****

**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 paper 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 paper 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](#_Toc69482696)

[第2章 相关研究工作 2](#_Toc69482697)

[2.1 基于语义的网页信息提取算法 2](#_Toc69482698)

[2.2 Block Level PageRank算法 2](#_Toc69482699)

[2.2.1 Block Level Web Graph 2](#_Toc69482700)

[2.2.2 Page Graph 3](#_Toc69482701)

[2.3 基于视觉的网页分块算法 3](#_Toc69482704)

[第3章 搜索引擎Quark模块 4](#_Toc69482705)

[3.1 模块结构 4](#_Toc69482706)

[3.2 网页分块算法 5](#_Toc69482707)

[3.3 网页主题内容提取 6](#_Toc69482708)

[3.4 算法效果演示 6](#_Toc69482709)

[第4章 SEWM中文Web信息检索评测 7](#_Toc69482710)

[4.1 评测任务介绍 7](#_Toc69482711)

[4.2 评测格式 7](#_Toc69482712)

[4.3 评测结果 7](#_Toc69482713)

[4.3.1 主题型网页发现任务评测结果 7](#_Toc69482714)

[4.4 评测综述 7](#_Toc69482716)

[第5章 网页分块的分布式应用 8](#_Toc69482717)

[第6章 总结与展望 9](#_Toc69482718)

[6.1 总结 9](#_Toc69482719)

[6.2 展望 9](#_Toc69482720)

[致 谢 10](#_Toc69482721)

[参考文献 11](#_Toc69482722)

[附 录 12](#_Toc69482723)

[附录A名词术语及缩略词 12](#_Toc69482724)

# 第1章 序言

## 1.1 研究背景与意义

随着互联网、社交媒体和工业文档的快速增长，信息过载已成为制约人们获取、处理和传播知识的主要瓶颈。长文本在新闻报道、技术白皮书等场景中广泛存在，其篇幅冗长、结构复杂，直接阅读和理解不仅耗时费力，而且容易导致重要信息的遗漏。智能文本摘要技术通过自动提炼源文档核心内容，能够显著降低用户阅读成本，提高信息获取效率，因而在信息检索、舆情监测、知识管理、智能问答等领域具有广阔的应用前景。

传统的文本摘要方法主要包括基于统计和图模型的抽取式算法（如TextRank、LSA）及基于RNN的生成式算法（如Seq2Seq+注意力机制）。虽然在简短文本或中等长度文档上取得了一定成果，但在超长文本摘要任务中依然面临如下挑战：一是输入长度受限，模型难以处理数千至上万词的长文；二是信息覆盖不足，抽取式方法易漏掉关键信息，生成式方法易出现重复或歧义；三是推理速度缓慢，难以满足实时或大规模批量处理需求。

近年来，Transformer架构及其衍生的预训练语言模型（如BERT、T5、Mengzi-T5等）凭借自注意力机制在捕捉长距离依赖、并行计算和迁移学习能力方面展现出显著优势，为长文本摘要研究带来新机遇。然而，直接应用大规模预训练模型往往伴随高昂的计算和存储开销，并且在超长文档场景中仍需进一步优化输入表示、解码策略和推理效率。

基于上述背景，本文旨在设计并实现一套兼顾质量与速度的智能长文本摘要系统：一方面探索层次化与多阶段抽取-生成混合策略，以增强对文档全局和局部信息的捕获；另一方面引入Early Stopping、动态填充与均匀长度批处理等推理优化技术，以提升摘要生成的实时性与资源利用率。本研究不仅能够丰富长文本自动摘要理论体系，也为智能文本处理在实际应用场景中提供创新方案和技术参考。

## 1.2 研究目标与内容

本研究旨在设计并实现一个基于Transformer架构的智能文本摘要系统，提升长文本处理的效率与质量。针对当前Transformer模型在处理长文本时面临的输入长度限制和信息覆盖不足等问题，结合迁移学习策略和多阶段摘要方法，提出一套优化方案，以实现高质量、快速的文本摘要生成。具体研究内容包括：

1、多领域摘要数据集构建与预处理：收集并构建包含新闻、技术文档和社交媒体等多领域文本的摘要数据集，进行数据清洗、标注和划分，确保训练数据的质量和多样性。

2、基于Transformer的摘要模型设计与优化：以预训练模型Mengzi-T5为基础，结合迁移学习策略，在下游摘要任务上进行微调，提升模型在不同领域文本摘要任务中的性能。

3、摘要质量优化方法研究：引入抽取-生成多阶段摘要和迭代式摘要方法，先对长文档执行抽取式摘要，选取关键句或段落，再在这些选取内容上进行抽象生成，降低输入长度限制，增强信息覆盖，提升长文本摘要的质量。

4、推理优化技术研究：采用动态填充、均匀长度批处理和Early Stopping等推理优化技术，提高摘要生成的效率，降低计算资源消耗，加快摘要生成速度。

5、系统实现与性能评估：实现一个支持在线输入文本并实时生成摘要的智能摘要系统，提供可视化界面，展示输入文本、生成摘要以及摘要质量评估结果。通过ROUGE、BLEU等指标对比基线方法，验证所提方法在长短文本摘要任务中的优越性。

## 1.3 论文结构安排

为清晰地呈现研究过程及成果，本文共分为七章，结构安排如下：

第一章绪论。介绍研究的背景与意义，明确本文的研究目标和主要内容，并对全文结构进行总体概述，为后续章节奠定基础。

第二章相关研究工作。回顾和评述文本摘要领域的经典方法与最新进展，重点介绍 Transformer 架构与预训练模型的发展，以及抽取式与生成式摘要技术的异同与应用场景。

第三章数据集构建与预处理。说明所使用数据集的来源、组成及其特点，详细描述数据清洗、标注和划分流程，为模型训练提供高质量的实验样本。

第四章模型训练与微调。阐述预训练模型的选择理由和迁移学习策略，给出模型训练的具体步骤、超参数设置及硬件环境，以保证实验可重复性。

第五章实验设计与结果分析。首先介绍实验的设置和评估指标，然后针对长文本摘要生成质量，分别讨论抽取–生成式摘要和层次化摘要的优化方法；接着分析推理速度优化手段，包括 Early Stopping 策略和动态填充与均匀长度批处理效果，并给出定量与定性对比。

第六章系统实现。从系统架构层面剖析整体流程与功能模块，重点展示摘要生成服务的接口设计、部署方案，以及用户界面与交互设计，说明系统如何实现端到端的自动摘要功能。

第七章总结与展望。概括本文的主要研究成果与创新点，系统性总结模型性能与系统实现效果，并对未来进一步提高摘要质量、拓展多语种或跨领域应用提出可行的研究方向和展望。

# 第2章 相关研究工作

## 2.1 文本摘要技术概述

自动文本摘要旨在从冗长文本中提炼出核心信息，生成简洁且准确的短文，以帮助用户高效获取关键信息 。根据输入文档数量和应用场景，摘要任务可分为单文档摘要（Single‑Document Summarization）和多文档摘要（Multi‑Document Summarization）两大类，前者针对单篇文章生成摘要，后者需融合多篇相关文档的内容。  
 在实际应用中，摘要系统广泛应用于新闻速览、法律文书提炼、科研文献检索等场景，其输出长度通常为源文 10%–50% 左右，以平衡信息覆盖与阅读成本。早期方法侧重基于统计特征（如词频、句子位置）进行简易筛选；随着图模型、序列标注、线性规划和子模函数等技术的发展，自动摘要进入特征工程时代。近年深度学习兴起后，预训练语言模型（如 BERT、T5、BART）结合 Seq2Seq 框架，显著提升了摘要质量，但同时引入了计算资源需求高和输入长度受限的问题。  
 常用评估指标包括 ROUGE（重叠率衡量）、BLEU（n‑gram 精确度）等自动化度量，以及人工评审（连贯性、一致性）。  
 当前研究热点还包括可控摘要（控制摘要长度、焦点）、事实一致性（减少“幻觉”问题）和多模态摘要（文本+图表）等方向 。

## 2.2 Transformer架构与预训练模型

Transformer是一种基于自注意力机制的深度学习架构，最早由Vaswani等人在2017年提出，用于解决序列到序列的任务，如机器翻译。与传统的循环神经网络（RNN）相比，Transformer完全摒弃了序列对齐的结构，依赖于自注意力机制来建模序列中各个位置之间的依赖关系，从而实现并行计算，提高了训练效率和模型性能。

Transformer架构主要由编码器（Encoder）和解码器（Decoder）组成，每个编码器层包含多头自注意力机制和前馈神经网络，每个解码器层在此基础上增加了编码器-解码器注意力机制。通过堆叠多个这样的层，Transformer能够捕捉复杂的语言结构和语义信息。Transformer架构如图2-1所示。

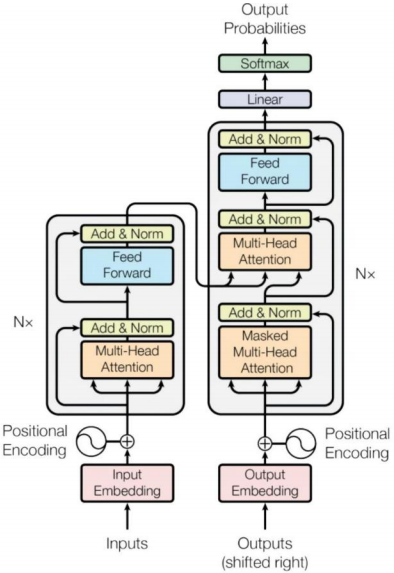


图2-1 Transformer架构

基于Transformer架构，研究者开发了多种预训练语言模型，这些模型在大规模文本数据上进行预训练，学习通用的语言表示，然后通过微调（Fine-tuning）适应具体的下游任务。典型的预训练模型包括：

1、BERT（Bidirectional Encoder Representations from Transformers）：采用双向编码器结构，通过掩码语言模型（Masked Language Model）和下一句预测（Next Sentence Prediction）任务进行预训练，擅长处理句子级别的理解任务。

2、GPT（Generative Pre-trained Transformer）系列：采用解码器结构，利用自回归语言模型进行预训练，擅长生成任务，如文本生成、对话系统等。

3、T5（Text-To-Text Transfer Transformer）：将所有的NLP任务统一为文本到文本的形式，采用编码器-解码器结构，增强了模型在多种任务上的泛化能力。

这些预训练模型在自然语言处理的各个领域取得了显著的成果，成为当前研究和应用的主流方法。在本研究中，选用了Mengzi-T5作为基础模型，结合迁移学习策略，在新闻、技术文档和社交媒体等多领域数据集上进行微调，旨在提升长文本摘要的质量和效率。

## 2.3 抽取式与生成式摘要方法

### 2.3.1 抽取式摘要

抽取式摘要通过选取原文中最具代表性的句子或段落，直接拼接生成摘要。该方法保留了原文表述，易于实现，计算效率高。具体方法包括：

1、统计特征：基于词频-逆文档频率（TF‑IDF）、句子位置、句长等手工特征进行句子评分。

2、图排序：如 TextRank、LexRank 等，将句子视为图中节点，边权由句子相似度决定，通过 PageRank 计算节点重要性。

3、监督学习：将句子抽取视作二分类或回归问题，使用 SVM、决策树或神经网络模型预测句子得分。  
 TextRank 是基于图的无监督抽取式摘要算法，最早由 Mihalcea 和 Tarau 提出，灵感来自 PageRank 网页排序算法。其主要流程如下：

1、构建句子图

2、初始化分数

3、迭代计算

4、收敛与排序

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 的优势在于：

1. 无需监督数据，通用性强。
2. 利用图模型能捕捉句子间全局结构信息，避免单纯基于局部特征。

### 2.3.2 生成式摘要

生成式摘要借助语言模型重新组织语言，生成新的句子表达原文含义。其基本思路是将摘要任务视为“文本到文本”转换，常用 Seq2Seq 编码-解码架构。具体方法包括：

1、Seq2Seq + Attention：早期 RNN‑Attention 模型推动了摘要生成技术发展。

2、预训练模型：如 T5、BART 等，通过大规模语料预训练后，再在摘要数据集上微调，实现更自然连贯的生成。

3、Copy/Pointer 机制：增强生成模型对原文词汇的复制能力，提升专业术语和人名的准确性。  
 生成式方法优势在于摘要更加灵活自然，但计算成本高，且易出现事实错误或重复内容。在本研究中，选用了Mengzi-T5作为基础模型，结合迁移学习策略，在新闻、技术文档和社交媒体等多领域数据集上进行微调，旨在提升长文本摘要的质量和效率。Mengzi-T5 架构跟 T5 本质相同，如图2-2所示。

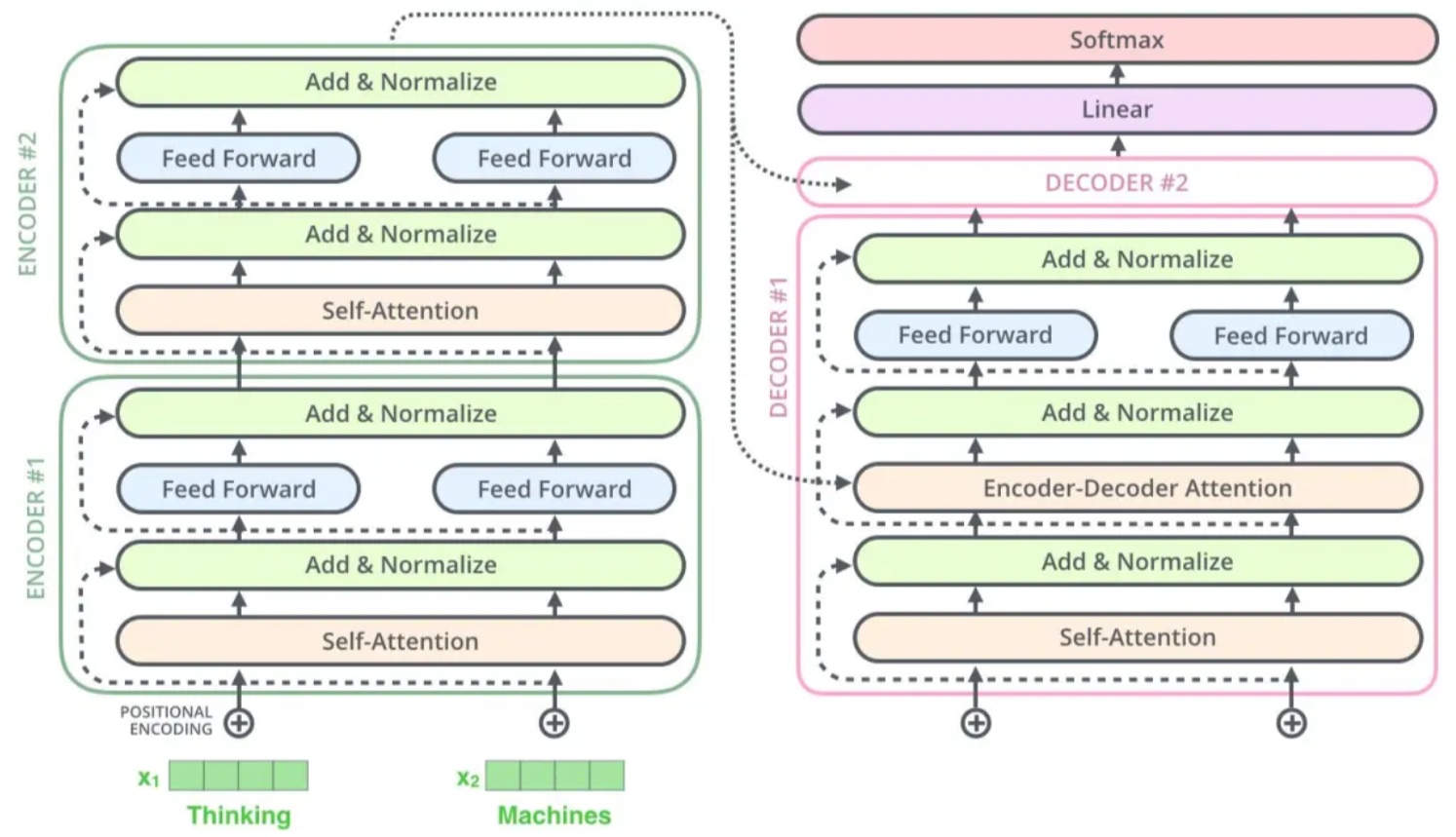


图2-2 T5架构

# 第3章 数据集构建与预处理

## 3.1 数据集来源与组成

为了构建具备跨领域泛化能力的文本摘要系统，本文选取了三个具有代表性的中文数据集，分别覆盖新闻、社交媒体和百科知识等不同文本类型。这三个数据集在文本风格、长度分布和摘要形式上各具特点，为模型的训练和评估提供了多样化的语料基础。

### 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 会议上发布，原用于新闻标题的多分类任务。该数据集包含来自新浪、今日头条等新闻门户的短文本标题，涵盖娱乐、体育、财经、科技等18个类别。每条数据由新闻标题和对应的类别标签组成。虽然该数据集原用于分类任务，但其短文本特性使其适合作为摘要任务中的输入文本。在本研究中，我们将新闻标题视为待摘要的文本，结合其他数据集进行统一的摘要建模。

### 3.1.2 LCSTS 中文短文本摘要数据集

LCSTS（Large-scale Chinese Short Text Summarization Dataset）是由华中科技大学于2015年发布的中文短文本摘要数据集，源自新浪微博的真实用户发布内容。该数据集包含超过200万条微博文本及其对应的人工撰写摘要，具有高度的抽象性和多样性。LCSTS 的数据分为三个部分：第一部分包含自动抓取的微博-摘要对，第二部分和第三部分则由人工标注相关性得分，确保数据质量。该数据集广泛用于中文自动摘要研究，尤其适合训练和评估生成式摘要模型。

### 3.1.3 WikiLingua多语言摘要数据集

WikiLingua 是由康奈尔大学等机构于2020年发布的大规模多语言抽象摘要数据集，源自 WikiHow 网站的“如何做”类文章。该数据集涵盖18种语言，包括中文，提供了约18,887对文章及其摘要。WikiLingua 的特点在于其跨语言、一致性强的摘要对齐方式，适用于多语言和跨语言摘要任务。在本研究中，我们选取其中的中文部分，用于评估模型在百科知识类长文本摘要任务中的性能。

## 3.2 数据清洗与划分

### 3.2.1 NLPCC 2017 数据处理

NLPCC 2017 第三任务（Task 3）为“单文档摘要（Single Document Summarization）”，旨在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语料库，包含 50,000 篇带有人类撰写摘要的新闻文章和 50,000 篇无摘要的新闻文章，供模型训练和半监督学习使用。本文将该数据集按 8:1:1 的比例划分为训练集、验证集和测试集，分别包含 40,000、5,000 和 5,000 条样本。

在数据预处理过程中，本文采取了以下措施：

1、格式统一：将每条样本整理为统一的 JSON 格式，包含“正文”和“摘要”两个字段，便于后续模型训练和评估。

2、文本清洗：去除正文和摘要中的特殊字符、HTML 标签和多余的空格，确保文本内容的纯净性。

### 3.2.2 LCSTS 数据处理

LCSTS 数据集包含超过200万条微博文本及其对应的人工摘要，数据质量较高，但仍需进行一定的清洗和筛选。具体处理步骤如下：

1、文本清洗：去除文本中的特殊字符、表情符号和多余的空格，统一文本格式。

2、长度限制：筛选长度适中的微博文本，剔除过长或过短的样本，确保模型训练的稳定性。

由于原始训练集规模过大，本文从中随机抽取 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）。

### 3.2.3 WikiLingua 数据处理

WikiLingua 数据集源自 WikiHow 网站的“如何做”类文章，包含约18,887对多语言文章及其摘要，经数据清洗后剩6000余条。该数据集结构清晰，但仍需进行一定的预处理：

1、段落合并：将文章中的多个段落合并为完整的文档，确保输入文本的连贯性。

2、文本清洗：去除HTML标签、特殊字符和多余的空格，统一文本格式。

3、摘要提取：从文章中提取对应的摘要部分，确保摘要内容的完整性和准确性。

本文将该数据集按 8:1:1 的比例划分为训练集、验证集和测试集，分别包含 5,233、654 和 654 条样本。

# 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

在本研究中，为了有效地处理中文文本摘要任务，选择了基于 Transformer 架构的 Mengzi-T5 预训练模型作为基础模型。该模型由 Langboat 团队开发，采用与 Google 提出的 T5（Text-to-Text Transfer Transformer）模型相同的架构，具备强大的文本生成能力，适用于多种自然语言处理任务，包括文本摘要、问答和翻译等。Mengzi-T5 模型在大规模中文语料上进行预训练，具备良好的中文语言理解和生成能力。其设计目标是提供一个通用的预训练模型，通过在特定任务上的微调，使其适应各种下游任务的需求。

为了充分利用预训练模型的能力，本文采用了迁移学习策略。具体而言，首先在大规模中文语料上进行预训练，学习通用的语言表示；然后，在特定的文本摘要任务上进行微调，使模型能够更好地理解和生成摘要内容。迁移学习的优势在于：

1、提高模型性能：通过在大规模语料上预训练，模型能够学习到丰富的语言知识，从而在下游任务中表现更好。

2、减少训练数据需求：由于模型已经在大量数据上进行预训练，微调时只需要较少的标注数据即可达到良好的性能。

3、加快训练速度：预训练模型提供了良好的初始化参数，使得微调过程更快收敛，节省训练时间。

在实际应用中，本文将 Mengzi-T5 模型应用于多个中文文本摘要数据集，包括 NLPCC 2017、LCSTS 和 WikiLingua。通过在这些数据集上的微调，使模型能够适应不同领域的文本摘要任务，提高摘要的质量和准确性。

## 4.2 模型训练与参数设置

在本节中，我们详细介绍在 LCSTS、NLPCC2017 和 WikiLingua 三个数据集上，对 Mengzi‑T5 模型进行下游微调的训练流程与超参数配置。所有实验均在单卡 NVIDIA RTX4070（8 GB 显存）环境下进行，PyTorch 1.11 与 Transformers 4.24.0 实现。

## 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。

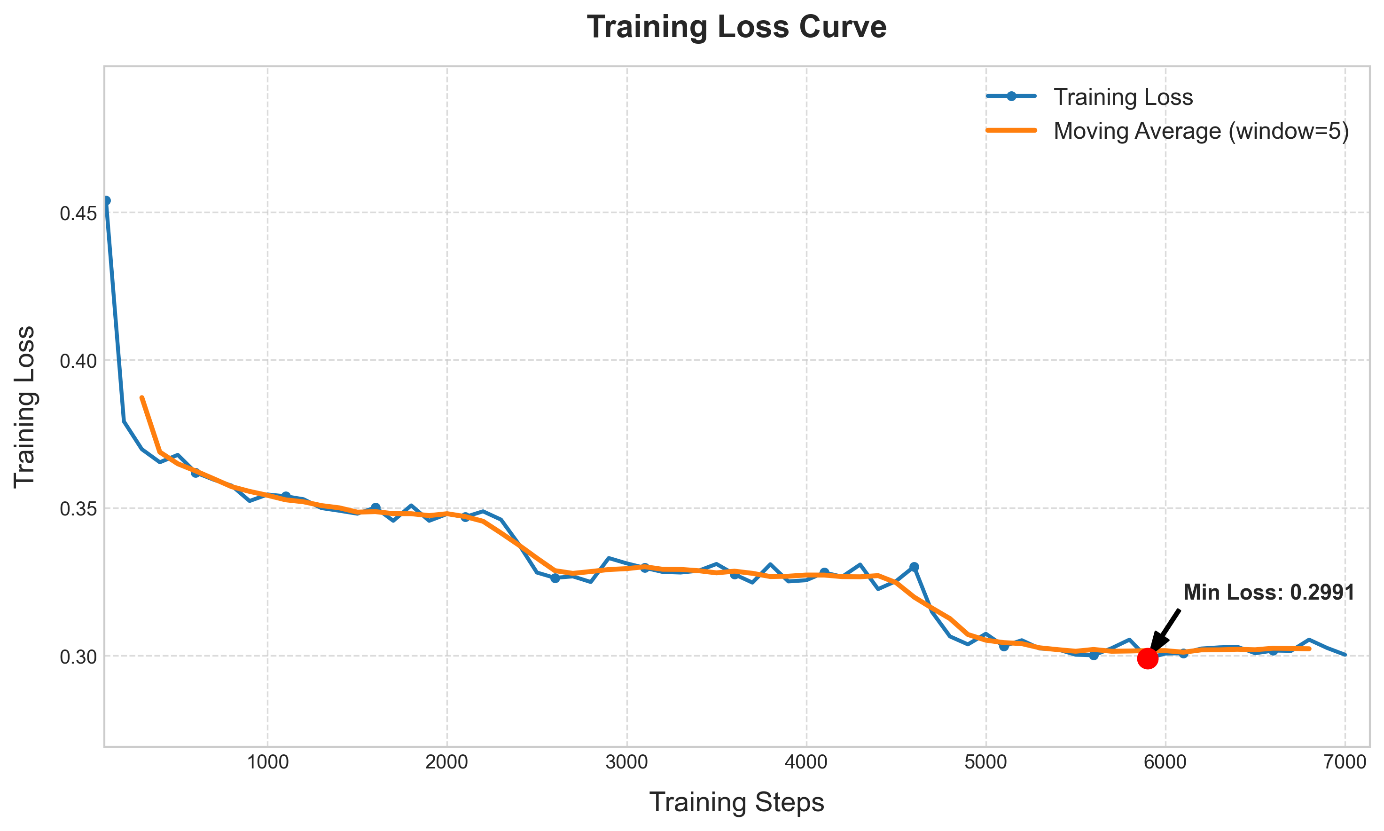


图4-1 训练损失曲线

图4-2则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。

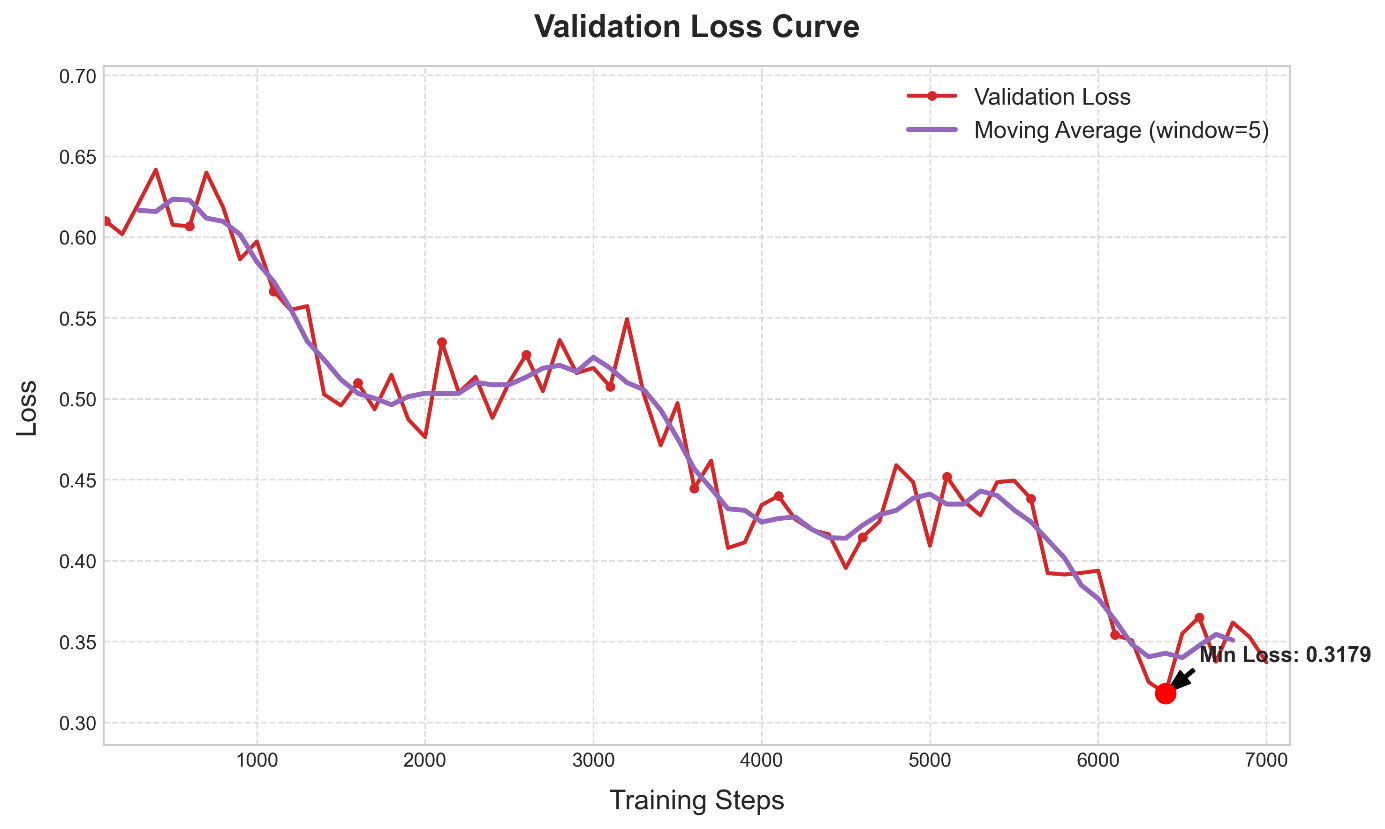


图4-2 验证损失曲线

# 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：单条平均推理时间：对比各方法在同一输入上的平均耗时。

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 |

抽取‑生成式方法在三项指标上均超越了直接生成式摘要，ROUGE‑1 提升约 3.2 个百分点，ROUGE‑2 与 ROUGE‑L 提升更显著，说明先行抽取能有效聚焦长文本核心信息，降低生成模型的无关预测负担；摘要的连贯性与内容覆盖度相较纯抽取式也有明显提升，生成阶段的 Mengzi-T5 能对前序抽取得到的关键句进行有效重述，使摘要更具可读性。

### 5.2.2 层次化摘要

本节采用“分层—迭代”策略，对长文本进行分段、逐段摘要，再将中间摘要整合生成最终摘要，以验证层次化方法在长文档摘要中的效果。方法流程如下：

1. TextTiling 自动分段

使用 Hearst（1997）提出的 TextTiling 算法，根据词汇共现和分布变化自动检测主题边界，无需人为指定段数，能模拟人类对话题变化的感知。

1. 分段摘要与整合

1）对每个 TextTiling 分得的段落使用微调后的 Mengzi-T5 模型（max\_length=100、num\_beams=4、early\_stopping=True）生成中间摘要。

2）将所有中间摘要按原文顺序拼接，以换行分隔，输入 Mengzi-T5（max\_length=150）进行第二次生成，得到最终摘要。此“摘要树”式流程可层层提炼信息，提升内容覆盖度。

1. 三段固定分割对照

将文本简单平均划分为三段，作为对照组；其他步骤与 TextTiling 流程相同，考察分段数目与分段方法对摘要质量的影响。

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，表明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。

3、节省解码步骤：Early Stopping 策略平均减少了约 **5** 次解码步（从平均 45 步减至约 40 步），与相关研究中减少重复生成、加速收敛的结论一致。

### 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

本文所述系统基于 Flask+前端静态资源构建，集成 Transformer（Mengzi-T5模型）与 TextRank 算法，提供单文档与批量文档的摘要生成、检索式问答与多维度评估功能。系统架构图如图6-1所示。

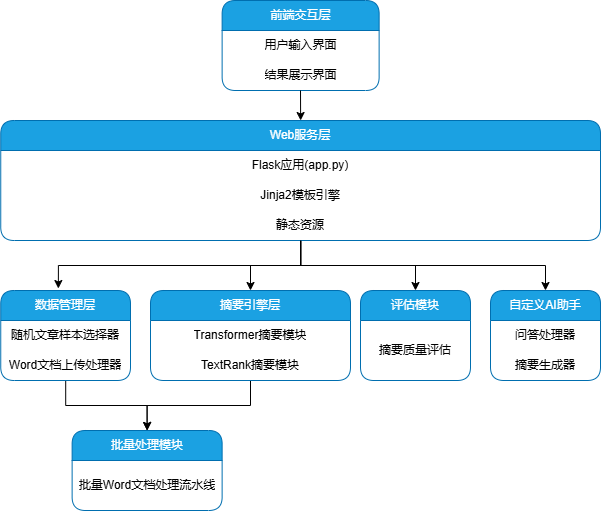


图6-1 系统架构

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、界面布局：界面采用直观的三段式布局：顶部为导航区，中部为主要操作区，右侧提供一个便捷的批量处理入口。

2、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。

3、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。

4、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。

5、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。

6、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。

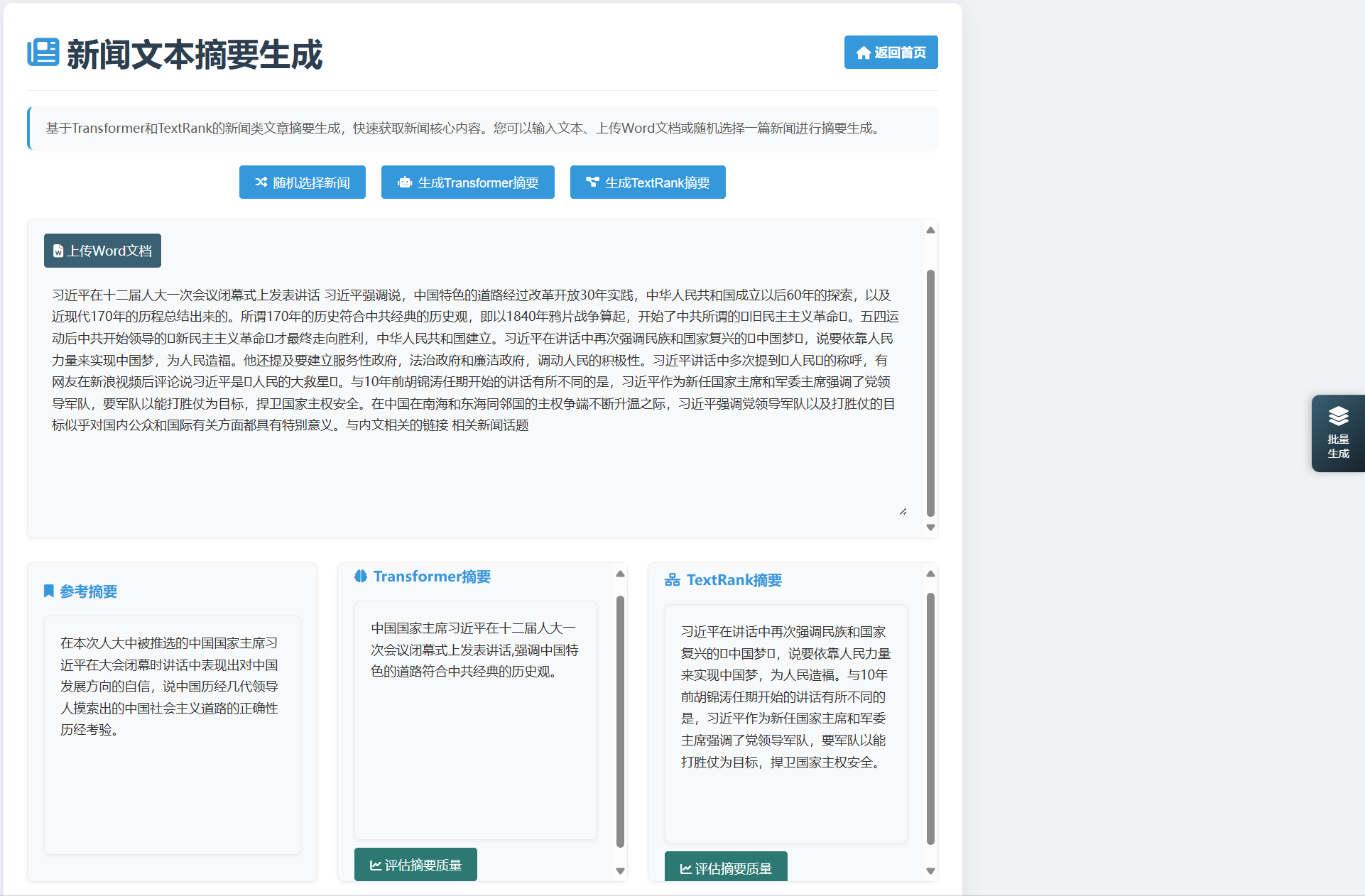


图6-3 新闻文本摘要生成界面

### 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，通过以下几方面的创新与实践，取得了显著成果：

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，为后续系统优化与迭代提供了平台。

## 7.2 未来研究展望

1、跨模态与多文档摘要

1）多模态融合：未来摘要系统可同时处理文本、图像、表格等多种模态信息，通过跨模态注意力或联合嵌入，将图表、代码段等重要非文本内容纳入摘要。

2）多文档摘要：对来自不同来源的多篇文档进行聚合与提炼，有助于生成全面的综合报告，需研究跨文档冲突检测与多视角整合策略。

2、事实一致性与鲁棒性

1）可解释性与可控性：引入对抗训练与不一致性损失，降低“幻觉”现象，确保生成摘要与原文事实高度一致。

2）鲁棒性评估与防御：针对噪声文本、对抗样本的稳健性测试，以及模型在不同噪声环境下的表现，是未来关键研究铺垫。

3、个人化与交互式摘要

1）用户定制化摘要：根据用户角色（如学生、研究员）或意图（概览、深入分析）的不同，动态调整摘要粒度和关注点。

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。

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。

# 致 谢

在即将要离开石大的时候，心中充满了留恋与感慨。。。

。。。

# 参考文献

[1] 严蔚敏,吴伟民.数据结构（C语言版）[M].北京:清华大学出版社,1997.4.

[2] 沈晴霓,聂青,苏京霞.现代程序设计—C++与数据结构面向对象的方法与实现[M].北京:北京理工大学出版社,2002.8.

[3] Thomas Connolly Carolyn Begg. Database systems[M].北京:电子工业出版社,2004.7.

[4] Roger Bate,Sandy Shrum. CMM Integration framework[J].CMU/SEI Spotlight 1998.9.

[5] J P Kuilboer,N Ashrafi. Software process and produt improvement[J].An empirical assessment,2000.4.

[6] 张美金 著.基于ASP技术的远程教育系统体系结构的研究.http://172.50.0.88:86 /~cddbn/Y517807/pdf/index.htm,2003-05-01.

[7] 王伟国,刘永萍,王生年,等.B/S模式网上考试系统分析与设计[J].石河子大学学报（自然科学版）,2003,6(2):145-147.

[8] …

[9] …

[10] …

# 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。