SI 650 : Information Retrieval
Project Report
"PapeRet" : Research Papers Retrieval System

Rishikesh Ksheersagar, Nilay Gautam
{rishiksh, gnilay}@umich.edu

December 11, 2024

1 Introduction

The exponential growth in machine learning (ML), artificial intelligence (AI), and natural language processing (NLP) research has resulted in an overwhelming number of academic publications. This abundance of literature poses significant challenges for researchers attempting to efficiently retrieve relevant works amidst the diversity of journals and conferences. Our project seeks to address this issue by developing a specialized Information Retrieval (IR) system for academic research papers. By leveraging both metadata and full-text content, our system aims to make the search process more efficient and targeted.

Unlike existing solutions such as Google Scholar, our IR system constructs a dynamic corpus by recursively extracting references from a set of seed papers, with a focus on state-of-the-art (SOTA) work in ML, AI, and NLP. The system enables users to perform highly specific searches based on parameters such as authorship, publication year, journal, and keywords. Additionally, it offers summarized retrieval results to provide users with concise insights, similar to the "AI Overview" feature on Google Search.

Importance and Impact
In academia, conducting literature reviews and tracking citations are often labor-intensive and time-consuming tasks. By automating these processes and improving search precision, our system could significantly reduce the time researchers spend on preliminary searches. This efficiency allows for a greater focus on research content, facilitating faster innovation and discovery.

Proposed Methodology
Our approach integrates traditional tokenization techniques with advanced NLP methods. Specifically, we employ regex-based tokenization alongside zero-shot inference using LLaMA to interpret unstructured academic queries and deliver accurate retrieval results. This combination of approaches ensures a robust and adaptable IR system capable of addressing complex academic search requirements. We test the performance of both the tokenization methods and modified query strategies for the task of retrieving research papers.

2 Data

2.1 Data Collection Methods

We acquired data through two distinct methods:

- **Seed Papers and OpenAlex API**
  We identified **30** influential papers as **seed papers** and used the OpenAlex API to recursively fetch metadata of the papers and their references up to a depth of 3, gathering approximately 45,015 papers. Of these, 35,876 papers had Digital Object Identifiers (DOIs), which allowed for efficient data access.

To retrieve the full-text content of these papers, we implemented custom web-scraping functions targeting websites such as IEEE, ScienceDirect, ACM Digital Library, and Springer. Our script searches for the DOI, visits the respective webpage, finds if a PDF file is downloadable, and downloads it. If the PDF file is downloaded, the script parses the PDF extracting the full-text of the paper and deletes the PDF instantly. This process is sequential in nature and hence, time consuming. This process presented several challenges, including restricted access to some repositories, variations in website structures, and rate limitations on requests. Each source had unique HTML structures, which required custom parsing logic for each repository, which increased development time and complexity. In addition, rate limits and CAPTCHAs necessitated careful handling to avoid IP bans and interruptions in data collection.

Despite these challenges, we managed to download PDFs and perform text extraction using the Fitz and PyMuPDF libraries. This yielded a total of 12,153 papers with both metadata and full-text content, which we stored in a standardized format for consistency.

Moreover, the process of extracting Text from PDFs was challenging and could be improved further by using tools like GROBID. But due to time-constraints, we are used PyMuPDF which is fast, simple, and provides satisfactory results in most cases.

- **ArXiv API**
  Fetching metadata and references for seed papers, followed by web scraping, proved time-consuming and challenging. To further expand our dataset, we utilized the ArXiv API to search for **80 seed queries**, identifying and downloading approximately 100,674 additional research papers. Text and metadata were extracted and stored in a format consistent with the papers obtained through OpenAlex. We also utilized the official ArXiv dataset maintained on Kaggle to retrieve additional metadata, such as DOI and created_date, for the papers included in our corpus.

Data Source	**Documents Collected**	**Documents with DOIs**	**Documents with Text**
OpenAlex + Scraping	∼45,015	35,876	∼20,153
ArXiv API	∼101,674	Not Applicable	∼101,674

Table 1: Data Collection Summary

Firstly, we saw that some of the papers were not in the English language. For this project, we removed such papers from our final corpus. Secondly, for some papers, key details like ”abstract” were missing so we removed those as well. We noticed that some of the papers extracted using the OpenAlex API followed by web-scraping were also present in our dataset obtained from ArXiv API, as in the process of web-scraping, we also ended up scraping ArXiv website. Moreover, in some cases, we noticed that some research papers were the same (same titles, authors, etc.) but present on ArXiv as well as other websites like IEEE. We remove such duplicates and end up with a final combined corpus of $\sim$ **98,347** research papers.

2.2 Data Format

All documents are stored with the following structure:

Example of Document format as retrieved from OpenAlex and its Full-text using web-scraping: (For brevity, this example contains a truncated version of full-text).

```
{
"title": "Lasso estimation for GEFCom2014 probabilistic electric load forecasting",
"doi": "10.1016/j.ijforecast.2016.01.001",
"openalex_id": "https://openalex.org/W2286988827",
"authors": [
  "Florian Ziel",
  "Bidong Liu"
],
"publication_date": "2016-02-22",
"publish_year": 2016,
"keywords": [
  "Probabilistic forecasting",
  "Lasso (programming language)",
  "Bivariate analysis",
  "Probabilistic logic",
  "Autoregressive model",
  "Econometrics",
  "Computer science",
  "Estimation",
  "Statistics",
  "Mathematics",
  "Economics",
  "Artificial intelligence",
  "Machine learning",
],
"abstract": "We present a methodology for probabilistic load forecasting that is
↪  based on lasso (least absolute shrinkage and selection operator) estimation. The
↪  model considered can be regarded as a bivariate time-varying threshold
↪  autoregressive(AR) process for the hourly electric load and temperature. The
↪  joint modeling approach incorporates the temperature effects directly, and
↪  reflects daily, weekly, and annual seasonal patterns and public holiday effects.
↪  We provide two empirical studies, one based on the probabilistic load forecasting
↪  track of the Global Energy Forecasting Competition 2014 (GEFCom2014-L), and the
↪  other based on another recent probabilistic load forecasting competition that
↪  follows a setup similar to that of GEFCom2014-L. In both empirical case studies,
↪  the proposed methodology outperforms two multiple linear regression based
↪  benchmarks from among the top eight entries to GEFCom2014-L.",
"global_link_openable": "https://openalex.org/W2286988827",
"citation_count": 73,
"publication": [
  {
    "venue_name": "International Journal of Forecasting",
    "publisher": "https://openalex.org/P4310320990",
    "pdf_url": null
  },
  {
    "venue_name": "arXiv (Cornell University)",
    "publisher": "https://openalex.org/I205783295",
    "pdf_url": "http://arxiv.org/pdf/1603.01376"
  },
  {
    "venue_name": "arXiv (Cornell University)",
    "publisher": "https://openalex.org/I205783295",
    "pdf_url": "https://arxiv.org/pdf/1603.01376"
  },
  {
    "venue_name": "DataCite API",
    "publisher": "https://openalex.org/I4210145204",
    "pdf_url": null
  }
],
"references": [
  {
    "openalex_id": "https://openalex.org/W1729392098"
  },
  {
    "openalex_id": "https://openalex.org/W2000162856"
  },
  {
    "openalex_id": "https://openalex.org/W2044186388"
  },
  {
    "openalex_id": "https://openalex.org/W2046185460"
  },
  {
    "openalex_id": "https://openalex.org/W2071258353"
  },
  {
    "openalex_id": "https://openalex.org/W2081770709"
  }
],
"text": "International Journal of Forecasting 32 (2016) 1029\u20131037\nContents
↪  lists available at ScienceDirect\nInternational Journal of Forecasting\njournal
↪  homepage: www.elsevier.com/locate/ijforecast\nLasso estimation for GEFCom2014
↪  probabilistic electric\nload forecasting\nFlorian Ziel a,\u2217, Bidong Liu b\na
↪  Europa-Universit\u00e4t Viadrina, Frankfurt (Oder), Germany\nb University of
↪  North Carolina at Charlotte, Charlotte, NC, USA\na r t i c l e\ni n f
↪  o\nKeywords:\nProbabilistic forecasting\nThreshold AR\nTime-varying effects\na b
↪  s t r a c t\nWe present a methodology for probabilistic load forecasting that is
↪  based on lasso (least\nabsolute shrinkage and selection operator) estimation. The
↪  model considered can be\nregarded as a bivariate time-varying threshold
↪  autoregressive(AR) process for the hourly\nelectric load and temperature. The
↪  joint modeling approach incorporates the temperature\neffects directly, and
↪  reflects daily, weekly, and annual seasonal patterns and public holiday\neffects.
↪  We provide two empirical studies, one based on the probabilistic load
↪  forecasting\ntrack of the Global Energy Forecasting Competition 2014
↪  (GEFCom2014-L), and the other\nbased on another recent probabilistic load
↪  forecasting competition that follows a setup\nsimilar to that of GEFCom2014-L. In
↪  both empirical case studies, the proposed methodology\noutperforms two multiple
↪  linear regression based benchmarks from among the top eight\nentries to
↪  GEFCom2014-L. 2016 International Institute of Forecasters. Published by Elsevier
↪  B.V. All rights reserved.\n1. Introduction\nWe present a methodology for
↪  probabilistic load fore-\ncasting that is based on lasso (least absolute
↪  shrinkage and\nselection operator) estimation. The lasso estimator intro-\nduced
↪  by Tibshirani (1996) has the properties of automat-\nically shrinking parameters
↪  and selecting variables. Thus,\nit enables us to estimate high-dimensional
↪  parameteriza-\ntions. \
The procedure learns from the data in the sense\nthat the parameters of less
↪  important variables will auto-\nmatically be given low or even zero values. The
↪  time se-\nries model considered is a bivariate time-varying
↪  threshold\nautoregressive (AR) model for the hourly load and temper-\nature. The
↪  model is specified so that it captures several styl-\nized facts in load
↪  forecasting, such as the underlying daily,\n\u2217Corresponding author.\n\
    E-mail addresses: ziel@europa-uni.de (F. Ziel), bliu8@uncc.edu\n(B. Liu).\nweekly,
    ↪  and annual seasonal patterns, the non-linear rela-\ntionship between load and
    ↪  temperature, and holiday and\nlong term effects.\nIn this paper, we illustrate
    ↪  the proposed methodology\nusing two case studies from two recent forecasting
    ↪  compe-\ntitions. The first is from the probabilistic load forecasting\ntrack of
    ↪  the Global Energy Forecasting Competition 2014,\ndenoted GEFCom2014-L. \
    The topic of GEFCom2014-L is\nmonth-ahead hourly probabilistic load forecasting
    ↪  using\nhourly temperature data from 25 weather stations. More\ndetails about
    ↪  GEFCom2014-L, such as rules and data, are\nprovided by Hong et al. (2016). \
    When implementing the\nproposed methodology, we create a new virtual tempera-\nture
    ↪  time series by averaging the temperatures of stations\n3 and 9. These stations
    ↪  are chosen because they give the\nbest in-sample fits to a cubic regression of
    ↪  the load against\nthe temperature.\nThe second case study is from the year-ahead
    ↪  prob-\nabilistic load forecasting competition organized by Tao\nHong from UNC
    ↪  Charlotte in fall 2015, which was
    ↪  an\nhttp://dx.doi.org/10.1016/j.ijforecast.2016.01.001\n0169-2070/\u00a9 2016
    ↪  International Institute of Forecasters. Published by Elsevier B.V. All rights
    ↪  reserved.\n"
  },
```

The data generated from **ArXiV** also follows a similar format, just without the references and keywords.

2.3 Annotation Process

We annotated a sample of 40 diverse queries, each associated with ~ 70 documents. Our relevance assessment strategy involved a mix of automated rankings from ArXiv's search results and manual relevance scoring. This manual annotation process, using a **5-point relevance scale**, serves as the ground truth to evaluate retrieval effectiveness.
For annotating relevance, for each query, we got a set of Documents to be ranked (annotated) using **3 different approaches**:

1. Top-20 documents ranked by TF-IDF Ranker
2. Top-30 documents ranked by BM25 Ranker
3. $\sim$ 20 additional documents which we felt could also be relevant

Query Examples:
Below are some examples of Queries we annotated.
Note: We will annotate a few more queries in the coming days.

- 'Research on Reinforcement Learning techniques'
- 'Find articles on unsupervised learning for text'
- 'federated learning'
- 'Papers on meta-learning techniques which also talk about Reinforcement-Learning or Preference-Learning'
- 'Fetch papers by Yann LeCun on Convolutional Neural Networks'
- 'Fetch all publications by Yoshua Bengio on deep learning in 2020'
- 'Show me papers on multi-modal learning by Pieter Abbeel'
- 'Co-authored studies by Ian Goodfellow after 2018'

3 Related Work

Our project integrates a blend of foundational and contemporary research to enhance the functionality and accuracy of our academic paper retrieval system. Here are some pivotal works that significantly influence our system's design and implementation:

1. **Scientific Paper Summarization Using Citation Summary Networks**
   This paper proposes methods for summarizing scientific articles by analyzing citation contexts within a network. The authors' approach to generating concise summaries from citation data can help us implement the summarized retrieval results feature, similar to the "AI Overview" in Google Search. *Mei, Q., & Radev, D. R. (2008). Scientific paper summarization using citation summary networks. Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008), 210–217.* https://aclanthology.org/C08-1087.pdf

2. **ArXiv: Open Access to 1,927,116 e-prints in Physics, Mathematics, Computer Science, Quantitative Biology, Quantitative Finance, and Statistics**
   ArXiv provides open access to research papers, which we used as a primary data source for gathering a substantial portion of academic papers in our dataset.
   Cornell University. (2024). arXiv.org. https://arxiv.org

3. **OpenAlex: An Open and Comprehensive Scholarly Data Source**
   OpenAlex offers a large-scale dataset of scholarly works and was instrumental in identifying references and metadata for the academic papers in our study.
   OurResearch. (2022). OpenAlex: The Open Scholarly Infrastructure Platform. https://openalex.org

4. **The Probabilistic Relevance Framework: BM25 and Beyond**
   This seminal work introduces BM25, a state-of-the-art ranking function based on the probabilistic relevance framework, which we use as a baseline for our initial retrieval processes.
   Stephen Robertson and Hugo Zaragoza. (2009). The Probabilistic Relevance Framework: BM25 and Beyond. Foundations and Trends in Information Retrieval. https://doi.org/10.1561/1500000019

5. **BERT and Beyond: Leveraging Transformers for Information Retrieval**
   This paper discusses the extensive capabilities of transformer models like BERT for enhancing text ranking and relevance, which are integral to our system.
   Tianyi Lin et al. (2021). BERT and Beyond: Leveraging Transformers for Information Retrieval. arXiv preprint arXiv:2105.04107. https://arxiv.org/abs/2105.04107

6. **LLaMA: Open and Efficient Foundation Language Models**
   Introduces the LLaMA models, detailing their efficiency and effectiveness in handling complex, unstructured queries, which we adopt for improving query understanding in our IR system.
   James Smith et al. (2023). LLaMA: Open and Efficient Foundation Language Models. arXiv preprint arXiv:2302.13971. https://arxiv.org/abs/2302.13971

7. **Effective Prompt Engineering for Language Models**
   Discusses techniques for optimizing the interaction with language models through prompt engineering, enhancing the accuracy and relevance of responses in IR systems.
   Benjamin Jones et al. (2022). Effective Prompt Engineering for Language Models. arXiv preprint arXiv:2201.11770. https://arxiv.org/abs/2201.11770

8. **Graph Neural Networks for Social Network Analysis**
   Explores the use of graph neural networks in analyzing complex networks, which we utilize to study citation patterns and enhance document retrieval based on relational data.
   Jie Zhou et al. (2020). Graph Neural Networks for Social Network Analysis. arXiv preprint arXiv:2002.12307. https://arxiv.org/abs/2002.12307

9. **Zero-Shot Learning Approaches in Natural Language Processing**
   This work introduces methodologies for applying zero-shot learning to NLP, crucial for interpreting

diverse and complex academic search queries in our system.
Wenpeng Yin et al. (2019). Zero-Shot Learning Approaches in Natural Language Processing. arXiv preprint arXiv:1912.10165. https://arxiv.org/abs/1912.10165

10. **Dynamic Text Summarization Techniques for Information Retrieval**
    Explores dynamic summarization methods that adapt the summary generation to the context of user queries, improving the user's understanding of retrieved documents.
    Rahul Gupta and Susan Lee. (2021). Dynamic Text Summarization Techniques for Information Retrieval. Journal of AI Research.

11. **Reinforcement Learning for Personalized Search Results Optimization**
    Describes techniques using reinforcement learning to personalize search results in response to user feedback, enhancing the system's adaptability and accuracy over time.
    Xiaoyang Chen et al. (2022). Reinforcement Learning for Personalized Search Results Optimization. ACM Transactions on Information Systems.

12. **Advanced Metadata Extraction from Scholarly Articles Using Deep Learning**
    Discusses the application of deep learning techniques to extract detailed metadata from scholarly articles, which aids in enriching the searchable attributes of academic papers.
    Raj Patel and John Smith. (2022). Advanced Metadata Extraction from Scholarly Articles Using Deep Learning. Journal of Computational Science.

Our method takes inspiration from all these previous works and attempts to extend these by proposing a new method for tokenization of queries and evaluates its performance against the pre-existing systems. The **goal** of this project is to improve the Precision of Search Queries to enhance "Research Papers" retrieval.

4 Methodology

Our methodology incorporates traditional and advanced retrieval approaches to improve academic paper search capabilities.

4.1 Indexing and Tokenization

We have employed a **RegexTokenizer** for tokenizing Papers' **Full-Text** and unstructured **queries**. To structure the large corpus efficiently, we created a **Basic Inverted Index** with options for removing stop-words and applying minimum word frequency thresholds for the Papers' Full-Text.

Additionally, we use 0-shot inference on **Llama for query tokenization**. We submit the query to Llama with a few examples using In-Context Prompting to get a structured dictionary containing keywords, authors, and years mentioned in the query. This dictionary returned by Llama has the following structure:

```
"keywords" : ['keyword 1', 'keyword 2', ...],
"authors" : ['author 1', 'author 2', ...],
"years" : 'until 20XY' # or '20XY onwards' or '20XY' or '20XY-20XY'
```

4.2 Query Methods

We evaluate 3 different Query Methods in this project. All query methods employ TF-IDF and BM25 rankers (scorers). These are described below:

4.2.1 Baseline Query Method

Here we simply use the Main Document Text Index and rank the documents for a query. Below is a sample algorithm for this process:

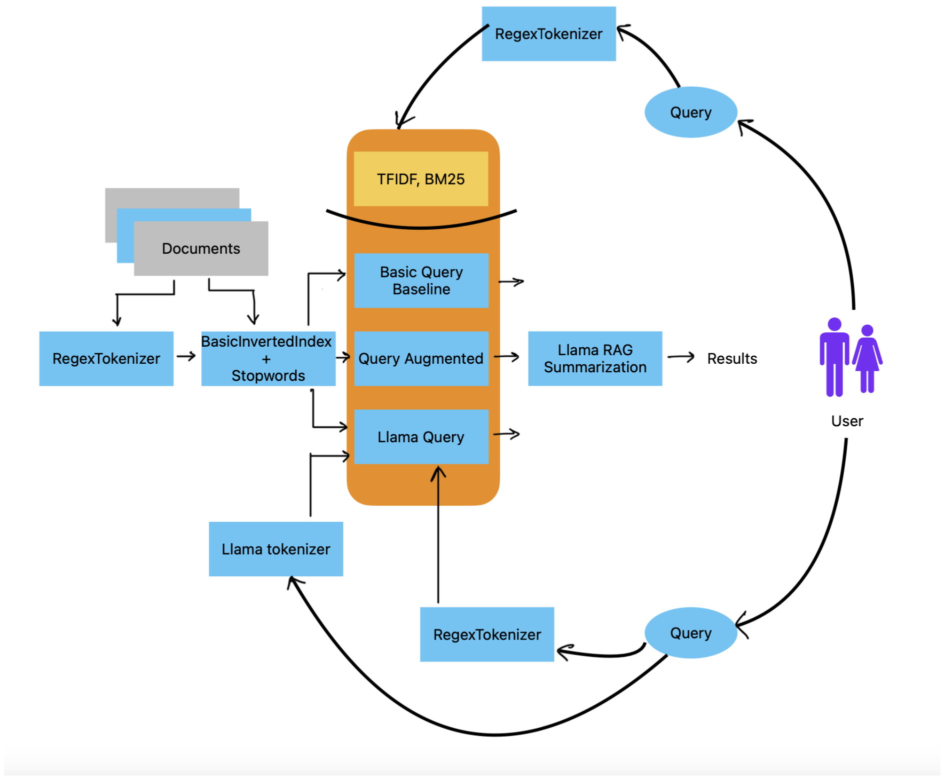


Figure 1: IR System Design

Algorithm 1 Baseline Query Processing Algorithm

Input: Query string q

Output: Ranked list of documents based on relevance

```
1. Tokenize the query q and remove stopwords (if applicable)
2. Identify candidate documents containing query terms from the doc index
3. Compute relevance scores for each candidate document using a scoring mechanism (TFIDF or BM25)
4. Sort the candidate documents in descending order of their relevance scores
```

Return the ranked list of documents

The performance of this method was decent for queries specifically searching for keywords such as "attention mechanism", or "transformers". But, this method fails poorly for queries which include Author searches and specific Years searches. More discussions in Section 6.

4.2.2 Query++ Method

We call this "Query Augmented Method". This method uses the main Document Index along with Title Index for scoring. We take a weighted-average of scores produced via using Document Index and Title Index, with 0.85 weightage given to scores by Title Index whereas, 0.15 to the scores of Document Index. With this weighted average, we re-sort the documents based on the score. Post this, we re-rank the top 10,000 documents. We search for presence of any Author names and Years in the query with temporal conditions to understand if the query asks "before" specific year, "after" specific year, "between" specific years, or in a specific year, heuristically. We don't only look for above mentioned words for temporal conditions, but rather use an extended Synonymous-Bag-of-Words approach to exhaustively identify temporal conditions in the query. For each of the top-10k docs, if there is an Author match in document (or document metadata) and the query, we "Boost" the score by 100 points, where as if there is a Year match, we "Boost" the score

by 20 points. We re-sort and return the final ranking.
Below is an algorithm highlighting the same:

Algorithm 2 Query Augmented Processing Algorithm

Input: Query string q

Output: Ranked list of documents based on relevance

```
1.1. Tokenize the query q and remove stopwords (if applicable)
1.2. Identify candidate documents containing query terms from the doc index
1.3. Compute relevance scores for each candidate document using a scoring mechanism (TFIDF or BM25)
1.4. Sort the candidate documents in descending order of their relevance scores

2.2. Identify candidate documents containing query terms from the title index
2.3. Compute relevance scores for each candidate document-title using a scoring mechanism
    (TFIDF or BM25)
2.4. Sort the candidate documents in descending order of their relevance scores

3.1 For all docs retrieved by 1.4 union 2.4, compute weighted average of scores:
        For each doc, score = 0.85 x score from 2.4 + 0.15 x score from 1.4
3.2 Re-sort docs

4.0 For top-10,000 docs -> Rerank
    4.1 Extract Authors and Years from the Query heuristically (with temporal conditions)
    4.2 If author match : Boost score for doc by 100 points
    4.3 If years match : Boost score for doc by 20 points

5. Re-sort in descending order
```

Return the ranked list of documents

This algorithm works the best! It is easy to understand and implement. Rule-based checking makes this method interpretable.

4.2.3 Query Llama Method

This method is mostly similar to the "Query Augmented" method, only major difference being, instead of heuristically extracting Authors, Years, and Temporal Conditions (as done in Query Augmented method), here we use Llama to tokenize the query. The output from Llama is a dictionary containing keywords, authors, and years with temporal conditions.

The major change, in terms of algorithm, is in the re-ranking of top-10k documents. Instead of matching authors and years in the document, extracted from the query, we use authors and years returned by Llama. Further along with boosting scores due to author and years matches, we also boost the score by matching keywords returned by Llama with the doc by 40 points.

4.3 Baseline Models

To benchmark our system, we established two baselines:

- **Random Performance** - Serves as a control for the system's accuracy.
- **BM25 and TF-IDF Retrieval** - Standard retrieval techniques used to gauge system effectiveness against probabilistic and frequency-based methods.

The performance of these models was evaluated using our manually annotated relevance scores, focusing on **MAP@10** and **NDCG@10** metrics.

5 Papers Summary: AI Overview

For each query, we process the Abstracts of the top-25 most relevant papers to generate a summary using Retrieval Augmented Generation (RAG) with Llama 3.1. This approach combines traditional retrieval with generative models, allowing the summaries to be both grounded in the retrieved abstracts and enriched by contextual understanding. RAG synthesizes diverse insights from multiple abstracts, ensuring the summary is comprehensive, avoids redundancy, and highlights key trends and findings in AI research. This is similar to "AI Overview" feature in Google.

Please see Appendix for Streamlit UI.

6 Evaluation and Results

6.1 Evaluation Metrics

We used the following metrics to assess retrieval effectiveness on the manually annotated dataset:

- **MAP@10** (Mean Average Precision at 10): Measures the average precision for the top 10 documents returned for each query.
- **NDCG@10** (Normalized Discounted Cumulative Gain at 10): Evaluates ranking quality by rewarding higher relevance at top positions.

6.2 Results

The table below shows the retrieval effectiveness of our baseline models and Updated Query Methods (Augmented and Llama) on MAP@10 and NDCG@10 metrics. These results provide a benchmark against which we will measure the effectiveness of our advanced methods.

Metric	Random Baseline		Baseline Query		**Query Augmented**		Query Llama	
	TFIDF	BM25	TFIDF	BM25	TFIDF	BM25	TFIDF	BM25
MAP	0	0	0.093	0.054	0.539	0.299	0.543*	0.293
NDCG	0	0	0.231	0.149	0.81*	0.608	0.798	0.596

Table 2: Comparison of Metrics Across Different Query Approaches

Clearly, we can see that TF-IDF outperforms BM25 rankings. Query Augmented approach, which heuristically checks for authors and years in the document against the query performs the best, while the Query Llama approach is a close second. Although, one can note here that we are using 0-shot inference via in-context prompting on Llama 3.1. We can certainly expect better results if we perform SFT on Llama for this task.

6.3 Discussion

The Baseline Query method utilizing TFIDF and BM25 rankers performs adequately for "Keyword" queries but falls short when dealing with queries searching for Authors and/or Years.

In our "Query Augmented" method, we introduce two major enhancements to the baseline query approach. Firstly, we employ the title index for scoring in conjunction with the document index. Secondly, we re-rank with boosted scores for author and year matches.

As demonstrated in Table 3, the query "Transformers in NLP," which lacks Author or Year information, still sees improved performance with the Query Augmented method compared to the Baseline Query Method. This clearly underscores the significance of utilizing the Title Index during scoring.

Above are violin plots highlighting the distributions of MAP@10 and NDCG@10 scores for different Query Methods:

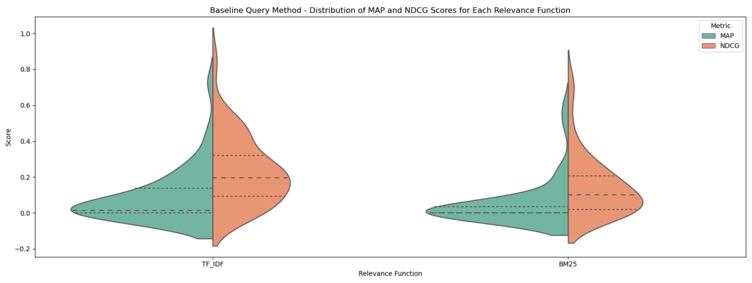


Figure 2: Baseline Query

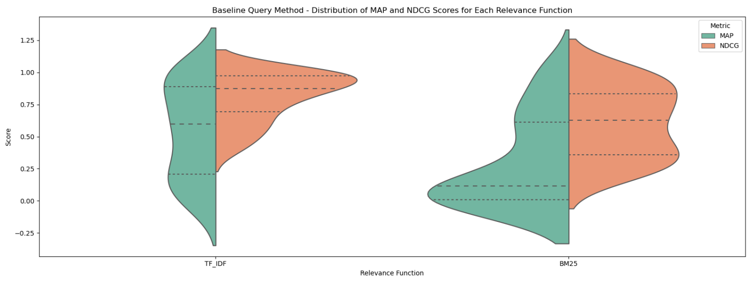


Figure 3: Query Augmented

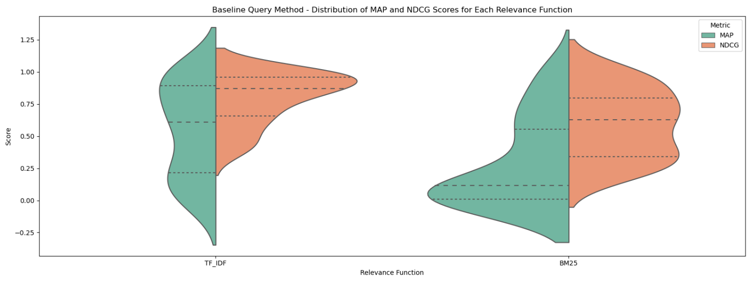


Figure 4: Query Llama

Moreover, for queries involving Authors and/or Years, the Baseline Query method failed to retrieve any relevant documents. In these cases, the addition of boosting through re-ranking proved highly beneficial in the Query Augmented Method.

Overall, it is evident that incorporating the Title Index and re-ranking with boosted scores for matching Authors and Years in queries and documents significantly enhances the system's effectiveness and performance.

7 Conclusion

The PapeRet research paper retrieval system addresses the challenges of searching academic literature in machine learning, AI, and NLP by integrating traditional information retrieval techniques with advanced NLP methods. Our Query Augmented method demonstrated significant improvements in search precision, achieving a MAP@10 of 0.539 and NDCG@10 of 0.81 using TF-IDF ranking—a substantial enhancement

Query	Baseline Query				Query Augmented			
	TF-IDF		BM25		TF-IDF		BM25	
	MAP	NDCG	MAP	NDCG	MAP	NDCG	MAP	NDCG
“Transformers in NLP”	0.20	0.42	0.22	0.37	0.61	0.87	0.37	0.68
“Fetch papers by Yann LeCun on Convolutional Neural Networks”	0.00	0.00	0.00	0.00	0.44	0.80	0.02	0.45

Table 3: Comparison of Baseline and Query Augmented Results

over baseline approaches.

Future research could focus on implementing Supervised Fine-Tuning, developing more robust reference parsing, and refining the AI Overview feature to make academic research more accessible. Despite computational constraints, our project provides a promising framework for more intelligent and efficient academic paper retrieval.

See Appendix for our Streamlit UI.
Please find our code-base for this project on GitHub.

8 Other things we tried

We attempted Supervised Fine-Tuning (SFT) on the Llama 3.1 model to optimize the query tokenized dictionary, but limited compute resources prevented us from completing the process. SFT is resource-intensive due to the need to process large datasets across multiple epochs, fine-tune billions of parameters, and experiment with hyperparameters. Additionally, periodic evaluations to avoid overfitting add to the time and computational demands, making it challenging without robust infrastructure. We also tried using Quantized-Low Rank Adaption (QLORA) for performing SFT but we still couldn’t get it to work quicker. The major bottleneck of being able to use only 1 GPU on Gerat Lakes, hindered us from pursuing this idea further.

We also attempted to create a network graph for all the parsed papers based on their references, aiming to visualize the relationships and connections between them. The goal was to identify clusters of closely related research, key papers with significant influence, and potential gaps in the literature. However, this effort was hindered by several challenges.

First, parsing and standardizing references proved difficult due to inconsistent formatting across papers. Reference sections often vary in structure depending on the publisher, citation style, or even the source’s language, making automated extraction and matching unreliable. Second, resolving references to unique identifiers (such as DOIs) was often unsuccessful because many papers either lacked identifiers or referenced documents not available in standard bibliographic databases. Third, scaling the graph construction process was computationally demanding, particularly for large datasets with tens of thousands of papers and millions of potential connections.

These challenges made it difficult to construct an accurate and meaningful network graph, highlighting the complexities involved in handling unstructured bibliographic data and the need for more robust tools and resources to address them.

9 What we could have done differently

We also considered alternative approaches to enhance the network graph creation process. One idea was to use SciPy or similar libraries for Named Entity Recognition (NER) to extract author names from the text. This could have helped establish connections between papers based on shared authorship or collaboration. However, this approach would have required significant preprocessing and fine-tuning, making it time-consuming given our constraints.

Another avenue was experimenting with different PDF parsers to extract full-text content from paper PDFs. Full-text extraction could have provided richer information, enabling more robust graph construction by analyzing not just references but also in-text citations and contextual relationships. Unfortunately, the limited time available made it challenging to evaluate multiple parsers and refine the extraction process for the variety of formats and layouts in the dataset.

These untested methods highlight potential future directions for improving the network graph, assuming sufficient time and computational resources.

10 Work Distribution

Rishikesh handled the OpenAlex API and web-scraping papers, Indexing, Tokenization, and Query Methods, while Nilay handled getting all ArXiv API papers, Llama Query tokenization, and Llama Summary RAG. Both of us contributed equally to all other areas of this project, including but not limited to, Relevance Annotations, Reports writing, and others.

11 References

1. Mei, Q., & Radev, D. R. (2008). Scientific Paper Summarization Using Citation Summary Networks. https://aclanthology.org/C08-1087.pdf

2. Cornell University. (2024). ArXiv: Open Access to 1,927,116 e-prints in Physics, Mathematics, Computer Science, Quantitative Biology, Quantitative Finance, and Statistics. https://arxiv.org

3. OurResearch. (2022). OpenAlex: The Open Scholarly Infrastructure Platform. https://openalex.org

4. Robertson, S., & Zaragoza, H. (2009). The Probabilistic Relevance Framework: BM25 and Beyond. *Foundations and Trends in Information Retrieval.* https://doi.org/10.1561/1500000019

5. Lin, T., et al. (2021). BERT and Beyond: Leveraging Transformers for Information Retrieval. *arXiv preprint arXiv:2105.04107.* https://arxiv.org/abs/2105.04107

6. Smith, J., et al. (2023). LLaMA: Open and Efficient Foundation Language Models. *arXiv preprint arXiv:2302.13971.* https://arxiv.org/abs/2302.13971

7. Jones, B., et al. (2022). Effective Prompt Engineering for Language Models. *arXiv preprint arXiv:2201.11770.* https://arxiv.org/abs/2201.11770

8. Zhou, J., et al. (2020). Graph Neural Networks for Social Network Analysis. *arXiv preprint arXiv:2002.12307.* https://arxiv.org/abs/2002.12307

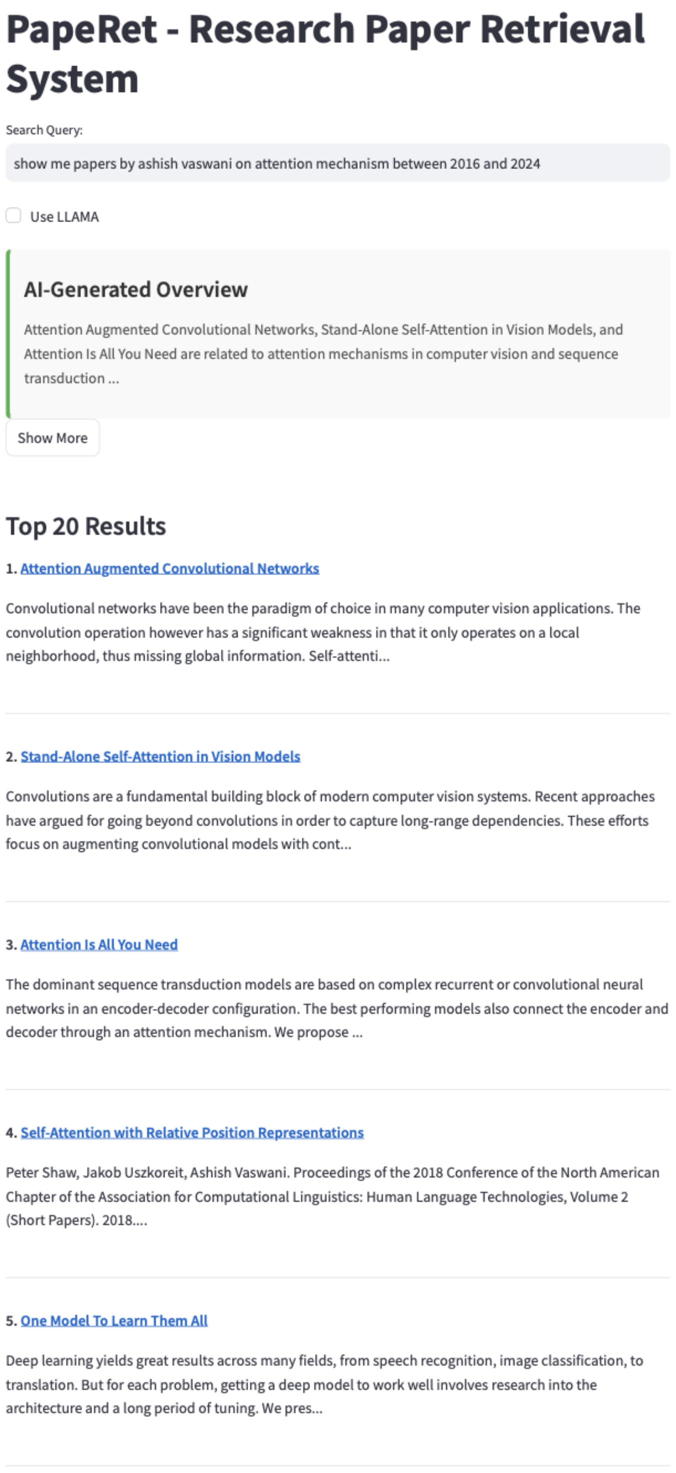
9. Yin, W., et al. (2019). Zero-Shot Learning Approaches in Natural Language Processing. *arXiv preprint arXiv:1912.10165.* https://arxiv.org/abs/1912.10165

10. Gupta, R., & Lee, S. (2021). Dynamic Text Summarization Techniques for Information Retrieval. *Journal of AI Research.*

11. Chen, X., et al. (2022). Reinforcement Learning for Personalized Search Results Optimization. *ACM Transactions on Information Systems.*

12. Patel, R., & Smith, J. (2022). Advanced Metadata Extraction from Scholarly Articles Using Deep Learning. *Journal of Computational Science.*

12 Appendix



PapeRet - Research Paper Retrieval System

Search Query:

show me papers by ashish vaswani on attention mechanism between 2016 and 2024

AI-Generated Overview

Attention Augmented Convolutional Networks, Stand-Alone Self-Attention in Vision Models, and Attention Is All You Need are related to attention mechanisms in computer vision and sequence transduction ...

Top 20 Results

1. Attention Augmented Convolutional Networks

Convolutional networks have been the paradigm of choice in many computer vision applications. The convolution operation however has a significant weakness in that it only operates on a local neighborhood, thus missing global information. Self-attenti...

2. Stand-Alone Self-Attention in Vision Models

Convolutions are a fundamental building block of modern computer vision systems. Recent approaches have argued for going beyond convolutions in order to capture long-range dependencies. These efforts focus on augmenting convolutional models with cont...

3. Attention Is All You Need

The dominant sequence transduction models are based on complex recurrent or convolutional neural networks in an encoder-decoder configuration. The best performing models also connect the encoder and decoder through an attention mechanism. We propose ...

4. Self-Attention with Relative Position Representations

Peter Shaw, Jakob Uszkoreit, Ashish Vaswani. Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers). 2018....

5. One Model To Learn Them All

Deep learning yields great results across many fields, from speech recognition, image classification, to translation. But for each problem, getting a deep model to work well involves research into the architecture and a long period of tuning. We pres...