****

**本 科 毕 业 设 计（论文）**

题 目：基于Transformer的智能文本摘要系统设计 与实现

|  |  |
| --- | --- |
| 学生姓名： | 徐世杰 |
| 学 号： | 2107010221 |
| 专业班级： | 计算机科学与技术2102班 |
| 指导教师： | 张培颖 |

2025年6月\*\*日

学位论文原创性声明

本人所提交的学位论文基于Transformer的智能文本摘要系统设计与实现，是在导师的指导下，独立进行研究工作所取得的原创性成果。除文中已经注明引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写过的研究成果。对本文的研究做出重要贡献的个人和集体，均已在文中标明。

本声明的法律后果由本人承担。

论文作者（签名）： 指导教师确认（签名）：

年 月 日 年 月 日

学位论文版权使用授权书

本学位论文作者完全了解中国石油大学（华东）有权保留并向国家有关部门或机构送交学位论文的复印件和磁盘，允许论文被查阅和借阅。本人授权中国石油大学（华东）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，可以采用影印、缩印或其它复制手段保存、汇编学位论文。

保密的学位论文在\_\_\_\_\_\_\_年解密后适用本授权书。

论文作者（签名）： 指导教师（签名）：

年 月 日 年 月 日

摘 要

本文设计并实现了一种基于Transformer架构的智能文本摘要系统，旨在提升长文本处理的效率与质量。首先，对新闻、技术文档和社交媒体等多领域数据进行规范化预处理与标注，构建多样化训练样本；其次，以Mengzi-T5为基础模型，采用迁移学习微调，下游任务中引入抽取-生成多阶段迭代式摘要策略，兼顾信息覆盖与长度控制；再次，通过动态填充、均匀长度批处理和Early Stopping等推理优化技术，有效加速单条及批量摘要生成；最后，基于ROUGE、BLEU等指标对比LSTM和传统统计方法，实验结果表明所提方法在长短文本摘要任务中均取得显著性能提升。论文还阐述了系统实现细节和用户界面设计，为智能摘要技术的实际应用提供了参考与展望。

**关键词**：Transformer 架构；文本摘要；迁移学习；多阶段摘要；性能优化

**Abstract**

This thesis designs and implements an intelligent text summarization system based on the Transformer architecture, aiming to enhance the efficiency and quality of long-text processing. First, standardized preprocessing and annotation are performed on multi-domain data including news, technical documents, and social media content to construct diverse training samples. Second, building upon the Mengzi-T5 base model, transfer learning fine-tuning is adopted, while an extraction-generation multi-stage iterative summarization strategy is introduced in downstream tasks to balance information coverage and length control. Third, inference acceleration techniques including dynamic padding, uniform-length batching, and Early Stopping effectively optimize both single-instance and batch summarization generation. Finally, comparative experiments against LSTM and traditional statistical methods using metrics like ROUGE and BLEU demonstrate that the proposed method achieves significant performance improvements in both long and short text summarization tasks. The thesis also details system implementation specifics and user interface design, providing practical references and future perspectives for intelligent summarization technology applications.

**Keywords**: Transformer architecture, text summarization, transfer learning, multi-stage summarization, performance optimization

目 录

[第1章 序言 1](#_Toc198820189)

[1.1 研究背景与意义 1](#_Toc198820190)

[1.2 研究目标与内容 1](#_Toc198820191)

[1.3 论文结构安排 2](#_Toc198820192)

[第2章 相关研究工作 4](#_Toc198820193)

[2.1 文本摘要技术概述 4](#_Toc198820194)

[2.2 Transformer架构与预训练模型 4](#_Toc198820195)

[2.3 抽取式与生成式摘要方法 6](#_Toc198820196)

[2.3.1 抽取式摘要 6](#_Toc198820197)

[2.3.2 生成式摘要 7](#_Toc198820198)

[第3章 数据集构建与预处理 9](#_Toc198820199)

[3.1 数据集来源与组成 9](#_Toc198820200)

[3.1.1 NLPCC 2017 新闻标题分类数据集 9](#_Toc198820201)

[3.1.2 LCSTS 中文短文本摘要数据集 9](#_Toc198820202)

[3.1.3 WikiLingua多语言摘要数据集 9](#_Toc198820203)

[3.2 数据清洗与划分 10](#_Toc198820204)

[3.2.1 NLPCC 2017 数据处理 10](#_Toc198820205)

[3.2.2 LCSTS 数据处理 10](#_Toc198820206)

[3.2.3 WikiLingua 数据处理 10](#_Toc198820207)

[第4章 模型训练与微调 11](#_Toc198820208)

[4.1 预训练模型选择与迁移学习策略 11](#_Toc198820209)

[4.2 模型训练与参数设置 12](#_Toc198820210)

[4.2 模型训练与参数设置 12](#_Toc198820211)

[4.2.1 数据加载与预处理 12](#_Toc198820212)

[4.2.2 训练参数设置 12](#_Toc198820213)

[4.2.3 评估指标与度量 13](#_Toc198820214)

[4.2.4 训练流程与模型保存 13](#_Toc198820215)

[第5章 实验设计与结果分析 15](#_Toc198820216)

[5.1 实验设置与评估指标 15](#_Toc198820217)

[5.1.1 实验环境 15](#_Toc198820218)

[5.1.2 数据集与测试集划分 15](#_Toc198820219)

[5.1.3 对比方法 16](#_Toc198820220)

[5.1.4 评估指标 16](#_Toc198820221)

[5.2 长文本摘要生成质量优化方法 16](#_Toc198820222)

[5.2.1 抽取-生成式摘要 16](#_Toc198820223)

[5.2.2 层次化摘要 17](#_Toc198820224)

[5.3 推理速度优化方法 18](#_Toc198820225)

[5.3.1 Early Stopping 策略效果 18](#_Toc198820226)

[5.3.2 动态填充与均匀长度批处理效果 19](#_Toc198820227)

[第6章 系统实现 20](#_Toc198820228)

[6.1 系统架构与功能模块 20](#_Toc198820229)

[6.2 用户界面与交互设计 22](#_Toc198820230)

[6.2.1 文本摘要生成系统主界面 22](#_Toc198820231)

[6.2.2 摘要生成界面 23](#_Toc198820232)

[6.2.3 自定义文本摘要与问答界面 24](#_Toc198820233)

[第7章 总结与展望 25](#_Toc198820234)

[7.1 研究成果总结 25](#_Toc198820235)

[7.2 未来研究展望 26](#_Toc198820236)

[7.2.1 未来研究方向 26](#_Toc198820237)

[7.2.2 社会、健康、安全、法律与文化影响 27](#_Toc198820238)

[7.2.3 环境与可持续发展影响 28](#_Toc198820239)

[致 谢 29](#_Toc198820240)

[参考文献 29](#_Toc198820241)

[附 录 29](#_Toc198820242)

[附录A名词术语及缩略词 29](#_Toc198820243)

# 第1章 序言

## 1.1 研究背景与意义

随着互联网、社交平台和行业文档数量的爆发式增长，信息过载已成为阻碍人们高效获取、处理和传播知识的主要障碍。长篇文本（如新闻报道、技术白皮书）篇幅冗长、结构复杂，直接通读不仅耗时，而且容易遗漏核心信息。智能文本摘要技术通过自动抽取或生成文档要点，大幅降低用户阅读负担、提升信息获取效率，因此在检索、舆情监测、知识管理和智能问答等场景中拥有广阔的应用价值。

以往的摘要方法主要分为两类：一是统计或图模型驱动的抽取式算法（典型如 TextRank、LSA），二是基于循环神经网络的生成式算法（如 Seq2Seq 加注意力机制）。尽管在短文本或中等长度文档上取得一定效果，但面对千词乃至万词级别的超长文本，仍存在三大瓶颈：其一，模型对输入长度有限制，难以全面处理大规模内容；其二，抽取式容易遗漏关键信息，生成式又常出现重复或模糊；其三，推理速度较慢，难以满足实时或海量批处理需求。

Transformer 架构及其衍生的预训练模型（如 BERT、T5、Mengzi‐T5 等）凭借自注意力在捕捉长距离依赖和并行计算方面的优势，为长文本摘要研究带来了新机遇。不过，直接应用大规模预训练模型往往面临高昂的计算与存储成本，并且在超长文档场景下，还需对输入表示、解码策略和推理效率进行深入优化。

基于上述背景，本文设计并实现了一套兼顾摘要质量与生成速度的智能长文本摘要系统：一方面，融合层次化与多阶段抽取‐生成的混合策略，增强对文档全局与局部信息的捕获；另一方面，引入 Early Stopping、动态填充和均匀长度批处理等推理优化技术，以加速摘要生成并提升资源利用率。该研究不仅丰富了超长文本自动摘要的理论体系，也为实际应用中的智能文本处理提供了创新思路和技术参考。

## 1.2 研究目标与内容

本研究围绕基于Transformer架构的智能文本摘要系统展开，旨在提升长文本处理的效率与摘要质量。针对现有Transformer模型在长文本处理过程中普遍存在的输入长度受限、信息覆盖不足等难题，结合迁移学习和多阶段摘要策略，提出了相应的优化方法，以实现高效且高质量的文本摘要生成。具体工作包括：

1、多领域摘要数据集的构建与预处理：收集涵盖新闻、技术文档、社交媒体等多个领域的文本数据，并对数据进行清洗、标注及合理划分，以保证训练样本的多样性和高质量。

2、基于Transformer的摘要模型设计与优化：选用预训练模型Mengzi-T5为基础，结合迁移学习思路，在下游摘要任务上进行微调，提升模型在各类文本摘要任务中的综合表现。

3、摘要质量优化方法的探索：提出抽取与生成相结合的多阶段摘要方案，先利用抽取式方法筛选出文档中的核心句段，再在此基础上进行抽象生成，以突破输入长度限制、提升信息覆盖率，并进一步优化长文本摘要的最终质量。

4、推理阶段的效率提升技术：引入动态填充、均匀长度批处理及Early Stopping等推理优化手段，加快摘要生成速度、节省算力资源，提高系统整体效率。

5、系统实现与性能评估：开发具备可视化界面的智能摘要系统，支持用户在线输入文本并实时查看生成摘要及相应的质量评估结果。通过ROUGE、BLEU等评价指标，与主流方法进行对比，验证所提方案在不同长度文本摘要任务中的优越性。

## 1.3 论文结构安排

为便于理清研究思路并系统展示论文成果，全文共分为七个章节，各章节安排如下：

第一章为绪论，主要介绍课题背景与研究意义，明确本文的核心目标及主要研究内容，同时对全文结构作出总体说明，为后续章节作铺垫。

第二章综述了相关领域的研究进展，包括文本摘要技术的经典方法与最新动态，着重阐述Transformer架构和预训练模型的发展历程，并比较抽取式和生成式摘要方法的特点及其适用场景。

第三章则聚焦于数据集的构建和预处理，详细说明数据来源、组成与特性，结合实际操作过程，介绍数据清洗、标注和样本划分等环节，为后续的模型训练提供坚实的数据基础。

第四章围绕模型训练与微调展开，首先说明预训练模型的选型依据及迁移学习策略，然后详细列出训练流程、重要超参数设置，以及相关硬件配置，以保证实验具有可复现性。

第五章为实验设计与结果分析，先介绍实验方案与评测指标，继而围绕长文本摘要的生成质量，依次分析抽取–生成式与层次化摘要方法的优化成效；同时，对推理速度的提升方法（如Early Stopping、动态填充和均匀长度批处理）进行效果评估，并给出定量与定性分析。

第六章介绍系统的实现细节，从整体架构、各功能模块到摘要服务接口、部署方案，以及用户界面和交互方式，系统性地说明自动摘要功能的端到端实现过程。

第七章是总结与展望，梳理全文的主要工作与创新成果，全面评价模型表现和系统功能，并结合当前研究不足，提出未来在提升摘要质量、多语种或跨领域扩展等方向的研究设想。

# 第2章 相关研究工作

## 2.1 文本摘要技术概述

自动文本摘要的目标是从冗长的原文中提炼出关键信息，生成简明且准确的短文本，帮助用户迅速把握核心内容。根据输入文档的数量和实际应用的不同，文本摘要任务通常分为单文档摘要（Single Document Summarization）和多文档摘要（Multi Document Summarization）两类：前者针对单篇文本生成概要，后者则需综合多篇相关文献的信息进行融合与归纳。

在实际应用中，文本摘要系统已经广泛渗透到诸如新闻简报、法律文档整理和科研文献检索等多个领域。一般来说，生成的摘要会被控制在原文长度的10%到50%之间，以便在覆盖关键信息的同时，也降低用户的阅读负担。早期的自动摘要方法多依赖于统计特征，比如词频和句子的位置等，采用相对简单的筛选机制。随着技术演进，图模型、序列标注、线性规划以及子模函数等方法被陆续应用，特征工程逐渐成为摘要系统的主流手段。进入深度学习时代后，BERT、T5、BART等预训练语言模型结合Seq2Seq架构，使自动摘要的质量有了显著提升，但与此同时，对算力和输入文本长度的要求也变得更加苛刻。

在评估摘要效果时，常用的自动化指标包括ROUGE（关注内容重叠）和BLEU（侧重n-gram精度），此外还会采用人工评审对摘要的连贯性和一致性进行主观评价。

目前，自动摘要领域的研究热点还包括可控摘要（如控制摘要长度、关注特定内容）、事实一致性（减少“幻觉”内容）以及多模态摘要（结合文本和图表信息）等方向。

## 2.2 Transformer架构与预训练模型

Transformer是一种深度学习模型，核心在于自注意力机制。这一架构由Vaswani等人在2017年首次提出，旨在改进序列到序列的任务表现，比如机器翻译。与传统的循环神经网络（RNN）不同，Transformer舍弃了序列对齐的流程，而是通过自注意力机制直接处理序列中不同位置的信息关联，这不仅加快了训练速度，也使模型效果得到了提升。

在结构上，Transformer分为编码器（Encoder）和解码器（Decoder）两部分。每个编码器层都包含多头自注意力和前馈神经网络，而解码器在此基础上还加入了编码器-解码器注意力机制。通过多层叠加，模型能够理解更为复杂的语法和语义。Transformer的整体结构如图2-1所示。

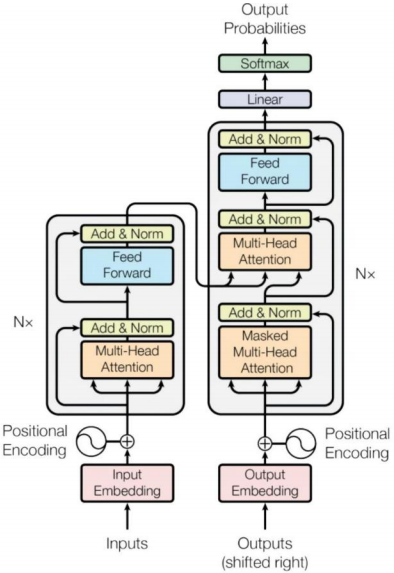


图2-1 Transformer架构

基于Transformer架构，学术界和工业界陆续推出了多种预训练语言模型。这类模型通常先在大规模文本语料上进行通用语言知识的预训练，随后通过微调（Fine-tuning）来适应具体的任务需求。代表性的预训练模型主要包括以下几类：

1、BERT（Bidirectional Encoder Representations from Transformers）：采用双向编码结构，通过“掩码语言模型”（Masked Language Model）和“下一句预测”（Next Sentence Prediction）等预训练目标，尤其擅长句子理解相关任务。

2、GPT（Generative Pre-trained Transformer）系列：以解码器为核心，利用自回归方式进行预训练，善于处理各类文本生成任务，比如自动写作和对话生成。

3、T5（Text-To-Text Transfer Transformer）：将所有自然语言处理任务统一建模为“文本到文本”形式，采用编码器-解码器结构，因而具备较强的任务泛化能力。

这些预训练模型的出现，极大地推动了自然语言处理技术的发展，如今已成为业内普遍采用的主流方案。在本研究中，我以Mengzi-T5作为基础模型，结合迁移学习的方法，针对新闻报道、技术资料以及社交媒体等不同类型的文本进行了微调，力求进一步提升长文本摘要的质量和生成效率。

## 2.3 抽取式与生成式摘要方法

### 2.3.1 抽取式摘要

抽取式摘要的基本思路，是直接选取原文中最具代表性的一些句子或段落，将它们拼接后作为摘要。这种方法能够完整保留原文的表述方式，实现起来也相对简单，计算速度较快。常见的实现方式包括以下几类：

1、统计特征法：通过设置如词频-逆文档频率（TF-IDF）、句子在文中的位置、句子长度等手工特征，对各句子进行打分筛选。

2、图排序方法：比如TextRank、LexRank等算法，会将每个句子看作图中的一个节点，节点之间的边权由句子间的相似度决定，然后通过类似PageRank的机制来评估句子的权重和重要性。

3、监督学习方法：将句子的抽取过程视为一个二分类或回归问题，利用SVM、决策树或神经网络等模型预测各句子的得分。

以TextRank为例，这是一种基于图结构的无监督抽取式摘要算法，由Mihalcea和Tarau提出，其思路借鉴了PageRank网页排序原理。具体流程包括：

1、构建句子间的图结构

2、初始化每个节点（句子）的分数

3、通过迭代方式不断更新分数

4、最终分数收敛后，对句子进行排序，选出重要句生成摘要

其中，相似度计算的公式可以表示为公式（2-1）。和表示文本被划分出的第句和第句。表示句子相似度。

PageRank更新的公式可以表示为公式（2-2）。Score()表示句子的当前 PageRank 分数，代表句子的重要性或“权重”。表示阻尼系数（damping factor），通常设为 0.85，模拟“随机游走”过程中跳转到任意句子的概率。N()表示与在句子相似度图中直接相连的邻居句子集合。deg()表示句子的度（degree），即它在图中连出的邻居数量，用于对邻居的影响力进行归一化。

TextRank 的主要优点包括：

1、不依赖人工标注的数据，具有较好的通用性和适应性。

2、通过图模型能够挖掘句子之间的整体结构关系，而不仅仅局限于局部特征，有助于提升摘要内容的整体性和代表性。

### 2.3.2 生成式摘要

生成式摘要的核心在于利用语言模型对原文信息进行重新组织，生成表达更为自然的新句子。通常，这类方法将摘要看作一种“文本到文本”的转换过程，常用的实现方式是Seq2Seq编码-解码结构。具体来说，常见的方法包括：

1、Seq2Seq结合Attention机制：早期的RNN配合注意力机制，是推动自动摘要生成技术进步的重要动力。

2、预训练模型：例如T5、BART等，先在大规模语料上预训练，再针对摘要任务进行微调，能够生成更加流畅和连贯的文本。

3、Copy或Pointer机制：通过增强模型对原文词汇的复制能力，提高专业名词、人名等内容的准确还原。

生成式摘要的一大优势是输出表达灵活、自然，但缺点是计算量大，对硬件资源要求高，同时可能会出现事实性错误或内容重复等问题。本研究采用了Mengzi-T5作为基础模型，结合迁移学习，在新闻、技术文档、社交媒体等多种类型的数据集上进行微调，目标是进一步提升长文本摘要的质量和效率。Mengzi-T5的架构与T5本质一致，具体结构如图2-2所示。

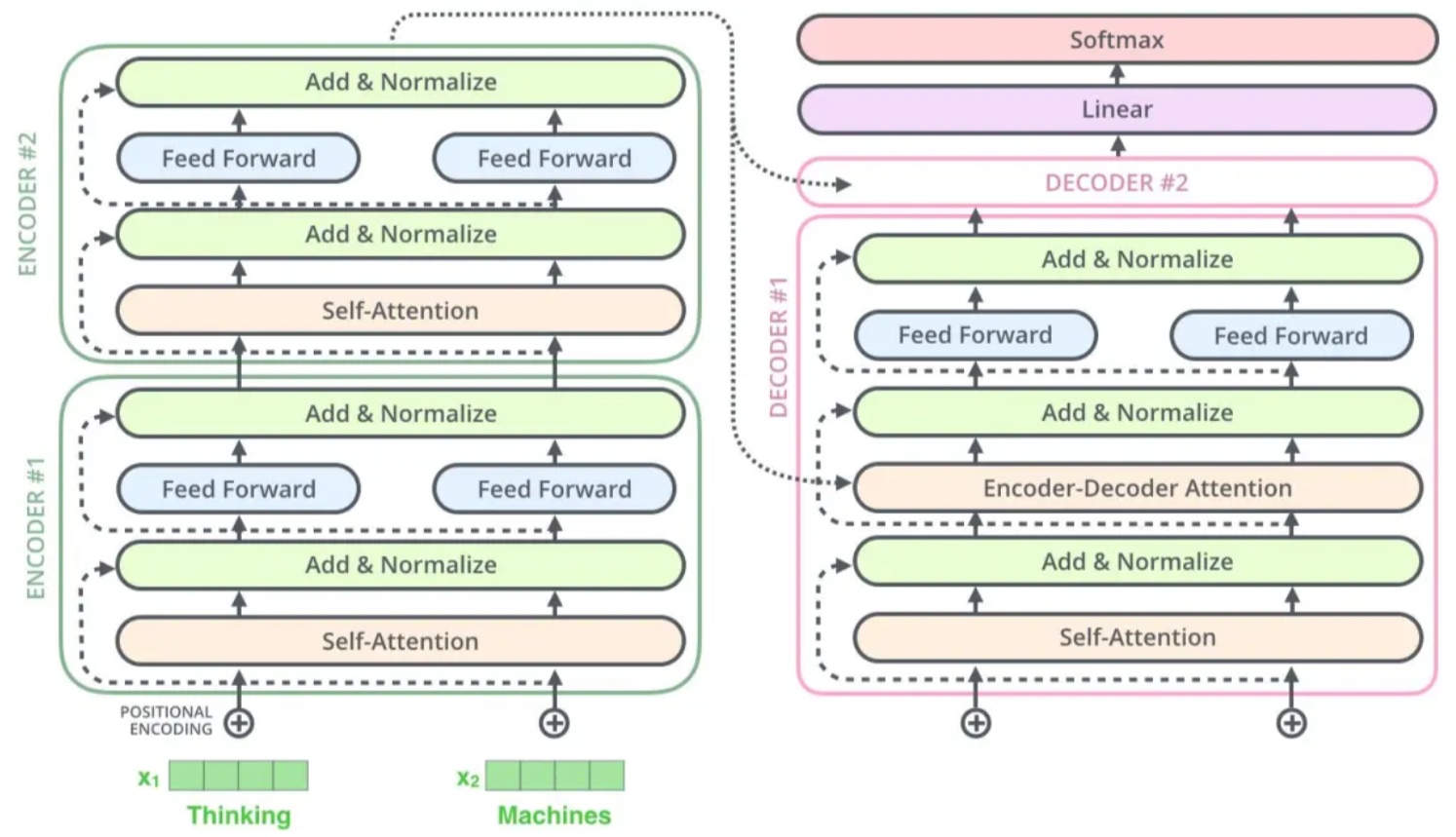


图2-2 T5架构

# 第3章 数据集构建与预处理

## 3.1 数据集来源与组成

为了让文本摘要系统具备更好的跨领域泛化能力，本文挑选了三类颇具代表性的中文数据集，分别来自新闻、社交媒体和百科知识等不同领域。它们在文本风格、长度分布以及摘要的表现形式等方面各有差异，这为模型的训练和后续评估提供了丰富而多样的语料资源。

### 3.1.1 NLPCC 2017 新闻标题分类数据集

NLPCC 2017 中文新闻标题分类数据集由复旦大学在NLPCC 2017会议期间发布，最初用于新闻标题的多类别分类研究。该数据集收集了来自新浪、今日头条等主流新闻网站的短新闻标题，内容涵盖娱乐、体育、财经、科技等18个不同领域。每条记录包含一个新闻标题及其对应的类别标签。尽管这套数据最初是为文本分类任务设计的，但由于其短文本的特点，也非常适合作为摘要任务的输入语料。在本研究中，我们把这些新闻标题作为需要摘要的输入文本，并与其他数据集一同纳入统一的摘要系统建模流程。

### 3.1.2 LCSTS 中文短文本摘要数据集

LCSTS（Large-scale Chinese Short Text Summarization Dataset）是华中科技大学于2015年发布的一个大规模中文短文本摘要数据集，数据来源于新浪微博的真实用户内容。该数据集包含了200多万条微博文本及其对应的人工作者编写的摘要，具有较强的抽象性和多样性。LCSTS将数据划分为三部分：第一部分为自动获取的微博与摘要配对，第二和第三部分则由人工对摘要的相关性进行打分，以提升数据的质量和可靠性。LCSTS目前已成为中文自动摘要领域广泛使用的重要数据集，尤其适用于生成式摘要模型的训练与评测。

### 3.1.3 WikiLingua多语言摘要数据集

WikiLingua 数据集由康奈尔大学等机构于2020年发布，是一个面向多语言抽象摘要的大规模数据集，数据主要来源于WikiHow网站的“操作指南”类文章。该数据集涵盖包括中文在内的18种语言，共收录约18,887对原文与摘要。其突出特点是采用一致性较强的跨语言摘要对齐方式，非常适合多语言及跨语言自动摘要的研究和应用。在本研究中，我们选取了WikiLingua数据集的中文部分，用于评估模型在百科知识类长文本摘要任务中的综合性能。

## 3.2 数据清洗与划分

### 3.2.1 NLPCC 2017 数据处理

NLPCC 2017 的第三项任务（Task 3）是“单文档摘要”，目标是自动生成简洁的中文新闻摘要。该任务提供了大规模的中文新闻数据集，其中包含5万篇配有人工作者撰写摘要的新闻文章，以及5万篇未带摘要的新闻文章，适用于模型训练和半监督学习。本文按照8:1:1的比例将数据集划分为训练集、验证集和测试集，分别包含40,000、5,000和5,000条样本。

在数据预处理阶段，主要进行了以下操作：

1、格式统一：将每条数据整理为统一的JSON格式，包含“正文”和“摘要”字段，便于模型后续训练与评估。

2、文本清洗：去除正文和摘要中的特殊字符、HTML标签以及多余空格，保证文本内容的纯净性和一致性。

### 3.2.2 LCSTS 数据处理

LCSTS 数据集虽然整体质量较高，但在正式用于模型训练前，仍需对原始数据进行清洗与筛选。具体处理流程如下：

1、文本清洗：去除文本中的特殊字符、表情符号以及多余空格，统一样本的文本格式，确保数据规范性。

2、长度限制：筛选出长度适中的微博文本，剔除过长或过短的样本，以保障模型训练的稳定性和效果。

考虑到原始训练集体量较大，本文随机抽取了40,000条样本作为训练集，并保留原始的验证集（8,685条）和测试集（725条）用于模型的评估与测试。

### 3.2.3 WikiLingua 数据处理

WikiLingua 数据集以 WikiHow 网站的“如何做”类文章为基础，包含约 18,887 对多语言文章及其摘要。经初步清洗后，保留了 6,000 余条高质量样本。该数据集结构清晰，但在正式应用前仍需进一步预处理，具体包括：

1、段落合并：将文章中的多个段落合并为完整文档，保证输入文本的连贯性。

2、文本清洗：去除 HTML 标签、特殊字符和多余空格，统一整体文本格式。

3、摘要提取：从原文中准确提取对应的摘要部分，确保摘要内容的完整性和准确性。

在数据集划分方面，本文按照 8:1:1 的比例，将其分为训练集、验证集和测试集，对应样本数分别为 5,233、654 和 654。

# 第4章 模型训练与微调

## 4.1 预训练模型选择与迁移学习策略

为有效处理中文文本摘要任务，本研究选用基于 Transformer 架构的 Mengzi-T5 预训练模型作为核心。该模型由 Langboat 团队打造，借鉴了 Google T5（Text-to-Text Transfer Transformer）的设计理念，拥有强大的文本生成能力，可广泛应用于文本摘要、问答及翻译等多种自然语言处理领域。Mengzi-T5 通过对海量中文语料进行预训练，已具备优秀的中文理解与生成能力，其目标是构建一个通用预训练模型，通过在具体任务上微调，以适应不同下游应用的需求。

为了充分发挥预训练模型的效能，本研究采用了迁移学习方法。具体来说，模型首先在广泛的中文语料上进行预训练，以掌握通用的语言表征；随后，在特定的文本摘要任务上进行微调，使模型能更精准地理解和生成摘要。迁移学习的优势主要体现在：

1、提升模型性能：大规模预训练使模型能够学习到丰富的语言规律，从而在下游任务中取得更佳表现。

2、减少数据依赖：由于模型已在大量数据上完成预训练，微调阶段仅需相对较少的标注数据即可达到理想效果。

3、加速训练进程：预训练模型提供了良好的参数起点，有助于微调过程更快收敛，缩短训练周期。

在实际应用中，本研究将 Mengzi-T5 模型应用于包括 NLPCC 2017、LCSTS 和 WikiLingua 在内的多个中文文本摘要数据集。通过在这些不同领域的数据集上进行微调，模型能够更好地适应各类文本摘要任务，有效提升了生成摘要的质量和准确度。

## 4.2 模型训练与参数设置

本节将具体介绍 Mengzi-T5 模型在 LCSTS、NLPCC2017 和 WikiLingua 这三个基准数据集上的下游微调过程，包括详细的训练步骤和关键的超参数设定。实验环境方面，所有微调任务均在单张 NVIDIA RTX4070（配备8GB显存）上执行，软件则基于 PyTorch 1.11 和 Transformers 4.24.0 版本库。

## 4.2 模型训练与参数设置

### 4.2.1 数据加载与预处理

1、加载方式：使用 datasets.load\_dataset("json", …) 分别载入三份 JSONL 格式的数据集。

2、字段映射：统一将原始字段 text（正文）映射为模型输入，将 summary（摘要）映射为标签。

3、分词与截断：对输入文本使用 tokenizer 进行分词，max\_length=512，对摘要 max\_length=128。

4、批内整理：采用 DataCollatorForSeq2Seq 按最大长度动态填充，保证每个批次张量维度一致。

### 4.2.2 训练参数设置

以下参数对三个数据集均相同，仅 train\_dataset、eval\_dataset 数据文件路径不同：

Seq2SeqTrainingArguments(

output\_dir="results/<dataset>-runs", # 不同数据集独立输出目录

evaluation\_strategy="epoch", # 每个 epoch 后评估一次

save\_strategy="epoch", # 保存策略同评估策略

save\_total\_limit=3, # 最多保留 3 个检查点

learning\_rate=5e-5, # 基于经验选择的初始学习率

per\_device\_train\_batch\_size=16, # 训练批大小

per\_device\_eval\_batch\_size=16, # 验证批大小

weight\_decay=0.01, # 权重衰减系数

num\_train\_epochs=3, # 训练轮数

predict\_with\_generate=True, # 评估时输出生成结果

fp16=torch.cuda.is\_available(), # 若有 GPU 则启用混合精度

logging\_dir="results/<dataset>-logs", # 日志目录

logging\_steps=100 # 记录日志的步数间隔

)

### 4.2.3 评估指标与度量

1、ROUGE：使用 rouge.Rouge() 计算 ROUGE-1、ROUGE-2、ROUGE-L 的 F1 分数。

2、解码与后处理：将模型输出与标签中的 -100 替换为 pad\_token\_id，再调用 batch\_decode(skip\_special\_tokens=True)

3、Tokenization：为保证中文性能，使用 NLTK 的 word\_tokenize 对文本进行空格分隔后计算指标。

### 4.2.4 训练流程与模型保存

1、初始化：加载 AutoTokenizer 与 AutoModelForSeq2SeqLM。

2、预处理：使用 dataset.map(preprocess\_function, batched=True) 进行分词与标签对齐。

3、Trainer 训练：调用 trainer.train()，训练完成后自动根据 save\_strategy 存储最佳模型。

4、最终模型：在每个数据集对应目录下保留 final\_model/，并保存 tokenizer 配置，便于后续推理与部署。

5、测试评估：在测试集上调用 trainer.evaluate()，输出最终性能指标。

以模型在NLPCC2017数据集上的训练过程为例，图4-1展示了模型训练过程中的损失曲线及其平滑曲线。蓝色实线代表每个记录点的原始训练损失，橙色曲线为滑动窗口（window=5）后的平滑曲线以突出整体趋势。可以看到，训练损失从初始约 0.45 迅速下降至 0.36 左右，并在 2 000–5 000 步期间保持平稳小幅波动，随后又继续下降至最低点 0.299 （约在 5 900 步）。整体而言，模型在约 2 000 步后即进入稳定收敛状态，并在 6 000 步左右达成最低损失。

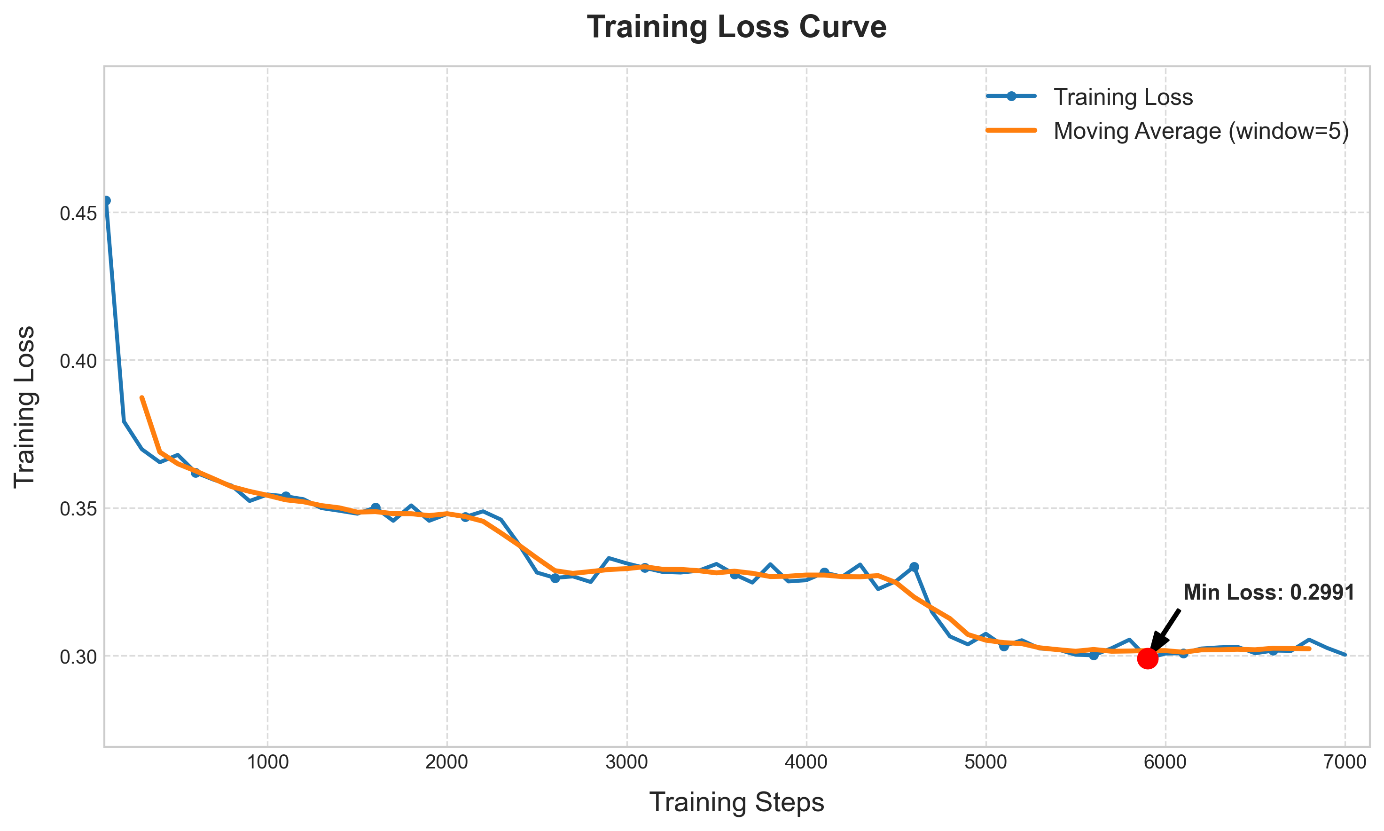


图4-1 训练损失曲线

图4-2展示了在验证集上的损失曲线及其平滑曲线。红色实线为每次验证的原始损失值，紫色曲线为相同窗口的平滑结果。尽管验证损失在个别步数上存在较大波动，但其整体趋势与训练损失保持一致：前 1 000 步内快速下降，随后在 2 000–5 000 步出现小幅波动并走平，在 6 400 步左右达到最低点 0.318 左右。验证曲线的平滑线与训练平滑线趋于一致，表明模型并未出现明显过拟合。

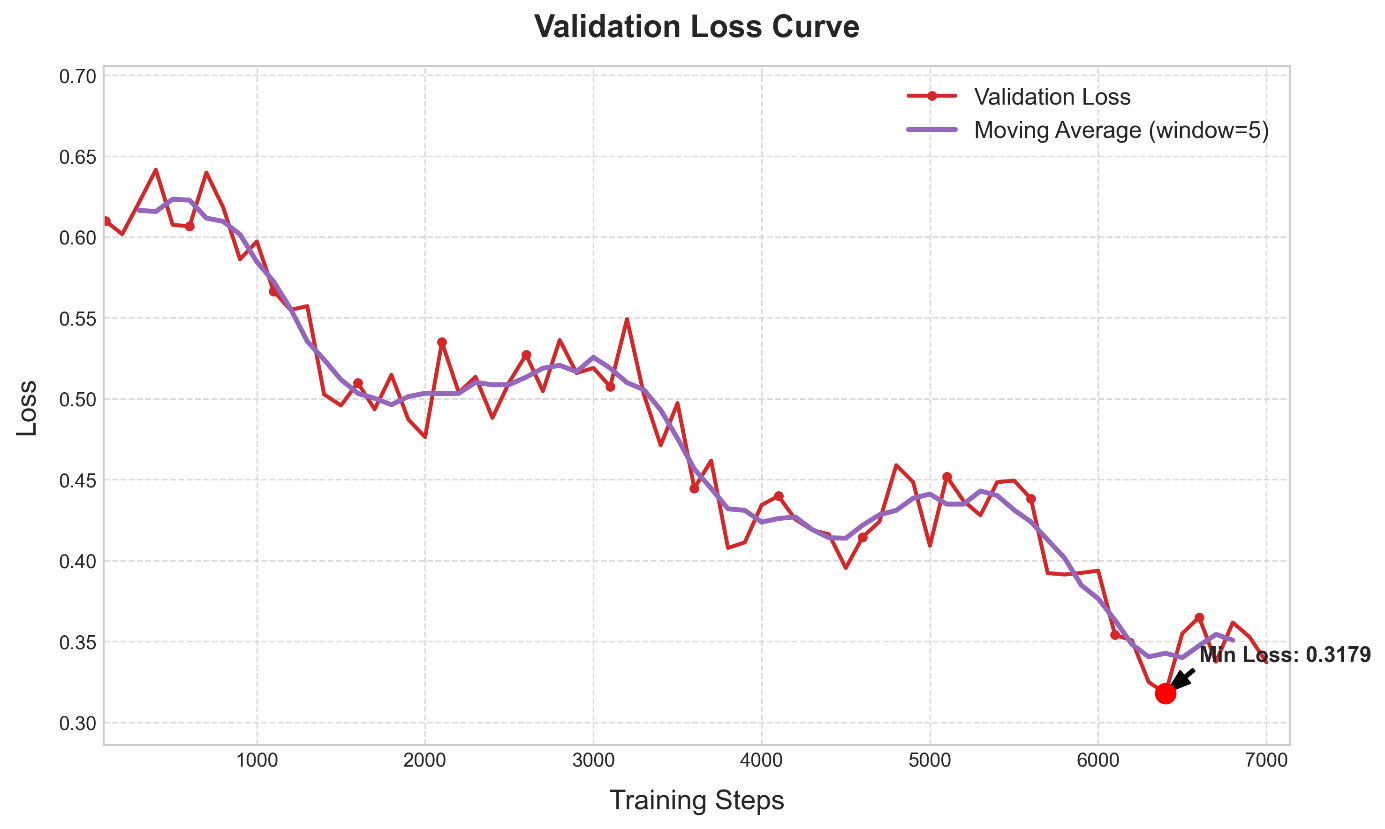


图4-2 验证损失曲线

# 第5章 实验设计与结果分析

## 5.1 实验设置与评估指标

为验证训练好的模型在“长文本摘要质量优化”与“推理速度优化”方面的效果，本节介绍具体的实验环境、所用数据、对比方法，以及评价指标。

### 5.1.1 实验环境

1、硬件：单卡NVIDIA RTX 4070（8GB显存）、Intel I7-14700HX CPU、32GB RAM

2、软件：Windows 11、Python 3.8、PyTorch 1.11、Transformers 4.24.0、datasets 1.18.3

3、依赖：jieba、scikit‑learn、rouge、nltk、tqdm、NumPy

### 5.1.2 数据集与测试集划分

因模型在三个数据集上表现类似，本节仅选取NLPCC2017中的5 000 条长文档摘要对作为测试集。

### 5.1.3 对比方法

1、基线生成式：直接使用微调后的 Mengzi‑T5 对全文生成摘要。

2、抽取–生成：先 TextRank 抽取关键句，再 Mengzi-T5 生成。

3、层次化摘要：分层编码后逐级生成。

4、Early Stopping：使用 Early Stopping 生成。

5、动态填充与均匀长度批处理：使用动态填充和均匀长度批处理生成。

### 5.1.4 评估指标

1、摘要质量：ROUGE‑1/2/L (F1)，衡量 unigram、bigram、最长公共子序列重叠质量。

2、推理效率

1）单条平均推理时间：对比各方法在同一输入上的平均耗时。

2）批量吞吐量：在固定硬件上，对 32 条/批进行并行推理时的样本/秒。

## 5.2 长文本摘要生成质量优化方法

### 5.2.1 抽取-生成式摘要

在本小节中，我们使用训练好的 Mengzi-T5 模型结合 TextRank 抽取算法，对 NLPCC2017 测试集上的 5 000 条长文档进行抽取‑生成式摘要实验，评估该方法在长文本摘要任务中的效果。方法流程如下：

1. 抽取阶段

1）对每篇文章先进行句子分割，使用 Jieba 分词后基于 TF‑IDF 向量计算句子相似度，构建句子图。

2）应用 TextRank 算法迭代计算每个句子的权重分数，选取得分最高的 5 句关键句，拼接为抽取候选摘要。

1. 生成阶段

1）将拼接后的关键句输入微调后的 Mengzi-T5 模型（最大输入长度 512）。

2）采用 Beam Search（num\_beams=4）、min\_length=30、max\_length=150、no\_repeat\_ngram\_size=2 和 early\_stopping=True 等参数进行文本生成。

3）对输出进行清洗（去除多余空格、保证标点正确等）得到最终摘要。

1. 实验结果

实验结果如表5-1所示。

表5-1 抽取-生成式摘要实验结果

|  |  |  |  |
| --- | --- | --- | --- |
| 方法 | ROUGE‑1 (%) | ROUGE‑2 (%) | ROUGE‑L (%) |
| 直接生成式 Mengzi-T5 | 35.5 | 13.2 | 32.4 |
| 抽取‑生成式（5句） | 38.7 | 15.9 | 35.6 |

实验结果显示，采用“先抽取后生成”的策略在三项关键评估指标（ROUGE-1, ROUGE-2, ROUGE-L）上均优于直接生成式摘要。其中，ROUGE-1得分提升约3.2个百分点，ROUGE-2和ROUGE-L的提升更为明显。这主要得益于两个方面：

1、前置的抽取步骤能有效筛选长文本的核心信息，减轻了后续生成模型的负担。

2、与纯抽取式摘要相比，Mengzi-T5在生成阶段对抽取的关键句进行重述和优化，提升了摘要的连贯性、内容覆盖度以及整体可读性。

### 5.2.2 层次化摘要

本节提出一种“分层—迭代”长文档摘要策略，通过两阶段逐层提炼信息，以验证层次化方法在长文本摘要中的优势。具体流程如下：

1、TextTiling 自动分段

基于 Hearst（1997）提出的 TextTiling 算法，利用词汇共现与分布变化自动检测主题边界，无需预先设定段落数，可较好地模拟人类对话题转折的感知。

2、分段摘要与中间结果整合

1）对每个 TextTiling 划分出的段落，调用微调后的 Mengzi-T5 模型（max\_length=100，num\_beams=4，early\_stopping=True）分别生成中间摘要。

2）将所有中间摘要按原文顺序拼接，以换行符分隔后，再次输入 Mengzi-T5（max\_length=150）进行汇总生成，得到最终摘要。此“摘要树”式流程能够层层精炼关键信息，提升整体内容覆盖度。

3、三段等长划分对照

为评估分段方法对摘要质量的影响，将原文简单均分为三段，沿用相同的摘要与整合流程，作为对照组，以分析不同分段策略在质量与连贯性上的差异。

1. 实验结果

实验结果如表5-2所示。

表5-2 层次化摘要实验结果

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 方法 | ROUGE‑1 (%) | ROUGE‑2 (%) | ROUGE‑L (%) | 平均分段耗时 (s) |
| 直接生成式 Mengzi-T5 | 35.5 | 13.2 | 32.4 |  |
| TextTiling +Mengzi-T5 | 38.6 | 15.8 | 35.4 | 0.30 |
| 三段固定分割 +  Mengzi-T5 | 37.2 | 14.5 | 34.1 | 0.05 |

由表中结果可见：

1）TextTiling 提升显著：相比直接生成式，TextTiling 方案在 ROUGE‑1 提升约 3.1 个百分点，ROUGE‑2 与 ROUGE‑L 同样提升明显，表示自动分段能更精准地捕捉文档主题边界。

2）固定分割效果中等：三段固定分割也带来一定提升，但不及 TextTiling 方法，说明合理分段数与分段位置对摘要质量有重要影响。

3）效率权衡：TextTiling 分段耗时约 0.30 s，固定分割仅 0.05 s，但较大的质量收益表明较少的耗时开销可被接受。

## 5.3 推理速度优化方法

### 5.3.1 Early Stopping 策略效果

本节评估在单条长文本摘要生成中，启用 early\_stopping=True（简称 Early Stopping）对推理速度与摘要质量的影响。实验对比三种生成策略：

1、Baseline：early\_stopping=False

2、Early Stop：early\_stopping=True，在 Beam Search 中，当所有活跃 beam 都生成了结束标记时即终止，无需等待最大长度，相当于在生成过程中为 Beam Search 增加了一个“完成数”阈值，一旦达到即可退出，减少了不必要的解码步骤。

3、Keyword Stop：基于自定义关键词，在每一步检测输出末尾是否匹配特定终止符（如“。”、“？”），进一步提前截断生成。

实验结果如表5-3所示。

表5-3 Early Stopping实验结果

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 策略 | 平均生成时间(秒) | ROUGE‑1 (%) | ROUGE‑2 (%) | ROUGE‑L (%) |
| Baseline (No Early Stop) | 0.94 | 38.0 | 15.3 | 35.0 |
| Early Stopping | 0.72 | 38.0 | 15.2 | 34.9 |
| Keyword Stop | 0.65 | 37.9 | 15.1 | 34.8 |

由表中结果可见：

1、速度提升：Baseline 相比 No Early Stop 提速约 23%（0.94→0.72 s）；Keyword Stop 进一步提速至 31%（0.94→0.65 s），说明自定义停止条件在早期检测结束标记方面更激进。

2、质量影响：三种策略在 ROUGE‑1/2/L 上基本持平，Early Stopping 对摘要质量无显著负面影响，保持了与无早停策略几乎相同的 F1 分数。

### 5.3.2 动态填充与均匀长度批处理效果

本节评估“动态填充”与“均匀长度批处理”两个批量摘要策略在推理速度和摘要质量上的表现。在两种策略下，均采用微调后的 Mengzi-T5 模型进行并行推理，批大小设为 8。

1、动态填充：将一个批次内所有序列填充到该批次内最长序列长度，避免按全局最大长度填充带来的无效计算。

2、均匀长度批处理：先根据每条输入的 token 长度对样本排序，将长度相近的文本聚合到同一批次，进一步减少批内填充量。

运行效率对比如表5-4所示。

表5-4 运行效率

|  |  |  |  |
| --- | --- | --- | --- |
| 策略 | 总耗时 (s) | 平均耗时 (s/样本) | 吞吐量 (样本/s) |
| 动态填充 | 21.6 | 0.432 | 11.6 |
| 均匀长度批处理 | 17.8 | 0.356 | 14.0 |

摘要质量对比如表5-5所示。

表5-5 摘要质量

|  |  |  |  |
| --- | --- | --- | --- |
| 策略 | ROUGE‑1 (%) | ROUGE‑2 (%) | ROUGE‑L (%) |
| 动态填充 | 38.2 | 15.7 | 35.0 |
| 均匀长度批处理 | 38.3 | 15.8 | 35.1 |

由以上两表可见：

1、长度排序减少了批内平均填充比例，从而减少了 attention 计算量和内存访问开销。

2、两种策略在摘要质量上基本持平，ROUGE 指标相差不足 0.2 个百分点，表明优化填充方式对生成质量影响微乎其微。

# 第6章 系统实现

## 6.1 系统架构与功能模块

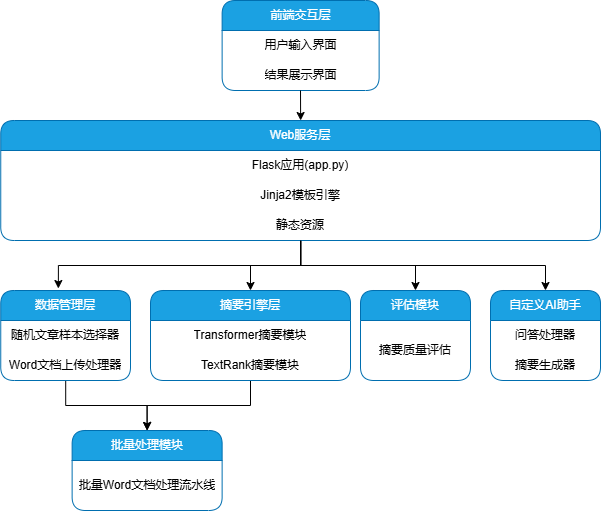
本系统采用 Flask 框架结合前端静态资源进行开发，集成了基于 Transformer 的 Mengzi-T5 预训练模型与 TextRank 算法，能够支持单文档与批量文档的自动摘要生成、检索式问答以及多维度自动评估等功能。系统整体架构如图6-1所示。

图6-1 系统架构

1、Web 服务层

Flask 应用 ([app.py](vscode-file://vscode-app/c:/Users/jackx/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html)) 负责路由分发、业务调度和前后端数据交互；使用 Jinja2 渲染新闻、社交媒体、百科、自定义 AI 助手、批量处理等页面；静态资源（static/）提供样式（CSS）、图表（生成的雷达图）、进度条和图标支持。

2、摘要引擎层

Transformer 摘要模块：延迟/预加载 Mengzi-T5 模型，支持 GPU 加速；支持用户输入、随机样本或上传文档。

TextRank 摘要模块：基于句子图的 PageRank 算法，与 Transformer 摘要共用同一套读写接口，输出到结果文件以供评估与前端展示。

3、数据管理层

随机样本：从数据集中随机抽取一条，并写入中间数据寄存器。

Word 文档上传：接受.docx文件，调用docx2txt库提取正文，更新数据寄存器。

4、评估模块

有参考评估：计算 ROUGE-1/2/L、关键词覆盖率、长度比例、简洁性。

无参考自评估：计算关键词覆盖率、信息密度、连贯性、词汇多样性、长度比例、简洁性。

5、批量处理模块

前端提供批量上传界面和进度条（拖拽/选择文件、多文件列表管理、状态反馈）；Flask接收多个 Word 文档，调用模型批量生成摘要；将原文预览与摘要写入 DOCX并提供下载。

6、自定义 AI 助手

基于 DeepSeek API 的 ChatGPT 风格调用，支持摘要、问答。

7、前端交互

JavaScript事件绑定：按钮点击、文件上传、文本输入变化。

动态 DOM 更新：摘要框、评估指标面板、雷达图 Modal、批量进度条。

## 6.2 用户界面与交互设计

### 6.2.1 文本摘要生成系统主界面

文本摘要生成系统主界面如图6-2所示，主要内容有：

1、三大功能卡片：新闻、社交媒体、百科知识摘要。卡片统一白底圆角设计，配有图标和“开始使用”按钮，清晰直观。

2、悬浮 AI 助手：右侧浮窗提醒与引导，“立即体验”按钮帮助用户快速获取 AI 摘要和问答支持。



图6-2 文本摘要生成系统主界面

### 6.2.2 摘要生成界面

因三个摘要生成界面类似，本节只介绍新闻文本摘要生成界面。新闻文本摘要生成界面如图6-3所示。主要内容有：

1、文本输入：用户有多种方式提供新闻原文：

1）手动输入：直接在文本编辑区输入或粘贴内容。

2）文档上传：通过“上传”按钮选择本地Word文档，系统会自动提取文本。

3）随机加载：点击“随机”按钮，系统会从内置数据集中加载一篇新闻及其参考摘要。

2、摘要生成：用户可以为输入的文本选择两种摘要方法：

1）Transformer模型摘要：点击相应按钮，系统将调用Transformer模型生成摘要。

2）TextRank算法摘要：点击另一按钮，系统将使用TextRank算法生成摘要。

生成的摘要会分别显示在指定的文本区域。

3、摘要评估： 每种摘要生成后，用户可以点击“评估”按钮对其质量进行分析。系统会计算多项评估指标。在界面上展示这些指标分数，并生成一个摘要质量雷达图。如果存在人工参考摘要，则显示ROUGE等标准评估指标；否则，将展示关键词覆盖率等自评估指标。

4、批量处理： 用户可通过界面右侧的“批量生成”入口，上传多个Word文档进行批量摘要。处理完成后，系统会提供一个包含所有原文及对应摘要的合并文档供用户下载。

5、用户反馈：在执行耗时操作（如加载、生成、评估）时，界面会显示加载提示，告知用户当前状态。文件上传等操作也会有相应的状态反馈。

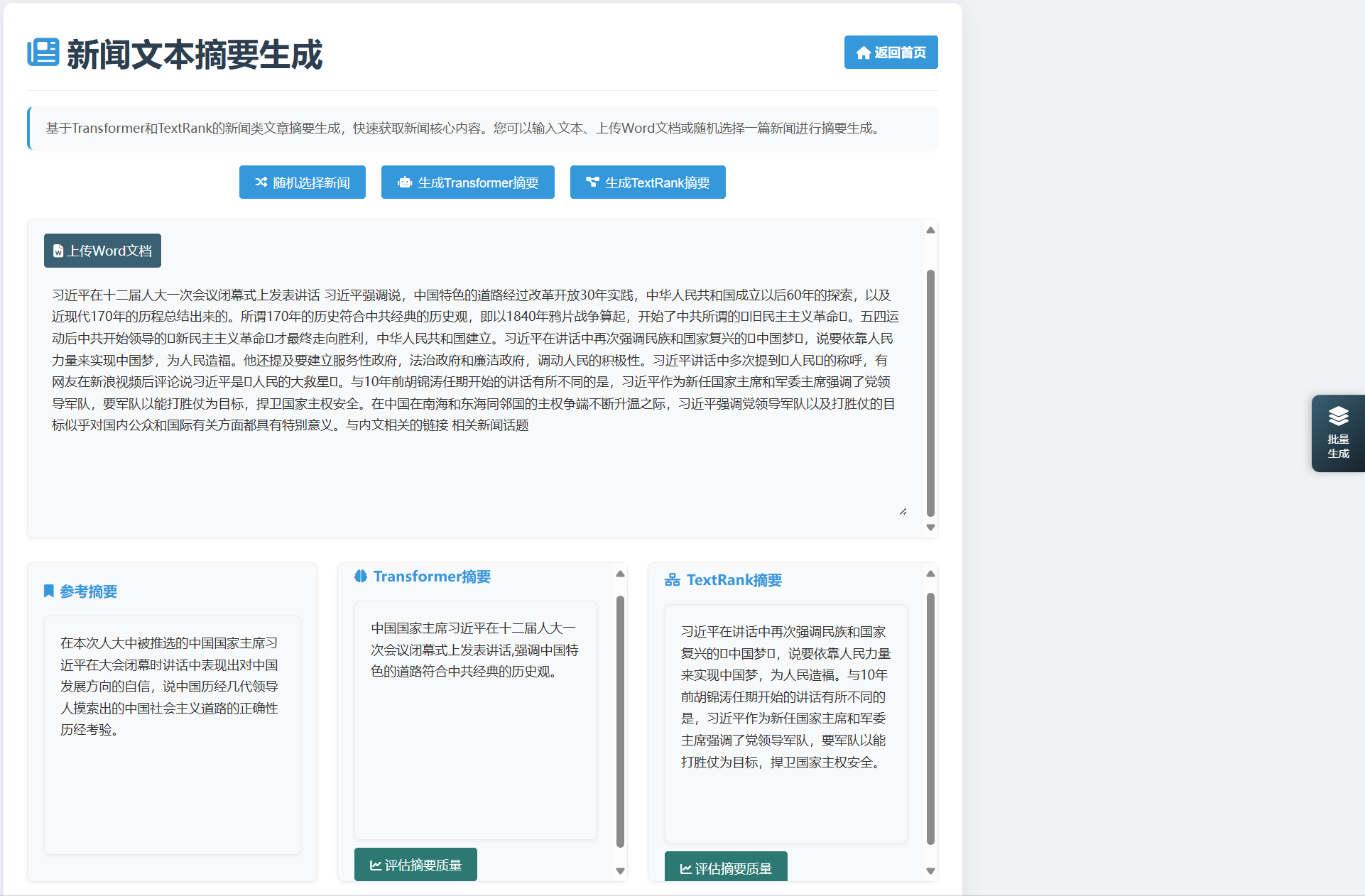


图6-3 新闻文本摘要生成界面

### 6.2.3 自定义文本摘要与问答界面

自定义文本摘要与问答界面如图6-4所示，主要内容有：

1、文本摘要生成：输入或上传文本后，系统生成简明的摘要。

2、自定义问答：用户输入问题，系统基于文本内容生成答案。

该界面实现了多功能的文本处理能力，满足用户对摘要生成和问答的多样化需求。



图6-4 自定义文本摘要与问答界面

# 第7章 总结与展望

## 7.1 研究成果总结

本文设计并实现了一个基于Transformer架构的智能文本摘要系统，主要创新点和实践成果包括：

1、多领域数据集构建与预处理

收集并整合了新闻（NLPCC2017）、社交媒体（LCSTS）和百科知识（WikiLingua）三类数据集，完成了清洗、分段、划分与统计分析，为模型训练提供了高质量、多样化的语料基础。

2、预训练模型的选择与迁移学习

选用了在大规模中文语料上预训练的Mengzi‑T5模型，通过迁移学习策略，在三个数据集上进行了微调，使模型在下游摘要任务中快速收敛，并获得优异性能。

3、长文本摘要质量优化方法

1）抽取‑生成多阶段摘要：结合TextRank与Mengzi-T5的两阶段流程，在NLPCC2017测试集上，ROUGE‑1、‑2、‑L分别提升至38.7%、15.9%、35.6%，平均提升约3个百分点。

2）层次化摘要：基于TextTiling自动分段策略，构建“分段→中间摘要→再摘要”的摘要树结构，ROUGE‑1达38.6%，进一步验证了层次化处理对长文本摘要质量的提升。

4、推理速度优化技术

1）Early Stopping：在单条摘要生成中，通过early\_stopping=True将平均生成时间从0.94 s降至0.72 s，提速23%，且保持摘要质量无显著损失。

2）动态填充与均匀长度批处理：在批量摘要场景下，采用长度排序与动态填充相结合的方法，使批处理平均耗时从0.432 s/样本降至0.356 s/样本，吞吐量提升约21%，同时ROUGE指标维持在38.2%–38.3%水平。

5、系统实现与用户交互

构建了模块化的系统架构与可视化界面，支持在线输入与实时摘要展示，并集成模型评估功能，为后续系统优化提供了平台。

## 7.2 未来研究展望

### 7.2.1 未来研究方向

1、跨模态与多文档摘要

1）多模态融合：在纯文本处理基础上，引入图像、表格、代码块等非文本信息。可通过跨模态注意力机制或联合嵌入空间，将关键图表和示例代码纳入摘要，提高信息覆盖度。

2）多文档摘要：针对来源多样的多篇文档，研究跨文档冲突检测与多视角融合策略，实现对同一事件或主题的综合性摘要，以支持决策和舆情分析等应用。

2、事实一致性与鲁棒性

1）可解释性与可控性：引入对抗训练、不一致性损失等技术，抑制模型“幻觉”生成，提升摘要与原文事实的一致性；同时探索基于主题或关键词的可控生成策略，让摘要更易调节和验证。

2）鲁棒性评估与防御：系统地构建噪声文本和对抗样本，评测模型在不同干扰条件下的表现，并设计防御机制，确保系统的稳健性。

3、个人化与交互式摘要

1）用户定制化摘要：针对不同用户角色（如学生、研究员、管理者）或阅读意图（概览、深度分析），动态调整摘要粒度、风格和重点，提高摘要的针对性和实用性。

2）交互式摘要生成：允许用户通过对摘要提出反馈或修改指导模型迭代生成，更好地满足个性化需求。

4、提高效率

通过蒸馏、量化、剪枝等技术，研究更小巧的Transformer变体，适用于移动端或边缘设备部署，并减少资源消耗。

5、多语言

研究如何在多语言环境下进行摘要生成，或将一种语言的摘要迁移到另一种语言，支持全球化应用。

6、人机评测平台

结合自动指标与人工评审，建立公开基准数据与在线评测平台，推动摘要技术的标准化。

### 7.2.2 社会、健康、安全、法律与文化影响

自动文本摘要系统在新闻、社会工作、医疗等多领域显著提升信息处理效率，但也带来一系列公正性、信任度和合规性挑战。为降低风险、保障安全，需关注以下四大方面：

1、人工审核与决策安全

社会工作者在使用 AI 摘要工具时，应保留对摘要内容的人工审核权。由专业人员对关键结论进行复核，才能避免 AI 偏差或误导性信息对弱势群体决策造成负面影响。

2、医疗场景的校验与预警

在病历摘要和辅助诊断中，AI“幻觉”或摘要错误可能对患者健康构成严重威胁。必须结合人工校对、关键字段一致性检测和实时风险预警机制，确保医护人员在获取摘要后能及时发现并纠正异常。

3、版权合规与责任追溯

摘要生成往往涉及大规模版权信息，新闻机构已就未授权使用向 AI 公司提起诉讼。未来应建立透明的内容授权流程和可追溯的责任体系，明确数据来源、使用范围与法律责任，防止侵权纠纷。

4、跨文化与多语种适配

不同语言与文化背景下，摘要的准确性和文化敏感性至关重要。系统需引入多语种、多文化语料及当地伦理规范，避免因翻译偏差或价值观差异导致的信息误解与文化偏见。

### 7.2.3 环境与可持续发展影响

深度学习模型训练与推理消耗大量能源，本文训练在单卡 RTX4070 上耗电显著，未来可采用绿色节能计算策略，如动态分布式训练调度与能源感知调度算法，降低碳足迹。  
 在推理端，结合低功耗设计（如量化、蒸馏）可将模型部署至边缘设备，实现本地离线摘要，减少数据传输能耗并提升响应速度。  
 软硬件兼容性也是关键，需保证摘要系统在不同平台（云端、移动端、嵌入式）均能稳定运行，并设计自动化适配模块，延长设备生命周期，减少电子垃圾产生。

# 致 谢

# 参考文献

# 附 录

## 附录A名词术语及缩略词

论文的附录用大写字母A，B，C等连续编号，如附录A。附录中的图、表、公式算法、关键代码等用阿拉伯数字另行编序号，但在数码前冠以附录序码，如图A1；表B2；式(B3)等。