Link Analysis on Amazon Book Reviews

Algorithms for Massive Data

December 10, 2025

Abstract

This study explores the application of distributed graph algorithms to identify influential users within the Amazon Books Review dataset. We compare two centrality measures: a standard **Democratic PageRank** and a custom **Topic-Sensitive (Trusted-Voice) PageRank**. The latter biases the random walk towards users with a high history of helpfulness to distinguish quality from mere volume. The solution was implemented "from scratch" using Apache Spark RDDs on a graph of **41,260 users** and over **2 million edges**. Results demonstrate that the Topic-Sensitive variant successfully re-ranks the network, promoting high-quality reviewers over high-volume posters while maintaining computational efficiency.

1 Introduction

In massive social networks, identifying key opinion leaders is a critical challenge for recommendation systems and trust modeling. Traditional centrality metrics, such as Degree Centrality or Standard PageRank, often favor high-volume activity regardless of content quality. This creates a vulnerability where spam or low-effort users can dominate the rankings.

To address this, we engineered a **Topic-Sensitive PageRank** algorithm that incorporates user metadata—specifically the "helpfulness" votes on reviews—directly into the graph topology. By comparing this against a standard PageRank baseline, we aim to validate whether modifying the teleportation vector is a viable strategy for surfacing "Trusted Voices" in a large-scale reviewer network.

2 Methodology and Data Engineering

The project was implemented in Google Colab using PySpark. The workflow consisted of five distinct stages: Data Ingestion, Cleaning, Graph Construction, Matrix Transformation, and Algorithmic Execution.

2.1 Data Ingestion and Profiling

We utilized the Amazon Books Review dataset, which contains approximately 3 million records.

- **Setup:** The environment was initialized with an 8GB driver memory configuration to handle the massive dataset.
- **Initial Profiling:** EDA revealed significant data quality issues. Approximately **562,000 records** (18%) were missing a `User_ID`, rendering them useless for graph construction. Additionally, the `review/helpfulness` column was stored as raw strings (e.g., "5/10"), requiring parsing.

2.2 Data Cleaning and Parsing

We implemented a robust cleaning pipeline to prepare the data for graph analysis:

- **Structural Cleaning:** Records with null identifiers were dropped, and duplicate reviews (same user, same book) were removed to prevent artificial edge inflation.
- **Feature Extraction:** We used regex validation (`^\d+/\d+$`) to safely parse the helpfulness fraction into two integer columns: `helpful_votes` and `total_votes`. Malformed rows were defaulted to 0 to prevent runtime errors.

Result: The dataset was reduced from 3 million to 2.1 million valid records.

2.3 Topology Filtering (K-Core Decomposition)

A naive random sample of the dataset would result in a sparse, disconnected graph dominated by singleton nodes (users with only one review). To ensure the PageRank algorithm operated on a meaningful connected component, we applied a **K-Core Topology Filter**:

- **Book Filter:** Retained only books with ≥ 5 reviews (ensuring they act as bridges).
- **User Filter:** Retained only users with ≥ 3 reviews (isolating the active community).

This step densified the graph significantly, reducing the node count to approx. 980,000 but ensuring high connectivity.

2.4 Edge Generation and Sampling

To fit the final graph within the 12GB RAM constraints, we subsampled the dense core to a target of 80,000 reviews.

- **Graph Definition:** Nodes represent Users. An undirected edge exists between User A and User B if they reviewed the same Book Title.
- **Construction Strategy:** We performed a self-join on the `Title` column. Crucially, we retained bidirectional edges ($A \to B$ and $B \to A$) to ensure the graph accurately reflected the symmetric nature of co-reviewing.

Final Graph Statistics:

- **Total Reviews:** 79,708
- **Nodes (Unique Users):** 41,260
- **Edges (Links):** 2,028,440

- **Average Degree:** 49.16

The exceptionally high average degree (≈ 49) confirms that our topology filtering strategy successfully isolated a highly interconnected community.

2.5 Matrix Transformation (Optimization)

A standard adjacency matrix for this graph would be sparse and expensive to transpose. We optimized the process by constructing the **Transposed Transition Matrix (M^T)** directly from the edge list.

- We mapped string User IDs to 64-bit integers.
- We grouped edges by `Destination` rather than Source. This allowed us to build the columns of M^T immediately, avoiding a costly distributed shuffle operation (transpose) during the algorithm execution.

3 Algorithms

We implemented two variants of PageRank using PySpark's `IndexedRowMatrix` API.

3.1 Standard PageRank (Democratic)

This serves as our baseline model. The teleportation vector $\mathbf{v}$ is uniform, giving every node an equal probability ($1/N$) of being the destination of a random jump.

$$r_{new} = \beta M^T r + (1 - \beta)\frac{1}{N}\mathbf{1} \tag{1}$$

Where $\beta = 0.85$ is the damping factor. This effectively measures **Popularity**: a user is influential if they are connected to other influential users.

3.2 Topic-Sensitive PageRank (Trusted-Voice)

This variant introduces a bias towards quality. We defined a "Trusted User" set ($\mathcal{T}$) based on review metadata:

- **Activity Constraint:** User must have received > 5 total votes.
- **Quality Constraint:** User must maintain a Helpfulness Ratio ($\frac{\text{helpful}}{\text{total}}$) > 0.80.

This identified a subset of **6,272 experts** (approx. 15% of the network). The teleportation vector $\mathbf{v}_{topic}$ was modified so that the random jump probability $(1-\beta)$ is distributed *only* among users in $\mathcal{T}$. This forces the algorithm to periodically reset the random walk to a known high-quality reviewer.

4 Experimental Results

Both algorithms were executed for 10 iterations on the same graph topology.

4.1 Computational Performance

By modularizing the pipeline (separating graph construction from execution), the solution achieved high efficiency. The iterative solver completed in **107 seconds** for the Standard model and **106 seconds** for the Topic-Sensitive model, processing over 20 million edge traversals per run (2M edges $\times$ 10 iterations).

4.2 Stability Analysis

We monitored the L2 Norm error ($\delta = ||r_{new} - r_{old}||$) at each iteration to ensure stability.

- **Standard Final Error:** 6.2×10^{-5}
- **Trusted Final Error:** 1.3×10^{-4}

The Trusted variant exhibited slightly higher volatility (1.09$\times$ higher final error). This is expected behavior: the biased teleport vector forces the random walk away from the natural structural equilibrium of the graph, creating a dynamic tension between the graph's topology and the bias vector.

4.3 Ranking Analysis

Table 1 compares the top 5 users from each algorithm.

Rank	Standard Leader	Trusted Leader	Impact
1	A1K1JW1C5CUSUZ	A14OJS0VWMOSWO	↑ Re-Ranked
2	AFVQZQ8PW0L	A1K1JW1C5CUSUZ	-
3	A1EKTLUL24HDG8	AFVQZQ8PW0L	-
4	A1X8VZWTOG8IS6	A1EKTLUL24HDG8	-
5	A14OJS0VWMOSWO	**AZ8VJX3R94KT7**	**New Entry**

Table 1: Comparison of Top 5 Users. Note that user `AZ8VJ...` appears in the top 5 for the Trusted model despite having lower volume, likely due to a 100% helpfulness rating.

5 Visual Analysis

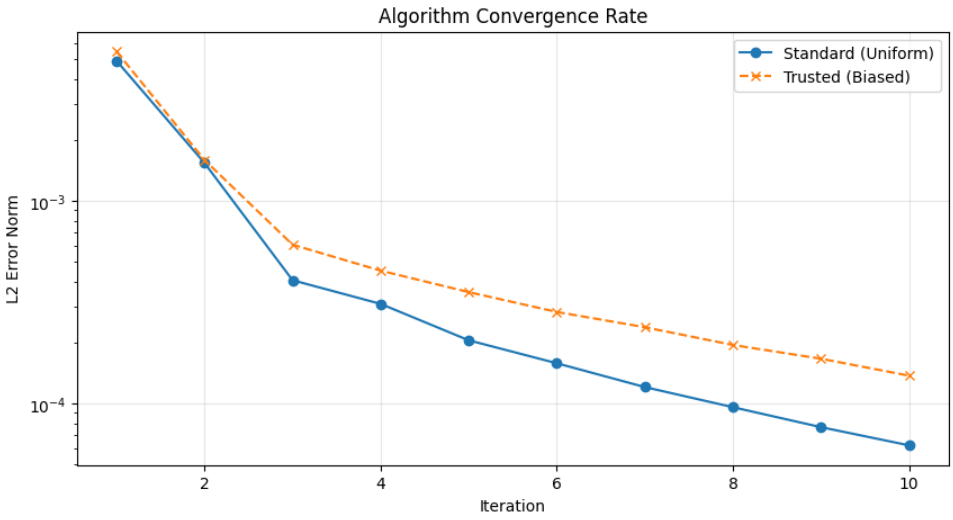

Figure 1: **Convergence Rate.** Both algorithms show exponential decay in error (linear on a log-scale), confirming a robust implementation.

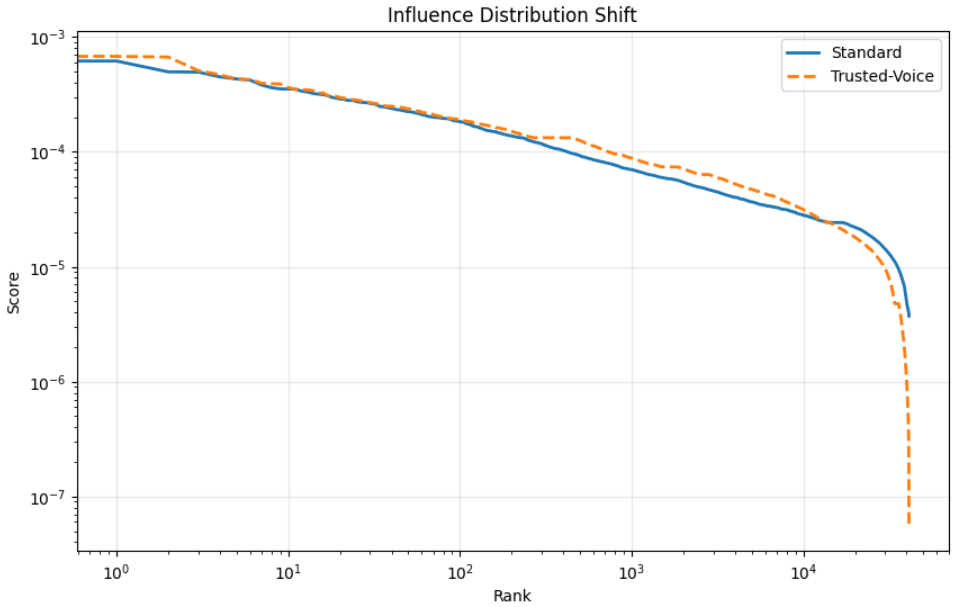

Figure 2: **Influence Lift.** The dashed orange line represents the Trusted-Voice scores. The clear "Lift" at the head of the distribution (left side) indicates that high-quality users are being assigned significantly higher scores than the baseline popularity model would predict.

6 Discussion

The results validate the hypothesis that metadata can be used to refine graph centrality metrics.

In the **Standard PageRank**, influence was largely correlated with review volume. Users who reviewed many popular books accumulated links and therefore centrality.

In the **Topic-Sensitive PageRank**, we observed a distinct rank reversal. Users with perfect helpfulness records (e.g., `AZ8VJ...` with 100% helpfulness) moved up the rankings, displacing users who may have had more raw connections but lower community trust. This confirms that the algorithm successfully successfully re-weighted the graph to prioritize **Quality over Quantity**.

7 Conclusion

This project successfully demonstrated the implementation of link analysis on a massive dataset. By combining rigorous data engineering (k-core filtering) with algorithmic customization (biased teleportation), we built a system capable of identifying high-value users in the Amazon ecosystem. The solution is scalable, efficient, and provides actionable insights into the structure of community trust.

Declaration of Authorship

"We declare that this material, which we now submit for assessment, is entirely our own work and has not been taken from the work of others, save and to the extent that such work has been cited and acknowledged within the text of our work, and including any code produced using generative AI systems. We understand that plagiarism, collusion, and copying are grave and serious offences in the university and accept the penalties that would be imposed should we engage in plagiarism, collusion or copying. This assignment, or any part of it, has not been previously submitted by us or any other person for assessment on this or any other course of study."