分类号 密级

UDC

学 位 论 文

面向神经机器翻译的模型训练

和解码优化方法研究

|  |  |  |  |
| --- | --- | --- | --- |
| 作者姓名： | 林野 | | |
| 指导教师： | 肖桐 副教授 | | |
|  | 东北大学计算机科学与工程学院 | | |
| 申请学位级别： | 硕士 | 学科类别： | 工学 |
| 学科专业名称： | 计算机应用技术 | | |
| 论文提交日期： | 2019年7月 | 论文答辩日期： | 2019年7月 |
| 学位授予日期： | 2019年7月 | 答辩委员会席： |  |
| 评阅人： |  | | |

东 北 大 学

2019年7月

##### A Thesis in Computer Application Technology

**Research of Model Training and Decoding Optimization Methods for Neural Machine Translation**

By Lin Ye

Supervisor: Associate Professor Xiao Tong

**Northeastern University**

**July 2019**

独创性声明

本人声明，所呈交的学位论文是在导师的指导下完成的。论文中取得的研究成果除加以标注和致谢的地方外，不包含其他人己经发表或撰写过的研究成果，也不包括本人为获得其他学位而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均己在论文中作了明确的说明并表示谢意。

学位论文作者签名：

日 期：

学位论文版权使用授权书

本学位论文作者和指导教师完全了解东北大学有关保留、使用学位论文的规定：即学校有权保留并向国家有关部门或机构送交论文的复印件和磁盘，允许论文被查阅和借阅。本人同意东北大学可以将学位论文的全部或部分内容编入有关数据库进行检索、交流。

作者和导师同意网上交流的时间为作者获得学位后：

半年 □ 一年□ 一年半□ 两年□

学位论文作者签名： 导师签名：

签字日期： 签字日期：

摘 要

随着相关技术的发展，机器翻译系统的性能已有显著的提升。相较于传统的统计机器翻译，神经机器翻译在模型性能和翻译流畅性等方面都有了明显的进步，但随着训练数据的增加以及模型规模的增大，神经机器翻译模型的计算代价和存储代价也在不断的增加，这严重阻碍了神经机器翻译的发展。为了解决以上问题，本文分别从模型训练和解码两个角度对神经机器翻译模型进行优化，旨在不影响模型性能的同时进行模型加速和模型压缩，达到降低模型存储大小和计算时间的目的，本文主要工作如下：

（1）模型训练优化。为了解决神经机器翻译模型训练代价大的问题，本文从模型训练优化问题入手，引入低精度运算方法来进行模型的训练加速，并提出一种分组卷积与自注意力机制结合的神经机器翻译模型，首次将分组卷积结构引入到机器翻译任务中。为了验证以上方法的有效性，本文在16万的德英方向IWSLT数据和180万的中英方向LDC两个数据集上分别进行实验，来验证在不同语料规模和不同模型规模上实验方法的有效性。使用低精度运算方法在神经机器翻译任务上最大可以达到58.70%的训练加速效果，使用分组卷积结构可以在降低计算复杂度的同时达到1.46倍的模型压缩率。

（2）模型解码优化。为了解决神经机器翻译模型解码时的资源受限问题，本文将模型存储压缩方法以及各种方法的组合方法应用在神经机器翻译任务中，为了探究不同模型结构对压缩效果的影响，我们分别在基于循环神经网络的机器翻译模型和基于自注意力机制的神经机器翻译模型上进行实验。为了缩减模型解码时间，我们提出了一种动态的Beam Search解码方法，可以根据不同句子长度动态选取beam size的大小来缩减解码搜索过程。使用模型存储压缩方法，在基于循环神经网络的机器翻译模型上可以达到11.7倍的压缩率，在基于自注意力机制的神经机器翻译模型上可以达到5.8倍的压缩率，应用动态Beam Search解码方法在不影响解码性能的前提下可以提升11.76%的解码速度。

**关键词：**神经机器翻译；自注意力机制；模型加速；模型压缩

Abstract

With the development of related technology, the performance of machine translation system has been improved significantly. Compared with the traditional statistical machine translation, neural machine translation has improved significantly in model performance and translation fluency, but with the increase of training data and model size, the computational and storage costs of neural machine translation are also increasing, which seriously hinders the development of neural machine translation. In order to solve the above problems, this thesis optimizes the neural machine translation model from the perspective of model training and decoding, aiming at model acceleration and model compression without affecting the performance. The main work of this thesis is as follows:

(1) Model training optimization. In order to solve the problem of high training cost of neural machine translation model, the low-precision training method is introduced to accelerate the training of model. To accelerate the model training process, this thesis proposes a new neural machine translation architecture, which introduces group convolution structure into the task of machine translation for the first time. To verify the effectiveness of the proposed method, experiments are carried out on 160,000 German-English IWSLT datasets and 1.8 million Chinese-English LDC datasets to explore the effectiveness of the proposed method in different corpus sizes and different model scales. It founds that using low-precision training method can achieve 58.70% training acceleration in neural machine translation task. Using group convolution structure can reduce the computational complexity and achieve 1.46× model compression rate at the same time.

(2) Model decoding optimization. In order to solve the problem of resource constraints in neural machine translation decoding, the model storage compression methods are applied to the neural machine translation task in this thesis. In order to explore the impact of different model structures, experiments are carried out on the model of neural machine translation based on recurrent neural network and self-attention mechanism respectively. In order to reduce the decoding time of the model, we propose a dynamic Beam Search decoding method, which can dynamically select the size of beam size according to the length of different sentences. By using the model storage compression method, the compression ratio can reach 11.7× on recurrent neural machine translation and 5.8× on the self-attention based neural machine translation. The decoding speed can be improved by 11.76% without affecting the decoding performance by using the dynamic Beam Search decoding method.

**Key words:** neural machine translation; self-attention; model acceleration; model compression

目 录

[独创性声明 I](#_Toc12970476)

[摘 要 II](#_Toc12970477)

[Abstract III](#_Toc12970478)

[第1章 绪 论 1](#_Toc12970479)

[1.1 研究背景 1](#_Toc12970480)

[1.2 研究内容 2](#_Toc12970481)

[1.3 论文组织结构 3](#_Toc12970482)

[第2章 相关技术概述 5](#_Toc12970483)

[2.1 神经机器翻译框架 5](#_Toc12970484)

[2.1.1编码解码框架 6](#_Toc12970485)

[2.1.2 注意力机制 7](#_Toc12970486)

[2.2 神经机器翻译模型结构 10](#_Toc12970487)

[2.3 模型训练和解码优化问题概述 14](#_Toc12970488)

[2.3.1模型训练优化问题 15](#_Toc12970489)

[2.3.2模型解码优化问题 16](#_Toc12970490)

[2.4 本章小结 17](#_Toc12970491)

[第3章 神经机器翻译模型训练优化方法 19](#_Toc12970492)

[3.1 研究动机 19](#_Toc12970493)

[3.2 方法描述 21](#_Toc12970494)

[3.2.1基于低精度运算的神经机器翻译模型加速方法 21](#_Toc12970495)

[3.2.2基于分组卷积的神经机器翻译模型加速方法 24](#_Toc12970496)

[3.3 实验 27](#_Toc12970497)

[3.3.1 实验设置 27](#_Toc12970498)

[3.3.2 实验结果 29](#_Toc12970499)

[3.3.3 结果分析 33](#_Toc12970500)

[3.4 本章小节 34](#_Toc12970501)

[第4章 神经机器翻译模型解码优化方法 41](#_Toc12970502)

[4.1 研究动机 41](#_Toc12970503)

[4.2 方法描述 42](#_Toc12970504)

[4.2.1 神经机器翻译模型存储压缩方法 42](#_Toc12970505)

[4.2.2 动态Beam Search解码方法 44](#_Toc12970506)

[4.3 实验 46](#_Toc12970507)

[4.3.1 实验设置 46](#_Toc12970508)

[4.3.2 实验结果 47](#_Toc12970509)

[4.3.3 结果分析 50](#_Toc12970510)

[4.4 本章小结 51](#_Toc12970511)

[第5章 总结与展望 53](#_Toc12970512)

[5.1 工作总结 53](#_Toc12970513)

[5.2 创新点分析 54](#_Toc12970514)

[5.3 未来工作 54](#_Toc12970515)

[5.3.1 低精度神经网络反向传播方法 55](#_Toc12970516)

[5.3.2 更低精度神经机器翻译模型训练方法 55](#_Toc12970517)

[参考文献 57](#_Toc12970518)

[致谢 63](#_Toc12970519)

[硕士期间参加的科研项目 65](#_Toc12970520)

[硕士期间取得的学术成果 67](#_Toc12970521)

第1章 绪 论

1.1 研究背景

在当今世界中信息呈爆炸式增长，文本中隐藏着大量有价值的知识，靠人力无法完成这些文本的翻译工作，这就形成了语言障碍，这种障碍阻碍了人与人之间的沟通交流。传统的人工翻译虽然翻译准确性较高，但是效率低下且人工成本较高，却未从根本上解决语言障碍这一问题。机器翻译相比于传统的人工翻译，机器翻译的效率更高、成本较低。机器翻译技术的发展为不同国家和民族之间信息传递带来了十分重要的正向作用，对促进民族团结和民族间文化交流有着重要意义。

自20 世纪40 年代提出至今，机器翻译理论发展已经历经了近70 年。在此期间先后提出了各种机器翻译技术，主要包括：基于规则的机器翻译[1]-[2][3]、基于统计的机器翻译[4]-[5][6][7][4][6]、基于神经网络的机器翻译[8]-[9][10]。近几年，随着深度学习的快速崛起，基于神经网络的机器翻译方法获得了飞速发展，各种各样的神经机器翻译也层出不穷，例如基于循环神经网络的神经机器翻译[11]，基于卷积神经网络的神经机器翻译[12]，以及基于自注意力机制的神经机器翻译[13]等等。

现代计算机的速度非常快，其速度甚至可以达到人脑中神经元速度的上百万倍，现代计算机的发展使人脑的能力在数值运算和逻辑运算方面更加高效。计算机的发展在控制和信息的处理等各方面为人们提供了更多先进的实现智能和自动化的手段，使人们对生物脑神经有了更具现化的了解，随着对生物脑的深入了解，人工神经网络也获得了长足的发展[14]。将人工神经网络应用于自然语言处理领域里，使得很多自然语言处理相关的任务性能均得到了显著的提高，其中就包括神经机器翻译任务[15]，机器翻译的发展为当今人们的出行和交流带来了巨大的便捷。

神经机器翻译（Neural Machine Translation，NMT）相较于传统的统计机器翻译是一种简单的新架构，用于将文本从一种语言翻译为另一种语言，在过去的几年中，神经机器翻译在许多翻译任务中取得了良好的翻译性能[16]。深层神经网络如此强大的关键性原因包括数据集规模的快速增长、网络架构的日益深入以及训练深层神经网络中各种技术的飞速发展。虽然更大更深规模的网络给当前神经网络的发展带来了巨大的进步，但随着网络规模的扩大，神经网络会变得更冗余，这种冗余的神经网络无疑是有害的，网络中的冗余参数会严重浪费模型的计算和存储资源[17]。为了使资源得到更加充分合理的利用，研究人员开始把模型加速和模型压缩作为神经网络的一项研究重点[18]。

目前神经网络模型加速和模型压缩研究工作在计算机视觉领域上已经取得了不错的进展[19]-[20][21]，但是针对神经机器翻译的模型加速和压缩的研究工作正处于起步阶段，其中的许多方法仍有待探索和挖掘。本文分别从训练和解码两个角度对神经机器翻译模型进行优化，目的是加快模型训练速度并降低模型所需存储。

1.2 研究内容

传统的基于循环神经网络的机器翻译模型由于其输入序列的顺序存在限制，必须按照时序顺序进行输入，因此模型的训练和解码都非常耗时[22]。随着神经网络的发展，基于自注意力机制的神经机器翻译克服了循环神经网络的输入顺序限制，并凭借其更近的信息传递距离在众多网络结构中脱颖而出，在机器翻译及计算机视觉等多项任务上取得了世界领先水平的性能[13]。通过注意力机制，词与词之间的关联程度可以得到更加充分的考虑，模型可以获得更准确的语义信息，模型的性能也得到了极大的提升。

虽然基于自注意力机制的神经网络极大的提升了神经网络的并行能力和计算速度，但其仍属于深层的神经网络，参数量十分巨大。并且由于注意力机制需要频繁的进行词与词之间关联程度的计算和注意力对齐的操作，导致模型的训练效率和解码效率仍然十分低下[23]，在计算资源受限和需要实时响应的环境下仍然存在着部署难题。针对该问题，本文分别从模型训练和模型解码两个角度对基于自注意力机制的神经机器翻译模型进行优化，在不影响模型性能或者模型性能损失较小的前提下对神经机器翻译模型的训练速度和解码速度进行提升，减小模型的训练和解码代价并降低模型所需的存储，达到计算资源、存储资源和模型性能上的平衡。本文主要的工作如下：

1. 神经机器翻译模型训练优化方法：在进行神经机器翻译模型训练优化方法研究的过程中，本文从两个角度来考虑，一是计算角度、二是模型结构角度。由于传统的神经网络基于单精度或双精度浮点运算，这两种浮点运算的计算效率低下且优化困难。本文采取一种基于半精度张量计算的低精度神经网络计算方法，将原本网络中的单精度浮点和双精度浮点用半精度浮点数来代替，可以有效提升神经网络的计算效率，在保证模型原有性能的前提下极大提升了模型的训练速度。从模型结构优化的角度来考虑，本文提出一种分组卷积网络和自注意力机制相结合的神经机器翻译模型，首次在神经机器翻译模型中引入一种分组卷积结构来代替传统的前馈神经网络，在保证模型性能的同时，显著的提升了模型的计算效率并减少了模型所需参数量，使模型结构更加科学有效。
2. 神经机器翻译模型解码优化方法：在解码阶段，我们主要关心两个方面：一是解码模型存储大小、二是模型的解码效率。由于传统的神经网络模型结构存在冗余，一些模型中的子结构在整个网络的计算过程中具有频繁的操作且这些子结构中参数规模较大，这些参数大多数是无意义甚至是冗余的，对硬件上的存储资源都造成了极大的浪费。本文针对神经机器翻译任务，分别从模型存储大小和解码效率两个角度对模型解码过程进行优化。前者使用模型裁剪和参数低精度化方法，在基于循环神经网络和自注意力机制的神经机器翻译模型上分别进行组合实验，探索不同模型结构下神经机器翻译模型可达到的最大的压缩比例。后者采取动态的Beam Search解码方法，在Beam Search的搜索过程中根据所解句子长度动态的改变参数beam size的大小，以加快模型解码速度。

1.3 论文组织结构

本文主要研究了神经机器翻译模型训练和解码优化方法，前者基于低精度神经网络方法和分组卷积结构来提升模型的训练效率，在提升模型计算效率的同时使模型结构更加科学合理；后者通过模型存储压缩方法和动态Beam Search解码方法减少模型存储并提升模型的解码效率。

本文主要内容如下：

第一章绪论主要介绍了论文的研究背景和本文的研究内容。

第二章相关技术概述主要介绍了论文研究过程中使用到的相关联的技术，主要包括神经机器翻译中广泛使用的相关技术以及模型训练优化和解码优化问题。其中，详细介绍了神经机器翻译的框架，包括传统的编码解码框架以及最近比较流行的注意力机制。随后介绍了几种常用的神经机器翻译模型的结构，包括基于循环神经网络的机器翻译模型、基于卷积神经网络的机器翻译模型和最近流行的基于自注意力机制的神经机器翻译模型。最后简要对模型训练和解码优化问题进行了概述，介绍了本文的研究动机。

第三章主要介绍了神经机器翻译模型训练优化方法，包括基于低精度运算的神经机器翻译模型加速方法和基于分组卷积的神经机器翻译模型加速方法，分别从计算角度和模型结构角度对模型训练过程进行优化，前者采取半精度浮点和单精度浮点混合计算的模型训练方法，后者用分组卷积结构来替代传统的前馈神经网络，在提升模型运算效率的同时显著减少了模型所需参数量。与此同时，本章还详细介绍了这两种方法的研究动机、实验结果分析和方法中存在的一些具体问题。

第四章主要介绍了神经机器翻译模型解码优化方法，包括模型存储压缩方法和动态Beam Search解码方法，前者通过模型裁剪和参数低精度化方法在不损失模型原有性能的前提下分别在基于自注意力机制的神经机器翻译模型和基于循环神经网络的机器翻译模型达到5.8×和11.7×的压缩率，后者在解码的Beam Search搜索过程中根据句子长度动态的改变参数beam size大小以提升模型的解码效率。与此同时，本章还详细介绍了这两种方法的研究动机、实验结果分析以及所采取方法的优缺点。

第五章对本文的研究工作进行了总结，对本文的创新点进行了分析，同时对本文的未来工作进行了讨论和展望。

第2章 相关技术概述

2.1 神经机器翻译框架

机器翻译发展至今大致可以分为三个阶段：1）基于规则的方法；2）基于统计的数据驱动的方法；3）基于深度学习的神经网络端到端方法，在不同阶段均取得了重要的成果及应用。

传统的基于规则的机器翻译虽然在短时间内满足了机器翻译的需要，但随着研究的深入，一系列问题也逐渐的暴露出来。例如，人工规则的覆盖度非常有限，规则与语种强相关，必须为每种语言自定义词典及规则，并且随着规则数量增多由人工规则导致的冲突也会增多。

为了解决以上问题，20世纪90年代初IBM和AT&T等机构首先提出了统计机器翻译( Statistical Machine Translation，SMT)的思想，思想可以概括为公式(2.1)，其中*s*和 *t*分别代表源文和译文，表示把s翻译成t的概率，𝑎𝑟𝑔𝑚𝑎𝑥表示从所有译文中搜索最佳译文的过程，这个问题的实质就是如何由源语言*s*求得对应的最大可能的目标语言*t*。

(2.1)

统计机器翻译相较于基于规则的机器翻译完全抛弃了对人工规则的依赖，统计机器翻译把翻译问题建模为搜索最大概率译文问题，显著加强了系统的鲁棒性和可扩展性。谷歌翻译也于2005年正式推出基于统计机器翻译的多语种在线机器翻译引擎，极大的便利了人们的生活。但统计机器翻译也有较明显的缺点,统计机器翻译过度依赖翻译语料，如果待翻译的句子在训练语料中没有相似的资料文本，得出的译文质量很差，而且由于在统计机器翻译过程中得不到上下文信息，译文质量无法得到保证。

为了进一步提升模型的表示能力，从2013年开始，研究人员开始尝试一种全新的机器翻译方法，即基于神经网络的机器翻译，这种基于神经网络的机器翻译方法采取了神经元网络直接对翻译问题进行建模，完全抛弃了人工书写规则以及设计翻译特征的翻译模式，整个翻译过程都在一种端到端的模型框架下完成，神经机器翻译训练的是一个序列到序列映射的神经网络，这个网络的输出可以是一个变长序列。神经机器翻译相较于传统的统计机器翻译是一种简单的新架构，用于将文本从一种语言翻译为另一种语言，神经机器翻译现如今已取得了显著的性能，显著的提高了机器翻译的流畅行和充分性。基于神经网络的机器翻译方法成功的将研究人员的研究重点转移到了对神经网络结构的设计，也就是科学合理的设计神经机器翻译的网络框架，对于模型性能来说，一个好的网络框架至关重要。

2.1.1编码解码框架

神经网络作为近些年来新兴起来的机器翻译模型，起初在包括图像识别、图像分类等计算机视觉领域取得了十分优异的性能和较快速的研究进展。但由于计算机视觉领域中的任务大多是输入输出维度固定的任务，而在机器翻译任务中，我们无法对输入输出维度预先进行判断，因此神经网络的应用并不能简单的从计算机视觉任务迁移到机器翻译任务中。面对这种变长维度的序列转换，如何合理的建立起一个合理的映射是一个十分值得研究的问题。

为了解决变长序列的映射问题，研究人员提出了编码-解码框架[9]。与统计机器翻译所考虑的思想完全不同，神经机器翻译是将机器翻译任务看作是序列到序列的转化问题，该问题其实是一个两阶段建模问题，一个是编码阶段，一个是解码阶段。编码阶段通过抽象输入的语义信息将单词序列进行编码，将语义信息表示为一个固定长度的实数向量。解码阶段将包含上述语义信息的向量进行解码以生成目标译文。

在编码解码框架中，翻译过程被看作从源语言序列转换成目标语言序列的过程，图2.1给出了一个使用循环神经网络来实现中文-英文翻译的例子。

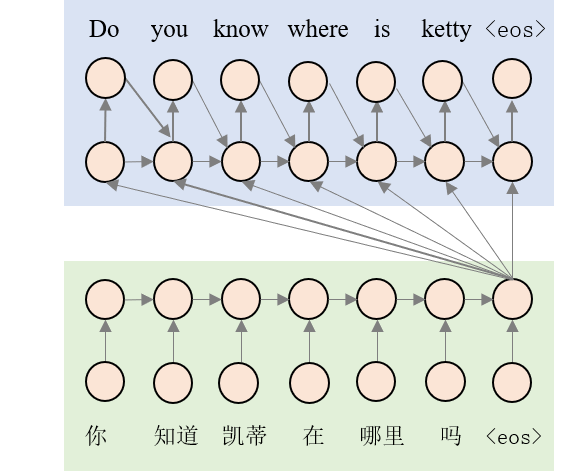


图2.1 编码器解码器框架

Fig. 2.1 Encoder-Decoder framework

其中<eos>代表结束标志，当输入一个中文句子时，编码器首先对每一个单词都逐词编码成为一个0-1向量表示，只有当前词在词表对应索引的那一位为1，其余为0，且这个向量的维度与词表大小相同，这样的0-1向量也被称作一维词向量，最终将源语言序列编码成源语言上下文向量，解码器会采用另一个神经网络将编码好的句子反向解码生成一个英文句子。

具体来说，在编码器-解码器框架中，编码器每个时间步都通过循环单元得到一个序列片段所对应的实数向量，上一个隐藏层状态以及当前时间步的隐藏层状态，直至最后一个序列片段结束我们会得到整个输入序列的上下文语义向量，这个语义向量即我们送至解码器的输入向量。在编码器中每个时间步的隐藏层状态更新如公式(2.2)所示。

(2.2)

在解码阶段，我们需要对编码器中声称的语义向量进行解码，我们用*c*来代表这一语义向量，为上一时间步的隐藏层状态，代表上一时间步的输入单词，代表隐藏层在当前时间步的状态，则解码器中隐层状态更新如公式(2.3)所示。

(2.3)

其中，为非线性的激活函数，根据这一层的隐藏层状态，我们可以得出当前时间步对应的单词概率如公式(2.4)所示。

(2.4)

由公式(2.4)我们可以得出第t个目标端单词的概率分布，其中是一个将向量归一化的函数，之后再回到公式(2.2)，计算编码器t+1时刻的隐层状态，之后计算解码器t+1时刻的隐层状态，再计算t+1时刻对应的单词概率分布，直至得到句子结束标记<eos>为止则解码完成。

编码解码框架的出现为神经网络在机器翻译任务中的应用提供了良好的建模方法。神经机器翻译缓解了许多统计机器翻译领域中面临的困境，如：神经网络通常不需要事先人工进行定义，能够自动地从训练数据中学习到样本特征，减少了人工成本，循环神经网络中的长短时记忆模型也能够有效的捕捉长距离依赖。但由于在编码解码框架中解码的过程完全依赖于源语句子的编码结果，神经网络并没有学习到源语和目标语之间关于词汇的对应信息，使得神经翻译模型在翻译长句子时较难捕捉到源语和目标语间的关系，翻译效果较差，从而限制了神经机器翻译的发展。

2.1.2 注意力机制

在最初的编码器-解码器框架中，由于上述原因导致神经网络难以学习到源端和目标端的对应信息，翻译系统对输入较长的句子的翻译效果较差。针对这一问题，研究人员提出了基于注意力机制的神经机器翻译[13]，此模型可以动态的计算源语言的上下文，有利于神经网络学到更多的全局信息，该方法的核心思想在于通过引入源语和目标语句中词汇之间的关联程度来补充网络中的全局信息，增加网络可获得的关于文本的语义信息。

简单的说，注意力机制可以帮助神经网络在生成目标端单词时有选择性的关注不同的源语言位置，投入更多的注意力在更重要的位置，而对不太重要的位置只需投入较少的注意力，使神经网络更容易的捕获到和当前目标端单词更相关的源语信息，从而使得神经网络对目标语言的预测更为准确。在生成一个目标语单词时，基于注意力机制的神经网络会基于得到的不同的注意力对不同源语位置向量进行加权求和，得到不同的上下文向量。注意力机制的引入使得不同源语言位置对目标语单词生成的贡献度不同，使得神经网络中信息流的传递变得更为高效，有助于神经网络模型的学习。基于注意力机制的编码器-解码器框架如图2.2所示。

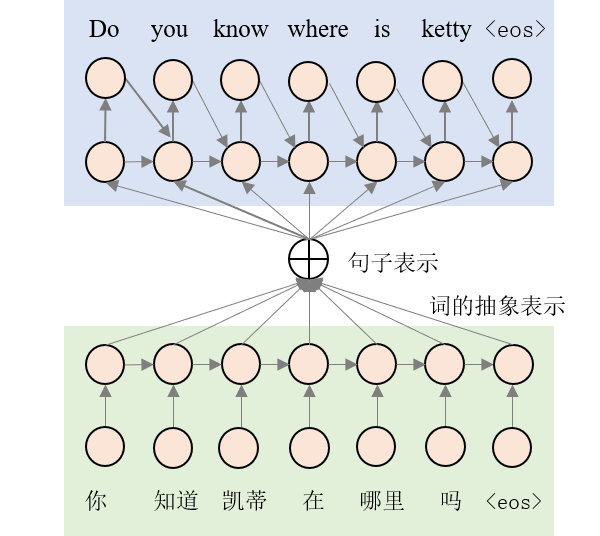


图2.2 基于注意力机制的编码器-解码器框架

Fig. 2.2 Neural machine translation model with attention mechanism

注意力机制总体上可以分为两大类：全局注意力和局部注意力[24] ，全局注意力又叫Global Attention，局部注意力又叫Local Attention。两类注意力机制的区别主要在于“注意力”是否放在所有源语位置，或者只是在少数源语位置上进行注意力对齐操作。全局注意力机制和局部注意力机制分别如图2.3左和图2.3右所示。

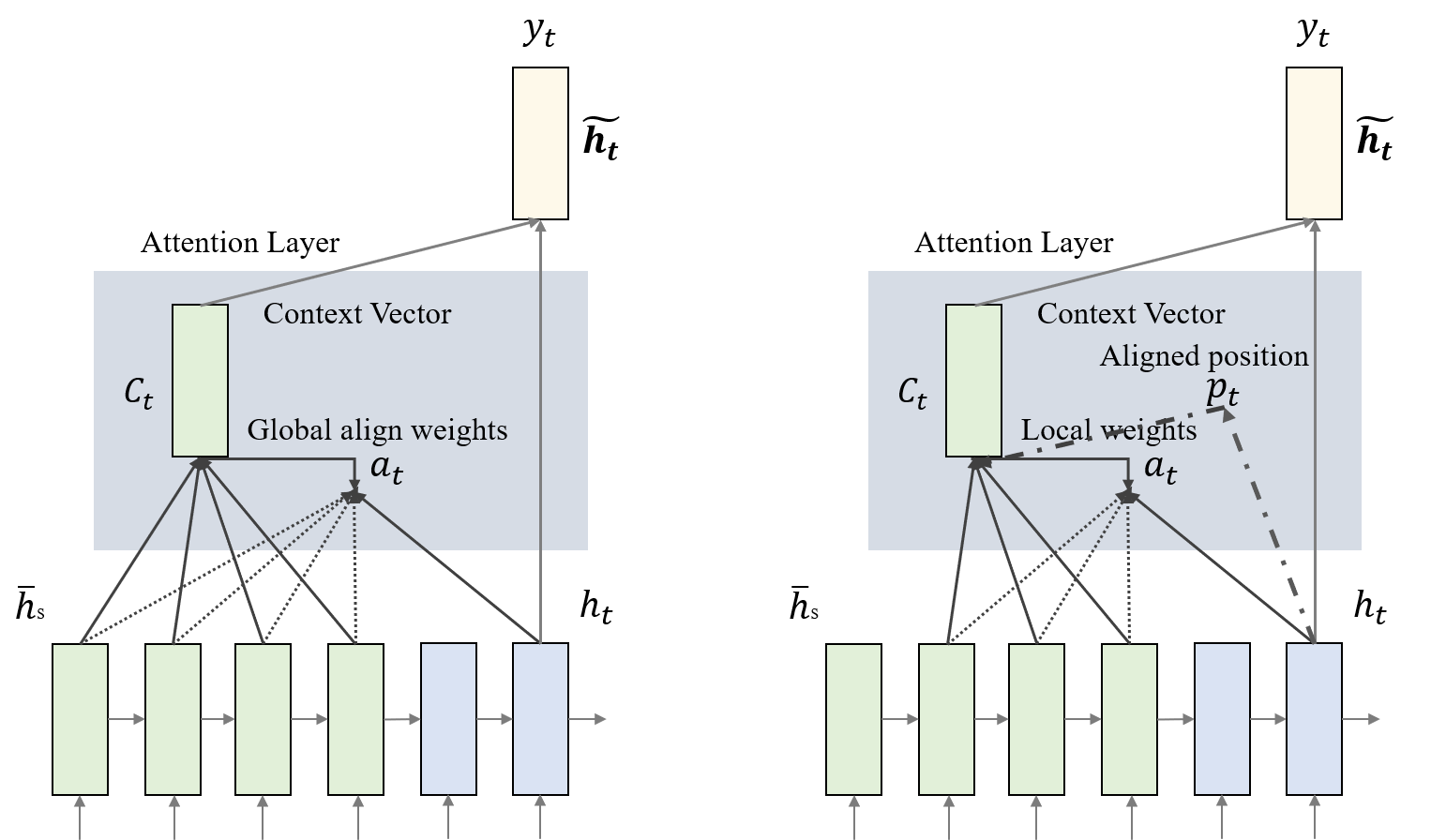


图2.3 全局注意力机制和局部注意力机制

Fig. 2.3 Global attention mechanism and Local attention mechanism

全局注意力机制的基本思想是充分考虑编码器的所有隐藏状态来计算上下文向量，通过将当前目标隐藏状态和每一个源语言隐藏层状态进行注意力权重的计算来得到可变长度的对齐向量，对齐概率的计算如公式(2.5)所示。

(2.5)

上述公式表示的是源语言第s个单词和目标语第t个单词的对齐概率，也就是我们前面所说的注意力的权重，其中，*score*是基于上下文的对齐概率打分函数，对于打分函数的计算有三种形式，分别是基于点乘的、通用的和基于连接的打分机制，如公式(2.6)所示。

(2.6)

全局注意力需要计算每个目标单词对应的所有源语单词，计算代价非常高，而且在翻译段落或者文档这种较长序列时，计算效率非常低下，计算效果也并不好。为了解决这一问题，研究人员又引入了一种局部注意力机制，对于每个目标单词，该机制只是在少数源语位置上进行注意力对齐操作。为了确定少数源语的位置范围，需要引入一个窗口机制，通过一个固定的窗口划出我们在进行目标端词汇预测所用到的词汇区间。我们用源端和目标端之间的对齐位置来确定窗口区间的范围[]，目标端和当前待预测词的对齐位置如公式(2.7)所示。

(2.7)

注意力机制为每个源语单词都生成一个包含全局信息的向量，充分考虑了不同位置单词序列间的复杂关系来得到源语或目标语中的语义信息。注意力机制的引入对于NMT的发展起到了非常重要的作用，后来很多研究人员的优秀工作均在此基础上来优化注意力机制，以此提升神经机器翻译模型的翻译性能。

2.2 神经机器翻译模型结构

神经机器翻译（Neural Machine Translation，NMT）相较于传统的统计机器翻译是一种简单的新架构，用于将文本从一种语言翻译为另一种语言，在过去的几年中，神经机器翻译在许多翻译任务中都取得了非常良好的翻译性能和很多优秀的研究成果。

神经机器翻译使用神经网络来对翻译问题进行建模，用端到端的形式从训练数据中学习模型参数。神经机器翻译的工作主要包含两个阶段：模型训练阶段和模型解码阶段。训练阶段主要是将机器翻译系统中的参数进行初始化随机，通过对大量双语平行句对的学习，利用反向梯度更新方式进行端到端的学习，然后将更新后的参数进行保存。训练完成后，输入待翻译句子，神经机器翻译模型能够通过上一步保存的模型参数进行解码，以此生成翻译译文。

相比于统计机器翻译(SMT)，神经机器翻译(NMT)训练的是从一个序列映射到另一个序列的神经网络，输出的是一个变长序列。在2.1节中介绍了神经机器翻译的基本框架是编码器-解码器框架，但基于这一框架，神经机器翻译的模型结构也在不断的发生变化，在模型结构不断的进行改革的过程中，研究人员研究的重点也有所变化，模型结构也变得越来越科学和高效。

我们从基础的神经网络中知道，神经网络主要包含输入层、隐藏层和输出层，其中通过激活函数进行输出控制，每层间均通过网络权值进行连接。基础的神经网络只在层间存在权连接，RNN与基础神经网络最大的不同之处就是在神经元间也建立了权连接[11]。基于RNN的神经机器翻译模型如图2.4示。

在当前，基于RNN的神经机器翻译系统虽然已经取得了显著成果，但是RNN中有一些传统的缺点难以克服，比如由于RNN当前时刻模型的输出依赖于上一时刻模型输出，因此很难进行并行训练，并且由于RNN结构比较复杂因此参数规模过于庞大，而且由于循环神经网络本身对输入序列的顺序是有要求的，因此并行性很差，计算效率较低下。

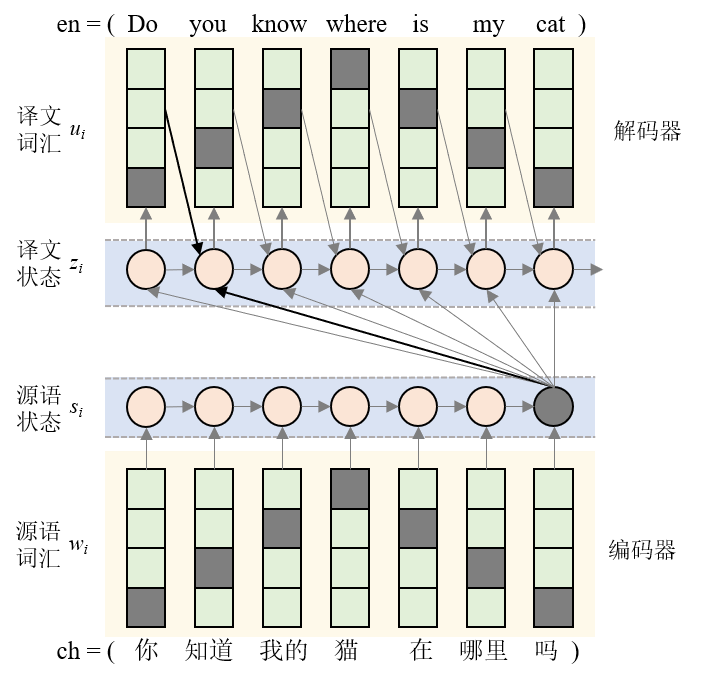


图2.4 基于循环神经网络的神经机器翻译框架

Fig. 2.4 Framework of RNN-based neural machine translation

为了解决以上问题，Facebook于2017年提出了ConvS2S[12]，这是卷积神经网络在自然语言处理领域的一次极其成功的应用，卷积神经网通过卷积窗口的移动来提取其覆盖区域的语义信息，也大大缩减了词汇间信息的传递距离，不仅减小了信息的传递距离还提升了模型的并行性。ConvS2S充分发挥了卷积神经网络高并行度的优势，在保证模型性能的前提下极大提升了模型的计算效率，基于卷积神经网络的模型结构如图2.5所示。

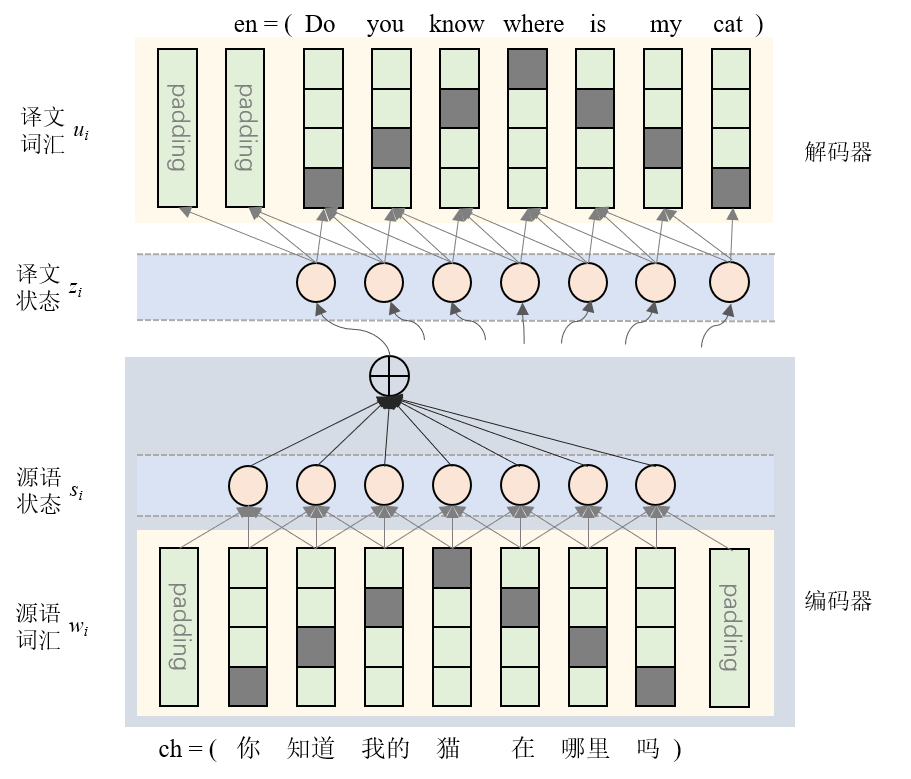


图2.5 基于卷积神经网络的神经机器翻译框架

Fig. 2.5 Framework of CNN-based neural machine translation

2017年，谷歌提出了一新框架Transformer[13]，将自注意力（self-attention）机制引入到机器翻译中来。由于此模型完全基于注意力机制，摒弃了传统的循环神经网络结构和卷积神经网络结构，进一步拉近了不同位置词汇之间的信息传输距离，有利于模型更充分地表示序列中不同位置之间的复杂关系。实验表明，这一模型在机器翻译任务上表现极好，并行度高，且训练时间更短，而且模型性能相比较RNN有较为明显的提升。

Transformer标准模型编码器由6层组成，这6层中的每个层都有两个子层，中间层分成8个头， 8个头的并行工作极大的提升了模型的并行性和模型计算效率。第二个子层是一个前馈神经网络，由于其对位置较敏感的特性，加上这一层可以进一步提升模型的表示能力。

Transformer解码器也由6层堆叠而成，与编码器中每层中两个子层不同，解码器每层有三个子层，前两个子层和编码器相同，在此不过多叙述，解码器的第三个子层即多头注意力机制，负责处理编码器的输出。无论编码器还是解码器，每一个子层后都会有残差连接和层正则化操作，以此来加速模型收敛过程，Transformer模型结构如图2.6所示。

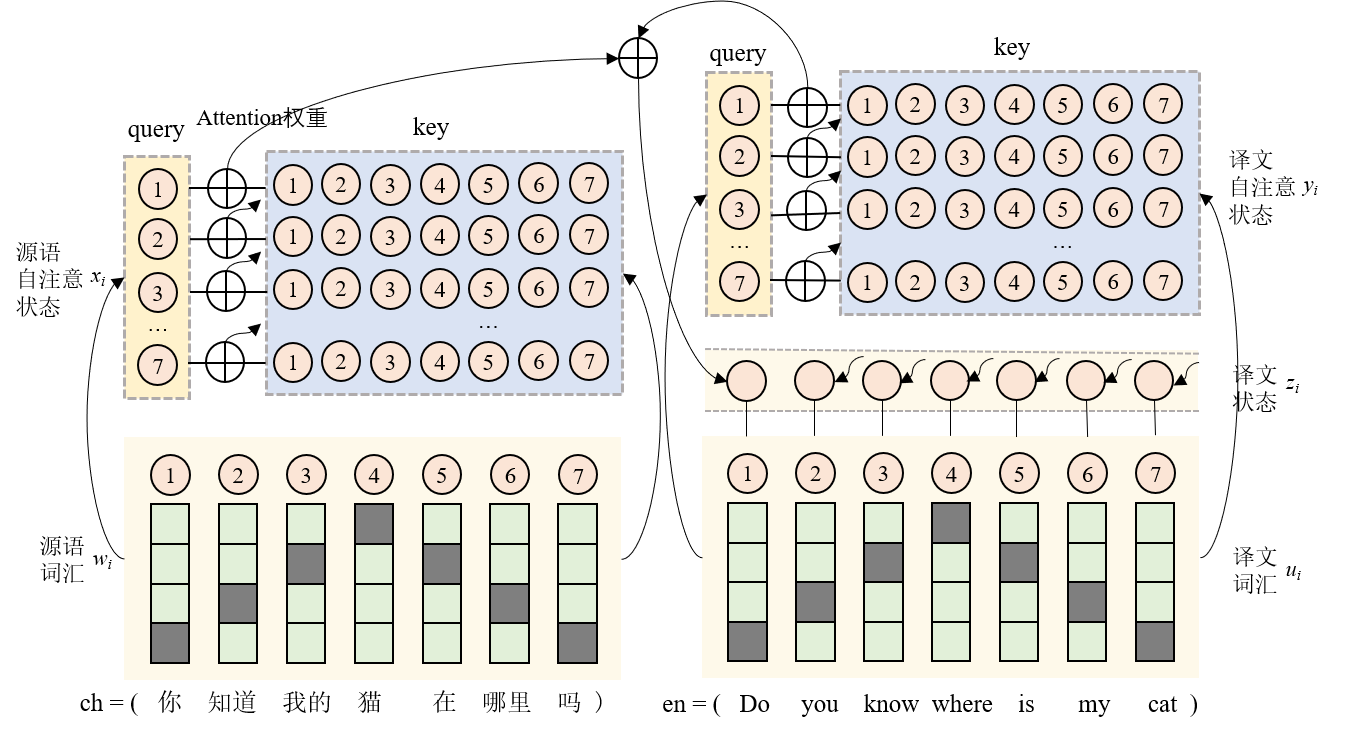


图2.6 引入注意力机制的神经机器翻译模型

Fig. 2.6 Neural machine translation model with attention mechanism

由于RNN计算顺序的限制，阻止了输入序列元素之间的并行化。Transformer和ConvS2S这些并行计算架构成功解决了RNN这一大缺点，使模型不用再按照固有顺序进行计算，并且在序列长度较长时解决了梯度消失问题。Transformer模型非常依赖自注意力机制来计算输入以及输出序列中的一系列上下文向量的空间表示，模型逐词预测输出序列时，这些空间表示可以用来预测后续单词的分布。Transformer的这种自注意力机制不仅提升了模型的并行性，而且由于每个词的表示都会与其他所有词关联起来，使整个序列都有很有效的全局感知性，而ConvS2S这种基于卷积架构的神经网络通常只有有限的感知视野，这也是Transformer模型性能优于ConvS2S的主要原因。

Transformer中的多头注意力机制主要由两部分组成：多头和自注意力，首先通过自注意力将输入通过矩阵转换变成多个维度较小的输出，之后将这些输出进行级联以得到最终的输出，这个过程如公式(2.8)所示。

(2.8)

其中是多头自注意力机制的输入矩阵，为多头自注意力机制中的n个头，在标准模型结构中n为8，为进行矩阵变换的参数矩阵，为我们所说的级联操作。所有的注意力操作都可以看作是一个到的键值对的映射，其中都是向量，注意力机制的数学表达式如公式(2.9)所示。

(2.9)

其中代表输入向量维度大小，函数是对计算结果做一个归一化的处理来进行之后的加权求和，多头注意力机制中每个头的计算过程如公式(2.10)所示。

(2.10)

多头注意力机制和普通的点乘注意力机制又叫Multi-Head Attention和Scaled Dot-Product Attention，二者结构如图2.7所示。

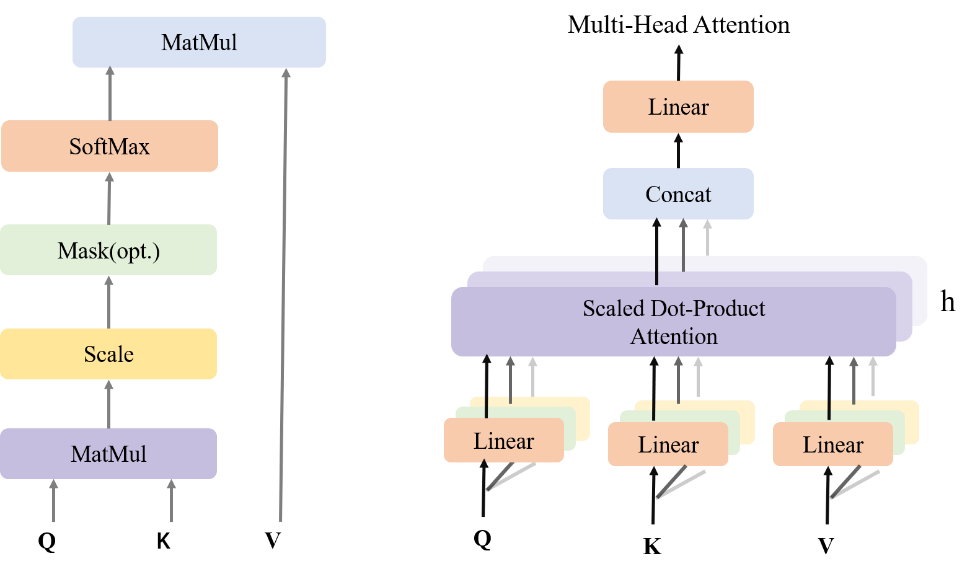


图2.7 多头注意力机制

Fig. 2.7 Multi-head attention mechanism

采用多头注意力机制后，机器翻译模型会更容易捕捉到不同位置的不同子空间信息，提升序列的全局感知性并提高模型的计算并行性，极大的提升了模型的性能和计算效率。

基于自注意力机制的神经机器翻译模型相较于之前的基于循环神经网络和基于卷积神经网络的机器翻译模型而言，模型的科学性和可解释性都得到了增强，每个子层中都包含一个自注意力子层和前馈神经网络子层，自注意力子层可以对语义信息进行更丰富的抽象，而前馈子层可以进一步增强模型的表示能力。为了进一步解释注意力机制，我们来举一个简单的例子，如图2.8所示，我们将一句话中不同注意力权重的词用不同的颜色来表示，当涉及到句子中间蓝色的词时，红色的词比绿色的词应该获得更多的注意力权重，也就是说对于蓝色的词来说，红色的词比绿色的词更为重要。

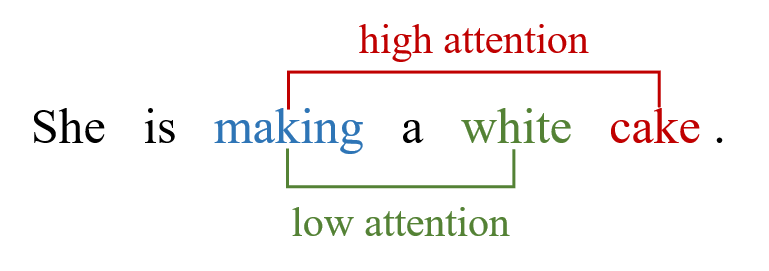


图2.8 注意力机制

Fig. 2.8 Attention mechanism

如公式(2.10)所示，注意力机制中主要包含了一个归一化的操作、一个缩放操作，还有之间的矩阵乘法，首先我们计算的是之前的矩阵乘法，这部分主要进行的是词与词之间关联程度的计算，之后这个相乘结果在经过缩放和一个归一化的操作转化为一个概率权重，这个概率权重再进一步的和相乘，得到的就是经若干不同权重加权求和所得的包含了词与词之间关联信息的新的神经网络向量，将通过此种方法计算的向量一层一层的传递下去，可以使模型获得更强的信息抽象能力和表示学习能力。虽然基于自注意力机制的神经机器翻译模型的抽象能力和表示能力相较于其他的神经机器翻译模型有了进一步的提升，但由于参数规模较大，参数计算比较繁重，对模型可使用的计算资源和存储资源造成了很大的压力。

2.3 模型训练和解码优化问题概述

近些年来，深层神经网络[14]在机器学习领域取得了十分优秀的研究成果，在解决图像分类[25]、对象检测[26]、图像捕捉[27]和机器翻译[28]等任务上都不断取得非常领先的水平和成功的应用。深层神经网络如此强大的关键性原因包括数据集规模的快速增长、网络架构的日益深入以及训练深层神经网络中各种技术的飞速发展[29]。虽然规模更大的网络为当前神经网络的性能带来了正向的效果，但随着网络规模的扩大，网络中的参数也会越变越多，计算速度变得越来越慢，网络变得越来越冗余和繁重[17]。

现如今，神经机器翻译在机器翻译领域已取得了不错的性能，显著的提高了机器翻译的流畅行和充分性，但随着神经机器翻译的发展，一些问题也逐渐的显现出来。和其他深度学习领域一样，随着模型性能的增强，神经机器翻译模型规模和参数数量显著增加，但这些参数并不全是有意义甚至对于模型性能是有害的，我们称之为模型的过参数化问题[30]。神经机器翻译通常会受到这种过参数化的影响，导致模型的存储空间大、训练效率低，而神经机器翻译的部署需要非常昂贵的GPU设备，从而导致了神经网络模型的训练和部署瓶颈[31]。针对以上问题，越来越多的研究人员开始致力于研究如何在尽可能短的时间内完成模型的训练过程并尽可能的减小模型所需存储资源，与此同时，模型翻译的质量仍不受影响。

我们把对神经机器翻译模型优化的研究分为模型训练优化和模型解码优化。模型训练优化是指对训练模型的整个过程进行优化，目的是节省模型的计算资源和训练时间，这对一些计算资源和计算时间受限的训练情景十分重要，可以帮助我们在更短的时间内用更少的成本得到同等性能的模型。模型解码优化仅针对神经机器翻译模型的解码过程进行优化，是指如何减小模型的所需存储并用可得到的模型以更快的速度进行解码。神经机器翻译模型的训练优化和解码优化问题已经成为一项研究人员的重点研究问题，本文通过这两种研究方法在保证模型性能的同时提升模型的计算效率并显著的减少模型的参数量。

2.3.1模型训练优化问题

机器翻译的翻译质量在一定程度上十分受限于训练语料的质量和规模，训练语料的质量越好、数据规模越大，模型的性能往往越好。但随着训练语料规模的增大，往往需要更复杂的模型来适应这种数据规模，一系列的问题也随之出现，其中就包括训练时间代价和计算资源代价。当前神经机器翻译模型一般在高性能GPU或是集群上进行训练，在数据规模和模型规模较小的时候，训练代价并不高，但随着数据规模和模型规模的增大，训练代价显著的增高，一般来说，上亿规模的数据在8卡GPU上至少需要10-15天的训练时间，这为模型的训练和模型性能的提升都带来了不小的难题。为了解决以上问题，研究人员也在不断的进行研究并提出一些合理的解决方案。

一些研究人员致力于研究更精细的模型设计，当前深层神经网络具有巨大的层级规模，这给网络的训练和应用都带来了困难，如何减少网络所需内存与计算量就显得极为重要，其中一种模型压缩和加速方法就是设计出更精细的网络结构，使用这种精细而高效的模型结构进行模型的训练和推断可以极大的减小模型存储和训练代价。此种方法与其他方法的区别是从根本上减小模型的存储与计算规模，而不是对现有模型进行操作。在传统的模型设计过程中，为了增强模型性能，研究人员往往只在模型宽度和模型深度两个方向进行研究，比如ResNet[32]、LeNet[33]、AlexNet[34]和VGGNet[35]。现如今，由于嵌入式小设备的普及，越来越多的研究人员关注于如何降低模型的复杂度。

还有一些研究人员通过量化和低精度方法达到模型加速的目的，二者都是用更少的比特数来表示模型参数进行模型的压缩和加速。在神经网络当中，参数都是用浮点数表示，量化方法总体思想就是将模型中原有浮点数用更低精度的数值进行表示，通过牺牲参数值精度减少每个模型参数存储所需要的比特位数(将32bit单精度浮点数或 16bit半精度浮点数量化为 8bit甚至更少位的定点数)。在量化过程中，可以用K-means聚类的方式量化参数[36]，也可以使用更少的bit数，例如1-2bit来表示每个模型参数，这种网络又叫二值网络，在二值网络中网络权重只能是1或者-1，这种方法是量化中最极端的方法，极大减少了浮点数的操作并简化了专门为深度学习设计的硬件的实现方式，比如BinaryConnect[37]、Binarynet[38]和XNORNetwork[39]。

然而无论是哪种针对模型训练的优化方法，在机器翻译任务上研究的都比较少，都需要在机器翻译任务上进一步的进行研究。

2.3.2模型解码优化问题

模型解码和模型训练面临的问题又有所不同，在机器翻译的实际应用中，模型训练常常在高性能GPU上，而训练完成的机器翻译模型往往需要部署在存储资源和计算资源都较受限的小设备上，也就是说模型解码时面临的存储资源受限和计算资源受限的问题更加严重。

为了解决存储资源受限的问题，研究人员对模型压缩方法进行了初步研究，其中模型裁剪是目前模型压缩中使用较为广泛的方法，裁剪过程中可以设置一个硬性的阈值，将低于这个阈值的权值或者整个神经元裁剪掉，也可以较为灵活的设置一个百分比x%，按x%的比例裁剪掉模型参数。总体来说，剪枝的整个过程就是将网络由稠密变得稀疏的过程。在Yang等人[40]的研究工作中将模型的带宽和能耗加入到模型裁剪的考虑当中，从能量利用率上的来对模型进行裁剪。Anwar等人[41]采取一种随机裁剪以获得局部最优裁剪，以局部最优裁剪组合作为全局裁剪方式。Sun等人[42]通过分析神经元与神经元之间的相关程度来衡量参数重要性，将不重要参数剪枝。Molchanov等人[43]从不同的角度进行思考，将剪枝问题看做一个组合优化问题来解决。但上面所提的大部分工作都是针对计算机视觉领域CNN上的研究工作，当前研究人员对于RNN和最新的Transformer上的剪枝方法研究比较少。

为了解决计算资源受限问题，研究人员也提出了一系列模型解码加速方法，其中就包括对Beam Search解码算法的优化，现在对Beam Search的优化大多针对解码终止条件的研究[44]，通过某些条件来提前终止Beam Search的搜索过程，以达到解码加速的目的。在机器翻译等文本生成类任务上，此种方法可以在提高模型解码速度的同时进一步优化模型的性能。

2.4 本章小结

本章主要围绕机器翻译的相关技术进行了介绍，内容主要包括：神经机器翻译框架、神经机器翻译模型结构，并对神经机器翻译的模型训练优化和解码优化问题进行了简要概述。

在神经机器翻译框架部分，本文主要介绍了神经机器翻译的编码器-解码器框架和当前主流的基于注意力机制的编码器-解码器框架，注意力机制又分为全局注意力机制和局部注意力机制。在基于注意力机制的编码器-解码器框架中，注意力机制为每个源语单词都生成一个包含全局信息的向量，增加了模型的全局视野，使模型更充分的学习到序列中不同位置的复杂语义信息，增强了模型表示学习能力。

在神经机器翻译模型结构部分，本文主要介绍了三种比较常用的神经机器翻译模型结构：基于循环神经网络、基于卷积神经网络以及基于自注意力机制的神经机器翻译模型。文中提到的Transformer和ConvS2S于2017年分别被Google和Facebook提出，使输入序列中的每个词和其他词更有效的关联起来，使机器翻译模型更容易捕捉到不同位置间的空间信息。相较于循环神经网络，卷积结构的神经网络和基于自注意力机制的神经网络并没有输入顺序限制问题，极大的提升了神经机器翻译模型的并行性。

在神经机器翻译的模型训练优化和解码优化的概述部分，本章简要介绍了神经网络当前存在的过参数化问题，以及由过参数化引起的神经机器翻译模型计算效率和存储效率低下的问题。在后续的内容中本文将从训练和解码两个角度围绕神经机器翻译模型进行计算效率和存储效率上的优化。

第3章 神经机器翻译模型训练优化方法

3.1 研究动机

机器翻译为当今人们的出行和交流带来了巨大的便捷，但是由于数据规模的急剧增长，受硬件资源的限制，神经机器翻译系统的训练速度和存储遭遇了极大的困难，这就使得研究人员逐渐将研究重点转向了神经网络模型的加速和压缩研究工作。目前对神经网络加速和压缩问题的研究在计算机视觉领域上已经取得了不错的进展，但是针对神经机器翻译模型相关问题的研究正处于起步阶段，本章主要针对神经机器翻译模型训练中的一些问题进行优化，分别从计算角度和模型结构进行考虑，在提升模型训练速度的同时减少相同性能下模型所需参数量。

从计算角度来说，我们采取了一种基于低精度运算的神经机器翻译模型加速方法。在计算机科学中，张量通常被定义为n维空间中的一种量，它具有n个分量，这种张量本质上是一个多维数组。在许多机器学习工具中，张量计算已经成为了神经网络前向、反向传播等过程的基本单元，应用十分广泛。神经网络中存在着大量的张量计算，尤其是浮点数类型的张量计算，为神经网络模型的存储和训练都带来了极大困难，因此存储优化和加速的一个策略就是利用较低精度的近似值来代替浮点运算。

当前的神经网络都是以浮点数为单位来进行计算，因此无论是在神经网络的训练过程还是推断过程，最耗时的部分莫过于以浮点数为单位的各种张量计算。浮点运算的计算效率低且难以进行优化，而神经网络模型中参数常常达到千万甚至亿级，网络中大量的浮点操作使得神经网络笨重且训练缓慢。为了解决神经网络训练效率低下的问题，研究人员不断致力于研究神经网络的模型加速和压缩工作。当前的一些加速方法（如：模型裁剪、参数量化、结构稀疏化、精细模型的设计等）并没有办法从根本上解决模型冗余且训练效率低下的问题，因此研究人员提出了神经网络低精度化方法。

对神经网络低精度化方法的研究最早可以追溯到20世纪90年代[45]-[46]，但由于当时条件和资源限制，并不能很好的验证神经网络低精度化方法在深层神经网络及大规模数据上的有效性。虽然当前的硬件设备发展迅速，硬件资源的计算速度与存储容量都在不断的得到提升，但随着当前深层神经网络的深度和训练数据的规模都在不断增大，模型的训练速度与存储大小正成为阻碍神经网络发展的瓶颈问题。低精度神经网络即将神经网络中权重用低精度的位数表示，这种低精度表示问题可以看作是将网络权重由连续分布转换为离散分布的过程。运行大型的神经网络需要大量的内存带宽和计算资源来获取权重和点积运算，而由于移动设备受电池限制，使得诸如此类的深度神经网络等耗电量大的应用难以部署，而低精度网络的目的就是解决由神经网络过参数化问题而导致的模型占存储空间大且训练和解码速度慢的问题。

对于神经网络低精度化方法，现已有研究人员成功的利用低精度的近似值来训练神经网络模型。通过调整定点运算，Vanhoucke等人[47]在语音识别任务上进行前馈神经网络的8位量化，Devlin等人[48]探索了机器翻译的16位量化，Quinn等人[49]展示了使用32位浮点值训练模型并用8位整数进行解码的有效性。虽然当前对于神经网络模型加速和压缩的研究工作已经取得了一定的进展，但是关于机器翻译任务的研究却很少，关于低精度神经网络模型的研究更是少之又少。神经网络中耗时最多的部分就是32位浮点运算，不仅因为单位浮点运算消耗时间，更是因为神经网络的训练和推断过程中都存在着大量的浮点运算，使得浮点运算的低效率成为一个不得不被解决的问题。低精度运算与其他模型加速方法的区别就是我们可以利用更低精度的网络权重代替原本的32位浮点数，大大减少了浮点运算时间，可以对模型运算速度进行显著的提升。由于现有硬件以及开源库的支持问题，研究人员关于低精度运算的研究工作大部分情况下只是模拟低精度运算的性能而并非真正实现了模型的加速。

为了解决以上问题，本文采用神经网络低精度化方法来优化神经机器翻译模型计算过程。为了尽量保持模型精度，我们采取一种混合精度计算的方法，将神经机器翻译模型中部分运算强度较高且对浮点位数并不敏感的运算挑选出来，将这些浮点操作换成更低精度的半精度操作，此种方法在尽可能保证模型精度的同时提升模型的计算效率。

从模型结构角度来说，我们采取了一种基于分组卷积结构的神经机器翻译模型加速方法。传统的循环神经网络和卷积神经网络中参数规模非常庞大，通常可以达到百万或亿级别，一些关于大规模分布式训练的工作甚至要用数以万计的CPU内核进行训练[49]，这些深层神经网络中的参数并非都对网络起到正向的效果，而是存在冗余，冗余参数常常会导致过拟合和过参数化问题。深层神经网络过参数化的理论告诉我们，网络中过多的参数很多对整个神经网络是无益甚至是有害的，这些参数不仅不能提升网络的性能，甚至会降低模型的训练速度和计算资源。

目前最主流的机器翻译模型，也就是基于自注意力机制的神经机器翻译模型，虽然相较于基于循环神经网络和基于卷积神经网络的机器翻译模型在模型并行性、训练速度和模型性能方面都取得了显著的进步，但由于其存在大量的注意力权重计算，包括编码端、解码端和编码端与解码端间的注意力结构，导致模型的训练速度仍然存在瓶颈[50]，在计算资源受限和实时响应的环境下仍难以部署和应用。

以基于自注意力机制的神经机器翻译模型Transformer为例，在模型的训练过程中，编码器和解码器中占时间比比较大的两个操作主要是词向量层的计算时间和前馈神经网络计算的时间。基于自注意力机制的Transformer模型常用模型训练配置主要有transformer\_base和transformer\_big，在NIST12中英180w数据集上训练时二者参数量如表3.1所示。

表3.1 Transformer模型参数量

Table 3.1 Transformer model parameters

|  |  |
| --- | --- |
| 模型名称 | 参数量 |
| transformer\_base | 101777408 |
| transformer\_big | 291635200 |

如表3.1所示，在神经机器翻译模型中，无论是较大模型还是较小模型，模型规模都十分庞大，模型参数都在千万级别甚至可以达到上亿。如此庞大的参数为模型训练带来了极大的困难，占用了宝贵的计算资源，其中前馈神经网络的计算时间占据了很大的比例。以transformer\_base模型为例，在其训练过程中，前馈神经网络的计算时间占总编码器和解码器计算时间比例大概为50%，模型规模越大，一般来说这部分占比时间越高，因此如何优化前馈神经网络计算的时间成为了一个亟待解决的问题。

3.2 方法描述

3.2.1基于低精度运算的神经机器翻译模型加速方法

神经网络中的权值都是单精度浮点数，单精度浮点数在计算机中为32bit，符号位占1bit，指数位占8bit，尾数位占23bit，单精度浮点数位数示意图如图3.1上图所示。在模型的训练过程，虽然保持权值的高精度可以更大概率的得到较优的模型，然而在研究过程中我们发现神经网络对参数精度并不敏感，适当降低参数精度不会影响模型性能。

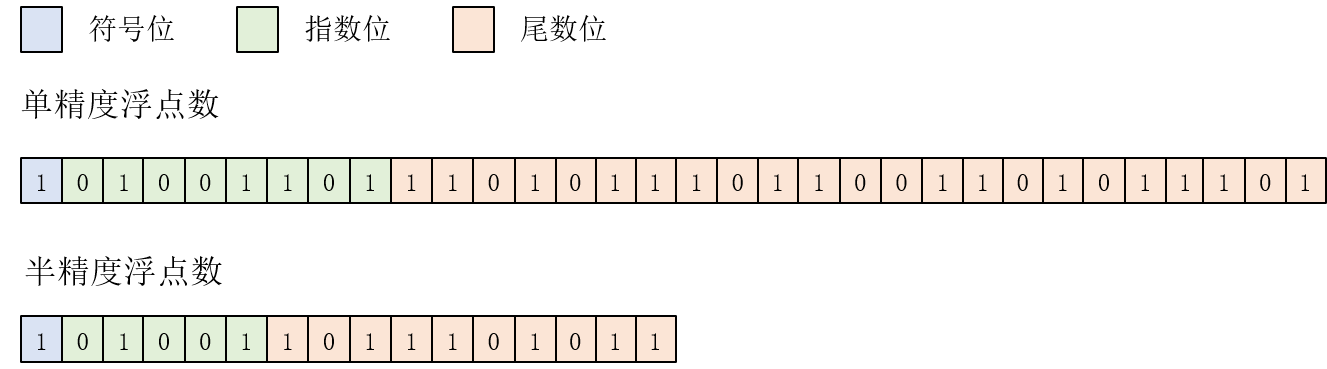


图3.1 单精度与半精度浮点数

Fig. 3.1 Single-precision float and Half-precision float

虽然在特定型号支持半精度运算的显卡上，半精度浮点数计算效率较高，但并不能将神经网络所有参数全部替换成半精度，这是由于半精度浮点和单精度浮点的表示范围不同，半精度浮点位数示意如图3.1所示。由图中可以看出单精度浮点数的符号位占1位，指数位占8位，尾数位占23位，半精度浮点数符号位占1位，指数位占5位，尾数位占10位，由此而言，单精度浮点数表示范围比半精度浮点数更广。

为了进一步分析低精度神经网络运算在机器翻译任务上的可行性，以Transformer模型的参数分布为例，在整个模型中参数属于连续正态分布，且整个参数的数值分布范围较大，因此一些可以用单精度浮点数表示的数值无法用半精度浮点数来表示，可能会存在溢出的情况。从以往的经验来看，若神经网络中模型参数存在上溢或者下溢的情况，对神经网络模型的性能会产生非常大的负面作用，为了解决这一问题，在某些情况下需要进行半精度浮点和单精度浮点之间的数据类型转换。

在神经机器翻译模型当中，有两个部分的参数容易存在参数溢出情况，用半精度浮点数很难表示，在低精度神经网络的计算过程中需要单独进行处理。一是反向梯度更新部分，二是softmax的计算部分。

Transformer有关softmax的计算主要有两处，一是在每个注意力层计算之后，二是在预测词汇概率之前，如图3.2所示。

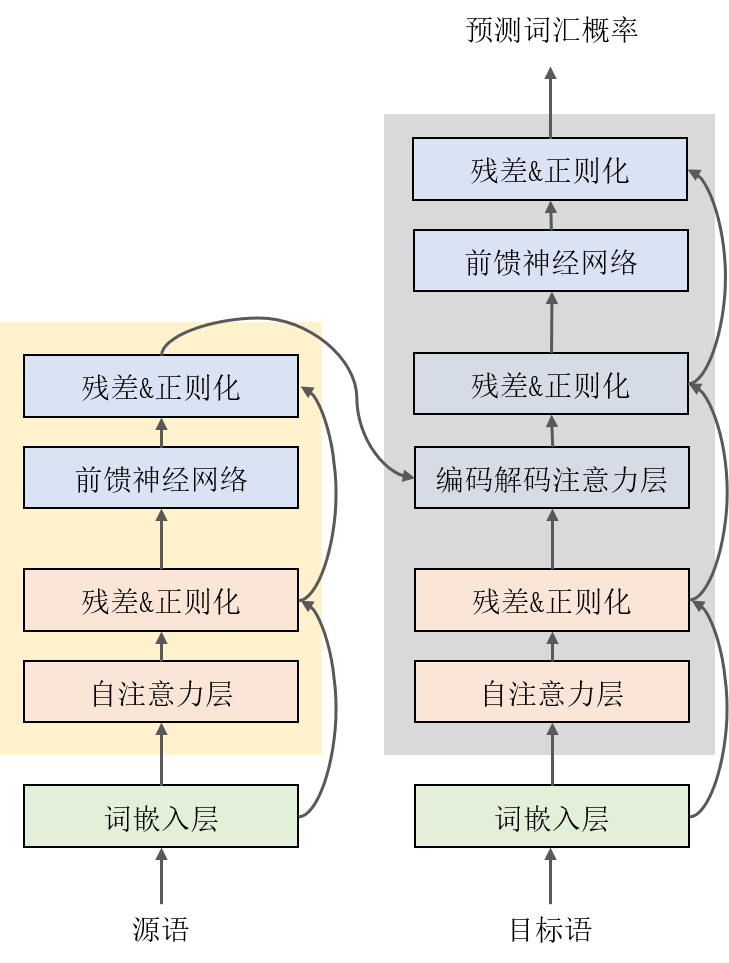


图3.2 Transformer模型结构示意图

Fig. 3.2 Transformer model architecture

神经网络的训练过程可以分成两个步骤，一是前向推断过程，二是反向梯度更新过程，反向更新过程是为了根据前向计算出来的损失值来进一步调整和更新参数，这一步在神经网络中至关重要，因此这一部分参数对精度十分敏感。而且梯度更新时的数值往往数量级较小，因此用半精度较难表示，需要将梯度平移到半精度浮点的表示范围内。

softmax函数是一种归一化的函数，可以将一个任意维度的向量变为一个归一化的向量，每一个元素都在(0,1)范围间，并且所有元素之和为1，softmax计算公式如公式(3.1)所示。

(3.1)

其中j=1,…,K，softmax将有限项数离散概率分布的向量归一化，其中分母是K个元素的指数项之和，在神经机器翻译中，K通常以万为数量级，这种大规模指数项求和作分母的计算公式用半精度浮点数较难表示。

为了解决以上问题，我们采取一种混合精度计算的方法，在整个神经网络中部分参数采用半精度浮点数，另一部分采用单精度浮点数，此种方法可以在保证模型性能的前提下最大程度的发挥低精度神经网络计算的优势。

在神经网络中存在两种角色，一是Weights，二是Activations。Weights，也就是我们经常说的网络权重，是一些进行线性变换的网络参数，这些参数也可以看作是神经网络模型特征的提取器。Activations是我们常说的激活函数，主要包括一些非线性变换，在神经网络中加入非线性变换是为了提升模型的表示和学习能力。

在整个低精度神经机器翻译模型的计算过程中，每个半精度浮点权重都有一个与之对应的单精度浮点的备份，我们用Master来表示这个主副本，这个备份用在反向更新过程中，也就是说反向更新参数的时候使用32位单精度浮点数来更新，这是为了使参数更新过程更加准确。我们用Weights表示半精度浮点权重，Weights(Master)表示单精度浮点权重，Activations代表网络中的激活函数，Weights Grad代表网络中权重的梯度，Activations Grad代表网络中激活函数的梯度，则低精度神经网络训练的计算过程如图3.3所示，整个神经机器翻译混合精度计算可分为以下步骤：

1. 在神经网络的前向计算过程中，由16位的Weights和16位的Activations计算得到16位的Activations。
2. 在反向计算过程中，由16位的Weights和16位的Activations Grad得到16位的Activations Grad。
3. 在反向计算过程中，由前两步得到的16位的Activations和16位的Activations Grad计算得到16位的Weights Grad。
4. 将第三步得到的Weights Grad进行16位半精度浮点到32位单精度浮点的数据类型转换。
5. 用数据类型转换过后的32位Weights Grad来更新参数的主副本Weights(Master)。
6. 将Weights(Master)进行32位单精度浮点到16位半精度浮点的数据类型转换得到下一轮计算需要的Weights参数。
7. 回到第一步，循环执行以上步骤直至训练过程结束。

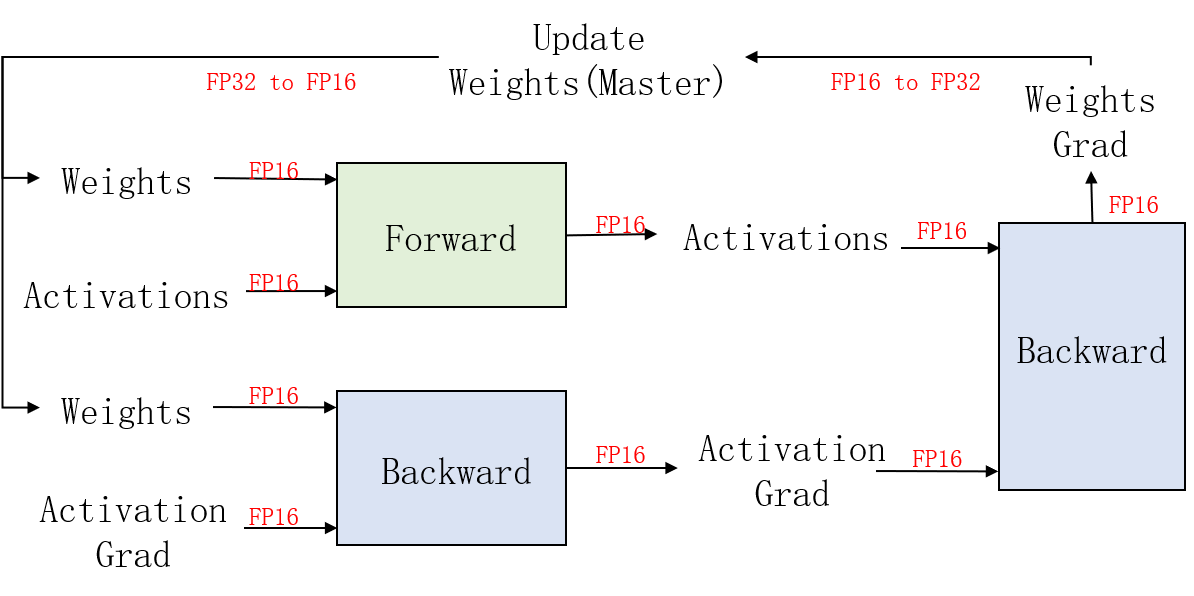


图3.3 神经机器翻译混合精度计算过程

Fig. 3.3 Mixed-precision computing process of neural machine translation

3.2.2基于分组卷积的神经机器翻译模型加速方法

在基于自注意力机制的神经机器翻译模型训练过程中，存在着大量的前馈神经网络计算操作，这些操作不仅计算次数多且参与运算的矩阵规模很大，虽然这些前馈之间存在着一定的并行性，但其在网络中仍具有很大的计算时间占比。为了解决这一问题，我们引入了一种分组卷积结构来代替传统的前馈神经网络操作，从而达到神经机器翻译模型训练过程加速的目的。

一般来说，前馈神经网络是一个全连接的稠密的结构，一个简单的前馈神经网络结构如图3.4所示。可以看出，由于前馈神经网络结构非常稠密，里面存在着大量的矩阵运算，这些矩阵运算非常的繁重且耗时[51]。

为了解决前馈神经网络耗时问题，研究人员提出可以用卷积神经网络来代替传统的前馈神经网络。卷积神经网络相比于前馈神经网络，模型结构的本质都是层级网络，只是层级结构的形式和功能发生了变化，卷积神经网络按层级结构进行划分，可以分为4部分：数据输入层，卷积计算层，池化层和全连接层[52]。数据输入层用于数据的输入，在卷积计算层中包含着一些神经元，这些神经元中的每个神经元都用来连接数据窗口，连接窗口的权重是固定的，这些窗口可以看作是一类模型特征。

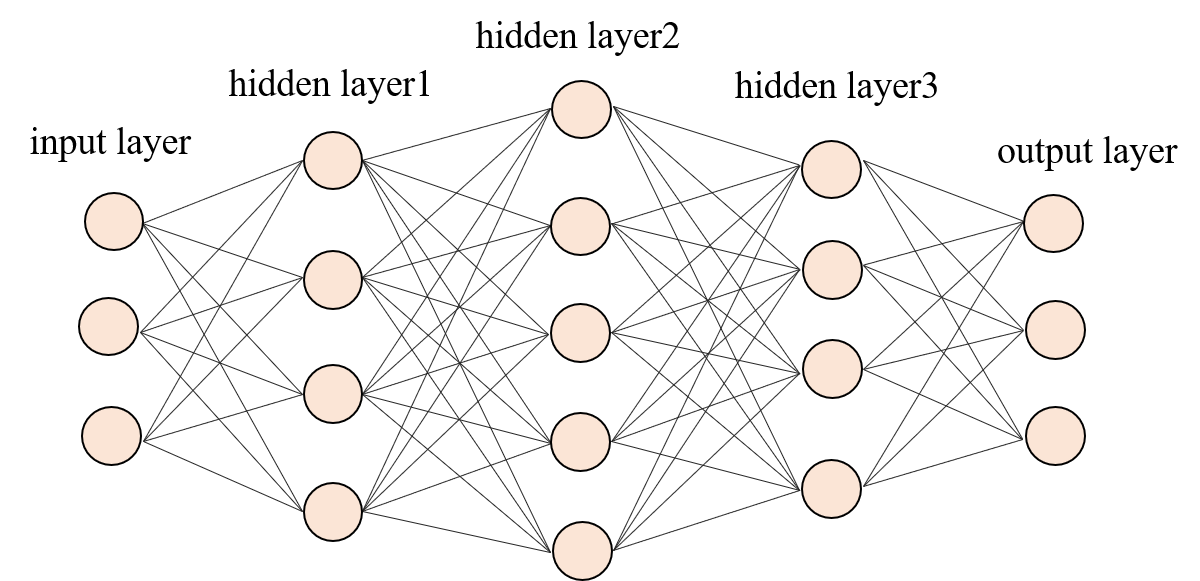


图3.4 前馈神经网络模型结构

Fig. 3.4 Feedforward neural network architecture

在图像处理任务中，以上模型特征可以是颜色、纹理、垂直边缘、水平边缘等等，在机器翻译任务中，这些特征包含的是一些语义信息。池化层夹在卷积层和卷积层之间，用于压缩参数数量。全连接层在整个卷积结构的末尾，传统的神经网络全连接方式一样，加入全连接层是为了进一步增强模型的表示能力，卷积神经网络模型结构如图3.5所示。

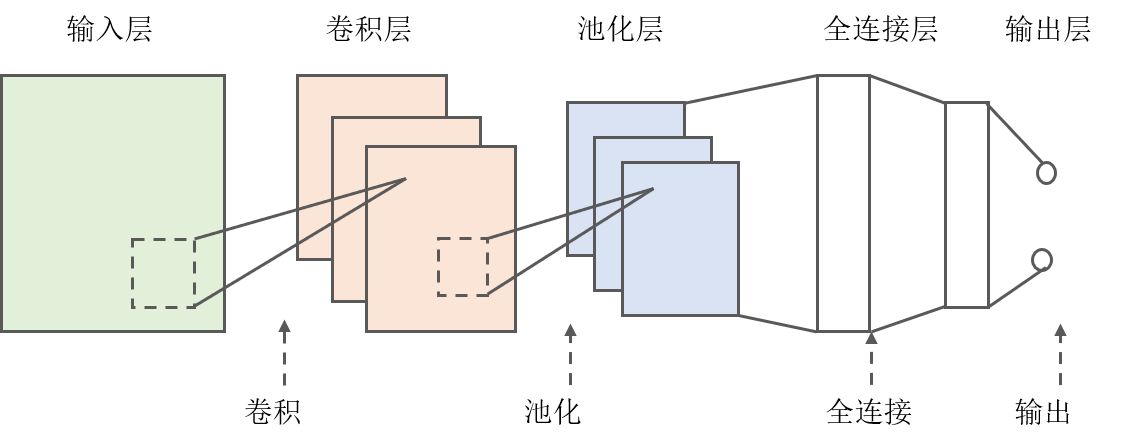


图3.5 卷积神经网络模型结构

Fig. 3.5 Convolutional neural network architecture

卷积神经网络大多应用在图像识别、图像分类等计算机视觉领域的任务上，在机器翻译任务中的应用较少，为了进一步的提升模型的训练速度，我们引入了一种分组卷积结构(Group Convolutions)来对神经机器翻译模型的训练过程进行进一步的加速，使模型的计算更为高效。

卷积神经网络中通常会包含很多重复模块，但计算复杂度也很高。为了使模型计算更加的高效，可以通过稀疏连接的方法将普通的卷积操作转换为分组卷积。其中每一个卷积都有对应的应用通道，且每个卷积仅在其对应的输入通道上进行操作，从而显著的降低计算消耗，分组卷积的模型结构如图3.6所示。通过分组卷积操作，原本卷积中全连接结构变得稀疏，不再存在着跨组连接，减少了卷积中的运算次数。但这种分组卷积会存在一个比较明显的问题，将卷积操作分组进行后，组与组之间的信息流通被阻碍了，每个组内信息只在其对应的通道内流过，降低了模型的信息表示能力。

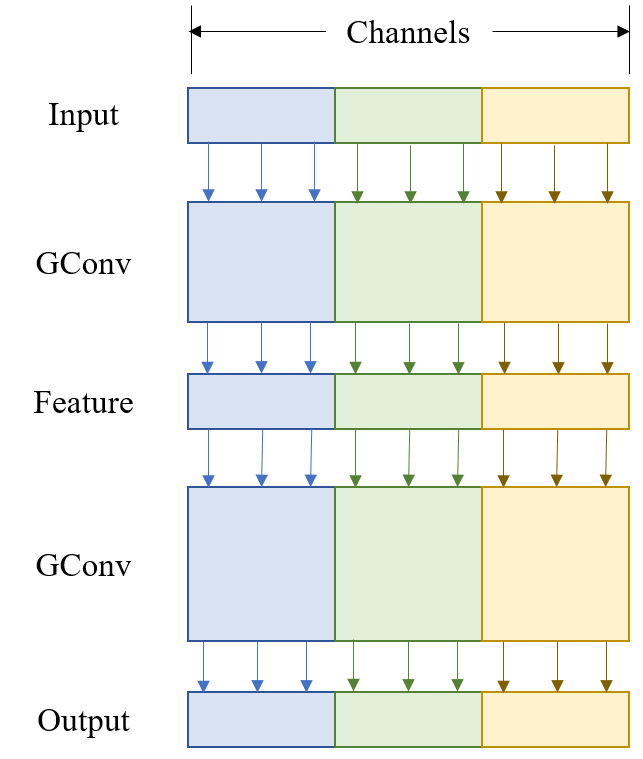


图3.6 分组卷积神经网络

Fig. 3.6 Group convolutional neural network

为了进一步提升分组卷积的信息表示能力，需要进行组与组之间的交流操作，为了保持计算效率，只在组间加入部分连接，在保证组间信息流动的同时仍能提高运算效率，达到计算效率和模型性能之间的平衡，如图3.7所示。

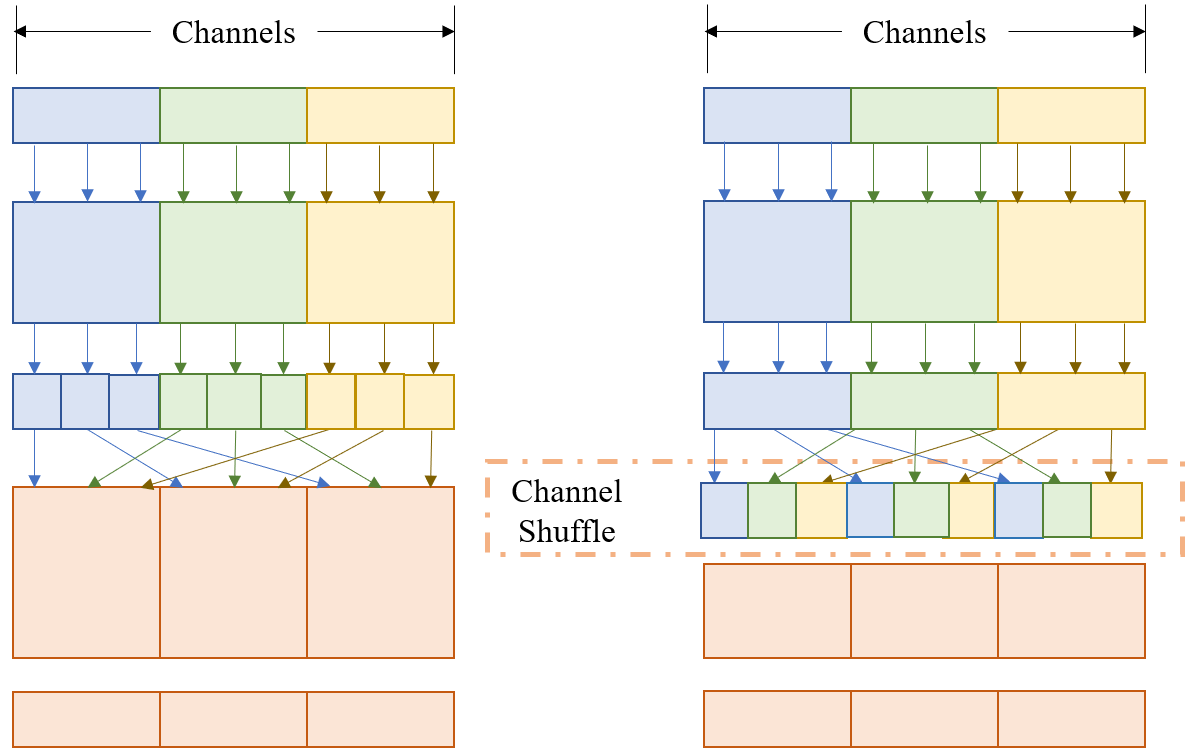


图3.7 分组卷积组间操作

Fig. 3.7 Group convolutional operation within the group

图3.7左是分组卷积的基本原理，在保证稀疏结构的前提下，增加卷积网络组间连接，目的是增加组与组之间的信息流通性，图3.7右是这一方法的具体实现方式，也就是将每个组内进行分割，之后将分割结果均匀排列，以提升模型的表示能力和识别精度。

3.3 实验

3.3.1 实验设置

对本文所采取的基于低精度运算的神经机器翻译模型加速方法和基于分组卷积的神经机器翻译模型加速方法，本文在德英和中英两个语言任务上进行了实验，其中德英数据用来测试小规模数据下方法的有效性，中英数据用来测试大规模数据下方法的有效性。本文实验采用的是机器翻译自动评价指标BLEU[53] 作为评价方式，并且采用Moses公开的multi-bleu测试脚本来对实验结果进行验证。

其中，德英数据来自IWSLT'14提供的16.0w的公开双语句对，由于其数据规模较小，因此本文用其来验证小规模数据在单卡训练时方法的准确性。本文从训练数据中抽取7283条数据作校验集，用test2012的6750条数据作测试集。我们采取BPE(Byte Pair Encoding)[54]-[55]子词切分算法对训练集、校验集和测试集基础分词结果进行更细粒度的切分，切分的merge次数为10k，训练集上源语端（德语）和目标语（英语）的词表大小分别是8844和6628。

中英训练数据来自中英方向的NIST12 OpenMT，其中包含双语平行句对189.8w。本文中实验采用NIST2006(MT06)作校验集，采用NIST2004(MT04)、NIST2005(MT05)、NIST2008(MT08)作测试集，所有长度超过50词的句子均被过滤掉。为了进一步提升模型性能，采用merge次数为32k的BPE子词切分技术，最终得到的源语端（中文）词表大小为44668，目标语端（英语）的词表大小为33948。以上所有数据集的统计信息如表3.2所示。

表3.2实验数据集统计

Table 3.2 Experiment dataset statistics

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **数据集种类** | **数据集来源** | **训练集双语句对** | **BPE** | **校验集** |
| 德语-英语 | IWSLT'14 | 160239 | 10000 | test2012 |
| 中文-英语 | NIST12 OpenMT | 1897809 | 32000 | NIST2006 |

关于训练设备，我们采用支持半精度神经网络计算的带图形处理器的服务器对神经机器翻译系统进行训练和解码，为了提升模型的训练速度并减少模型的训练周期，我们采用8张基于Volatile架构的TITAN V进行模型的多卡训练，该设备的具体信息如表3.3所示。

表3.3 服务器配置信息

Table 3.3 Server configuration information

|  |  |
| --- | --- |
| **组件** | **配置** |
| 图形处理单元 | NVIDIA TITAN V |
| 中央处理器 | Intel(R) Xeon(R) Gold 5118 CPU @ 2.30GHz |
| 物理CPU个数 | 2 |
| CPU核心数 | 12 |
| 逻辑CPU个数 | 48 |
| 内存 | 128GB |
| 显存 | 12GB |
| 硬盘 | 5TB |

在模型训练方面，我们采用的是基于PyTorch的神经机器翻译模型，可以看作是Torch的改进版本，支持动态图且提供了方便的Python接口，本文即在其上进行二次开发，以此训练出实验模型。

在模型训练时我们采取的参数设置为transformer\_base、transformer\_small和transformer\_big，两个模型的规模不同，transformer\_base是较为基础的参数设置，transformer\_small是较小规模的参数设置，transformer\_big是较大规模参数设置，我们分别在两套标准参数配置上进行实验来更加合理的验证实验结论的正确性，三套参数配置的具体参数配置如表3.4所示。

表3.4模型训练参数设置

Table 3.4 Model parameter configuration

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **参数** | | **transformer\_small** | | **transformer\_base** | | **transformer\_big** | |
| 层数 | | 6 | | 6 | | 6 | |
| 隐层大小 | 256 | | 512 | | 1024 | |
| batch大小 | 1024 | | 4096 | | 4096 | |
| 头数 | 4 | | 8 | | 16 | |
| 优化器种类 | Adam[56] | | Adam | | Adam | |
| 更新轮数 | 45 | | 15 | | 15 | |

其中，层数为6代表模型中编码器和解码器中各有6层，隐层大小通常代表模型表示能力的大小，一般来说隐层越大则模型的表示能力越强，batch大小代表模型每次更新读取的数据量，过滤器可以进一步提升模型表示能力。

在模型解码过程我们用单张的TITAN V进行解码，在所有数据集上模型训练统一在15轮，为了进一步提升模型的稳定性，我们采用checkpoint ensemble方法对每次训练结束的最后几个模型进行参数平均，在两个数据集上模型解码参数如表3.5所示。

表3.5 模型解码参数设置

Table 3.5 Model decoding parameter configuration

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **数据集种类** | **batch大小** | **长度惩罚因子** | **beam大小** | **ensemble个数** |
| 德语-英语 | 128 | 1.0 | 5 | 5 |
| 中文-英语 | 128 | 1.0 | 12 | 5 |

3.3.2 实验结果

3.3.2.1 基于低精度运算的神经机器翻译模型加速方法

为了验证低精度神经网络方法在神经机器翻译任务上的有效性和加速效果，我们分别在德英和中英任务上进行实验，其中英德数据集规模比较小，训练集为160239个双语句对。为了保证模型的训练效果，避免过拟合现象的产生，我们在单卡上进行模型训练，并在模型规模较小的transformer\_small和transformer\_base参数配置上进行实验，来验证数据规模较小时模型规模大小对低精度神经网络训练效果的影响，实验结果如表3.6所示。

表3.6 de-en翻译任务上的低精度运算实验

Table 3.6 Low-percision experiments on de-en machine translation task

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **系统** | **开发集** | **测试集** |  | **加速比** |
|  | **test2012** | **平均** |
| small | 34.43 | 33.7 | 34.07 | - |
| small+低精度运算 | +0.07 | -0.09 | -0.01 | 39.84% |
| base | 35.77 | 34.51 | 35.14 | - |
| base+低精度运算 | -0.08 | -0.04 | -0.06 | 52.34% |

从上表结果可以看出，低精度运算方法在机器翻译任务上可以极大提升神经机器翻译模型的计算效率，同时对模型的性能基本上不产生任何影响。随着模型规模的增大，模型中参数量增加，模型中神经元连接数增加，低精度运算方法对神经网络训练的加速效果会更明显。

在低精度运算实验中，中英数据集有1897809个双语句对，数据规模较大，单卡训练过程较长。为了加速模型训练和收敛过程，我们采取多卡训练方法，用8张显卡同时训练。由于数据规模较大，我们采取transformer\_base和transformer\_big的参数配置，来验证数据规模较大时模型规模大小对低精度神经网络训练效果的影响，这部分的实验结果如表3.7所示。

表3.7 zh-en翻译任务上的低精度运算实验

Table 3.7 Low-percision experiments on zh-en machine translation task

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **系统** | **开发集** | **测试集** |  |  |  | **加速比** |
| **mt06** | **mt04** | **mt05** | **mt08** | **平均** |
| base | 50.86 | 53.8 | 49.15 | 44.19 | 49.50 | - |
| base +低精度运算 | -0.17 | -0.19 | -0.11 | +0.02 | -0.11 | 52.52% |
| big | 51.24 | 54.49 | 50.27 | 46.78 | 50.70 | - |
| big +低精度运算 | +0.07 | +0.02 | +0.13 | -0.01 | +0.05 | 58.70% |

在中英任务上的实验进一步验证了低精度运算方法的有效性，在神经机器翻译任务上，模型规模越大，加速的效果越明显。

3.3.2.2 基于分组卷积的神经机器翻译模型加速方法

在神经机器翻译的训练过程中，存在着大量的前馈神经网络计算。接下来的实验主要对比基于分组卷积的神经机器翻译模型和基于前馈神经网络的神经机器翻译模型的性能、计算速度以及模型参数量，我们主要在德英数据集上进行对比分析实验，在中英数据集上对所得结论进行进一步的验证。

首先我们在德英数据集上基于transformer\_small和transformer\_base参数配置来探究分组卷积中组数对模型性能的影响，在校验集和测试集上的性能变化如表3.8所示。

其中group1是全连接的结构，即等同于前馈结构，在这里我们将group1作为基线结构，随着group-n中n的增大，模型参数量逐渐减少，模型中计算复杂度逐渐减小，当group为n时，计算复杂度为原本的1/n，但模型性能所受影响也逐渐增大。相对于模型规模较小的transformer\_small来说，transformer\_base模型性能所受影响更小，且group相同时，transformer\_base比transformer\_small的压缩比例和加速效果都更强。以上实验结果说明，随着模型规模的增大，模型中存在的冗余参数越多，模型可压缩和加速的比例就越大。由于在transformer\_base参数配置下模型性能更好，可压缩和加速的潜力也更大，接下来的实验中我们都以transformer\_base作为实验参数设置。

为了进一步验证以上结果，我们在更大的数据集上进行分组卷积实验，我们选取数据集大小为180w的中英数据集在transformer\_base上进行实验，为了加速模型训练和收敛过程，我们采取多卡训练方法，用8张显卡同时训练。当group分别为1，2，4，8时中英四个测试集的性能分别如图3.8所示，group为1时性能即为全连接的基线系统性能。与以上结论相同，随着分组卷积中group的增大，虽然模型中参数量会越来越少，模型性能也会随之变差。

表3.8 de-en翻译任务上的分组卷积实验

Table 3.8 Group convolution experiments on de-en machine translation task

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **系统** | **开发集** | **测试集** |  | **压缩比例** |
|  | **test2012** | **平均** |
| small+group1 | 34.46 | 33.82 | 34.14 | 1.00× |
| small+group2 | -0.62 | -0.72 | -0.67 | 1.23× |
| small+group4 | -1.27 | -1.42 | -1.35 | 1.39× |
| small+group8 | -0.93 | -1.90 | -1.42 | 1.49× |
| small+group16 | -1.08 | -1.20 | -1.14 | 1.55× |
| small+group32 | -1.15 | -1.15 | -1.15 | 1.57× |
| base+group1 | 35.53 | 34.28 | 34.91 | 1.00× |
| base+group2 | -0.10 | -0.07 | -0.09 | 1.29× |
| base+group4 | -0.52 | -0.30 | -0.41 | 1.52× |
| base+group8 | -0.94 | -0.82 | -0.88 | 1.66× |
| base+group16 | -1.30 | -1.09 | -1.20 | 1.74× |
| base+group32 | -1.58 | -1.57 | -1.58 | 1.78× |

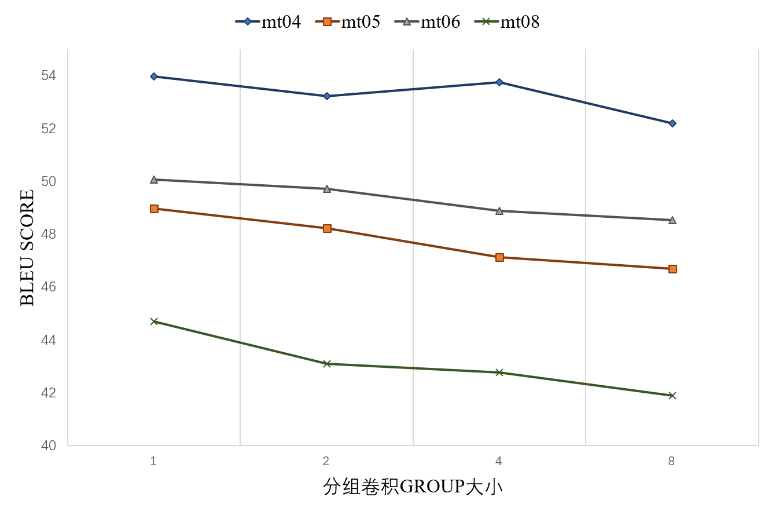


图3.8 zh-en翻译任务上的分组卷积实验

Fig. 3.8 Group convolution experiments on zh-en machine translation task

在Transformer模型中，编码器和解码器中分别有6层，为了探索每层中组数变化对模型性能的影响，我们将每相邻的两层作为一组，分为encoder01、encoder23、encoder45、decoder01、decoder23、decoder45六组，使每相邻两层中分组卷积的组数保持一致，为了使每组实验结果对比更明显，我们选取分组卷积实验的组数为对模型性能影响较大的32，该实验的实验结果如表3.9所示。

表3.9 de-en翻译任务上不同层采取不同组数的分组卷积实验

Table 3.9 Group convolution experiments on de-en machine translation task with different group

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **系统** | **开发集** | **测试集** |  | **压缩比例** |
|  | **test2012** | **平均** |
| base+group1 | 35.53 | 34.28 | 34.91 | 1.00× |
| base+encoder01group32 | -0.14 | -0.28 | -0.21 | 1.08× |
| base+encoder23group32 | -0.06 | +0.04 | -0.01 | 1.08× |
| base+encoder45group32 | +0.03 | +0.02 | +0.03 | 1.08× |
| base+decoder01group32 | -0.18 | -0.11 | -0.15 | 1.08× |
| base+decoder23group32 | -0.42 | -0.69 | -0.56 | 1.08× |
| base+decoder45group32 | -1.69 | -1.68 | -1.69 | 1.08× |

如表3.9所示，在encoder中越高层次用分组卷积对模型性能影响越小，在decoder中越高的层次使用分组卷积方法对模型性能的影响越大。也就是说越靠近输入和输出的地方使用分组卷积方法对模型性能造成的影响更大，这是由于encoder低层比较靠近数据编码层，对其削减会对数据输入造成较大影响，而decoder的高层靠近概率输出层，对其削减会使模型失去重要的特征参数。

为了进一步验证以上结果，我们在中英数据集上进行进一步的验证实验，在保持其他层数卷积组数为1的前提下，分别对encoder的较低层012层、encoder较高层345层、decoder层较低层012层、decoder层较高层345层组数变为4，实验结果如图3.9所示。

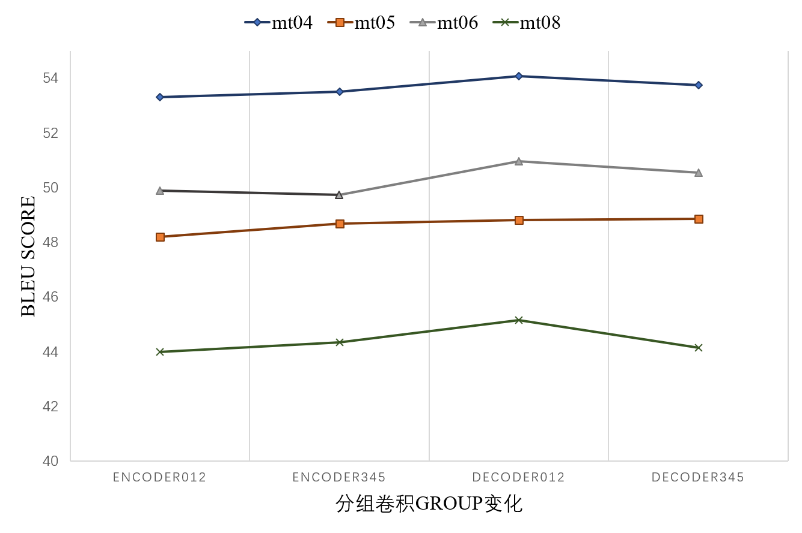


图3.9 zh-en翻译任务上不同层采取不同组数的分组卷积实验

Fig. 3.9 Group convolution experiments on zh-en machine translation task with different group

为了探究在Transformer中使用分组卷积方法所能达到的最大的模型压缩和模型加速的比例，我们进一步进行各层分组卷积的组合实验如表3.10所示。

表3.10 de-en翻译任务上分组卷积组合实验

Table 3.9 Group convolution combined experiments on de-en machine translation task

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **系统** | **encoder group** | | | | | | **deocder group** | | | | | | **开发集** | **测试集** |  | **压缩比例** |
| **0** | **1** | **2** | **3** | **4** | **5** | **0** | **1** | **2** | **3** | **4** | **5** |  | **test2012** | **平均** |
| base | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 35.53 | 34.28 | 34.91 | 1.00× |
| base | 8 | 8 | 8 | 8 | 8 | 8 | 4 | 4 | 4 | 4 | 4 | 4 | -0.64 | -0.65 | -0.65 | 1.58× |
| base | 4 | 8 | 8 | 8 | 8 | 8 | 4 | 4 | 4 | 4 | 4 | 4 | -0.72 | -0.24 | -0.48 | 1.57× |
| base | 4 | 4 | 8 | 8 | 8 | 8 | 4 | 4 | 4 | 4 | 4 | 4 | -0.72 | -0.54 | -0.63 | 1.56× |
| base | 4 | 4 | 4 | 8 | 8 | 8 | 4 | 4 | 4 | 4 | 4 | 4 | **-0.25** | **-0.11** | **-0.18** | **1.55×** |
| base | 4 | 4 | 4 | 4 | 8 | 8 | 4 | 4 | 4 | 4 | 4 | 4 | -0.54 | -0.32 | -0.43 | 1.54× |
| base | 4 | 4 | 4 | 4 | 4 | 8 | 4 | 4 | 4 | 4 | 4 | 4 | -0.28 | -0.10 | -0.19 | 1.53× |
| base | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | -0.52 | -0.30 | -0.41 | 1.52× |
| base | 4 | 4 | 4 | 8 | 8 | 8 | 4 | 4 | 4 | 4 | 4 | 2 | -0.35 | -0.27 | -0.31 | 1.57× |
| base | 4 | 4 | 4 | 8 | 8 | 8 | 4 | 4 | 4 | 4 | 2 | 2 | -0.40 | -0.23 | -0.32 | 1.51× |
| base | 4 | 4 | 4 | 8 | 8 | 8 | 4 | 4 | 4 | 2 | 2 | 2 | -0.23 | -0.09 | -0.16 | 1.48× |
| base | 4 | 4 | 4 | 8 | 8 | 8 | 4 | 4 | 2 | 2 | 2 | 2 | **-0.06** | **+0.01** | **-0.03** | **1.46×** |
| base | 4 | 4 | 4 | 8 | 8 | 8 | 4 | 2 | 2 | 2 | 2 | 2 | -0.10 | -0.29 | -0.20 | 1.44× |
| base | 4 | 4 | 4 | 8 | 8 | 8 | 2 | 2 | 2 | 2 | 2 | 2 | -0.26 | -0.27 | -0.27 | 1.42× |

如表3.10所示，随着分组卷积中group的增大，模型性能逐渐下降，用各层间不同group的组合实验可以使模型压缩比例最大达到1.46倍而不影响模型的性能。

3.3.3 结果分析

3.3.3.2 模型性能

关于神经机器翻译模型训练优化方法，根据上一小节的实验我们得到了一些经验性的结论。

(1) 当使用低精度运算方法来进行神经机器翻译训练过程的优化时，模型性能基本不会受到影响。

我们首先在较小数据集的德英任务上进行实验，实验结果如表3.6所示，我们选取了适合小规模数据集的两套参数配置transformer\_small和transformer\_base进行实验，无论在哪种参数配置上，低精度运算方法对模型性能基本没有影响。为了验证以上结论，我们进一步在较大规模的中英数据集上进一步进行实验，选取了适合大规模数据集的transformer\_base和transformer\_big参数配置，进一步确认了以上结论。这是由于在神经机器翻译模型中，并不是所有参数对精度的要求都很高，对精度要求较高的部分，我们保持单精度浮点数原有数据精度，对精度要求并不高的参数，我们可以将其转变为半精度浮点数来进行神经网络中的计算以提升模型的计算效率。

(2) 在分组卷积方法中，分组卷积组数group越大，对模型性能的影响就越大。

如表3.8和图3.8所示，在德英数据集和中英数据集上，无论采取何种参数配置，分组卷积组数group越大，对模型性能的破坏就越大。这是由于随着分组卷积中组数的增大，卷积中神经元的连接减少，虽然可以在一定程度上降低参数的冗余，但不可避免的会削减掉一些对于模型重要的参数或特征，从而对模型的性能产生影响。

(3) 在分组卷积方法中，encoder中越低层次使用分组卷积对模型性能影响越大，在decoder中越高的层次使用分组卷积方法对模型性能的影响越大。

如表3.9和图3.9所示，无论是在德英还是中英数据集上，encoder越低层对分组卷积中组数的变化越敏感，decoder越高层对分组卷积组数的变化越敏感。这是由于encoder中较低层距离对数据进行编码的词向量层越近，词向量层就相当于模型的输入部分，离词向量层越近的层对数据的传入也就越重要，削减其对模型性能的影响也就越大。而decoder越高层离概率分布输出层越近，包含的重要的模型特征也就越多，因此对于分组卷积中组数的变化也就越敏感。

3.3.3.2 模型效率

从实验结果来看，使用低精度运算方法可以在保证模型性能不变的情况下极大地提升模型效率，使用该方法，在德英数据集上采用transformer\_small配置可提升约39.84%的运算速度，采用transformer\_base配置可提升约52.34%的运算速度，在中英数据集上采用transformer\_base配置可提升约52.52%的运算速度，采用transformer\_big配置可提升约58.70%的运算速度。

由于分组卷积方法中encoder和decoder中不同层次对卷积组数group的敏感程度不同，我们充分利用该特性，对不同层次采取不同的分组卷积策略，可以在保证模型性能不变的前提下显著降低模型计算复杂度并达到1.46倍的模型压缩率。

3.4 本章小节

本章主要包括三部分内容：对神经机器翻译模型训练优化方法的研究动机、对研究方法的描述、实验结果以及结果分析。其中研究方法主要包括低精度运算加速方法和分组卷积结构加速方法。

首先我们了解了神经机器翻译的研究现状，当前虽然神经机器翻译模型的性能不断的得到了提升，但由于计算资源和存储资源受限问题，导致模型的训练过程受到了限制。本章分别从计算角度和模型结构角度对神经机器翻译模型的训练过程进行优化，在减小计算量的同时优化了模型结构。

首先我们从计算角度出发，采取低精度运算方法，在保证模型性能的前提下，减小单位数量张量操作的计算量，将神经机器翻译模型中部分操作由单精度浮点计算转换为半精度浮点计算，在不影响模型性能的前提下，可以达到39.84%-58.70%的模型训练加速效果。

然后我们从优化模型结构的角度出发，提出了一种基于分组卷积结构的模型训练加速方法，可以在保证模型性能不受影响的前提下降低模型参数量并降低模型计算复杂度，同时使模型结构变得更加科学合理；其次我们探索了神经机器翻译模型中不同层次对分组卷积的敏感程度，在encoder中越低层次使用分组卷积对模型性能影响越大，在decoder中越高的层次使用分组卷积方法对模型性能的影响越大，并利用以上结论显著降低了模型计算复杂度并达到1.46倍的模型压缩率。

本章是对神经机器翻译模型的训练过程进行优化，下一章是对神经机器翻译模型的解码过程进行优化，二者虽然都是对机器翻译模型进行优化，但优化的角度和阶段都不同。

第4章 神经机器翻译模型解码优化方法

4.1 研究动机

在机器翻译过程中，模型解码可以看作是与模型训练分离开的过程，是指模型训练结束后，用训练好的模型将原文翻译成译文的过程。相比于传统的统计机器翻译，神经机器翻译无论是从翻译的流畅性还是准确性而言，都取得了十足的进步。虽然神经机器翻译在性能上有了明显的提升，但由于网络中存在着大量的神经元连接，因此模型训练效率和解码效率都很低下。在机器翻译系统的实际应用中，相比于模型训练效率，实用系统对解码效率的要求会更高。因为实用系统往往需要实时的响应和较低的延迟，而且实际部署过程中的存储资源非常受限，为了使神经机器翻译能够顺利的部署和应用，在保证系统性能的前提下必须提高神经机器翻译的响应速度并降低模型所需存储资源。

目前在神经机器翻译模型中参数量数以亿计，导致模型规模十分庞大且解码效率也十分低下。解码过程和训练过程又有所不同，训练过程往往可以使用大量的计算资源，可以在高性能GPU或者GPU集群上进行计算，计算资源限制和计算效率要求并不严苛。但当翻译模型训练结束后，往往需要部署到资源受到严格制约的移动设备或嵌入式系统等离线小设备上，如何在保证模型性能的前提下减少模型解码过程中的计算代价使其能够部署在资源受限环境下就显得尤为重要。

为了解决神经机器翻译系统解码效率低下的问题，本文主要从以下两个方面入手：

1. 神经机器翻译模型存储压缩方法：为了解决神经机器翻译模型对存储资源的消耗问题，本文采取模型裁剪和参数低精度化方法来减少模型训练完成后所需存储大小，分别在基于自注意力机制的神经机器翻译模型和基于循环神经网络的机器翻译模型上进行实验，并详细分析了此种方法的优缺点。
2. 动态Beam Search解码加速方法：Beam Search搜索策略相较于贪婪的搜索策略，搜索空间更大，模型性能也往往更好。但如此大的搜索空间也使得模型的解码速度变得十分缓慢，为了提升解码效率，我们提出了一种动态Beam Search解码方法，根据句子长度动态的变化参数beam size的大小，并进一步分析了此种方法的优缺点。

4.2 方法描述

4.2.1 神经机器翻译模型存储压缩方法

本文采取了两种模型压缩策略来进行神经机器翻译任务上模型存储的压缩，一是模型裁剪方法，二是参数低精度化方法。

4.2.1.1 模型裁剪

模型裁剪[57]是指将神经网络模型中冗余的参数裁掉，以参数量减少的代价来缩减模型规模。当前，模型裁剪方法在图像识别等计算机视觉领域已经取得了不错的进展，有的研究工作可以在完全不损失模型性能的前提下大幅缩减模型规模[58]，但在机器翻译任务上应用较少，仍有待进一步的研究和探索。

本文使用了两种模型裁剪方法来减少机器翻译模型中的冗余参数：整体裁剪方法和分层裁剪方法。整体裁剪方法是指将整体模型结构中所有参数无差别对待，按整体模型参数比例进行裁剪，这种方法可以保证裁剪掉的模型参数一定是最接近0的，缺点是在模型参数分布不均时一定会对某一子层裁剪过多，从而影响模型的性能；分层裁剪方法是指将神经机器翻译模型中不同子层分别进行参数的裁剪，这种方法的优点是在每层之间的裁剪一定比较均匀，但若某一层参数均比较重要，则会对模型性能造成更大的损失。使用模型整体裁剪和分层裁剪方法后模型参数分布如图4.1所示，如图可以看出整体裁剪方法相较于分层裁剪所剪参数会更集中。

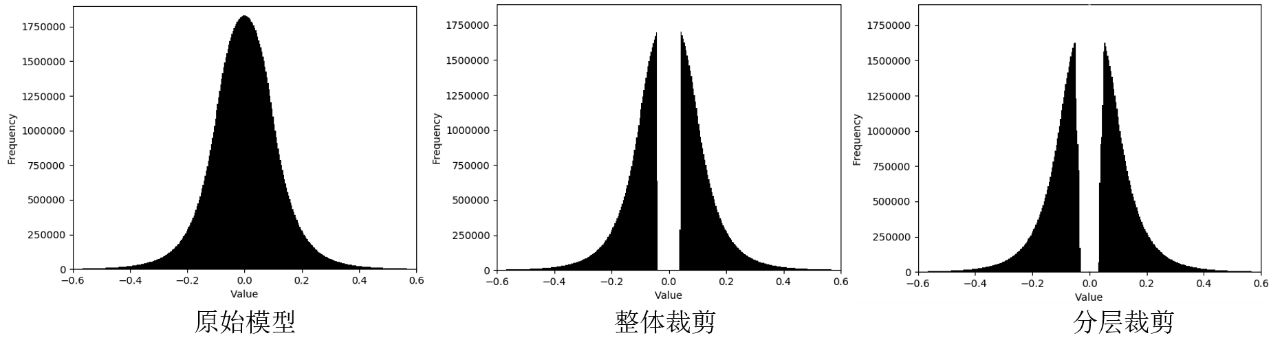


图4.1 整体裁剪和分层裁剪模型参数分布图

Fig. 4.1 Class-blind pruning and Class-uniform pruning

4.2.1.2 参数低精度化方法

在神经网络当中，参数都是用浮点数表示，参数低精度化方法实际上就是用更低精度的数值表示模型中参数，通过牺牲精度的方式减少每个模型参数存储所需要的比特位数，进而减小模型总体存储。在本文中，我们采取了两种参数低精度化方法，一是参数量化方法、二是尾数裁剪方法。

在量化方法中，考虑到神经网络参数总体呈正态分布，所选量化区间以0为中心比较合理。量化过程如图4.2左侧所示，图中每个颜色代表不同的量化区间，当进行n -bit量化时，量化区间会分为2n个等长区间，其中每个区间由0至2n-1的索引来表示。在量化过程中我们仅存储每个区间的索引值，而不存储模型实际值以减小模型存储大小，量化时仅对量化区间内的值进行索引值的映射，需将索引值恢复为原区间的值来减少模型解码性能上的损失，以图3左侧为例，解码也是参数的恢复过程。在量化的过程中，可以将网络的参数量化至固定的4-12 bits[59]，更极端的情况下也可以用1-2 bits来表示模型参数[60]。

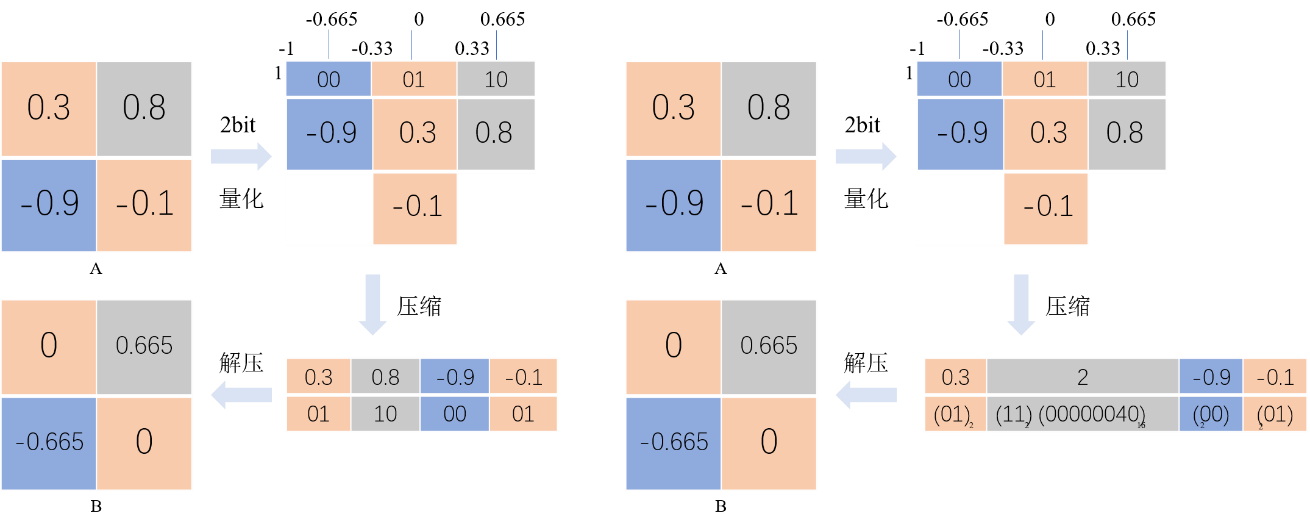


图4.2 量化过程图

Fig. 4.2 Quantization process

在量化方法的实际实现过程中，我们需要对量化区间内和量化区间外的值增加标志位进行区分，为了最大程度的保证存储优化率，进行n bits量化时，仅将量化区间分为2n-1个等长区间，用0至2n-2索引表示，用2n-1作为区分是否在量化区间外的标志，例如图4.2右侧所示的实际量化过程。图中展示了2bit量化，压缩，解压的过程。原始矩阵A是一个2\*2的矩阵。量化区间为[-1,1]，2bit把量化区间均分成三个子区间，[-1,-0.33)、[-0.33,0.33)和[0.33,1]，把A矩阵在量化区间内的权值装进对应区间中，分别用二进制00，01，10作为索引，11用来代表量化区间外的权值。对每个在量化区间里的值，我们仅存储对应子区间的索引，例如权值0.3，存储为01；对每个在量化区间外的值，先保存一个11，再按原值十六进制的00 00 00 40来存储。解压时遇到00，01，10则恢复成子区间中值（或左值，右值），例如01恢复成0；遇到11，则读后面的四个字节，恢复成十进制。例如，00000040恢复成2。

在量化方法中，量化区间大小也是一个非常值得研究的问题，如图4.3所示，在量化为相同bit的前提下，随着量化区间的增大，模型的压缩率虽然得到了提升，但与此同时模型性能的损失也随之增大，因此如何均衡考虑存储优化率和模型性能之间的关系是一个非常重要的问题。

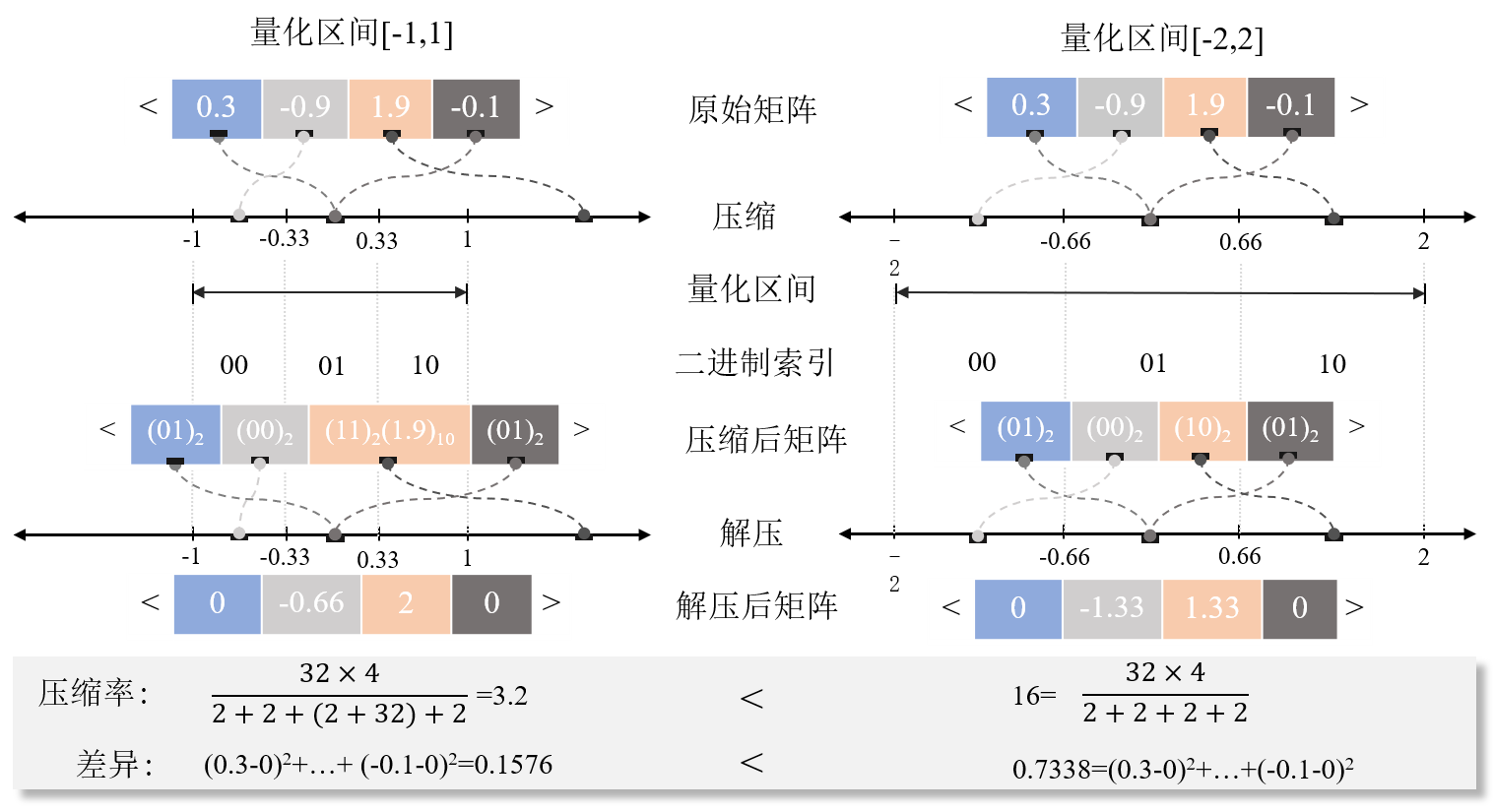


图4.3 不同量化区间量化过程图

Fig. 4.3 Quantization process with different interval

为了减少模型所需存储，我们还可以通过裁掉单精度浮点数尾数部分来进行模型的压缩。这种尾数裁剪方法中是将浮点数的尾数部分去掉16位，以此节省模型参数存储空间，在实际的实现过程中32 bit单精度浮点数会被裁剪成16 bit的浮点数来进行参数的压缩存储。

经过参数低精度化方法，模型以一个更低精度的形式来保存，有利于研究人员将其成功嵌入到小型离线设备上[61]。

4.2.2 动态Beam Search解码方法

机器翻译的解码过程实际上就是对最优答案的搜索过程，也就是搜索最可能的句子的过程，通常我们使用集束搜索（Beam Search）算法来解决这一问题。

首先我们来介绍贪心搜索（Greedy Search）过程，贪心搜索的思想非常简单，即在搜索答案的过程中，直接选择每个最大概率的词输出，直至出现终止标志符号或者其他的终止条件，贪心搜索过程如图4.4所示。

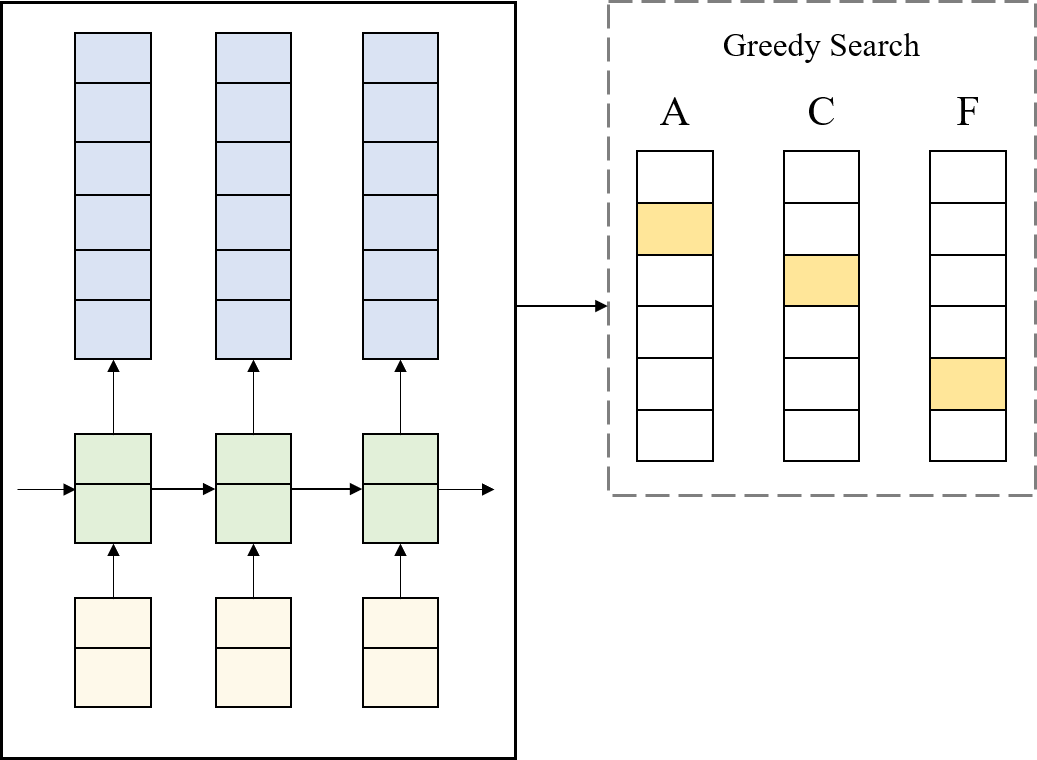


图4.4 贪心搜索过程图

Fig. 4.4 Greedy search process

而集束搜索与贪心搜索过程不同，这种方法在每一步的搜索过程不仅仅只保留一个词，而是保留概率最高的n个词，这个n我们通常称之为beam size，我们用beam size这个参数来限制每一步保留下来的可能的词的数量。通常采取集束搜索方法我们可以获得更好的准确性，在一定程度上避免了局部最优解对模型性能带来的负面作用。集束搜索方法通常比较耗时，在训练阶段并不会用到，而在解码过程中使用这种方法可以得到更好的解码答案，Beam Search解码过程如图4.5所示。

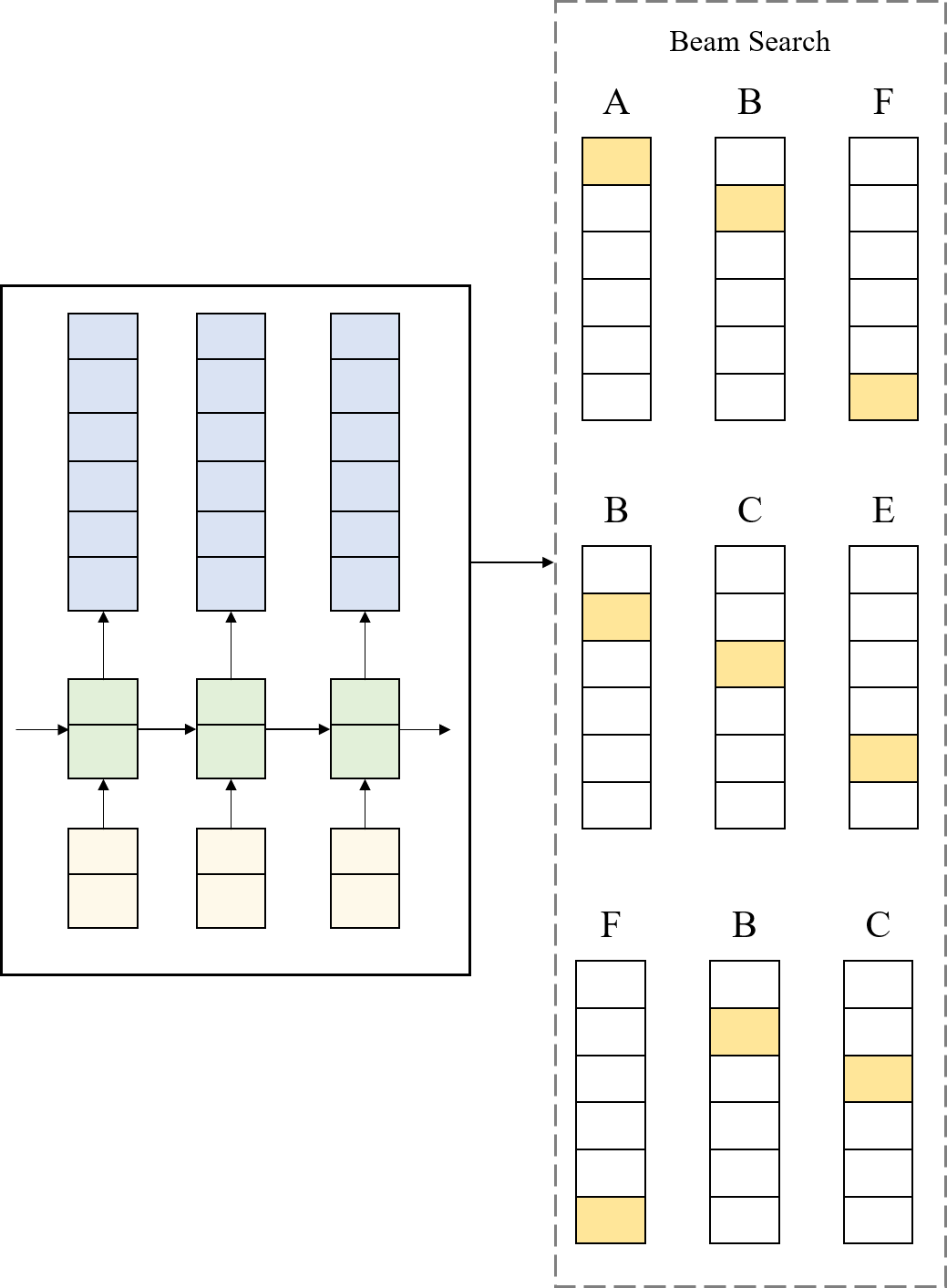


图4.5 集束搜索过程图

Fig. 4.5 Beam search process

在集束搜索过程中，假设我们的词表为[a,b,c,d]，我们的beam size选择为2，那么每步的搜索过程我们都保留概率最大的两个词，直至遇到终止标志符号或者其他终止条件为止。

在对神经机器翻译解码过程的研究中我们发现，Beam Search方法扩大了模型解码搜索空间，随着beam size的增大，模型的性能得到了显著的提升，但由于搜索空间较大，导致解码速度缓慢，在实际部署和应用的时候对计算资源和搜索时间造成了极大的浪费。在传统的Beam Search搜索方法中，无论对于较长句子还是较短句子，都采用统一的beam size大小，而我们发现，通常较短句子对beam size的要求并不高，在Beam Search搜索过程对所有长度的句子统一使用同一beam size对搜索资源造成了极大的浪费。

为了提升模型的解码效率，降低模型搜索代价，我们提出了一种动态Beam Search解码方法，可以根据解码句子长度自动调整解码所用的beam size，对于较短句子，我们采取较小的beam size，对于较长的句子，我们采取较大的beam size来保证模型的性能。通过动态Beam Search解码方法，在保证模型性能的同时，可以显著的提升模型的解码速度，使模型可以更轻易的部署在资源受限的离线小设备上。

4.3 实验

4.3.1 实验设置

本小节实验数据基于中英机器翻译任务，中英训练数据来自中英方向的NIST12 OpenMT，其中主要包含双语平行句对189.8w。本节中实验采用NIST2006(MT06)作测试集，所有长度超过50词的句子均被过滤掉。

对本文所采取的模型存储压缩方法，包括模型裁剪和参数低精度化方法，分别基于基于注意力机制的神经机器翻译模型Transformer和基于循环神经网络的机器翻译模型GNMT在中英翻译任务上进行实验，采取Transformer和GNMT两种模型是为了验证在不同结构模型中的存储优化效果。对于本文所采取的动态Beam Search解码方法，由于其本身与模型结构无关，本文主要基于Transformer模型进行验证。

关于训练设备，由于模型解码相比于模型训练对硬件的计算能力并不高，因此我们采取NVIDIA TITAN X来进行模型解码优化的研究，NVIDIA TITAN X设备的具体信息如表4.1所示。

表4.1 服务器配置信息

Table 4.1 Server configuration information

|  |  |
| --- | --- |
| **组件** | **配置** |
| 图形处理单元 | NVIDIA TITAN X |
| 中央处理器 | 12 Intel(R) Core(TM) i7-5930K CPU @ 3.50GHz |
| 物理CPU个数 | 1 |
| CPU核心数 | 6 |
| 逻辑CPU个数 | 12 |
| 内存 | 128GB |
| 显存 | 12GB |
| 硬盘 | 5TB |

对于Transformer系统和GNMT的循环神经网络RNN系统，所采用的模型参数如表4.2所示。

表4.2实验模型参数设置

Table 4.2 Model parameter configuration

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 参数 | | Transformer | | GNMT | |
| 层数 | | 6 | | 2 | |
| 隐层大小 | 512 | | 1024 | |
| batch大小 | 4096 | | 128 | |
| 学习率 | 0.001 | | 1.0 | |
| 头数 | 8 | | - | |
| 残差dropout | 0.1 | | 0.2 | |
| 优化器种类 | Adam | | SGD | |
| 更新轮数 | 15 | | 15 | |

4.3.2 实验结果

4.3.2.1 神经机器翻译模型存储压缩方法

本节主要对比模型裁剪和参数低精度化方法对Transformer和RNN性能的影响以及使用各种方法的组合方法在Transformer和RNN模型上能达到的最大压缩比例，其中参数低精度化方法主要包括训练后量化方法和尾数裁剪方法。基于以上方法，我们将在中英任务上进行实验，以MT06作为测试集，来进行各种模型存储压缩方法在Transformer和RNN上的对比实验。

本节实验基于Transformer和RNN两种机器翻译模型结构，采取整体剪枝和分层剪枝方法进行两种模型上的剪枝，在不损失模型性能的前提下，整体剪枝方法在Transformer和RNN上分别可以裁剪15%和40%的参数；分层剪枝方法在Transformer和RNN上分别可以裁剪15%和25%的参数。整体剪枝方法在RNN及Transformer上的实验结果如表4.3所示，剪枝比例为0性能代表基线系统性能。

表4.3 整体剪枝和分层剪枝对Transformer和RNN性能影响情况

Table 4.3 Class-blind pruning and Class-uniform pruning methods on Transformer and RNN

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **系统** | **剪枝比例** | | | | | | |
| **0%** | **15%** | **20%** | **25%** | **30%** | **35%** | **40%** |
| Transformer+整体剪枝 | 41.76 | -0.15 | -0.37 | -0.96 | -1.88 | -3.74 | -7.27 |
| RNN+整体剪枝 | 39.55 | -0.07 | -0.08 | -0.05 | -0.19 | -0.09 | -0.05 |
| Transformer+分层剪枝 | 41.76 | +0.18 | -0.24 | -0.48 | -1.94 | -2.92 | -5.35 |
| RNN+分层剪枝 | 39.55 | +0.06 | -0.03 | +0.15 | -0.13 | -0.13 | -0.45 |

在剪枝过程中，Transformer和RNN模型性能整体变化趋势如图4.4所示，从图中可以看出无论采取整体剪枝还是分层剪枝方法，随着剪枝比例的增大，模型性能逐渐下降，这是由于模型中连接数量的减少带来的性能损失。在Transformer和RNN在整体剪枝和分层剪枝方法下的性能对比实验中，无论何种剪枝方法，RNN的模型裁剪程度均大于Transformer，且当保持剪枝比例相同时，Transformer的性能损失明显大于RNN。

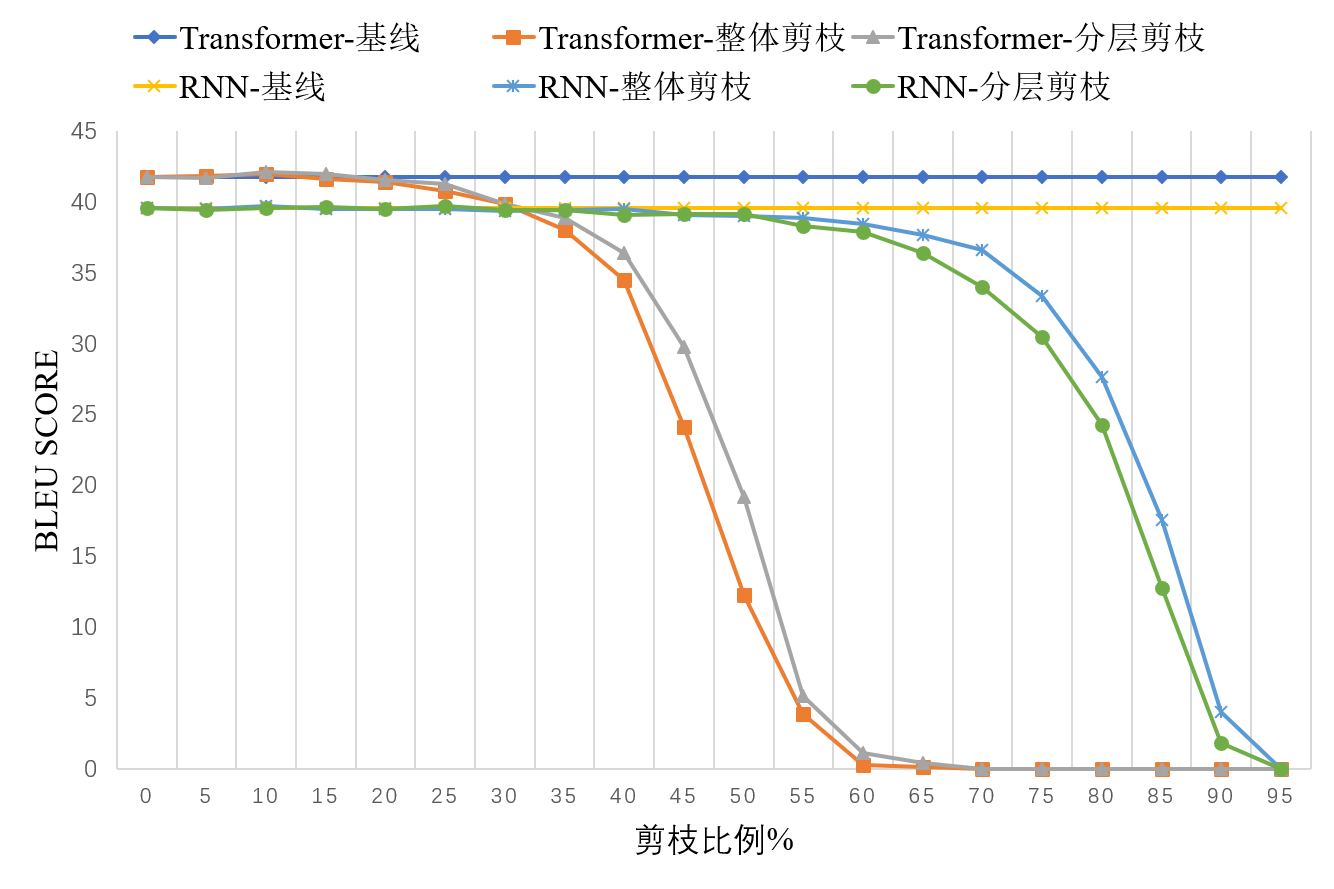


图4.4 Transformer和RNN在整体剪枝和分层剪枝方法下的性能对比

Fig. 4.4 Class-blind pruning and Class-uniform pruning methods on Transformer and RNN

从总体剪枝效果来看，在保证模型性能不变的前提下，RNN最高可以进行40%参数的模型裁剪，而Transformer中只能进行15%左右的模型参数的裁剪，此结果也从侧面反应出Transformer的模型设计更科学合理，Transformer模型的设计中冗余参数较少，相较于RNN来说更加高效。

为了实验Transformer和RNN的最大压缩比例，我们用模型裁剪方法和参数低精度化方法分别在两种模型上进行组合实验，组合实验结果如表4.4所示。

表4.4 模型压缩组合实验实验结果

Table 4.4 Model storage compression combined experiments

|  |  |  |  |
| --- | --- | --- | --- |
| **系统** | **压缩方法** | **BLEU** | **压缩比例** |
| RNN | baseline | 39.55 | 1× |
| RNN | +模型裁剪(40%) | -0.05 | 1.6× |
| RNN | +训练后量化(4-bit+98%) | -0.11 | 10.5× |
| RNN | +尾数裁剪 | -0.12 | 11.7× |
| Transformer | baseline | 41.76 | 1× |
| Transformer | +模型裁剪(15%) | +0.18 | 1.2× |
| Transformer | +训练后量化(6-bit+98%) | -0.16 | 5.6× |
| Transformer | +尾数裁剪 | -0.19 | 5.8× |

4.3.2.2 动态Beam Search解码方法研究

在Beam Search解码过程中，对模型性能和解码速度产生影响的一个重要的因素即beam size的大小，通常在一定范围内，随着beam size的增大，模型解码搜索空间变大，模型性能会得到提升，但模型的解码速度也会更为缓慢。为了探究beam size大小对模型性能及解码速度的影响，我们在同一模型和同一测试集上采取不同beam size大小进行实验，实验结果如表4.5所示。

为了验证动态Beam Search解码方法的正确性，我们在该模型上进一步进行实验。对于较短句子，我们采取beam size较小，对于较长的句子，我们采取解码的beam size较大。为了使实验更加的科学可比，我们以beam size为12的解码结果为我们的基线系统，在动态beam search解码方法当中，实验采取的beam size变化范围为6-12，我们将待解码句子长度划分为4个长度区间，对句长所属不同区间的句子采取不同的beam size大小，实验结果如表4.6所示。

表4.5 采取不同beam size对模型性能及解码速度影响

Table 4.5 Effects of different beam size on model performance and decoding speed

|  |  |  |
| --- | --- | --- |
| **beam size** | **解码速度** | **测试集性能** |
| 12 | 61.2 | 41.76 |
| 11 | 58.5 | -0.12 |
| 10 | 56.3 | -0.16 |
| 9 | 52.2 | -0.15 |
| 8 | 50.1 | -0.27 |
| 7 | 50.0 | -0.26 |
| 6 | 43.6 | -0.31 |
| 5 | 43.0 | -0.40 |

表4.6 动态Beam Search解码实验

Table 4.6 Experiment of dynamic beam search decoding method

|  |  |  |  |
| --- | --- | --- | --- |
| **系统** | **beam size** | **解码速度** | **测试集性能** |
| baseline | 12 | 61.2 | 41.76 |
| 动态Beam Search | 5-12 | 54.0 | +0.01 |

4.3.3 结果分析

(1) 在神经机器翻译模型中，神经网络中存在一定数量的冗余参数，适当的对模型参数量进行压缩对模型性能不会产生影响。

如表4.3和4.4所示，采取各种模型压缩方法的组合方法可以极大的压缩模型规模，而不会对模型性能产生影响。为了验证这一结论的普适性，我们分别在不同模型结构的Transformer和RNN网络上进行实验，可以在不影响模型性能的前提下，在RNN和Transformer模型结构中达到11.7和5.8倍的压缩效果。

(2) 相比于Transformer，RNN网络中存在的冗余参数更多。

如表4.3和表4.4所示，无论采取模型裁剪还是参数低精度化方法，RNN可压缩的程度都更大，这说明相比于RNN来说，Transformer的模型结构更为合理，模型中存在冗余也更小。

(3) 在Beam Search算法中对不同长度的句子采取不同的beam size大小，可以显著提升解码速度。

在对神经机器翻译解码过程的研究中我们发现beam size大小对模型解码速度和模型性能的影响非常大，随着beam size的增大，模型性能往往更好，模型的解码速度也更慢。而往往一些较长句子对beam size大小比较敏感，对于较短句子采取较小的beam size对模型性能并没有明显的影响，通过采取这种动态Beam Search解码方法，可以在不影响解码性能的前提下提升11.76%的解码速度。

4.4 本章小结

本章主要介绍了三部分内容：神经机器翻译解码优化的动机，对模型存储压缩方法和动态Beam Search解码方法这两种优化方法的描述，最终的实验结果及其分析。

首先我们介绍了进行神经机器翻译模型解码优化的动机，神经机器翻译模型的训练过程往往在集群或者高性能GPU和CPU条件下进行，计算资源和存储资源受限程度并不是很大，但神经机器翻译模型在实际应用的场景中往往需要部署在计算资源和存储资源及其受限的离线小设备上，这就使得神经机器翻译解码优化问题变得十分重要。

接下来我们介绍了模型存储压缩方法，我们采取了模型裁剪和参数低精度化方法，在RNN和Transformer上分别可以达到11.7和5.8倍的模型压缩率。

针对神经机器翻译模型解码过程采取Beam Search方法导致解码效率低下的问题，我们采取了一种动态Beam Search方法，可以在解码过程中根据不同的句长选择不同beam size大小进行解码，在不影响解码性能的前提下可提升11.76%的解码速度。

第5章 总结与展望

5.1 工作总结

在当今这个信息量爆炸的世界，深度学习技术随着信息技术的提升也获得了突飞猛进的发展，机器翻译作为其中的一项重要技术在性能上也获得了较大的提升，机器翻译技术对克服语言间障碍、加强不同语言间的交流起到了十分重要的意义。在机器翻译任务的发展过程中，出现了各种各样的机器翻译技术：基于规则的机器翻译、基于统计的机器翻译，直至现在最流行的神经机器翻译。神经机器翻译中，模型结构也在不断的进行改进：基于循环神经网络的机器翻译模型、基于卷积神经网络的机器翻译模型和基于自注意力机制的机器翻译模型等等。在以上发展过程中，机器翻译模型的结构变得越来越复杂，模型中参数量也越来越多，模型规模越来越庞大，模型训练解码的计算代价和存储代价也越来越高。本文主要围绕神经机器翻译模型的计算代价和存储代价高的问题，对神经机器翻译模型中的训练过程和解码过程进行优化，旨在不损失模型性能或对模型性能损害较小的前提下，提升模型的训练和解码效率，最终达到模型计算、存储和性能上的平衡。

通过分析得知，神经机器翻译系统虽然相较于传统的统计机器翻译系统在模型性能和翻译流畅度上都有了较明显的提升，但十分庞大的模型规模为神经机器翻译模型的计算和存储带来了巨大的困难。针对该问题，本文从模型训练和模型解码两个角度分别对神经机器翻译模型进行优化。在模型训练过程中采取了低精度计算的方式加速模型训练，并提出了一种基于分组卷积的自注意力机制神经机器翻译模型来加速模型训练过程并削减模型参数量。在模型解码过程中，我们采取模型存储压缩手段来压缩模型存储大小，并提出一种动态Beam Search解码方法，缩短Beam Search解码过程以达到加速模型解码的目的。

首先，对于神经机器翻译模型训练优化问题，我们从两个角度来解决：一是计算角度，也就是缩短单位计算量矩阵运算的计算强度；二是优化模型结构角度，缩减模型中冗余参数，使模型结构变得更加高效合理。从计算角度来考虑，低精度神经网络运算方法可以在保证模型性能不受影响的前提下极大地提高模型计算效率，利用半精度浮点数的计算优势，在不同规模的模型上此方法可以对模型训练过程达到39.84%至63.70%的加速效果。为了进一步优化神经机器翻译的模型结构，本文提出一种基于分组卷积的神经机器翻译模型训练加速手段，将网络中耗时的前馈神经网络替换为结构较稀疏的分组卷积网络，可以在缩减模型参数的同时降低计算复杂度，达到模型加速和模型压缩的效果，最终可以在显著降低模型计算复杂度的同时达到1.46倍的模型压缩效果。

此外，对于神经机器翻译模型解码优化问题，本文采取模型存储压缩方法来降低模型存储所需的存储空间来节省宝贵的存储资源，通过模型裁剪和参数低精度化的手段，在模型结构不同的RNN和Transformer上分别进行实验，在RNN和Transformer上通过组合方法可以分别达到11.7倍和5.8倍的压缩率而不对模型性能产生负面影响。为了优化模型解码效率，我们研究并改进了传统的Beam Search搜索方法，根据不同的解码句长采取不同的beam size大小来减小解码搜索空间，最终可以提升11.76%的解码速度。

本文所提出的所有方法都是基于神经机器翻译模型优化问题，分别从训练和解码角度进行考虑，优化模型速度以及模型存储空间，减小模型所需存储并提升模型的计算效率。

5.2 创新点分析

本文主要有三大创新点：第一，针对神经机器翻译训练优化的目的，本文采取了低精度张量来进行神经网络的训练并创新性的将分组卷积神经网络结构引入到机器翻译任务中，通过将神经机器翻译模型中的前馈神经网络替换为较稀疏的分组卷积来达到模型加速和模型压缩的目的。通过实验我们发现，在神经机器翻译任务中将全连接的前馈神经网络替换为较稀疏的分组卷积结构是可行的，此结论进一步证明了神经网络的冗余特性。第二，针对神经机器翻译解码优化任务，本文针对传统的基于循环神经网络的机器翻译系统和基于自注意力机制的神经机器翻译系统分别进行模型存储压缩实验，以探索不同结构的神经机器翻译模型可以压缩的最大比例，实验结果证明，基于循环神经网络的机器翻译系统可压缩的比例更大，这进一步证明了循环神经网络相较于基于自注意力机制的神经机器翻译系统有更大的冗余空间。第三，本文提出一种动态的Beam Search解码方法，即根据不同句子长度动态的调节beam size大小，以减少Beam Search搜索空间过大导致的时间消耗。

5.3 未来工作

本文中研究了基于低精度运算的神经机器翻译模型加速方法和基于模型优化的神经机器翻译模型加速方法，从以上两种角度对神经机器翻译的模型训练过程进行加速并取得了不错的效果。其中，在对基于低精度运算的神经机器翻译模型加速方法的研究中，本文采取了一种混合精度计算的方法，模型参数部分采取半精度浮点数，另一部分采取单精度浮点数，在提升模型训练速度的前提下尽量减少模型性能的损失。虽然此种方法可以显著提升了模型的训练速度，但由于模型中部分结构对精度变化比较敏感，无法将所有模型中计算全部转换成半精度浮点计算，并没有完全发挥低精度网络的计算优势。为了解决以上问题，本文从以下两个角度对未来工作进行了展望。

5.3.1 低精度神经网络反向传播方法

神经机器翻译模型的训练过程主要分为两大步骤，前向计算和反向更新过程，其中反向更新部分对参数精度的损失十分敏感，一部分原因是反向计算梯度的数值往往数量级较小，位数的损失对其影响较大，另一部分原因是神经网络参数训练过程非常依赖反向更新过程，反向更新过程影响了参数如何变化，因此在反向传播过程如何充分利用低精度浮点计算的优势就成为了一个十分值得研究的问题。我们可以把低精度的反向传播过程看作是一个离散优化问题，这样可以把梯度由连续分布转换为离散分布，与此同时尽量不损失模型性能。

5.3.2 更低精度神经机器翻译模型训练方法

本文中采取了半精度浮点和单精度浮点的混合精度计算方法，半精度浮点计算虽然比单精度浮点的计算速度快，但半精度浮点仍占有16bit，模型加速效果有限，因此如何利用更低精度的数值进行神经网络的训练仍是一个值得研究的问题，例如采取更低精度浮点数甚至8bit整数。但随着模型参数表示位数的减小，模型性能不可避免的会受到影响，因此如何在不损失模型性能的前提下使用更低精度参数进行模型训练很值得我们进一步研究。

参考文献

1. Mayor A, Alegria I, De Ilarraza A D, et al. Matxin, an open-source rule-based machine translation system for Basque[J]. Machine translation, 2011, 25(1): 53.
2. Dugast L, Senellart J, Koehn P. Statistical post-editing on SYSTRAN's rule-based translation system[C] // Proceedings of the Second Workshop on Statistical Machine Translation. Association for Computational Linguistics, 2007: 220-223.
3. Makoto Nagao (1989). Machine Translation: How Far Can it Go? Oxford University Press, Oxford, U.K. Translated by Norman Cook.
4. 戴新宇, 尹存燕, 陈家骏, et al. 机器翻译研究现状与展望[J]. 计算机科学, 2004, 31(11):176-179.
5. Xiao T, Zhu J, Zhang H, et al. NiuTrans: an open source toolkit for phrase-based and syntax-based machine translation[C]//Proceedings of the ACL 2012 System Demonstrations. Association for Computational Linguistics, 2012: 19-24.
6. 刘群. 基于句法的统计机器翻译模型与方法[J]. 中文信息学报, 2011, 25(6):63-71.
7. 肖桐. 树到树统计机器翻译优化学习及解码方法研究[D]. 东北大学, 2012.
8. Wu Y, Schuster M, Chen Z, et al. Google's neural machine translation system: Bridging the gap between human and machine translation[J]. arXiv preprint arXiv:1609.08144, 2016.
9. Sutskever I, Vinyals O, Le Q V. Sequence to sequence learning with neural networks[C]//Advances in neural information processing systems. 2014: 3104-3112.
10. Bahdanau D, Cho K, Bengio Y. Neural machine translation by jointly learning to align and translate[J]. arXiv preprint arXiv:1409.0473, 2014.
11. Kalchbrenner N, Blunsom P. Recurrent continuous translation models[C]//Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing. 2013: 1700-1709.
12. Gehring J, Auli M, Grangier D, et al. Convolutional Sequence to Sequence Learning[J]. international conference on machine learning, 2017: 1243-1252.
13. Vaswani A, Shazeer N, Parmar N, et al. Attention is All you Need[J]. neural information processing systems, 2017: 5998-6008.
14. Lecun Y, Bengio Y, Hinton G E, et al. Deep learning[J]. Nature, 2015, 521(7553): 436-444.
15. 刘洋 . 神经机器翻译前沿进展 [J]. 计算机研究与发展 计算机研究与发展 , 2017, 54(6): 1144-1149.
16. Hutchins W J . Machine Translation: Past, Present, Future[J]. Modern Language Journal, 1986, 71(4):437.
17. Denil M, Shakibi B, Ranzato M, et al. Predicting parameters in deep learning[C]// International Conference on Neural Information Processing Systems. Curran Associates Inc. 2013:2148-2156.
18. Cheng Y, Wang D, Zhou P, et al. A Survey of Model Compression and Acceleration for Deep Neural Networks.[J]. arXiv: Learning, 2017.
19. Howard A G, Zhu M, Chen B, et al. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications[J]. 2017.
20. Courbariaux M, Hubara I, Soudry D, et al. Binarized Neural Networks: Training Deep Neural Networks with Weights and Activations Constrained to +1 or -1[J]. 2016.
21. Rastegari M , Ordonez V , Redmon J , et al. XNOR-Net: ImageNet Classification Using Binary Convolutional Neural Networks[J]. 2016.
22. Mikolov T, Karafiat M, Burget L, et al. Recurrent neural network based language model[C]. conference of the international speech communication association, 2010: 1045-1048.
23. Zhang B, Xiong D, Su J, et al. Accelerating Neural Transformer via an Average Attention Network[J]. meeting of the association for computational linguistics, 2018: 1789-1798.
24. Luong T, Pham H, Manning C D, et al. Effective Approaches to Attention-based Neural Machine Translation[J]. empirical methods in natural language processing, 2015: 1412-1421.
25. Krizhevsky A, Sutskever I, Hinton G E. ImageNet classification with deep convolutional neural networks[C]// International Conference on Neural Information Processing Systems. Curran Associates Inc. 2012:1097-1105.
26. R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 580–587, 2014.
27. Johnson J, Karpathy A, Li F F. DenseCap: Fully Convolutional Localization Networks for Dense Captioning[C]// Computer Vision and Pattern Recognition. IEEE, 2016:4565-4574.
28. Zhang J, Zong C. Deep Neural Networks in Machine Translation: An Overview[J]. IEEE Intelligent Systems, 2015, 30(5):16-25.
29. Hansen L K, Salamon P. Neural network ensembles[J]. IEEE Transactions on Pattern Analysis & Machine Intelligence, 2002, 12(10):993-1001.
30. Hu H, Peng R, Tai Y W, et al. Network Trimming: A Data-Driven Neuron Pruning Approach towards Efficient Deep Architectures[J]. 2016.
31. Han S, Mao H, Dally W J. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding[J]. Fiber, 2015, 56(4):3--7.
32. He K, Zhang X, Ren S, et al. Deep Residual Learning for Image Recognition[J]. 2015:770-778.
33. Lecun Y, Bottou L, Bengio Y, et al. Gradient-based learning applied to document recognition[J]. Proceedings of the IEEE, 1998, 86(11):2278-2324.
34. Krizhevsky A, Sutskever I, Hinton G E. ImageNet classification with deep convolutional neural networks[C]// International Conference on Neural Information Processing Systems. Curran Associates Inc. 2012:1097-1105.
35. K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” CoRR, vol. abs/1409.1556, 2014.
36. Y. Gong, L. Liu, M. Yang, and L. D. Bourdev, “Compressing deep convolutional networks using vector quantization,” CoRR, vol. abs/1412.6115, 2014.
37. Courbariaux M, Bengio Y, David J P. BinaryConnect: Training Deep Neural Networks with binary weights during propagations[J]. 2015:3123-3131.
38. M. Courbariaux and Y. Bengio, “Binarynet: Training deep neural networks with weights and activations constrained to +1 or -1,” CoRR, vol. abs/1602.02830, 2016.
39. M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi, “Xnor-net: Imagenet classification using binary convolutional neural networks,” in ECCV, 2016.
40. Yang T J, Chen Y H, Sze V. Designing Energy-Efficient Convolutional Neural Networks Using Energy-Aware Pruning[J]. 2016:6071-6079.
41. Anwar S, Sung W. Coarse Pruning of Convolutional Neural Networks with Random Masks[J]. 2016.
42. Sun Y, Wang X, Tang X. Sparsifying Neural Network Connections for Face Recognition[C]// Computer Vision and Pattern Recognition. IEEE, 2016:4856-4864.
43. Molchanov P, Tyree S, Karras T, et al. Pruning Convolutional Neural Networks for Resource Efficient Inference[J]. 2016.
44. Yang Y, Huang L, Ma M, et al. Breaking the Beam Search Curse: A Study of (Re-)Scoring Methods and Stopping Criteria for Neural Machine Translation[J]. empirical methods in natural language processing, 2018: 3054-3059.
45. Fiesler E, Choudry A, Caulfield H J. Weight discretization paradigm for optical neural networks[J]. Proc. SPIE-Int. Soc. Opt. Eng. (USA), 1990, 1990, 1281:164--173.
46. Marchesi M, Orlandi G, Piazza F, et al. Fast neural networks without multipliers[J]. IEEE Transactions on Neural Networks, 1993, 4(1):53-62.
47. Vanhoucke V, Senior A, Mao M Z, et al. Improving the speed of neural networks on CPUs[C]// Deep Learning and Unsupervised Feature Learning Workshop, NIPS. 2011.
48. Devlin J. Sharp Models on Dull Hardware: Fast and Accurate Neural Machine Translation Decoding on the CPU[J]. 2017.
49. Quinn J, Ballesteros M. Pieces of Eight: 8-bit Neural Machine Translation[J]. north american chapter of the association for computational linguistics, 2018: 114-120.
50. See A, Luong M, Manning C D, et al. Compression of Neural Machine Translation Models via Pruning.[J]. conference on computational natural language learning, 2016: 291-301.
51. Socha K, Blum C. An ant colony optimization algorithm for continuous optimization: application to feed-forward neural network training[J]. Neural Computing & Applications, 2007, 16(3):235-247.
52. Meng F, Lu Z, Wang M, et al. Encoding Source Language with Convolutional Neural Network for Machine Translation[J]. Computer Science, 2015.
53. Papineni K, Roukos S, Ward T, et al. Bleu: a Method for Automatic Evaluation of Machine Translation.[C]// Proc Meeting of the Association for Computational Linguistics. 2002.
54. Britz D, Goldie A, Luong M, et al. Massive Exploration of Neural Machine Translation Architectures[J]. empirical methods in natural language processing, 2017: 1442-1451.
55. Sennrich R, Haddow B, Birch A, et al. Neural Machine Translation of Rare Words with Subword Units[J]. meeting of the association for computational linguistics, 2016: 1715-1725.
56. Kingma D P, Ba J. Adam: A Method for Stochastic Optimization[J]. international conference on learning representations, 2015.
57. Cun Y L, Denker J S, Solla S A. Optimal brain dam-age[C]// International Conference on Neural Information Processing Systems. MIT Press, 1989:598-605.
58. Han S, Mao H, Dally W J. Deep Compression: Compress-ing Deep Neural Networks with Pruning, Trained Quanti-zation and Huffman Coding[J]. Fiber, 2015, 56(4):3--7.
59. Lin D D, Talathi S S, Annapureddy V S. Fixed Point Quantization of Deep Convolutional Networks[J]. Computer Science, 2016.
60. Hubara I, Soudry D, Yaniv R E. Binarized Neural Networks[J]. 2016.
61. Mishra A, Marr D. Apprentice: Using Knowledge Distillation Techniques To Improve Low-Precision Network Accuracy[J]. 2017

致谢

转眼间我的研究生生活已接近尾声，回首这两年的学习和生活历程，我的心中充满了感激之情。我很荣幸能加入自然语言处理实验室这个十分温暖的大家庭，在此毕业之际，向实验室的所有老师、同学们表示衷心的感谢。

首先我想感谢我的导师肖桐老师，您在学术上认真严谨，在生活中待人热情、幽默风趣，是我的良师益友。您告诉了我做研究要一丝不苟，不能急于求成，要看清自己的优缺点，要有合理的计划和预期才能做好研究。在生活中，您能够理解和包容我的不足，在我迷茫的时候，您会站在我的角度思考问题并给予我非常中肯的建议。从您身上学到了很多学习经验和为人处事的道理，非常感谢您！

然后我要感激实验室的负责人朱靖波老师，您不仅教会了我许多专业知识，还教会了我许多宝贵的品质。是您教会了我要有梦想有志向，要为了自己的梦想努力奋斗。您为实验室付出了心血，一步步带着实验室的团队发展壮大，您的很多建议在我今后的生活和学习中都有着非常大的指导意义，再次感谢您！

还有我要感谢实验室的其他老师们，是你们让实验室有了家一样的感觉，让我们快乐的在这个温暖的集体中成长。老师们不仅在学习上对我们认真指导，在生活中也给予了我们悉心的帮助，对待我们就像对待自己的孩子一样，谢谢各位老师！

还要感谢实验室的各位同学，你们在学习和生活上都给予我们非常大的帮助，你们在我沮丧时给我安慰，在我高兴时能够分享我的喜悦。感谢各位学长、学姐、学弟、学妹们，因为有你们的存在，实验室才越来越好，谢谢大家！

最后我要感谢我的父母，是你们的爱与关怀伴随着我长大，你们用无微不至的关心和鼓励支撑着我的前行，用包容和理解让我自由的发展。小时候你们用温暖的臂膀为我遮风挡雨，长大后我会用自己的行动来回报你们的养育之恩！

硕士期间参加的科研项目

科研项目：

1. 参与国家自然科学基金面上项目“面向神经机器翻译的结构学习方法研究”。

硕士期间取得的学术成果

学术成果：

1. 于2019年1月在《中文信息学报》上以第一作者发表《面向神经机器翻译的模型存储压缩方法分析》论文一篇。