浙江大学计算机学院

Java 程序设计课程报告

2023 - 2024 学年 秋冬 学期

题目	Scholar Search Engine
学号	3210106360
学生姓名	杨沛山
所在专业	计算机科学与技术
所在班级	计科 2104

目　　录

1 引言

随着科技的快速发展和学术研究的不断深入，获取和处理大量学术文献成为学者和研究人员的一项重要任务。传统的文献检索方式往往耗时且效率低下，这促使了对更为高效、智能的学术搜索引擎的需求。本项目旨在开发一个学术搜索引擎，能够爬取、索引和检索学术文献。通过这个搜索引擎，用户可以便捷地查找相关论文，从而支持学术研究和文献回顾工作。这是一个综合性的项目，需要涉及到爬虫、数据库、检索算法、前端等多个方面的知识，可以让我们对 Java 语言的应用有一个更加深入的了解，提高我们的编程能力。

1.1 设计目的

本项目旨在开发一个高效的学术搜索引擎，能够自动爬取、索引和检索 arxiv 上的学术文献。通过这个搜索引擎，用户可以快速准确地找到所需的学术资源，大大提高学术研究的效率。具体而言，我们需要实现的功能如下：

1. 写一个 Web 爬虫，能够自动爬取 arxiv 上的学术文献的网站以及 PDF 文件。
2. 解析网页内容，提取出论文的标题、作者、摘要、关键词、引用等信息。
3. 解析 PDF 文件，提取出论文的正文内容。
4. 将提取到的信息建立索引
5. 通过命令行进行文件内容检索，并展示内容列表
    - 可通过作者、标题、摘要、会议来检索论文

1.2 设计说明

本程序采用 Java 程序设计语言，在 IntelliJ IDEA 开发环境下进行开发。具体程序由报告作者独立开发完成。

2 总体设计

2.1 功能模块设计

本程序需实现的主要功能有：

1. 用户可以通过本程序爬取 arxiv 上的最新论文来更新本地的索引。
2. 用户可以通过本程序对本地的索引进行检索，以查找所需的论文。
3. 用户在检索时可以通过作者、标题、摘要、会议等信息来检索论文。

本项目根据用户需求，主要包括以下几个模块：

1. 爬虫模块：负责爬取 arxiv 上的论文网页和 PDF 文件。
2. 解析模块：负责解析爬取到的网页和 PDF 文件，提取出论文的标题、作者、摘要、关键词、引用等信息。
3. 索引模块：负责将解析到的信息建立索引。
4. 检索模块：负责通过命令行进行文件内容检索，并展示内容列表。

程序的主要模块如图 1 所示。

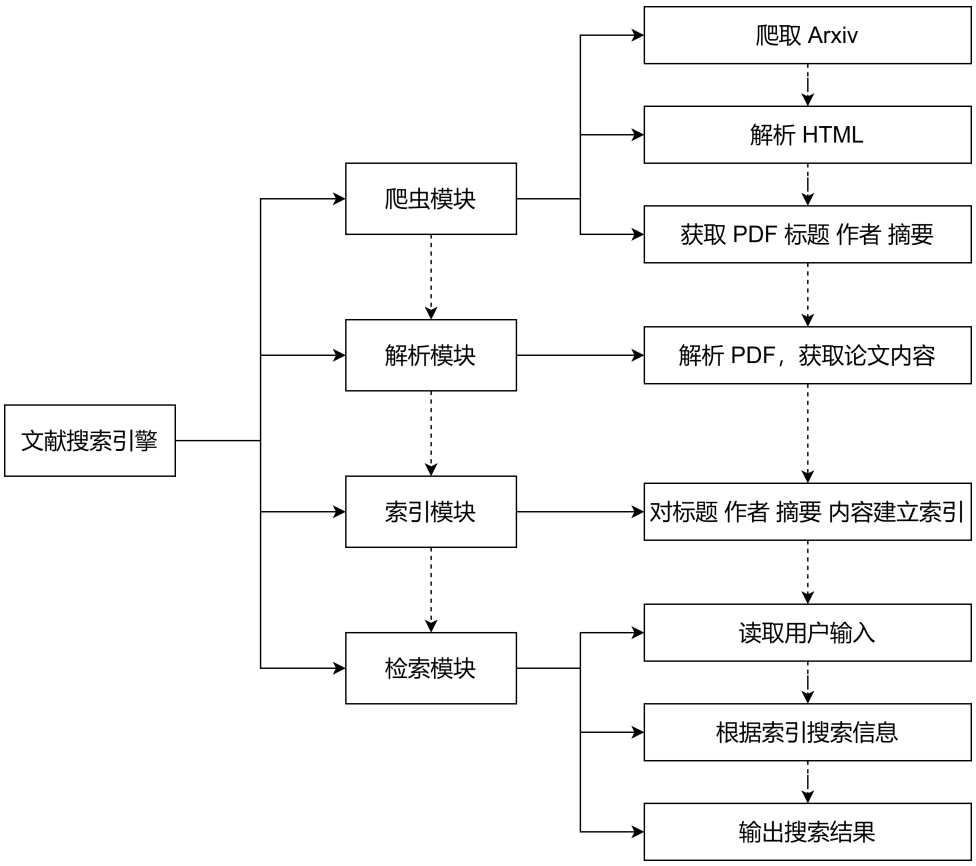

图 **1** 系统模块架构图

2.2 流程图设计

程序总体流程图如图 2 所示。

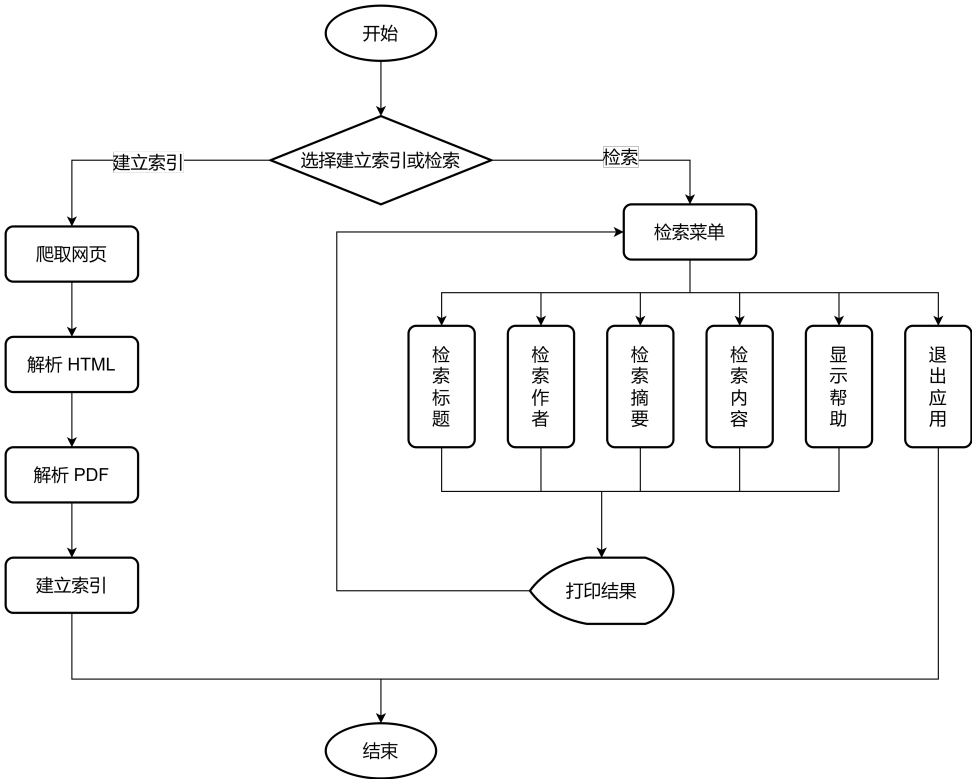
开始
选择建立索引或检索
建立索引
检索
爬取网页
解析 HTML
解析 PDF
建立索引
检索菜单
检索标题
检索作者
检索摘要
检索内容
显示帮助
退出应用
打印结果
结束

图 **2** 总体流程图

3 详细设计

3.1 爬虫模块设计

本项目的爬虫模块主要负责爬取 arxiv 上的论文网页和 PDF 文件。

本项目的网页爬虫基于 `crawler4j` 框架开发，用于爬取 arxiv 上的论文网页。爬虫配置 `CrawlerConfig` 设定了爬虫的一些基本参数，如爬取的网页数量、爬取的网页深度、爬虫的储存路径等。

爬虫类 `ArxivCrawler` 继承自 `WebCrawler` 类，重写了 `shouldVisit` 和 `visit` 方法。`shouldVisit` 方法用于判断当前网页是否应该被爬取，在该方法中，我们通过正则表达式，排除了与论文无关的网页，如 CSS 文件、图片文件等等，并通过字符串匹配前缀确保爬虫只爬取论文网页。

`visit` 方法用于处理爬取到的网页，使用 `Jsoup` 解析网页内容，提取出论文的标题、作者、摘要以及 PDF 文件的下载链接。爬取到的网页内容会被保存到 `PDFDoc` 类中，该类包含了上述提取出的信息。为了防止重复，我们使用 `HashSet` 来保存已经爬取过的网页链接，用于后续的 PDF 内容解析与索引建立。

爬虫模块流程如图 3 所示。

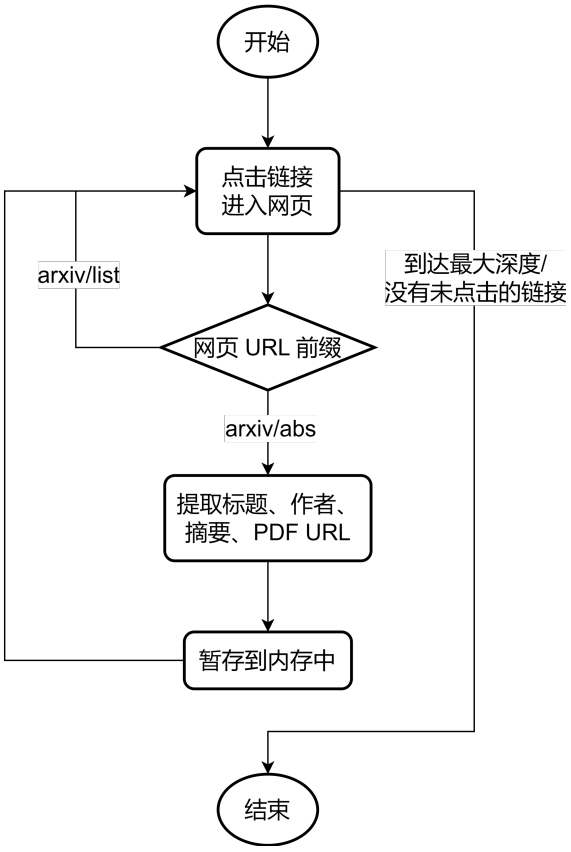

图 **3** 爬虫模块流程图

3.2 解析模块设计

PDF 文件解析模块主要负责解析爬取到的 PDF 文件，提取出论文的正文内容。

该模块使用了 Apache PDFBox 库来解析 PDF 文件。在 PDFDoc 类中，我们使用方法 parse 来下载并解析类内储存的 url 指向的 PDF 文件。解析过程包括使用 HTTP 连接下载 PDF 文件并将其加载到 PDDocument 对象中。然后，使用 PDFTextStripper 类提取 PDF 文档的文本内容。最后，将提取到的文本内容保存到 PDFDoc 类中的 text 字段中。

解析模块流程如图 4 所示。

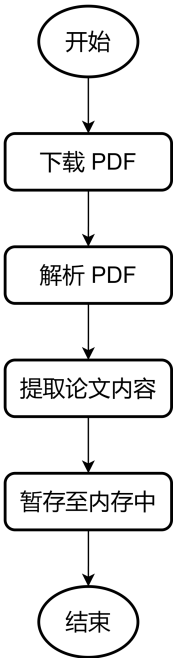

图 **4** 解析模块流程图

3.3 索引模块设计

索引模块主要负责将解析到的信息建立索引。

索引构建使用 Apache Lucene 库。PDFIndexer 类负责将解析的 PDF 文档转换为 Lucene 文档（ Document ），并将其添加到索引中。每个文档包括标题、作者、摘要、文本内容和 PDF url 等字段。在该类中有以下几个方法：

1. 构造函数：初始化索引目录和分词器和 IndexWriter。我们使用了 StandardAnalyzer 对文本进行分析，包括标记化、去除停用词等处理，以优化搜索的准确性和效率。
2. addDocument 方法：将解析的 PDF 文档添加到索引中，包括标题、作者、摘要、文本内容和 PDF url 五个字段。
3. close 方法：关闭 IndexWriter 对象，释放资源。

索引构建过程考虑了批量处理的需求，能够有效地处理大量文档，并确保索引构建的性能和稳定性。

索引模块流程如图 5 所示。

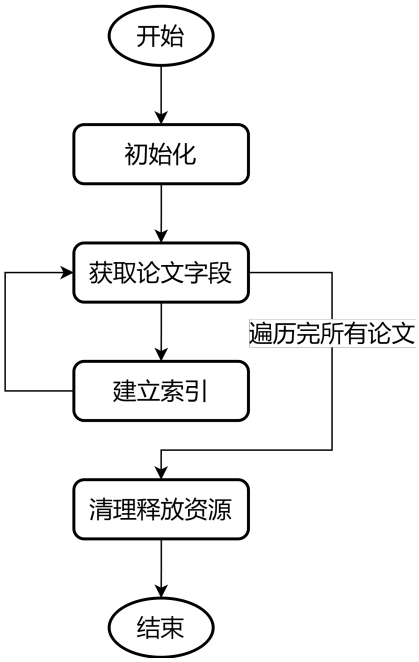

图 **5** 索引模块流程图

3.4 检索模块设计

检索模块主要负责通过命令行进行文件内容检索，并展示内容列表。

检索系统同样基于 Apache Lucene 库。PDFSearcher 类实现了对已构建索引的查询功能。使用 StandardAnalyzer 对用户的查询进行分析，并创建了 Lucene 查询（ Query ）。查询结果通过 TopDocs 和 ScoreDoc 对象返回。之后，再将查询结果高亮处理后打印到命令行中。

尽管是命令行程序，为了提高用户体验，使用了 HighLighter 类对查询结果进行高亮处理，标注查询关键词在文本中的位置，便于用户快速定位信息。同时，对每个匹配的文档显示了标题、作者和 PDF 链接，以及匹配的文本片段。

在 PDFSearcher 类中有以下几个方法：

1. 构造函数：初始化索引目录和分词器和 IndexSearcher。同样地，我们使用了 StandardAnalyzer 对文本进行分析。
2. searchText 方法：查询文本内容，将查询结果打印到命令行中。查询结果包括标题、作者、PDF 链接和匹配的文本片段。其中，匹配的文本片段使用 HighLighter 类进行高亮处理。
3. searchAuthor 方法：查询作者，将查询结果打印到命令行中。查询结果包括标题、作者、PDF 链接。同样地，作者中相匹配的字段会进行高亮处理。
4. searchTitle 方法：查询标题，将查询结果打印到命令行中。查询结果包括标题、作者、PDF 链接。同样地，标题中相匹配的字段会进行高亮处理。
5. searchAbstract 方法：查询摘要，将查询结果打印到命令行中。查询结果包括标题、作者、PDF 链接以及匹配的摘要中的文本片段。同样地，摘要中相匹配的字段会进行高亮处理。

检索模块流程如图 6 所示。

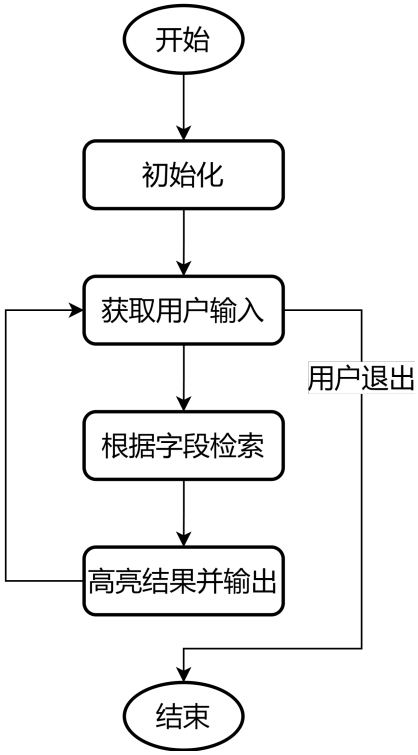

图 **6** 检索模块流程图

4 测试与运行

4.1 程序测试

在完成学术搜索引擎的核心代码开发后，进行了一系列的测试来确保程序的稳定性和功能性。首先对网络爬虫模块进行了测试，验证了其能够有效地爬取指定学术网站的文献页面和 PDF 文件。在测试中发现了一些小问题，例如若在 https://arxiv.org/list 网页尝试爬取摘要会导致解析错误，因此我们只能在 https://arxiv.org/abs 爬取需要的信息。经过调整和优化后，爬虫模块表现稳定。

PDF 解析和索引构建模块也经过了严格测试。测试中发现 PDF 解析时无法提取到作者标题，通过调整爬虫模块，从网页中获取论文标题、作者和摘要来解决该问题。索引构建模块能够高效地处理大量文档，并支持快速检索。

检索系统模块在用户界面和检索功能上都进行了综合测试。初期测试中发现了 Lucene Highlighter 默认不支持命令行的高亮。经过修改后，通过在高亮部分前后加上 ANSI 转义字符，最终实现了一个既直观又高效的用户检索体验。

总体来说，尽管在测试过程中遇到了一些挑战和问题，但通过团队的共同努力，大部分问题都得到了解决，使得整个系统在功能和性能上都达到了预期目标。但还有一些细节需要在未来的迭代中进一步完善和优化。

4.2 程序运行

由于索引和检索模块通常不一起运行，因此我们采用命令行参数来区分两个模块。在命令行中，用户可以通过 `index` 和 `search` 来分别运行索引和检索模块。

4.2.1 索引模块

使用 `java -jar ScholarSearchEngine.jar index` 命令来运行索引模块。在运行时，程序会提示用户输入爬取的网页数量和爬取的网页深度。程序会自动爬取 arxiv 上的论文网页和 PDF 文件，并将解析到的信息建立索引。索引文件会保存在 `data/index` 目录下。在创建索引过程中，程序会自动在控制台输出一些日志。在索引结束时，程序会提示用户索引的文档数量。

例如，在本次作业一起提交的索引后的文件使用的是 `java -jar ScholarSearchEngine.jar index` 命令生成的，最终打印的日志中说明了总共索引的文档数量为 956，如图 7 所示。

```
Title: High-Quality Facial Geometry and Appearance Capture at Home
Indexed: High-Quality Facial Geometry and Appearance Capture at Home
Total pdf indexed: 956

Process finished with exit code 0
```

图 **7** 索引模块运行示例

4.2.2 检索模块

使用 `java -jar ScholarSearchEngine.jar search` 命令来运行检索模块。在运行时，程序会提示用户输入检索的关键词。程序会自动在索引中检索关键词，并将检索结果打印到命令行中。

在打开程序后，程序会在控制台输出以下欢迎信息：

```
Welcome to Arxiv Searcher!
You can search by title, author, abstract or content.
Use following commands to search:
title [query]: search with default field `title`
author [query]: search with default field `author`
abstract [query]: search with default field `abstract`
text [query]: search with default field `text`
quit: quit
Type 'help' to show this help.

Tips:
1. You can use `title:`, `author:`, `abstract:` or `text:` to specify the field.
2. You can use `AND`, `OR` or `NOT` to combine queries.
3. You can use `(` and `)` to group queries.
Type 'tips' to show more tips.

Query:
```

图 **8**

用户可以通过输入 `help` 来查看帮助信息，如图 9 所示。

```
Query: help
You can search by title, author, abstract or content.
Use following commands to search:
title [query]: search with default field `title`
author [query]: search with default field `author`
abstract [query]: search with default field `abstract`
text [query]: search with default field `text`
tips: show query tips.
quit: quit
help: show this help.
```

图 **9** 帮助信息

用户可以通过输入 `author`、`title`、`abstract`、`text` 来指定检索的字段。例如，用户可以通过输入 `author Yann` 来检索作者中含有 `Yann` 的论文，如图 10 所示。

```
Query: author Yann
Found results in 4 documents.
Document 579
Title: Focal Length and Object Pose Estimation via Render and Compare
Author: Georgy Ponimatkin, Yann Labbé, Bryan Russell, Mathieu Aubry, Josef Sivic
URL: https://arxiv.org/pdf/2204.05145.pdf
Document 73
Title: Fine Dense Alignment of Image Bursts through Camera Pose and Depth Estimation
Author: Bruno Lecouat, Yann Dubois de Mont-Marin, Théo Bodrito, Julien Mairal, Jean Ponce
URL: https://arxiv.org/pdf/2312.05190v1.pdf
Document 902
Title: Fine Dense Alignment of Image Bursts through Camera Pose and Depth Estimation
Author: Bruno Lecouat, Yann Dubois de Mont-Marin, Théo Bodrito, Julien Mairal, Jean Ponce
URL: https://arxiv.org/pdf/2312.05190.pdf
Document 268
Title: FocalPose++: Focal Length and Object Pose Estimation via Render and Compare
Author: Martin Cífka, Georgy Ponimatkin, Yann Labbé, Bryan Russell, Mathieu Aubry, Vladimir Petrik, Josef Sivic
URL: https://arxiv.org/pdf/2312.02985.pdf
```

图 **10** 检索作者运行示例

用户可以通过输入 `title 4d` 来检索标题中含有 `4d` 的论文，如图 11 所示。

```
Query: title 4d
Found results in 3 documents.
Document 322
Title: Towards 4D Human Video Stylization
Author: Tiantian Wang, Xinxin Zuo, Fangzhou Mu, Jian Wang, Ming-Hsuan Yang
URL: https://arxiv.org/pdf/2312.04143.pdf
Document 142
Title: Gaussian-Flow: 4D Reconstruction with Dynamic 3D Gaussian Particle
Author: Youtian Lin, Zuozhuo Dai, Siyu Zhu, Yao Yao
URL: https://arxiv.org/pdf/2312.03431.pdf
Document 284
Title: SingingHead: A Large-scale 4D Dataset for Singing Head Animation
Author: Sijing Wu, Yunhao Li, Weitian Zhang, Jun Jia, Yucheng Zhu, Yichao Yan, Guangtao Zhai
URL: https://arxiv.org/pdf/2312.04369.pdf
```

图 **11** 检索标题运行示例

用户可以通过输入 `abstract nerf` 来检索摘要中含有 `nerf` 的论文，如图 12 所示。

```
Query: abstract nerf
Found results in 27 documents.
Document 18
Title: Prompt2NeRF-PIL: Fast NeRF Generation via Pretrained Implicit Latent
Author: Jianmeng Liu, Yuyao Zhang, Zeyuan Meng, Yu-Wing Tai, Chi-Keung Tang
URL: https://arxiv.org/pdf/2312.02568.pdf
Fragment 1
This paper explores promptable NeRF generation (e.g., text prompt or single image prompt) for direct conditioning and fast generation of NeRF parameters for the underlying 3D
 scenes, thus undoing complex intermediate steps while providing full 3D generation with conditional control. Unlike previous diffusion-CLIP-based pipelines that involve tedious
 per-prompt optimizations, Prompt2NeRF-PIL is capable of generating a variety of 3D objects with a single forward pass, leveraging a pre-trained implicit latent space of NeRF
 parameters. Furthermore, in zero-shot tasks, our experiments demonstrate that the NeRFs produced by our method serve as semantically informative initializations, significantly
 accelerating the inference process of existing prompt-to-NeRF methods. Specifically, we will show that our approach speeds up the text-to-NeRF model DreamFusion and the 3D
 reconstruction speed of the image-to-NeRF method Zero-1-to-3 by 3 to 5 times.
Document 883
Title: RePaint-NeRF: NeRF Editting via Semantic Masks and Diffusion Models
Author: Xingchen Zhou, Ying He, F. Richard Yu, Jianqiang Li, You Li
URL: https://arxiv.org/pdf/2306.05668.pdf
Fragment 1
The emergence of Neural Radiance Fields (NeRF) has promoted the development of synthesized high-fidelity views of the intricate real world. However, it is still a very demanding
 task to repaint the content in NeRF. In this paper, we propose a novel framework that can take RGB images as input and alter the 3D content in neural scenes. Our work leverages
 existing diffusion models to guide changes in the designated 3D content. Specifically, we semantically select the target object and a pre-trained diffusion model will guide the
 NeRF model to generate new 3D objects, which can improve the editability, diversity, and application range of NeRF. Experiment results show that our algorithm is effective for
 editing 3D objects in NeRF under different text prompts, including editing appearance, shape, and more. We validate our method on both real-world datasets and synthetic-world
 datasets for these editing tasks. Please visit this https URL for a better view of our results.
Document 895
Title: Re-Nerfing: Enforcing Geometric Constraints on Neural Radiance Fields through Novel Views Synthesis
```

图 **12** 检索摘要运行示例

用户可以通过输入 `text "3d gaussian"` 来检索正文中含有 `3d gaussian` 的论文，如图 13 所示。

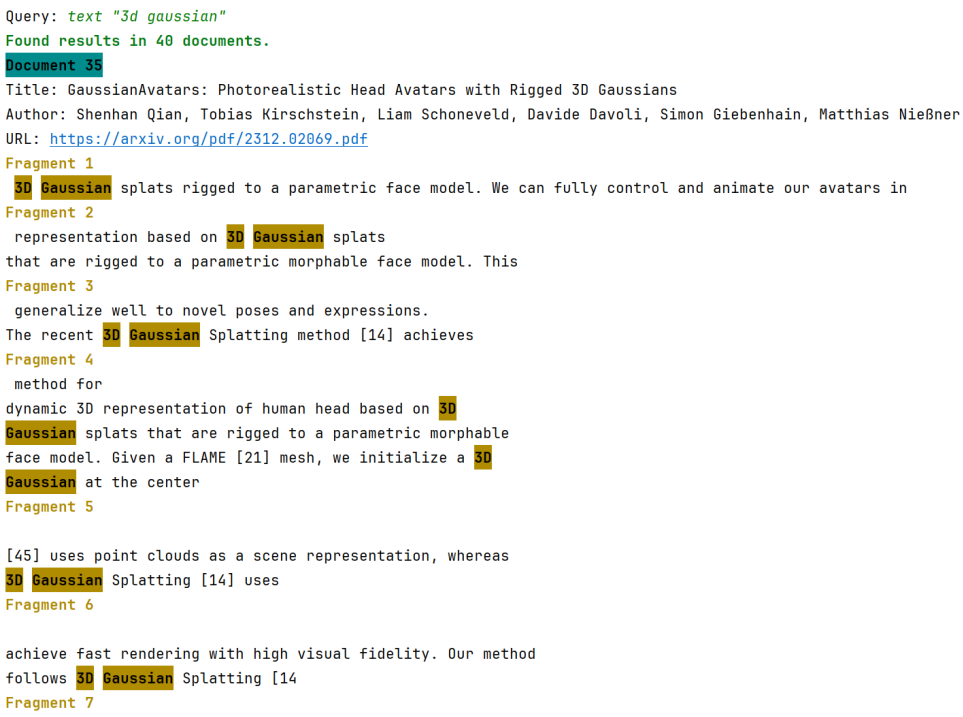

图 **13** 检索正文运行示例

除此之外，Apache Lucene 还支持更多的检索功能，例如模糊检索、通配符检索、范围检索等等[1]，我们可以利用这些功能达到一些更加复杂的检索需求，例如想要检索标题中含有 editable 或 3d gaussian 的 Guosheng Lin 的论文，如图 14 所示。

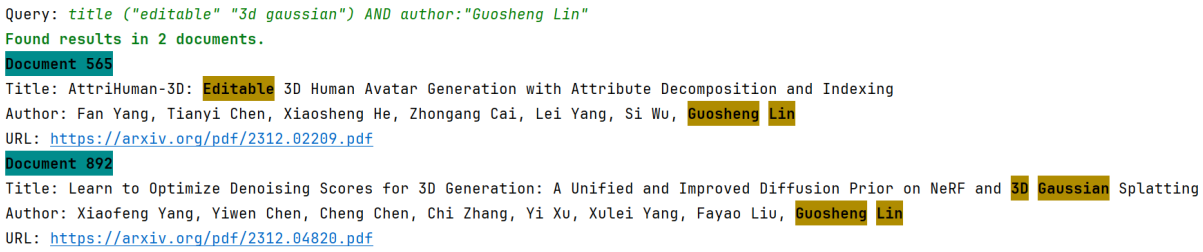

图 **14** 检索标题复杂运行示例

由于 Apache Lucene 的检索功能十分强大，我们可以通过不同的检索方式来满足不同的检索需求。而用户输入的第一个提示词只决定了默认的检索字段，用户可以通过输入其他提示词来指定检索字段，从而实现更加精准的检索。通过输入 `tips`，用户可以查看 Apache Lucene 支持的部分检索规则，如图 15 所示。

[1]https://lucene.apache.org/core/9_9_0/queryparser/org/apache/lucene/queryparser/classic/package-summary.html

```
Query: tips
Tips:
1.  You can use `title:`, `author:`, `abstract:` or `text:` to specify the field.
2.  You can use `AND`, `OR` or `NOT` to combine queries.
3.  You can use `(` and `)` to group queries.
4.  You can use `"` to search for an exact phrase.
5.  You can use `*` to search for a wildcard.
6.  You can use `~` to search for a fuzzy query.
7.  You can use `^` to boost a term.
8.  You can use `[]` to search for a range.
9.  You can use `{}` to search for a set.
10. You can use `\` to escape special characters.
11. You can use `+` to require a term.
12. You can use `-` to exclude a term.
13. You can use `?` to match a single character.
14. You can use `|` to search for a term in multiple fields.
15. You can use `&&` to require a term in multiple fields.
16. You can use `||` to search for a term in multiple fields.
See https://lucene.apache.org/core/9_8_0/queryparser/org/apache/lucene/queryparser/classic/package
 -summary.html#package.description for more details.
```

图 **15** 检索提示词运行示例

5 总结

这个学术搜索引擎的主要实现都是通过调用相关库来实现，因此难度较小，但也考验了我们使用他人开发的库的能力。在编程过程中，由于库的版本等原因，遇到了许多磕绊。遇到这样的问题，我们需要仔细检查库的官方文档，找到每个方法对应的参数、作用和返回值，经过自己的不断调试，这个学术搜索引擎终于大功告成。

通过这个学术搜索引擎项目的开发，我深刻理解了网络爬虫、PDF 解析、索引构建和用户检索系统的设计和实现。项目的开发过程中遇到了不少挑战，但这些挑战最终转化为了宝贵的经验。我认识到，注重细节的重要性，细小的错误可能会导致整个系统的不稳定。同时，这个项目也强化了我们作为程序员的严谨性和解决问题的能力。

这个项目不仅仅是对我们编程技能的锻炼，更是一次理论与实践结合的深刻体验。它让我们从实践中更好地理解了计算机科学的基本原理，同时也让我们意识到了自己的不足之处，比如在算法的应用和系统设计方面还有提升的空间。

在 Java 课程设计中，这个项目是一个宝贵的学习机会，不仅提高了我们的技术能力，也增强了我们团队合作和问题解决的能力。我们相信，通过这次经验的积累，我们将能够在未来的学习和工作中做得更好。