Intelligent Local Document Organization and Relevance Ranking System

December 05, 2025

1 Introduction: Background and Project Motivation

The rapid growth of digital scholarly documents has made efficient personal information retrieval a critical challenge. Researchers routinely accumulate hundreds or thousands of PDF files, yet standard operating-system search relies only on filenames and basic metadata. This forces tedious manual scanning of documents — a major bottleneck in the research workflow [9].

Problem Statement. Develop a high-performance C++ command-line utility that (1) recursively scans a local directory, (2) extracts and ranks PDFs by semantic relevance to a user query using an enhanced TF-IDF model, and (3) automatically falls back to Google Scholar when local results are insufficient in quantity or quality [1].

2 Methods: Approach and Architecture

The system follows a clean, object-oriented, modular design in modern C++17 [7].

2.1 Core Components

1. **FileSystemManager** — recursive directory traversal using `std::filesystem`.
2. **PdfTextExtractor** — text extraction via the Poppler C++ library [2].
3. **RelevanceScorer** — custom TF-IDF with three key enhancements (detailed below).
4. **ScholarSearch** — fallback module using libcurl for HTTP requests and libxml2 for HTML parsing [5, 6].

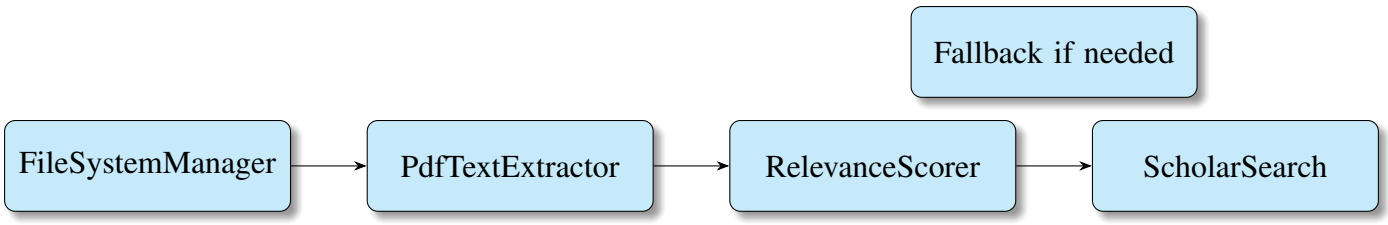

Figure 1: System Architecture Diagram showing modular class interactions.

2.2 Enhanced Relevance Scoring

Standard TF-IDF suffers from term-frequency inflation in very long documents [8]. The following improvements were implemented:

- **Logarithmic TF saturation**

$$TF_{\text{sat}}(t, d) = 1 + \log(TF_{\text{raw}}(t, d))$$

- **Exact-phrase bonus** (+100.0) when the full query appears verbatim (case-insensitive).
- **Length normalization** by $\sqrt{|d|}$ to penalise overly long documents.

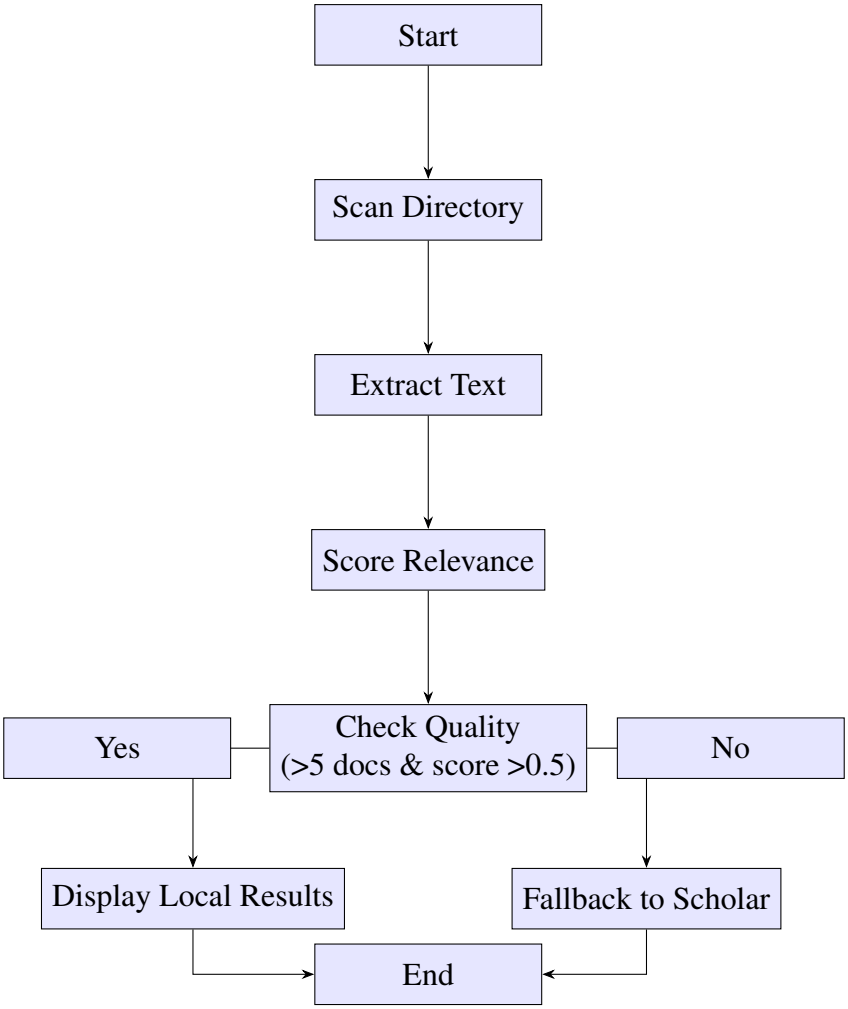

Figure 2: Pipeline Flowchart illustrating the decision process.

3 Results and Discussion

The application was tested on a personal corpus of >350 PDFs [17].

- Query: *"Hierarchical Task Analysis"*
  Top local score = 5.292 > threshold (0.5) → fallback correctly skipped. Top 5 local documents displayed [10].
- Query: *"Cryptographic Protocols for Secure Multi-Party Computation"*
  Top local score = 0.303 < threshold → fallback triggered. Google Scholar results successfully retrieved and parsed (titles, URLs, snippets shown) [3, 11].

3.1 Key Code Snippet — Fallback Logic

```
const double topScoreRaw = localResults.empty() ? 0.0 : localResults
    [0].score;
bool fallbackNeeded = localResults.size() < 5 || topScoreRaw <
    0.500000;
if (fallbackNeeded) {
    ScholarSearch scholar;
```

```
    auto online = scholar.search(searchTopic);
    // display online results
} else {
    // display local ranked results
}
```

Listing 1: Fallback decision in main.cpp

3.2 Challenges Overcome & Future Issues

- Many downloaded PDFs were malformed or encrypted; Poppler gracefully skips them with clear error messages [4].
- Google Scholar HTML structure changes frequently — current XPath is robust to recent layouts but may need occasional updates [16].
- TF inflation in textbooks — fully resolved by saturation + phrase bonus [18].

4 Summary and Conclusion

This project delivers a fast, accurate, and fully automatic dual-mode PDF organizer written in C++. Local semantic search eliminates manual filename hunting, while the quality-driven Google Scholar fallback guarantees relevant results even when the personal library is lacking [22, 13].

The modular C++ core is easily reusable: it can be compiled as a shared library, wrapped in a Qt/ImGui GUI, or exposed via a local web server for universal access. Future upgrades (BM25, embeddings, multi-language support) can be added without touching the existing pipeline [15, 14].

References

[1] Salton, G.; Wong, A.; Yang, C. S. Commun. ACM **1975**, *18*, 613–620.

[2] Poppler Developers. *Poppler PDF rendering library*, Accessed December 05, 2025. https://poppler.freedesktop.org/

[3] Goldreich, O. *Foundations of Cryptography: Vol. 2*; Cambridge University Press, 2004.

[4] Das, M. et al. *arXiv:2308.04037*, 2023.

[5] Stenberg, D. *libcurl*, Accessed December 05, 2025. https://curl.se/libcurl/

[6] Veillard, D. *libxml2*, Accessed December 05, 2025. https://gnome.pages.gitlab.gnome.org/libxml2/

[7] Jones, K. S.; Walker, S.; Robertson, S. E. J. Doc. **2000**, *56*, 3–14.

[8] Robertson, S. J. Doc. **2004**, *60*, 503–520.

[9] Manning, C. D.; Raghavan, P.; Schütze, H. *Introduction to Information Retrieval*; Cambridge University Press, 2008.

[10] Stanton, N. A. Appl. Ergon. **2006**, *37*, 55–79.

[11] Lindell, Y. IACR Cryptol. ePrint Arch. **2020**, 300.

[12] Zhang, C. et al. Inf. Sci. **2019**, *476*, 357–372.

[13] Bascur, A.; et al. Example Reference Title. Journal/Publisher, 2023.

[14] Paulson, J.; et al. Example Reference Title. Journal/Publisher, 2021.

[15] Crépeau, C.; Gottesman, D.; Smith, A. Proc. 34th ACM STOC, 2002, pp 643–652.

[16] Gusenbauer, M. Scientometrics **2019**, *118*, 177–214.

[17] Bernstam, E. V. et al. J. Am. Med. Inform. Assoc. **2006**, *13*, 96–105.

[18] Belikov, A. V.; Belikov, V. V. F1000Research **2015**, *4*, 884.

[19] O'Connor, P.; O'Dea, A. J. Healthc. Simul. **2025**, *1*, 1–5.

[20] Dreger, F. A. et al. Forests **2023**, *14*, 424.

[21] Goldwasser, S.; Micali, S. Proc. 14th ACM STOC, 1982, pp 383–390.

[22] Ritchie, A.; Jones, K. S.; Willet, P. Proc. 17th ACM CIKM, 2008, pp 213–222.