分类号 密级

UDC

学 位 论 文

基于神经机器翻译的**壮汉翻译系统的研究与实现**

|  |  |  |  |
| --- | --- | --- | --- |
| 作者姓名： | 张哲旸 | | |
| 指导教师： | 朱靖波 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11月 | 论文答辩日期： | 2019年11月 |
| 学位授予日期： | 2019年11月 | 答辩委员会席： | 王宝库 |
| 评阅人： | 张俐 | | |

东 北 大 学

2019年11月

##### A Thesis in Computer Application Technology

**Research and Implementation of Neural Machine Translation Based Zhuang-Chinese**

**Translation system**

By Zhang Zheyang

Supervisor: Professor Zhu Jingbo

**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，不包含其他人已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已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

机器翻译是将一种自然语言转换为另一种自然语言的过程。机器翻译发展从上世纪五十年代起，先后经历了基于规则的机器翻译、基于统计的机器翻译到目前被学术界以及工业界大量使用的基于神经网络的神经机器翻译。神经机器翻译是使用标注的数据训练机器翻译模型，然后使用训练好的神经机器翻译模型讲一种自然语言翻译为另一种自然语言。神经机器翻译语传统的基于规则的机器翻译、基于统计的机器翻译相比，具有----的优点。

~~虽然当前神经机器翻译系统的翻译质量已经很高，极大的方便了客户的使用，但是当前学术界和工业界的主要研究对象都是面向汉语、英语、德语、俄语等语言之间的相互翻译。然而我国有55个少数民族，其中很多少数民族拥有自己的语音和文字，所以研究我国少数民族语言语汉语之间的~~

神经机器翻译的发展极大地方便了人们的日常翻译使用。不过当前学术界和工业界的主要研究对象都是面向汉语、英语、德语、俄语等不同国家的语言之间的相互翻译。然而我国是一个多民族国家，其中很多少数民族拥有自己民族的语言和文字，因此研究汉语与少数民族语言之间的神经翻译系统有着重要的意义。本文主要研究壮语到汉语的翻译工作，实现壮汉神经机器翻译系统。

本论文旨在研究在壮汉翻译场景如何在利用较少的计算资源前提下提升翻译质量，提出两个方法，一是如何使用数据加工的方式提高翻译质量，二是使用知识精炼的训练方法减少计算资源的占用并提高翻译质量。

在大多数情况下，机器翻译的性能与数据的选取息息相关。训练一个翻译质量优秀的机器翻译系统通常需要大量--- 然而壮语与汉语的双语数据数量相对较少，壮语与英文一样使用拉丁字母拼写，在网上获取的壮汉数据中包含大量噪声，

本文使用数种数据

本论文在互联网上人工获取数据构建壮汉翻译测试集，测试集共1500句，其中80%为新闻报告相关，20%为口语相关数据。。

**关键词：**机器翻译；稀缺资源；神经网络；预训练

Abstract

Machine translation is the technology of translating one language into another one by computer. With the development of neural networks, machine translation has evolved from statistical-based method to neural network-based method, a.k.a., neural machine translation, and the translation results are more accurate and fluent than before. Neural machine translation is modeled based on the encoder-decoder framework. The encoder extracts the information from the source sentence and encodes it into a vector representation. The decoder decodes the vector representation into the target sentence. The model learns in an end-to-end manner without manual intervention.

The translation performance of neural machine translation model has been significantly improved by a large amount of bitext. However, there are some low resource scenarios in machine translation tasks, such as minority language and specific domain. In these scenarios, simply deploying neural machine translation model cannot achieve the desired performance. At the same time, because of the high cost of manually labeling data, its application is constrained. Therefore, low resources machine translation has attracted wide attention.

This thesis studies how to use the pre-training method to improve translation performance in the low resource scenario and proposes two methods, one pre-trains the encoder and the other one pre-trains both encoder and decoder.

Aiming at the existing pre-trained model, this thesis proposes three methods to exploit the encoder-based pre-trained model, whose word embedding, hidden layer feature and the whole encoder are integrated respectively. The three methods integrate the pre-trained model into the encoder of neural machine translation model from different perspectives, enhancing the model capacity. Experimental results show that all three methods can significantly improve the performance of the model by nearly two BLEU points in IWSLT German-English speaking dataset.

To suit machine translation tasks better, this thesis proposes a new encoder-decoder-based pre-training method, which addresses the issues of the previous encoder-based pre-training method through the document-level text generation task. Starting from this new pre-trained model, the neural machine translation model possesses the ability of text generation and cross-language modeling in the initial stage and is able to achieve better performance via fine-tuning on the small bilingual corpus. Experiments show that the proposed method can greatly improve the performance of the translation model in the low resource scenario. On three WMT18 news data sets consisted of 10,000, 50,000 and 100,000 randomly selected sentence-pairs, we improve the BLEU points of English-German direction by 9.81, 6.53 and 3.91 respectively, and improve the BLEU points of German-English direction by 13.49, 6.99 and 4.54 respectively.

**Key words:** machine translation; low resource; neural network; pre-training

目 录

[独创性声明 I](#_Toc13062040)

[摘 要 II](#_Toc13062041)

[Abstract III](#_Toc13062042)

[第1章 绪 论 1](#_Toc13062043)

[1.1 研究背景 1](#_Toc13062044)

[1.2 研究内容 2](#_Toc13062045)

[1.3 论文组织结构 3](#_Toc13062046)

[第2章 相关技术概述 5](#_Toc13062047)

[2.1 机器翻译发展介绍 5](#_Toc13062048)

[2.2 神经机器翻译 7](#_Toc13062049)

[2.1.1 编码器-解码器框架 7](#_Toc13062050)

[2.1.2 注意力机制 8](#_Toc13062051)

[2.1.3 基于自注意力机制的神经机器翻译模型 13](#_Toc13062052)

[2.3 预训练方法发展介绍 17](#_Toc13062053)

[2.3.1 基于词嵌入的预训练方法 17](#_Toc13062054)

[2.3.2基于语言模型的预训练方法 20](#_Toc13062055)

[2.4 本章小结 22](#_Toc13062056)

[第3章 基于编码器的预训练 25](#_Toc13062057)

[3.1 研究动机 25](#_Toc13062058)

[3.2 方法描述 27](#_Toc13062059)

[3.2.1 基于词嵌入的集成方法 27](#_Toc13062060)

[3.2.2 基于隐藏层特征的集成方法 28](#_Toc13062061)

[3.2.3 基于编码器微调的集成方法 29](#_Toc13062062)

[3.3 实验 31](#_Toc13062063)

[3.3.1 实验设置 31](#_Toc13062064)

[3.3.2 实验结果和分析 33](#_Toc13062065)

[3.4 性能分析 37](#_Toc13062066)

[3.4.1 预训练模型对模型收敛速度的影响 38](#_Toc13062067)

[3.4.2 预训练模型对模型鲁棒性的影响 38](#_Toc13062068)

[3.4.3 三种预训练模型集成方法的对比 39](#_Toc13062069)

[3.5 本章小结 40](#_Toc13062070)

[第4章 基于编码器-解码器的预训练 41](#_Toc13062071)

[4.1 研究动机 41](#_Toc13062072)

[4.2 方法描述 41](#_Toc13062073)

[4.3 实验 44](#_Toc13062074)

[4.3.1 实验设置 44](#_Toc13062075)

[4.3.2 实验结果 47](#_Toc13062076)

[4.3.3 实验分析 49](#_Toc13062077)

[4.4 本章小结 50](#_Toc13062078)

[第5章 总结与展望 53](#_Toc13062079)

[5.1 工作总结 53](#_Toc13062080)

[5.2 创新点分析 54](#_Toc13062081)

[5.3 未来工作 54](#_Toc13062082)

[参考文献 57](#_Toc13062083)

[致谢 61](#_Toc13062084)

[硕士期间参加的科研项目 63](#_Toc13062085)

第1章 绪 论

1.1 研究背景

机器翻译（ Machine Translation ，简写为 MT ）属于计算机语言学的范畴，它是计算机语言学的一个分支，机器翻译是人工智能的最终目标之一，并且具有重要的科学研究价值以及工程应用前景。所谓机器翻译，就是研究如何利用计算机软件将文本从一种自然语言翻译成另一种自然语言，这个过程也被称之为从源语言翻译为目标语音。随着现代社会的不断发展，世界经济正向着全球化的方向迅猛进程，各地区间进行交流的需求不断增加，而语言的隔阂是各地区间进行交流的最大障碍，因此机器翻译的需求日益增加，机器翻译在人们的生活扮演着越来越重要的角色。尤其是近几年来，基于人工神经网络的神经机器翻译的出现使机器翻译性能大幅提高，《一带一路》《机器翻译公司雨后春笋》

近几十年来，随着信息技术的不断提高，机器翻译技术也不断提升，期间先后提出了各种机器翻译技术。这些机器翻译技术大致可分为以下几类：基于规则的机器翻译（Rule-Based  Machine Translation，简写为RBMT）、基于实例的机器翻译（Example-Based  Machine Translation，简写为EBMT）、基于统计的机器翻译（Statistics  Machine Translation，简写为SMT）、基于人工神经网络的机器翻译（Neural   Machine Translation，简写为NMT），其中基于实例的机器翻译、统计机器翻译、神经网络机器翻译都属于基于语料库的机器翻译（Corpus-Based  Machine Translation，简写为CBMT）。在二十世纪九十年代以前机器翻译的研究主要集中在基于规则的机器翻译，这种机器翻译技术需要重点研究语法、句法等语言学知识，受限于人工书写的规则的覆盖度，极大的依赖于语言学家的知识与经验。规则的获取难度大、人工书写的规则覆盖度有限、不同规则之间相互冲突等等问题导致基于规则的机器翻译在大部分情况下可用性很低。

在二十世纪九十年代，科学家提出完全基于统计方法的机器翻译，这一方法完全抛弃了对语言学家的知识、经验的依赖，把机器翻译看作是一个信息传输的过程，用信息论中的噪声信道模型对机器翻译进行解释。统计机器翻译方法建立概率模型求解机器翻译问题，统计机器翻译的发展极大提高了机器翻译的可用性。相比于传统的基于规则和统计的机器翻译，神经机器翻译使用神经网络对机器翻译任务建模，使用深度神经网络获取源语言与目标语音直接的映射关系。神经机器翻译的翻译性能相比传统方法的到极大的提高，逐渐应用到新闻、金融、军事等各种场景。

壮族是我国人数最多的少数民族，壮族人口有1700万人，壮族人民主要分布在广西壮族自治区，在广东省、云南省、贵州省也有很多壮族人民聚居。在壮族聚集地很大一部分壮族人民间日常交流使用壮族的民族语言壮语，因此开发壮汉机器翻译系统是十分有必要的。但是国内的高校、互联网公司却很少有针对壮语进行研究、开发系统。当前互联网企业主要使用神经机器翻译作为服务客户线上系统，而神经机器翻译系统的训练需要大量高质量的、双语对照的训练语料。例如中英、英德等机器翻译系统的线上模型往往使用上千万甚至上亿的双语平行语料进行训练，然而很多语种之间的语料数据相对较少，通过人工标注的方式获取双语料数据耗时耗力，通过网络爬虫爬取的互联网上的双语语料往往混杂其他语种，导致质量很差难以使用，这就导致这些语种之间训练神经机器翻译系统比较困难，往往性能不佳，这些平行语料较少的情况一般被称为稀缺资源问题，壮汉翻译就属于这一稀缺资源问题。

此外，我国有五十五个民族，不同民族使用不同的文字共有54种文字，在国家“一带一路”大战略背景下，机器翻译服务的语种大幅增加，一带一路涉及65个沿线国家和地区、60余种官方语言、200余种民族语言及方言。企业提供线上机器翻译服务的神经机器翻译系统往往需要针对不同语种组合单独训练，而神经机器翻译系统需要大量计算资源支持，如何使用更少的计算资源提高机器翻译质量也变成了一个亟待解决的问题。

因此如何在已有平行语料数量较少或者质量较差的情况下，提高壮汉机器翻译质量，并使用较少量计算资源将其部署，具有重要的研究意义和商业价值。

1.2 研究内容

随着信息化社会的不断发展，现代计算机的计算性能迅猛提高，其速度甚至可以达到每秒数十千万亿次，这使得计算机在处理数值运算以及逻辑运算方面更加高效。计算机的发展为人们在科学计算、信息处理、过程控制等领域提供了大量自动化的手段。随着计算性能的迅猛提高以及研究人员对人工神经网络的不断改进，人工神经网络技术也获得了长足的发展[14]。近几年来，研究人员将人工神经网络应用于自然语言处理领域里，使得自然语言处理领域的很多子任务性能都得到了显著的提高，神经机器翻译就其中的代表。

传统的神经机器翻译主要使用基于循环的神经网络（Recurrent Neural Network, 简写为RNN）训练神经机器翻译，然而循环网络的机器翻译模型的输入受到很大的限制，循环神经网络计算当前状态时必须依赖于上一时刻，这导致循环神经网络必须必须按照时序进行输入，这就导致循环神经网络训练和解码都十分耗时，而且由于长距离依赖问题的存在，基于循环神经网络的机器翻译在处理长句子时候效果一般。随着神经网络技术的不断发展，基于自注意力（Self-attention）机制的神经网络在众多网络结构中脱颖而出。基于自注意力机制的神经网络凭借极短信息传递距离、可并行化的优秀设计，具有优秀的翻译性能以及较为快速的训练、解码速度，在机器翻译等多项任务上取得了最优秀的性能水平。故本课题使用基于自注意力机制的神经机器翻译系统进行壮语-汉语神经机器翻译系统的研究与实现工作。

虽然基于自注意力机制的神经机器翻译极大地提高了机器翻译的性能，相比较传统方法提高了训练、解码速度，但是神经机器翻译翻译的固有问题仍然存在：神经机器翻译高度依赖于高质量的语料并且对计算资源需求巨大。为了在数据资源匮乏的情况下提高模型性能，并减少其在部署时候消耗的计算资源，本课题使用以下两个方面进行课题研究：

（1）数据处理与增强方法：神经机器翻译的关键在于数据，机器翻译的训练是向模型中送入双语平行语料数据，在机器翻译的解码过程中，源语言文本数据在经过神经机器翻译模型后被转换为目标语音文本数据。本课题针对壮语、汉语的语言特点，结合壮-汉机器翻译训练流程，提出在壮汉机器翻译预处理、模型训练、后处理三个部分，使用数据处理与增强的方法提升壮汉机器翻译性能。在预处理部分，本课题通过研究壮语的语言学知识，抽取其内在特性，应用在壮汉机器翻译的预处理部分，进行数据筛选等预处理操作提升数据质量。在模型训练部分，本课题使用反向翻译进行数据增强，通过有限的高质量双语数据训练反向翻译模型，将大量的、易获得的汉语单语数据送入反向翻译模型进行解码，获取壮语译文，得到大量的汉语、壮语双语伪数据，通过实验研究伪数据与高质量双语数据的混合方式，提升壮汉机器翻译模型性能。在后处理部分，本课题针对机器翻译在处理一些特殊专有名字、人名、品牌名时容易翻译错误的问题，提出了一种利用原文信息的后处理算法，可有效提升译文中特殊专有名字、人名、品牌的翻译准确率。~~后处理三个方面提出 数据处理与增强方法解决方法~~

（2）基于知识蒸馏的模型训练方法：机器翻译系统应用的主要形式是部署为线上系统，而部署线上机器翻译服务通常需要大量计算资源，随着机器翻译服务提供商的语种越来越多，如何让线上在不损失翻译质量的前提下节省计算资源变成了一个很有研究价值和商业价值的问题。本课题提出联合集成学习和知识蒸馏两种方法进行模型训练，这种方式训练出来的模型在保住节省计算资源的同时翻译性能同样优秀。集成学习可有效提高神经机器翻译译文质量;，但是集成学习的效果与参与解码的模型数量密切相关，模型数量增多会导致消耗大量的计算资源，本课题采用知识蒸馏的方法学习集成学习的结果，在保证译文优秀的同时大幅节省了计算资源。

集成学习是神经机器翻译中提升翻译质量的有效手段，集成学习是一种联合多个学习器进行协同决策的机器学习方法。集成学习应用高通过在神经机器翻译的

1.3 论文组织结构

本文主要研究了通过数据处理与增强的方法以及通过基于知识蒸馏的模型训练方法提高壮汉神经机器翻译系统的翻译性能以及计算效率，在数据处理与增强部分，我们针对壮语、汉语语言特性进行研究，通过预处理方法获取高质量数据，通过反向翻译方式获取伪数据提升模型性能，通过后处理方式解决部分专有名字、品牌名、人名翻译错误的问题，在基于知识蒸馏的模型训练部分，我们联合集成学习思想以及知识蒸馏理论，在保证模型翻译性能的同时大幅减低了计算资源的开销。

本文主要内容如下：

第1章主要介绍了本论文的研究背景以及研究内容。

第2章主要介绍了本论文研究内容中所使用到的相关技术，主要针对神经机器翻译中的模型架构以及相关技术进行介绍。

第5章对本文的研究工作和创新点进行了总结，同时对未来的工作进行了展望。

第2章 相关技术概述

2.1 神经机器翻译框架

机器翻译（Machine Translation，简写为MT）属于计算语言学的范畴，主要研究使用计算机程序将文字从一种自然语言翻译为另一种自然语言。简单来说，机器翻译是通过计算机将一个自然语言的文本转换成另一个自然语言的文本。机器翻译这一概念最早可以追溯到十七世纪，哲学家笛卡尔在1629年提出世界语言的概念：将不同语言中相同含义的词汇用统一的符号进行表示。近代以来，由于工业化的进程以及信息技术的发展，机器翻译的可行性大幅增加。二十世纪初便有数位科学家相继提出机器翻译的的理论以及实现计划。随着计算机的发明，美国政府和一些企业相继投入大量资金，用于机器翻译的研究上。但是此时的机器翻译主要集中在基于规则的机器翻译，大幅依赖人工规则的书写，随着研究的进行暴露出规则的获取难度大、人工书写的规则覆盖度有限、不同规则之间相互冲突等一系列问题。在二十世纪七十年代美国自动语言处理顾问委员会（Automatic Language Processing Advisory Committee）的报告机器翻译研究进展缓慢，使得政府和企业对机器翻译的兴趣大幅减少。

二十世纪九十年代，IBM的研究人员提出统计机器翻译（Statistical Machine Translation，简写为SMT），他基于概率与统计而不是语法与规则。大幅提高了机器翻译的翻译质量，政府与企业再次提起对机器翻译兴趣。与传统的基于规则的机器翻译不同，IBM的研究人员基于噪声信道模型把翻译问题建模为搜索翻译概率最大的译文的问题。（这里可增加噪声信道图）

基于噪声信道模型的统计机器翻译假定，源语言中的句子*s*{\displaystyle f}s是由目标语言中的句子*t*{\displaystyle e}经过含有噪声的信道编码后得到的。那么，如果已知了源语言中的句子*s*{\displaystyle f}和信道的性质，我们可以得到通过源语言句子产生目标语言句子的概率{\displaystyle p(e|f)}。而寻找最佳的翻译结果{\displaystyle {\tilde {e}}}也就等同于寻找求解这个概率的最大化，即：（这里可翻译模型语言模型）

公式2.1中*s*和 *t*分别代表源文和译文，表示把源问s翻译成译文t的概率，𝑎𝑟𝑔𝑚𝑎𝑥表示统计机器翻译模型从所有可能的译文中搜索最佳译文的过程。通过大量的双语语料，确定计算所需的模型参数的过程被称为模型训练。指定模型参数和待翻译句子，搜索概率最高的最佳的译文的过程又被称为机器翻译的解码。在过去的二十年里，研究人员在统计机器翻译领域开展研究工作，基于最大熵思想的统计机器翻译方法，基于句法结构的统计翻译模型、基于短语的统计机器翻译方法相继出现，机器翻译的性能大幅提高。在2005年，Google公司推出了在线机器翻译服务，其内在模型就是使用统计机器翻译进行训练的，机器翻译技术可以服务广大普通用户。随着信息化社会的不断发展，科技的不断发展以及普通民主的关注都促使机器翻译大幅发展。

随着机器学习等学科的进步，机器翻译的研究不断深入，研究人员发现统计机器翻译存在很多难以解决的问题。比如翻译特征严重依赖人工设计、特征工程耗费巨大且不同语种之间不适配、使用离散的表示方式导致泛华能力很差等等。为了解决这一问题，研究人员尝试使用人工神经网络搭建神经机器翻译系统。

所谓神经机器翻译（Neural Machine Translation，简写为NMT），是指直接使用深度学习的多层神经网络对翻译问题进行建模，这个过程不依赖人工定义的特征，模型泛化能力很强，神经机器翻译以端到端（End-to-End）的方式对翻译模型进行训练。神经机器翻译模型的训练过程大部分是通过反向传播算法应用梯度下降进行训练的，神经机器翻译的翻译解码过程是对已经训练好的神经网络进行前向推断计算的过程。

2.2.1编码器-解码器框架

机器翻译任务与图像识别、物体检测、字符识别等计算机视觉领域的任务有很大不同。计算机视觉领域的任务大多是以图片作为输入输出，即输入输出的维度是估计的问题。而在机器翻译中，用户输入的待翻译的源语言文本的长度无法控制，同时应该获取的译文的长度以难以控制，这就导致了我们无法预先对神经机器翻译模型输入输出的维度进行预判，无法把已取得优秀结果的计算机视觉领域的神经网络框架直接应用到神经机器翻译任务中。

为了解决机器翻译这种变长序列到序列的映射问题，Kyunghyun Cho等人最早提出将“编码器-解码器”框架应用到神经机器翻译任务中，神经机器翻译与传统的基于规则的机器翻译、统计机器翻译不同，它将翻译问题视为序列到序列的转换问题。“编码器-解码器”框架由两个神经网络组成，他们分别充当编码器和解码器。其中编码器负责将神经机器翻译的输入序列转化成一个固定长度的向量，这个输入序列允许输入可变长度的序列。而解码器将之前生成的固定长度的向量表示再转化成输出序列。

“编码器-解码器”框架是一个解决机器翻译等序列到序列映射问题的通用框架，它可以学习以另一个可变长度序列为条件的可变长度序列的条件分布。在Cho等人提出的方法中，编码器和解码器都使用循环神经网络，后续的科研工作者将GRU、LSTM等循环神经网络的变种应用到“编码器-解码器”框架中。具体来说就是在编码阶段利用编码器的神经网络单元，如RNN、GRU、LSTM等结构对输入序列进行编码，对于一个长度为m输入序列X={x0,x1,x2,…,xm},这种方法每个时间步将输入的源语言序列中的一个序列片段所对应的的向量xt送入编码器的神经网络单元，并输出当前的状态ct，并读取下一个时间步的输入，直到输入句子中的最后一个片段被读取完。神经元的输出ct被称为状态向量，又被称为隐层状态ht。隐层状态ht中包含着从输入序列开始到当前位置的所有信息，也就是说神经网络将序列的信息编码到隐层状态中不断传递。在每个时间步t中隐层状态的更新方式如公式2.1所示：

公式中的函数f代表“编码器-解码器”框架中使用的网络单元，如循环神经网络的单元RNN、LSTM等。在公式中我们可以看到，每一个时刻的隐藏状态t中包含着0到t时刻的所有序列片段的信息，因此最后一个时刻的隐藏状态中包含着整个输入序列的全部信息，我们把它称作为语义向量c。整个编码器过程就是编码器利用神经网络，将输入的源语言序列编码为一个固定维度的语义向量c，也被成为上下文向量。

而在解码阶段，解码器获得编码器编码的上下文向量c(即编码器最后一个时间步的隐层状态)，这个向量里包含着输入序列的全部信息，解码器根据向量c每个时间步生成一个输出序列的片段，最终预测得到完整的输出序列。具体的预测过程与编码过程类似，也是在每一个时间步生成一个隐层状态ht。假的输入的源语言序列都应的待输出的目标语言序列的长度是n，目标语言序列是Y= { y0,y1,y2,…,,yn}，解码过程中的隐层状态ht的更新方式如下公式所示：

(2.3)

函数f表示非线性变化，解码器多次循环，知道预测出完整的目标语言输出序列。计算每一个输出译文片段yt的概率的公式如公式—所示。

公式中的函数代表归一化的操作（如*softmax*），这是因为输出译文片段通过概率的形式表示，概率需要保证大于等于0且小于1。在解码的过程中，解码器在计算第一个时间步时需要的输入y0是预先定义的起始标识符c为从输入序列中提取出的语义向量。解码时从c中抽取信息作为预测输出序列的依据并不断循环。具体来说，初始状态时向解码器中输入一个起始标识符，解码器通过这个起始标识符、*c*计算出下一时间步的序列片段的概率分布，这个概率分布式整个词表中每个词出现在y1位置的可能概率。最后解码器选取概率最高的序列片段作为第一个时间步的预测结果。不断重复这一流程，解码器根据、和*c*预测出的概率分布，并输出概率最高的结果对应片段，解码器一直重复可以流程直到预测到结束标志，结束标志同样预先定义的，通常使用<EOS>（End Of Sentence）表示，也就得到了目标语言的输出译文。整个解码器端预测第*t*时刻的目标译文输出片段可用如下公式计算。

“编码器-解码器”框架应用到神经机器翻译当中最早由Kyunghyun Cho等人提出，在他们的论文中提出使用上下文向量在解码的每个时间步参与计算。不过这个设计会是计算量较大，模型更为复杂。Cho这个设计在随后的Ilya Sutskever等人的研究工作中得到改善。

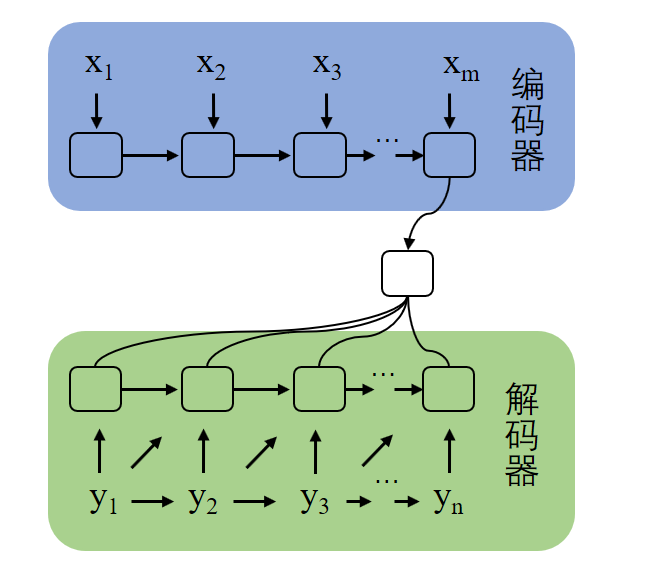


图2.1 Kyunghyun Cho等人“编码器-解码器”框架

Ilya Sutskever等人的论文中的解码器计算方式可用如下公式表示。

Ilya等人改变了解码器端的计算方式，他们将解码器的计算进行简化，编码器计算得到的上下文向量不再参与解码器中每个时间步的计算，而是与编码器的计算方式类似，上下文向量只参与解码器端第一个时间步的计算。Ilya等人的研究工作简化了“编码器-解码器”框架，编码部分输出的上下文向量只在解码器第一个时间步使用的设计简化了模型结构，在工程应用中大幅减少了计算量。

“编码器-解码器”框架的可以将语音、句子序列等直观表示和它们的语义表示之间建立映射。因此，“编码器-解码器”框架十分适合处理序列映射任务，当前这个框架已经广泛应用于机器翻译、语音识别、文本生成等任务。

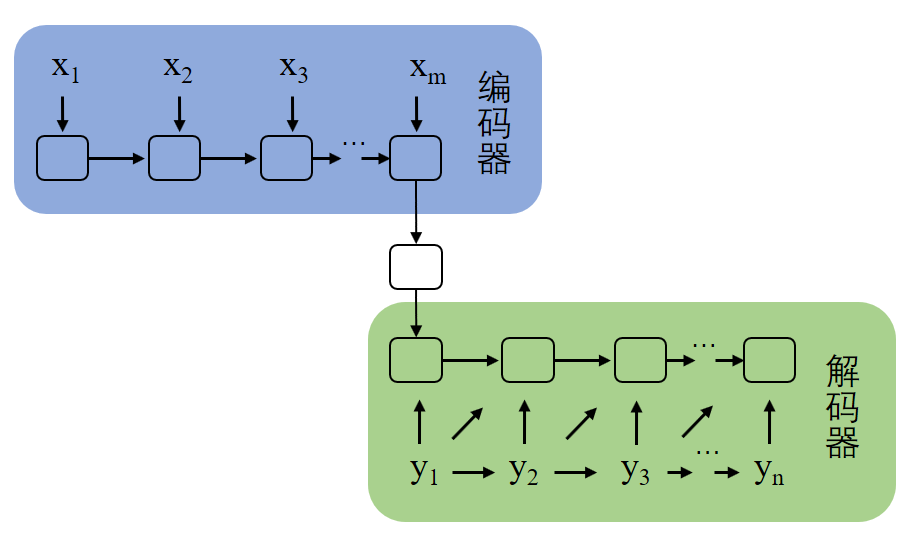
“编码器-解码器”框架完全把机器翻译问题转换为输入序列到输出序列的网络计算问题，不取决于人工设计的特征，机器翻译模型训练的输入和输出是清洗好的的双语语料库，大幅简化了训练过程，除此之外，模型可以更好地捕获不同语言之间的对应关系。基于“编码器-解码器”框架的神经机器翻译被研究机构和企业广泛接受。研究人员以此框架为基础，进行了许多改进和提高，在“编码器-解码器”框架提出并应用到神经机器翻译的几年里，机器翻译质量获得了跨越式的发展。

图2.2 Ilya Sutskever等人简化的“编码器-解码器”框架

2.2.2 注意力机制

“编码器-解码器”框架的提出将机器翻译问题转化为序列到序列的映射问题，神经机器翻译的可行性大幅提高。但是在“编码器-解码器”刚刚提出的时候，神经机器翻译的性能并没有想象中优秀，这是由于在“编码器-解码器”框架中解码的过程完全依赖于源语句子的编码结果上下文向量c，同时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等，这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。

Cho等人提出的“编码器-解码器”框架以及Ilya等人简化的版本，都是使用编码器对输入的源语言句子进行编码。编码器会通过RNN、LSTM等神经网络结构进行信息抽取，并将最后一个时间步的隐藏状态上下文向量c作为编码器输出。解码器使用这个上下文向量c进行解码，输出翻译后的目标语言译文。这在输入的源语言句子过长的时候会出很很多问题，假设输入的源语言句子中包含上百个单词，编码器把这上百个单词的全部信息集成到一个固定维度的向量中，这就会导致解码器无法有效解析以及信息丢失等问题。除此之外的，传统的、未使用注意力（Attention）机制的“编码器-解码器”框架中，神经网络还存在这难以学习到源语言端和目标语言端的对应信息

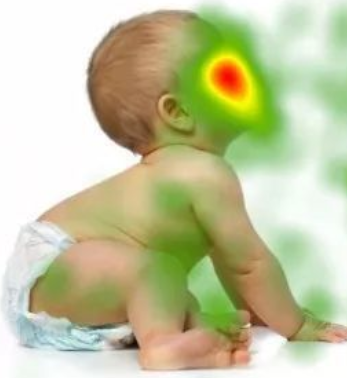
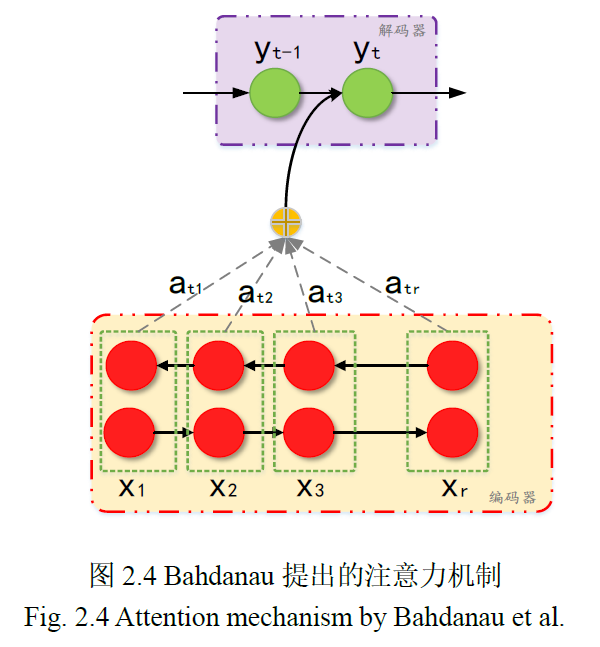
为了解决这种长输入序列编码到定长向量时出现的信息损失、难以解析问题，研究人员为“编码器-解码器”框架引入了注意力机制（Attention Mechanism）。顾名思义，注意力机制类似于人类观察图片等事物的视觉识别过程，人类在观察事物时视觉系统是不是同时关注待观察事物的各个细节，而是使用选择性注意力机制。人类视觉获得需要关注的目标区域，即关注的焦点，然后在该区域投入更多的关注资源以获得关于目标的更详细的信息。如图2.？所示。

图2.3 人类的视觉注意力

图2.3中的红色代表重点关注的区域，绿色次之。视觉系统选择性注意力机制人类利用有限的资源从大量信息中快速筛选出高价值信息的一种手段。注意机制极大地提高了视觉信息处理的效率。(还可以增加文本问题的注意力机制例子)

深度学习中的注意机制与人类选择性视觉注意机制本质上是相似的。核心目标是从众多信息中提取出对当前任务目标更为关键的信息。注意力机制在深度学习中最早由计算机视觉领域研究人员提出并应用，之后Bahdanau等人的研究工作把注意力机制应用到了神经网络机器翻译当中，这也是自然语言处理(Natural Language Processing，简写为NLP)中第一个使用注意力机制的工作。

注意力机制可以分为两个种类: 软注意力（Soft Attention）和硬注意力（Hard Attention）。 这两个注意力机制的不同之处在于，硬注意力只关注较小的区域，而软注意力机制则更加宽泛。在神经机器翻译待翻译的目标语言中每个单词的预测阶段，可以将硬注意力理解为仅涉及输入的源语言句子中的一个单词。问题是，源语言句子的重点信息很大可能不止一个，待预测的词可能和源语言句子中的多个词相关。 因此，现在关于神经机器翻译的研究主要以软注意力机制为主。

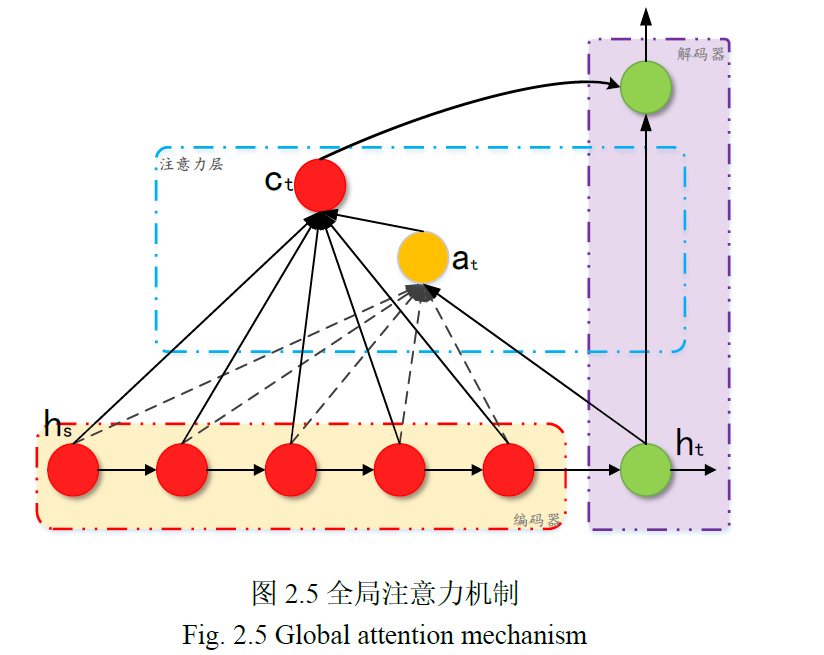
Bahdanau等人在2014年的论文中提出将注意力机制引入到“编码器-解码器”框架中。这种方法的核心思想是每次输出目标语言翻译结果时，都必须与源语言各时刻生成的隐层状态h交互。这确保在解码器中的每个片段的输出中可以考虑到源语言中的每个片段，并且根据源语言中不同状态的值确定当前时间的输出。图2.4注意力机制的示意图。

Bahdanau等人提出的注意力机制可以帮助神经网络在生成目标词时有选择地关注不同的源语位置，更多地关注更重要的位置，而较少地关注较不重要的位置。注意力机制使神经网络更容易捕捉与解码当前预测的目标语言片段更相关的源语言信息，从而使神经网络更准确地预测目标语言译文。基于注意力机制的神经网络在生成目标语言单词片段时，根据得到的不同的注意力对不同的源语言句子中位置的向量进行加权求和，从而得到不同的上下文向量。解码器在预测不同的目标语言片段时使用得到的不同的上下文向量进行预测。注意力机制的引入使得不同的源语言位置的单词片段对目标语言的待预测片段生成的贡献度不同，从而使得信息流在神经网络中的传输更加高效，有助于神经机器翻译等神经网络模型的学习。

不同于Cho等人以及Ilya等人使用的“编码器-解码器”框架，Bahdanau等人使用编码器中每个时间步的隐层状态参与解码器中片段的预测，而不是仅仅使用编码器最后一个时间步的隐藏状态。不过这个设计在面临长句子时候也存在着需要进行很多无效计算、信息量过多影响正常预测的问题。解码预测每个片段时都要使用编码器中每个时间步的隐层状态参与计算，会导致在神经网络模型需要对所有的片段进行多次计算，时间开销很大。

Luong等人对这一问题提出了解决办法，他们将注意力机制应用于神经机器翻译的模型结构中。Luong等人将注意力机制分为两类，一类是全局注意力机制（Global Attention），一类是局部注意力机制（Local Attention）。全局注意力机制与Bahdanau等人提出的方法相似，在神经机器翻译模型训练和解码的过程中，解码器根据前面时间步的已翻译目标语言状态，得到当前时刻要预测的目标语言序列段和源语言端序列段之间的相关度，然后将该值作为源语言序列中每个片段的信息权重，然后使用这个分段信息的权重进行加权求和，得到输入序列的上下文向量。需要注意的是，全局注意力不再仅仅使用编码器的最后一个时间步的隐藏状态作为上下文向量，同时，解码器计算时当前时间步的隐层状态不再与上一个隐层状态直接相关，隐层状态的计算方式可以用如下公式表示。

表示包含注意力的隐层状态，解码器预测时通过Softmax函数，预测当前时间步的目标语言片段的概率分布，如公式（2.6）所示。

从两个公式中可以看出，包含注意力的隐层状态不再与直接相关，而是通过当前时间步的隐层状态计算得到。在解码器进行目标端序列的推测时，Luong等人提出的全局注意力机制在计算时只用当前时间步的隐层状态，在模型实现以及计算时都比Bahdanau 等人提出的使用上一个时间步的隐层状态的设计简单。全局注意力机制的结构图如图2.？所示。

如上图所示，全局注意力，使用编码器的所有隐藏状态来生成上下文向量。在计算时，首要根据编码器当前时间步的隐藏层状态和编码器的隐藏状态，生成一个长度可变的向量，代表着注意力分布，可以理解为解码器中不同时间步状态对编码器当前预测片段的贡献度，如公式（2.6）所示。

其中是分数函数，即计算当前解码片段与编码器每个时间步的隐藏状态的相关程度。Luong等人对分数函数计算方式提出了改进，分为三种计算方法包括点乘、通用和级联：

点乘的计算方法如公式（2.8）所示。

通用的计算方法如公式（2.9）所示。

级联的计算方法如公式（2.10）所示。

利用以上公司，可以获得当前时间步解码状态与编码器的每个时间步的隐藏状态之间的相关性分布，并且对和当前预测比较相关的源语言端的隐藏状态分配相对大的权重，而不相关的被分配较小的权重。最后进行加权求和，得到最终的上下文向量。

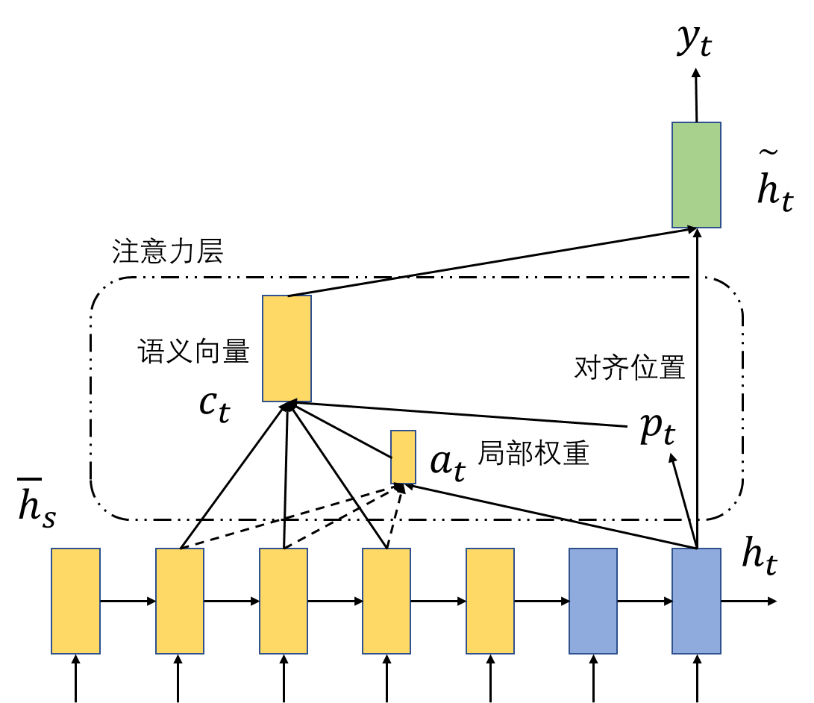


图2.6 局部注意力

Fig. 2.6 Local attention

然而全局注意力机制的编码器端输出的所有时间步的隐层状态都需要参与计算，通常解码器中一些时间步的输出并不与编码端所有时间步的隐层状态都相关，或者影响很小。这就容易导致一些关系不大的编码端时间步隐层状态会影响输出，同时与所有时间步相关会导致计算次数增多，计算开销变大。针对这一问题，Luong等人提出了局部注意力机制（Local Attention），局部注意力仅基于编码端的隐藏状态的一部分来计算解码器端当前预测的片段。在局部注意力机制中，使用长度为2D的窗口来选取需要关注的隐藏层状态信息。由于只需要选取并计算编码器端隐藏层状态的一部分，所以消耗的计算资源较少。 此时，局部注意力需要额外解决的是找到定位局部窗口中心点的位置，从而将区间[−D，+D]的局部信息作为要关注的部分。Loung等人提出两种确定v的方法有两种解决方法:1）=t，即编码器中与解码器当前时间步一直；2）使用公式计算来寻找，如公式2.7所示;

在公式中，和都是模型训练过程中可学习的参数矩阵，是输入源语言的序列长度，sigmoid函数结果区间是(0,1)，所以最终公式的结果是大于0小于输入序列长度的点。局部注意力的结构如图所示。

局部注意仅针对中心词窗口内的词进行计算，因此中心词的位置的预测以及窗口大小的选取至关重要。由于只涉及输入的源语言句子中的部分单词，局部注意可能会忽略一些信息，并且由于句子的长度不同，在计算力提升后研究人员很少使用局部注意力方法。

自注意力（Self-Attention）机制是对传统注意机制的改进，它更关注序列的内部相关性。自注意力机制是一种当前被研究人员广泛使用的方法。自注意机制减少了对外部信息的依赖，并注重捕捉自身的内部因素。序列关注其内部信息以获得序列本身的内部联系，从而更准确地捕捉词与词之间的语义特征或者句法信息。

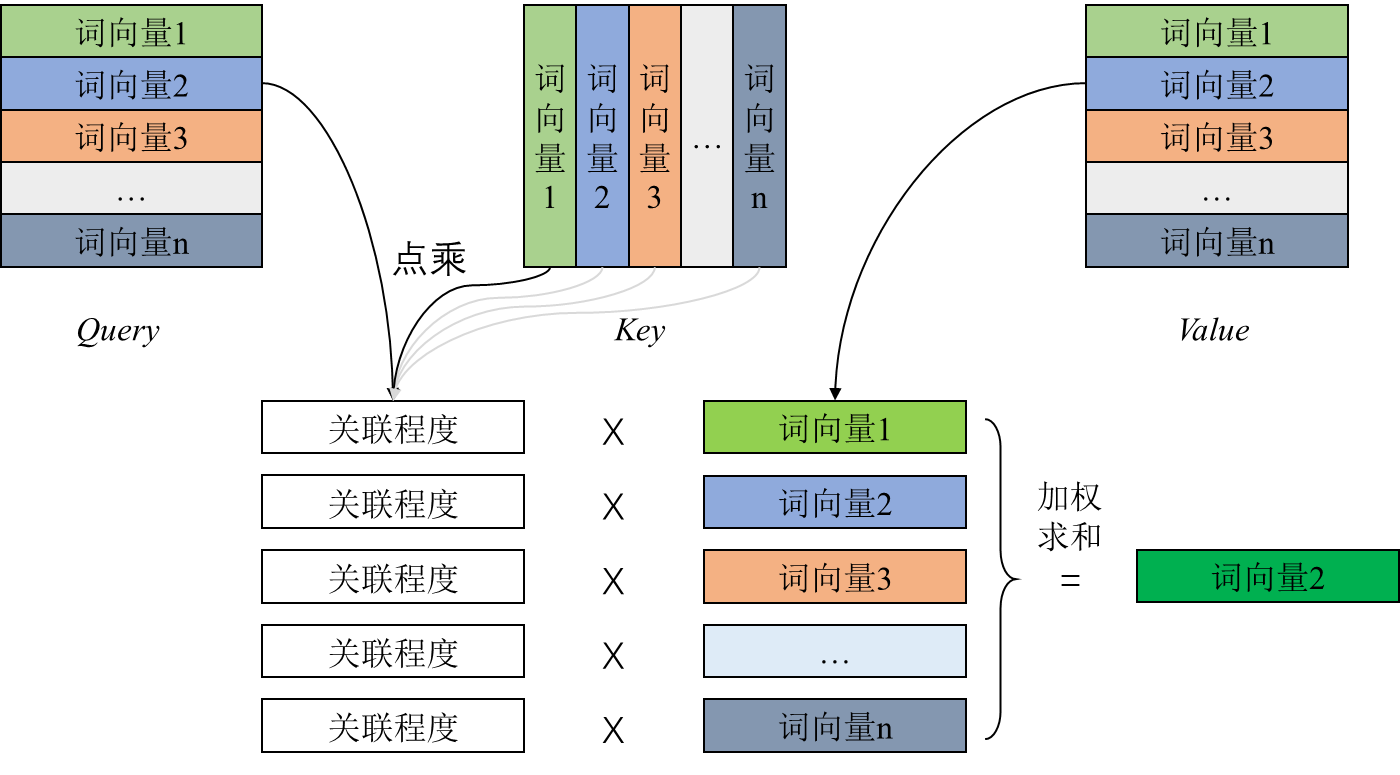


图2.7 自注意力机制

Fig. 2.7 Self-attention mechanism

自注意力机制可以理解为查询（Query）到键（Key）-值（Value）的映射。在计算注意力时主要分为三步，第一步是将query和每个key进行相似度计算得到权重，常用的相似度计算函数有拼接，点积，感知机等；第二步使用softmax函数对这些权重进行归一化处理；最后将权重和相对应的键值value进行加权求和得到最后的注意力。在自注意力机制中，Query、Key和Value都是一样的，从而可以实现对自身序列的信息提取。

2.2 神经机器翻译模型结构

2.2.1 基于循环神经网络的神经机器翻译模型

机器翻译、情感分析、语言理解等任务作为自然语言处理领域的重要任务，都需要能够有效处理可变长度序列的模型。神经机器翻译(NMT)的核心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。在神经网络中，变长序列问题的最经典解决方案之一是通过循环神经网络对变长序列问题进行建模。循环神经网络（Recurrent Neural Network, 简写为RNN）将时间序列之一概念引入到神经网络的结构中，因此循环神经网络在处理可变长度序列方面具有独特的优势。

Cho等人的研究工作中就使用循环神经网络分别实现了编码器和解码器，循环神经网络由输入层、隐藏层和输出层三部分为基础，输出激活激活控制，每层之间通过网络权重连接。基本的神经网络只有层之间进行权重连接。循环神经网络和基本的神经网络之间的主要区别是在神经元之间也建立了权重连接。循环神经网络的网络结构如图2.4所示。

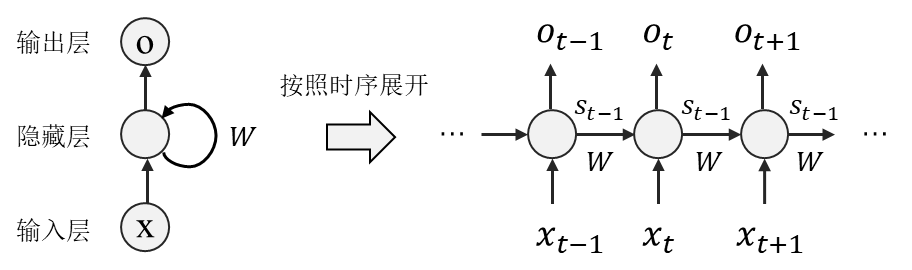


图2.4 循环神经网络结构

Fig. 2.4 Structure of recurrent neural network

一个最简单的循环神经网络由输入向量X、前一个时间步的隐层状态向量S、权重W组成，核心计算方式是每个时刻向循环神经网络中送入输入向量X，然后经过网络计算并将结果送入隐藏层，隐藏层中使用上一个时间步计算得到的隐层章台以及计算后的输入向量相结合，获得当前时刻的隐层状态，并将其送入输出层以及传递到下一时刻参与计算。循环神经网络重复这一过程，直到输入序列中所有的输入都已经送入网络计算完毕。

图2.4中左侧部分表示隐藏层中的数据传输模式，右侧部分将左侧的图根据时间顺序扩展表示，标出每个时刻输入、输出的之，以获取每个时间点循环神经网络的输入和输出。右侧部分图中，在时间点*t*时，输入层传递给隐藏层的输入向量为，同时上一时刻计算得到隐藏层的状态为，那么计算当前时刻隐层状态可以使用公司(2.16)表示。

(2.16)

在公式中*W*代表为输入向量的权重矩阵，*U*代表隐层状态的变换矩阵，*b*是偏置量，表示*tanh*、*sigmoid*等激活活函数，通过以上公式的计算，可以获得当前时刻隐藏层状态的输出。循环神经网络中存在梯度爆炸和梯度消失的问题，在神经网络模型的训练时十分困难，因此研究人员在RNN上进行改进，提出了长短时记忆网络（Long Short-Term Memory，简写为LSTM）来缓解该问题。LSTM属于循环神经网络的一个变种，但是由于内部加入门控制单元的原因，它比普通的循环神经网络复杂，LSTM在处理长距离依赖时，改善效果显著。

长短时记忆网络主要引入门控机制和记忆向量来处理长距离依赖的能力，LSMT存储每个时间步的有意义的信息并从头向尾进行传输。记忆向量从序列的开始到结束保存着所有的有意义的信息，同时网络单元中的门控机制判断着哪些信息是有意义的，并将其添加到记忆向量中。门控机制还负责判断哪些信息是无意义的并使其不再参与记忆向量计算。这些门控机制有数种，包括输入门(input gate) 、输出门(output gate)和遗忘门(forget gate)。

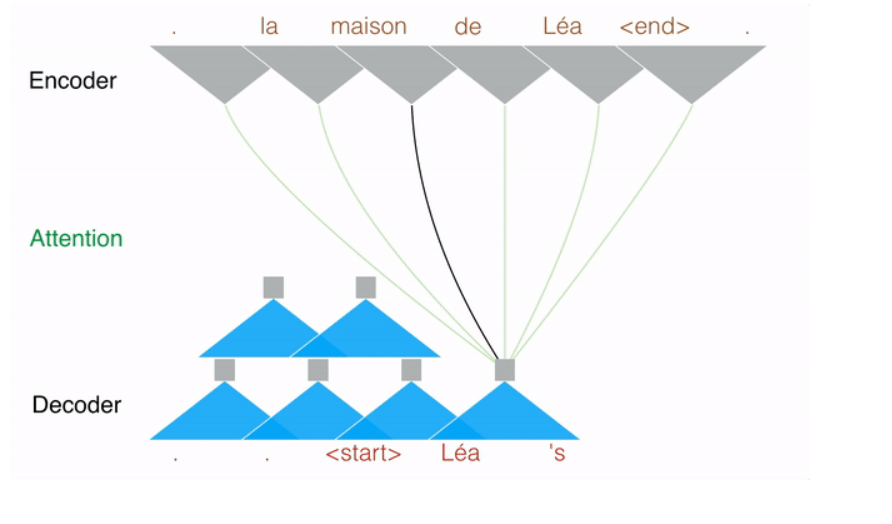
长短时记忆网络有很多个版本，其中一个重要的版本是门限递归单元[38]（Gatedrecurrent unit或GRU），这是为了缓解长短时记忆网络结构复杂而设计的，GRU将遗忘门和输入门结合输入到单个更新门（update gate）中，同时还将记忆向量和隐藏状态合并，并提出出一些其他的改进。所得的模GRU型比标准的LSTM模型结构简单，且效果与长短时记忆网络相当。

基于LSTM或者GRU等循环神经网络结构搭建的循环神经网络可以Cho等人提出的“编码器-解码器”框架图。在基于循环神经网络的机器翻译中，一般编码器端由一到两个双向循环神经网络组成。使用双向循环神经网络是为了是编码器能同时捕获输入序列的正向信息以及反向信息。解码器端一般由一个单向的循环神经网络组成，解码器类似于语言模型的设计。在解码时，解码端首先从预先设定好的起始符<sos>开始利用上下文向量等信息循环计算，直到预测出定义好的结束符<eos>。解码器端预测输出片段的公式如下。其中和分别代表权重矩阵和偏置。

(2.19)

2.2.2 基于卷积神经网络的神经机器翻译模型

目前，基于循环神经网络的神经机器翻译系统已经取得了令人瞩目的成果，很多机器翻译中的重要成果如注意力机制等都是基于使用循环神经网络搭建的机器翻译系统研究的。但是循环神经网络中存在一些难以克服传统缺点，这主要是由于循环神经网络的结构设计导致的。比如，因为循环神经网络当前时刻的输出取决于前一时刻的输出，该模型很难进行并行训练，并且由于循环神经网络的结构复杂，参数太大以及循环神经网络要求输入序列是按顺序输入的，因此并行性非常差，计算效率低下。

上述问题困扰着研究人员数年，在神经机器翻译刚刚起步时，训练一个优秀的机器翻译系统往往需要数周甚至更多。在2017年上半年，Facebook的FAIR团队提出了ConvS2S，他们将卷积神经网络应用到自然语言处理中并获得成功。ConvS2S同样是基于“编码器-解码器”框架，与循环神经网络的依赖时序信息不同，基于卷积神经网络的机器翻译通过移动卷积窗口来提取其覆盖区域的语义信息，大大减少了单词之间的信息传递距离。卷积窗口的使用不仅减少了信息传递距离，而且提高了模型的并行性。ConvS2S充分利用了卷积神经网络高并行性的优点，在保证模型性能的前提下大大提高了模型的计算效率，节省了训练时间。基于卷积神经网络的模型结构如图2.5所示。

正如将卷积运算应用于机器翻译中一样，卷积神经网络可以广泛应用于从其他序列到序列任务（例如摘要提取，文章改写等）的任务。由于卷积网络结构可并行化的优点，使用这种结构的模型在计算效率上更好，并且可以更有效地利用GPU设备的并行功能来有效地加快神经网络模型的训练和解码速度，尤其是在处理序列长度较长的时候，基于卷积网络搭建的模型比传统的基于循环神经网络的模型更有优势。

2.2.3 基于自注意力机制的神经机器翻译模型（可以加RNN CNN）

在注意力机制提出以后，基于“编码器-解码器”框架的神经机器翻译性能得到了大幅提升，无论是传统的基于循环神经网络的机器翻译还是FAIR研究研究提出的基于卷积神经网络的机器翻译中，注意力机制对最终翻译性能影响十分关键，不管是在哪个神经机器翻译模型中，注意力机制都是不可缺少的一部分。

 一些研究人员开始着重研究注意力机制在神经机器翻译中的更多应用，2017年，谷歌的研究人员提出一种新的机器翻译模型Transformer[13]，这个完全基于注意力机制，抛弃了传统的循环神经网络结构和卷积神经网络结构等单元，研究人员将自注意力（self-attention）机制引入到机器翻译中来，自注意力机制有着抽取信息能力强、可并行化的优点。自注意力机制的使用减少了不同位置输入序列片段间的信息传输距离，Transformer模型可以更完整地表示序列中不同位置之间的复杂关系。实验表明，Transformer模型在机器翻译任务中表现非常好，翻译性能优秀，并行度高，训练时间短。Transformer的模型结构如如所示。

Transformer模型同样采用“编码器-解码器”框架设计，Transformer标准模型的编码器由部分6个相同构造层堆叠的组成，这6层中的每个都有两个子层：多头注意力（Multi-Head Attention）和前向连接层。标准Transformer的解码器也由6层堆叠而成，与编码器中每层由两个子层组成不同，解码器中每层由三个子层组成，第一个子层以及最后一个子层和编码器比较相似，分别是掩盖的多头注意力(Masked Multi-Head Attention)和前向连接层，解码器的中间子层即编码器-解码器多头注意力 (Encoder-Decoder Multi-Head Attention)，负责处理编码器的输出。无论是Transformer的编码器还是解码器，每一个子层后都会有残差连接和层正则化（Layer Norm）操作，以此来加速Transformer模型收敛过程。

多头注意力层就是自注意力机制的核心，多头注意力主要由由部分组成，头部(head)以及自注意力。首先，使用不同的权重矩阵将输入的信息压缩为相对较小的几个维度的输出，然后对这几个输出分别做自注意力处理，然后对所有结果进行级联操作合并，最后对最终输出结果进行线性变换。如图2.8所示。

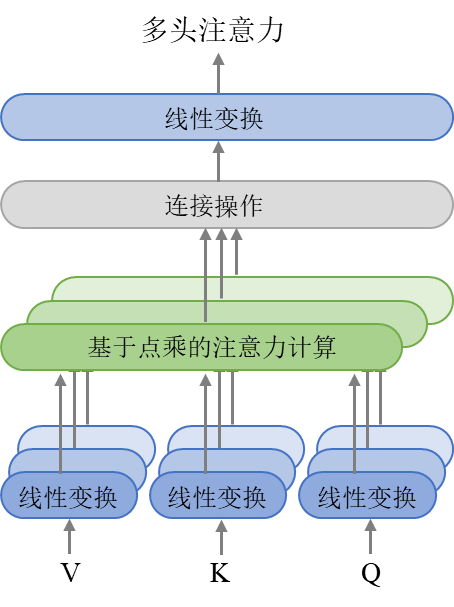


图2.8 多头注意力机制

Fig. 2.8 Multi-Head attention

多头注意力整体计算过程如公式(2.20)所示。

(2.20)

公式(2.20)中的表示输入信息经过压缩后得到的几个维度较小的矩阵，也就是多头注意力的“多头”部分。计算方式如公式(2.21)所示。

(2.21)

标准的Transformer模型中多头注意力部分分成8个头，8个头的并行计算极大的提升了模型的并行化程度，提高了模型计算效率。

对于编码器来说

根据实验证明，注意力机制对模型性能有着显著地提升，那么一个自然的疑问是: 注意力机制既然如此有效，那么我们为什么不去掉模型中的RNN部分，仅仅利用注意力机制呢？针对这个问题，Google于2017年提出了完全基于注意力机制的模型结构，称为Transformer，该模型摒弃了传统的基于循环神经网络和基于卷积网络的方法，达到了更好的性能，并且有效提高了模型的并行程度，加快了训练速度，Transformer模型的结构如图（2.8）所示。

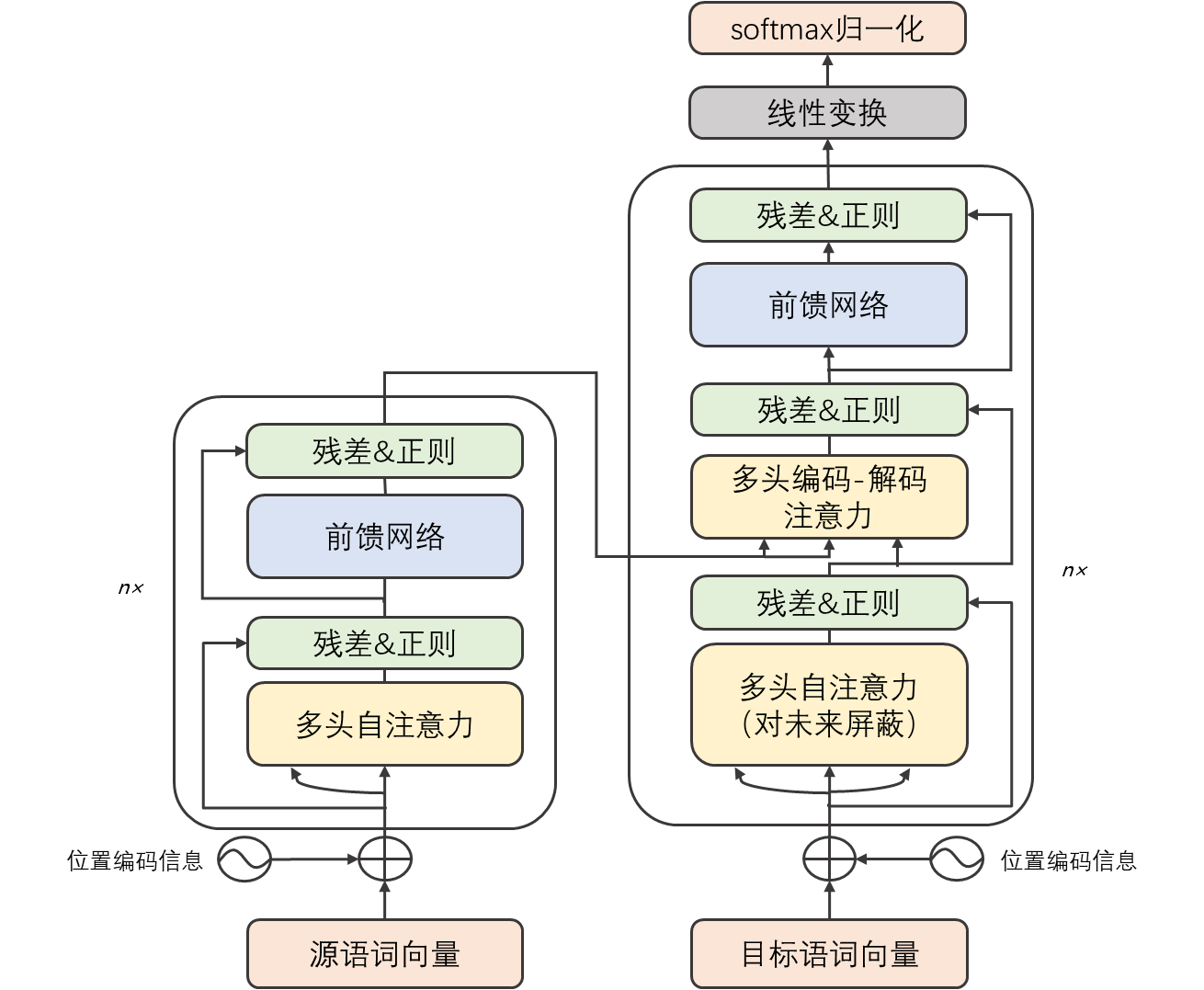


图2.8 基于自注意力机制的Transformer

Fig. 2.8 Transformer based on self-attention mechanism

Transformer的网络结构完全是由注意力机制组成。更准确地讲，Transformer仅由注意力机制和前馈神经网络组成。Transformer仍是基于编码器-解码器框架，通过堆叠多个相同的栈，分别组成了编码器和解码器，编码器和解码器的子层结构略有不同。论文中的实验基于编码器和解码器各6层的模型结构，在机器翻译的WMT14英德和英法任务上有了显著的性能提高，而且训练速度更快。

Transformer的核心组成部分是多头自注意力（Multi-headed Self-Attention）机制层，编码器的每个块包含了一个多头自注意力子层和一个前馈神经网络子层，为了能够更好地优化深层网络结构，Transformer在每个子层之中使用了残差连接和批正则化。如公式（2.14）所示。

多头自注意力机制层的结构如图（2.9）所示，可以理解为多个自注意力结构的结合，即将1次计算转换为多次计算，这种方法可以允许模型的每个头在不同的表示空间学习到不同的信息，学习到的内容更加丰富。

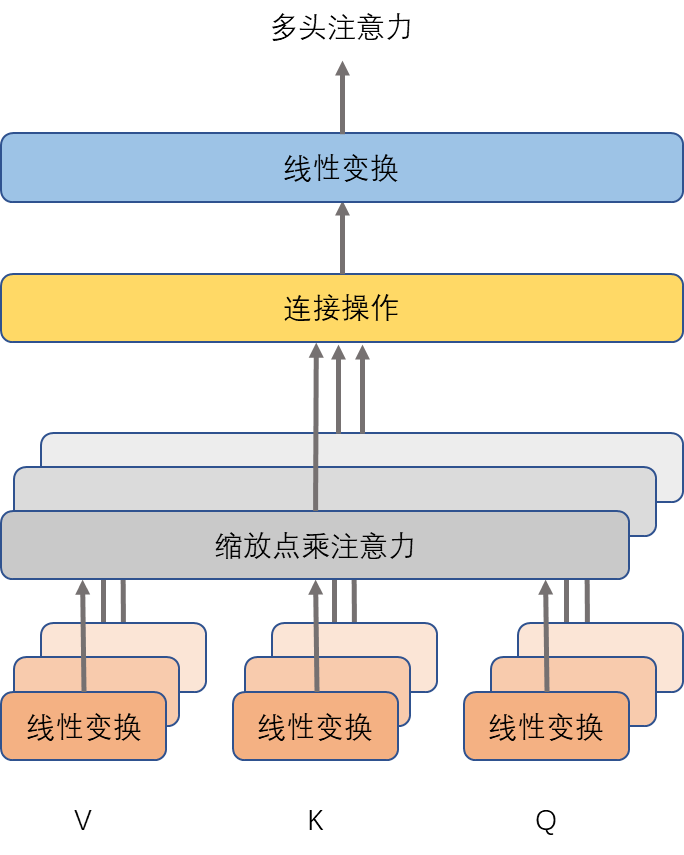


图2.9 多头自注意力

Fig. 2.9 Multi-headed Attention

前面说过，在自注意力机制中，查询（Query，Q），键（Key，K）和值（Value，V）来自相同的内容，首先对三个矩阵分别进行线性变换，然后进行缩放点积操作，即计算Query与Key进行点积计算，为了防止计算结果过大，会除以Key的维度来达到调节作用。对于上述计算结果，通过Softmax函数进行归一化为概率分布，然后再乘以矩阵V就得到权重求和的表示，如公式（2.15）所示。

如果仅仅依赖注意力机制，Transformer并不具备捕获序列位置信息的能力，无论句子的结构怎么打乱，Transformer都会得到类似的结果。换句话说，Transformer只是一个功能更强大的词袋（Bag of Word）模型。为了解决这个问题，Transformer在词嵌入层引入了位置嵌入（Position Embedding）的特征。具体地说，位置嵌入在词嵌入基础上加入了单词的位置信息，通过这种方法，Transformer可以区分不同位置的单词。位置嵌入的公式如（2.16）和（2.17）所示。

其中，表示单词的位置，表示单词的维度。模型结果设计的原因是考虑在NLP任务中，除了单词的绝对位置，单词的相对位置也非常重要。

根据公式（2.18）和（2.19），可以看出第个词的位置嵌入可以表示为第个词位置嵌入的线性变换，这为模型提供了捕获相对位置信息的可能性。

相比编码器，解码器同样堆叠多个相同的栈，除了编码器栈中的两个子层，解码器的栈中额外包含一个子层，在编码器的输出在执行注意力操作，一般称为编码器-解码器注意力子层，用于捕获当前翻译和编码的特征向量之间的联系。和编码器类似，在每个子层后都包含残差连接和批正则化操作。此外，解码器的多头自注意力子层需要屏蔽未来的序列，因此一般称为带掩码的多头自注意力子层。

所谓掩码，即是在自注意力计算过程中，在进行缩放点积操作之后，将表示未来位置部分的权重设置为，这样在经过Softmax函数归一化之后，未来位置的概率无限趋近于0。这样做的意义在于，在解码过程中，模型一般自左到右进行翻译，在翻译过程中并不知道未来的序列内容。如果使用普通的多头自注意力，模型可以看到完整序列，也就是可以看到自己的预测结果，这样会导致训练解码不一致问题。因此，带掩码的多头自注意力层的引入有效解决了这个问题，通过掩码操作，在训练期间模型也无法获得将来的信息，从而和解码阶段相匹配。

在解码器的顶层，模型对输出的特征进行线性变换，得到词表大小维度的输出矩阵，然后通过Softmax函数，对词表大小维度的矩阵进行归一化，从而得到词表中每个词的概率分布。

相比之前基于循环神经网络和基于卷积神经网络的神经机器翻译模型，Transformer的模型结构优点在于：

（1）Transformer仅仅利用前馈神经网络（或者是一维卷积）和注意力机制，没有使用各种复杂的模型结构，却达到了更优的性能。

（2）Transformer基于多头自注意力机制，自注意力机制将任意两个单词的距离缩短为1，这对解决自然语言处理任务中长期依赖问题是非常有效的，而且通过多头的计算方式可以使模型捕获到更丰富的特征，增加模型的并行性，从而提高了计算速度。

（3）Transformer不仅应用在机器翻译领域，在自然语言处理的其他任务上也取得了优异的性能，甚至可以不局限于NLP领域，在计算机视觉、语音等领域，都具备非常的潜力。

2.3 预训练方法发展介绍

预训练方法最初在图像领域提出，拥有较长的历史，而且达到了很好的效果。图像领域一般针对于各种分类任务，采用预训练的方法一般分为以下两步：设计好一个通用的神经网络模型结构，在一个训练数据十分充分的数据集合上进行学习，该数据集一般包含大量且通用的训练样本，可以让模型学习到一个很好的状态；之后对于一个新的任务，可以将这个预训练好的模型参数初始化新的模型，固定模型中的部分参数使其不参与训练，然后通过任务特定的训练数据来微调该模型。

这种做法的好处是训练代价很小，预训练的模型参数可以让新的模型达到更快的收敛速度，并且能够有效地提高模型性能，尤其是对一些训练数据比较稀缺的任务，在神经网络参数十分庞大的情况下，仅仅依靠任务自身的训练数据可能无法训练充分，预训练方法可以认为是让模型基于一个更好的初始状态进行学习，从而能够达到更好的性能。

预训练方法之所以能够有效，主要有两个原因。一是神经网络学习到的特征是包含一定规律的，研究人员发现，网络底层一般抽取较为通用的特征，几乎相同领域的所有任务都需要对这些特征进行建模，而网络顶层学习到的特征一般是任务特定的，与训练数据所对应的任务本身更为相关，因此在使用预训练模型的过程中，一般是固定模型底层的参数，然后对顶层的参数进行微调。二是因为预训练模型的训练数据足够充分且通用，比如图像领域的ImageNet数据集，包含一千四百多万张图片，两万多种类别，是图像领域十分通用的集合，所以在该集合上进行训练之后，得到的模型可以很好地泛化到图像领域的其他任务之上。

预训练方法在图像领域得到了巨大的成功，研究人员们尝试将预训练方法应用于自然语言处理任务中，从最初基于词嵌入的预训练方法发展到了现在基于语言模型的预训练方法。预训练方法在自然语言处理领域取得了长足的进展，能够有效提升诸多下游任务的性能，同时也减小了研究人员的训练代价。

2.3.1 基于词嵌入的预训练方法

词嵌入是自然语言处理任务中对词汇的一种表示方法。在使用计算机处理文本的过程中，将一个词表示为计算机可以理解的形式是一个必经的流程。不同于计算机视觉任务，可以简单使用像素值来表示图像的信息，语言本身具有高度抽象性。现如今，自然语言处理领域一般使用离散空间表示和连续空间表示两种词表示方法。

离散空间表示，一般也称为one-hot表示，一般被用于传统的统计自然语言处理任务中。离散表示将每个词看成是一个离散的符号，在计算机中通过一个唯一的id进行表示。对于一个文本，统计将其中所有的单词，可以得到该文本对应的词表。而离散表示可以看作是一个词表大小的向量，其中只有该词对应的id的位置为1，其他位置为1，所以称为one-hot表示。

连续空间表示，一般也称为分布式表示（Distributed Representation）、词向量、词嵌入（Word Embedding），后文统称为词嵌入。连续空间表示将每个词表示为连续空间的一组向量，而不是简单的一个id，这样可以使用更少的维度，又可以包含更丰富的信息。在连续空间表示中，词与词之间的联系可以通过在连续空间中的位置信息表示，不再是彼此独立的关系，因此可以表示更丰富的词汇信息。在神经网络中，普遍使用连续空间表示作为词汇的表示方法。

2003年，Bengio等人提出了神经语言模型（Neural Network Language Model），被认为是神经网络在自然语言处理领域的开篇之作[33]。神经语言模型的结构是一个单隐藏层的前向神经网络，通过序列的前n个词来预测下一个词，即n元语言模型。对于一个长度为N的序列，将其表示为，n元语言模型通过序列建模下一个词出现的概率，如公式（2.20）所示。

神经语言模型在训练过程中，不仅学习到预测下一个词的概率分布，同时也得到了一个副产品：词嵌入表示。相比随机初始化的词嵌入，模型训练完成后的词嵌入已经包含了词汇之间的信息。

2013年，Mikolov等人提出了word2vec工具[34]-[35]，其中包含了CBOW（Continue Bag of Words）模型和Skip-gram模型，该工具仅仅利用海量的单语数据，通过无监督的方法训练得到词嵌入。

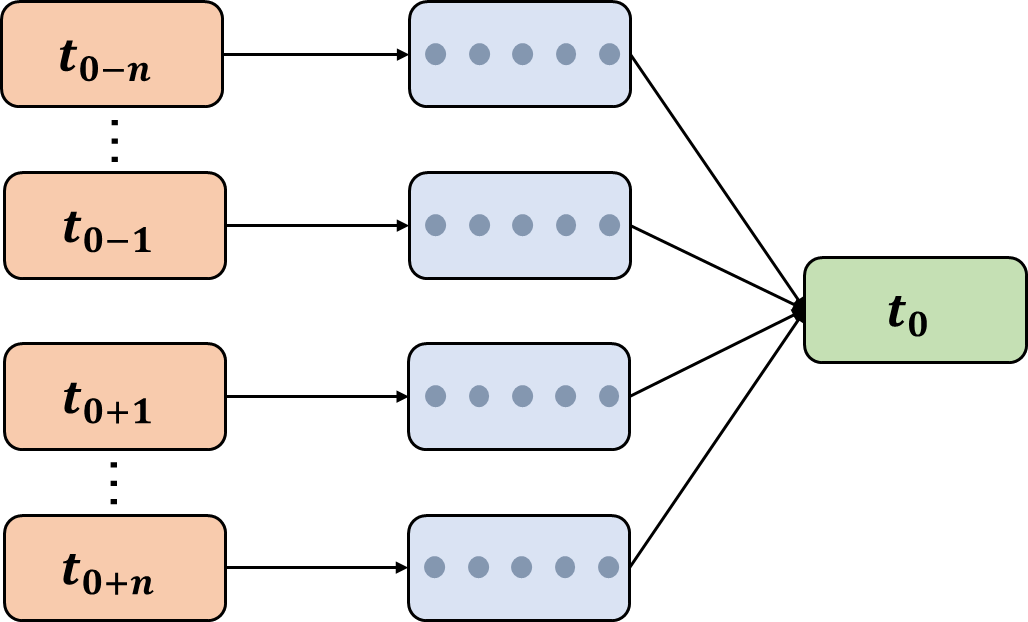


图2.10 CBOW模型

Fig. 2.10 CBOW model

CBOW模型的结构如图（2.10）所示，该模型忽略了词序的关系，对于一个长度为N的序列，将其表示为，假设词表大小为，通过一个中间词的上下文的词嵌入平均值来预测该词，如公式（2.21）所示。

其中，表示Softmax函数的权重矩阵，的计算公式如（2.22）所示。

其中，是窗口大小，是词嵌入权重矩阵。

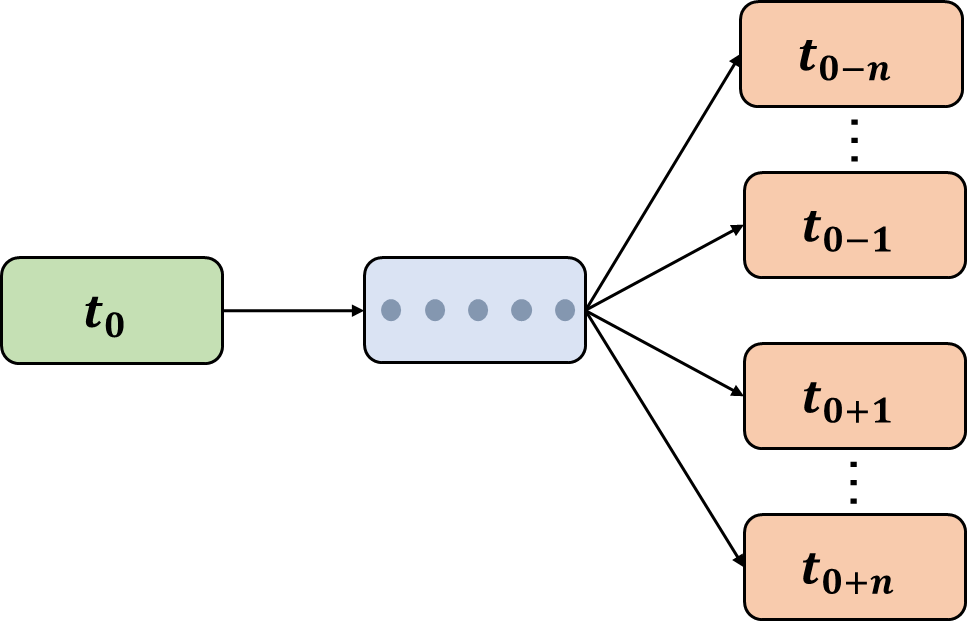


图2.11 Skip-gram模型

Fig. 2.11 Skip-gram model

Skip-gram模型结构如图（2.11）所示，对于一个长度为N的序列，将其表示为，假设词表大小为，该模型通过一个中间词来预测上下文的词嵌入，如公式（2.23）所示。

其中，的计算方法如公式（2.24）所示。

在神经语言模型中，语言模型是真正的建模目标，而词嵌入只是一个副产品。word2vec提出CBOW和Skip-gram模型的目的就是为了简单有效地训练词嵌入。两种方法的主要区别在于计算复杂度，使用神经语言模型生成词嵌入需要的计算量过大，这也是早期时候，词嵌入在NLP领域没有广泛应用的原因，而word2vec提出的两个模型的网络结构十分简单，没有使用隐藏层和非线性变换，因此可以快速训练。通过word2vec预训练词嵌入的代价大大减小，研究人员们使用大量的单语数据对词嵌入进行预训练，然后用来初始化下游任务的模型，对训练数据较少的任务有了明显帮助。

2.3.2基于语言模型的预训练方法

以往的神经网络预训练方法主要是针对网络的输入层，通过大量的单语数据预训练词嵌入，这种方法在很多数据量较少的任务上也取得了一定的成功。但词嵌入本身具有一些局限性，最主要的缺点是无法解决一词多义问题，不同的词在不同的上下文中会有不同的意思，而词嵌入对模型中的每个词都分配了一个固定的表示，这明显是不合理的。

针对上述问题，Peters等人提出了ELMo（Embedding from Language Model），即使用语言模型来获取深层的上下文表示。ELMo的具体做法是，基于每个词所在的上下文，利用双向LSTM的语言模型来获取这个词的表示。ELMo的网络结构如图（2.12）所示。

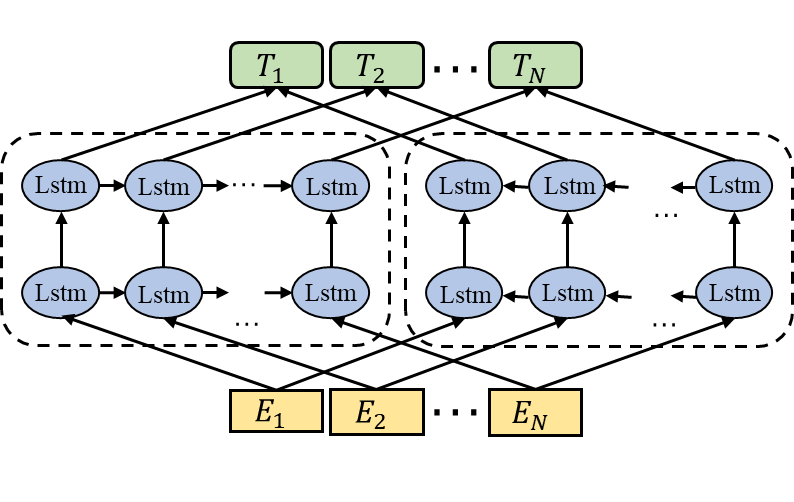


图2.12 基于双向LSTM的ELMo

Fig. 2.12 ELMo based on bi-LSTM

对于一个长度为N的序列，将其表示为，前向语言模型在给定序列的情况下，建模下一个词出现的概率，如公式（2.25）所示。

而反向的语言模型与前向类似，在给定序列的情况下，建模前一个词出现的概率，如公式（2.26）所示。

双向LSTM的语言模型可以通过最大化前向语言模型和反向语言模型的对数似然来训练，如公式（2.27）所示。

其中，是词嵌入层的参数，和分别表示前向和反向两个语言模型的参数，而表示Softmax层的参数。通过前向和反向两个语言模型，每层都可以得到两个表示和。当模型完成预训练之后，可以将每层的两个表示级联出来，然后下游任务根据自己的需求对每层的表示进行加权组合，以使其更适用于本任务。

ELMo通过预训练模型来提取特征，下游任务基于更加丰富的特征进行训练，在自然语言处理领域的6个子任务上有效提高了模型性能，包括问答、文本蕴含、语义作用标注、指代消解、命名实体识别和语义分析。ELMo的提出推动了预训练方法在自然语言处理领域的发展，引起了研究人员对预训练方法的关注。

ELMo的方法能够提取丰富的特征给下游任务使用，但是相比图像领域，能够预训练整个网络，然后使用网络参数去初始化新的模型，仅仅进行特征提取可能远远没有发挥预训练的潜力，另外一个不足之处是，现如今在自然语言处理领域，普遍使用的是基于自注意力机制的transformer模型结构，相比LSTM能够更有效地捕获长距离依赖，对句子中信息进行更充分的建模。

针对上述两个问题，Radford等人提出了GPT（Generative Pre-Training），即生成式的预训练。GPT使用提取能力更强的单向transformer来训练语言模型，预训练完成之后，对于不同的下游任务，GPT使用预训练好的模型参数来初始化新的模型，然后进行微调，其中不同的下游任务，微调的方式可能略有不同。GPT在自然语言处理的12个任务上刷新了9个任务的state-of-art的成绩，相比ELMo有了更大的提升。

但是GPT也存在着一个重要的不足，即通过单向的transformer模型来训练语言模型，也就是只能通过前面的词来预测后面的词，相比ELMo可以建模双向的信息来说，这种做法可能遗漏了部分信息。

针对上面的问题，Devlin等人提出了BERT（Bidirectional Encoder Representations from Transformers），即基于transformer的双向编码器表示。BERT和GPT的结构和方法十分相似，最主要的不同之处在于GPT模型使用的是单向语言模型，可以认为是基于transformer的解码器表示，而BERT使用的基于transformer的编码器能够对来自过去和未来的信息进行建模，能够提取更丰富的信息，如图（2.13）所示。但如果简单使用编码器建模也存在着一个重大的问题，即语言模型可以在输入中看到自己的预测结果，从而可能导致网络并不会进行体征提取，而是简单选择输入中的词进行输出。

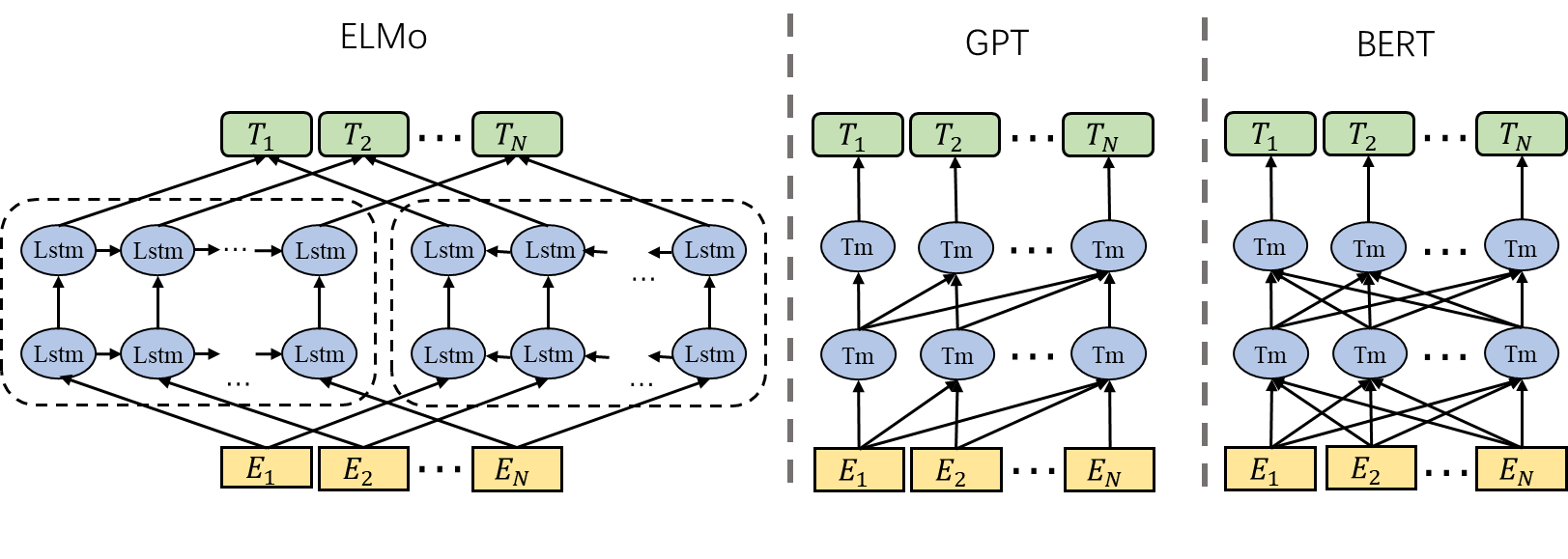


图2.13 ELMo、GPT和BERT

Fig. 2.13 ELMo, GPT and BERT

为了解决这个问题，BERT使用MLM（Masked Language Model）进行训练，即屏蔽语言模型进行训练。在预训练语言模型的过程中，会随机将输入中的部分词屏蔽，将其转换为[MASK]标记，从而让模型在输出层预测屏蔽位置所对应的词。通过这种做法，模型需要充分提取上下文中的信息，来预测屏蔽词。

除了MLM，BERT的另外一个训练目标是NSP（Next Sentence Prediction），即下一个句子的预测，自然语言处理领域的很多任务需要句子间的信息，比如问答和自然语言推理任务等。BERT在输入中给定两个句子，通过分隔符分开，模型预测这第二个句子是否是第一个句子的下一句，通过这种办法，BERT也拥有了获取句子间联系的能力。

和GPT十分相似，BERT在微调的时候，也是通过预训练的参数来初始化新的模型，通过下游任务特定的数据对模型参数进行少量的finetuning，这种方法加速了下游任务模型的收敛速度，而且可以使其达到更好的性能。

BERT模型之所以能够有效的一个重要原因，是使用了海量的单语数据，在预训练阶段，BERT使用了BooksCorpus数据集（8亿单词）和英语Wikipedia数据集（25亿单词），海量的单语语料中包含着丰富的语言学信息，而且经过充分的训练，预训练模型会有很强的鲁棒性和提取能力，从而可以提供给下游任务进行微调。

2.4 本章小结

本章主要介绍了机器翻译的发展过程，机器翻译从基于规则的方法，发展到基于统计的方法，以及现如今广泛使用的神经机器翻译，翻译性能有了明显的提高。

神经机器翻译基于编码器-解码器框架，编码器对源语言序列的信息进行抽取，将其编码为固定维度的向量表示，解码器根据编码器生成的向量表示，解码得到目标语言序列。为了增加解码状态和源语言的关联，神经机器翻译引入了注意力机制，在解码器端，每个时序的解码状态和源语言隐藏状态计算得到关联程度的概率分布，经过加权求和得到语义向量，从而避免了解码器只能依赖于固定维度的向量表示进行预测。

神经机器翻译模型的网络结构也不断变化，从最早基于循环神经网络的神经机器翻译，发展到完全基于自注意力机制的神经机器翻译，完全基于自注意力机制的模型可以捕获长距离依赖，并且并行程度更高，训练速度更快，可以得到更好的性能。因此，目前广泛使用基于自注意力机制的神经机器翻译模型。

最后，本章介绍了预训练方法在自然语言处理领域的发展，从早期基于词嵌入的预训练方法，发展到目前比较流行的基于语言模型的预训练方法，在诸多下游任务中都取得了优异的性能。因此，本文将预训练方法应用到神经机器翻译模型中，用于提高翻译模型的性能。

第3章 基于编码器的预训练

3.1 研究动机

现如今，神经机器翻译已经取得了巨大的提升，相比之前基于规则和基于统计的机器翻译模型，神经机器翻译可以实现更好的翻译质量，翻译结果更加流利通顺。然而，神经机器翻译存在的一个问题是对数据的依赖性极强，如果有足够的训练数据，即源语言和目标语言的双语语料，那么翻译模型就可以达到很好的性能，但在训练数据稀缺的情况下，翻译模型便无法达到理想的效果，如图（3.1）所示。稀缺资源场景一般可以分为语种数据稀缺和领域数据稀缺。对于训练数据十分充分的语种，我们一般称之为富资源语种，比如汉语-英语、英语-德语、英语-法语等，但是相比现在世界上数千种语言，富资源语种只占其中的很小一部分，其他语种大都是只有少量双语语料甚至没有双语语料的稀缺资源语种。而不同领域的数据量也存在着很大区别，比如新闻、口语等领域的数据一般较为容易获得，而一些专业领域，比如航空、考古学等，则很难获取到丰富的训练数据。

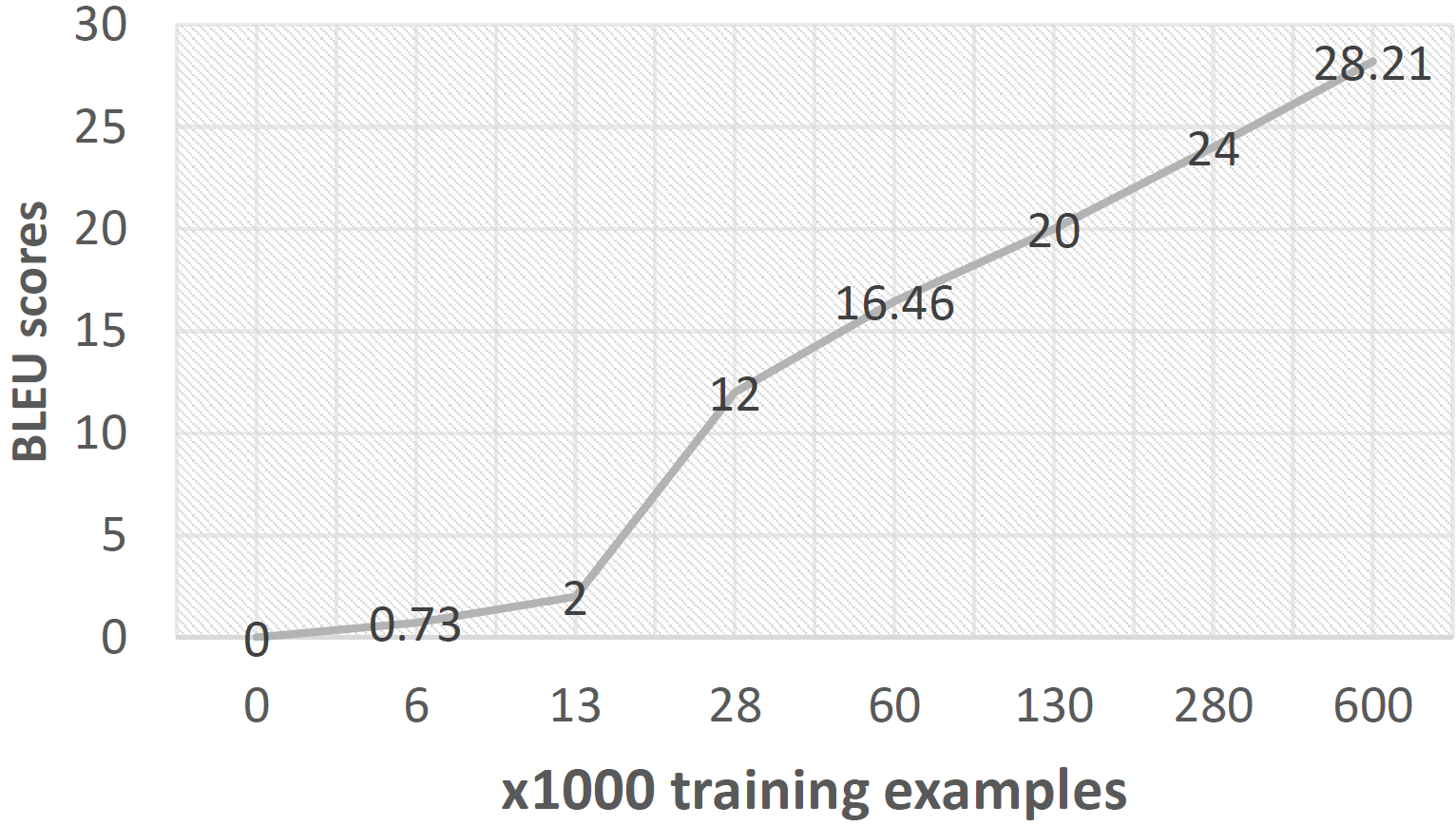


图3.1 翻译性能和训练数据量的关系

Fig. 3.1 Relationship between translation performance and the amount of training data

针对稀缺资源场景，想要提升翻译质量的一个有效方法是提高训练数据的数据量，但是由于该方法所需的人力和时间成本都很高，需要专业的翻译人员进行人工翻译，所以该方法实际上很难直接实现。因此，研究人员们主要从技术角度，尝试多种方法来提高稀缺资源的机器翻译质量。

在统计机器翻译时代，使用海量的单语语料来训练语言模型可以有效提高翻译结果的流畅性。而现如今普遍使用的神经机器翻译模型基于端到端的结构，对于一个输入句子，模型输出一个输出句子，神经机器翻译模型不需要统计机器翻译模型繁琐的流程，不需要人工设计特征，也就没有了显式的语言模型。研究人员尝试将语言模型集成到神经机器翻译模型的预测模块，但并未取得理想的性能。目前，单语数据比较有效的使用方法是进行数据增强，通过生成伪数据来增加训练数据的数量，其中最常用的数据增强技术是回译（back translation）。回译首先使用真实的双语数据训练得到反向翻译模型，通过该模型将目标语言的单语数据翻译为源语言，通过这种方式生成的源语言句子中会包含一些噪声，但目标语言句子可以保持通顺流畅，将生成的源语言句子和原始的目标语言句子组合成为伪数据，混合伪数据和真实的平行语料进行训练可以得到更好的翻译模型。通过对上述方法进行不断迭代，每次都可以生成质量更好的伪数据，因此可以有效提高模型的翻译性能。

本文从预训练的角度利用单语数据来提高神经机器翻译模型的性能。目前已有研究人员尝试将预训练方法应用于神经机器翻译模型[43]-[44]，取得了不错的效果，证明了预训练方法在机器翻译任务上的潜力，但目前针对预训练的研究仍处于初步阶段，仍有很大的进步空间。

前面说到，预训练的语言模型在自然语言处理领域已经取得了巨大的进展，在诸多任务上都取得了最优的成绩，包括命名实体识别、文本蕴含、问答、语义角色标注、指代消解和情感分析等，然而这些任务和神经机器翻译有着一个重大的区别，即这些任务属于自然原因理解任务，仅仅需要一个编码器就可以提取输入句子中的信息，而机器翻译属于自然语言生成任务，基于编码器-解码器框架，编码器读取一个源语言句子，提取其中的信息，并将其编码为一个向量表示，解码器对编码器提供的向量表示进行解码，将其生成目标语言句子，如图（3.2）所示。神经机器翻译模型仅仅依赖一个编码器不足以获取跨语言之间的信息并生成目标语言句子，因此预训练模型无法直接有效地应用于机器翻译任务中。

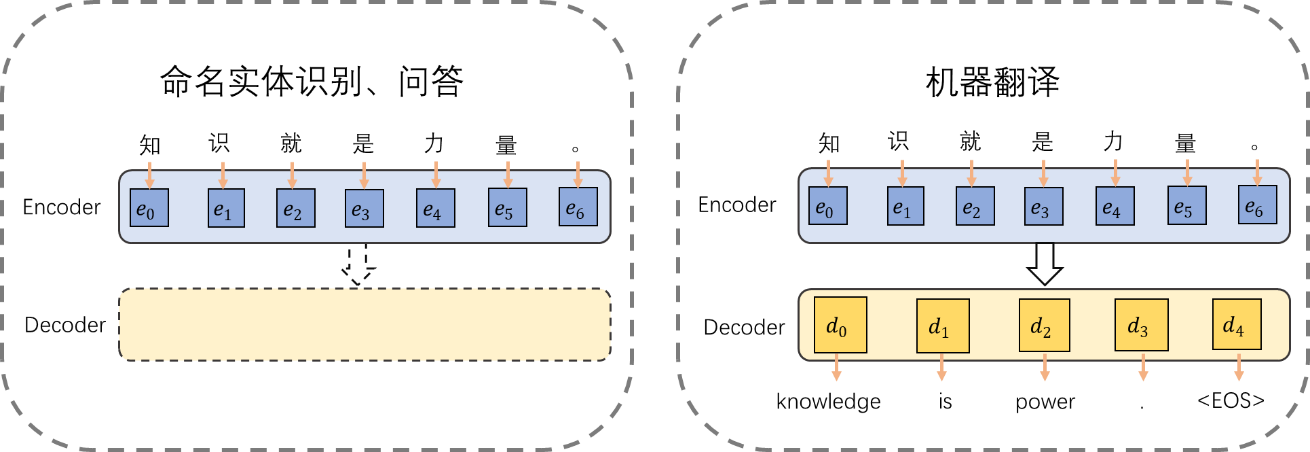


图 3.2 机器翻译和其它任务的模型结构

Fig. 3.2 Model structure for machine translation and others tasks

虽然基于语言模型的预训练模型和神经机器翻译模型的网络结构无法完全匹配，但是预训练模型可以看作是神经机器翻译模型结构的一部分，如果可以利用使用大量单语数据预训练的语言模型，那么神经机器翻译模型便能够拥有更充分的信息以及经过加工处理后的特征，对翻译任务可能很有帮助，尤其是对稀缺资源场景，少量的训练数据很可能无法将模型训练充分，很容易产生过拟合问题。因此，如果能够基于预训练的模型作为翻译模型的起点，或是通过预训练模型为翻译模型提供一些信息，对于稀缺资源场景的机器翻译来说是很有帮助的。因此，本章将预训练的语言模型以多种方式融入到神经机器翻译模型中，以此提高模型的翻译质量。

前面说过，现有的预训练的语言模型主要包括ELMo、GPT和BERT三种，而研究人员们一般认为，BERT是一个里程碑式的预训练模型，在多个数据集上都取得了最佳效果，而且使用起来更为方便。因此，在本章中我们研究如何将预训练模型BERT集成到神经机器翻译模型中。集成方法具有通用性，针对其他的预训练模型，也都可以采用相同的思想进行操作。

3.2 方法描述

针对面向自然语言理解任务的预训练模型，本章提出了三种集成方法将其集成到神经机器翻译模型的编码器中，分别是基于词嵌入的集成方法，基于隐藏层特征的集成方法和基于编码器微调的集成方法，从而为编码器提供更丰富的输入信息或增强编码器的建模能力。本小节分别针对这三种集成方法进行详细描述。

3.2.1 基于词嵌入的集成方法

在自然语言处理任务中，词嵌入是表示词汇信息的一种常用方法，我们一般认为，词嵌入中包含了词汇的基本信息，神经机器翻译模型基于输入句子的词汇信息，进一步提取句法、语义等信息，并学习跨语言之间表示的映射，来完成机器翻译任务。因此，词嵌入是神经机器翻译模型的基础。预训练模型基于海量的单语数据进行训练，词嵌入可以包含更加丰富准确的信息，因此我们将预训练模型的词嵌入集成到神经机器翻译模型中，为模型提供更有效的输入信息，从而提高神经机器翻译模型的翻译性能。

本章主要通过三种方法来集成预训练模型的词嵌入，分别是基于词嵌入替换的集成方法、基于词嵌入平均的集成方法和基于词嵌入融合的集成方法。三种方法从不同的角度将预训练模型的词嵌入与随机初始化的词嵌入结合，从而达到更好的性能。具体方法如图（3.3）所示。

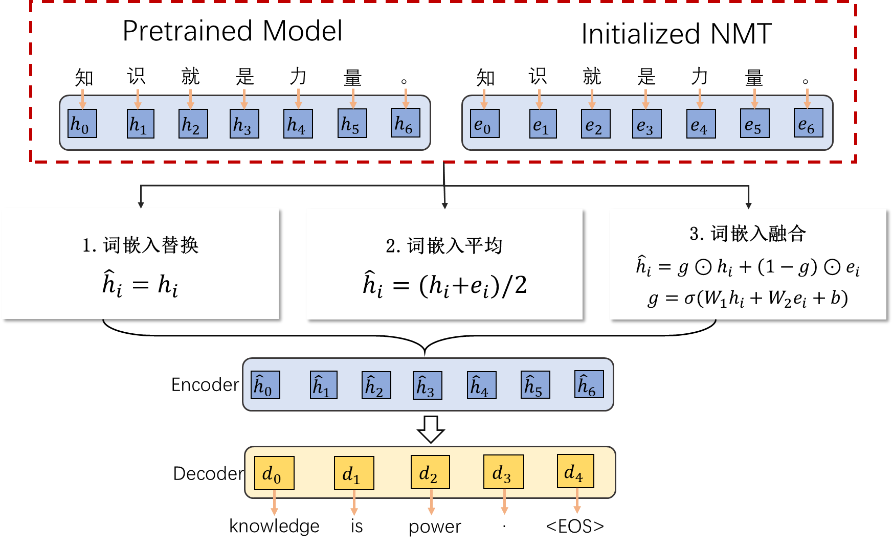


图 3.3 基于词嵌入的三种集成方法

Fig. 3.3 Three integration methods based on word embedding

三种方法在思路上都十分简单，而且容易操作。将模型自身随机初始化的词嵌入表示为，将预训练的词嵌入表示为，模型最后使用的词嵌入表示为。

神经机器翻译模型的词嵌入默认随机初始化，从零开始学习，基于嵌入替换的集成方法使用预训练模型的词嵌入来初始化，如公式（3.1）所示。

基于词嵌入平均的集成方法使用预训练模型的词嵌入和神经机器翻译模型随机初始化进行学习的词嵌入进行平均，如公式（3.2）所示。

基于词嵌入融合的集成方法则是使用预训练模型的词嵌入和神经机器翻译模型随机初始化进行学习的词嵌入进行加权融合，如公式（3.3）所示。

其中，是一个门控单元，计算方法如公式（3.4）所示。

其中，、和都是可学习的参数，是sigmoid函数，将计算结果缩放到之间。通过对两个词嵌入进行加权求和，来加强模型的输入，提供更丰富的信息。

3.2.2 基于隐藏层特征的集成方法

前文中提到的ELMo基于特征融合的方法在下游任务中进行应用，预训练阶段完成之后，对于下游任务中的一个句子，可以通过预训练模型对其进行编码，提取句子中的特征，将其得到的隐藏层的表示进行加权求和，最后得到的表示作为下游任务的一个输入，为下游任务提供经过加工处理后的丰富信息。

神经机器翻译模型经过多层的网络传播，不同的隐藏层也会提取不同的特征。我们一般认为，神经网络的底层会提取一些比较通用的特征，而模型顶层会提取一些相对抽象的特征。因此除了将预训练模型的词嵌入集成到模型中，还可以利用预训练模型来提取输入句子中的特征，然后将该特征作为一个强化的输入信息送给神经机器翻译模型。本文参考ELMo提出的方法，对预训练模型不同层的特征进行加权求和，得到更丰富的表示内容，并乘以一个缩放系数，使预训练模型的特征和神经机器翻译模型的特征在一个相同的量级上，如图（3.4）所示。

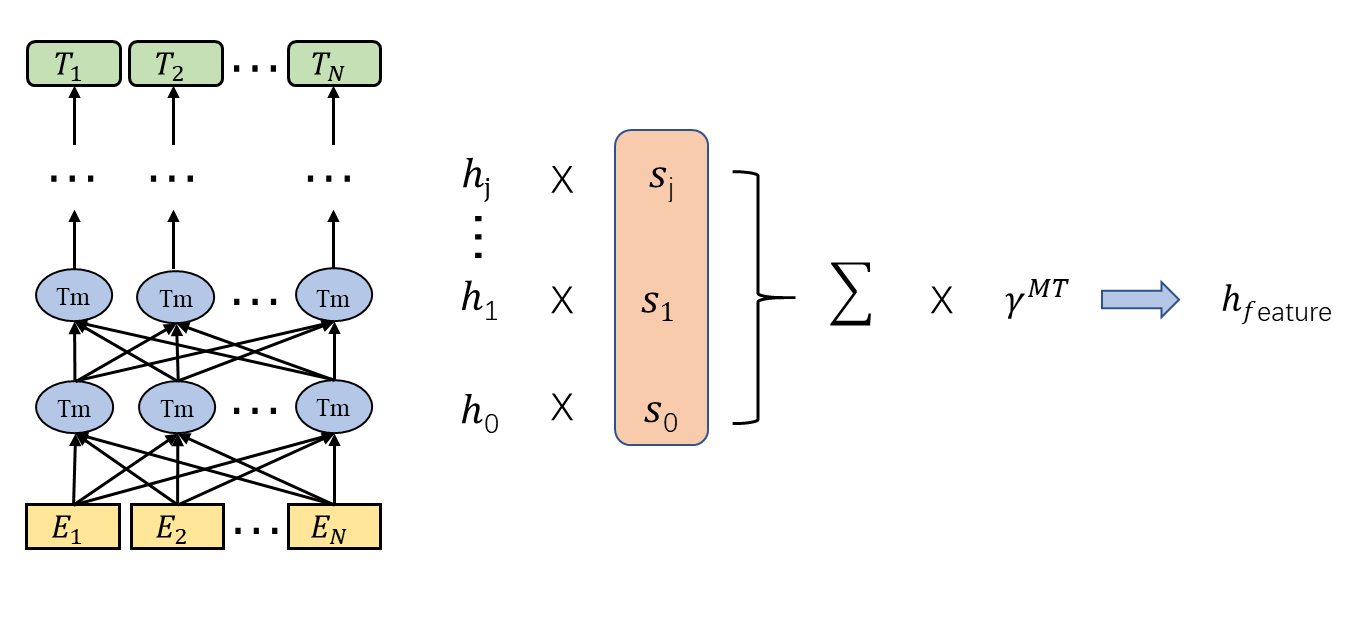


图 3.4 基于隐藏层特征的集成方法

Fig. 3.4 Integration method based on hidden layer features

第个词的特征提取方法如公式（3.5）所示。

其中，是预训练模型的隐藏层特征，是隐藏层的层数，是归一化的概率分布，用来对多个隐藏层的特征进行加权求和，是缩放系数，用于将提取到的特征缩放到翻译任务的特征范围内。

针对神经机器翻译模型，我们基于上述思想和方法，将动态的隐藏层特征集成到神经机器翻译模型的词嵌入中。我们在上一小节将预训练模型的词嵌入集成到编码器的词嵌入层，可以看作是对底层特征的一个集成，而本小节将预训练模型的隐藏层特征集成到词嵌入层，可以看作是对抽象特征的一个集成，隐藏层特征的集成方法同词嵌入的集成方法相同，也是基于替换、平均和融合的方法。

3.2.3 基于编码器微调的集成方法

不同于ELMo采用特征融合的方法，后来提出的预训练方法GPT和BERT均采用微调的方法，即通过海量的单语数据预训练模型，然后针对不同的下游任务，通过任务特定的训练数据对预训练模型进行微调，使模型更拟合该任务。因此，本文也实验了通过微调的方法将预训练模型集成到神经机器翻译模型当中。

目前已经有很多实验证明，在神经机器翻译模型中，相比解码器，编码器的作用更为重要[45]-[46]。对于一个源语言句子，编码器会对句子中的信息进行提取，得到该句子对应的表示，然后解码器根据该表示进行解码，生成目标语言的句子。如果编码器提取的信息不准确，或者是不够充分，那么解码器也无法生成高质量的翻译结果。目前的很多工作也是在编码器端进行，尝试去提高编码器的建模能力，甚至有实验证明，在解码器端即使不使用自注意力机制的模型也能达到很好的性能。

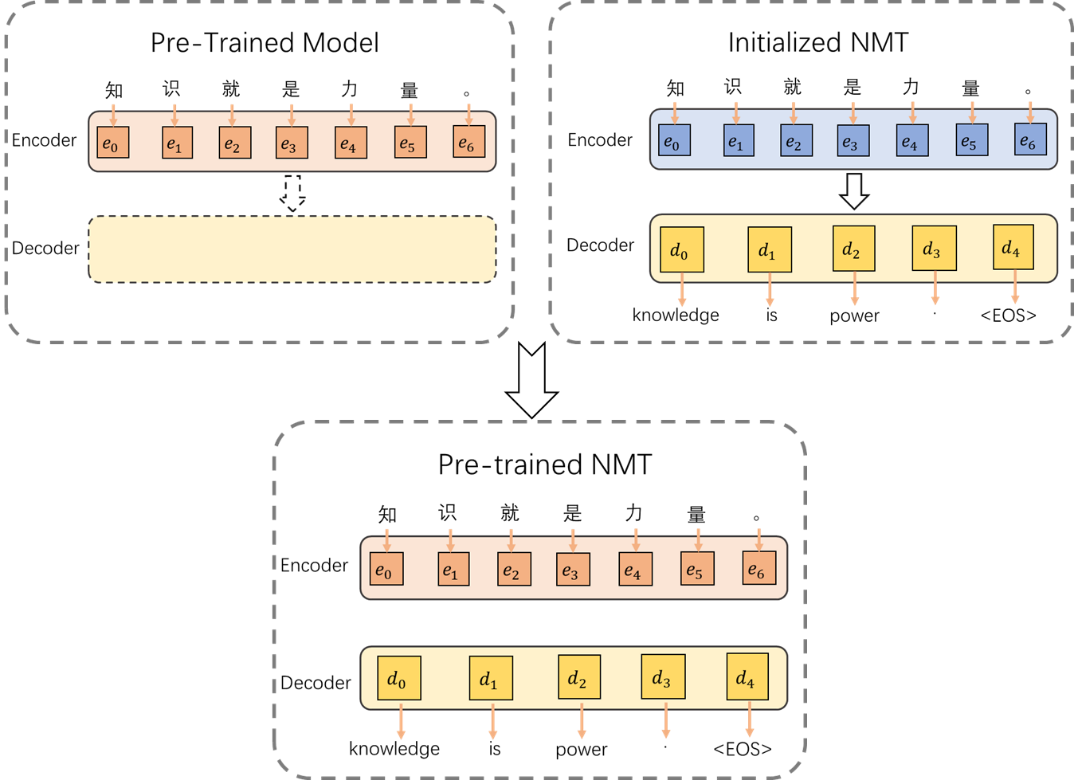


图3.5 基于编码器微调的集成方法

Fig. 3.5 Integration method based on encoder fine-tuning

本小节提出的方法仍然十分简单，神经机器翻译模型基于编码器-解码器框架，而BERT是基于编码器的预训练语言模型，因此我们可以简单使用BERT的参数来初始化神经机器翻译模型的编码器，而解码器仍然随机初始化参数，如图（3.5）所示。通过这种方法，相当于拥有了一个增强的编码器，该编码器拥有更强的提取能力，对于稀缺资源任务的帮助更加明显。

3.3 实验

3.3.1 实验设置

在本节中，对上述提出的方法主要在预训练模型BERT和iwslt德英方向的机器翻译任务上进行验证，iwslt（International Workshop on Spoken Language Translation）是国际上顶级的口语机器翻译评测，从2004年开始举办，吸引了世界范围内机器翻译领域的多个知名企业和机构参加。

BERT在上文中已经提到，2018年由Devlin等人提出，在自然语言处理的多个任务上取得了最好的成绩，是现如今比较流行的预训练模型，相比ELMo和GPT更为强大，因此本文选择BERT作为预训练模型。

iwslt德英方向的翻译任务是口语数据集，训练集双语数据的数据量为160232，校验集双语数据的数据量为7283，测试集双语数据的数据量为6750。其中，校验集数据是从官方提供的数据中进行不放回地抽取，校验集数据只作为判断模型是否收敛的标准，并未进行其他调整。之所以选择该数据集，是因为很多研究人员基于该数据集进行实验，有较好的参数配置和基线分数，可以更合理地对比本方法带来的提升。

对于实验使用的评价方式，本文基于机器翻译任务目前比较流行的自动评价方法BLEU（Bilingual Evaluation Understudy）进行评分[47]，该方法通过统计参考译文和机器翻译译文中n-gram共同出现的次数，来衡量两者的相似性，一般认为BLEU分数越高，机器翻译的质量越高，本文使用开源的bleu计算工具multi-bleu.perl作为打分工具。

在数据预处理方面，本文使用开源工具moses作为基础分词工具，之后使用分别使用BPE[48]和wordpiece工具对数据进行细粒度的切分，将词切分为更小单位的子词来提高机器翻译性能，其中BPE的merge次数为10000，最终得到的源语言词表大小为8836，目标语言词表大小为6636，共享词表的情况下词表大小为10140；wordpiece使用BERT提供的词表进行切分，最终得到的源语言词表大小为16604，目标语言词表大小为16132，共享词表的情况下词表大小为20604。

在系统实现方面，本文使用开源的神经机器翻译系统fairseq进行开发，该系统基于深度学习框架pytorch进行研究，是用于深度学习研发的一个开源库，支持语言模型、机器翻译等任务，系统已经实现了基于卷积的神经机器翻译模型和基于自注意力机制的神经机器翻译模型，而且易于修改，因此本文使用该系统进行开发。

为了加速神经机器翻译模型的计算速度，我们使用带有GPU（Graphics Processing Unit，即图形处理器）的服务器进行训练和解码（本章实验均使用单个GPU进行计算），具体信息如表（3.1）所示。

表3.1 服务器配置

Table 3.1 Server configuration

|  |  |
| --- | --- |
| **组件** | **配置** |
| 中央处理器 | Intel(R) Core(TM) i7-6850K CPU @ 3.60GHz |
| 内存 | 128GB |
| 图形处理单元 | NVIDIA 1080TI |
| 显存 | 11GB |
| 硬盘 | 3TB |

对于预训练模型BERT，我们使用BERT提供的预训练好的模型BERT-base-multilingual-cased，该模型在104中语言的单语数据上进行训练，具体参数配置如表（3.2）所示。

表3.2 预训练模型参数配置

Table 3.2 Pretrained model parameter configuration

|  |  |
| --- | --- |
| 参数 | 值 |
| 隐藏层大小 | 768 |
| 过滤器大小 | 3072 |
| 隐藏层层数 | 12 |
| 自注意力层头数 | 12 |
| 前馈层激活函数 | gelu |
| 最大句长 | 512 |
| 词表大小 | 119547 |

因为预训练模型的词表过大，因此在处理双语数据时，并不直接使用预训练模型的完整词表，如果双语数据中的词出现在预训练模型的词表中，则将其加入到翻译任务的词表中，如果不出现则将词转换为UNK，最终保证翻译任务的词表是预训练模型词表的子集，然后根据翻译任务的词表，挑选对应的词嵌入，通过这种方法，可以有效减少显存消耗。

对于神经机器翻译模型，我们使用了两个不同的参数配置，分别称为iwslt\_base参数配置和iwslt\_big参数配置。基于词嵌入和隐藏层特征的集成方法只需要模型的隐藏层大小和BERT的隐藏层大小一致，对网络的其他参数没有约束，因此我们使用参数量更少的iwslt\_base参数配置，该配置参考了在该数据集上的标准配置，保证可以达到一个合理的基线分数。基于编码器微调的集成方法需要神经机器翻译模型的结构和预训练模型的结构一致，为此我们采用iwslt\_big参数配置，如表（3.3）所示。虽然使用了不同的参数配置，但是我们的实验结果证明了预训练模型在不同参数量的模型上都得到了显著提升，最后也在相同的模型上，对比了三种方法带来的性能提升。

表3.3参数配置

Table 3.3 Parameter configuration

|  |  |  |
| --- | --- | --- |
| 参数 | iwslt\_base | iwslt\_big |
| 隐藏层大小 | 768 | 768 |
| 过滤器大小 | 1536 | 3072 |
| 隐藏层层数 | 6 | ? |
| 自注意力层头数 | 6 | 12 |
| 批大小 | 4096 | 4096 |
| 学习率 | 0.001 | 0.0005 |
| warmup步数 | 8000 | 4000 |
| 标签平滑 | 0.1 | 0.1 |
| 前馈层激活函数 | relu | gelu |
| dropout[49] | 0.3 | 0.3 |
| 批正则化[50] | before | after |
| 自注意力dropout | 0.1 | 0.0 |
| 激活函数dropout | 0.1 | 0.0 |
| SGD算法 | Adam | Adam |
| Adam\_beta1 | 0.9 | 0.9 |
| Adam\_beta2 | 0.997 | 0.98 |
| 最大句长 | 512 | 512 |
| 更新次数 | 50000 | 50000 |

因为训练数据的数据量过少，很容易导致过拟合问题，因此本章实验使用单模型进行解码，根据校验集选择最佳的检查点，没有采用ensemble技术，解码的参数如表（3.4）所示。

表3.4 解码参数配置

Table 3.4 Decoding parameter configuration

|  |  |
| --- | --- |
| **参数** | **值** |
| 长度惩罚 | 1.0 |
| beam大小 | 5 |

3.3.2 实验结果和分析

3.3.2.1 基于词嵌入的集成方法

本小节主要通过实验分析基于词嵌入的集成方法在神经机器翻译模型上的作用。

首先，我们通过实验验证基于词嵌入的集成方法对神经机器翻译模型的作用，因为预训练模型基于wordpiece进行子词切分，而双语数据要保证和预训练模型一致的切分方式，这和目前比较流行的BPE的切分方式并不相同，因此我们也对比了使用BPE子词切分方式的基线结果。实验结果如表（3.5）所示。

表3.5 基于词嵌入的集成方法

Table 3.5 Integrated method based on word embedding

|  |  |
| --- | --- |
| **集成方法** | **测试集** |
| 基线(BPE) | 34.27 |
| 基线 | 32.75 |
| 基线+替换 | 33.69 |
| 基线+平均 | 33.28 |
| 基线+融合 | 33.74 |

从实验结果可以看出，预训练模型的三种词嵌入集成方法都可以有效提高神经机器翻译模型的性能。需要说明的是，使用BPE子词切分方法的基线要更高，一个可能的原因是该BPE的切分是在本数据集学习得到的，而wordpiece的切分方法是由预训练模型BERT在104种语言的单语数据上使用统一的词表学习得到的，与本实验使用的数据集学习到的词表存在着一定的出入，而且wordpiece的切分粒度更细，最后得到的词表相比BPE的词表更大，模型学习过程可能更加困难，因此分数会略低一些。

然而，正是因为预训练模型在多语言的单语数据上共同训练，预训练的词嵌入本身可能已经包含了一部分的跨语言信息，尤其对于英德这种字符表十分相似的语言，语言本身就有很大的关联性。因此，我们在训练过程中使用共享设置，即在处理数据的时候，采用源语言和目标语言共享相同词表的方法，在模型训练过程中使用权重共享，共享编码器和解码器的词嵌入以及Softmax层权重矩阵，从而增加编码器和解码器之间的联系。采用共享设置的实验结果如表（3.6）所示。

表3.6 共享设置下基于词嵌入的集成方法

Table 3.6 Integrated method based on word embedding under share setting

|  |  |
| --- | --- |
| **集成方法** | **测试集** |
| 基线(BPE) | 34.92 |
| 基线 | 34.76 |
| 基线+替换 | 36.36 |
| 基线+平均 | 36.42 |
| 基线+融合 | 36.42 |

从实验结果上可以看出，使用共享设置的神经机器翻译模型能够有效提高翻译性能。对比使用BPE和wordpiece子词切分的基线模型分数，可以看出使用BPE的切分方式在共享词嵌入的时候并没有带来很大提高，而使用wordpiece的切分方式在共享词嵌入时提升显著，这也证实了我们前面的猜想。此外，从实验结果上可以看出，在共享设置下，预训练模型的词嵌入带来的提升更为明显，这是因为预训练模型的词嵌入可以应用到模型的更多位置，从而为模型带来更大的帮助。

为了进一步验证模型性能提高的因素，我们进行了一组对比实验，分析涨幅到底是由于词表共享还是权重共享带来的，本组实验均基于替换的词嵌入集成方法，实验结果如表（3.7）所示。

表3.7 词表共享和权重共享的影响

Table 3.7 Effect of shared vocabulary and shared weight

|  |  |  |  |
| --- | --- | --- | --- |
| **集成方法** | **词表** | **权重** | **测试集** |
| 基线 | 不共享 | - | 32.75 |
| 基线 | 共享 | 不共享 | 34.17 |
| 基线 | 共享 | 共享 | 34.76 |
| 基线+替换 | 共享 | 共享 | 36.36 |

通过实验可以看出，仅仅使用共享词表的方式，模型达到了1.42的BLEU值提高，而同时共享词嵌入可以带来2.01的BLEU值提升，这说明共享的方式可以带来很大的涨幅，而且共享设置下使用预训练模型的词嵌入替换随机初始化的词嵌入，模型可以比最初的基线高出3.59个BLEU值。因此在后面实验中，我们一般使用共享词表和共享词嵌入的方法，基于更高的基线分数来证明预训练模型的有效性。

我们通过一组对比实验来证明预训练词嵌入的充分性。我们在共享设置下，对比了基线模型和基于词嵌入替换的集成方法分别进行参数更新和固定参数情况下的得分，实验结果如表（3.8）所示。

表3.8 预训练模型的词嵌入充分性分析

Table 3.8 Adequacy analysis of word embedding in pre-trained model

|  |  |
| --- | --- |
| **集成方法** | **测试集** |
| 基线 | 34.76 |
| 基线(固定参数) | 23.23 |
| 基线+替换(固定参数) | 34.96 |
| 基线+替换(参数更新) | 36.36 |

通过上述实验可以看出，预训练模型的词嵌入在不进行参数更新的情况下，翻译性能也可以得到有效提高，相比翻译模型随机初始化参数然后进行训练仍要高出0.2个BLEU值，如果使用随机初始化的词嵌入而不进行参数更新，会对模型的性能造成极大的伤害。这说明，预训练模型的词嵌入已经包含了十分丰富且准确的信息，通过在神经机器翻译模型上进行参数更新，可以对翻译任务拟合地更好，从而可以带来更高的性能提升。

为了进一步为了对比预训练模型的词嵌入对编码器和解码器的重要性，我们在不使用共享设置的条件下，我们对比了下面几组实验，实验结果如表（3.9）所示。

表3.9 预训练模型的词嵌入对编码器和解码器的影响

Table 3.9 Effect of word embedding in pre-training model on encoder and decoder

|  |  |
| --- | --- |
| **集成方法** | **测试集** |
| 基线 | 32.75 |
| 基线+替换编码器词嵌入 | 33.69 |
| 基线+替换解码器词嵌入 | 33.27 |
| 基线+编码器解码器均替换 | 34.4 |

实验结果表明，使用预训练模型的词嵌入替换编码器和解码器随机初始化的词嵌入对神经机器翻译模型都有着不同的贡献，替换编码器词嵌入相比替换解码器词嵌入可以带来更大的提升，编码器和解码器同时使用预训练模型的词嵌入时效果最好。

3.3.2.2 基于隐藏层特征的集成方法

在本小节，我们证明基于隐藏层特征的集成方法对神经机器翻译模型性能的提高，并对实验结果进行分析。

我们基于iwslt\_base参数配置和共享设置，分别提取预训练模型BERT的最顶层特征、最上面4层特征和最顶层特征，进行加权融合和缩放操作，然后集成到神经机器翻译模型的词嵌入中，实验结果如图（3.6）所示。

图 3.6 基于隐藏层特征的集成方法

Fig. 3.6 Integration method based on features of hidden layers

可以看出，基于隐藏层特征的集成方法可以有效提高模型的翻译质量。仅仅利用预训练模型的顶层特征，模型就可以达到显著的性能提升，随着隐藏层层数的增多，翻译模型的性能有了进一步提升，在集成12层特征的时候，相比使用较少层数的特征，性能有了明显提高，我们猜测可能是因为不同的隐藏层可以捕获到不同层次的信息，将其进行加权求和，对翻译性能可以起到很好的帮助。

3.3.2.3 基于编码器微调的集成方法

在前面两个小节，我们证明了基于词嵌入和隐藏层特征的集成方法在神经机器翻译模型上的有效性。在本小节，我们证明基于编码器微调的集成方法有效性，并对实验结果进行的分析。

我们基于前面描述的iwslt\_big参数配置和共享设置，在不同的模型层数上验证了基于编码器的集成方法的有效性，需要说明的是，如果编码器的层数小于预训练模型的层数，我们使用预训练模型的底层来初始化神经机器翻译模型的编码器参数。实验结果如图（3.7）所示。

图3.7 基于编码器微调的集成方法

Fig. 3.7 Integration method based on encoder fine-tuning

可以看出，随着编码器-解码器层数的增多，使用随机初始化参数的神经机器翻译模型的性能会出现先上升后下降的趋势，这是因为训练数据的数据量过少，而模型的参数量过大，很容易产生过拟合现象，模型无法收敛到很好的状态，性能反而会下降。但是如果使用预训练模型的参数来初始化编码器，模型的性能随网络层数增多不断增长，这说明预训练模型具备很强的特征提取能力和鲁棒性，在训练数据不充分的情况下，可以提高模型的泛化能力，避免过拟合的风险。

3.4 性能分析

上一节已经通过大量的实验证明了预训练模型对于提高神经机器翻译模型性能的帮助，在本节我们通过观察模型训练过程中的信息，分析预训练模型对收敛速度和鲁棒性的影响，并对三种集成方法进行对比分析。

3.4.1 预训练模型对模型收敛速度的影响

对于模型的收敛速度，我们主要观察训练过程中best\_loss这一指标的下降过程。best\_loss是模型在训练过程中，经过一定的步数或轮数，在校验集上计算得到的最优损失值，一般认为，best\_loss越小，模型的性能越好，best\_loss不再下降也是判断模型收敛的一项指标。

我们针对编码器、解码器均为四层的网络结构，对比模型随机初始化和基于编码器微调的集成方法的实验结果，观察它们的收敛速度。best\_loss随训练步数的变化如图（3.8）所示。

图3.8 预训练模型对收敛速度的影响

Fig. 3.8 Effect of pre-trained model on convergence rate

通过上图我们可以发现，基于预训练编码器的模型相比基线模型可以达到更快的收敛速度，而且可以达到更好的性能。这是因为预训练的模型本身已经拥有了很强的特征提取能力，针对翻译任务，模型可以基于之前的经验继续学习，因此可以快速收敛，并且达到更好的收敛状态。

3.4.2 预训练模型对模型鲁棒性的影响

因为预训练模型BERT的层数是12层，因此可以进一步验证集成不同的编码器层数对模型性能的影响。我们基于iwslt\_big模型参数，固定解码器层数为4层，仅仅改变编码器的层数，同时为了消除解码器端的影响，我们不使用共享的设置，对比随机初始化参数和基于编码器的集成方法的实验结果，如图（3.9）所示。

图3.9编码器层数的影响

Fig. 3.9 Effect of the number of encoder layers

通过上述实验可以发现，随着编码器层数的提升，使用随机初始化参数的神经机器翻译系统的性能在层数较少时不断上升，但在达到一定层数的时候，模型性能会迅速下降，但集成了预训练模型编码器的神经机器翻译模型性能几乎是不断提升的，仅在最后达到12层的时候，性能有了下降的趋势。该实验证明了预训练模型的每一层都具有一定贡献的，可能包含了不同层次的信息。而且预训练模型的鲁棒性很强，在数据量较少的数据集上也可以避免过拟合的风险，从而使翻译性能得到很大的提升。

3.4.3 三种预训练模型集成方法的对比

最后，我们基于iwslt\_big参数配置和共享设置，使用编码器解码器各4层的网络结构，对比随机初始化参数、基于词嵌入替换的集成方法、基于最上面4层隐藏层特征替换的集成方法和基于编码器微调的集成方法，实验结果如表（3.10）所示。

表3.10 三种预训练模型的集成方法对比

Table 3.10 Comparison of integration methods of three pre-training models

|  |  |
| --- | --- |
| **集成方法** | **测试集** |
| 基线 | 34.15 |
| 基线+词嵌入 | 35.04 |
| 基线+隐藏层特征 | 35.26 |
| 基线+编码器 | 35.12 |

通过观察可以发现，在编码器层数较少时，基于隐藏层特征的集成方法可以达到最好的效果，因为预训练模型的词嵌入仅仅包含了部分信息，而基于编码器微调的集成方法只能使用预训练模型的底层参数，无法充分利用预训练模型的全部信息，而基于隐藏层特征的集成方法可以融合预训练模型所有隐藏层的信息，从而可以带来更显著的性能提升。

然而，在编码器层数较多时，基于编码器微调的集成方法可以带来更大的性能提升，因为在小数据集上，模型参数量巨大，如果随机初始化参数，很难训练充分，基于词嵌入和隐藏层特征的集成方法虽然可以为模型带来一部分额外信息，但是模型仍然具有庞大的参数量需要训练，很难达到收敛状态，而基于编码器微调的集成方法可以利用预训练模型的参数初始化编码器，从而减轻模型训练的压力，从而可以达到更好的性能。

3.5 本章小结

本章主要包括三部分内容：基于编码器的预训练的研究动机、三种预训练模型的集成方法、实验结果及分析。

首先，我们针对稀缺资源机器翻译的现状进行了分析，稀缺资源场景下只有少量的双语数据，但一般可以获取到丰富且高质量的单语数据，因此人们经常采用一些数据增强的技术，利用单语数据来提高模型性能。本章从另外一个角度出发，将使用单语数据预训练的模型集成到神经机器翻译模型的编码器中。

由于目前的预训练方法普遍基于语言模型任务，而语言模型和神经机器翻译模型的网络结构不匹配，我们提出了基于编码器的预训练，通过三种方法将预训练模型的信息集成到神经机器翻译模型的编码器端，分别是基于词嵌入的集成方法、基于隐藏层特征的集成方法和基于编码器微调的集成方法，三种方法都可以带来显著的性能提升。

我们通过实验验证了三种集成方法的有效性，实验表明，基于编码器微调的集成方法对神经机器翻译模型的性能提升最大，因为预训练模型的每一层都包含了丰富的特征，可以对神经机器翻译模型带来额外的信息。此外，基于编码器微调的集成方法在翻译质量、收敛速度、鲁棒性三个方面都得到了有效地提升。

下一章我们将研究基于编码器-解码器框架的预训练方法，使其无需进行任何修改，可以直接应用于神经机器翻译模型上。

第4章 基于编码器-解码器的预训练

4.1 研究动机

现如今，在自然语言处理领域，预训练方法一般基于语言模型任务，通过海量的单语数据进行预训练，然后应用到下游任务中。预训练方法在自然语言处理的多个任务上都取得了最优的成绩，因此在机器翻译任务上使用预训练方法是直觉的想法。但存在的问题是，神经机器翻译模型基于编码器-解码器框架，而现有的预训练语言模型如GPT可以看作是一个解码器，BERT使用的屏蔽语言模型则可以看作是一个编码器，网络结构并不匹配，因此现如今比较流行的预训练语言模型无法直接应用到神经机器翻译模型上。

在上一章节中，我们提出了基于编码器的预训练，通过三种方法将预训练模型BERT集成到神经机器翻译模型的编码器中，分别是基于词嵌入的集成方法、基于隐藏层特征的集成方法和基于编码器微调的集成方法，三种方法可以为翻译模型提供额外信息，或是提高编码器的特征提取能力，有效地提高了模型性能，加快了收敛速度，增强了模型的鲁棒性。但这种基于编码器的集成方法有一个重要的不足，即只能将预训练模型的部分信息应用到神经机器翻译模型中，或是只能应用到神经机器翻译模型的部分模块，无法初始化整个模型，网络仍有部分参数需要从零开始学习，限制了预训练模型的收益。针对这个问题，本章研究基于编码器-解码器框架的预训练方法，使预训练模型能够直接应用于神经机器翻译模型上。

在自然语言处理领域，基于编码器-解码器框架的模型一般应用于机器翻译、文本生成、智能对话等条件生成任务中，而预训练需要海量的数据，这意味着只能依赖单语数据进行训练，因为无法获得充分的标注数据。因为该预训练方法主要是面向机器翻译任务，因此预训练模型需要在多语言的单语数据上进行学习。针对以上条件和思想，面向机器翻译任务，本文提出了基于编码器-解码器的预训练方法，利用海量的文档级单语数据预训练文本生成任务。

4.2 方法描述

神经机器翻译模型基于编码器-解码器框架，编码器提取输入的源语言句子中信息，将其编码为一个向量表示，解码器根据编码器提供的向量表示生成目标语言句子，因此可以将解码器看作是一个条件语言模型，根据编码器提供的源语言表示，生成目标语言的内容。基于这个想法，我们可以使用类似的任务来预训练编码器-解码器模型。但是预训练方法一个必要条件是具备海量的单语数据，而不能仅仅依靠双语数据和标注数据，因此，本文提出了基于文本生成任务的预训练方法，即利用文档级单语数据，构建训练数据，输入为一个普通的句子，输出是该句子的下一句，分别简称为前句-后句，解码器可以基于编码器提取到的前句中的信息进行解码，生成后句的内容，如图（4.1）所示。

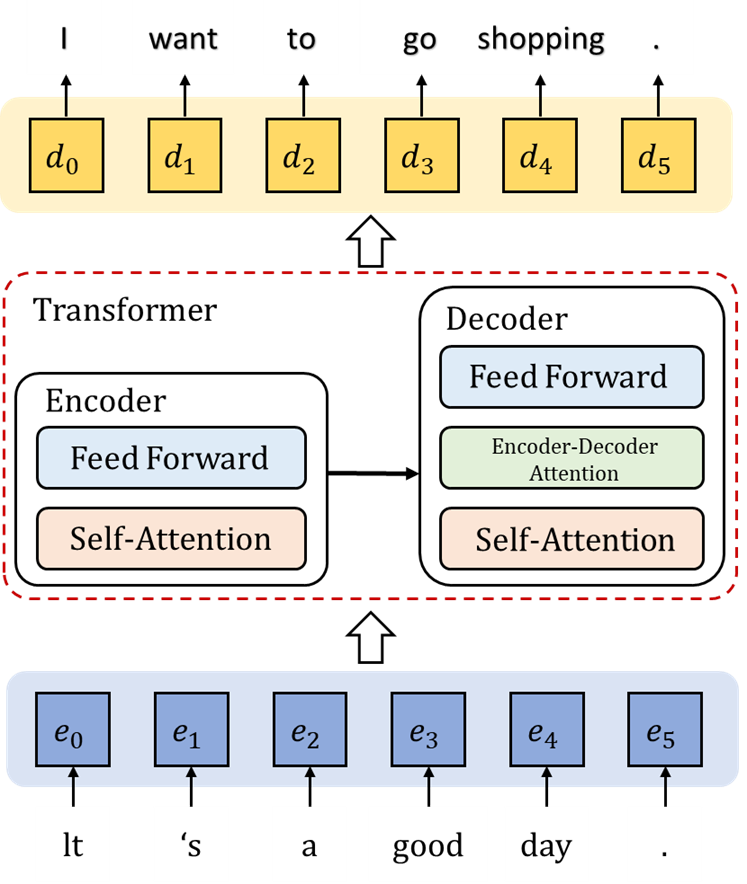


图4.1 基于文本生成任务的预训练方法

Fig. 4.1 Pre-trained method based on text generation task

翻译任务的训练目标是生成目标语言相同语义的内容，解码器的条件输入可以看作是输入句子的语义信息。而文本生成任务的训练目标将生成相同语言相关语义的内容，解码器的条件输入同样是输入句子的语义信息，因此，在模型的预训练阶段，可以使用文本生成任务的训练目标，编码器学习如何提取输入句子中的语义信息，解码器学习如何根据语义信息来生成输出结果。

然而，机器翻译任务需要学到跨语言之间的联系，仅仅基于文本生成任务进行预训练只能使模型具备一定的生成能力，而无法学习到不同语言之间的信息转换，针对这个问题，我们对多语言的训练数据使用词表共享和权重共享，共享设置可以减小不同语言间表示的区别，从而学习到跨语言之间的联系。然后，我们使用同一个模型同时预训练多语言文本生成任务，如图（4.2）所示，多语言的训练数据并不需要互译，只需要具备前句-后句的关系即可。通过这种方式，预训练模型不仅可以具备提取不同语言序列特征的能力，而且可以具备生成不同语言序列的能力。

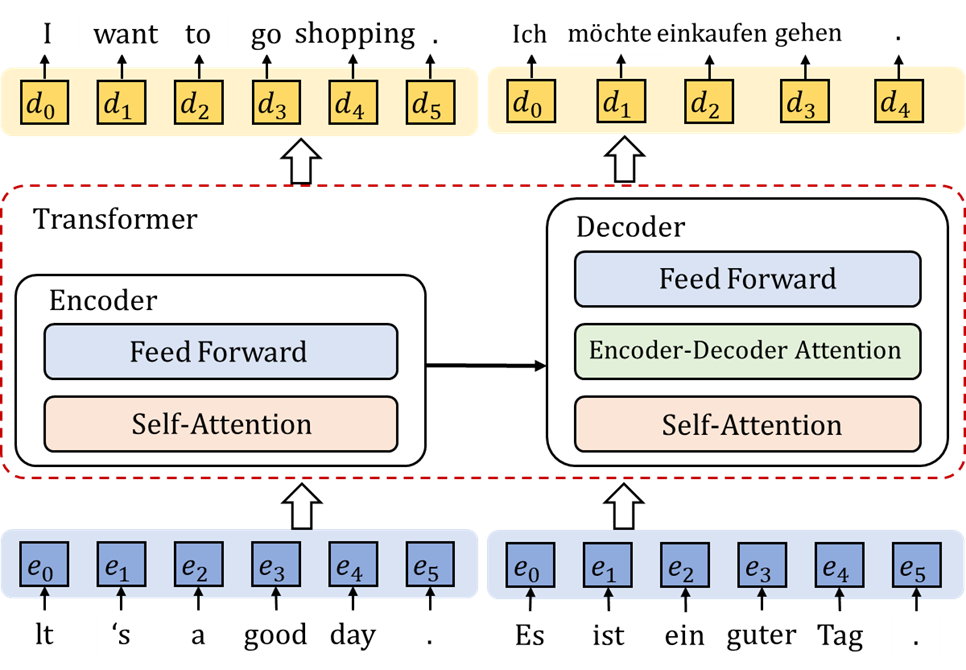


图4.2 基于多语言文本生成任务的预训练方法

Fig. 4.2 Pre-trained method based on multilingual text generation task

但是，这又带来了另外一个问题，我们希望语言之间的差异性减小，但同时也希望不同的语言可以保存语言自身的特性，而不是完全统一处理。针对这个问题，我们学习多语言神经机器翻译系统的思想[22]，在句子前面添加一个语言标识，表明这个句子的语言类别。通过这种方法，模型在提取句子特征的过程中，也可以针对不同的语言进行适度的区分，如图（4.3）所示。

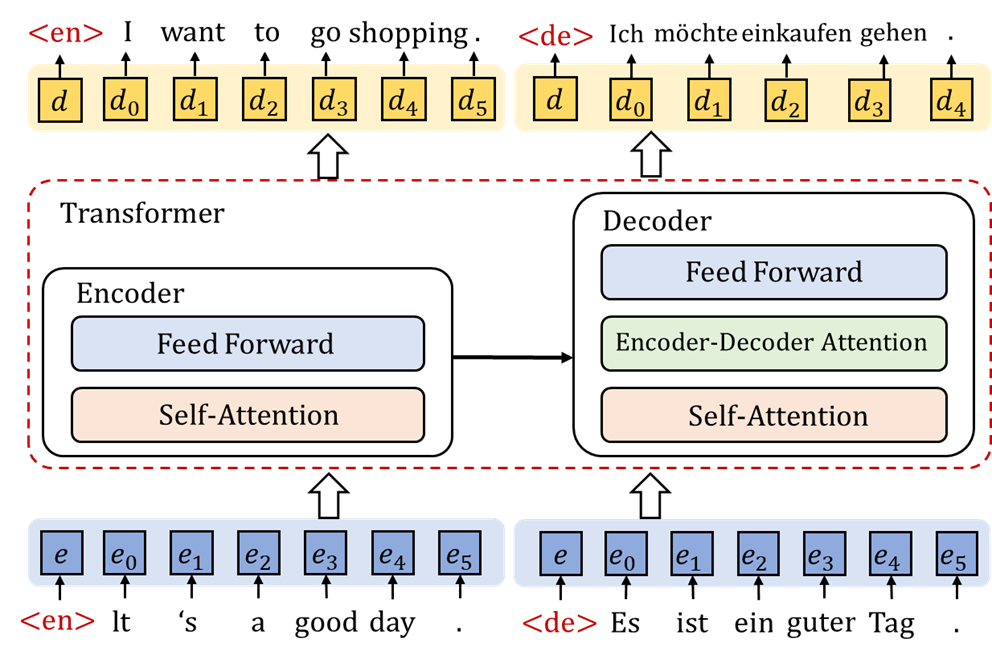


图4.3 加入语言标识的预训练方法

Fig. 4.3 Pre-trained method with language identification

另外，模型还有一个潜在的问题是，预训练阶段和翻译阶段有一个重要的区别，即预训练阶段模型只学习生成相同语言的内容，而翻译阶段则需要生成不同语言的内容，编码器在提取信息的过程中，无法得知解码器要生成何种语言的内容，在预训练阶段会引导模型朝着仅生成相同语言内容的方向学习，这和翻译阶段是不匹配的。针对这个问题，我们在编码器的词嵌入层加入一个语言嵌入[44]，表示解码器要生成的目标语言类别，如图（4.4）所示。

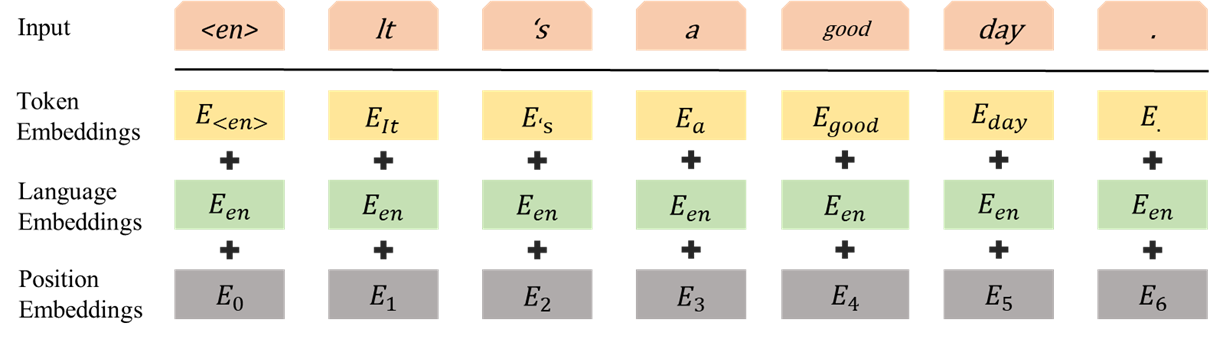


图4.4 预训练方法的输入表示

Fig. 4.4 Input representation of pre-trained method

预训练阶段完成之后，只需要保证双语数据的词表和预训练模型的词表一致，就可以直接利用预训练模型的参数初始化新的神经机器翻译模型，翻译模型不需要从零开始训练，而是基于学习文本生成任务的经验，在双语数据上进行新的学习，从而达到更快的收敛速度和更好的翻译质量。

4.3 实验

4.3.1 实验设置

本章提出的预训练方法理论上可以同时在多语言数据上学习，但由于数据和设备的限制，我们仅仅在英语和德语两个语言上进行实验，模仿稀缺资源场景实验，实验数据来自于WMT（Workshop on Machine Translation）提供的新闻数据集。WMT是国际上公认的机器翻译顶级评测，各大科技公司和高校参加该评测来展示自己在机器翻译任务上的实力，WMT的英德方向数据集也是目前英德机器翻译任务上使用最多的集合，诸多研究人员在提出了新的方法和技术，在该集合上进行验证，同时该数据集也提供了海量的单语数据，因此我们也使用该数据集来验证提出的预训练方法。

需要说明的是，之所以使用和上一章不同的实验数据，主要原因在于，上一章基于其他研究人员提供的预训练模型BERT，通过多种方法将其集成到神经机器翻译模型的编码器中，并不需要自己进行预训练，而IWSLT德英方向的数据集是广泛使用的一个数据集，拥有较好的参数设置和基线分数，因此在上一章中选择该数据集作为训练数据。而本章提出基于文本生成的预训练方法，需要通过海量单语数据预训练模型，而WMT提供了新闻领域英语和德语的海量文档级单语数据，可以用来预训练文本生成任务，而IWSLT是口语领域数据集，如果使用新闻领域的单语数据进行预训练，使用口语领域的数据进行验证翻译性能，领域差异可能会降低预训练方法带来的收益，导致无法判断方法的有效性，因此本章使用WMT数据集来验证提出的预训练方法。

WMT英德方向的新闻数据集包含海量的单语数据，由于设备限制，我们仅仅使用其中的部分数据来验证我们方法的有效性。我们使用2007年-2018年的news-docs文档数据集构造预训练模型所需要的训练数据，删除长度超过250的句子以及长度比超过1.5的句子，最后我们从中抽取出英语和德语各5000万条作为预训练阶段的训练数据。

对于翻译任务所需要的双语训练数据，我们使用WMT2018新闻数据集，使用除括ParaCrawl之外的其他双语数据集，删除长度超过250的句子以及长度比超过1.5的句子，最后剩余518万双语数据，但因为本文的目标是面向稀缺资源的机器翻译任务，因此并不会使用所有的双语数据，而是从中随机抽取一部分作为训练数据，从而模拟稀缺资源场景下的数据情况。我们从全部双语训练数据中不放回地抽取5000句作为校验集，校验集数据只作为判断模型是否收敛的标准，并未进行其他调整。我们使用官方提供的wmt14、wmt15、wmt16、wmt17和wmt18五个测试集，在多个集合上证明我们提出的预训练方法有效性。对于评价方式，使用机器翻译任务目前比较流行的自动评价方法BLEU（Bilingual evaluation understudy）进行评分，一般认为BLEU分数越高，机器翻译的质量越高，本文使用开源的bleu计算工具multi-bleu.perl作为打分工具。

对预训练任务和翻译任务的训练数据，我们使用开源工具moses作为基础分词工具，之后使用分别使用BPE对数据进行细粒度的切分，BPE的merge次数为32000。基于预训练任务的单语数据，我们采用词表共享的方式生成词表，然后应用到双语数据中，保证单语数据和双语数据的词表大小一致，最终词表大小为34636。

在系统实现方面，本章使用开源的神经机器翻译系统fairseq进行开发，该系统基于深度学习框架pytorch进行研究，是用于深度学习研发的一个开源库，支持语言模型、机器翻译等任务，系统已经实现了基于卷积的神经机器翻译模型和基于自注意力机制的神经机器翻译模型，而且易于修改，因此本文使用该系统进行开发。

为了加速神经机器翻译模型的计算速度，我们使用带有GPU（Graphics Processing Unit，即图形处理器）的服务器进行训练，具体信息如表（4.1）所示。在预训练阶段，因为数据量很大，因此我们使用八张GPU进行训练，加快训练速度，而在翻译模型的训练阶段，因为数据量很少，因此我们一般使用单张GPU进行训练。

表4.1 服务器配置

Table 4.1 Server configuration

|  |  |
| --- | --- |
| **组件** | **配置** |
| 中央处理器 | Intel(R) Core(TM) i7-6850K CPU @ 3.60GHz |
| 内存 | 128GB |
| 图形处理单元 | NVIDIA 1080TI |
| 显存 | 11GB |
| 硬盘 | 3TB |

我们通过Base参数配置进行预训练，参数配置如表（4.2）所示。训练翻译任务的参数配置和预训练模型的参数配置几乎完全相同，不同之处会在实验部分注明。

表4.2 预训练模型参数配置

Table 4.2 Pre-trained model parameter configuration

|  |  |
| --- | --- |
| 参数 | Small参数配置 |
| 隐藏层大小 | 512 |
| 过滤器大小 | 2048 |
| 隐藏层层数 | 6 |
| 自注意力层头数 | 8 |
| 批大小 | 4096 |
| 累积梯度 | 4 |
| 学习率 | 0.001 |
| warmup步数 | 8000 |
| 标签平滑 | 0.1 |
| 前馈层激活函数 | gelu |
| dropout | 0.1 |
| 批正则化 | before |
| 自注意力dropout | 0.1 |
| 激活函数dropout | 0.1 |
| SGD算法 | Adam |
| Adam\_beta1 | 0.9 |
| Adam\_beta2 | 0.997 |
| 最大句长 | 512 |

因为翻译任务训练数据的数据量过少，很容易导致过拟合问题，因此实验使用单模型进行解码，根据校验集选择最佳的检查点，没有采用ensemble技术，解码的参数如表（4.3）所示。

表4.3 解码参数配置

Table 4.3 Decoding parameter configuration

|  |  |
| --- | --- |
| **参数** | **值** |
| 英德方向长度惩罚 | 0.6 |
| 德英方向长度惩罚 | 1.6 |
| beam大小 | 4 |

4.3.2 实验结果

因为英语和德语属于相近语种，词汇表十分相似，因此可以使用同一个模型同时学习英语-德语和德语-英语的双向翻译。我们在预训练阶段，同时训练英语和德语两个语种的单语数据，因此预训练模型本身也具备对于英语-德语方向和德语-英语方向的建模能力。我们对两种语言的训练数据采用相同的预处理方法，并使用共享的设置，通过英语-德语方向和德语-英语方向的双语数据进行训练，两个方向的双语数据完全相同，只是语言方向相反。在解码阶段，无论模型的输入语言是英语或德语，模型均具备对应的翻译能力。因此，本章的实验结果均来源于双向的翻译模型。

首先，从处理后的WMT英语和德语文档级单语数据集中，随机抽取各5000万数据作为预训练任务的训练数据，基于Base参数配置预训练模型14万步。然后从处理后的WMT英语和德语双语数据集中随机 抽取10万双语数据作为翻译任务的训练数据，对比翻译模型随机初始化和使用预训练模型参数初始化的BLEU得分，其中基线模型使用的1e-3的学习率，基于预训练的翻译模型使用1e-4的学习率，batch大小均为4096，训练步数5万步，实验结果如表（4.4）所示。

表4.4 基于Small参数配置的BLEU得分（100k）

Table 4.4 BLEU scores based on Small parameter configuration (100k)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **模型** | **语言**  **方向** | **测试集** | | | | | |
| **wmt14** | **wmt15** | **wmt16** | **wmt17** | **wmt18** | **平均** |
| 基线 | 英德 | 15.04 | 16.85 | 19.57 | 16.15 | 22.40 | 18.00 |
| 德英 | 18.58 | 20.32 | 23.65 | 20.52 | 25.31 | 21.68 |
| 基线+预训练 | 英德 | 18.81 | 20.58 | 23.63 | 19.60 | 26.95 | 21.91 |
| 德英 | 22.85 | 24.05 | 28.64 | 25.43 | 30.13 | 26.22 |

从实验结果可以看出，使用预训练模型，在英德方向的五个测试集上平均带来了3.91个BLEU值的提升，在德英方向的五个测试集上平均带来了4.54个BLEU的提升，提升十分显著。

因为本文的目标面向稀缺资源场景，因此针对更少的双语数据，我们也进行了验证。使用10k双语数据的实验结果如表（4.5）所示。

表4.5 基于Small参数配置的BLEU得分（10k）

Table 4.5 BLEU scores based on Small parameter configuration (10k)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **模型** | **语言**  **方向** | **测试集** | | | | | |
| **wmt14** | **wmt15** | **wmt16** | **wmt17** | **wmt18** | **平均** |
| 基线 | 英德 | 2.04 | 2.57 | 2.56 | 2.34 | 2.9 | 2.48 |
| 德英 | 2.85 | 3.33 | 3.32 | 3.12 | 3.76 | 3.28 |
| 基线+预训练 | 英德 | 11.11 | 11.65 | 12.65 | 11.46 | 14.59 | 12.29 |
| 德英 | 14.97 | 15.78 | 17.41 | 16.68 | 19.01 | 16.77 |

使用5万双语数据的实验结果如表（4.6）所示。

表4.6 基于Small参数配置的BLEU得分（50k）

Table 4.6 BLEU scores based on Small parameter configuration (50k)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **模型** | **语言**  **方向** | **测试集** | | | | | |
| **wmt14** | **wmt15** | **wmt16** | **wmt17** | **wmt18** | **平均** |
| 基线 | 英德 | 10.37 | 12.29 | 13.83 | 11.20 | 15.41 | 12.62 |
| 德英 | 14.24 | 15.33 | 17.87 | 15.57 | 19.32 | 16.47 |
| 基线+预训练 | 英德 | 16.53 | 18.37 | 20.63 | 16.86 | 23.34 | 19.15 |
| 德英 | 20.67 | 21.68 | 25.12 | 22.87 | 26.95 | 23.46 |

最后，我们验证了预训练方法在100万双语数据上的实验结果。需要说明的是，上述实验在小数据集上进行，均使用单张GPU进行训练，训练步数为5万步，而在100万的数据集上，为了提高基线分数，使实验对比更加合理，我们使用八张GPU进行训练，训练轮数为21轮。实验结果如表（4.7）所示。

表4.7 基于Small参数配置的BLEU得分（1m）

Table 4.7 BLEU scores based on Small parameter configuration (1m)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **模型** | **语言**  **方向** | **测试集** | | | | | |
| **wmt14** | **wmt15** | **wmt16** | **wmt17** | **wmt18** | **平均** |
| 基线 | 英德 | 23.34 | 26.15 | 30.49 | 24.59 | 36.44 | 28.20 |
| 德英 | 27.81 | 29.34 | 34.43 | 30.06 | 36.58 | 31.64 |
| 基线+预训练 | 英德 | 23.81 | 27.01 | 30.57 | 24.91 | 36.46 | 28.55 |
| 德英 | 28.46 | 29.64 | 34.85 | 30.82 | 36.67 | 32.09 |

从上述实验结果可以看出，在双语训练数据越少的情况下，基于预训练的方法可以带来更大的性能提升。为了更方便的对比，不同大小的训练数据在英德方向五个测试集上的平均分数如图（4.5）所示。

图4.5 不同大小的训练数据的平均BLEU得分（英-德）

Fig. 4.5 Average BLEU scores of different sizes of training data（en-de）

不同大小的训练数据在德英方向五个测试集上的平均分数如图（4.6）所示。

图4.6 不同大小的训练数据的平均BLEU得分（德-英）

Fig. 4.6 Average BLEU scores of different sizes of training data (de-en)

由此可以看出，在训练数据很少的情况下，使用预训练方法可以带来极大的收益，验证了我们提出的预训练方法在稀缺资源机器翻译任务上的作用。

4.3.3 实验分析

4.3.3.1 学习率对模型性能的影响

在本小节，我们对比微调阶段的学习率对模型性能的影响，在神经机器翻译模型上，一般使用0.001的学习率训练基线模型，但是由于预训练模型已经达到了不错的收敛状态，如果使用大的学习率，有可能会破坏之前学习好的状态，因此，在基于预训模型的微调过程中，一般使用较小的学习率。

我们在10k的双语数据上对比了0.00005、0.0001、0.0002、0.0005、0.001五个学习率在五个测试集的平均分数，实验结果如图（4.7）所示。

图 4.7 学习率对模型性能的影响

Fig. 4.7 Effect of learning rate on model performance

从实验结果可以看出，在使用0.0001学习率的情况下，模型性能可以得到更大的涨幅。我们猜测认为，在预训练阶段完成后，模型已经具备了一定的能力，只需要在双语训练数据上进行微调，使预训练模型更拟合翻译任务，所以较小的学习率可以达到更好的收敛状态。如果在基于预训练的模型上使用较大的学习率进行微调，由于模型结构较为复杂，而且训练数据很少，模型可能无法收敛到最佳状态。

4.3.3.2 batch大小对模型性能的影响

在先前研究人员提出的预训练方法中，提到在微调阶段使用小的batch对下游任务更有帮助，而在神经机器翻译模型中，我们一般使用大小为4096的batch。本小节针对batch大小对翻译模型的影响进行实验，对比batch大小分别为1024、2048和4096时的模型性能，实验结果如图（4.8）所示。

图 4.8 batch大小对模型性能的影响

Fig. 4.8 Effect of batch size on model performance

从实验结果可以看出，使用比较大的batch可以带来更好的性能，这一结论符合神经机器翻译模型训练的常识，也说明了翻译任务和自然语言领域的其它任务存在着一定的区别。

4.4 本章小结

本章主要包括三部分内容：基于编码器-解码器的预训练研究动机、方法描述、实验结果和分析。

上一章提出了基于编码器的预训练，通过三种方法将现有的预训练模型集成到神经机器翻译模型的编码器中，显著提高了模型性能。但是这种集成方法仍然具有一些局限性，即模型仍有一部分参数需要从头学习，降低了预训练模型带来的收益。因此，本章提出了基于编码器-解码器框架的预训练方法，预训练阶段完成后，模型可以完整迁移到神经机器翻译模型中，无需从零开始学习。

本章提出的预训练方法基于编码器-解码器框架，利用海量的文档级单语数据，使用同一个模型训练多语言的文本生成任务，对不同语言通过起始符和语言嵌入来区分，这样在增加跨语言联系的同时，也保留了语言自身的特性。预训练阶段完成之后，模型参数可以用来初始化神经机器翻译模型，使翻译模型在开始阶段就可以具备特征抽取能力和跨语言建模能力，通过双语数据对模型微调，可以达到更好的翻译性能。

我们通过实验证明了该方法的有效性。我们在WMT新闻数据集上，抽取多个不同大小的双语数据，采用预训练方法的神经机器翻译模型性能相比基线有了显著提升，并验证了在双语数据越少的情况下，预训练方法带来的收益更大。

第5章 总结与展望

5.1 工作总结

随着相关技术的进步，机器翻译技术也得到了长足的发展。从上世纪70年代基于规则的方法，到80年代基于实例的方法，90年代基于统计的方法，到如今基于神经网络的方法，经过多年的技术积累和进步，机器翻译已经可以达到良好的翻译质量，为日常交流、国际合作等带来了极大的便利。神经机器翻译的成功离不开深度学习技术的发展，神经机器翻译模型基于编码器-解码器框架和注意力机制，网络结构从最早基于循环神经网络，发展到基于卷积神经网络和基于自注意力机制，模型的翻译质量已经得到了很大程度的提升。然而，基于数据驱动的神经机器翻译模型需要大规模的双语训练数据，但是存在着很多稀缺资源场景，如语种数据稀缺和领域数据稀缺等，无法获得充分的训练数据，如何利用少量的训练数据提高机器翻译模型的性能是目前的一个研究热点。因此，面向稀缺资源的神经机器翻译研究具有很大的理论意义和现实意义。

目前，面向稀缺资源的神经机器翻译一般基于数据增强的方法，即利用少量的双语数据和丰富的单语数据，生成伪数据来提高训练数据的数量，从而提高机器翻译模型的性能。本文从另外一个角度出发，基于海量的单语数据，使用预训练的方法来提高模型性能。基于语言模型的预训练方法在自然语言处理领域已经取得了很大的成功，在命名实体识别、指代消解、问答等任务上都达到了目前最好的成绩，但是预训练方法在机器翻译任务上的使用仍然很少。本文从两个方向出发，一是基于编码器的预训练，提出了三种方法来将现有的预训练模型集成到神经机器翻译模型，二是基于编码器-解码器的预训练，提出了面向自然语言生成任务的预训练方法，预训练模型可以简单有效地迁移到神经机器翻译模型上。

本文提出了基于编码器的预训练，提出了三种集成方法将现有的预训练模型集成到神经机器翻译模型的编码器中，分别是基于词嵌入的集成方法、基于隐藏层特征的集成方法和基于编码器微调的集成方法。基于前人已提出的预训练模型，三种方法分别将预训练模型的信息融入到神经机器翻译模型的编码器中，提高了模型的翻译性能。

本文提出了基于编码器-解码器的预训练，该方法面向自然语言生成任务，基于编码器-解码器框架，通过海量的多语言文档级单语语料预训练文本生成任务。预训练之后的模型已经具备了一定的特征提取能力和跨语言建模能力，通过预训练模型的参数初始化神经机器翻译模型，可以使翻译模型达到更好的性能。

本文通过实验证明了上述两种方法都有效地提高了稀缺资源的机器翻译性能，验证了预训练方法在机器翻译任务上的有效性。

5.2 创新点分析

本文面向稀缺资源机器翻译任务，利用预训练方法有效提高了翻译性能。本文基于这样一个思想：在海量单语数据上预训练的模型虽然不包括显式的语言间的转换，但是预训练模型已经具备了信息抽取能力和语言建模能力，相比随机初始化的模型，使用预训练模型可以加快收敛速度，达到更好的模型性能。本文的创新点如下：

1. 本文提出了基于编码器的预训练，通过三种方法将现有的预训练模型集成到神经机器翻译模型中，分别是基于词嵌入的集成方法、基于隐藏层特征的集成方法、基于编码器微调的集成方法。之前的研究人员通过预训练的词嵌入初始化神经机器翻译模型的词嵌入，对模型性能有一定的帮助，然而，之前的方法一般通过Word2vec或Glove等进行预训练，并没有尝试过将预训练模型的词嵌入融入到神经机器翻译模型中。基于隐藏层特征的集成方法在自然语言处理领域的一些下游任务中得到了优异的性能，但之前也并没有将其应用到神经机器翻译模型的先例。此外，目前已有的预训练模型和神经机器翻译模型的网络结构并不完全匹配，因此本文提出的基于编码器微调的集成方法可以充分利用预训练模型的能力，三种方法均可以将预训练模型捕获的信息集成到神经机器翻译模型中，实验证明三种方法均显著提高了模型性能。本文对三种集成方法进行了充分的分析，证明了基于编码器的预训练可以显著提高翻译性能，加快收敛速度，增强模型的鲁棒性。

2. 面向机器翻译任务，本文提出了基于编码器-解码器框架的预训练方法。该方法通过海量的文档级单语数据预训练文本生成任务，从而可以捕获到跨语言信息。预训练阶段完成之后，预训练的模型可以直接用来初始化神经机器翻译模型，解决了目前提出的预训练模型一般只能利用神经机器翻译模型的部分模块的问题。实验证明，在稀缺资源场景下，该预训练方法能够有效提升模型性能。

5.3 未来工作

本文面向稀缺资源机器翻译任务，提出了两种基于预训练的方法，将现有预训练模型集成到神经机器翻译模型中，并提出了新的预训练方法应用于机器翻译任务上。两种方法均有效提高了模型性能，加快了收敛速度，但仍具有一定的不足之后。未来我们将针对下面几个工作展开进一步的研究讨论。

1. 覆盖度更广的预训练模型集成方法。本文中通过三种方法将预训练模型集成到神经机器翻译模型中，翻译性能、收敛速度和鲁棒性均得到了显著提高，但这三种方法只能利用预训练模型的部分能力，或是只能将预训练模型用来初始化神经机器翻译模型的编码器部分，模型仍有很大一部分参数需要从头学习。而目前已经提出的预训练模型，包括ELMo，GPT和BERT等，ELMo是基于双向LSTM的语言模型，和现如今神经机器翻译使用的基于自注意力机制的模型结构完全不匹配，而GPT和BERT都是基于自注意力机制的模型结构，GPT可以看作是神经机器翻译模型的解码器，BERT可以看作是神经机器翻译模型的编码器，研究如何在神经机器翻译模型上同时利用GPT和BERT是我们将来的一个研究方法。一个比较简单而又直观的想法是分别使用BERT和GPT来初始化神经机器翻译模型的编码器和解码器，但是神经机器翻译模型的解码器相比GPT要多出一个子层，即编码器-解码器注意力层，因此直接将预训练模型的参数用于初始化翻译模型的解码器模块仍然需要进一步的思考和验证。

2. 预训练方法的改进。本文针对机器翻译任务，提出了基于编码器-解码器的预训练方法，在稀缺资源任务上取得了一定的提升，但该方法存在着一些局限性，其中最主要的问题是编码器和解码器之间的依赖性很弱，文本生成任务的输出结果自由度很大，即解码器对于编码器提取的特征的依赖性并不高，而机器翻译虽然也有着一定的自由度，但输入输出之间的联系仍然十分密切，在语义上表示相同的意思，因此研究如何提高编码器-解码器之间的依赖性是很重要的，一个思想是引入多任务学习，除了预测下一句这个目标之外，增加额外的训练目标，来增强编码器和解码器之间的联系，并提高编码器的提取能力。另外一个问题是，本文提出的预训练方法需要基于共享词表训练，这对于相似语种来说是十分方便的，然而对于中文-英文这样的语言，因此词汇的组成差距很大，简单使用共享的方法可能无法有效地提升模型性能。因此，研究在不相近语种上的预训练方法也是我们将来的一个方向。

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): 53-82.
2. Lagarda A L, Alabau V, Casacuberta F, et al. Statistical Post-Editing of a Rule-Based Machine Translation System[C]. north american chapter of the association for computational linguistics, 2009: 217-220.
3. Hua W, Haifeng W. Improving statistical word alignment with a rule-based machine translation system[C]. international conference on computational linguistics, 2004.
4. Somers H L. Review Article: Example-based Machine Translation[J]. Machine Translation, 1999, 14(2): 113-157.
5. Sumita E. Example-based machine translation using DP-matching between word sequences[C]. meeting of the association for computational linguistics, 2001: 1-8.
6. Liu Z, Wang H, Wu H, et al. Example-based machine translation based on tree---string correspondence and statistical generation[J]. Machine Translation, 2006, 20(1): 25-41.
7. Brown P F, Pietra V J, Pietra S D, et al. The mathematics of statistical machine translation: parameter estimation[J]. Computational Linguistics, 1993, 19(2): 263-311.
8. Och F J, Ney H. Discriminative Training and Maximum Entropy Models for Statistical Machine Translation[C]. meeting of the association for computational linguistics, 2002: 295-302.
9. Koehn, Philipp, Josef F, et al. Statistical phrase-based translation[J]. Proc. HLT-NAACL, 2003, 2003.
10. Och F J. Minimum Error Rate Training in Statistical Machine Translation[C]. meeting of the association for computational linguistics, 2003: 160-167.
11. Koehn, Philipp, Hoang, et al. Moses: open source toolkit for statistical machine translation[J]. in Proceedings of the Association for Computational Linguistics (ACL’07, 2007, 9(1):177--180.
12. Nirenburg S, Somers H, Wilks Y. A Statistical Approach to Machine Translation[J]. Computational Linguistics, 1990, 16(2):79-85.
13. Xiao T, Zhu J, Zhang H, et al. NiuTrans: An Open Source Toolkit for Phrase-based and Syntax-based Machine Translation[C]. meeting of the association for computational linguistics, 2012: 19-24.
14. 刘洋. 树到串统计翻译模型研究[D]. 中国科学院计算技术研究所, 2007.
15. 肖桐, 李天宁, 陈如山,等. 面向统计机器翻译的重对齐方法研究[J]. 中文信息学报, 2010, 24(1):110-117.
16. 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.
17. 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.
18. Sutskever I, Vinyals O, Le Q V, et al. Sequence to Sequence Learning with Neural Networks[J]. neural information processing systems, 2014: 3104-3112.
19. 刘洋. 神经机器翻译前沿进展[J]. 计算机研究与发展, 2017(6).
20. Bahdanau D, Cho K, Bengio Y, et al. Neural Machine Translation by Jointly Learning to Align and Translate[J]. international conference on learning representations, 2015.
21. 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.
22. Johnson M, Schuster M, Le Q V, et al. Google's Multilingual Neural Machine Translation System: Enabling Zero-Shot Translation[J]. Transactions of the Association for Computational Linguistics, 2017, 5(1): 339-351.
23. Wu Y, Schuster M, Chen Z, et al. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation[J]. arXiv: Computation and Language, 2016.
24. Cho K, Van Merrienboer B, Gulcehre C, et al. Learning Phrase Representations using RNN Encoder--Decoder for Statistical Machine Translation[J]. empirical methods in natural language processing, 2014: 1724-1734.
25. Chung J, Gulcehre C, Cho K, et al. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling[J]. arXiv: Neural and Evolutionary Computing, 2014.
26. Gehring J, Auli M, Grangier D, et al. Convolutional sequence to sequence learning[J]. international conference on machine learning, 2017: 1243-1252.
27. LeCun Y, Bengio Y. Convolutional networks for images, speech, and time series[J]. The handbook of brain theory and neural networks, 1995, 3361(10): 1995.
28. Vaswani A, Shazeer N, Parmar N, et al. Attention is All you Need[J]. neural information processing systems, 2017: 5998-6008.
29. Shaw P, Uszkoreit J, Vaswani A, et al. SELF-ATTENTION WITH RELATIVE POSITION REPRESENTATIONS[J]. north american chapter of the association for computational linguistics, 2018: 464-468.
30. Lin Z, Feng M, Santos C N, et al. A STRUCTURED SELF-ATTENTIVE SENTENCE EMBEDDING[J]. international conference on learning representations, 2017.
31. Mettes P, Koelma D C, Snoek C G, et al. The ImageNet Shuffle: Reorganized Pre-training for Video Event Detection[J]. international conference on multimedia retrieval, 2016: 175-182.
32. Simon M, Rodner E, Denzler J, et al. ImageNet pre-trained models with batch normalization[J]. arXiv: Computer Vision and Pattern Recognition, 2016.
33. Bengio Y, Ducharme R, Vincent P, et al. A neural probabilistic language model[J]. Journal of Machine Learning Research, 2003, 3(6): 1137-1155.
34. Mikolov T, Chen K, Corrado G S, et al. Efficient Estimation of Word Representations in Vector Space[J]. international conference on learning representations, 2013.
35. Mikolov T, Sutskever I, Chen K, et al. Distributed Representations of Words and Phrases and their Compositionality[J]. neural information processing systems, 2013: 3111-3119.
36. Peters M E, Ammar W, Bhagavatula C, et al. Semi-supervised sequence tagging with bidirectional language models[J]. meeting of the association for computational linguistics, 2017: 1756-1765.
37. Mccann B, Bradbury J, Xiong C, et al. Learned in Translation: Contextualized Word Vectors.[J]. neural information processing systems, 2017: 6294-6305.
38. Peters M E, Neumann M, Iyyer M, et al. DEEP CONTEXTUALIZED WORD REPRESENTATIONS[J]. north american chapter of the association for computational linguistics, 2018: 2227-2237.
39. Radford A, Narasimhan K, Salimans T, et al. Improving language understanding with unsupervised learning. Technical report, OpenAI.
40. Devlin J, Chang M, Lee K, et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding[J]. arXiv: Computation and Language, 2018.
41. Chung J, Gulcehre C, Cho K, et al. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling[J]. arXiv: Neural and Evolutionary Computing, 2014.
42. Hochreiter S, Schmidhuber J. Long short-term memory[J]. Neural Computation, 1997, 9(8): 1735-1780.
43. Lample G, Conneau A. Cross-lingual Language Model Pretraining.[J]. arXiv: Computation and Language, 2019.
44. Song K, Tan X, Qin T, et al. MASS: Masked Sequence to Sequence Pre-training for Language Generation[J]. arXiv: arXiv:1905.02450, 2019
45. Chen M X, Firat O, Bapna A, et al. The Best of Both Worlds: Combining Recent Advances in Neural Machine Translation[J]. meeting of the association for computational linguistics, 2018: 76-86.
46. Domhan T. How Much Attention Do You Need? A Granular Analysis of Neural Machine Translation Architectures[C]. meeting of the association for computational linguistics, 2018: 1799-1808.
47. Papineni K, Roukos S, Ward T, et al. Bleu: a Method for Automatic Evaluation of Machine Translation[C]. meeting of the association for computational linguistics, 2002: 311-318.
48. 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.
49. Srivastava N, Hinton G E, Krizhevsky A, et al. Dropout: a simple way to prevent neural networks from overfitting[J]. Journal of Machine Learning Research, 2014, 15(1): 1929-1958.
50. Ba J L, Kiros J R, Hinton G E. Layer normalization[J]. arXiv preprint arXiv:1607.06450, 2016.

致谢

值此毕业之际，我要向实验室的所有老师和同学们表示由衷的感谢。

首先，我要感谢我的导师朱靖波老师，在我就读硕士研究生期间，您的指导对我产生了很大的帮助。您不仅教给了我知识，更重要的是教给了我做人做事的态度，教会了我如何去做研究。您教育我们要做高水平研究，要做一些真正有贡献的工作，而不要将目标放在找到一个好的工作或是发表顶级论文上，而且做一项研究要有合理的动机，不要只是为了做而做，这些对我以后的求学之路都会产生深刻的影响。

然后，我要感谢实验室的肖桐老师，您严谨的治学态度、渊博的学识永远是我学习的方向。在研究过程中，您给我提出了很多宝贵的意见，尤其是在跟您做开源工具的过程中，您坚持做精品的态度对我产生了深刻的影响。不仅如此，您很尊重我们的想法，会站在我们的角度思考问题。在学术上，您是我们的良师，在生活中，您是我们的益友。

我还要感谢实验室的其他老师，正是每一位老师的默默付出，实验室才会有这么好的氛围，你们对待每一个学生都像对待自己的孩子一样。我要感谢实验室的学长、学姐、学弟、学妹们，感谢你们在生活和学习上对我的帮助，通过和你们交流我进步成长了很多。我还要感谢我的舍友们，你们陪我度过了愉快的研究生生活，祝你们前程似锦，幸福快乐。

正是实验室家一样的氛围让我决定留下攻读博士学位，继续在实验室度过之后几年时光，继续去攀登研究的高峰。

我想要感谢我的父母和姐姐，感谢你们对我的这么多年的照顾，感谢你们支持我攻读硕士学位和博士学位。在我就读硕士研究生这两年，每年回家的次数都很少，而且时间很短，平时因为学习和生活的压力和你们的联系也少了很多，感谢你们的体谅，我以后会安排好时间，和你们多点交流，多点关心，常和你们视频，多和你们说说话，你们照顾了我这么多年，现在年纪大了，就让我好好照顾你们吧。

最后，感谢东北大学和计算机科学与工程学院对我们的培养，希望东大以后可以发展地更好！

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重点项目“面向资源稀缺型语言的机器翻译理论方法及关键技术研究”，起止时间 2018年1月-2022年12月。