Link Analysis

Kinda Kutkut

June 2025

Abstract

This project implements and analyzes link analysis techniques for ranking books in the Amazon Books dataset using the PageRank algorithm. A graph where books are connected based on shared user reviews is constructed and PageRank is applied to determine the most central nodes. Topic-Sensitive PageRank (TSPR) is also explored, which introduces teleportation bias toward specific book categories, allowing for more targeted importance estimation. The implementation is done from scratch in Python and tested on both the full dataset and synthetic networks. The convergence behavior and scalability of the algorithm is evaluated, demonstrating its practical effectiveness on graphs with tens of thousands of nodes.

Introduction

The explosive growth of online content has led to an increased reliance on recommendation systems and ranking algorithms to help users find relevant information. Link analysis provides a graph-based approach to determining the relative importance of items within a network by leveraging structural relationships. One of the most well-known algorithms in this domain is PageRank, which revolutionized web search by ranking pages based on their connectivity.

In this project, PageRank is used for the Amazon Books dataset. Instead of web pages, nodes represent books, and edges are formed between books that were reviewed by the same users. This creates a graph in which central nodes correspond to books that are co-reviewed frequently with many others, implicitly reflecting influence or visibility within the ecosystem.

Beyond classic PageRank, Topic-Sensitive PageRank (TSPR), a variation that adjusts the teleportation step to favor nodes belonging to a particular category is explored. This allows computation of category-specific rankings, highlighting influential books within specific genres such as "Fiction" or "History".

The goal is not only to identify top-ranked books but also to evaluate the performance, scalability, and interpretability of link-based ranking in a real-world dataset. This project showcases the applicability of graph algorithms for centrality estimation and recommendation in large datasets.

1 Dataset

The dataset used in this project is the Amazon Books Reviews dataset, published on Kaggle[1]. The version accessed for this work was retrieved on June 17, 2025. It consists of two primary CSV files:

- `Books_rating.csv`
- `Books_data.csv`

The core of the project relies on Books_rating.csv, which contains user-generated book reviews. Specifically, we use three columns from this file:

- ID – the unique identifier for each book
- Title – the name of the book
- User ID – the identifier for the user who reviewed the book

These fields were used to construct a book-book graph, where two books are linked if they were reviewed by the same user.

The second file, Books_data.csv, is used solely for the purpose of Topic-Sensitive PageRank (TSPR). In particular, we extract the categories or genres

[1] `https://www.kaggle.com/datasets/mohamedbakhet/amazon-books-reviews`

of books to construct topic-specific teleportation vectors. Other columns in the dataset, such as publication year, ISBN, or publisher, were not relevant for the ranking objectives of this project and were therefore omitted from analysis.

2 Preprocessing

Since the graph is constructed solely using the `Books_rating.csv` file, only this dataset was subjected to preprocessing. The file was loaded into a dataframe referred to as `ratings_df`.

The first step involved analyzing the distribution of review counts per user. A histogram revealed a typical long-tail (power-law) distribution which is a pattern commonly observed in real-world datasets involving items such as books, music, or web pages. On average, each user had approximately 2.4 reviews, which rounds to 2 when converted to an integer.

A similar analysis was performed on the number of reviews per book. This distribution also followed a long-tail pattern, with an average of approximately 13 reviews per book.

Given these patterns, the graph would be sparsely connected if books with very few reviews were included. In particular, books with only one or two reviews would have weak connectivity in the resulting graph. Therefore, filtering thresholds were applied to enhance the quality of the co-review graph:

- Minimum reviews per user: 5
- Minimum reviews per book: 10

Although the average number of reviews per user is 2.4, the objective is not to model the average user, but to build a meaningful co-review graph. In this graph, books are connected if two users reviewed both books. Users who review only one or two books cannot contribute to meaningful overlap between books. A threshold of five reviews per user was chosen to ensure that each user contributes to multiple connections in the graph.

After applying these filters and retaining only the relevant columns, the dataset was reduced to 1,011,440 rows. Duplicate entries were then checked and specifically, multiple reviews by the same user for the same book. A total of 28,021 such duplicate (user, book) pairs were found. These may have originated from users updating or editing reviews, or from noise and redundancy in the original dataset.

In the context of the co-review graph, only one review per (user, book) pair is meaningful, since multiple reviews should not artificially strengthen the connection between books. Therefore, duplicates were removed, resulting in a cleaned dataset referred to as `ratings_df_filtered`, which contains 983,419 unique (user, book) reviews.

3 Graph Construction

To construct the co-review graph, an inverted index–based approach was used to ensure scalability. Instead of comparing every pair of books to identify shared users, which would result in a $O(n^2)$ complexity for n books, the graph was built by iterating over each user and retrieving the set of books they had reviewed. This method, commonly referred to as inverted indexing, is a standard technique in large-scale information retrieval systems [3].

For each user, all possible combinations of the books they reviewed were generated using the $\binom{k}{2}$ formula, where k is the number of books reviewed by that user. Each such pair corresponds to a potential undirected edge in the graph. Edge weights were incremented according to the number of users who reviewed the same pair of books, encoding co-review frequency into the edge weights.

To reduce noise and ensure meaningful connectivity, only edges with a weight of at least 2 were retained. In other words, two books were connected only if at least two users had reviewed both of them.

Although the initial graph is undirected, the PageRank algorithm requires a directed graph. To adapt the graph, each edge was treated as bidirectional and two structures were created: `out_links`, containing the list of nodes each book points to, and `in_links`, representing the reverse direction.

Dangling nodes, which are nodes with no outgoing edges, are known to cause convergence issues (dead end) in PageRank [1]. However, no such nodes were observed in the final graph. This is a result of the preprocessing phase, which removed books with very few reviews and users with limited activity. As a result, all remaining nodes participate in sufficient connectivity, eliminating dead ends in the graph.

The inverted index method has a time complexity of $O\left(\sum_{i=1}^{U} k_i^2\right)$, where k_i is the number of books reviewed by user i. In practice, since most users review only a few books, this approach is significantly more efficient than the naive method, making it suitable for large-scale graphs.

This construction approach ensured both correctness and computational efficiency, making it feasible to process a massive dataset while producing a high-quality, sparse network suitable for PageRank-based ranking.

The final co-review graph contains a total of 30,921 nodes and 3,773,388 edges. The graph is relatively sparse, with a density of approximately 0.00789, meaning that only around 0.79% of all possible connections between nodes are realized. The average degree is 244.07, which reflects the mean number of connections per book.

The node with the highest degree is connected to 6,512 other books, indicating that some books are co-reviewed extremely frequently. This skew is characteristic of real-world networks, where a few highly connected nodes coexist with a large number of low-degree nodes [6].

A histogram of node degrees, plotted on a log scale, confirms this heavy-tailed distribution. The majority of books have relatively few connections, while a small number have exceptionally high degrees.

4 PageRank Implementation

The PageRank algorithm is traditionally defined using a transition probability matrix M, where M_{ij} represents the probability of transitioning from node j to node i [1]. The PageRank vector $\vec{v}^{(t)}$ is updated iteratively using the following equation:

$$\vec{v}^{(t+1)} = \beta M \vec{v}^{(t)} + (1 - \beta) \cdot \frac{\vec{e}}{n}$$

Here, β is the damping factor (typically between 0.8 and 0.9), $\vec{e}$ is the teleport vector (uniform by default), and n is the total number of nodes in the graph. The term $(1 - \beta) \cdot \vec{e}/n$ accounts for the random teleportation step, ensuring that the process remains ergodic and converges. In this implementation, a damping factor of $\beta = 0.85$ was used, matching the value implemented by Google PageRank [4].

Rather than storing an explicit transition matrix M, a dictionary-based structure was used for scalability. Specifically, two mappings were maintained:

- `in_links[i]`: the list of nodes pointing to node i
- `out_degree[j]`: the number of outgoing edges from node j

Each node's PageRank value was initialized to $1/n$, where n is the total number of nodes. In each iteration, the new PageRank for node i was computed by summing over its in-neighbors:

$$v_i^{(t+1)} = \beta \sum_{j \in \text{in}(i)} \frac{v_j^{(t)}}{\text{out\_degree}(j)} + \frac{1 - \beta}{n}$$

This simulates the matrix-vector multiplication $M\vec{v}$ using dictionaries [5]. For each node, the algorithm takes the PageRank score of each in-neighbor j from the previous iteration, divides it by the number of nodes j points to, and adds this contribution to the new rank of i. This approach is mathematically equivalent to the standard matrix formulation but far more efficient in terms of memory usage for large sparse graphs.

The power iteration method was used to update scores until convergence, defined by an L_1 norm threshold of 10^{-6} between successive vectors [1]. A maximum number of iterations was also set to 75. The standard PageRank implemented in part 4 converged after 40 iterations by convergence.

The final PageRank vector is a probability distribution that sums to 1, with each entry indicating the relative "importance" or centrality of that node in the graph.

In the results of the standard PageRank computation, the top-ranked book is *Harry Potter and the Sorcerer's Stone*, with a PageRank score of 0.000979.

This means it holds approximately 0.0979% of the total PageRank mass in the network. Given that the graph contains 30,921 nodes, the uniform baseline score for each node would be approximately $1/30921 \approx 0.000032$.

Therefore, a score of 0.000979 is more than 30 times larger than the average, indicating that this book is significantly more central and influential in the co-review network than most others. Similar patterns were observed for other top-ranked books such as *Blink: The Power of Thinking Without Thinking* and *The Catcher in the Rye*, which also appeared multiple times due to variations in format or metadata (e.g., audiobook editions).

During preprocessing, it was observed that multiple book IDs can correspond to the same title. This is typical of commercial platforms like Amazon, where the same book may exist in various formats (e.g., hardcover, Kindle edition, audiobook) or be offered by different vendors.

Although it would have been possible to unify these entries using techniques such as fuzzy string matching or ASIN normalization, the duplicates were intentionally retained. This decision reflects a realistic modeling assumption: distinct listings can independently attract user attention and engagement, and are therefore modeled as separate nodes.

By treating each entry as a unique node, the graph more closely mirrors the structure of a real-world e-commerce site. This choice also exposes an interesting behavior in PageRank: the tendency to surface structurally redundant but semantically similar nodes.

5 Experiments

Top-Ranked Books by PageRank

The standard PageRank algorithm was applied to the co-review graph, and the top 10 ranked books were extracted based on their final PageRank scores. The top result was *Harry Potter and the Sorcerer's Stone*, followed by multiple editions of *Blink: The Power of Thinking Without Thinking* and *The Catcher in the Rye*. These results align with expectations, as these books are well-known and widely reviewed.

Interpreting Scores

PageRank produces a probability distribution over the nodes. For example, the top-ranked book scored approximately 0.000979, meaning it absorbs around 0.0979% of the total importance mass. In a graph with 30,921 nodes, the uniform baseline would be roughly $1/30921 \approx 0.000032$. Hence, this book is nearly 30 times more central than average, demonstrating how PageRank identifies disproportionately influential books in the network.

Comparison with Degree Centrality

To assess whether PageRank simply reflects node degree, a side-by-side comparison was made between the top 10 books by PageRank and the top 10 by out-degree. While there is some overlap, the two rankings are different.

PageRank takes into account not only how many books a node connects to, but also how central those neighbors are. In contrast, degree centrality considers only the number of connections, without considering their quality or influence. For example, while several versions of *The Great Gatsby* dominate the degree ranking, they do not appear in the top 10 PageRank list. This suggests that although highly connected, they may not be connected to other influential nodes.

Duplication Observations

It was also observed that multiple IDs for the same title frequently appear in the top ranks. This reflects the earlier modeling decision to retain duplicate book entries (e.g., print and audiobook editions) as distinct nodes. As a result, semantically identical books can each accumulate their own score. PageRank naturally surfaces these entries when they are structurally embedded in influential neighborhoods, reinforcing the modeling assumption that separate listings may warrant separate rankings.

Summary

These experiments show that PageRank yields meaningful results aligned with human expectations of influence and popularity, while also differentiating itself from simple degree-based metrics. The overlap between rankings is partial, suggesting that PageRank captures a deeper notion of centrality that considers the global structure of the graph.

6 Topic-Sensitive PageRank

Topic-Sensitive PageRank (TSPR) is a variation of the standard PageRank algorithm that personalizes the teleportation step to favor a specific subset of nodes, typically belonging to a chosen topic or category [1]. In this project, TSPR was applied to the Amazon Books graph to identify influential books within the `Fiction` category which is the top 1 most common category.

Method

TSPR modifies the teleportation vector $\vec{e}$ in the PageRank formula:

$$\vec{v}^{(t+1)} = \beta M \vec{v}^{(t)} + (1 - \beta)\vec{e}$$

where:

- M is the normalized transition matrix,
- β is the damping factor (set to 0.85),
- $\vec{e}$ is a teleportation vector assigning non-uniform restart probabilities based on topic relevance.

Instead of teleporting uniformly to all nodes, $\vec{e}$ was constructed to assign equal probability only to books labeled as `Fiction`. All other nodes were given zero probability. The same PageRank function was reused, with the teleport vector updated accordingly. Convergence was reached after 62 iterations with a tolerance threshold of 10^{-6}.

Results

The top books ranked under the `Fiction` teleport vector included:

- *Harry Potter and the Sorcerer's Stone* — PageRank: 0.000816
- *Five People You Meet in Heaven* — PageRank: 0.000640
- *Prey* — PageRank: 0.000596
- Multiple entries of *The Hobbit* — PageRank: $\approx$ 0.000582

These results reflect books that are structurally central within the fiction subset of the network. A notable difference from the standard PageRank ranking is the absence of *John Adams*, which had ranked 8th globally but was excluded from the top TSPR results due to not being categorized as `Fiction`. This demonstrates the effect of personalized teleportation: importance becomes context-dependent.

Category Selection

The category `Fiction` was chosen based on an analysis of the most frequent book genres in the dataset. With over 23,000 labeled instances, it was the most prevalent category by a significant margin, followed by `Religion`, `History`, and `Juvenile Fiction`. This made `Fiction` a natural choice for demonstrating topic-specific ranking behavior.

7 Scalability and Performance

To assess the performance of the custom PageRank implementation, execution time was measured on synthetic Barabási–Albert graphs with increasing numbers of nodes: 1,000, 3,000, 5,000, and 8,000 [7]. These graphs simulate realistic scale-free networks, similar to the Amazon co-review graph used in this project.

As shown by the *Execution Time vs. Graph Size (Custom PageRank)* graph in the code section, execution time increases with graph size in an approximately linear fashion. This aligns with the theoretical time complexity of the power iteration method for sparse graphs:

$$\mathcal{O}(k \cdot (|V| + |E|)),$$

where k is the number of iterations, $|V|$ is the number of nodes, and $|E|$ is the number of edges.

The following execution times were observed:

- 0.13 seconds for 1,000 nodes
- 0.45 seconds for 3,000 nodes
- 1.88 seconds for 5,000 nodes
- 1.64 seconds for 8,000 nodes

The slight dip in time from 5,000 to 8,000 nodes may be due to hardware-level caching effects or minor runtime fluctuations, but the overall trend remains consistent with linear scaling.

The implementation was also successfully run on the full Amazon co-review graph containing 30,921 nodes and 3,773,388 edges. This confirmed that the approach scales efficiently to large, real-world datasets.

On the Choice of Centrality Algorithms

While the HITS algorithm (Hyperlink-Induced Topic Search) is a standard method in link analysis, it was not implemented in this project. The rationale lies in the nature of the graph: books are connected if they are co-reviewed by users, forming a symmetric (undirected) relationship. In such graphs, the concepts of hubs and authorities collapse into the same structural role, as every edge is bidirectional in effect. In this context, HITS typically yields very similar or identical scores for hubs and authorities, diminishing its analytical value. Thus the focus was on PageRank and its topic-sensitive variant, which are well-defined on undirected graphs and still provide meaningful relative importance scores.

Conclusion

This project explored the application of link analysis techniques, particularly PageRank and Topic-Sensitive PageRank, to a large-scale Amazon Books dataset. A co-review graph was constructed from user review data, and centrality scores were computed to identify influential books. The use of dictionary-based structures enabled scalable computation on both real-world and synthetic graphs. Experimental results confirmed that PageRank effectively highlights key nodes beyond simple degree counts, and the topic-sensitive variant further allowed for category-specific ranking. Overall, the methods demonstrated strong applicability for ranking and recommendation tasks in large, sparse networks.

References

1. Jure Leskovec, Anand Rajaraman, and Jeffrey Ullman. *Mining of Massive Datasets*. Cambridge University Press, 2014.

2. Kaggle. Amazon Book Reviews Dataset. https://www.kaggle.com/datasets/mohamedbakhet/amazon-books-reviews

3. Spot Intelligence. "Inverted Indexing: The Secret Behind Search Engines." October 2023. https://spotintelligence.com/2023/10/30/inverted-indexing/

4. Codestax AI. "Unveiling the Magic Behind Google's PageRank Algorithm." https://www.codestax.ai/technologies-dark/unveiling-the-magic-behind-googles-pag

5. ACME Lab, Brigham Young University. "PageRank." https://acme.byu.edu/00000180-6956-dde7-ad8c-6dde900c0001/pagerank

6. Albert-László Barabási. *Network Science*, Chapter 4: The Scale-Free Property. https://networksciencebook.com/chapter/4

7. Nair Gokul. "Random Graphs and Network Models." https://nairgokul.github.io/notes/Random-graphs-report.pdf