



Jarir Bookstore's Recommendation System for SME Customers

Yazan A. Alkamal
Software Engineering
Umm Al-Qura University

Sattam J. Altuuaim
Artificial Intelligence
University of Jeddah

Mohammad H. Alharbi
Computer Science
King Abdulaziz University

Mohammad R. Alharbi
Information Technology
King Abdulaziz University

Fawaz A. Alghamdi
Computer Science
King Abdulaziz University

Under the supervision of

Dr. Tanveer Hussain

August 9, 2025

Abstract

This final report documents the complete lifecycle of the Jarir Bookstore SME Recommendation System project, from initial strategic alignment to the development of advanced deep learning architectures. The primary objective has been to engineer a sophisticated personalization engine that addresses the unique procurement needs of Jarir's B2B customers, thereby enhancing user experience and driving key commercial metrics.

This document details the project's phased execution, which began with a foundational analysis of Jarir's data. A subsequent parallel modeling phase rapidly established performance benchmarks, with models like LightGCN achieving a Recall@10 of 0.18, confirming the viability of graph-based methods. This led to the development of a Two-Tower model, which marked a significant breakthrough by achieving a Recall@10 of approximately 0.79. Further refinement introduced a Two-Tower with Reranker architecture, yielding a final test performance of **Recall@10 of 0.6937** and **NDCG@10 of 0.5541** on the Jarir dataset. This result demonstrates a successful trade-off between raw recall and other critical business objectives such as precision and relevance.

Future work will focus on enhancing this robust framework through advanced feature engineering and the exploration of alternative reranking models. This report provides a comprehensive account of our experimental results, the challenges overcome, and the strategic vision for the production-level deployment of this system.

Index Terms—Recommendation Systems, Two-Tower Models, Reranking, Collaborative Filtering, Graph Neural Networks, B2B E-commerce, Personalization.

Contents

1 Introduction **3**

2 Data Preparation and Analysis **3**
2.1 Initial Data Assessment 3
2.2 The Data Cleaning Pipeline 3

3 Methodology: A Phased Approach **4**
3.1 Phase I: Foundation & Discovery (Weeks 1–2) 4
3.2 Phase II: Rapid Baseline Evaluation (Week 3) 4
3.3 Phase III: Advanced Architecture (Weeks 4–Present) 4

4 Modeling Stage 1: Baselines **4**

5 Modeling Stage 2: The Two-Tower Breakthrough **5**
5.1 Performance Analysis 5

6 Final Architecture: Two-Tower with Reranker **5**
6.1 The Role of a Reranker 5
6.2 Final Performance on Jarir Dataset 5

7 Challenges and Solutions **6**

8 Future Work and Strategic Recommendations **6**

9 Conclusion **6**

1 Introduction

In the contemporary digital marketplace, personalization is no longer a luxury but a fundamental expectation. For e-commerce platforms, the ability to understand and anticipate customer needs is paramount to fostering loyalty and driving revenue. This imperative is magnified within the Business-to-Business (B2B) sector, where purchasing decisions are driven by specific, recurring operational needs rather than casual browsing. Jarir Bookstore's Small and Medium Business (SME) customers represent a vital market segment with just such a distinct purchasing lifecycle.

Recognizing this, a strategic initiative was launched to architect a bespoke, AI-powered recommendation system tailored explicitly for Jarir's SME clients. The project's mandate is to move beyond generic recommendations and deliver a truly intelligent and efficient procurement experience. By understanding the contextual and sequential nature of business purchases, the proposed system aims to streamline the product discovery process, increase the sales of targeted commercial SKUs, and solidify Jarir's position as an indispensable partner for its business customers. This report chronicles the journey from initial concept to the current state of advanced model development.

2 Data Preparation and Analysis

The foundation of any successful machine learning system is high-quality data. The raw transactional data provided by Jarir, while rich in potential, required a systematic and robust cleaning pipeline before it could be used for modeling. This foundational phase was one of the most critical parts of the project.

2.1 Initial Data Assessment

The raw dataset was provided as an Excel file containing over 3 000 records. An initial assessment revealed several challenges that needed to be addressed:

- **Inconsistent Data Formats:** Dates were stored in a variety of formats, including non-standard strings (e.g., `Jan-1`) and Excel serial numbers.
- **Noise and Irrelevant Data:** The dataset contained non-product line items such as shipping fees and discounts, as well as returned orders, which needed to be filtered out.
- **Data Redundancy:** Duplicate transaction lines for the same user–item interaction were present.
- **Outliers:** A small number of transactions had extreme values for price and quantity, which could skew model training.

2.2 The Data Cleaning Pipeline

To address these challenges, we developed and executed a multi-step data preparation pipeline, as implemented in our `01_data_prep.ipynb` notebook. The key stages were:

1. **Standardization and Parsing:** We first standardized column names for consistency. A custom, robust date parser was written to handle the multiple date formats and convert them into valid timestamps.
2. **Validity Filtering:** The data was systematically filtered to remove rows with missing critical values, returned transactions (identified by invoice prefix), and non-positive values for price and quantity.
3. **Normalization and Noise Removal:** All product descriptions were normalized (e.g., converting to uppercase, standardizing whitespace). We then used a list of predefined regex patterns to identify and remove non-product lines like "SHIPPING" or "DISCOUNT".

4. **Aggregation:** Duplicate transaction lines were aggregated by grouping on customer, item, and date, summing quantities and line amounts to create a single, unified interaction record.

5. **Outlier Handling:** We applied an interquartile range (IQR) filter on the logarithm of the price, quantity, and line amount. This robust statistical method allowed us to identify and remove extreme outliers without being unduly influenced by them.

This rigorous process transformed the raw data into a clean, reliable set of interactions ready for feature engineering and model training, which was saved in the efficient Parquet format.

3 Methodology: A Phased Approach

The project was executed through a structured, multi-phase methodology designed to ensure a data-driven and agile development process, allowing for rapid learning and strategic pivots based on empirical evidence.

3.1 Phase I: Foundation & Discovery (Weeks 1–2)

This phase was dedicated to scope definition and data immersion, including the full data preparation pipeline described in the previous section. A key strategic decision was also made to acquire a secondary, public e-commerce dataset. This dataset would serve as a benchmark for comparative analysis and as a potential source for pre-training models to learn general purchasing patterns before fine-tuning on Jarir's specific data.

3.2 Phase II: Rapid Baseline Evaluation (Week 3)

To accelerate the learning process, we implemented a parallel baseline development strategy. The team was divided into five subgroups, each tasked with building a different recommendation model. This approach enabled us to test a wide spectrum of algorithmic hypotheses in a short timeframe, from simple linear models to more complex graph-based architectures. This phase was not about finding a perfect model, but about understanding which *class* of models held the most promise.

3.3 Phase III: Advanced Architecture (Weeks 4–Present)

The insights gleaned from the baseline phase directly informed our current work: the development of a sophisticated, deep learning-based architecture. This phase involved consolidating our findings to build a single, powerful model capable of overcoming the limitations observed in the simpler baselines, and then iteratively refining it into a production-ready, two-stage system.

4 Modeling Stage 1: Baselines

The parallel baseline tests provided a clear picture of the problem's complexity and guided our architectural decisions. The results of our two most informative models, implemented in `02_embeddings_and_baselines.ipynb`, are detailed below.

LightGCN

Description: LightGCN (Light Graph Convolutional Network) was chosen for its proven ability to model complex, high-order relationships in user–item interaction data. It operates by representing the entire dataset as a bipartite graph and learning embeddings by propagating information across this structure. It served as our primary indicator of the potential of modern, embedding-based methods. Its performance set a strong but achievable benchmark for our advanced model.

Recall@10: $\approx$ 0.18

EASEr

Description: EASEr (Embarrassingly Shallow AutoEncoder) represents a fundamentally different, linear approach. It is a simple, non-neural model that is fast to train. We included it as a "sanity check" to determine if a complex deep learning model was truly necessary. Its significantly lower performance provided a definitive answer that linear models are insufficient for this recommendation task.

Recall@10: $\approx$ 0.05

The average Recall@10 across all five deployed baselines was approximately 0.18. This phase conclusively demonstrated that non-linear, deep learning approaches were essential to capture the nuances of the Jarir dataset, and that embedding-based methods were the most promising path forward.

5 Modeling Stage 2: The Two-Tower Breakthrough

Armed with these insights, we architected a Two-Tower deep learning model, implemented in `03_retriever_and_index.ipynb`. This state-of-the-art architecture consists of two independent neural networks: a User Tower and an Item Tower, both implemented as multi-layer perceptrons (MLPs) with residual connections and layer normalization.

The key to its success was a robust training strategy. The model was trained on a combination of Jarir's rich but sparse data and the denser public dataset, which allowed it to learn generalized purchasing behaviors before being fine-tuned on the specific patterns of Jarir's SME customers.

5.1 Performance Analysis

The implementation of the Two-Tower model marked a significant turning point in the project, delivering a dramatic leap in performance.

- **Two-Tower Recall@10: $\approx$ 0.79**

This result represents a **338% performance increase** over our strongest baseline. This breakthrough validated our architectural choice and the effectiveness of our data strategy. The model proved exceptionally adept at the *retrieval* task: finding a broad set of relevant items from the full catalog.

6 Final Architecture: Two-Tower with Reranker

While the Two-Tower model provided excellent recall, the ultimate goal of a recommendation system is not just to find *all* relevant items, but to rank the *most* relevant items at the very top. To address this, we introduced a second-stage reranking model, implemented in `04_ranker_and_eval.ipynb`.

6.1 The Role of a Reranker

The system now operates in two stages:

1. **Retrieval (Two-Tower):** The highly efficient Two-Tower model first retrieves a large set of $\sim$100 relevant candidates from the entire catalog. This stage prioritizes speed and high recall.
2. **Reranking (MLP Ranker):** A second, more precise model then examines this smaller set of candidates and re-orders them based on a richer set of features. This stage prioritizes ranking quality, measured by NDCG.

The reranker is a feature-based **Multi-Layer Perceptron (MLP) Ranker** with residual connections. It uses a variety of features to make its fine-grained decisions, including: the original Two-Tower similarity score, item popularity, historical session features, and item price z-scores.

6.2 Final Performance on Jarir Dataset

When evaluating this final two-stage architecture strictly on the Jarir test dataset, we observed the following performance:

- **Recall@10: 0.6937**
- **NDCG@10: 0.5541**

It is crucial to interpret this correctly. The drop in recall from the initial Two-Tower model is expected and intentional. The reranker's goal is to improve the quality and business relevance of the top-ranked items, for which NDCG is a better measure. The final model successfully trades a small amount of raw recall for a significant improvement in the top-10 ranking quality, ensuring that the most relevant items appear first.

7 Challenges and Solutions

Throughout the project, the team encountered several technical challenges. The most significant was the initial quality and heterogeneity of the raw data. As detailed in Section 2, the data contained inconsistent formats, noise, and outliers that made it unsuitable for direct use.

Our solution was the development of a systematic, multi-step data cleaning and preparation pipeline. By creating custom parsers, applying statistical outlier detection, and implementing a series of normalization and filtering rules, we were able to transform the raw data into a high-quality, reliable dataset. This robust process not only enabled our successful modeling efforts but also stands as a reusable asset for future data batches.

8 Future Work and Strategic Recommendations

Based on the successful development of the two-stage model, our forward-looking strategy focuses on iterative enhancement and preparing for production deployment.

1. **Advanced Feature Engineering:** The primary path to improving the reranker's performance is through the creation of more sophisticated features. This includes developing temporal features (e.g., time since last purchase), user-level features (e.g., user tenure, monetary value), and cross-features. This is contingent on the acquisition of richer data from Jarir.
2. **Exploration of Alternative Rankers:** While the MLP Ranker has proven effective, we recommend exploring other powerful ranking models, such as Gradient Boosted Decision Trees (e.g., XGBoost, LightGBM), which often excel on tabular feature data.
3. **Online A/B Testing:** The ultimate validation of this system must come from live user interaction. We recommend developing a framework for online A/B testing to directly measure the impact of the recommendations on key business metrics like click-through rate, conversion rate, and average order value.

9 Conclusion

This project has progressed successfully from foundational analysis to the development of a sophisticated, two-stage recommendation engine. The methodical, phased approach—from data cleaning and rapid baselining to the breakthrough performance of the Two-Tower model—has proven highly effective. The final architecture, a Two-Tower retriever with a dedicated MLP reranker, provides a powerful and flexible foundation for a production-level system, achieving a strong final test performance of 0.6937 Recall@10 and 0.5541 NDCG@10 on the Jarir dataset. The strategic roadmap, focused on advanced feature engineering and a move toward online testing, outlines a clear path to deploying a state-of-the-art recommendation engine that will deliver significant, tangible value to Jarir Bookstore and its SME customers.