2BLEU，已经接近自回归模型在这一数据集上的最先进水平。

表 6-3 在更大的模型尺寸和更丰富的教师模型下，DDRS 方法在 WMT14 英德数据集上的性能。

Table 6-3 The performance of DDRS under larger model size and more teacher models on the test set of WMT14 En-De.

Models	BLEU	Speedup
Transformer-big (teacher)	**28.64**	0.9×
R2L-Transformer-big (teacher)	27.96	0.9×
DDRS-big w/o finetune	28.24	14.1×
DDRS-big + NMLA	29.11	14.1×
DDRS-big + NMLA + beam&lm	**30.06**	4.8×

6.4.3　消融实验

我们通过消融实验来验证 DDRS 方法中各项损失函数的作用，实验结果如表6-4所示。其中，λ 是课程学习的参数，控制第一阶段训练的时长，BLEU_1 表示验证集上的 BLEU 值，BLEU_2 表示测试集上的 BLEU 值，CTC 基线模型在验证集的 BLEU 值为 24.57。首先，我们单独用损失 $\mathcal{L}_{sum}(\theta)$、$\mathcal{L}_{max}(\theta)$、$\mathcal{L}_{mid}(\theta)$ 来训练模型，表中前三行为相应的结果。损失 $\mathcal{L}_{sum}(\theta)$ 的效果与基线模型相差不大，表明直接用多个参考译文训练非自回归模型并没有明显增益。损失 $\mathcal{L}_{mid}(\theta)$ 和 $\mathcal{L}_{max}(\theta)$ 都相对于 CTC 基线模型有较明显的提升，说明多样蒸馏要与译文选择结合后才能起到效果。

随后，我们用两阶段的课程学习方式来结合这三种损失，表6-4中的后四行给出了参数 λ 取不同值时的结果。其中，λ 为 0 表示仅包含第二阶段，损失从 $\mathcal{L}_{mid}(\theta)$ 过渡到 $\mathcal{L}_{max}(\theta)$，$\lambda$ 为 1 表示仅包含第一阶段，损失从 $\mathcal{L}_{sum}(\theta)$ 过渡到 $\mathcal{L}_{mid}(\theta)$。可以看到，课程学习方法确实能有效地结合这三种损失函数的优势，模型性能比单独使用任意一种损失时都有明显提升。模型在 λ 取 $2/3$ 时表现最佳，相比于直接做译文选择的 $\mathcal{L}_{max}(\theta)$ 损失提升约 0.3 BLEU。

表 6-4 在 WMT14 英德数据集上的消融实验。

Table 6-4 Ablation study on WMT14 En-De.

$\mathcal{L}_{sum}$	$\mathcal{L}_{mid}$	$\mathcal{L}_{max}$	λ	BLEU_1	BLEU_2
✓				24.61	25.97
	✓			25.23	**26.90**
		✓		**25.31**	26.88
	✓	✓	0	25.41	26.99
✓	✓	✓	$1/3$	25.48	27.13
✓	✓	✓	$2/3$	**25.59**	**27.18**
✓	✓		1	25.45	27.09

6.4.4 自回归模型上的效果

DDRS 方法是本章针对非自回归机器翻译中的多峰性问题提出的一种解决方案，但方法中使用的几种损失函数在自回归模型中也能使用。因此，我们在本节中对 DDRS 方法在自回归模型上的效果进行实验验证。在表6-5中，我们给出了自回归模型和非自回归模型分别使用本章提出的一系列损失函数训练后的结果，其中 $\mathcal{L}_{CE}$ 表示仅使用单句参考译文的交叉熵损失，$\mathcal{L}_{sum}$、$\mathcal{L}_{mid}$、$\mathcal{L}_{max}$ 为本章提出的使用多句参考译文的损失函数。可以看到，在非自回归模型上效果最差的求和损失 $\mathcal{L}_{sum}$ 反而在自回归模型上效果最好，而基于译文选择的损失 $\mathcal{L}_{mid}$、$\mathcal{L}_{max}$ 甚至起到了反作用，性能弱于使用静态参考译文的交叉熵损失。

表 6-5 使用不同损失函数时，自回归与非自回归模型在 WMT14 英德测试集上的性能。

Table 6-5 The performance of AT and NAT models on the test set of WMT14 En-De when using different loss functions.

Models	$\mathcal{L}_{CE}$	$\mathcal{L}_{sum}$	$\mathcal{L}_{mid}$	$\mathcal{L}_{max}$
AT	27.70	28.08	27.37	27.21
NAT	26.09	25.97	26.90	26.88

上述实验结果其实是符合我们预期的。非自回归模型直接生成整句译文，因此输出结果很可能与参考译文不一致，所以多样蒸馏与译文选择方法能为模型

提供更匹配的参考译文，帮助非自回归模型更好地进行训练。相比之下，自回归模型一般采用 teacher forcing 算法[88] 进行训练，基于参考译文的前 t 个词来生成第 $t+1$ 个词。因此，自回归模型的输出结果本就与参考译文一致，基本不受多峰性问题的影响，也就不需要做译文选择。

6.4.5　训练时间

本章所用多样蒸馏方法的一个缺陷是它需要训练 k 个教师模型，并且每个教师模型都要解码一次训练集，因此多样蒸馏的计算代价是序列级知识蒸馏的 k 倍。然而，我们认为这一蒸馏代价的增加是可以接受的。非自回归模型的训练代价包括数据蒸馏和模型训练两部分，其中数据蒸馏占比一般都较小。因此，我们可以通过减小模型训练的代价来弥补蒸馏代价的提升。我们基于 8 张 GeForce RTX 3090 GPU 的条件来测量训练时间，用 'Distill' 表示序列级知识蒸馏或多样蒸馏的耗时，在 WMT14 英德数据集上进行实验，在表6-6中给出了 CTC 和 DDRS 方法在不同训练 batch 大小下的训练耗时和测试集 BLEU 值。可见，尽管 DDRS 方法的蒸馏代价较高，但通过将 batch 大小降为 32K，它的总训练耗时与 64K batch 大小的 CTC 模型相近，并远小于使用 128K batch 大小的 CTC 模型。另外，DDRS 方法在较小 batch 大小下的性能仍有明显优势，BLEU 值比使用 128K batch 大小的 CTC 模型还要高 0.59，整体上看性价比更高。

表 6-6 CTC 和 DDRS 方法在不同 batch 大小下的训练耗时和性能对比。
Table 6-6 The performance and training cost comparison between CTC and DDRS under different batch sizes.

Models	Distill	Train	Total	BLEU
CTC (64K)	5.5h	26.4h	31.9h	26.34
CTC (128K)	5.5h	52.5h	58.0h	26.59
DDRS (32K)	16.5h	15.7h	32.2h	**27.18**

6.5　本章小结

受多峰性问题影响，非自回归模型的输出结果可能会与参考译文不一致，导致交叉熵损失无法正确评估模型的输出。我们认为不应使用静态的参考译文训练模型，并提出多样蒸馏与译文选择结合的解决方案，多样蒸馏模块为每个原文提供多个可选的参考译文，译文选择模块根据模型输出的情况选择最适合的参考译文来训练模型。实验结果显示，我们的方法显著地改善了 CTC 基线模型的翻译质量，并能与上一章的非单调对齐方面的改进结合来进一步提升模型性能，使非自回归模型在所用数据集上均达到甚至超越了自回归 Transformer 模型的翻译质量。

第 7 章　基于改写器的动态参考译文

7.1　引言

非自回归神经机器翻译[8]（Non-Autoregressive Neural Machine Translation，NAT）模型的翻译速度有显著优势，但在翻译质量上与自回归模型有较大差距。非自回归模型翻译质量较差的主要原因是多峰性问题，即同一原文可能有多种正确的译文，而参考译文只是正确译文中的一种。当模型输出的结果与其他译文更接近时，基于参考译文计算的交叉熵损失就会给出偏高的损失值，无法正确评估模型的输出。在这种训练方式下，非自回归模型倾向于生成多种译文混合的结果，通常会包含一些重复词并遗漏一些信息，因此翻译质量较差。

如何克服多峰性问题一直以来都是非自回归机器翻译的研究重点。一种常用的解决方案是序列级知识蒸馏[13]，通过用自回归模型的译文替换训练集的目标端句子来减小数据中的多峰性。此外，一种研究思路是使用隐变量或词对齐信息来减少翻译过程中的不确定性[39,40,47,134]，另一种研究思路是用表达能力更强的结构来建模目标端概率分布[19,21,24,25]，也有研究者对损失函数做改进来减轻多峰性的影响[69,70,77,135]。

与之前的研究工作不同，我们从参考译文的角度出发，认为不应使用静态的参考译文训练模型，而应根据模型的输出动态调整参考译文，使其能够与模型输出匹配。在上一章中，我们给出了基于多样蒸馏与译文选择的解决方案，从多个参考译文中选取最合适的一个来训练模型。然而，多样蒸馏的代价通常较高，并且生成的参考译文数目也比较有限，不一定能覆盖所有情况。在本章中，我们提出了另一种生成动态参考译文的解决方案，引入改写器的结构来根据模型的输出动态改写参考译文。如图7-1所示，当参考译文为 “I ate pizza this morning’’ 但模型的输出结果为 “this morning I ate apple’’ 时，尽管模型唯一的预测错误是将披萨翻译为了 ‘apple’，但基于参考译文计算的交叉熵损失会对所有输出位置都给出较高的损失。如果我们能将参考译文改写为与模型输出匹配的 “this morning I ate pizza’’，交叉熵损失就能准确地评估模型的输出结果。

动态参考译文的质量取决于改写器的好坏，因此如何训练得到理想的改写器就是这一方法的核心问题。由于没有对改写器的直接监督信号，我们将对改写器的要求量化为奖赏函数，并通过强化学习方法来优化改写器。首先，改写器的改写结果应与非自回归模型的输出较匹配，以避免交叉熵损失偏高导致的训练不准确问题。其次，改写器仅对参考译文做形式上的改写，不应改变参考译文的语义。因此，奖赏函数也应包含两部分，第一部分与非自回归模型训练时的损失值有关，第二部分与参考译文和改写结果的语义相似度有关。在基于强化学习方法的训练下，改写器应能根据模型的输出动态改写参考译文，使改写结果在保持参考译文原有语义的同时与模型输出匹配，这时我们就能直接用改写结果训练非自回归模型。

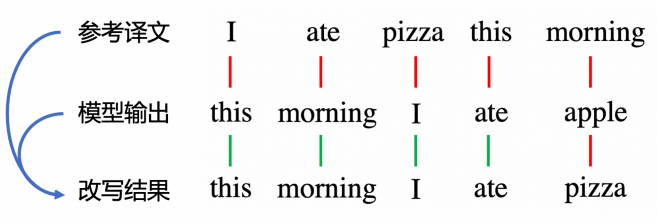


图 7-1 当参考译文不适合训练模型时，理想的改写器输出结果的案例。
Figure 7-1 An example of the desired rephraser output when the reference is inappropriate for the training.

我们在非自回归机器翻译最常用的四个公共数据集（WMT14 En↔De，WMT16 En↔Ro）上进行了实验。结果显示，改写器方法在多种非自回归基线模型上都有显著的提升，效果最好的模型能在将解码加速 14.7 倍的同时达到与自回归模型相似的翻译质量。

7.2 研究背景

在本节中，我们对非自回归神经机器翻译的几种主流模型做简要介绍。我们用 X 表示源端句子，用 $Y = \{y_1, ..., y_T\}$ 表示目标端的参考译文。

Vanilla NAT 我们一般用 Vanilla NAT 表示基础的非自回归模型，它对译文中所有词的生成做独立的概率建模，并基于 Transformer 结构实现了译文的并行解码。在训练时，Vanilla NAT 的解码器长度设为参考译文长度 T，损失函数通常为交叉熵损失：

$$\mathcal{L}_{NAT}(\theta) = -\sum_{t=1}^{T} \log(p_t(y_t|X,\theta)). \tag{7-1}$$

基础的非自回归模型还包含一个长度预测模块，以模型对源端句子的编码结果作为输入来预测译文的长度。在解码时，模型先预测出译文的长度，再并行生成指定长度的译文。

CMLM CMLM 是一种条件掩码式语言模型（Conditional Masked Language Model, CMLM），训练和解码均采用掩码-预测的方式。在训练时，CMLM 模型将目标端译文加上随机的掩码来作为解码器的输入，并训练模型预测输入中的掩码：

$$\mathcal{L}_{CMLM}(\theta) = -\log(p(Y_{mask}|X, Y_{obs}, \theta)), \tag{7-2}$$

其中，Y_{mask} 和 Y_{obs} 分别表示目标端译文中的掩盖部分和可见的部分。在解码时，CMLM 模型通过掩码-预测的方式迭代地优化译文。在第一轮解码时，解码器的输入均为掩码。在之后的每一轮解码中，前一轮预测结果中自信度较低的部分会被加上掩码，模型重新预测这些掩码来优化译文。

CTC 基于 CTC 的非自回归模型[20,24,25] 具有生成变长译文的能力，在翻译质量上有明显优势，是目前备受关注的一种非自回归模型结构。CTC 模型不需要长

度预测模块，而是会假设模型输出的序列比目标序列要长，并允许输出序列中包含空白词和重复词。在解码时，映射 β^{-1} 会移除这些多余的词来得到译文 Y。在训练时，CTC 考虑所有对应参考译文 Y 的输出序列 A，并通过动态规划算法求得译文 Y 的对数似然损失：

$$\mathcal{L}_{CTC}(\theta) = -\log \sum_{A \in \beta(Y)} p(A|X,\theta), \tag{7-3}$$

其中，输出序列 A 的概率 $p(A|X,\theta)$ 由非自回归的 Transformer 模型来建模：

$$p(A|X,\theta) = \prod_{t=1}^{T_A} p_t(a_t|X,\theta). \tag{7-4}$$

7.3　集成改写器的非自回归模型

在本节中，我们对基于改写器的动态参考译文方法做详细介绍。首先，我们展示了集成改写器的非自回归模型的基本结构。随后，我们对模型的训练方法做了详细介绍，包括如何用强化学习方法训练改写器，以及如何基于改写器训练非自回归模型。最后，我们介绍了将改写器结构扩展到其他非自回归模型上的方法。

7.3.1　模型概览

在图7-2中，我们展示了集成改写器的非自回归模型的基本结构。除传统的编码器-解码器结构外，我们引入了改写器模块来根据模型输出改写参考译文，目标是给非自回归模型提供更合适的训练目标。改写器仅用于训练阶段，不影响非自回归模型的解码，因此模型的解码速度不变。

如图7-1所示，我们希望改写器能将参考译文改写为与模型输出形式匹配、适合模型训练的形式。因此，改写器的输入应包含参考译文和非自回归模型的输出两部分。我们令改写器也采用 Transformer 的模型结构，底层输入为参考译文的词嵌入，并通过注意力机制来关注到翻译模型解码器的输出。另外，改写器要在训练时改写每句参考译文，必须要有较快的改写速度，否则训练效率会大幅下降。因此，我们将改写器的层数减小到 $N_r = 2$，并令其按非自回归的方式并行生成改写结果，概率建模方式如下所示：

$$p_r(Y_r|X,Y,\theta) = \prod_{t=1}^{T} p_t^r(y_t^r|X,Y,\theta), \tag{7-5}$$

其中，$Y_r = \{y_1^r, ..., y_T^r\}$ 表示改写器的改写结果，p_r 表示改写结果的概率，p_t^r 为改写器在位置 t 的概率分布。生成改写器的概率分布后，我们直接用 argmax 解码方法得到改写结果 Y_r，并将改写结果 Y_r 作为翻译模型的训练目标，损失函数计算如下：

$$\mathcal{L}_r(\theta) = -\sum_{t=1}^{T} \log(p(y_t^r|X,\theta)). \tag{7-6}$$

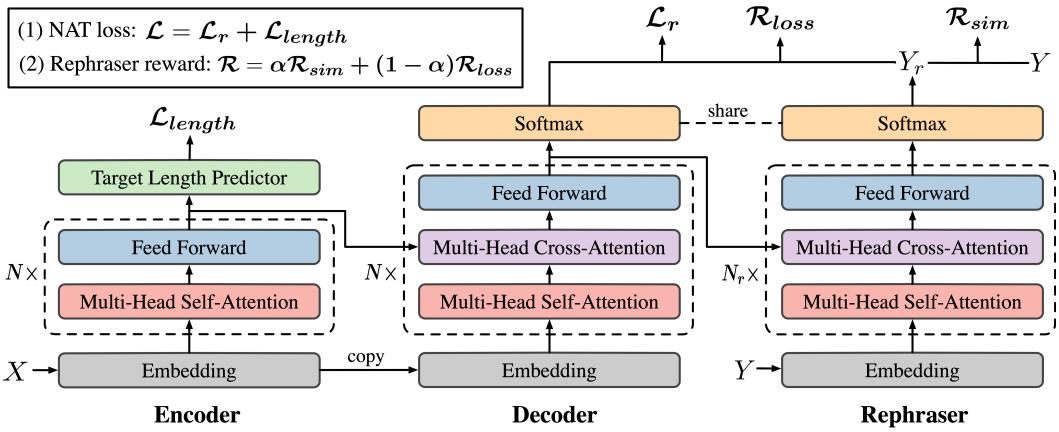


图 7-2 集成改写器的基础非自回归模型的模型结构。改写器位置在模型解码器之后，用于给模型提供更合适的训练目标，非自回归模型基于改写器的输出来计算交叉熵损失。改写器的训练目标为优化两个奖赏函数，分别对应非自回归模型的损失值和改写结果与参考译文的语义相似度。

Figure 7-2 The architecture of the vanilla NAT with rephraser. The rephraser is stacked behind the decoder to provide a better training target for NAT. The cross-entropy loss is calculated based on the rephraser output. The rephraser is trained to optimize two rewards, corresponding to the loss of NAT model and the semantic similarity between the rephraser output and reference sentence respectively.

7.3.2 训练方法

在本节中，我们对模型的训练方法做详细介绍。由于没有对改写器的直接监督信号，我们将对改写器的要求量化为奖赏函数，并使用强化学习方法来优化改写器。在用强化学习方法训练前，我们先对改写器做预训练以建立较好的初始状态，以避免改写器的学习陷入局部最优。我们对改写器的要求包含两部分，通过一个超参数插值来得到最终对改写器的奖赏函数，这一超参的取值对改写器有关键的影响，因此我们设计了一种退火策略来寻找最优的超参。

7.3.2.1 强化学习

由于没有对改写器的直接监督信号，我们使用强化学习方法来优化改写器。改写器的目标是为非自回归模型提供合适的参考译文，这一目标具体分为两部分，每部分都可以被量化为一个奖赏函数。首先，改写器的改写结果应与非自回归模型的输出较匹配，以避免交叉熵损失偏高导致的训练不准确问题。因此，我们使用奖赏 $\mathcal{R}_{loss}$ 来鼓励改写器降低模型的训练损失。其次，改写器仅对参考译文做形式上的改写，不应改变参考译文的语义。因此，我们使用奖赏 $\mathcal{R}_{sim}$ 来衡量改写结果与参考译文之间的语义相似度。

给定改写结果 $Y_r = \{y_1^r, ..., y_T^r\}$，我们将奖赏 $\mathcal{R}_{loss}$ 定义为负的模型损失：

$$\mathcal{R}_{loss}(Y_r) = \frac{\sum_{t=1}^{T} \log(p(y_t^r | X, \theta))}{T}. \tag{7-7}$$

上式中对长度 T 的归一化能保持奖赏值尺度的稳定。奖赏 $\mathcal{R}_{loss}$ 较高表示非自回归模型对改写结果有较高的预测概率，即改写结果与非自回归模型的输出较匹配。

另一方面，我们希望改写结果与参考译文之间的语义相似度较高。我们用 $\mathcal{S}$ 表示语义相似度的度量函数，$\mathcal{S}(Y_1, Y_2)$ 表示译文 Y_1 与 Y_2 之间的语义相似度。给定参考译文 Y 与改写结果 Y_r，奖赏 $\mathcal{R}_{sim}$ 定义如下：

$$\mathcal{R}_{sim}(Y_r) = \mathcal{S}(Y, Y_r). \tag{7-8}$$

度量函数 $\mathcal{S}$ 可以从机器翻译的评价指标中选取，例如 BLEU[79]、METEOR[136]、BERTScore[137] 等，我们最终选择 BLEU 为语义相似度的度量函数。我们用一个超参数 $\alpha \in [0,1]$ 对上述两个奖赏插值来得到最终对改写器的奖赏函数：

$$\mathcal{R}(Y_r) = \alpha \mathcal{R}_{sim}(Y_r) + (1-\alpha)\mathcal{R}_{loss}(Y_r). \tag{7-9}$$

我们以最大化期望奖赏为训练目标，采用 REINFORCE 算法[85] 来优化改写器：

$$\begin{aligned} \nabla_\theta \mathcal{J}(\theta) &= \nabla_\theta \sum_{Y_r} p_r(Y_r|X,Y,\theta) \cdot \mathcal{R}(Y_r) \\ &= \mathop{\mathbb{E}}_{Y_r \sim p_r} [\nabla_\theta \log(p_r(Y_r|X,Y,\theta)) \cdot \mathcal{R}(Y_r)]. \end{aligned} \tag{7-10}$$

基于上式，我们用蒙特卡洛采样方法，从改写器预测的概率分布 $p_r(Y_r|X,Y,\theta)$ 中采样出一个改写结果 Y_r，计算它的奖赏函数 $\mathcal{R}(Y_r)$，并得到对梯度的无偏估计 $\nabla_\theta \log(p_r(Y_r|X,Y,\theta)) \cdot \mathcal{R}(Y_r)$，用估计得到的梯度来训练改写器。为了减小梯度估计的方差，我们额外进行 $K=2$ 次采样并计算采样结果的奖赏均值，从 $\mathcal{R}(Y_r)$ 中减去奖赏均值来作为最终奖赏。

7.3.2.2　预训练

直接用强化学习方法训练改写器时，改写器容易陷入局部最优的状态，只输出一些由常见词组成的无意义句子。这是强化学习中的一个常见问题，因为它是基于探索的方法，对初始状态比较敏感。对于随机初始化的模型，它的采样结果均为无意义的句子，所得的奖赏也基本相同，模型难以从中学习到有效的知识。因此，我们先对改写器做预训练以建立较好的初始状态，随后再用强化学习方法训练改写器。正常来说，改写结果应该是参考译文和模型输出结果的一种组合，在形式上与模型输出匹配，但在语义上与参考译文对齐。因此，在预训练阶段，我们令改写器从参考译文和模型输出中学习。具体地，我们使用两个损失函数的平均值来训练改写器，第一个为根据参考译文计算的交叉熵损失，第二个为根据模型输出的 argmax 解码结果计算的交叉熵损失。另外，在预训练阶段，我们也基于参考译文计算交叉熵损失来训练翻译模型。

7.3.2.3 退火策略

在使用强化学习方法训练改写器时，我们使用一个超参数 α 插值来得到最终对改写器的奖赏函数，这一超参的取值对改写器有关键的影响。如果我们通过网格搜索来寻找最优的 α 值，训练成本就会成倍地增加。针对这个问题，我们设计了一种对超参数 α 的退火策略来代替网格搜索方法，可以让我们只进行一次训练就得到最终模型。

改写器的改写结果一般是参考译文和模型输出结果的一种组合，在 α 取值较高时，改写结果更接近参考译文，在 α 较小时，改写结果更接近翻译模型的输出结果。相比来说，α 偏低对模型的危害更大，会使翻译模型变得过于自信，甚至可能会导致模型的崩溃。而偏高的 α 仅会使改写结果更接近参考译文，相当于继续对模型做预训练，没有太大的负面影响。因此，我们可以首先使用较高的 α 来训练改写器，随后逐渐降低 α 的值，并观察模型在验证集上的性能变化。在这一过程中，改写结果逐渐从参考译文过渡为模型的输出，模型的性能也将会逐步提升，直到达到最优 α 值后开始下降，我们通过验证集就能找出这一模型性能最佳的检查点。

具体地，我们用两个超参数 α_{max}、α_{min} 来代替超参 α，使用一种线性退火策略来将 α 的值从 α_{max} 逐渐降低到 α_{min}。假设这一阶段的总训练步数为 T，则 α 在时间步 t 的取值为：

$$\alpha = \frac{t}{T}\alpha_{min} + (1 - \frac{t}{T})\alpha_{max}. \tag{7-11}$$

使用这一退火策略后，模型性能对超参 α_{max}、α_{min} 均不太敏感，因此我们不需要仔细调整这两个参数，在不同的翻译数据集上都可以直接使用同一组超参值。

7.3.3 结构扩展

目前，我们已经讨论了在基础的非自回归模型中集成改写器结构的方法。改写器方法并没有对非自回归模型的结构做具体限制，也可以扩展到其他的模型中。在本节中，我们将改写器方法扩展到了 CMLM 和 CTC 中，它们是7.2节中介绍的非自回归模型目前的另外两种主流结构。

CMLM 条件掩码式语言模型 CMLM 与基础的非自回归模型之间的主要区别在于训练方法。CMLM 模型在已观察到部分目标译文的情况下预测输入中的掩码，因此我们在改写参考译文时也应保持观察到的目标译文不变，仅对掩码部分的参考译文进行改写。

CTC CTC 模型的一个特点是解码器长度与参考译文长度无关，生成译文的长度不固定。因此，我们在改写参考译文时也不用局限于原来的句长，可以采用更灵活的改写器结构。我们令改写器也为基于 CTC 的解码模型，将输出序列去除重复词和空白词后作为改写结果。基于 CTC 的改写器长度与参考译文不一致，因此我们将参考译文与模型输出调换位置来适应改写器的长度，将翻译模型解码

器的输出作为改写器底部的输入，并通过注意力机制来关注到参考译文的词嵌入。

7.4　实验结果与分析

在本节中，我们将通过实验来验证改写器方法的实际效果，首先介绍相关的实验设置，再给出主要的实验结果，最后对所得结果进行分析。

7.4.1　实验设置

数据集　我们在非自回归机器翻译最常用的四个公共数据集上验证了所提的方法：WMT14 英语 ↔ 德语（En↔De，约 450 万个句对）、WMT16 英语 ↔ 罗马尼亚语（En↔Ro，约 61 万个句对）。对于 WMT14 英德数据集，我们分别将 *newstest2013* 和 *newstest2014* 作为验证集和测试集。对于 WMT16 英罗数据集，我们分别将 *newsdev-2016* 和 *newstest-2016* 作为验证集和测试集。我们用 32K 次操作的联合 BPE 模型[100] 来切分源端和目标端词汇，并在源端和目标端共享词表。我们使用 BLEU[79] 来评估模型的翻译质量。

知识蒸馏　序列级知识蒸馏[12,13] 是非自回归模型中的一项关键技术。在所有翻译任务中，我们都应用序列级知识蒸馏技术，首先训练一个自回归模型作为教师，再令它翻译一遍训练数据集的源端，将原文与译文组成句对来构建蒸馏数据集。最后，我们使用蒸馏数据集来训练非自回归模型。

基线模型　我们在三种主流的非自回归模型结构上进行实验来验证改写器方法的效果，包括基础的非自回归模型 Vanilla NAT[8]、掩码模型 CMLM[35] 和 CTC 模型[24]。对于 Vanilla NAT 模型和 CTC 模型，我们用平均拷贝的方法[8] 来构建解码器的输入。在 CTC 模型上，我们也尝试将改写器方法与第四章的非单调对齐方法结合起来共同训练模型。

实现细节　在所有数据集上，我们均将 α_{max} 设为 0.75，α_{min} 设为 0.5，采样次数 K 设为 2，改写器层数 N_r 设为 2。我们将 CTC 中的过采样比例 λ 设置为 3，即解码器的长度是编码器的三倍。我们用 Adam 优化器[101] 来优化模型，其参数为 $\beta = (0.9, 0.98)$、$\epsilon = 10^{-8}$。对于 Vanilla NAT 模型和 CTC 模型，我们将训练时的 batch 大小设为 6.4 万词。对于 CMLM 模型，为了与之前的工作保持一致[35,69,70]，我们将 batch 大小设为 12.8 万词，在解码时同时生成 5 种不同长度的候选译文。在预训练阶段，WMT14 英德数据集的训练步数为 30 万步，WMT16 英罗数据集的训练步数为 15 万步。在微调阶段，WMT14 英德数据集的训练步数为 3 万步，WMT16 英罗数据集的训练步数为 1.5 万步。在微调时，我们每隔 500 步存储一个检查点，并将 5 个最优检查点的参数取平均来得到最终模型。我们用 8 张 GeForce RTX 3090 GPU 卡来训练模型，用单张卡来测试解码速度。我们在开源代码框架 `fairseq`[117] 上实现我们的模型。

7.4.2 主要实验结果

表 7-1 我们的方法与现有方法的性能对比。

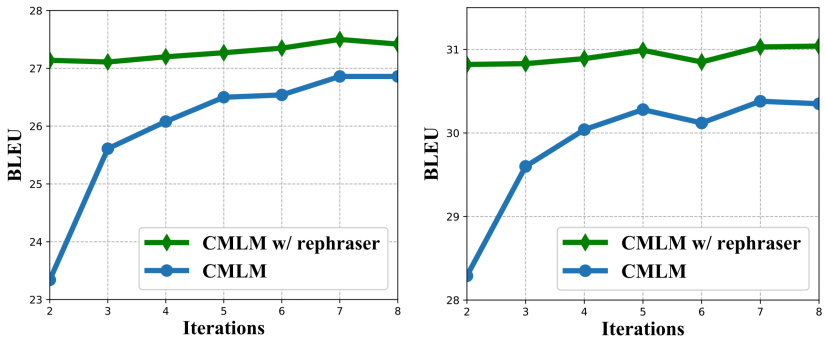
Table 7-1 Performance comparison between our models and existing methods.

Models		Iter	Speed	WMT14		WMT16	
				EN-DE	DE-EN	EN-RO	RO-EN
AT	Transformer (teacher)	T	1.0×	27.54	**31.57**	**34.26**	**33.87**
	Transformer (12-1)	T	2.6×	26.09	30.30	32.76	32.39
	+ KD	T	2.7×	**27.61**	31.48	33.43	33.50
Vanilla NAT	NAT-FT[8]	1	15.6×	17.69	21.47	27.29	29.06
	EM[15]	1	16.4×	24.54	27.93	–	–
	GLAT[50]	1	15.3×	25.21	**29.84**	31.19	**32.04**
	Vanilla NAT (ours)	1	15.6×	20.42	24.88	29.21	29.37
	Vanilla NAT w/ rephraser	1	15.6×	**25.33**	29.57	**31.63**	31.72
CMLM	CMLM[35]	1	–	18.05	21.83	27.32	28.20
	CMLM + DSLP[28]	1	15.0×	21.76	25.30	30.29	30.89
	CMLM + AXE[69]	1	–	23.53	27.90	30.75	31.54
	CMLM + OAXE[70]	1	–	26.10	30.20	32.40	**33.30**
	CMLM (ours)	1	15.0×	18.21	22.86	28.15	28.97
	CMLM w/ rephraser	1	15.0×	**26.65**	**30.70**	**32.72**	33.03
CTC	CTC[24]	1	–	16.56	18.64	19.54	24.67
	Imputer[25]	1	–	25.80	28.40	32.30	31.70
	REDER[27]	1	15.5×	26.70	30.68	33.10	33.23
	Fully-NAT[26]	1	16.8×	27.20	31.39	33.71	34.16
	NMLA	1	14.7×	27.57	31.28	33.86	33.94
	CTC (ours)	1	14.7×	26.34	29.58	33.45	33.32
	CTC w/ rephraser	1	14.7×	27.32	30.97	33.80	33.84
	NMLA w/ rephraser	1	14.7×	**27.81**	**31.53**	34.08	34.03

在表7-1中，我们给出了三种非自回归基线模型在集成改写器前后的性能，其中 'AT' 表示自回归模型，'12-1' 表示由 12 层编码器和单层解码器构成的 Transformer 模型，'Iter' 表示模型迭代的轮数。通过对参考译文进行改写，我们的改写器方法在这三种基线模型上都取得了显著提升，并且在每种基线模型上都达到了与其最先进方法相当的水平。在 Vanilla NAT 上，改写器能带来约 3.6 BLEU 的平均提升。在 CMLM 模型上，改写器的平均提升为 6.2 BLEU，在 WMT14 英德与德英数据集上甚至有约 8 BLEU 的性能改进。在强基线模型 CTC 和 NMLA 上，改写器方法也有较好的效果，能够达到与自回归 Transformer 模型相当的性能，同时也有着十余倍于自回归模型的解码速度。

我们进一步在迭代式非自回归模型上验证了改写器方法的效果。在图7-3中，我们给出了 CMLM 模型在不同迭代轮数下的性能。在 WMT14 的英德和德英数

据集上，集成改写器的 CMLM 模型在各个迭代轮数下均相对于 CMLM 基线模型有稳定的提升，在迭代轮数较低时的提升尤为明显。集成改写器后的 CMLM 模型在只进行 2 次解码迭代时，就能超越基线模型迭代 8 次时的性能，表明改写器方法在迭代式模型上仍非常有效。



(a) WMT14 En-De　　(b) WMT14 De-En

图 7-3 在不同迭代轮数下，CMLM 模型在集成改写器前后的性能对比。

Figure 7-3 Performance comparison between CMLM and CMLM with rephraser under different iterations.

在上述实验中，我们均使用了序列级知识蒸馏技术，用教师模型的译文代替参考译文来训练非自回归的学生模型。由于我们的改写器方法也对参考译文做了修改，我们进一步在不使用知识蒸馏的原始数据集上进行实验，以验证改写器方法对原始参考译文的改写效果。如表7-2所示，改写器方法能将 CMLM 基线模型的性能平均提升 10 BLEU 以上，并且也优于其他基于 CMLM 的模型。与应用知识蒸馏的模型对比，改写器方法在 WMT14 原始数据集上的性能下降了约 3.5 BLEU，在 WMT16 原始数据集上的性能下降在 1 BLEU 以内。以上结果说明改写器方法目前还不能完全取代知识蒸馏，但这一相对较小的性能损失也说明了动态参考译文这一方向有较大的发展潜力。

表 7-2 集成改写器的 CMLM 模型在原始数据集上的性能。

Table 7-2 The performance of CMLM with rephraser on raw data.

Models	Iter	WMT14		WMT16	
		EN-DE	DE-EN	EN-RO	RO-EN
CMLM + AXE[69]	1	20.40	24.90	30.47	31.42
CMLM + OAXE[70]	1	22.40	26.80	–	–
CMLM (ours)	1	10.82	14.64	23.51	24.25
CMLM w/ rephraser	1	**23.12**	**27.44**	**32.30**	**32.07**

7.4.3 消融实验

在本节中，我们对改写器方法进行消融实验来验证其中各项设置的作用，包括模型大小、改写器架构、奖赏函数和退火策略等。我们在 WMT14 验证集上给出了相应的实验结果，如下所示。

模型大小与训练步数 改写器方法中引入的改写器模块由 2 个 Transformer 解码器层组成，一定程度上增大了模型的参数量和训练时间。另外，改写器方法在预训练模型后还需进行 3 万步的微调，这也使总训练时长略微上升。考虑到这些因素可能也会加强非自回归基线模型的性能，我们在强基线模型 CTC 上进行实验以观察这些因素的影响。我们用 'Params' 表示模型的参数量，'Speed' 表示相对自回归模型的解码加速比。如表7-3所示，加深解码器、增加训练步数对 CTC 模型的性能都只有轻微影响。相比之下，集成改写器的 CTC 模型显著优于 CTC 基线模型的各种变种，在解码速度上也略有优势。

表 7-3 额外增加 2 个解码器层、3 万训练步数对 CTC 模型的影响。

Table 7-3 The effect of 2 extra decoder layers and 30K extra training steps on the CTC model.

CTC		Params	Speed	BLEU
+2 layers	+30K steps			
		62.10M	14.7×	24.77
✓		70.51M	12.9×	24.89
	✓	62.10M	14.7×	24.80
✓	✓	70.51M	12.9×	24.87
CTC w/ rephraser		70.51M	14.7×	25.54

改写器架构 作为本章方法的核心部分，改写器的好坏会对模型最终的翻译性能产生决定性的影响，而改写器主要受其模型架构、奖赏函数和训练方法三方面影响。我们首先对改写器架构的影响进行分析。改写器的输入为参考译文和翻译模型解码器的输出，输出为对参考译文的改写结果。目前，我们采用 Transformer 解码器作为改写器的模型结构，层数 N_r 为 2，底层输入为参考译文的词嵌入，并通过注意力机制来关注到翻译模型解码器的输出。除当前结构外，还有其他结构也能支持改写器的输入输出形式。例如，我们可以交换改写器两个输入的位置，以翻译模型解码器的输出作为底层输入，通过注意力关注到参考译文的词嵌入，我们将这种架构称为 "swap''。另外，我们也可以用前馈神经网络将两个输入序列压缩为一个，并使用 Transformer 编码器结构来处理这一输入，我们将这种架构称为 "encoder''。此外，我们也可以直接改变改写器的层数，例如使用较浅的改写器 $N_r = 1$ 或更深的改写器 $N_r = 4$。表7-4给出了 Vanilla NAT 模型在不同改写器架构下的翻译性能。

上述结果显示，Transformer 解码器的模型架构优于编码器，交换解码器的

表 7-4 Vanilla NAT 模型在不同改写器架构下的翻译性能。

Table 7-4 The performance of Vanilla NAT with different rephraser architectures.

Models	base	swap	encoder	$N_r = 1$	$N_r = 4$
BLEU	24.06	24.01	23.61	23.53	24.09

两个输入对模型性能没有明显影响。将改写器的层数 N_r 从 2 增大到 4 不会显著改变模型的性能，但将层数降低到 1 会使模型性能明显下降，可见当前的改写器层数设置较为合理，能兼顾到模型性能与训练代价。

奖赏函数　改写器的奖赏函数包括模型的训练损失和与参考译文的语义相似度两部分，我们目前使用 BLEU 来衡量改写结果与参考译文的语义相似度。除 BLEU 外，我们也可以用其他机器翻译的评估指标来衡量语义相似度，例如 METEOR[136] 与 BERTScore[137]。在表7-5中，我们给出了 Vanilla NAT 模型在使用不同相似度函数时的性能。为了避免可能存在的偏差，我们同时用 BLEU、METEOR、BERTScore 这三种指标来评估模型性能。可以看到，使用 BLEU 作为相似度函数的模型在这三种评估指标下都表现出了最好的性能，而基于预训练模型的 BERTScore 的性能反而是最差的。这个结果较为出乎意料，因为 BERTScore 这类基于预训练模型的指标理应能更好地评估语义相似度。我们推测 BLEU 的优势在于它完全基于统计上的 n 元组匹配，因此具备较好的鲁棒性。

表 7-5 Vanilla NAT 模型使用不同相似度函数时的性能。

Table 7-5 The performance of Vanilla NAT with different similarity functions.

Metrics	BLEU	METEOR	BERTScore
BLEU	**24.06**	**52.88**	**84.24**
METEOR	23.71	52.65	84.11
BERTScore	23.07	52.02	83.80

退火策略　我们使用的退火策略将静态的超参 α 转换为了由超参 α_{max}、α_{min} 确定的动态退火策略。这种方式的好处是模型对超参的敏感性下降了，因此我们不需要精细地去调整超参。为了证实这一点，我们将超参数 $\{\alpha, \alpha_{max}, \alpha_{min}\}$ 均调整到最优，并使其偏离最优值 Δ 来重新训练模型。我们用 'Anneal' 表示使用退火策略，用 'Static' 表示使用静态超参 α，在表7-6给出了这两种情况下模型性能受超参数扰动的影响。可以看到，模型的性能受静态超参 α 的影响较大，而退火策略中超参 α_{max}、α_{min} 的扰动对模型性能基本没有影响，这证实了退火策略对超参不敏感的优势。

表 7-6 Vanilla NAT 模型在超参偏离最优值 Δ 的扰动下的性能。

Table 7-6 The performance of Vanilla NAT with deviation Δ from the optimal setting.

Δ	0	0.05	-0.05
Anneal	24.06	24.03	23.88
Static	23.97	23.55	23.63

7.4.4 结果分析

在本节中，我们对模型生成的结果做定量分析，以更好地理解改写器方法给模型带来的改变。在以下实验中，我们从不同方面分析了模型在 WMT14 英德测试集上的输出结果。

预测自信度 受多峰性问题影响，非自回归模型在预测时可能会同时考虑多种翻译结果，导致模型的预测自信度较低。因此，我们对比了基线模型与集成改写器的模型的预测自信度，以探究改写器方法是否可以减小多峰性问题的影响。我们用信息熵 $H(X) = -\sum_x p(x) \log p(x)$ 来衡量预测自信度，熵越低表示模型预测的概率分布越尖锐，即自信度越高。表7-7中的结果显示，模型在集成改写器后的预测自信度显著提升，说明改写器方法确实能减小多峰性问题的影响。

表 7-7 各个非自回归模型的平均信息熵。

Table 7-7 The average entropy of NAT models.

Models	**Vanilla NAT**	**CMLM**	**CTC**
w/o rephraser	2.07	2.64	0.26
w rephraser	1.35	1.52	0.06

重复词比例 受多峰性问题影响，非自回归模型的生成结果可能为多种译文的混合，其中一般会包含较多连续的重复词。为验证改写器方法是否能缓解重复词问题，我们对集成改写器前后的模型生成结果进行分析，计算重复词在译文中的比例。由于 CTC 模型本身就包含了去重操作，我们仅在 Vanilla NAT 模型与 CMLM 模型上进行实验，结果如表7-8所示。可以看到，集成改写器后模型生成结果中的重复词比例显著降低，说明了改写器方法的有效性。

表 7-8 不同模型预测结果中重复词的比例。

Table 7-8 The percentage of token repetitions in the generated outputs of different models.

Models	**Vanilla NAT**	**CMLM**
w/o rephraser	10.2%	15.4%
w rephraser	2.6%	2.3%

7.4.5　改写案例分析

我们通过表7-9中的改写案例来展示多峰性问题的影响和改写器的作用。可以看到，机器翻译中确实存在多峰性问题，例如参考译文中的 “but our loyalty is clear.’’ 与模型的输出结果 “but it is not about our loyalty.’’ 具备相同的语义，但两者之间没有对齐，会使得基于交叉熵损失的训练存在较大的噪声。在改写参考译文后，改写结果基本保持了原有的语义，并且与非自回归模型的输出有着很好的对齐，使得交叉熵损失也能较好地评估模型的输出，提升了训练的准确性。

表 7-9 在 WMT14 德英验证集上的一个改写结果案例。

Table 7-9 A case of the rephraser output from the validation set of WMT14 De-En.

Reference	Our identity might sometimes be fu@@ zz@@ y , but our loy@@ alty is clear .
NAT	Our identity may sometimes be urred , but it is not about our loy@@ alty .
Rephraser	Our identity might sometimes be vague , but that is not about our loy@@ alty .

7.5　本章小结

在非自回归模型的训练中，当模型输出的结果与参考译文没有对齐时，交叉熵损失就会给出偏高的损失值，无法正确评估模型的输出。我们认为不应使用静态的参考译文训练模型，并引入改写器结构来根据模型的输出动态改写参考译文，希望改写结果能保持参考译文原来的语义，并与与模型输出在形式上匹配。本文将对改写器的要求量化为奖赏函数，应用强化学习方法训练改写器，并直接使用改写结果来训练翻译模型。实验结果显示，改写器方法在多种非自回归基线模型上都有显著的提升，效果最好的模型能在将解码加速十余倍的同时达到与自回归模型相似的翻译质量。

第 8 章　总结与展望

8.1　总结

非自回归神经机器翻译模型在解码速度上相较于自回归模型有显著优势，但在翻译质量上与自回归模型有较大差距，主要原因在于极大似然估计的训练目标与非自回归模型表达能力的不匹配。本文对非自回归神经机器翻译模型的训练目标进行了全面的研究，旨在通过更适配非自回归模型的训练目标来使其学习到正确的翻译方式。

本文按照研究思路的不同将所研究的训练方法分成三个部分，分别为基于强化学习的序列级训练方法、基于 n 元组匹配的训练方法、基于动态参考译文的训练方法，详细研究内容如下：

第一部分：基于强化学习的序列级训练方法

1. 针对词级交叉熵损失无法准确评估模型输出的问题，我们提出对非自回归模型做序列级训练，从序列层面整体评估模型的输出结果并训练模型。序列级指标通常都是离散的，无法直接用于训练模型。对此，我们提出了基于强化学习的序列级训练方法，并针对非自回归模型改进强化学习算法来降低梯度估计的方差。实验结果表明，基于强化学习的序列级训练显著地改善了非自回归模型的翻译质量，针对非自回归模型设计的降低估计方差的算法能进一步增强模型性能，大幅减小了与自回归模型的性能差距。

第二部分：基于 n 元组匹配的训练方法

2. 在基于强化学习的序列级训练方法中，我们需要用梯度估计的方式来训练模型，这使得训练中过程存在噪声，也导致训练代价较大。由于机器翻译的评估指标主要基于 n 元组的匹配准确率来评估模型的翻译质量，若能直接基于 n 元组的匹配来设计损失函数，就能更准确、高效地训练非自回归模型。我们将这类训练目标形式化为最小化 n 元组词袋间的距离，并针对非自回归模型设计了优化一阶 n 元组词袋距离的高效算法。实验结果表明，n 元组词袋距离与翻译质量有较强的相关性，最小化这一训练目标能显著提升非自回归模型的翻译质量。

3. 基于隐式对齐的 CTC 模型能够考虑所有与参考译文的单调对齐来建模翻译概率，并具有生成变长译文的能力，是目前备受关注的一种非自回归模型结构。然而，CTC 模型的概率建模基于严格的单调对齐假设，无法处理参考译文与模型输出之间的非单调对齐，而非单调对齐是机器翻译中广泛存在的现象。针对这一问题，我们提出了基于二部图匹配和 n 元组匹配的两种非单调匹配方法，均能为 CTC 模型提供建模非单调对齐的能力。实验结果显示，我们的方法能显著改善模型的翻译质量，仅进行单轮解码就达到甚至超越了自回归模型的性能水平，并且仍相对自回归模型有可观的解码加速。

第三部分：基于动态参考译文的训练方法

4. 在机器翻译任务中，同一原文可能有多种正确的译文，而参考译文只是正确译文中的一种。当模型没有按照参考译文的形式输出结果时，交叉熵损失就无法正确地评估模型输出的好坏。我们认为不应使用静态的参考译文训练模型，并提出多样蒸馏与译文选择结合的方法来提供动态的参考译文，多样蒸馏模块为每个原文提供多个可选的参考译文，译文选择模块则从多个参考译文中选择最适合当前模型输出的一个来计算损失函数。实验结果显示，我们的方法显著地改善了 CTC 基线模型的翻译质量，并能与非单调对齐方面的改进有效结合，使非自回归模型在所用的数据集上均达到甚至超越了自回归模型的翻译质量。

5. 当非自回归模型没有按照参考译文的形式输出结果时，交叉熵损失就会给出偏高的损失值，无法正确评估模型的输出。多样蒸馏方案生成多个参考译文以供模型选择，然而多样蒸馏的代价通常较高，并且生成的参考译文数目也比较有限，不一定能覆盖所有情况。我们进一步引入改写器结构来根据模型的输出动态改写参考译文，希望改写结果能保持参考译文原来的语义，并与与模型输出在形式上匹配。由于没有对改写器的直接监督信号，我们将对改写器的要求量化为奖赏函数，并通过强化学习方法来优化改写器。实验结果显示，改写器方法在多种非自回归基线模型上都有显著的提升，集成改写器的 CTC 模型能在将解码加速十余倍的同时达到与自回归模型相似的翻译质量。

8.2 展望

自 2017 年问世以来，非自回归机器翻译模型已经从最初远落后于自回归模型的翻译质量发展到如今与自回归模型相当的性能水平。展望非自回归模型未来的发展，我认为下面这几个方向是值得研究的。

首先，我们目前为非自回归模型设计的训练目标主要还是基于实践层面的，能够解决交叉熵损失不够准确的问题，但还是缺少理论层面上的依据。因此，在未来的研究中，可以在理论层面上分析非自回归模型能够优化什么样的训练目标，为非自回归模型的训练建立理论框架，用于指导非自回归模型训练目标的设计。

其次，目前非自回归模型已经发展到了与自回归模型相当的性能水平，但主要还是依赖于自回归模型进行知识蒸馏，而知识蒸馏会固定非自回归模型的能力上限。为了进一步发掘非自回归模型的潜力，未来的研究应尝试摆脱模型对知识蒸馏的依赖，这可以通过提升非自回归模型的建模能力和改进非自回归模型的训练策略来实现。

最后，我们可以扩展非自回归模型的应用场景，探索它在机器翻译的各个场景下的应用，例如多语言翻译、同声传译、语音翻译等。另外，我们也可以探索非自回归模型在机器翻译之外的其他应用场景，例如语音识别、语音合成等。

参考文献

[1] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need [C/OL]//Guyon I, Luxburg U V, Bengio S, et al. Advances in Neural Information Processing Systems: volume 30. Curran Associates, Inc., 2017. https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[2] Weaver W. Translation [C]//Proceedings of the Conference on Mechanical Translation. 1952.

[3] Brown P F, Cocke J, Della Pietra S A, et al. A statistical approach to machine translation [J/OL]. Computational Linguistics, 1990, 16(2): 79-85. https://aclanthology.org/J90-2002.

[4] Brown P F, Della Pietra S A, Della Pietra V J, et al. The mathematics of statistical machine translation: Parameter estimation [J/OL]. Computational Linguistics, 1993, 19(2): 263-311. https://aclanthology.org/J93-2003.

[5] Cho K, van Merriënboer B, Gulcehre C, et al. Learning phrase representations using RNN encoder–decoder for statistical machine translation [C/OL]//Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). Doha, Qatar: Association for Computational Linguistics, 2014: 1724-1734. https://www.aclweb.org/anthology/D14-1179. DOI: 10.3115/v1/D14-1179.

[6] Sutskever I, Vinyals O, Le Q V. Sequence to sequence learning with neural networks [C/OL]//Ghahramani Z, Welling M, Cortes C, et al. Advances in Neural Information Processing Systems: volume 27. Curran Associates, Inc., 2014. https://proceedings.neurips.cc/paper_files/paper/2014/file/a14ac55a4f27472c5d894ec1c3c743d2-Paper.pdf.

[7] Bahdanau D, Cho K, Bengio Y. Neural machine translation by jointly learning to align and translate [C/OL]//Bengio Y, LeCun Y. 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings. 2015. http://arxiv.org/abs/1409.0473.

[8] Gu J, Bradbury J, Xiong C, et al. Non-autoregressive neural machine translation [C/OL]//6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings. 2018. https://openreview.net/forum?id=B1l8BtlCb.

[9] He D, Xia Y, Qin T, et al. Dual learning for machine translation [C/OL]//Lee D, Sugiyama M, Luxburg U, et al. Advances in Neural Information Processing Systems: volume 29. Curran Associates, Inc., 2016. https://proceedings.neurips.cc/paper_files/paper/2016/file/5b69b9cb83065d403869739ae7f0995e-Paper.pdf.

[10] Ba J L, Kiros J R, Hinton G E. Layer normalization [Z]. 2016.

[11] Huang F, Tao T, Zhou H, et al. On the learning of non-autoregressive transformers [C/OL]//Chaudhuri K, Jegelka S, Song L, et al. Proceedings of Machine Learning Research: volume 162 Proceedings of the 39th International Conference on Machine Learning. PMLR, 2022: 9356-9376. https://proceedings.mlr.press/v162/huang22k.html.

[12] Hinton G, Vinyals O, Dean J. Distilling the knowledge in a neural network [Z]. 2015.

[13] Kim Y, Rush A M. Sequence-level knowledge distillation [C/OL]//Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. Austin, Texas: Association for Computational Linguistics, 2016: 1317-1327. https://www.aclweb.org/anthology/D16-1139. DOI: 10.18653/v1/D16-1139.

[14] Zhou C, Gu J, Neubig G. Understanding knowledge distillation in non-autoregressive machine translation [C/OL]//8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020. 2020. https://openreview.net/forum?id=BygFVAEKDH.

[15] Sun Z, Yang Y. An EM approach to non-autoregressive conditional sequence generation [C/OL]//III H D, Singh A. Proceedings of Machine Learning Research: volume 119 Proceedings of the 37th International Conference on Machine Learning. PMLR, 2020: 9249-9258. http://proceedings.mlr.press/v119/sun20c.html.

[16] McLachlan G J, Krishnan T. The em algorithm and extensions [M]. John Wiley & Sons, 2007.

[17] Ganchev K, Graça J, Gillenwater J, et al. Posterior regularization for structured latent variable models [J/OL]. Journal of Machine Learning Research, 2010, 11(67): 2001-2049. http://jmlr.org/papers/v11/ganchev10a.html.

[18] Ding L, Wang L, Liu X, et al. Understanding and improving lexical choice in non-autoregressive translation [C/OL]//International Conference on Learning Representations. 2021. https://openreview.net/forum?id=ZTFeSBIX9C.

[19] Sun Z, Li Z, Wang H, et al. Fast structured decoding for sequence models [M/OL]//Advances in Neural Information Processing Systems 32. 2019: 3016-3026. http://papers.nips.cc/paper/8566-fast-structured-decoding-for-sequence-models.pdf.

[20] Graves A, Fernández S, Gomez F, et al. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks [C/OL]//ICML '06: Proceedings of the 23rd International Conference on Machine Learning. New York, NY, USA: Association for Computing Machinery, 2006: 369–376. https://doi.org/10.1145/1143844.1143891.

[21] Huang F, Zhou H, Liu Y, et al. Directed acyclic transformer for non-autoregressive machine translation [C/OL]//Chaudhuri K, Jegelka S, Song L, et al. Proceedings of Machine Learning Research: volume 162 Proceedings of the 39th International Conference on Machine Learning. PMLR, 2022: 9410-9428. https://proceedings.mlr.press/v162/huang22m.html.

[22] Lafferty J D, McCallum A, Pereira F C N. Conditional random fields: Probabilistic models for segmenting and labeling sequence data [C]//ICML '01: Proceedings of the Eighteenth International Conference on Machine Learning. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2001: 282–289.

[23] Viterbi A. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm [J/OL]. IEEE Transactions on Information Theory, 1967, 13(2): 260-269. DOI: 10.1109/TIT.1967.1054010.

[24] Libovický J, Helcl J. End-to-end non-autoregressive neural machine translation with connectionist temporal classification [C/OL]//Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. Brussels, Belgium: Association for Computational Linguistics, 2018: 3016-3021. https://www.aclweb.org/anthology/D18-1336. DOI: 10.18653/v1/D18-1336.

[25] Saharia C, Chan W, Saxena S, et al. Non-autoregressive machine translation with latent alignments [C/OL]//Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP). Online: Association for Computational Linguistics, 2020: 1098-1108. https://www.aclweb.org/anthology/2020.emnlp-main.83. DOI: 10.18653/v1/2020.emnlp-main.83.

[26] Gu J, Kong X. Fully non-autoregressive neural machine translation: Tricks of the trade [C/OL]//Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021. Online: Association for Computational Linguistics, 2021: 120-133. https://aclanthology.org/2021.findings-acl.11. DOI: 10.18653/v1/2021.findings-acl.11.

[27] Zheng Z, Zhou H, Huang S, et al. Duplex sequence-to-sequence learning for reversible machine translation [C/OL]//Ranzato M, Beygelzimer A, Dauphin Y, et al. Advances in Neural Information Processing Systems: volume 34. Curran Associates, Inc., 2021: 21070-21084. https://proceedings.neurips.cc/paper_files/paper/2021/file/afecc60f82be41c1b52f6705ec69e0f1-Paper.pdf.

[28] Huang C, Zhou H, ZaÃŕane O R, et al. Non-autoregressive translation with layer-wise prediction and deep supervision [J/OL]. Proceedings of the AAAI Conference on Artificial Intelligence, 2022, 36(10): 10776-10784. https://ojs.aaai.org/index.php/AAAI/article/view/21323. DOI: 10.1609/aaai.v36i10.21323.

[29] Zhan J, Chen Q, Chen B, et al. Non-autoregressive translation with dependency-aware decoder [Z]. 2022.

[30] Shao C, Ma Z, Feng Y. Viterbi decoding of directed acyclic transformer for non-autoregressive machine translation [C/OL]//Findings of the Association for Computational Linguistics: EMNLP 2022. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, 2022: 4390-4397. https://aclanthology.org/2022.findings-emnlp.322.

[31] Qian L, Wang M, Liu Y, et al. Diff-glat: Diffusion glancing transformer for parallel sequence to sequence learning [Z]. 2022.

[32] Ma Z, Shao C, Gui S, et al. Fuzzy alignments in directed acyclic graph for non-autoregressive machine translation [C/OL]//International Conference on Learning Representations. 2023. https://openreview.net/forum?id=LSz-gQyd0zE.

[33] Qi W, Gong Y, Jiao J, et al. Bang: Bridging autoregressive and non-autoregressive generation with large scale pretraining [C/OL]//Meila M, Zhang T. Proceedings of Machine Learning Research: volume 139 Proceedings of the 38th International Conference on Machine Learning. PMLR, 2021: 8630-8639. https://proceedings.mlr.press/v139/qi21a.html.

[34] Jiang T, Huang S, Zhang Z, et al. Improving non-autoregressive generation with mixup training [Z]. 2021.

[35] Ghazvininejad M, Levy O, Liu Y, et al. Mask-predict: Parallel decoding of conditional masked language models [C/OL]//Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP). 2019: 6112-6121. https://www.aclweb.org/anthology/D19-1633.

[36] Huang F, Ke P, Huang M. Directed acyclic transformer pre-training for high-quality non-autoregressive text generation [Z]. 2023.

[37] Kingma D P, Welling M. Auto-encoding variational bayes [Z]. 2022.

[38] van den Oord A, Vinyals O, kavukcuoglu k. Neural discrete representation learning [C/OL]// Guyon I, Luxburg U V, Bengio S, et al. Advances in Neural Information Processing Systems: volume 30. Curran Associates, Inc., 2017. https://proceedings.neurips.cc/paper/2017/file/7a98af17e63a0ac09ce2e96d03992fbc-Paper.pdf.

[39] Shu R, Lee J, Nakayama H, et al. Latent-variable non-autoregressive neural machine translation with deterministic inference using a delta posterior [C/OL]//The Thirty-Fourth AAAI

Conference on Artificial Intelligence, AAAI 2020, New York, NY, USA, February 7-12, 2020. AAAI Press, 2020: 8846-8853. https://aaai.org/ojs/index.php/AAAI/article/view/6413.

[40] Ma X, Zhou C, Li X, et al. FlowSeq: Non-autoregressive conditional sequence generation with generative flow [C/OL]//Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP). Hong Kong, China: Association for Computational Linguistics, 2019: 4282-4292. https://www.aclweb.org/anthology/D19-1437. DOI: 10.18653/v1/D19-1437.

[41] Rezende D, Mohamed S. Variational inference with normalizing flows [C/OL]//Bach F, Blei D. Proceedings of Machine Learning Research: volume 37 Proceedings of the 32nd International Conference on Machine Learning. Lille, France: PMLR, 2015: 1530-1538. https://proceedings.mlr.press/v37/rezende15.html.

[42] Kaiser L, Bengio S, Roy A, et al. Fast decoding in sequence models using discrete latent variables [C/OL]//Dy J, Krause A. Proceedings of Machine Learning Research: volume 80 Proceedings of the 35th International Conference on Machine Learning. PMLR, 2018: 2390-2399. http://proceedings.mlr.press/v80/kaiser18a.html.

[43] Roy A, Vaswani A, Neelakantan A, et al. Theory and experiments on vector quantized autoencoders [Z]. 2018.

[44] Bao Y, Huang S, Xiao T, et al. Non-autoregressive translation by learning target categorical codes [C/OL]//Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Online: Association for Computational Linguistics, 2021: 5749-5759. https://aclanthology.org/2021.naacl-main.458. DOI: 10.18653/v1/2021.naacl-main.458.

[45] Bao Y, Zhou H, Huang S, et al. latent-GLAT: Glancing at latent variables for parallel text generation [C/OL]//Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Dublin, Ireland: Association for Computational Linguistics, 2022: 8398-8409. https://aclanthology.org/2022.acl-long.575. DOI: 10.18653/v1/2022.acl-long.575.

[46] Ran Q, Lin Y, Li P, et al. Guiding non-autoregressive neural machine translation decoding with reordering information [C/OL]//Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Virtual Event, February 2-9, 2021. AAAI Press, 2021: 13727-13735. https://ojs.aaai.org/index.php/AAAI/article/view/17618.

[47] Song J, Kim S, Yoon S. AligNART: Non-autoregressive neural machine translation by jointly learning to estimate alignment and translate [C/OL]//Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, 2021: 1-14. https://aclanthology.org/2021.emnlp-main.1. DOI: 10.18653/v1/2021.emnlp-main.1.

[48] Akoury N, Krishna K, Iyyer M. Syntactically supervised transformers for faster neural machine translation [C/OL]//Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. Florence, Italy: Association for Computational Linguistics, 2019: 1269-1281. https://www.aclweb.org/anthology/P19-1122. DOI: 10.18653/v1/P19-1122.

[49] Liu Y, Wan Y, Zhang J, et al. Enriching non-autoregressive transformer with syntactic and semantic structures for neural machine translation [C/OL]//Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics:

Main Volume. Online: Association for Computational Linguistics, 2021: 1235-1244. https://aclanthology.org/2021.eacl-main.105. DOI: 10.18653/v1/2021.eacl-main.105.

[50] Qian L, Zhou H, Bao Y, et al. Glancing transformer for non-autoregressive neural machine translation [C/OL]//Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers). Online: Association for Computational Linguistics, 2021: 1993-2003. https://aclanthology.org/2021.acl-long.155. DOI: 10.18653/v1/2021.acl-long.155.

[51] Wang C, Zhang J, Chen H. Semi-autoregressive neural machine translation [C/OL]// Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. Brussels, Belgium: Association for Computational Linguistics, 2018: 479-488. https://www.aclweb.org/anthology/D18-1044. DOI: 10.18653/v1/D18-1044.

[52] Ran Q, Lin Y, Li P, et al. Learning to recover from multi-modality errors for non-autoregressive neural machine translation [C/OL]//Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. Online: Association for Computational Linguistics, 2020: 3059-3069. https://www.aclweb.org/anthology/2020.acl-main.277. DOI: 10.18653/v1/2020.acl-main.277.

[53] Wang Q, Hu X, Chen M. Hybrid-regressive neural machine translation [Z]. 2022.

[54] Stern M, Shazeer N, Uszkoreit J. Blockwise parallel decoding for deep autoregressive models [C/OL]//Bengio S, Wallach H, Larochelle H, et al. Advances in Neural Information Processing Systems: volume 31. Curran Associates, Inc., 2018. https://proceedings.neurips.cc/paper/2018/file/c4127b9194fe8562c64dc0f5bf2c93bc-Paper.pdf.

[55] Xia H, Ge T, Wei F, et al. Lossless speedup of autoregressive translation with generalized aggressive decoding [Z]. 2022.

[56] Stern M, Chan W, Kiros J, et al. Insertion transformer: Flexible sequence generation via insertion operations [C/OL]//Chaudhuri K, Salakhutdinov R. Proceedings of Machine Learning Research: volume 97 Proceedings of the 36th International Conference on Machine Learning. PMLR, 2019: 5976-5985. https://proceedings.mlr.press/v97/stern19a.html.

[57] Chan W, Kitaev N, Guu K, et al. Kermit: Generative insertion-based modeling for sequences [Z]. 2019.

[58] Lee J, Mansimov E, Cho K. Deterministic non-autoregressive neural sequence modeling by iterative refinement [C/OL]//Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. Brussels, Belgium: Association for Computational Linguistics, 2018: 1173-1182. https://www.aclweb.org/anthology/D18-1149. DOI: 10.18653/v1/D18-1149.

[59] Guo J, Xu L, Chen E. Jointly masked sequence-to-sequence model for non-autoregressive neural machine translation [C/OL]//Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. Online: Association for Computational Linguistics, 2020: 376-385. https://aclanthology.org/2020.acl-main.36. DOI: 10.18653/v1/2020.acl-main.36.

[60] Kasai J, Cross J, Ghazvininejad M, et al. Non-autoregressive machine translation with disentangled context transformer [C/OL]//Proceedings of Machine Learning Research: volume 119 Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event. PMLR, 2020: 5144-5155. http://proceedings.mlr.press/v119/kasai20a.html.

[61] Ghazvininejad M, Levy O, Zettlemoyer L. Semi-autoregressive training improves mask-predict decoding [Z]. 2020.

[62] Gu J, Wang C, Zhao J. Levenshtein transformer [C/OL]//Wallach H M, Larochelle H, Beygelzimer A, et al. Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada. 2019: 11179-11189. https://proceedings.neurips.cc/paper/2019/hash/675f9820626f5bc0afb47b57890b466e-Abstract.html.

[63] Geng X, Feng X, Qin B. Learning to rewrite for non-autoregressive neural machine translation [C/OL]//Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, 2021: 3297-3308. https://aclanthology.org/2021.emnlp-main.265. DOI: 10.18653/v1/2021.emnlp-main.265.

[64] Gao Z, Guo J, Tan X, et al. Difformer: Empowering diffusion models on the embedding space for text generation [Z]. 2023.

[65] Wang Y, Tian F, He D, et al. Non-autoregressive machine translation with auxiliary regularization [C/OL]//The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, Honolulu, Hawaii, USA, 2019. AAAI Press, 2019: 5377-5384. https://doi.org/10.1609/aaai.v33i01.33015377.

[66] Shao C, Feng Y, Zhang J, et al. Retrieving sequential information for non-autoregressive neural machine translation [C/OL]//Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. Florence, Italy: Association for Computational Linguistics, 2019: 3013-3024. https://www.aclweb.org/anthology/P19-1288. DOI: 10.18653/v1/P19-1288.

[67] Shao C, Zhang J, Feng Y, et al. Minimizing the bag-of-ngrams difference for non-autoregressive neural machine translation [J/OL]. Proceedings of the AAAI Conference on Artificial Intelligence, 2020, 34(01): 198-205. https://ojs.aaai.org/index.php/AAAI/article/view/5351. DOI: 10.1609/aaai.v34i01.5351.

[68] Shao C, Feng Y, Zhang J, et al. Sequence-level training for non-autoregressive neural machine translation [J/OL]. Computational Linguistics, 2021, 47(4): 891-925. https://aclanthology.org/2021.cl-4.29. DOI: 10.1162/coli_a_00421.

[69] Ghazvininejad M, Karpukhin V, Zettlemoyer L, et al. Aligned cross entropy for non-autoregressive machine translation [C/OL]//Proceedings of Machine Learning Research: volume 119 Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event. PMLR, 2020: 3515-3523. http://proceedings.mlr.press/v119/ghazvininejad20a.html.

[70] Du C, Tu Z, Jiang J. Order-agnostic cross entropy for non-autoregressive machine translation [C/OL]//Meila M, Zhang T. Proceedings of Machine Learning Research: volume 139 Proceedings of the 38th International Conference on Machine Learning. PMLR, 2021: 2849-2859. https://proceedings.mlr.press/v139/du21c.html.

[71] Shao C, Feng Y. Non-monotonic latent alignments for ctc-based non-autoregressive machine translation [C/OL]//Koyejo S, Mohamed S, Agarwal A, et al. Advances in Neural Information Processing Systems: volume 35. Curran Associates, Inc., 2022: 8159-8173. https://proceedings.neurips.cc/paper_files/paper/2022/file/35f805e65c77652efa731edc10c8e3a6-Paper-Conference.pdf.

[72] Shao C, Wu X, Feng Y. One reference is not enough: Diverse distillation with reference selection for non-autoregressive translation [C/OL]//Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Seattle, United States: Association for Computational Linguistics,

2022: 3779-3791. https://aclanthology.org/2022.naacl-main.277. DOI: 10.18653/v1/2022.naacl-main.277.

[73] Shao C, Zhang J, Zhou J, et al. Rephrasing the reference for non-autoregressive machine translation [C]//The Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI 2023, Washington, DC, USA, February 7-14, 2023. AAAI Press, 2023.

[74] Guo L, Liu J, Zhu X, et al. Fast sequence generation with multi-agent reinforcement learning [Z]. 2021.

[75] Li Y, Cui L, Yin Y, et al. Multi-granularity optimization for non-autoregressive translation [C/OL]//Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, 2022: 5073-5084. https://aclanthology.org/2022.emnlp-main.339.

[76] Liu G, Yang Z, Tao T, et al. Don't take it literally: An edit-invariant sequence loss for text generation [C/OL]//Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Seattle, United States: Association for Computational Linguistics, 2022: 2055-2078. https://aclanthology.org/2022.naacl-main.150. DOI: 10.18653/v1/2022.naacl-main.150.

[77] Du C, Tu Z, Wang L, et al. ngram-OAXE: Phrase-based order-agnostic cross entropy for non-autoregressive machine translation [C/OL]//Proceedings of the 29th International Conference on Computational Linguistics. Gyeongju, Republic of Korea: International Committee on Computational Linguistics, 2022: 5035-5045. https://aclanthology.org/2022.coling-1.446.

[78] Wu Y, Schuster M, Chen Z, et al. Google's neural machine translation system: Bridging the gap between human and machine translation [Z]. 2016.

[79] Papineni K, Roukos S, Ward T, et al. Bleu: a method for automatic evaluation of machine translation [C/OL]//Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics. Philadelphia, Pennsylvania, USA: Association for Computational Linguistics, 2002: 311-318. https://www.aclweb.org/anthology/P02-1040. DOI: 10.3115/1073083.1073135.

[80] Lin C Y. ROUGE: A package for automatic evaluation of summaries [C/OL]//Text Summarization Branches Out. Barcelona, Spain: Association for Computational Linguistics, 2004: 74-81. https://www.aclweb.org/anthology/W04-1013.

[81] Ranzato M, Chopra S, Auli M, et al. Sequence level training with recurrent neural networks [C/OL]//Bengio Y, LeCun Y. 4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings. 2016. http://arxiv.org/abs/1511.06732.

[82] Bahdanau D, Brakel P, Xu K, et al. An actor-critic algorithm for sequence prediction [C/OL]// 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings. 2017. https://openreview.net/forum?id=SJDaqqveg.

[83] Yang Z, Chen W, Wang F, et al. Improving neural machine translation with conditional sequence generative adversarial nets [C/OL]//Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers). New Orleans, Louisiana: Association for Computational Linguistics, 2018: 1346-1355. https://www.aclweb.org/anthology/N18-1122. DOI: 10.18653/v1/N18-1122.

[84] Wu L, Tian F, Qin T, et al. A study of reinforcement learning for neural machine translation [C/OL]//Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. Brussels, Belgium: Association for Computational Linguistics, 2018: 3612-3621. https://www.aclweb.org/anthology/D18-1397. DOI: 10.18653/v1/D18-1397.

[85] Williams R J. Simple statistical gradient-following algorithms for connectionist reinforcement learning [J/OL]. Mach. Learn., 1992, 8(3–4): 229-256. https://doi.org/10.1007/BF00992696.

[86] Ng A Y, Harada D, Russell S J. Policy invariance under reward transformations: Theory and application to reward shaping [C]//ICML '99: Proceedings of the Sixteenth International Conference on Machine Learning. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1999: 278-287.

[87] Weaver L, Tao N. The optimal reward baseline for gradient-based reinforcement learning [C]//UAI '01: Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2001: 538-545.

[88] Williams R J, Zipser D. A learning algorithm for continually running fully recurrent neural networks [J/OL]. Neural Comput., 1989, 1(2): 270-280. https://doi.org/10.1162/neco.1989.1.2.270.

[89] Bengio S, Vinyals O, Jaitly N, et al. Scheduled sampling for sequence prediction with recurrent neural networks [C/OL]//Cortes C, Lawrence N, Lee D, et al. Advances in Neural Information Processing Systems: volume 28. Curran Associates, Inc., 2015. https://proceedings.neurips.cc/paper_files/paper/2015/file/e995f98d56967d946471af29d7bf99f1-Paper.pdf.

[90] Venkatraman A, Hebert M, Bagnell J. Improving multi-step prediction of learned time series models [J/OL]. Proceedings of the AAAI Conference on Artificial Intelligence, 2015, 29(1). https://ojs.aaai.org/index.php/AAAI/article/view/9590. DOI: 10.1609/aaai.v29i1.9590.

[91] Zhang W, Feng Y, Meng F, et al. Bridging the gap between training and inference for neural machine translation [C/OL]//Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. Florence, Italy: Association for Computational Linguistics, 2019: 4334-4343. https://www.aclweb.org/anthology/P19-1426. DOI: 10.18653/v1/P19-1426.

[92] Sutton R S, McAllester D, Singh S, et al. Policy gradient methods for reinforcement learning with function approximation [C]//NIPS'99: Proceedings of the 12th International Conference on Neural Information Processing Systems. Cambridge, MA, USA: MIT Press, 1999: 1057-1063.

[93] Yu L, Zhang W, Wang J, et al. Seqgan: Sequence generative adversarial nets with policy gradient [J/OL]. Proceedings of the AAAI Conference on Artificial Intelligence, 2017, 31(1). https://ojs.aaai.org/index.php/AAAI/article/view/10804. DOI: 10.1609/aaai.v31i1.10804.

[94] Wu L, Xia Y, Tian F, et al. Adversarial neural machine translation [C/OL]//Zhu J, Takeuchi I. Proceedings of Machine Learning Research: volume 95 Proceedings of The 10th Asian Conference on Machine Learning. PMLR, 2018: 534-549. http://proceedings.mlr.press/v95/wu18a.html.

[95] Shen S, Cheng Y, He Z, et al. Minimum risk training for neural machine translation [C/OL]//Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Berlin, Germany: Association for Computational Linguistics, 2016: 1683-1692. https://www.aclweb.org/anthology/P16-1159. DOI: 10.18653/v1/P16-1159.

[96] Norouzi M, Bengio S, Chen z, et al. Reward augmented maximum likelihood for neural structured prediction [C/OL]//Lee D, Sugiyama M, Luxburg U, et al. Advances in Neural Infor-

mation Processing Systems: volume 29. Curran Associates, Inc., 2016. https://proceedings.neurips.cc/paper_files/paper/2016/file/2f885d0fbe2e131bfc9d98363e55d1d4-Paper.pdf.

[97] Edunov S, Ott M, Auli M, et al. Classical structured prediction losses for sequence to sequence learning [C/OL]//Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers). New Orleans, Louisiana: Association for Computational Linguistics, 2018: 355-364. https://www.aclweb.org/anthology/N18-1033. DOI: 10.18653/v1/N18-1033.

[98] Ma S, Sun X, Wang Y, et al. Bag-of-words as target for neural machine translation [C/OL]// Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers). Melbourne, Australia: Association for Computational Linguistics, 2018: 332-338. https://www.aclweb.org/anthology/P18-2053. DOI: 10.18653/v1/P18-2053.

[99] Shao C, Chen X, Feng Y. Greedy search with probabilistic n-gram matching for neural machine translation [C/OL]//Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. Brussels, Belgium: Association for Computational Linguistics, 2018: 4778-4784. https://www.aclweb.org/anthology/D18-1510. DOI: 10.18653/v1/D18-1510.

[100] Sennrich R, Haddow B, Birch A. Neural machine translation of rare words with subword units [C/OL]//Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Berlin, Germany: Association for Computational Linguistics, 2016: 1715-1725. https://www.aclweb.org/anthology/P16-1162. DOI: 10.18653/v1/P16-1162.

[101] Kingma D P, Ba J. Adam: A method for stochastic optimization [Z]. 2017.

[102] Guo J, Tan X, He D, et al. Non-autoregressive neural machine translation with enhanced decoder input [J/OL]. Proceedings of the AAAI Conference on Artificial Intelligence, 2019, 33(01): 3723-3730. https://ojs.aaai.org/index.php/AAAI/article/view/4257. DOI: 10.1609/aaai.v33i01.33013723.

[103] Li Z, Lin Z, He D, et al. Hint-based training for non-autoregressive machine translation [C/OL]//Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP). Hong Kong, China: Association for Computational Linguistics, 2019: 5708-5713. https://www.aclweb.org/anthology/D19-1573. DOI: 10.18653/v1/D19-1573.

[104] Joachims T. Text categorization with support vector machines: Learning with many relevant features [C/OL]//ECML'98: Proceedings of the 10th European Conference on Machine Learning. Berlin, Heidelberg: Springer-Verlag, 1998: 137-142. https://doi.org/10.1007/BFb0026683.

[105] Pang B, Lee L, Vaithyanathan S. Thumbs up? sentiment classification using machine learning techniques [C/OL]//Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002). Association for Computational Linguistics, 2002: 79-86. https://www.aclweb.org/anthology/W02-1011. DOI: 10.3115/1118693.1118704.

[106] Li B, Zhao Z, Liu T, et al. Weighted neural bag-of-n-grams model: New baselines for text classification [C/OL]//Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers. Osaka, Japan: The COLING 2016 Organizing Committee, 2016: 1591-1600. https://www.aclweb.org/anthology/C16-1150.

[107] Joulin A, Grave E, Bojanowski P, et al. Bag of tricks for efficient text classification [C/OL]// Proceedings of the 15th Conference of the European Chapter of the Association for Computa-

tional Linguistics: Volume 2, Short Papers. Valencia, Spain: Association for Computational Linguistics, 2017: 427-431. https://aclanthology.org/E17-2068.

[108] Wei B, Wang M, Zhou H, et al. Imitation learning for non-autoregressive neural machine translation [C/OL]//Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. Florence, Italy: Association for Computational Linguistics, 2019: 1304-1312. https://www.aclweb.org/anthology/P19-1125. DOI: 10.18653/v1/P19-1125.

[109] Bao Y, Zhou H, Feng J, et al. Non-autoregressive transformer by position learning [Z]. 2019.

[110] Guo J, Tan X, Xu L, et al. Fine-tuning by curriculum learning for non-autoregressive neural machine translation [J/OL]. Proceedings of the AAAI Conference on Artificial Intelligence, 2020, 34(05): 7839-7846. https://ojs.aaai.org/index.php/AAAI/article/view/6289. DOI: 10.1609/aaai.v34i05.6289.

[111] Kasner Z, Libovický J, Helcl J. Improving fluency of non-autoregressive machine translation [Z]. 2020.

[112] Stewart R, Andriluka M, Ng A. End-to-end people detection in crowded scenes [J]. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016: 2325-2333.

[113] Kuhn H W. The hungarian method for the assignment problem [J/OL]. Naval Research Logistics Quarterly, 1955, 2(1-2): 83-97. https://onlinelibrary.wiley.com/doi/abs/10.1002/nav.3800020109. DOI: https://doi.org/10.1002/nav.3800020109.

[114] Carion N, Massa F, Synnaeve G, et al. End-to-end object detection with transformers [C]// European Conference on Computer Vision. Springer, 2020: 213-229.

[115] Graves A. Sequence transduction with recurrent neural networks [Z]. 2012.

[116] Heafield K. KenLM: Faster and smaller language model queries [C/OL]//Proceedings of the Sixth Workshop on Statistical Machine Translation. Edinburgh, Scotland: Association for Computational Linguistics, 2011: 187-197. https://aclanthology.org/W11-2123.

[117] Ott M, Edunov S, Baevski A, et al. fairseq: A fast, extensible toolkit for sequence modeling [C/OL]//Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations). Minneapolis, Minnesota: Association for Computational Linguistics, 2019: 48-53. https://aclanthology.org/N19-4009. DOI: 10.18653/v1/N19-4009.

[118] Kasai J, Pappas N, Peng H, et al. Deep encoder, shallow decoder: Reevaluating non-autoregressive machine translation [C/OL]//International Conference on Learning Representations. 2021. https://openreview.net/forum?id=KpfasTaLUpq.

[119] Huang X S, Perez F, Volkovs M. Improving non-autoregressive translation models without distillation [C/OL]//International Conference on Learning Representations. 2022. https://openreview.net/forum?id=I2Hw58KHp8O.

[120] Castilho S, Moorkens J, Gaspari F, et al. Is neural machine translation the new state of the art? [J/OL]. Prague Bull. Math. Linguistics, 2017, 108: 109-120. http://ufal.mff.cuni.cz/pbml/108/art-castilho-moorkens-gaspari-tinsley-calixto-way.pdf.

[121] Fomicheva M, Specia L. Taking MT Evaluation Metrics to Extremes: Beyond Correlation with Human Judgments [J/OL]. Computational Linguistics, 2019, 45(3): 515-558. https://doi.org/10.1162/coli_a_00356. DOI: 10.1162/coli_a_00356.

[122] Li J, Monroe W, Jurafsky D. A simple, fast diverse decoding algorithm for neural generation [Z]. 2016.

[123] Vijayakumar A K, Cogswell M, Selvaraju R R, et al. Diverse beam search for improved description of complex scenes [C/OL]//McIlraith S A, Weinberger K Q. Proceedings of

the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018. AAAI Press, 2018: 7371-7379. https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/17329.

[124] He X, Haffari G, Norouzi M. Sequence to sequence mixture model for diverse machine translation [C/OL]//Proceedings of the 22nd Conference on Computational Natural Language Learning. Brussels, Belgium: Association for Computational Linguistics, 2018: 583-592. https://aclanthology.org/K18-1056. DOI: 10.18653/v1/K18-1056.

[125] Shen T, Ott M, Auli M, et al. Mixture models for diverse machine translation: Tricks of the trade [C/OL]//Chaudhuri K, Salakhutdinov R. Proceedings of Machine Learning Research: volume 97 Proceedings of the 36th International Conference on Machine Learning. PMLR, 2019: 5719-5728. https://proceedings.mlr.press/v97/shen19c.html.

[126] Sun Z, Huang S, Wei H, et al. Generating diverse translation by manipulating multi-head attention [C/OL]//The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, New York, NY, USA, February 7-12, 2020. AAAI Press, 2020: 8976-8983. https://aaai.org/ojs/index.php/AAAI/article/view/6429.

[127] Wu X, Feng Y, Shao C. Generating diverse translation from model distribution with dropout [C/OL]//Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP). Online: Association for Computational Linguistics, 2020: 1088-1097. https://aclanthology.org/2020.emnlp-main.82. DOI: 10.18653/v1/2020.emnlp-main.82.

[128] Li J, Gao P, Wu X, et al. Mixup decoding for diverse machine translation [C/OL]//Findings of the Association for Computational Linguistics: EMNLP 2021. Punta Cana, Dominican Republic: Association for Computational Linguistics, 2021: 312-320. https://aclanthology.org/2021.findings-emnlp.29. DOI: 10.18653/v1/2021.findings-emnlp.29.

[129] Sennrich R, Haddow B, Birch A. Improving neural machine translation models with monolingual data [C/OL]//Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). 2016: 86-96. https://www.aclweb.org/anthology/P16-1009.

[130] Zhang J, Zong C. Exploiting source-side monolingual data in neural machine translation [C/OL]//Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. 2016: 1535-1545. https://www.aclweb.org/anthology/D16-1160.

[131] Zhou J, Keung P. Improving non-autoregressive neural machine translation with monolingual data [C/OL]//Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. Online: Association for Computational Linguistics, 2020: 1893-1898. https://www.aclweb.org/anthology/2020.acl-main.171. DOI: 10.18653/v1/2020.acl-main.171.

[132] Nguyen X, Joty S R, Wu K, et al. Data diversification: A simple strategy for neural machine translation [C/OL]//Larochelle H, Ranzato M, Hadsell R, et al. Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual. 2020. https://proceedings.neurips.cc/paper/2020/hash/7221e5c8ec6b08ef6d3f9ff3ce6eb1d1-Abstract.html.

[133] Freitag M, Al-Onaizan Y, Sankaran B. Ensemble distillation for neural machine translation [Z]. 2017.

[134] Gu J, Cho K, Li V O. Trainable greedy decoding for neural machine translation [C/OL]// Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing. Copenhagen, Denmark: Association for Computational Linguistics, 2017: 1968-1978. https://www.aclweb.org/anthology/D17-1210. DOI: 10.18653/v1/D17-1210.

[135] Tu L, Pang R Y, Wiseman S, et al. ENGINE: Energy-based inference networks for non-autoregressive machine translation [C/OL]//Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. Online: Association for Computational Linguistics, 2020: 2819-2826. https://www.aclweb.org/anthology/2020.acl-main.251. DOI: 10.18653/v1/2020.acl-main.251.

[136] Banerjee S, Lavie A. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments [C/OL]//Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization. Ann Arbor, Michigan: Association for Computational Linguistics, 2005: 65-72. https://aclanthology.org/W05-0909.

[137] Zhang* T, Kishore* V, Wu* F, et al. Bertscore: Evaluating text generation with bert [C/OL]// International Conference on Learning Representations. 2020. https://openreview.net/forum?id=SkeHuCVFDr.

致 谢

我在 2017 年底来到实验室，到如今即将完成博士学业，五年多的时间一晃而过。在接近博士生涯终点的此刻，我想向所有给予过我支持和帮助的人们表达我最诚挚的谢意。

首先，我要感谢我的导师冯洋老师。冯老师是我科研道路上的引路人，带我领略了自然语言处理领域的风景。在初入师门时，冯老师对我的指导无微不至，时常会与我在生态舱展开一个多小时的讨论，使我能从一个科研小白快速成长。冯老师对我的悉心指导和严格要求贯彻了我五年多的科研生涯，激励我克服困难，不断前行。同时，冯老师在学术和生活上都给了我最大程度的包容，使我能够自由地探索感兴趣的方向，感受到了科研带来的乐趣。

感谢实验室的各位老师。赵红梅老师待人亲切温和，对我们关怀备至，令人如沐春风。刘琳老师和程一老师工作认真负责，将实验室的事务处理地井井有条，是实验室坚固的后盾。李秀星老师就像一位见多识广的师兄，时常向我们传授科研和人生道路上的经验，祝愿他在未来的学术道路上一片坦途。

感谢实验室的师兄师姐们，谢军、李响、孟凡东、孙萌、张文、张金超、张源、丁春发、胡镓伟、薛海洋、杨郑鑫、谷舒豪、王树根师兄和马青松、刘舒曼、李京谕、申磊、欧蛟师姐。在我刚进入实验室时，张文师兄、薛海洋师兄、马青松师姐带我组建了‘‘吃饭小分队’’，帮助我快速融入到了实验室的大家庭中。张金超师兄与孟凡东师兄的专业功底十分深厚，在我科研入门阶段给予了细致的指导和帮助，一起熬夜修改论文的经历仍历历在目。在百度实习的一年里，孙萌师兄在工作和生活上都提供了诸多支持，帮助我快速适应了陌生的环境。在我找工作的一年中，谢军师兄与孟凡东师兄给予了我许多的建议和支持，希望我未来也能将这份帮助传承下去。谷舒豪师兄作为实验室目前的大师兄，经常组织各种活动来提升实验室的凝聚力，也使我们的精神能在科研之余得到休整。感谢所有师兄师姐在我刚进入实验室时给予我的耐心指导和无私帮助。

感谢我的同窗好友单勇陪我度过了博士生涯的前三年，感谢我的舍友李绩成、郭登级陪我度过了在 F136 的一段快乐时光。感谢非自回归小组的小伙伴们，有幸能与你们探索同一方向，每次小组讨论都能碰撞出思想的火花，使科研变得充满了趣味。感谢我的师弟师妹们，李泽康、张倬诚、张绍磊、伍烜甫、黄浪林、房庆凯、马铮睿、刘龙祥、桂尚彤、郭守涛、张珂豪、鄢子文、杨哲、卜梦煜、周奘、雨田师弟和赵彤钰师妹，还有实验室的实习生谢婉莹、张良、赵紫毫、高博飞、庞征锋、董宁、郭雯钰、田畅。在我成为你们的师兄之后，我有幸见证了你们的成长和进步。你们的热情和努力，也让我感受到了实验室团队的凝聚力和活力。

感谢我的父母，感谢你们一直以来对我的包容和支持，让人在他乡的我也能感受到家的温暖和坚实的后盾。

感谢我的另一半卫李赋凌，你是我生命中的天使，感谢能遇见你 ♡。

作者简历及攻读学位期间发表的学术论文与其他相关学术成果

基本信息

姓名：邵晨泽　性别：男　出生日期：1996.04.29　籍贯：浙江温州

作者简历

2014 年 9 月——2018 年 6 月，在中国科学院大学计算机科学与技术学院获得学士学位。

2018 年 9 月——2023 年 6 月，在中国科学院计算技术研究所攻读博士学位。

攻读博士学位期间发表的论文

(1) Chenze Shao, Yang Feng, Jinchao Zhang, Fandong Meng, Jie Zhou. Sequence-level training for non-autoregressive neural machine translation [J/OL]. Computational Linguistics, 2021, 47(4): 891-925.

(2) Chenze Shao, Jinchao Zhang, Jie Zhou, Yang Feng. Rephrasing the reference for non-autoregressive machine translation [C]//The Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI 2023, Washington, DC, USA, February 7-14, 2023. AAAI Press, 2023.

(3) Chenze Shao, Yang Feng. Non-monotonic latent alignments for ctc-based non-autoregressive machine translation [C/OL]//Koyejo S, Mohamed S, Agarwal A, et al. Advances in Neural Information Processing Systems: volume 35. Curran Associates, Inc., 2022: 8159-8173.

(4) Chenze Shao, Zhengrui Ma, Yang Feng. Viterbi decoding of directed acyclic transformer for non-autoregressive machine translation [C/OL]//Findings of the Association for Computational Linguistics: EMNLP 2022. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, 2022: 4390-4397.

(5) Chenze Shao, Xuanfu Wu, Yang Feng. One reference is not enough: Diverse distillation with reference selection for non-autoregressive translation [C/OL]//Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Seattle, United States: Association for Computational Linguistics, 2022: 3779-3791.

(6) Chenze Shao, Yang Feng. Overcoming catastrophic forgetting beyond continual learning: Balanced training for neural machine translation [C/OL]//Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Dublin, Ireland: Association for Computational Linguistics, 2022: 2023-2036.

(7) Chenze Shao, Jinchao Zhang, Yang Feng, Fandong Meng, Jie Zhou. Minimizing the bag-of-ngrams difference for non-autoregressive neural machine translation [J/OL]. Proceedings of the AAAI Conference on Artificial Intelligence, 2020, 34(01): 198-205.

(8) Chenze Shao, Yang Feng, Jinchao Zhang, Fandong Meng, Xilin Chen, Jie Zhou. Retrieving sequential information for non-autoregressive neural machine translation [C/OL]//Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. Florence, Italy: Association for Computational Linguistics, 2019: 3013-3024.

(9) Chenze Shao, Yang Feng, Xilin Chen. Greedy search with probabilistic n-gram matching for neural machine translation [C/OL]//Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. Brussels, Belgium: Association for Computational Linguistics, 2018: 4778-4784.

(10) 冯洋, 邵晨泽。神经机器翻译前沿综述. 中文信息学报. 2020, 34(7): 1-18

(11) Zhengrui Ma, Chenze Shao, Shangtong Gui, Min Zhang, Yang Feng. Fuzzy alignments in directed acyclic graph for non-autoregressive machine translation [C/OL]//International Conference on Learning Representations. 2023.

(12) Yong Shan, Yang Feng, Chenze Shao. Modeling coverage for non-autoregressive neural machine translation [C/OL]//2021 International Joint Conference on Neural Networks (IJCNN). 2021: 1-8.

(13) Yang Feng, Shuhao Gu, Dengji Guo, Zhengxin Yang, Chenze Shao. Guiding teacher forcing with seer forcing for neural machine translation [C/OL]//Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers). Online: Association for Computational Linguistics, 2021: 2862-2872.

(14) Xuanfu Wu, Yang Feng, Chenze Shao. Generating diverse translation from model distribution with dropout [C/OL]//Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP). Online: Association for Computational Linguistics, 2020: 1088-1097.

(15) Yang Feng, Wanying Xie, Shuhao Gu, Chenze Shao, Wen Zhang, Zhengxin Yang, Dong Yu. Modeling fluency and faithfulness for diverse neural machine translation [J/OL]. Proceedings of the AAAI Conference on Artificial Intelligence, 2020, 34(01): 59-66.

攻读博士学位期间的获奖情况

(1) CWMT2018 第十四届全国机器翻译评测-多语言翻译，获得第三名。

(2) CCMT2019 第十五届全国机器翻译评测-藏汉翻译，获得第一名。

(3) 2019 年计算所一等博士学业奖学金
(4) 2019 年计算所虑得投资博士生奖
(5) 2021 年中国科学院大学校级三好学生
(6) 2021 年计算所一等博士学业奖学金
(7) 2022 年计算所一等博士学业奖学金

攻读博士学位期间申请的发明专利

(1) 名称：翻译方法、机器翻译模型的训练方法、装置及存储介质，作者：邵晨泽、张金超、孟凡东、冯洋、周杰，专利号：CN110442878A
(2) 名称：基于机器翻译模型的翻译方法、装置、设备和存储介质，作者：邵晨泽、张金超、孟凡东、冯洋、周杰，专利号：CN110457713A
(3) 名称：模型训练方法、装置、计算机可读存储介质和计算机设备，作者：邵晨泽、张金超、孟凡东、周杰、冯洋，专利号：CN110263349A
(4) 名称：语料评估模型训练方法、装置、存储介质和计算机设备，作者：邵晨泽、张金超、孟凡东、周杰、冯洋，专利号：CN110263350A

攻读博士学位期间参加的科研项目

(1) 2019 年 01 月—2019 年 12 月：基于强化学习的神经机器翻译研究（国家自然科学基金面上项目，课题编号：61876174）
(2) 2019 年 08 月—2022 年 07 月：基于神经网络的汉泰机器翻译研究（国家重点研发计划政府间国际科技创新合作重点专项，课题编号：2017YFE0192900）