



A Hybrid Two-Stage Neuro-Symbolic Framework for Explainable Recommendation in Sparse Data

By

Ermias Alemayehu

Submitted to the School of Information Technology and Engineering
in partial fulfillment of the requirements for the degree of
Master of Science in Artificial Intelligence

Advisor: Beakal Gizachew(PhD)

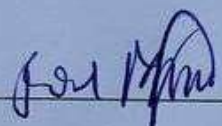
College of Technology and Built Environment
Addis Ababa University
Addis Ababa, Ethiopia

February 2026

APPROVAL

This is to certify that this thesis titled "*A Hybrid Two-Stage Neuro-Symbolic Framework for Explainable Recommendation in Sparse Data*" is prepared by **Ermias Alemayehu** and submitted in partial fulfillment of the thesis-option requirements for the Degree of Master of Science in Artificial Intelligence at the School of Information Technology and Engineering, College of Technology and Built Environment.

Name	Signature	Date
<u>Dr. Bekele Gizachew</u>		<u>may-21-2026</u>
(Advisor)		

<u>Dr. Solomon T.</u>		<u>may-21-2026</u>
(External Examiner)		

<u>Dr. Sanjathun Boyale</u>		<u>may-21-2026</u>
(Internal Examiner)		

Abbreviations

Symbol	Meaning
CF	Collaborative Filtering
GCN	Graph Convolutional Network
LightGCN	Light Graph Convolutional Network
BPR	Bayesian Personalized Ranking
MLP	Multi-Layer Perceptron
HR@K	Hit Ratio at rank K
NDCG@K	Normalized Discounted Cumulative Gain at rank K
Recall@C	Candidate-set Recall at candidate size C
TopC	Top- C candidates produced by the retriever
Union (candidates)	Candidate list formed by combining retriever and rule-expanded candidates
SSL	Self-Supervised Learning
SGL	Self-supervised Graph Learning
SimGCL	Simplified Graph Contrastive Learning
InfoNCE	Contrastive learning objective (Noise-Contrastive Estimation variant)
supp	Support (association-rule frequency measure)
conf	Confidence (association-rule conditional strength)
lift	Lift (association strength relative to independence)
MIN_CONF	Minimum confidence threshold for keeping rules
MIN_LIFT	Minimum lift threshold for keeping rules
TOPK_PER_Y	Maximum number of rules retained per consequent item y
G_{UI}	User-item bipartite interaction graph
A_{bui}	Symmetrically normalized adjacency of G_{UI}
G_r	Rule-induced item graph derived from mined rules
A_r	(Row-)normalized adjacency/operator on G_r
R^+	Implicit positive interaction set (training positives)
K (layers)	Number of LightGCN propagation layers (context-dependent)
K (eval)	Length of the ranked list in top- K evaluation (context-dependent)
C	Candidate set size used in retrieval (Top- C)
ML-1M	MovieLens-1M dataset

Contents

Abbreviations	2
Contents	i
List of Figures	v
List of Tables	vi
Abstract	1
1 Introduction	2
1.1 Background and Motivation	3
1.2 Statement of the Problem	4
1.3 Research Questions	6
1.4 Research Objectives	6
1.4.1 General Objective	6
1.4.2 Specific Objectives	7
1.5 Significance of the Study	7
1.6 Scope of the Study	8
1.7 Contributions	9
2 Literature Review	11
2.1 Introduction and Chapter Roadmap	11
2.1.1 Implicit feedback as MNAR data and exposure bias	12
2.1.2 Negative sampling choices and their downstream effects	13
2.1.3 Full-ranking vs. sampled-ranking evaluation	15
2.2 Multi-Stage Recommendation Pipelines and the Retrieval Bottleneck	16
2.2.1 Candidate generation families in practice	17
2.2.2 Candidate set construction and union policies	18
2.2.3 Training-serving mismatch in two-stage pipelines	18
2.2.4 Candidate recall and the retrieval ceiling	19
2.3 Graph Collaborative Filtering for Stage-1 Retrieval	20
2.3.1 Why not sequential models as the main retriever	22
2.3.2 Known weaknesses of graph collaborative filtering under sparsity	24

2.4	Explainable Recommendation: Mechanisms, Forms, and Evaluation	25
2.4.1	Explanation objectives: user-facing vs developer-facing	26
2.4.2	Faithfulness vs plausibility: what the literature warns about	28
2.5	Symbolic and Association-Rule Evidence for Recommendation	30
2.5.1	Rule mining choices and quality control	32
2.5.2	Bias and failure modes of rule evidence	34
2.6	Explainable Reranking with Explicit Evidence Features	35
2.6.1	Learning-to-Rank Framing for Reranking under Candidate Conditioning	37
2.7	Beyond-Accuracy Properties as Secondary, Controllable Outcomes	38
2.8	Summary of Gaps and Research Positioning	40
3	Methodology	43
3.1	Problem Setting and Notation	43
3.1.1	Implicit-Feedback Recommendation Setting	44
3.1.2	Two-Stage Recommendation Formulation	44
3.2	Data Acquisition	45
3.3	Data Preprocessing	46
3.3.1	Interaction Construction	46
3.3.2	Chronological Leave-One-Out Protocol	46
3.3.3	Training-Item Masking During Evaluation	46
3.3.4	Leakage-Safe Use of Symbolic Evidence	47
3.4	Overview of the Proposed Architecture	47
3.5	Stage-1 Collaborative Retrieval with LightGCN	48
3.5.1	User-item graph construction	48
3.5.2	LightGCN propagation	49
3.5.3	Retriever training objective	49
3.5.4	Base candidate retrieval	49
3.6	Train-only Symbolic Candidate Expansion	50
3.6.1	Association-rule mining from training histories	50
3.6.2	Rule-based scoring and expansion	50
3.6.3	Optional metadata/text expansion	51
3.6.4	Budgeted union and candidate capping	51
3.7	Stage-2 Explainable Reranking	52
3.7.1	Reranker design rationale	52
3.7.2	Feature construction	53
3.7.3	Reranker model	53
3.7.4	Reranker training with candidate-conditioned slates	53
3.8	Explanation Construction and Evidence Reporting	54

3.8.1	Evidence-aligned explanation principle	54
3.8.2	Template-based explanation generation	54
3.8.3	Explanation coverage (EC@K)	55
3.9	Faithfulness Assessment via Evidence Removal	55
3.10	Experimental Protocol and Evaluation Metrics	56
3.10.1	Top- K ranking metrics	56
3.10.2	Candidate recall and retrieval-ceiling interpretation	56
3.10.3	Bounded-candidate evaluation and sampled negatives	57
3.10.4	Ablation design and reporting protocol	57
4	Experiments	58
4.1	Experimental Setup	59
4.1.1	Implementation and compute environment	59
4.1.2	Datasets and preprocessing	59
4.1.3	Evaluation protocol	60
4.1.4	What I mean by “hybrid” and “explainable” in this chapter	60
4.1.5	Compared configurations (A0/A1/A2/A3 and FULL)	61
4.1.6	EFE-proxy reranking (definition and purpose)	61
4.1.7	Fixed hyperparameters (FULL configuration)	62
4.2	External Baseline Retriever Comparison (Fixed Budget)	62
4.3	Latency and Memory Footprint (Feasibility Evidence)	63
4.3.1	Runtime / throughput snapshot (MovieLens-1M)	63
4.3.2	Approximate memory footprint (lower-bound accounting) . .	63
4.4	Stage-1 Candidate Generation Analysis	64
4.4.1	Candidate recall under fixed budgets	65
4.4.2	Retrieval ceiling interpretation	65
4.5	Stage-2 Reranking Performance	66
4.5.1	End-to-end results (all users)	66
4.5.2	Conditional reranking (candidate-hit users only)	66
4.6	Explainability and Evidence Diagnostics	67
4.6.1	Explanation coverage (EC@K)	67
4.6.2	Faithfulness via evidence removal (fidelity-drop)	67
4.7	Ablation Studies	68
4.7.1	Ablation: Candidate Recall@C	68
4.7.2	Ablation: End-to-end HR/NDCG at K=10	68
4.8	Limitations	68
4.9	Research Questions and Answers	69

5	Conclusion and Future Work	71
5.1	Conclusion	71
5.2	Future Work	72
	References	73

LIST OF FIGURES

3.1	Research methodology design following a Design Science process [1].	44
3.2	Overview of HiX2S: Stage 1 collaborative retrieval, train-only symbolic expansion, budgeted candidate union with a fixed cap, and Stage 2 evidence-aware reranking.	47

LIST OF TABLES

2.1	Representative synthesis matrix using broad system properties. ✓ = explicit/central; ◦ = partial/tangential; × = not a primary focus.	41
4.1	Software and compute environment used for the reported experiments.	59
4.2	Dataset and split statistics used in the reported experiments. . . .	60
4.3	FULL configuration used in the reported experiments.	62
4.4	Retrieval-stage comparison under fixed candidate budget ($C_{\text{cap}} = 2000$). Values are test Recall@C.	63
4.5	Retriever-only ranking under fixed candidate budget ($C_{\text{cap}} = 2000$): test HR@10 and NDCG@10.	63
4.6	Retriever-only ranking under fixed candidate budget ($C_{\text{cap}} = 2000$): test HR@20 and NDCG@20.	64
4.7	Runtime snapshot for the reported MovieLens-1M configuration (environment-specific).	64
4.8	Approximate memory footprint (lower-bound estimates, float32 embeddings, COO int32 edges).	65
4.9	Amazon Books Stage 1 candidate Recall@C at fixed cap ($C_{\text{cap}} = 2000$).	65
4.10	MovieLens-1M Stage 1 candidate Recall@C at fixed cap ($C_{\text{cap}} = 2000$).	66
4.11	Yelp Stage 1 candidate Recall@C under fixed capped candidates. .	66
4.12	Stage-2 end-to-end ranking performance (all users) under fixed candidate budgets.	67
4.13	Explanation coverage on the test split (FULL configuration). . . .	67
4.14	Retained fixed-candidate fidelity-drop diagnostics. Positive drops indicate operational dependence on rule evidence in Stage 2 scoring.	68
4.15	Candidate Recall@C across ablations under $C_{\text{cap}} = 2000$	68
4.16	End-to-end metrics at $K = 10$ across ablations (test).	69

Abstract

Large-scale recommender systems commonly employ two-stage pipelines in which a fast retriever generates a candidate set and a more expressive reranker produces the final top- K recommendations. In sparse implicit-feedback settings, however, end-to-end performance is constrained by candidate availability, and explanation mechanisms are often weakly connected to ranking decisions. This thesis proposes a hybrid two-stage neuro-symbolic framework for explainable recommendation in sparse data that integrates a LightGCN retriever, train-only symbolic candidate expansion based on association-style co-occurrence rules, and an evidence-aware MLP reranker under a fixed candidate-budget constraint. The framework is evaluated using Candidate Recall@C, HR@K, NDCG@K, explanation coverage (EC@K), and a fidelity-style evidence-removal diagnostic to assess whether symbolic features influence ranking behavior under fixed candidates. Experiments on Amazon Books, MovieLens-1M, and Yelp show that the impact of symbolic expansion and explainable reranking is dataset-regime dependent; in retrieval-bottlenecked settings, improvements in Candidate Recall@C are accompanied by corresponding gains in HR@K and NDCG@K, consistent with a retrieval-ceiling interpretation of two-stage recommendation. On Amazon Books, the full configuration achieves Candidate Recall@C = 0.3135/0.2527, HR@10 = 0.0698/0.0388, and NDCG@10 = 0.0391/0.0201 on validation/test, with EC@10 approximately 0.86. Overall, the thesis contributes a neuro-symbolic two-stage recommendation framework and an evaluation strategy that jointly analyzes retrieval constraints, ranking effectiveness, explanation coverage, and evidence dependence under fixed serving budgets.

Chapter 1

Introduction

Modern recommender systems help users navigate large catalogs by estimating which items are most relevant for each user. At the scale of real platforms, recommendations are rarely produced by a single model that scores every item. Instead, a standard multi-stage design is commonly adopted: a computationally efficient retrieval module first nominates a small candidate set, and a more expressive ranking or re-ranking model then refines these candidates into the final top-(K) list [2, 3]. This design is driven by scalability constraints, since scoring an entire catalog with an expressive model is usually infeasible under latency and cost requirements [2, 3]. As a consequence, retrieval must be both efficient and high-recall because it determines which items are available to the downstream ranker [2, 3].

In a two-stage pipeline, retrieval quality becomes a bottleneck. If the held-out relevant item is missing from the candidate set, a downstream ranker cannot recover it regardless of model capacity; therefore, end-to-end performance is bounded by candidate-set recall [3]. For transparency, retrieval quality (Candidate Recall@C) is reported alongside end-to-end ranking metrics such as HR@K and NDCG@K, and these results are interpreted under the known sensitivity of offline recommendation evaluation to sampling choices and protocol details [4].

Within this setting, a two-stage neuro-symbolic recommender is instantiated in which LightGCN retrieves candidates and a symbolic expansion component

injects explicit evidence into both the candidate pool and the reranking features. LightGCN is a representative graph-based collaborative filtering retriever that simplifies graph convolution by retaining neighborhood aggregation over the user-item interaction graph, and it is effective and efficient for sparse implicit-feedback recommendation [5]. To make the system more inspectable, symbolic expansion is implemented using association-rule co-occurrence signals mined from training histories only, which avoids leakage into validation and test evaluation [6, 7]. An explainable reranker then consumes neural retrieval scores together with rule-derived features to produce the final top-(K) list. Explanation behavior is characterized using explanation coverage (EC@K) and a fidelity-style masking diagnostic in which rule-derived features are removed at inference time and the resulting metric drop is measured [8–10].

1.1 Background and Motivation

Recommendation is studied in the implicit-feedback regime, where observed interactions act as positives and unobserved user-item pairs are treated as unknown rather than explicit negatives [11–13]. In large catalogs, producing personalized recommendations under latency and cost constraints is commonly handled with multi-stage pipelines, where a fast candidate generation stage narrows the item space and a downstream ranker or reranker refines a bounded slate [2, 14, 15]. Under this architecture, candidate generation imposes a hard ceiling on end-to-end performance: items not retrieved cannot be promoted later, so retrieval quality should be measured directly alongside top- K metrics [15]. In the implemented pipeline, retrieval is instantiated with LightGCN, a simplified graph-based collaborative filtering model that serves as a strong and efficient baseline for implicit recommendation [5].

Latent collaborative models can be difficult to audit because their decisions are encoded in embeddings and do not inherently expose human-interpretable

evidence [16]. To improve inspectability, symbolic evidence is integrated via association rules mined from training histories only, using the interpretable rule form $x \rightarrow y$ [6]. This symbolic channel supports (i) candidate expansion under a strict budget and (ii) explicit reranker features, so that explanations are grounded in signals that can influence the final ordering rather than being attached purely post-hoc [6, 16]. When optional metadata representations are available, semantic-distance proxies are treated as auxiliary controls rather than as a required signal, so the pipeline remains applicable when text coverage is incomplete [17].

Because plausible explanations can be misleading if they do not reflect the causes of a model’s output, evaluation is aligned with explainability work that distinguishes plausibility from faithfulness and recommends testing whether explanations are supported by evidence dependence [8, 9]. In this setting, evidence dependence is operationalized with an evidence-removal test that ablates the rule-derived feature at inference time by setting the symbolic rule score $s_{\text{rule}}(u, i)$ to zero while keeping the collaborative retrieval score $s_{\text{enc}}(u, i)$ unchanged; no other evidence channels are removed in this diagnostic. The corresponding drop in ranking quality (HR@10 and NDCG@10) is then measured as a proxy for whether symbolic evidence contributes materially to decisions [8–10]. Experiments are conducted on the Amazon Books domain derived from the Amazon reviews corpus [18], Yelp dataset [19] and Movielens1m[20]. Conclusions are restricted to these datasets.

1.2 Statement of the Problem

In large-scale recommender systems typically implemented as multi-stage pipelines in which a fast retrieval component first narrows the item catalog to a limited candidate set, after which a ranking or reranking component produces the final top- K recommendations [2, 3, 15]. This architecture is necessary in practice because latency and computation constraints make full-catalog scoring with

expressive models infeasible, which means that the quality of retrieval directly determines which items are available for downstream ranking [2, 3]. In implicit-feedback settings, graph-based collaborative filtering methods such as LightGCN are widely used as efficient retrievers [5]. At the same time, explainable recommendation research and the broader XAI literature emphasize that explanations should be connected to the factors that actually influence model outputs and should be evaluated using faithfulness-oriented criteria rather than plausibility alone [9, 21].

Despite this progress, two unresolved methodological limitations remain in production-aligned two-stage recommendation under implicit feedback. The first is the lack of explicit accountability for the retrieval bottleneck. In a two-stage pipeline, the reranker can improve final ranking quality only when the relevant held-out item is present in the retrieved candidate set; if retrieval misses that item, no downstream reranking strategy can recover it [3, 15]. Consequently, end-to-end metrics such as $HR@K$ and $NDCG@K$ are constrained by candidate-set quality, and observed changes in these metrics may be incorrectly attributed to reranking even when they are primarily caused by retrieval success or failure [4, 15]. This creates a methodological need to evaluate candidate-set recall explicitly as the proportion of evaluated users whose held-out relevant item appears in the retrieved candidate set, and to interpret end-to-end ranking results in relation to that retrieval ceiling.

The second limitation is the absence of measurable and decision-linked explainability in many recommender evaluations. Existing methods often produce human-readable rationales, but evaluation frequently does not quantify how often those explanations apply to the actual delivered top- K recommendations, nor whether the claimed evidence is truly connected to the scoring function that determines the final ordering [9, 21]. Faithfulness-oriented evaluation in XAI recommends perturbation or masking tests in which the proposed evidence is

removed while other inputs are held fixed, and the change in model output is measured to determine whether the evidence is genuinely influential [9, 10]. The central problem addressed in this thesis is therefore the design and evaluation of a production-aligned two-stage recommender for implicit feedback in which retrieval and reranking are assessed under an explicit retrieval-bottleneck discipline, and explanation behavior is evaluated through both top- K explanation coverage and a fidelity-style evidence-masking procedure that tests whether symbolic evidence contributes to the final ranking decisions rather than being appended post hoc [9, 15, 21].

1.3 Research Questions

- RQ.1** What improvements in recommendation quality ($\text{HR}@K$, $\text{NDCG}@K$) are achieved by rule-based candidate expansion and an explainable MLP reranker compared to an encoder-only baseline under the same fixed candidate budget C in a two-stage retrieval→reranking pipeline?
- RQ.2** How does Candidate Recall@ C bound and explain end-to-end ranking performance ($\text{HR}@K$, $\text{NDCG}@K$), and how do changes in Candidate Recall@ C relate to observed changes in ranking quality across model variants?
- RQ.3** How does train-only symbolic evidence affect explanation coverage ($\text{EC}@K$), and what trade-offs in accuracy ($\text{HR}@K$, $\text{NDCG}@K$) and novelty ($\text{NOV}@K$) arise when switching from standard reranking to EFE-proxy reranking under the same retrieval and candidate construction?

1.4 Research Objectives

1.4.1 General Objective

The general objective is to design and evaluate a two-stage recommender for implicit feedback that combines LightGCN-style retrieval with train-only sym-

bolic expansion and an explainable reranker, while reporting results in a way that makes retrieval ceilings (Recall@ C) and explanation behavior (EC@ K and fidelity-drop) explicit.

1.4.2 Specific Objectives

- Train a LightGCN-style retriever on the user-item interaction graph and use it for top- C candidate generation under a fixed retrieval budget.
- Construct symbolic evidence from *training interactions only* and use it for candidate expansion and structured explanations (co-occurrence/association evidence), avoiding validation/test leakage.
- Build a union candidate set by combining retriever candidates with rule-expanded candidates under a strict cap (C_{union}), and quantify the resulting change in candidate Recall@ C .
- Train an explainable MLP reranker that re-scores candidates using compact, inspectable features (including rule-derived features) and evaluate HR@ K /NDCG@ K on the resulting ranked lists.
- Report candidate Recall@ C alongside end-to-end HR@ K /NDCG@ K so that retrieval ceilings are explicit and reranking gains are interpreted under the correct system-level constraint.
- Quantify explainability via EC@ K (explainability coverage evaluation) and validate faithfulness via a fidelity-style feature removal test that masks rule features at inference time and measures $\Delta\text{HR@}K/\Delta\text{NDCG@}K$.

1.5 Significance of the Study

From a systems perspective, multi-stage recommenders are standard because they make large-catalog recommendation computationally feasible, but they also in-

introduce a hard dependency: if the ground-truth item does not appear in the candidate set, reranking cannot recover it. [15, 22, 23] Our work is significant because report candidate Recall@ C together with end-to-end HR@ K /NDCG@ K , which makes this dependency measurable rather than implicit. [23]

From an explainability perspective, many recommendation explanations are evaluated qualitatively, yet broader XAI literature argues that explanations should be assessed for faithfulness (whether the cited evidence actually influences the decision). [21, 24, 25] I operationalize explainability in a way that can be audited: we report how often symbolic evidence applies (EC@ K) and test causal dependence through feature removal (fidelity-drop), so explanation behavior is tied to measurable changes in ranking outcomes. [25, 26]

1.6 Scope of the Study

We study implicit-feedback recommendation on the Amazon Books review, Yelp and Movielens1m datasets after converting ratings to implicit positives using preprocessing (e.g., `min_rating` = 4.0) and splitting interactions by time per user into train/validation/test. [27] the proposed pipeline is strictly two-stage: (i) a LightGCN retriever generates top- C_{base} candidates, (ii) a symbolic expansion step adds rule-based candidates mined from training histories only, and (iii) an MLP reranker produces the final top- K list from the capped union set.

The symbolic component we evaluate is rule-based co-occurrence evidence constructed from training histories (and used both for candidate expansion and reranker features). Although the broader motivation includes symbolic signals such as item attributes, explicit path features, or bandit-style selection, we do not include those components in the reported experiments unless stated and supported by results.

Recommendation quality evaluation is performed using HR@ K and NDCG@ K , and we evaluated system behavior using candidate Recall@ C , EC@ K , and a

fidelity-style rule-feature removal test.

1.7 Contributions

This thesis presents a hybrid two-stage explainable neuro-symbolic framework for recommendation under sparse implicit feedback. The framework integrates graph-based retrieval with train-only symbolic evidence to improve both *candidate quality* and the *interpretation of ranking decisions* under a fixed candidate budget. In contrast to approaches that report only final top- K accuracy, the evaluation protocol used in this thesis makes the retrieval bottleneck explicit and quantifies explanation behavior as a first-class outcome. The main contributions are summarized as follows:

- **Hybrid two-stage framework for explainable recommendation under sparse implicit feedback.** This thesis instantiates a production-aligned retrieval→reranking design with (i) LightGCN as an efficient stage-1 retriever for implicit feedback and (ii) an explainable MLP reranker that consumes both neural retrieval signals and symbolic evidence features. The resulting system supports standard ranking metrics ($\text{HR}@K$, $\text{NDCG}@K$) while enabling evidence-backed explanations for recommended items.
- **Train-only symbolic evidence channel for budgeted candidate expansion and reranking features.** Single-antecedent association rules of the form $x \rightarrow y$ are mined from training histories only and used in two roles: (i) expanding the stage-1 candidate pool under a strict cap (C), and (ii) generating interpretable reranker features (e.g., rule support/strength signals) that can be surfaced as explanations tied to inputs that influence ranking, rather than attached post-hoc.
- **Bottleneck-aware evaluation protocol with explicit candidate-set quality reporting.** Candidate Recall@ C is reported alongside end-to-end

$\text{HR@}K/\text{NDCG@}K$ in order to make the retrieval ceiling visible and to avoid misattributing failures to the reranker when the ground-truth item is not retrieved. This supports direct analysis of how changes in candidate construction affect downstream ranking outcomes under a fixed candidate budget.

- **Quantified explanation behavior through coverage and evidence-dependence diagnostics.** Explanation Coverage ($\text{EC@}K$) is reported as an operational metric of how often top- K recommendations are supported by train-only symbolic evidence. In addition, a fidelity-style evidence removal test is used as a diagnostic of evidence dependence by masking rule-derived features at inference time and measuring the resulting changes in $\text{HR@}K/\text{NDCG@}K$.
- **Empirical evaluation with consistent cross-stage reporting across datasets.** The framework is evaluated under a consistent two-stage protocol, candidate-budget discipline, and metric suite (Candidate Recall@ C , $\text{HR@}K$, $\text{NDCG@}K$, $\text{EC@}K$, and auxiliary analyses such as novelty where reported). This supports dataset-sensitive analysis of when symbolic evidence improves candidate quality, ranking accuracy, and explanation availability.

Chapter 2

Literature Review

2.1 Introduction and Chapter Roadmap

Recommender systems in large-scale settings are best understood as constrained decision pipelines rather than standalone prediction models. In practice, recommendation quality depends not only on the final ranking model, but also on candidate generation quality, the evidence available to later stages, and the evaluation protocol used to interpret system behavior. This is especially important in sparse implicit-feedback settings, where observations are positive-only signals, non-interactions are ambiguous, and retrieval failures can dominate end-to-end performance because downstream ranking cannot recover items that were never retrieved [2, 11–13].

This thesis adopts a pipeline-oriented perspective and studies a two-stage explainable recommender framework in which a graph-based collaborative retriever (LightGCN) generates an initial candidate set, a symbolic evidence channel expands candidates under an explicit budget, and an evidence-aware reranker produces the final top- K recommendations. A central premise is that explainability should be treated as an *evidence* problem rather than only a presentation problem: explanation signals should participate in candidate construction and/or scoring so that they can be evaluated as operational components of the recommendation process.

Accordingly, this chapter is organized around the concepts needed to support the methodology and evaluation used later in the thesis. It first reviews

the implicit-feedback and one-class recommendation setting, including evaluation choices that affect interpretation of offline top- K results. It then examines multi-stage recommendation pipelines, with emphasis on the retrieval bottleneck and the need to interpret $\text{HR}@K$ and $\text{NDCG}@K$ together with candidate $\text{Recall}@C$ under bounded candidate sets. Next, it reviews graph collaborative filtering as the stage-1 retrieval backbone, focusing on LightGCN and related variants relevant to sparse implicit-feedback retrieval.

The chapter then turns to explainability and evidence mechanisms. It reviews explainable recommendation from the perspective of intrinsic versus post-hoc explanation, faithfulness versus plausibility, and offline explanation evaluation. It then examines symbolic and association-rule evidence as a practical, interpretable signal source in settings where metadata or curated knowledge graphs are limited, and motivates explainable reranking with explicit evidence features. Finally, it summarizes the literature gap addressed by this thesis and positions the proposed HiX2S framework as a two-stage neuro-symbolic design that jointly evaluates retrieval quality, bounded-candidate ranking quality, explanation coverage, and evidence dependence.

2.1.1 Implicit feedback as MNAR data and exposure bias

A core difficulty in implicit-feedback recommendation is that missing interactions are not random. The observed user–item matrix is typically *missing-not-at-random* (MNAR) because interaction depends on exposure, interface position, timing, and prior platform decisions, not only on user preference [28–30]. As a result, unobserved items cannot be interpreted as clean negatives.

Let $R \in \{0, 1\}^{|U| \times |I|}$ denote the implicit interaction matrix:

$$R_{ui} = \begin{cases} 1, & \text{if user } u \text{ interacted with item } i, \\ 0, & \text{otherwise.} \end{cases} \quad (2.1)$$

Under MNAR, the observation process is driven by exposure. Let $E_{ui} \in \{0, 1\}$ denote whether item i was exposed to user u , and let $Y_{ui} \in \{0, 1\}$ denote the latent preference (relevance) outcome conditional on exposure. Then the probability of observing an interaction can be decomposed as

$$\mathbb{P}(R_{ui} = 1) = \mathbb{P}(E_{ui} = 1) \cdot \mathbb{P}(Y_{ui} = 1 \mid E_{ui} = 1). \quad (2.2)$$

This decomposition formalizes why non-interactions are ambiguous: a missing R_{ui} can arise from lack of exposure ($E_{ui} = 0$) rather than lack of preference. Implicit recommendation is therefore a positive-only learning problem with uncertain non-observations, motivating one-class collaborative filtering formulations [11, 12].

This also affects evaluation. Offline datasets reflect historical exposure policies, so a model may appear strong partly because it reproduces previous exposure patterns rather than discovering truly preferred items under alternative exposure conditions [28, 31]. In sparse settings, this problem becomes more severe because short histories and long-tail items amplify uncertainty about whether missing data implies irrelevance.

For this thesis, the MNAR framing motivates stage-aware reporting that makes retrieval effects visible in addition to final ranking quality, and it motivates evidence-tracing explainability: explicit evidence channels (e.g., rule-derived co-occurrence signals) improve transparency about what the pipeline used, even though they do not remove exposure bias.

2.1.2 Negative sampling choices and their downstream effects

Because implicit-feedback datasets do not contain explicit negatives, most training objectives rely on sampled non-interactions. The negative-sampling strategy is therefore part of the problem definition: it determines which pairwise or list-wise comparisons the model is encouraged to separate and can materially affect

learned ranking behavior [13, 32].

A common approach is pairwise training with Bayesian Personalized Ranking (BPR), where each observed positive (u, i) is contrasted against a sampled unobserved item j . Let $\hat{y}_{ui}$ denote the predicted relevance score. The BPR objective is

$$\mathcal{L}_{\text{BPR}} = - \sum_{(u,i,j)} \log \sigma(\hat{y}_{ui} - \hat{y}_{uj}) + \lambda \|\Theta\|_2^2, \quad (2.3)$$

where $\sigma(\cdot)$ is the logistic sigmoid and Θ denotes model parameters [13]. Under this formulation, the sampling distribution over j directly shapes the gradients and the effective notion of “hardness” that the model learns.

Uniform sampling is simple and widely used, but in large catalogs it often yields easy negatives, encouraging coarse separation rather than fine discrimination among plausible alternatives. Popularity-aware or hard-negative sampling can create more informative comparisons and improve top- K discrimination, but it also changes the optimization emphasis by concentrating learning on a narrower region of the item space [32].

This issue is amplified in two-stage pipelines. At serving time, the reranker rarely compares a positive item against random catalog items; it compares among retrieval-conditioned candidates that are already plausible. If reranker training uses mostly random negatives, a training–serving mismatch can arise because the model is optimized on an easier contrast set than the one encountered at inference. For this reason, negative sampling is treated as a pipeline-level design choice whose role is to align training comparisons with the difficulty structure induced by retrieved candidate sets, so that end-to-end results and evidence-feature ablations remain interpretable within the intended two-stage recommendation setting.

2.1.3 Full-ranking vs. sampled-ranking evaluation

Offline top- K evaluation in implicit recommendation is commonly performed using either *full-ranking* or *sampled-ranking* protocols. In full-ranking evaluation, the held-out positive is ranked against the full item universe (after exclusions). In sampled-ranking evaluation, it is ranked against a fixed set of sampled negatives for computational efficiency [4]. These two protocols do not measure exactly the same quantity.

Sampled-ranking evaluation is attractive because it is fast and practical for repeated experiments, ablations, and hyperparameter tuning. However, the resulting $\text{HR}@K$ and $\text{NDCG}@K$ values depend on the number and difficulty of sampled negatives. Therefore, sampled metrics should be interpreted as performance under a *specific evaluation task*, not as direct substitutes for full-catalog ranking quality [4].

This sensitivity also limits cross-paper comparability. Two studies may report the same metric name while using different negative-pool sizes, sampling distributions, filtering rules, or split protocols. In such cases, the numerical values are not directly comparable unless the evaluation protocol is aligned.

In this thesis, I use sampled-ranking evaluation (including the 99-negative setting) as a computationally practical and internally consistent protocol for comparing variants under the same conditions. I do not treat sampled HR/NDCG as equivalent to full-ranking performance. Instead, I interpret these metrics jointly with candidate $\text{Recall}@C$ under fixed candidate budgets, so that ranking outcomes are read in the context of stage-1 candidate availability as well as stage-2 ordering quality.

2.2 Multi-Stage Recommendation Pipelines and the Retrieval Bottleneck

Large-scale recommender systems are typically implemented as multi-stage pipelines rather than single-stage full-catalog rankers. A fast candidate-generation stage first selects a manageable set of items, and a downstream ranking/reranking stage then refines that set under richer scoring logic [2, 14, 15]. This design is driven by latency and cost constraints, but it also changes how recommendation quality should be interpreted.

In a two-stage pipeline, the retrieval stage determines the feasible set available to the reranker. If a relevant item is not retrieved, no downstream model can recover it. Candidate quality is therefore a structural determinant of end-to-end performance rather than a secondary implementation detail.

A two-stage pipeline can be written as a composition of candidate retrieval and reranking. For user u , Stage 1 produces a candidate set under a budget C ,

$$\mathcal{C}_u = \text{Retrieve}_C(u), \quad (2.4)$$

and Stage 2 produces the final top- K ranking by scoring and selecting from that set,

$$\pi_u^{(K)} = \text{Rank}_K(u, \mathcal{C}_u). \quad (2.5)$$

This thesis adopts the pipeline perspective and analyzes retrieval quality, ranking performance, and explanation behavior under explicit candidate-budget constraints. Under this design, the retrieval bottleneck becomes a central analytical concept: stage-2 improvements can only be interpreted correctly when stage-1 candidate availability is reported and evaluated explicitly.

2.2.1 Candidate generation families in practice

Production recommender systems rarely rely on a single candidate generator. Instead, they often combine multiple generators because different methods cover different users, items, and failure modes [2, 14]. A generator that performs well for dense-history users may fail for sparse users; a generator optimized for collaborative relevance may underperform on cold or under-exposed items.

One major family is *embedding-based retrieval*, where users and items are represented in a shared vector space and candidates are retrieved by nearest-neighbor search. This family is attractive because it supports high-throughput retrieval and integrates naturally with collaborative learning and ANN infrastructure [33, 34].

A second family is *co-visitation / co-occurrence retrieval*, which generates candidates from historical interaction patterns (e.g., items frequently consumed together or in close sequence). These methods are often fast and practical, but they can inherit exposure and popularity bias directly from logs.

A third family is *rule-based or symbolic candidate expansion*, including association-rule-style generators mined from interaction histories. These generators are especially relevant because they provide explicit evidence that can support both candidate expansion and explanation [6, 16].

A fourth family is *content-based retrieval*, which relies on metadata, attributes, or text similarity. This family is useful for sparse and cold-start settings but depends strongly on metadata coverage and quality.

These families are complementary rather than mutually exclusive, which motivates multi-generator retrieval in practice. The stage-1 design studied in this thesis follows that logic by combining a collaborative retriever with symbolic rule-based expansion under an explicit candidate-budget discipline.

2.2.2 Candidate set construction and union policies

When multiple candidate generators are used, the candidate-construction policy becomes a core part of the pipeline design. This step is often under-reported, but it directly affects candidate Recall@C, reranking difficulty, and the interpretation of gains under a fixed budget.

The simplest policy is *set union* followed by truncation to a target budget. This is easy to implement and avoids requiring all generators to share a common score scale. However, when the union exceeds the budget, the truncation rule becomes the effective gatekeeper of which sources actually contribute to the final feasible set.

A common alternative is *weighted merging*, in which generator scores are combined before truncation. This can reward candidates supported by multiple sources, but it introduces score-calibration challenges because embedding scores, rule heuristics, and content similarities often have different scales and meanings.

Another option is *quota-based mixing or interleaving*, where each source contributes candidates according to a fixed quota or alternating policy. This can preserve source diversity and prevent one generator from dominating the budget, but it may reduce total candidate recall if weaker generators consume too many slots.

For fixed-budget ablations, the candidate-construction policy should support clean interpretation. A cap-after-union policy with explicit source budgeting is methodologically useful because it limits score-fusion confounding and makes the marginal contribution of each candidate source easier to attribute under the same candidate cap.

2.2.3 Training-serving mismatch in two-stage pipelines

In a two-stage recommender pipeline, the reranker is trained and deployed on items conditioned on retrieval. At serving time, it does not rank arbitrary catalog

items; it ranks a bounded candidate set produced by Stage 1 retrieval under a specific budget and union policy [2, 35]. This creates a distributional dependency between retrieval and reranking.

If reranker training uses negatives drawn from a very different distribution (e.g., mostly random catalog negatives), a training-serving mismatch can occur. Random negatives are often easier than retrieval-conditioned candidates, so the reranker may learn broad relevance cues but underperform when it must discriminate among plausible alternatives that survive retrieval.

For this reason, reranking is treated as a candidate-conditioned learning problem. Training and evaluation are aligned with the candidate distributions induced by the same retrieval process and the same candidate-budget discipline, which supports fair comparison across variants and more defensible attribution of ranking gains.

2.2.4 Candidate recall and the retrieval ceiling

In a two-stage pipeline, candidate recall is a direct diagnostic of whether the relevant item enters the feasible set available to the reranker. Let $\mathcal{C}_u$ denote the retrieved candidate set for user u under budget C , and let i_u^* denote the held-out relevant item. Candidate Recall@ C is defined as

$$\text{Recall@}C = \frac{1}{|U|} \sum_{u \in U} \mathbb{I}(i_u^* \in \mathcal{C}_u). \quad (2.6)$$

Because the final top- K list is selected from $\mathcal{C}_u$, hit-based ranking metrics are upper-bounded by candidate recall in the single-held-out-item setting:

$$\text{HR@}K \leq \text{Recall@}C \quad (K \leq C). \quad (2.7)$$

A similar interpretation applies to NDCG@ K : when the held-out item is absent from the candidate set, its contribution is zero.

This retrieval-ceiling perspective shapes the evaluation discipline adopted throughout the thesis: candidate recall is treated as a first-class metric, and changes in top- K ranking outcomes are interpreted in relation to the retrieval bottleneck under fixed candidate-budget conditions.

2.3 Graph Collaborative Filtering for Stage-1 Retrieval

Candidate generation in implicit-feedback recommendation is commonly designed to maximize coverage of relevant items under strict efficiency constraints. In this stage, retrieval methods are valued not only for predictive accuracy but also for throughput, scalability, and compatibility with bounded-candidate downstream ranking. Classical collaborative filtering methods based on latent factors remain an important foundation because they learn user and item representations directly from interactions and support efficient similarity-based retrieval [13, 36]. Their main limitation in sparse settings is that they may underutilize higher-order connectivity structure in the interaction graph, especially when user histories are short or item exposure is highly uneven.

Graph collaborative filtering addresses this limitation by modeling recommendation on the user-item bipartite graph and propagating collaborative signal through neighborhood structure [5, 37]. In this view, recommendation is not only a matrix reconstruction or pairwise ranking problem; it is also a representation-learning problem on a sparse graph where multi-hop connectivity carries useful preference information. Early graph-CF models such as NGCF introduced nonlinear transformations and feature interactions during propagation, improving expressive capacity but also increasing computational complexity and tuning burden [37]. This trade-off is especially important in a retrieval stage, where stable high-recall behavior and operational simplicity are often more valuable than maximal architectural complexity.

LightGCN is particularly relevant because it simplifies graph collaborative

filtering by removing feature transformation and nonlinear activation from graph propagation, retaining only neighborhood aggregation and layer-wise embedding combination [5]. Let $\mathcal{N}(u)$ denote the item neighbors of user u and $\mathcal{N}(i)$ denote the user neighbors of item i in the bipartite interaction graph. LightGCN updates user embeddings by normalized neighborhood aggregation:

$$\mathbf{e}_u^{(k+1)} = \sum_{i \in \mathcal{N}(u)} \frac{1}{\sqrt{|\mathcal{N}(u)| |\mathcal{N}(i)|}} \mathbf{e}_i^{(k)}, \quad (2.8)$$

with an analogous update for item embeddings:

$$\mathbf{e}_i^{(k+1)} = \sum_{u \in \mathcal{N}(i)} \frac{1}{\sqrt{|\mathcal{N}(i)| |\mathcal{N}(u)|}} \mathbf{e}_u^{(k)}. \quad (2.9)$$

Final representations are computed by combining layer-wise embeddings:

$$\mathbf{e}_u = \sum_{k=0}^K \alpha_k \mathbf{e}_u^{(k)}, \quad \mathbf{e}_i = \sum_{k=0}^K \alpha_k \mathbf{e}_i^{(k)}, \quad (2.10)$$

where α_k are fixed or learned layer weights. A standard preference score is then given by the inner product:

$$\hat{y}_{ui} = \mathbf{e}_u^\top \mathbf{e}_i. \quad (2.11)$$

This design preserves the ability to capture high-order collaborative signals while keeping inference compatible with efficient vector-similarity retrieval. In a multi-stage pipeline, this makes LightGCN a strong stage-1 backbone: it is collaborative, sparse-data compatible, and operationally aligned with candidate-budgeted retrieval.

The broader graph-CF retrieval literature also includes efficiency-oriented and robustness-oriented variants relevant for stage-1 baselines. UltraGCN reduces reliance on explicit multi-layer propagation by encoding graph constraints into the optimization objective, aiming to preserve graph-informed recommendation

quality while improving efficiency [38]. Self-supervised graph-CF variants such as SGL, LightGCL, and contrastive extensions improve representation robustness under sparsity and noise through augmentation or contrastive regularization [39–41]. These methods indicate that graph-CF remains a strong and evolving family for implicit-feedback retrieval, even when the research emphasis is accuracy or robustness rather than explainability.

At the same time, stage-1 selection in this thesis is determined not only by leaderboard performance, but by how well the retriever supports a production-aligned two-stage design under explicit candidate budgets and how clearly retrieval gains can be interpreted in relation to reranking and evidence features. In that context, graph-CF methods—and LightGCN in particular—offer a favorable balance between retrieval quality, implementation simplicity, and compatibility with candidate-ceiling analysis (Recall@ C versus downstream HR/NDCG), which is central to the evaluation discipline adopted in this thesis.

2.3.1 Why not sequential models as the main retriever

A natural alternative for stage-1 retrieval in implicit-feedback recommendation is a sequential recommender, since many recommendation tasks are temporally ordered and next-item prediction can benefit from modeling recency, order, and short-term intent. Recurrent and self-attention-based models such as GRU4Rec, SASRec, and BERT4Rec have demonstrated strong performance in sequential recommendation benchmarks, particularly when user behavior exhibits meaningful temporal patterns and adequate interaction history length [42–44]. These models are therefore important comparators in the broader retrieval literature.

However, strong sequential ranking performance does not automatically imply that a sequential model is the best choice for *main stage-1 retrieval* under the constraints used in this thesis. First, many sequential models are computationally heavier at serving time than graph-CF retrievers when deployed as

high-throughput candidate generators. In a production-aligned pipeline, stage-1 retrieval is often required to generate candidates for many users under tight latency budgets, and this favors retrievers that can be implemented as approximate nearest-neighbor search over relatively stable user/item embeddings. Graph-CF methods align well with this requirement because they naturally produce embedding spaces suitable for efficient similarity retrieval, whereas autoregressive or deep attention-based sequential models are more often used as rankers or rerankers over smaller candidate sets unless additional indexing, distillation, or approximation layers are introduced.

Second, the data regime emphasized in this thesis is sparse implicit feedback, where many users have short or irregular histories. Sequential models can be highly effective when order effects are strong and histories are sufficiently long, but their advantages may diminish in sparse settings where sequence signal is weak relative to collaborative neighborhood signal. In such cases, a graph-CF retriever can provide stronger stage-1 coverage because it aggregates information through the user-item graph and can leverage higher-order collaborative connectivity even when an individual user sequence is short. This property is especially useful when the evaluation objective prioritizes candidate recall under a fixed budget.

Third, graph-CF retrieval provides a stable collaborative backbone from which the marginal effect of symbolic expansion on $\text{Recall}@C$ can be measured and downstream ranking gains can be interpreted under fixed candidates. Using a more complex sequential retriever as the primary stage-1 model can reduce attribution clarity because gains or failures may become entangled with temporal modeling choices, sequence truncation policies, or serving approximations.

This does not imply that sequential recommenders are inferior in general, nor that they are unsuitable for multi-stage pipelines. Rather, under bounded candidate generation, sparse implicit data, and evidence-centered interpretability analysis, a graph-CF retriever is a better stage-1 fit for the thesis than a heavier

sequential model. Sequential models remain relevant as potential rerankers or future extensions, especially in settings where stronger temporal intent modeling is required.

2.3.2 Known weaknesses of graph collaborative filtering under sparsity

Although graph collaborative filtering is a strong stage-1 retrieval family, it has known weaknesses relevant to the motivations of this thesis. The first is persistent difficulty with cold-start and near-cold-start regimes. Graph propagation can improve representation quality by diffusing signal over neighborhoods, but it cannot create reliable collaborative evidence for users or items with extremely limited interaction history. When nodes are weakly connected, graph-based embeddings may remain under-informed, which can reduce candidate recall for long-tail items or users with short histories.

A second weakness is popularity amplification. In sparse implicit graphs, frequent items are connected to many users and therefore accumulate strong collaborative signal during propagation. This can improve overall accuracy, but it may also bias candidate generation toward head items and reduce exposure opportunities for less frequent but relevant items. Self-supervised graph methods partially address robustness and representation collapse concerns, but they do not by themselves guarantee transparent control over popularity-driven retrieval behavior [39, 40].

A third weakness, particularly important for explainable recommendation, is the opacity of purely collaborative evidence. Even when graph-CF retrieves relevant candidates effectively, the retrieval score is typically a latent similarity value in embedding space. This is operationally useful but difficult to audit in human-readable terms. As a result, graph-CF alone provides limited support for explanation generation, retrieval-stage debugging, or evidence auditing.

These weaknesses motivate a complementary symbolic channel developed later in the literature review. Association-rule or co-occurrence-based evidence augments graph-CF retrieval by introducing explicit, inspectable signals that can (i) expand candidate sets under a controlled budget, (ii) provide human-readable evidence fragments linked to user history, and (iii) be reused as explicit reranker features for faithfulness-oriented diagnostics. The literature therefore supports a balanced stage-1 conclusion: graph-CF methods such as LightGCN remain strong retrieval backbones because they are efficient, scalable, and effective under sparse implicit feedback, while their limitations in cold-ish regimes, popularity concentration, and lack of explicit evidence channels justify the addition of a symbolic evidence pathway in a two-stage explainable pipeline.

2.4 Explainable Recommendation: Mechanisms, Forms, and Evaluation

Explainability in recommender systems is motivated by both user-facing and system-facing needs. From the user perspective, explanations can improve transparency, trust, satisfaction, and decision support by clarifying why a particular item is being recommended [16, 45]. From the system perspective, explanations provide a practical interface for debugging model behavior, identifying spurious correlations, and understanding failure modes that are not visible from aggregate accuracy metrics alone [46, 47]. These motivations are especially important in implicit-feedback recommendation, where observed interactions are ambiguous and the model must infer preferences from incomplete behavioral traces.

The explainable recommendation literature commonly distinguishes between intrinsic and post-hoc explanations, as well as model-specific and model-agnostic explanation methods [16]. Intrinsic approaches are designed so that interpretability is part of the model structure itself, while post-hoc approaches attempt to explain predictions after training. Model-agnostic methods such as LIME and

SHAP can be valuable for local inspection, but their application to recommender systems is often constrained by sparse high-dimensional inputs, latent representations, and multi-stage pipeline dependencies that make perturbations difficult to interpret operationally [8, 48]. By contrast, model-specific recommendation explanations can exploit structured signals such as user history, item attributes, or graph relations, but they vary substantially in how closely those signals are tied to the actual scoring function [16].

At the level of explanation form, recommender systems frequently present explanations as feature-based statements, example-based statements, or rule-/path-based reasoning chains [46]. Feature-based explanations refer to attributes such as category or genre alignment, example-based explanations refer to prior user interactions or similar items, and rule-/path-based explanations present a more explicit reasoning chain linking historical evidence to the recommended item.

2.4.1 Explanation objectives: user-facing vs developer-facing

A useful way to deepen the evaluation perspective is to distinguish explanation objectives by stakeholder. In the recommender-systems literature, user-facing objectives typically emphasize transparency, trust, satisfaction, persuasion quality, and decision support [16, 45]. In this setting, an explanation is often judged by whether it helps a user understand a recommendation, increases confidence in the system, or supports better choices. These goals are important and well-established, but they do not fully capture the needs of system development and auditing.

Developer-facing objectives are different. Here, explanations are valuable because they expose the signals that influenced the model, reveal failure modes, and support diagnosis when aggregate metrics degrade or shift across cohorts [46, 47]. In practical terms, a developer-facing explanation is less about persuasive language and more about inspectable evidence: which historical interactions, rules,

attributes, or graph relations contributed to an item being retrieved or ranked highly. This distinction matters in implicit-feedback recommendation because the data itself is noisy and incomplete. When non-interactions are ambiguous, debugging requires more than observing that a score is high or low; it requires tracing which evidence channels were active and whether they are plausible under the data-generation process.

This user-facing versus developer-facing split is especially relevant in a two-stage pipeline. In a multi-stage system, recommendation behavior emerges from the interaction of candidate generation and reranking, not from a single monolithic predictor. A user-facing explanation attached only to the final ranked item may appear coherent while hiding the fact that the item entered the candidate set through a heuristic shortcut, popularity effect, or spurious correlation. Conversely, a developer-facing explanation can help determine whether the retrieval stage failed to include the relevant item at all, whether symbolic expansion introduced useful candidates, or whether the reranker relied on explicit evidence features versus latent scores.

This framing motivates the emphasis on *auditable evidence channels*. By auditable evidence channels, I mean signals that are both (i) operationally used in candidate construction or reranking and (ii) inspectable in human-readable form, such as rule-derived co-occurrence evidence, history-linked support, or interpretable feature indicators. This is a narrower and more defensible objective than generic explanation generation. It does not claim to fully capture human reasoning or user intent. Instead, it aims to ensure that at least part of the recommendation process is grounded in explicit evidence that can be traced, evaluated for coverage, and tested for behavioral influence under controlled ablations.

The literature supports this emphasis. Survey work on explainable recommendation repeatedly notes that explanations may serve multiple functions and should be evaluated according to the intended function [16, 45]. When the ob-

jective is system understanding and model accountability rather than only user persuasion, explanation mechanisms that participate in scoring and can be stress-tested are methodologically stronger than detached narrative templates. This observation informs us follow a design choice to preserve explicit symbolic evidence from retrieval/candidate expansion into reranking, rather than generating explanations solely after the final scores have been produced.

2.4.2 Faithfulness vs plausibility: what the literature warns about

A central concern across explainable AI and explainable recommendation is the distinction between plausibility and faithfulness. An explanation can be *plausible* in the sense that it appears coherent, intuitive, or persuasive to a human observer, yet still be weakly connected to the model mechanism that actually produced the decision [9, 49]. This concern is not merely philosophical; it affects how explanation claims should be interpreted in empirical recommendation studies.

The literature repeatedly warns against treating convenient explanatory artifacts as direct evidence of causation. In particular, attention weights, saliency patterns, or post-hoc attributions may correlate with model behavior without constituting a faithful account of why a specific prediction was made [9, 50]. In recommender systems, this risk is amplified by sparse inputs, latent embeddings, and pipeline decomposition. A feature-level attribution on the final reranker score may look convincing even if the decisive constraint was upstream candidate omission, score clipping, or a retrieval heuristic that the explanation does not expose. In other words, multi-stage settings introduce additional opportunities for explanation mismatch: the explanation can be locally plausible at the reranking step while still failing to describe the pipeline-level decision process.

This issue is particularly important for post-hoc methods in two-stage recommendation. Model-agnostic explainers such as LIME and SHAP are useful for local sensitivity analysis, but their perturbation assumptions can become difficult

to interpret when the model being “explained” is actually a pipeline conditioned on a retrieved candidate set [8, 48]. Perturbing features of a reranker input while holding the candidate set fixed may ignore the retrieval process that made the item rankable in the first place. Conversely, perturbing the candidate set itself changes the task definition and can blur whether the explanation targets retrieval, reranking, or both. This does not make post-hoc analysis invalid; rather, it means the scope of the explanation must be stated explicitly and the resulting claims should remain modest.

For this reason, the present thesis emphasizes faithfulness-oriented *behavioral proxies* rather than relying only on plausible explanation examples. A practical proxy is a masking or removal test: if a proposed explanation channel (e.g., rule-derived evidence features) genuinely participates in scoring, then removing or masking that channel at inference time should change ranking outcomes in a measurable way under otherwise fixed conditions [9, 16].

At the same time, masking tests have important limitations and should be interpreted carefully. First, a metric drop under feature removal demonstrates dependence, not necessarily semantic correctness: the model may rely on a signal for the wrong reasons (e.g., popularity-correlated rules) while still showing sensitivity to that signal. Second, the magnitude of the drop depends on model capacity, feature redundancy, and compensatory channels; a small drop may indicate weak usage, but it may also mean the information is duplicated elsewhere in the feature set. Third, masking tests are typically offline diagnostics and do not measure whether users find explanations understandable or useful. Therefore, they are best treated as a faithfulness-oriented proxy within a broader evaluation framework, not as a complete explanation-quality metric.

This balanced interpretation aligns with the broader literature on explanation evaluation, which recommends matching the evaluation method to the explanation objective and avoiding overclaiming from a single metric [9, 16, 45]. In the

context of this research, the goal is not to claim perfect causal interpretability. The goal is to establish a stronger link between explanation signals and model behavior than would be available from post-hoc narrative generation alone, while remaining transparent about the limits of offline faithfulness proxies.

Taken together, the explainable recommendation literature supports an evidence-centered evaluation perspective for this thesis. Explanations can serve user-facing communication goals and developer-facing auditing goals, but these goals should not be conflated. Likewise, plausible explanations should not be treated as faithful by default, especially in multi-stage pipelines where upstream and downstream decisions interact. These concerns motivate the design choice to use explicit evidence channels in candidate expansion and reranking, and to evaluate them using measurable properties such as explanation coverage and fidelity-style masking diagnostics alongside ranking accuracy.

Evaluation of explanations therefore requires more than qualitative examples. User studies remain the strongest approach for measuring perceived usefulness, trust, and satisfaction, but they are often expensive and outside the practical scope of a thesis-scale experimental pipeline [45]. Offline evaluation complements user studies by measuring operational properties such as explanation coverage (how often an explanation can be provided) and faithfulness-oriented diagnostics (whether removing an explanation signal changes model behavior in a consistent way) [9, 16]. These criteria emphasize measurable explanation availability and evidence-dependence alongside ranking accuracy under explicit candidate-budget constraints.

2.5 Symbolic and Association-Rule Evidence for Recommendation

Symbolic evidence mechanisms complement purely latent recommendation models by exposing explicit, human-readable relations that can be inspected, audited,

and reused for explanation. Among these mechanisms, association rules are particularly relevant in sparse implicit-feedback settings because they can be mined directly from interaction histories without requiring dense metadata or a curated knowledge graph [6]. In recommender settings, such rules capture co-occurrence regularities (e.g., items frequently consumed together), providing an interpretable evidence channel grounded in observed behavior.

The literature supports two main uses of association-rule evidence in recommendation. First, rules can support *candidate expansion* by adding plausible items that may be missed by a purely collaborative embedding retriever. Second, the same rules can support *explanation* by linking recommended items to evidence in the user’s history [16, 51]. This dual role is especially useful in a two-stage pipeline because it connects retrieval and explanation: evidence that helps an item enter the candidate set can also help explain its appearance in the final ranked list.

Association-rule evidence is compatible with explicit candidate budgets. Rule-based expansion can be capped and merged with collaborative candidates under a fixed retrieval budget, allowing symbolic evidence to be evaluated without confounding gains through larger candidate pools or higher compute. This budgeted perspective is essential because downstream $\text{HR@}K/\text{NDCG@}K$ must be interpreted relative to the candidate ceiling established by stage-1 retrieval (Section 2.2).

At the same time, association rules require disciplined use. Co-occurrence patterns may reflect popularity and exposure effects rather than stable preference structure, and rules mined from sparse data may be noisy or overly specific. Moreover, symbolic evidence is heuristic rather than causal: a rule may be operationally useful for retrieval and explanation without being a definitive account of user intent [9, 51].

The broader explainable and neuro-symbolic recommendation literature in-

cludes knowledge-graph and path-based approaches that provide richer relational semantics than simple co-occurrence rules [16, 52]. These approaches are conceptually important, but they often require substantial side information, entity linking quality, and additional engineering complexity. Association-rule evidence offers a pragmatic alternative for sparse implicit-feedback datasets: it is easy to derive from interaction logs, transparent enough for explanation, and compatible with a production-aligned two-stage pipeline.

2.5.1 Rule mining choices and quality control

The usefulness of association-rule evidence depends strongly on how rules are mined and filtered. In practice, rule mining is a quality-control problem: threshold and pruning choices determine whether the rule set behaves as a helpful evidence channel or as a noisy expansion mechanism. Classical frameworks such as Apriori provide the key quantities for this control, especially support and confidence [6]. For itemsets A and B , confidence is defined as

$$\text{conf}(A \rightarrow B) = \frac{\text{support}(A \cup B)}{\text{support}(A)}. \quad (2.12)$$

In recommender usage, support and confidence thresholds induce an immediate coverage–precision trade-off. Low thresholds retain many weak or accidental co-occurrences, increasing candidate pressure and noise. High thresholds produce a smaller rule set dominated by very frequent patterns, which may reduce the incremental value of symbolic expansion beyond what collaborative retrieval already captures. In a budgeted two-stage pipeline, the goal is therefore not to maximize the number of rules, but to retain a bounded subset that improves candidate recall and provides interpretable evidence without flooding the candidate pool with low-quality consequents.

A second issue is rule redundancy. Transactional data often yields near-

duplicate rules, overlapping antecedents, or repeated consequents concentrated on high-frequency items. If left unfiltered, such redundancy can distort candidate expansion and make explanations repetitive rather than informative. Practical systems therefore apply pruning or consolidation (e.g., limiting consequents per antecedent, keeping stronger variants among overlapping rules, or deduplicating at candidate-construction time). The exact policy is implementation-specific, but the methodological principle is stable: symbolic evidence should be bounded and curated so that it functions as a controlled signal channel.

Rule mining choices also determine how rule evidence is converted into a user–item signal for candidate expansion. Given a user history H_u , a common scoring form aggregates rule confidence contributions from historical items $j \in H_u$ to a candidate item i :

$$s_{\text{rule}}(u, i) = \sum_{j \in H_u} \text{conf}(j \rightarrow i). \quad (2.13)$$

This produces an interpretable expansion score: each term corresponds to a specific history item j that supports recommending i through a mined rule.

Quality control also requires leakage prevention. Because rules are mined from interaction logs, mining on the full dataset can leak validation/test information into candidate generation and reranker features. A leakage-safe protocol therefore requires *train-only* rule mining: the rule base is constructed strictly from the training split, and validation/test candidate expansion uses only those training-derived rules. This preserves the temporal meaning of evaluation and ensures that gains from rule expansion reflect generalizable co-occurrence structure rather than future-data contamination.

Finally, rule mining choices interact with candidate-set construction policies (Section 2.2). Even a high-quality rule set can appear ineffective if expansion is merged naively with collaborative candidates or if capping policies systematically remove rule-derived candidates before reranking. The value of rules must

therefore be interpreted at the pipeline level: mining thresholds and pruning determine rule quality, while merge and cap policies determine whether that quality survives into the final candidate set.

2.5.2 Bias and failure modes of rule evidence

Although association rules are transparent and operationally useful, they inherit the biases and ambiguities of the interaction logs from which they are mined. A common failure mode is popularity-driven rule formation. In sparse implicit-feedback data, highly exposed and highly interacted items appear in many transactions and therefore become frequent consequents for many antecedents. As a result, the rule channel may overproduce head items even when the apparent association is driven more by exposure concentration than by preference-specific structure [17, 53]. In a budgeted two-stage pipeline, this is costly because popularity-heavy rule expansion can consume slots that could otherwise support more personalized or long-tail candidates.

A second failure mode is spurious co-consumption. Co-occurrence in implicit logs does not necessarily imply semantic similarity, substitutability, or stable preference continuity. Items may co-occur due to promotions, bundles, interface placement, seasonality, or session artifacts that are not explicitly represented in the data. Consequently, rule evidence can be operationally predictive while remaining semantically weak or unstable over time. For explainability, this creates a specific risk: a rule-based explanation may look precise because it cites user history, while the underlying co-occurrence pattern may not represent the true reason a user would prefer the item. This is why rule evidence should be treated as *heuristic* rather than causal evidence [9, 51].

The heuristic nature of rule evidence is not itself a defect. Many effective recommendation signals (including latent collaborative similarity) are heuristic. The methodological question is whether the system design acknowledges this and

prevents a single heuristic channel from dominating ranking decisions without verification. In a two-stage design, rule signals can expand candidates and contribute explicit features, while final ranking is determined jointly with collaborative relevance and other evidence under supervised reranking. This makes rule evidence an informative channel rather than a proof of user intent.

A further failure mode is evidence over-interpretation in evaluation. Because rule evidence is explicit and human-readable, it is tempting to treat the mere presence of a rule as evidence of explanation quality. However, coverage alone does not show that symbolic evidence actually influenced ranking. For this reason, explanation coverage should be interpreted together with faithfulness-oriented diagnostics (e.g., channel masking) and ranking outcomes under fixed candidate budgets. The key question is not only whether a rule can be shown, but whether symbolic evidence participates in the decision process in a measurable way.

Overall, the literature supports association-rule evidence as a transparent, low-overhead, interaction-log-based signal that is especially useful when meta-data and knowledge-graph resources are limited. At the same time, rule quality depends on thresholding, pruning, and leakage-safe mining, and rule behavior is constrained by known biases in implicit interaction data. These considerations motivate controlled, budgeted, and reranker-mediated use of symbolic evidence in a two-stage explainable pipeline.

2.6 Explainable Reranking with Explicit Evidence Features

Reranking is the natural stage in a multi-stage recommender pipeline for integrating explainability-relevant evidence into the final decision process. In large-scale systems, retrieval is optimized for efficiency and broad candidate coverage, whereas reranking operates on a bounded candidate set and can therefore support richer feature computation and more expressive scoring [2, 35]. This stage-wise

separation is crucial because it provides a practical point at which explicit evidence can be incorporated without violating retrieval-time constraints.

A common limitation in explainable recommendation is that explanations are generated after ranking through templates or post-hoc attribution, which can weaken the link between the explanation and the variables that actually determined the final order [9, 16]. An evidence-aware reranking design addresses this by incorporating interpretable signals directly into the scoring stage. When collaborative relevance, symbolic rule evidence, and other interpretable consistency signals are represented as explicit reranker features, the same variables used for ranking can also be inspected as explanation channels. This makes explanation behavior more auditable and reduces reliance on detached post-hoc narratives.

In a candidate-conditioned pipeline, reranking becomes the stage where the system can improve ordering while preserving a traceable basis for why items move up or down in the final list. This supports stronger evaluation practice: evidence-feature masking can be applied under otherwise fixed conditions to test whether an evidence channel is operational in ranking rather than merely attached after the fact [9, 16]. Let $\phi_{\text{rule}}(u, i)$ denote the rule-derived evidence feature vector for user u and item i . A generic evidence-aware reranker can be written as

$$\hat{y}_{ui}^{(2)} = f_{\theta}(\mathbf{e}_u, \mathbf{e}_i, \phi_{\text{rule}}(u, i)), \quad (2.14)$$

where $\mathbf{e}_u$ and $\mathbf{e}_i$ are the stage-1 embeddings (or embedding-derived features) and f_{θ} is an expressive scoring function (e.g., an MLP). This formulation makes the evidence channel explicit in the decision rule and therefore available for feature-level auditing.

A second issue is training-serving consistency. The reranker is deployed on candidates produced by a specific retrieval process, under a specific budget and candidate-construction policy. If reranker training uses negatives or candidate

distributions that differ substantially from those seen at evaluation or deployment, the measured contribution of evidence features may be misleading [2, 15]. Explainable reranking should therefore be analyzed as a stage conditioned on retrieval design rather than as an isolated ranking model. This requirement is central to multi-stage evaluation discipline, where retrieval construction, candidate budgeting, and reranking outcomes are reported jointly.

2.6.1 Learning-to-Rank Framing for Reranking under Candidate Conditioning

The reranking stage in a two-stage recommender pipeline is naturally framed as a learning-to-rank (LTR) problem, but with an important restriction: the reranker orders items within a candidate set already produced by stage-1 retrieval, rather than over the full catalog [2, 35]. This makes reranking a *candidate-conditioned* ranking task and changes both the training distribution and the interpretation of reranking gains.

Classical LTR methods are commonly grouped into pointwise, pairwise, and listwise formulations [35]. In recommender reranking, pointwise methods are simple and feature-friendly but may underemphasize relative competition among candidates; pairwise methods align well with implicit feedback because they optimize comparative preferences (ranking positives above competing items) [13]; and listwise methods are more directly tied to top- K objectives but are often more complex and sensitive to candidate-set construction. In practice, the preferred formulation depends on stability, feature design, and alignment with the actual stage-1 candidate distribution.

A key issue is training-serving mismatch. If the reranker is trained with random catalog negatives, many negatives are too easy and unlike the plausible items it will face after retrieval. In deployment, however, the reranker must discriminate among hard candidates already selected by collaborative similarity

or related heuristics. This mismatch can weaken end-to-end performance and confound ablation interpretation, because apparent gains may reflect better distribution matching rather than inherently better reranking [2, 35].

Finally, evidence dependence can be probed through fixed-candidate masking diagnostics. Let $\text{HR@}K_{\text{full}}$ denote the hit ratio computed with the full feature set and $\text{HR@}K_{\text{masked}}$ denote the hit ratio after masking the rule-derived evidence features while keeping candidates fixed. The fidelity-style drop is defined as

$$\Delta\text{HR@}K = \text{HR@}K_{\text{full}} - \text{HR@}K_{\text{masked}}, \quad (2.15)$$

and analogously for NDCG:

$$\Delta\text{NDCG@}K = \text{NDCG@}K_{\text{full}} - \text{NDCG@}K_{\text{masked}}. \quad (2.16)$$

Under a fixed candidate set, positive values of $\Delta\text{HR@}K$ and $\Delta\text{NDCG@}K$ provide operational evidence that the masked channel contributes to ranking behavior under the tested perturbation, whereas near-zero values indicate limited sensitivity under that intervention.

Overall, the literature supports a clear principle: reranking should be treated as a candidate-conditioned LTR stage, trained and evaluated on distributions that reflect the retrieval process, and designed such that interpretable evidence is represented as explicit scoring inputs that can be audited through controlled feature-masking diagnostics.

2.7 Beyond-Accuracy Properties as Secondary, Controllable Outcomes

Top- K accuracy metrics such as $\text{HR@}K$ and $\text{NDCG@}K$ remain central in recommender-system evaluation, but they do not fully characterize ranked-list quality in prac-

tice. A model can achieve strong hit-based accuracy while still producing repetitive lists, concentrating exposure on a narrow subset of items, or under-serving potentially relevant alternatives [17, 53]. For this reason, the literature also considers complementary list-level properties such as diversity, novelty, and catalog utilization as additional signals of ranked-list behavior [53, 54].

In a two-stage pipeline, these properties are best treated as *secondary, controllable outcomes* rather than substitutes for retrieval quality. The retrieval stage defines the feasible item pool through candidate composition and budget constraints, while reranking is the natural control layer for shaping final-list behavior under additional objectives [2, 17]. This distinction matters methodologically: reranking can only trade off diversity or novelty *within* the retrieved candidate set and cannot compensate for missing relevant items caused by weak stage-1 coverage (Section 2.2).

When novelty is reported, a common operationalization is to measure how dissimilar recommended items are from a user’s recent interaction history in a semantic embedding space [54]. Let $\mathbf{z}_i$ denote a semantic embedding for item i (e.g., derived from text/metadata), H_u denote the user’s history, and π_u^K denote the top- K ranked list. A standard novelty score can be defined via average dissimilarity to history:

$$\text{NOV@}K = \frac{1}{|U|} \sum_{u \in U} \frac{1}{K} \sum_{i \in \pi_u^K} \left(1 - \max_{j \in H_u} \text{sim}(\mathbf{z}_i, \mathbf{z}_j) \right), \quad (2.17)$$

where $\text{sim}(\cdot, \cdot)$ is a similarity function (commonly cosine similarity) and the max-over-history term summarizes the closest historical neighbor for each recommended item. This form captures *relative novelty* with respect to what the user has already interacted with, rather than novelty defined only by global popularity.

In two-stage systems, such secondary metrics should be interpreted under

the same bounded-candidate discipline as accuracy and explainability metrics. Candidate recall constrains the achievable accuracy ceiling (Section 2.2), and candidate composition also constrains the attainable diversity/novelty range: if the candidate set is dominated by a narrow region of the item space, reranking has limited capacity to diversify the final list without harming relevance. As a result, beyond-accuracy reporting is most informative when accompanied by the core pipeline metrics ($\text{Recall}@C$, $\text{HR}@K$, $\text{NDCG}@K$) and explanation-oriented diagnostics, so that changes in list behavior are not conflated with changes in candidate availability.

A further reason for restrained claims is that beyond-accuracy metrics are sensitive to signal availability and metric design. Novelty and diversity measures often require item representations from metadata or text, and these signals can be incomplete or uneven across sparse implicit-feedback datasets [17, 53]. Therefore, beyond-accuracy results are treated as descriptive indicators of pipeline behavior under fixed candidate-budget conditions, rather than as primary optimization targets.

Overall, the literature supports a limited but useful role for beyond-accuracy measures in this thesis: they provide visibility into final-list composition, but they are interpreted as secondary outcomes shaped primarily at reranking time and bounded by the same retrieval constraints that govern end-to-end accuracy.

2.8 Summary of Gaps and Research Positioning

Paper	Cold-start	Long-tail	Expl	Scal	Epi	EC@K	Hybrid	Semantic	Pareto
YouTube two-stage recommender [2]	◦	◦	×	✓	×	×	×	◦	×
LightGCN [5]	×	✓	×	✓	×	×	×	×	×
UltraGCN [38]	×	✓	×	✓	×	×	×	×	×
SGL [39]	×	✓	×	✓	×	×	×	×	×
LightGCL [40]	×	✓	×	✓	×	×	×	×	×
SimDCL [41]	×	✓	×	✓	×	×	×	×	×
KGIN [55]	✓	✓	✓	◦	×	×	✓	✓	×
LOGGER [56]	◦	✓	✓	◦	×	×	✓	✓	×
Our work:Hybrid two-stage neuro-symbolic pipeline	×	◦	✓	✓	×	✓	✓	◦	◦

Table 2.1: Representative synthesis matrix using broad system properties. ✓ = explicit/central; ◦ = partial/tangential; × = not a primary focus.

The Related work shows that prior work has made strong but largely isolated advances along separate dimensions of the recommendation problem. Graph collaborative filtering models such as LightGCN, UltraGCN, SGL, LightGCL, and related contrastive variants substantially improve collaborative retrieval under sparse implicit feedback, yet they do not incorporate explanation mechanisms or evaluate evidence dependence. Industrial two-stage systems, exemplified by the YouTube architecture, establish scalable retrieval→reranking pipelines but do not explicitly analyze retrieval ceilings under fixed candidate budgets or integrate symbolic reasoning into ranking decisions. Knowledge-aware and explainable models such as KGIN and LOGGER move closer to hybrid neural-symbolic reasoning and generate interpretable relational signals, but they do not systematically evaluate explanation coverage (EC@K), nor do they test whether symbolic evidence is operational in scoring through fidelity-style diagnostics within a bounded two-stage framework.

The contribution of this thesis is positioned at the intersection of these strands. It retains a strong collaborative retrieval backbone, integrates train-only symbolic evidence into both candidate construction and reranking, and evaluates the

system under explicit candidate-budget constraints with retrieval-ceiling accountability (Recall@C), ranking metrics (HR@K, NDCG@K), explanation coverage (EC@K), and evidence-removal diagnostics. Rather than claiming superiority across all desirable properties, the work advances a disciplined evaluation perspective: retrieval quality, ranking performance, and explanation behavior are analyzed jointly within the same protocol. In doing so, it extends existing retrieval and explainability research by making the interaction between candidate availability, bounded reranking, and evidence-aware explanations empirically visible and measurable under realistic serving constraints.

Chapter 3

Methodology

This thesis adopts a **Design Science Research** methodology to guide the development and validation of the proposed HiX2S recommendation artifact [1]. The process is implemented as an iterative cycle that links problem framing, artifact design, leakage-safe protocol construction, evaluation, and refinement. Figure 3.1 summarizes the workflow and the feedback loop used when evaluation criteria are not met.

Attribution and scope Several components used in this chapter are established prior work rather than original contributions of this thesis: two-stage retrieval→reranking pipelines [2, 22], implicit-feedback recommendation [11, 12], graph collaborative filtering with LightGCN [5], association-rule mining [6, 7], and top- K evaluation metrics such as NDCG [57]. The contribution of the thesis lies in a **budgeted candidate-construction design** (encoder candidates plus train-only symbolic candidates under a fixed cap) and a **pipeline-aligned explainability evaluation** that jointly reports Candidate Recall@C, bounded-candidate HR/NDCG, explanation coverage (EC@K), and evidence-removal sensitivity (fidelity-drop) as an evidence-dependence diagnostic [9, 58].

3.1 Problem Setting and Notation

This section defines the recommendation setting and introduces the notation used throughout the thesis. The formulation targets sparse implicit-feedback recommendation under a two-stage retrieval→reranking pipeline, which is widely

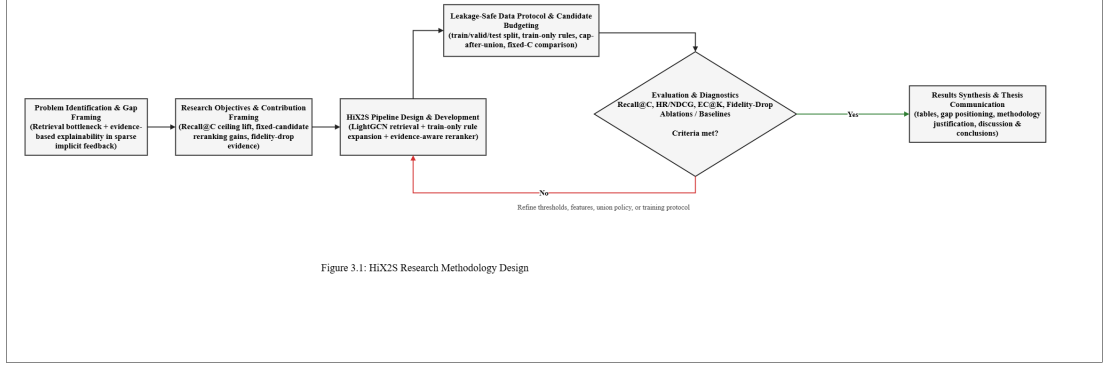


Figure 3.1: Research methodology design following a Design Science process [1].

adopted in large-scale recommendation because retrieval must be budgeted while reranking can be more expensive on a limited candidate set [2, 22, 58].

3.1.1 Implicit-Feedback Recommendation Setting

The setting is one-class implicit-feedback recommendation, where observed user–item interactions are treated as positive feedback and unobserved interactions are treated as missing feedback rather than explicit negative feedback [11, 12]. Let U denote the set of users and I denote the set of items. The observed positive interaction set is

$$\mathcal{R}^+ \subseteq U \times I. \quad (3.1)$$

For each user $u \in U$, the historical interaction set is

$$\mathcal{H}_u = \{i \in I \mid (u, i) \in \mathcal{R}^+\}. \quad (3.2)$$

Given a user u , the recommendation task is to rank unseen items in $I \setminus \mathcal{H}_u$ such that future relevant items appear as high as possible in the final ranked list.

3.1.2 Two-Stage Recommendation Formulation

The proposed framework follows a two-stage formulation. In Stage 1, a retriever generates a bounded candidate set for each user. In Stage 2, a reranker scores only those candidates to produce the final top- K recommendation list [2, 22].

Let $C_u \subseteq I \setminus \mathcal{H}_u$ denote the candidate set generated for user u , with maximum size C_{cap} . Let $s_{\text{enc}}(u, i)$ denote the Stage-1 retrieval score, and let $s_{\text{rr}}(u, i)$ denote the Stage-2 reranker score. The final recommendation list is:

$$\hat{R}_u^K = \text{TopK}_{i \in C_u} s_{\text{rr}}(u, i). \quad (3.3)$$

Retrieval bottleneck and candidate recall. If a held-out relevant item $i_u^* \notin C_u$, Stage 2 cannot recover it, regardless of reranker quality. Therefore, Stage 1 quality and Stage 2 quality are evaluated separately and interpreted jointly. Candidate Recall@C is used to quantify Stage 1 availability:

$$\text{Recall@C} = \frac{1}{|U|} \sum_{u \in U} \mathbb{I}(i_u^* \in C_u). \quad (3.4)$$

This stage-aware view is central to analyzing retrieval ceilings in bounded two-stage pipelines [58, 59].

3.2 Data Acquisition

Acquiring and preparing appropriate datasets is essential for evaluating HiX2S under a fair and reproducible implicit-feedback setting. Multiple domains are used (Amazon Books, MovieLens-1M, and Yelp) to assess robustness across different catalog sizes and interaction regimes. In datasets with explicit ratings, ratings are converted into implicit positives using a fixed threshold, consistent with standard implicit-feedback practice [11, 13]. Dataset-specific filtering thresholds and final statistics are reported in the Experiments chapter; this chapter focuses on the shared protocol and methodological controls used across datasets and baselines.

3.3 Data Preprocessing

A unified preprocessing and split protocol is applied to all datasets to ensure comparability across HiX2S and retrieval baselines. The preprocessing steps are designed for leakage-safe training, chronological evaluation, and fair top- K comparison under fixed candidate budgets.

3.3.1 Interaction Construction

For datasets with explicit ratings, interactions are binarized by retaining only ratings at or above a specified positive threshold. The remaining user-item pairs form the positive interaction set $\mathcal{R}^+$ used throughout training and evaluation, supporting one-class learning and pairwise ranking objectives [11, 13].

3.3.2 Chronological Leave-One-Out Protocol

To reduce temporal leakage and align evaluation with next-item recommendation behavior, a chronological leave-one-out protocol is used. For each user with sufficient interactions, interactions are ordered by timestamp and the most recent interactions are held out for validation and test [22].

Let the timestamp-ordered sequence for user u be $(i_{u,1}, i_{u,2}, \dots, i_{u,T_u})$. The split is defined as:

$$\mathcal{H}_u^{\text{train}} = \{i_{u,1}, \dots, i_{u,T_u-2}\}, \quad i_u^{\text{val}} = i_{u,T_u-1}, \quad i_u^{\text{test}} = i_{u,T_u}. \quad (3.5)$$

3.3.3 Training-Item Masking During Evaluation

During validation and test evaluation, items already observed in the user’s training history are masked to prevent the model from being rewarded for recommending previously consumed items:

$$I_u^{\text{eval}} = I \setminus \mathcal{H}_u^{\text{train}}. \quad (3.6)$$

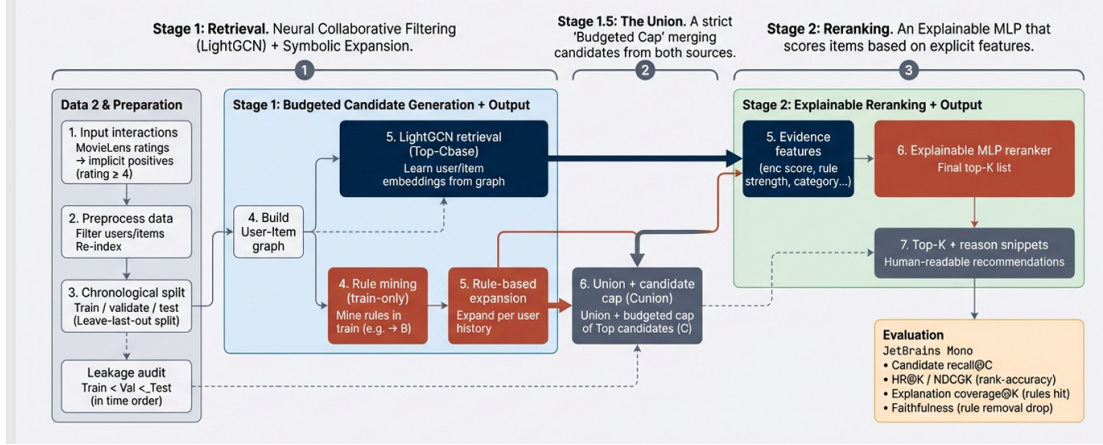


Figure 3.2: Overview of HiX2S: Stage 1 collaborative retrieval, train-only symbolic expansion, budgeted candidate union with a fixed cap, and Stage 2 evidence-aware reranking.

This masking rule is applied consistently across retrieval methods and across both stages of the pipeline.

3.3.4 Leakage-Safe Use of Symbolic Evidence

Because association-rule evidence is integrated into candidate expansion and reranking, leakage control is central. Association rules are mined using training interactions only; no validation or test interactions are used during rule extraction, scoring, pruning, or candidate expansion. This prevents future interactions from contaminating symbolic evidence and preserves the integrity of the chronological evaluation protocol [6, 7].

3.4 Overview of the Proposed Architecture

HiX2S is a two-stage **neuro-symbolic** recommendation architecture for sparse implicit-feedback settings. It combines a LightGCN-based collaborative retriever, train-only symbolic candidate expansion, and an evidence-aware reranker under a fixed candidate-budget constraint. The design addresses a key limitation of two-stage recommendation: a reranker cannot recover relevant items that are absent from the candidate set [2, 22].

Terminology (for consistency across the thesis). **Hybrid approach:**

“Hybrid” refers to neuro-symbolic integration: neural collaborative retrieval (LightGCN) combined with symbolic association-rule evidence mined from training histories. The symbolic channel participates in (i) candidate construction and (ii) reranker features.

Explainable: “Explainable” refers to evidence-aligned and testable explanations. Explanations are grounded in auditable evidence variables used by the pipeline, availability is measured by EC@K, and evidence dependence is tested by a fixed-candidate evidence-removal diagnostic (fidelity-drop), consistent with faithfulness-oriented XAI principles [9, 51].

3.5 Stage-1 Collaborative Retrieval with LightGCN

Stage 1 produces a high-recall candidate set under a fixed computational budget. LightGCN is adopted as the Stage 1 retriever because it is a strong and efficient graph collaborative filtering backbone that simplifies propagation by removing feature transforms and nonlinear activations [?]. Competing graph-based CF models (e.g., NGCF) are discussed as alternatives in the literature [?].

3.5.1 User–item graph construction

Stage 1 operates on a user–item bipartite graph constructed from training interactions only:

$$\mathcal{G} = (\mathcal{V}, \mathcal{E}), \quad \mathcal{V} = U \cup I, \quad \mathcal{E} = \{(u, i) \mid (u, i) \in \mathcal{R}_{\text{train}}^+\}. \quad (3.7)$$

Restricting graph construction to training interactions is required for leakage-safe evaluation.

3.5.2 LightGCN propagation

Let $\mathbf{e}_x^{(0)} \in \mathbb{R}^d$ denote the trainable initial embedding of node $x \in \mathcal{V}$. LightGCN propagates collaborative signal through normalized neighbor aggregation [?]:

$$\mathbf{e}_x^{(k+1)} = \sum_{y \in \mathcal{N}(x)} \frac{1}{\sqrt{|\mathcal{N}(x)|} \sqrt{|\mathcal{N}(y)|}} \mathbf{e}_y^{(k)}, \quad (3.8)$$

where $\mathcal{N}(x)$ denotes the neighborhood of node x . After L layers, the final embedding is computed as:

$$\mathbf{e}_x = \frac{1}{L+1} \sum_{k=0}^L \mathbf{e}_x^{(k)}. \quad (3.9)$$

The Stage 1 retrieval score is:

$$s_{\text{enc}}(u, i) = \mathbf{e}_u^\top \mathbf{e}_i. \quad (3.10)$$

3.5.3 Retriever training objective

The retriever is trained with Bayesian Personalized Ranking (BPR), a pairwise objective widely used in implicit-feedback recommendation [13]. For user u , positive $i \in \mathcal{H}_u^{\text{train}}$, and sampled negative $j \notin \mathcal{H}_u^{\text{train}}$:

$$\mathcal{L}_{\text{BPR}} = - \sum_{(u, i, j) \in \mathcal{D}} \log \sigma(s_{\text{enc}}(u, i) - s_{\text{enc}}(u, j)) + \lambda \|\Theta\|_2^2. \quad (3.11)$$

3.5.4 Base candidate retrieval

After training, a base candidate list is generated by scoring unseen items with $s_{\text{enc}}(u, i)$ and selecting the top- C_{base} :

$$C_u^{\text{base}} = \text{TopC}_{\text{base}} \left(\{s_{\text{enc}}(u, i) : i \in I \setminus \mathcal{H}_u^{\text{train}}\} \right). \quad (3.12)$$

C_{base} is held fixed within each comparison family to prevent confounding from unequal candidate budgets.

3.6 Train-only Symbolic Candidate Expansion

Symbolic expansion aims to improve Stage 1 candidate coverage under a fixed budget by introducing a complementary evidence channel derived from co-occurrence patterns in training histories. Unlike post-hoc explanation attachment, symbolic evidence participates directly in candidate construction, allowing its effect to be evaluated at the retrieval stage (Recall@C) and at the reranking stage (bounded-candidate HR/NDCG and evidence-removal tests).

3.6.1 Association-rule mining from training histories

Association rules are mined from users' training histories using classical association-rule frameworks [6, 7]. Each user's training interaction history is treated as a transaction. Rules have the form:

$$X \rightarrow y, \tag{3.13}$$

where $X \subset I$ is an antecedent itemset and $y \in I$ is a consequent item. Support/confidence-based pruning and implementation constraints (e.g., co-occurrence thresholds and per-antecedent caps) are applied. All statistics are computed on training transactions only.

3.6.2 Rule-based scoring and expansion

For a target user u , fired rules produce candidate consequents not in $\mathcal{H}_u^{\text{train}}$. Let $\mathcal{R}_u^{\text{fire}}(i)$ denote the fired rules that recommend item i . A rule evidence score aggregates fired rule weights:

$$s_{\text{rule}}(u, i) = \sum_{r \in \mathcal{R}_u^{\text{fire}}(i)} w(r). \tag{3.14}$$

Algorithm 1 Train-only symbolic rule extraction for HiX2S

Require: Training interaction matrix $\mathbf{R}_{\text{train}}$, minimum co-occurrence threshold τ_{cooc} , minimum confidence threshold τ_{conf} , per-antecedent rule cap L_{rule}
Ensure: Rule index $\mathcal{M}_{\text{rule}}$ mapping antecedent items to scored consequent items

- 1: Compute item supports $\text{supp}(i)$ from $\mathbf{R}_{\text{train}}$
- 2: Compute item–item co-occurrence matrix $\mathbf{C} \leftarrow \mathbf{R}_{\text{train}}^\top \mathbf{R}_{\text{train}}$
- 3: Set diagonal of $\mathbf{C}$ to zero
- 4: **for** each antecedent item $a \in I$ **do**
- 5: Initialize candidate rule list $\mathcal{L}_a \leftarrow \emptyset$
- 6: **for** each item $b \in I$, $b \neq a$, with $C_{ab} \geq \tau_{\text{cooc}}$ **do**
- 7: Compute confidence $\text{conf}(a \rightarrow b) \leftarrow C_{ab} / \max(1, \text{supp}(a))$
- 8: **if** $\text{conf}(a \rightarrow b) \geq \tau_{\text{conf}}$ **then**
- 9: Append $(b, \text{conf}(a \rightarrow b))$ to $\mathcal{L}_a$
- 10: **end if**
- 11: **end for**
- 12: Sort $\mathcal{L}_a$ by configured rule priority and keep top L_{rule}
- 13: Assign to $\mathcal{M}_{\text{rule}}[a]$
- 14: **end for**
- 15: **return** $\mathcal{M}_{\text{rule}}$

The symbolic candidate set is selected by:

$$C_u^{\text{rule}} = \text{TopC}_{\text{rule}} \left(\{s_{\text{rule}}(u, i) : i \notin \mathcal{H}_u^{\text{train}}\} \right). \quad (3.15)$$

3.6.3 Optional metadata/text expansion

When metadata is available, an optional auxiliary candidate channel can be enabled (e.g., lexical similarity via TF-IDF). When metadata is unavailable or the auxiliary channel is disabled, this component is set to the empty set [60]:

$$C_u^{\text{text}} = \emptyset. \quad (3.16)$$

3.6.4 Budgeted union and candidate capping

Candidates from available sources are merged and capped under a fixed budget:

$$C_u^{\text{union}} = C_u^{\text{base}} \cup C_u^{\text{rule}} \cup C_u^{\text{text}}, \quad C_u = \text{TopC}_{\text{cap}}(C_u^{\text{union}}), \quad |C_u| \leq C_{\text{cap}}. \quad (3.17)$$

Algorithm 2 HiX2S Stage-1 candidate generation (collaborative + symbolic + optional text)

Require: User u , training history $\mathcal{H}_u^{\text{train}}$, encoder scorer s_{enc} , rule index $\mathcal{M}_{\text{rule}}$, optional text module $\mathcal{M}_{\text{text}}$, budgets $(C_{\text{base}}, C_{\text{rule}}, C_{\text{text}}, C_{\text{cap}})$

Ensure: Final capped candidate set C_u

- 1: Retrieve C_u^{base} using $s_{\text{enc}}(u, i)$ and budget C_{base}
 - 2: Accumulate rule scores $s_{\text{rule}}(u, i)$ from fired rules over $\mathcal{H}_u^{\text{train}}$
 - 3: Select C_u^{rule} with budget C_{rule}
 - 4: **if** text/metadata channel enabled **then**
 - 5: Select C_u^{text} with budget C_{text}
 - 6: **else**
 - 7: Set $C_u^{\text{text}} \leftarrow \emptyset$
 - 8: **end if**
 - 9: Form union $C_u^{\text{union}} \leftarrow C_u^{\text{base}} \cup C_u^{\text{rule}} \cup C_u^{\text{text}}$
 - 10: Apply cap: $C_u \leftarrow \text{TopC}_{\text{cap}}(C_u^{\text{union}})$
 - 11: **return** C_u
-

This cap-after-union discipline is applied consistently across experiments and ablations to ensure that improvements reflect better candidate composition rather than larger candidate pools.

3.7 Stage-2 Explainable Reranking

Stage 2 reranks the bounded candidate set C_u using an evidence-aware scoring model. Reranking is the appropriate stage to combine evidence channels because scoring is performed on a limited slate rather than the full catalog, and explicit evidence features can be retained for audit and diagnostics [2, 51].

3.7.1 Reranker design rationale

The reranker is designed around explicit evidence channels so that ranking decisions can be examined at the feature level. This supports faithfulness-oriented diagnostics because the same evidence variables used for scoring can be selectively masked and re-evaluated under controlled conditions [9].

3.7.2 Feature construction

For each user–candidate pair (u, i) , a compact feature vector is constructed from explicit evidence channels:

$$\mathbf{x}(u, i) = [s_{\text{enc}}(u, i), s_{\text{rule}}(u, i)]. \quad (3.18)$$

When available and enabled, auxiliary interpretable metadata features are appended:

$$\mathbf{x}(u, i) = [s_{\text{enc}}(u, i), s_{\text{rule}}(u, i), \mathbf{a}(u, i)]. \quad (3.19)$$

3.7.3 Reranker model

A compact multilayer perceptron (MLP) is used as the reranker:

$$s_{\text{rr}}(u, i) = f_{\text{MLP}}(\mathbf{x}(u, i)). \quad (3.20)$$

3.7.4 Reranker training with candidate-conditioned slates

To reduce training–serving mismatch in two-stage pipelines, the reranker is trained on candidate-conditioned slates. Let $\mathcal{S}_u$ denote a training slate containing one positive item i_u^+ and sampled negatives. A listwise softmax defines:

$$p(i \mid u, \mathcal{S}_u) = \frac{\exp(s_{\text{rr}}(u, i))}{\sum_{j \in \mathcal{S}_u} \exp(s_{\text{rr}}(u, j))}. \quad (3.21)$$

A listwise cross-entropy objective is then used [61]:

$$\mathcal{L}_{\text{rr}} = - \sum_u \log p(i_u^+ \mid u, \mathcal{S}_u). \quad (3.22)$$

Algorithm 3 HiX2S Stage-2 explainable reranker training and inference

Require: Training users, Stage 1 candidate generator, feature builder $\phi(u, i)$, reranker f_{MLP} , epochs T

Ensure: Trained reranker and final reranked lists $\hat{R}_u^K$

```
1: Initialize reranker parameters
2: for epoch  $t = 1$  to  $T$  do
3:   for each training user  $u$  do
4:     Generate candidate-conditioned slate  $\mathcal{S}_u$ 
5:     Compute scores  $\{s_{\text{rr}}(u, i)\}_{i \in \mathcal{S}_u}$  from  $\mathbf{x}(u, i) \leftarrow \phi(u, i)$ 
6:     Update parameters using listwise cross-entropy loss
7:   end for
8: end for
9: for each evaluation user  $u$  do
10:  Obtain capped candidates  $C_u$ 
11:  Rank  $C_u$  by  $s_{\text{rr}}(u, i)$  and return  $\hat{R}_u^K$ 
12: end for
```

3.8 Explanation Construction and Evidence Reporting

Explanation construction is treated as evidence reporting grounded in operational pipeline signals rather than free-form post-hoc text generation. This design follows the distinction between plausible explanations and decision-aligned explanations in explainable ML [8, 9, 51].

3.8.1 Evidence-aligned explanation principle

Explanations are constructed only from evidence signals present in the retrieval pathway and/or the reranker feature space (e.g., fired rules and auxiliary interpretable features when enabled). If no eligible evidence is available, no explanation is fabricated. This constraint supports later dependence testing and reduces the risk of post-hoc rationalization [9].

3.8.2 Template-based explanation generation

Template-based explanations are instantiated from available evidence channels for each recommended item. Templates standardize outputs across datasets and ablations, enabling consistent measurement of explanation availability (EC@K).

Algorithm 4 HiX2S explanation construction and evidence reporting

Require: Final top- K list $\hat{R}_u^K$, user history $\mathcal{H}_u^{\text{train}}$, evidence records

Ensure: Explanation set and explainability indicators for EC@ K

- 1: **for** each recommended item $i \in \hat{R}_u^K$ **do**
 - 2: Collect eligible rule evidence (if any) and instantiate rule template(s)
 - 3: Collect eligible auxiliary evidence (if enabled) and instantiate template(s)
 - 4: Mark explainable(u, i) = 1 iff at least one statement is produced
 - 5: **end for**
-

3.8.3 Explanation coverage (EC@K)

Explanation Coverage at top- K (EC@K) measures explanation *availability*. Let $\mathbb{I}(\text{explainable}(u, i))$ indicate whether item i in the final list has at least one valid evidence statement. EC@K is:

$$\text{EC@K} = \frac{1}{|U|} \sum_{u \in U} \frac{1}{K} \sum_{i \in \hat{R}_u^K} \mathbb{I}(\text{explainable}(u, i)). \quad (3.23)$$

EC@K does not prove evidence influenced ranking; dependence is tested via evidence removal (next section).

3.9 Faithfulness Assessment via Evidence Removal

Faithfulness is operationalized as evidence dependence in ranking: if the reranker uses the symbolic evidence feature, masking that feature under fixed candidates should change ranking metrics [9]. Candidates are cached, the trained reranker is kept fixed, and the rule-evidence feature is replaced with zero at inference time:

$$\mathbf{x}(u, i) = [s_{\text{enc}}(u, i), s_{\text{rule}}(u, i), \mathbf{a}(u, i)], \quad \mathbf{x}_{\text{no-rule}}(u, i) = [s_{\text{enc}}(u, i), 0, \mathbf{a}(u, i)]. \quad (3.24)$$

Metric drops are defined as:

$$\Delta \text{HR@K} = \text{HR@K}_{\text{full}} - \text{HR@K}_{\text{no-rule}}, \quad \Delta \text{NDCG@K} = \text{NDCG@K}_{\text{full}} - \text{NDCG@K}_{\text{no-rule}}. \quad (3.25)$$

Algorithm 5 Faithfulness assessment via evidence removal (fidelity-drop test)

Require: Trained reranker, fixed candidate sets $\{C_u\}$, evaluation targets $\{i_u^*\}$, cutoff K

Ensure: Metric drops $\Delta\text{HR}@K$ and $\Delta\text{NDCG}@K$

- 1: Evaluate full metrics on $\{C_u\}$ using $\mathbf{x}(u, i)$
 - 2: Evaluate no-rule metrics on $\{C_u\}$ using $\mathbf{x}_{\text{no-rule}}(u, i)$
 - 3: Compute $\Delta\text{HR}@K$ and $\Delta\text{NDCG}@K$
-

A positive drop indicates operational dependence on the rule-evidence channel; near-zero drop indicates weak usage or redundancy under the tested regime.

3.10 Experimental Protocol and Evaluation Metrics

This section specifies the evaluation protocol used consistently across experiments so that Stage 1 candidate availability and Stage 2 ordering quality remain separable.

3.10.1 Top- K ranking metrics

Final recommendation quality is reported with $\text{HR}@K$ and $\text{NDCG}@K$, standard top- K metrics in recommendation evaluation [57]. Let i_u^* denote the held-out target and $\hat{R}_u^K$ the predicted list:

$$\text{HR}@K = \frac{1}{|U|} \sum_{u \in U} \mathbb{I}(i_u^* \in \hat{R}_u^K), \quad (3.26)$$

$$\text{NDCG}@K = \frac{1}{|U|} \sum_{u \in U} \frac{\mathbb{I}(i_u^* \in \hat{R}_u^K)}{\log_2(\text{rank}_u(i_u^*) + 1)}. \quad (3.27)$$

3.10.2 Candidate recall and retrieval-ceiling interpretation

Candidate Recall@C is reported for Stage 1:

$$\text{Recall}@C = \frac{1}{|U|} \sum_{u \in U} \mathbb{I}(i_u^* \in C_u). \quad (3.28)$$

Because Stage 2 ranks only within C_u , Recall@C provides the ceiling context for interpreting end-to-end HR/NDCG in two-stage settings [58, 59].

3.10.3 Bounded-candidate evaluation and sampled negatives

Offline evaluation uses a bounded-candidate setting with sampled negatives for scalability. Sampled ranking metrics depend on negative-pool size and composition; therefore, comparisons are interpreted only under matched sampling conditions [4, 62]. Where reported, conditional (candidate-hit) evaluation is used to isolate Stage 2 ordering quality from Stage 1 misses.

3.10.4 Ablation design and reporting protocol

Ablations isolate component effects under a fixed candidate-budget regime. Variants include encoder-only, encoder+rules, optional metadata/text, and hybrid configurations, evaluated under the same chronological split, masking policy, and candidate-budget constraints. Reported outcomes include Recall@C, HR@K, NDCG@K, EC@K, and (where applicable) fidelity-drop, ensuring that retrieval availability, ranking effectiveness, explanation availability, and evidence dependence remain analytically separable.

Chapter 4

Experiments

In this chapter, I evaluate HiX2S under a two-stage retrieval→reranking pipeline with a strict candidate-budget constraint. The experiments answer three operational questions:

1. **Stage-1 (retrieval-stage) improvement:** does train-only symbolic expansion improve candidate availability under a fixed serving cap (measured by Candidate Recall@C)?
2. **Stage-2 (reranking) improvement:** given fixed candidates, does evidence-aware reranking improve end-to-end ranking quality (measured by HR@K and NDCG@K)?
3. **Explainability behavior:** are explanations merely displayed post-hoc, or does the symbolic evidence channel measurably affect scoring (tested via EC@K and a fixed-candidate evidence-removal diagnostic)?

Because Stage 2 cannot recover a held-out positive that is absent from the Stage 1 candidate set, I report Candidate Recall@C alongside end-to-end HR@K and NDCG@K throughout. I also report explanation availability via EC@K and evidence dependence via a fixed-candidate masking test (fidelity-drop). Finally, to address feasibility concerns raised by the examiners, I include a latency/throughput snapshot and approximate memory-footprint accounting.

Table 4.1: Software and compute environment used for the reported experiments.

Component	Specification
Python	3.10
Deep learning framework	PyTorch
Execution mode	CUDA-enabled (<code>cuda</code>)
GPU	$2 \times$ NVIDIA T4 (15 GB each)
CPU	4-core AMD EPYC 7B12
RAM	30 GB

4.1 Experimental Setup

4.1.1 Implementation and compute environment

I implemented the pipeline in Python using PyTorch for training and inference, with standard data/scientific libraries for preprocessing and evaluation. The reported runs use CUDA execution on `cuda`. The environment uses Python 3.10 with two NVIDIA T4 GPUs.

Reproducibility note. Exact seed values and deterministic backend flags were not retained in the summarized logs for all runs. Small rerun-level variation is therefore possible unless deterministic settings are enforced in the archived code.

4.1.2 Datasets and preprocessing

I evaluate three datasets that represent different retrieval regimes: Amazon Books (large and sparse), MovieLens-1M (smaller and denser), and Yelp (intermediate). I use an implicit-feedback formulation and a chronological leave-one-out style split (one validation and one test positive per eligible user). For rating-based datasets, I convert explicit ratings to implicit positives using `min_rating=4.0`.

I set `filter_unseen_val_test_items=True` and `drop_val_test_users_not_in_train=True` to keep evaluation well-defined for the trained encoder (i.e., avoid true cold-start evaluation in these experiments). Rule mining is train-only to avoid leakage into symbolic expansion and explanation features.

Optional channel note. The framework supports a text/metadata retrieval channel; however, in the reported Amazon Books and MovieLens-1M FULL configura-

Table 4.2: Dataset and split statistics used in the reported experiments.

Dataset	Train-space users/items	Train / Valid / Test positives
Amazon Books	769,383 / 493,012	6,714,851 / 764,600 / 764,601
MovieLens-1M	6,038 / 3,524	563,209 / 6,031 / 6,032
Yelp	188,266 / 139,291	2,254,708 / 150,346 / 150,346

tion, the text channel is intentionally disabled with $C_{\text{text_extra}} = 0$ and $w_{\text{text}} = 0.0$ to isolate the neural+rule contribution. In Yelp ablations, an attribute/category channel is evaluated.

4.1.3 Evaluation protocol

I evaluate ranking at $K \in \{10, 20\}$ with sampled ranking using `sampled_negatives=99` (one held-out positive + 99 sampled negatives per user). Comparisons are interpreted only within matched sampling conditions.

Stage 1 uses a strict budgeted candidate construction:

$$C_{\text{base}} = 1200, \quad C_{\text{rule_extra}} = 800, \quad C_{\text{cap}} = 2000,$$

and candidates are formed via cap-after-union. Thus, any Recall@C gains reflect better candidate composition under the same serving cap, not relaxed candidate volume. Training interactions are masked at evaluation time so previously seen items are not recommended.

4.1.4 What I mean by “hybrid” and “explainable” in this chapter

Hybrid (neuro-symbolic). In this thesis, “hybrid” means I combine (i) a neural collaborative retriever (LightGCN) and (ii) a symbolic rule channel mined from train histories. The rule channel is used both for candidate construction (extra candidates under a fixed cap) and as explicit evidence features for reranking/explanations. When the attribute/category channel is enabled (Yelp ablation), it is treated as an auxiliary augmentation channel.

Explainable (evidence-aligned and testable). In this thesis, “explain-

able” does not mean the MLP is inherently interpretable. It means the system exposes an auditable evidence channel: (i) explanations are *evidence-aligned* (derived from mined rules/feature signals rather than invented narratives), (ii) availability is quantified via EC@K, and (iii) dependence is tested via a fixed-candidate evidence-removal diagnostic (fidelity-drop).

4.1.5 Compared configurations (A0/A1/A2/A3 and FULL)

I use the following variants:

- **A0 (encoder-only):** Stage 1 uses the neural retriever only; no rule expansion; Stage 2 uses the base reranker features.
- **A1 (encoder+rules):** train-only rule expansion augments encoder candidates under the same cap; rule-derived evidence features are available to Stage 2.
- **A2 (text/attr channel):** activates a non-rule augmentation channel. In Yelp, this corresponds to an attribute/category channel; in Amazon/MovieLens the text channel is disabled (so A2 collapses to A0 in those runs).
- **A3 (hybrid):** combines rules with the auxiliary channel under the same cap (collapses to A1 when the auxiliary channel is disabled).
- **FULL (HiX2S):** capped Stage 1 candidate construction followed by Stage 2 evidence-aware MLP reranking, with explanation diagnostics.

4.1.6 EFE-proxy reranking (definition and purpose)

In addition to standard reranking, I report an **EFE-proxy** reranking mode as a diagnostic. The goal is *not* to claim a new learning algorithm, but to test whether emphasizing evidence changes ranking behavior under the same candidate sets.

Let $s_{\text{std}}(u, i)$ denote the standard reranker score for user u and item i . Let $e(u, i) \in \{0, 1\}$ indicate whether item i has at least one valid evidence path (e.g.,

Table 4.3: FULL configuration used in the reported experiments.

Parameter	Value
<code>min_rating</code>	4.0
<code>embed_dim, n_layers</code>	64, 2
<code>lr, l2, epochs_full</code>	2×10^{-3} , 1×10^{-6} , 10
<code>C_base, C_rule_extra, cap_union</code>	1200, 800, 2000
<code>C_text_extra, w_text</code>	0, 0.0
<code>w_rule</code>	1.0
<code>rr_hidden, rr_epochs, rr_neg_per_pos</code>	128, 4, 5
<code>K_list, sampled_negatives</code>	(10, 20), 99

a fired rule from the user history to i under the current evidence construction).

The EFE-proxy score is:

$$s_{\text{EFE}}(u, i) = s_{\text{std}}(u, i) + \lambda \cdot e(u, i),$$

where $\lambda \geq 0$ is a fixed evidence bonus. This is a *proxy* because it does not solve a full multi-objective optimization; it provides a controlled knob to test whether “evidence preference” would materially change the top- K list under fixed candidates. If EFE-proxy and standard results nearly overlap, it indicates that (under the chosen λ and featureization) evidence preference does not strongly alter ranking outcomes.

4.1.7 Fixed hyperparameters (FULL configuration)

4.2 External Baseline Retriever Comparison (Fixed Budget)

To contextualize the retrieval ceiling, I include a retriever-only baseline study under the same candidate budget ($C_{\text{cap}} = 2000$) and the same split/masking protocol. These baselines are *context* for Stage 1 ceilings; my primary claim remains the delta from encoder-only LightGCN to budgeted union under the same cap.

Table 4.4: Retrieval-stage comparison under fixed candidate budget ($C_{\text{cap}} = 2000$). Values are **test** Recall@C.

Retriever	Amazon Books	MovieLens-1M	Yelp
UltraGCN	0.00322	0.97879	0.03695
SGL	0.08614	0.60166	0.59788
LightGCL	0.14238	0.97995	0.50052
SimDCL	0.11847	0.97863	0.57460
LightGCN (encoder-only Stage 1)	0.16010	0.90517	0.42964
HiX2S Stage 1 (budgeted union)	0.25266	0.91214	0.48507

Table 4.5: Retriever-only ranking under fixed candidate budget ($C_{\text{cap}} = 2000$): **test** HR@10 and NDCG@10.

Retriever	Amazon Books	MovieLens-1M	Yelp
UltraGCN	0.00002 / 0.00001	0.05087 / 0.02375	0.00013 / 0.00008
SGL	0.00942 / 0.00494	0.01773 / 0.00845	0.03413 / 0.01707
LightGCL	0.00966 / 0.00482	0.05750 / 0.02861	0.02498 / 0.01271
SimDCL	0.01021 / 0.00518	0.07092 / 0.03408	0.02852 / 0.01429

4.3 Latency and Memory Footprint (Feasibility Evidence)

This section addresses feasibility concerns by reporting (i) an environment-specific throughput snapshot and (ii) approximate memory-footprint accounting. I do not claim universal wall-clock superiority, since runtime depends on batching, kernels, and hardware. Instead, I expose the dominant terms and show the additional overhead introduced by evidence-aware scoring and symbolic structures.

4.3.1 Runtime / throughput snapshot (MovieLens-1M)

For reference, these throughputs correspond to approximately 40.75–41.34 ms/user for standard reranking and 60.94–63.90 ms/user for EFE-proxy reranking (computed as 1000/users-per-second). The slowdown is expected because EFE-proxy introduces an evidence preference term and additional bookkeeping under the same candidate sets.

4.3.2 Approximate memory footprint (lower-bound accounting)

To provide a transparent memory comparison, I report a lower-bound accounting for float32 embeddings and a COO edge list for the interaction graph. With

Table 4.6: Retriever-only ranking under fixed candidate budget ($C_{\text{cap}} = 2000$): test HR@20 and NDCG@20.

Retriever	Amazon Books	MovieLens-1M	Yelp
UltraGCN	0.00004 / 0.00001	0.09080 / 0.03372	0.00028 / 0.00012
SGL	0.01419 / 0.00615	0.03016 / 0.01154	0.05606 / 0.02257
LightGCL	0.01650 / 0.00654	0.11036 / 0.04183	0.04105 / 0.01674
SimDCL	0.01892 / 0.00736	0.12063 / 0.04658	0.04636 / 0.01877

Table 4.7: Runtime snapshot for the reported MovieLens-1M configuration (environment-specific).

Component / measurement	Value
Rule mining time	1.24 s
Retained rule edges	675,128
LightGCN training time (10 epochs)	3.8 s
Reranker training time (4 epochs)	2166.9 s
Standard reranking throughput (valid)	24.54 users/s
Standard reranking throughput (test)	24.19 users/s
EFE-proxy throughput (valid)	15.65 users/s
EFE-proxy throughput (test)	16.41 users/s

embedding dimension $d = 64$ and float32 storage, the embedding memory is:

$$\text{EmbMem} \approx (|U| + |I|) \cdot d \cdot 4 \text{ bytes.}$$

For an interaction graph stored as COO int32 pairs, edge memory is:

$$\text{EdgeMem} \approx |E| \cdot 2 \cdot 4 \text{ bytes.}$$

Rule-index note. The retained rule graph contains 675,128 edges (MovieLens snapshot). If stored as (antecedent int32, consequent int32, weight float32), this is about $675,128 \times 12 \text{ bytes} \approx 7.73 \text{ MiB}$, excluding hashmap overhead.

4.4 Stage-1 Candidate Generation Analysis

This section analyzes whether train-only symbolic expansion improves candidate availability under a strict cap.

Table 4.8: Approximate memory footprint (lower-bound estimates, float32 embeddings, COO int32 edges).

Dataset	#Users	#Items	Embeddings (MiB)	Train edges (MiB)
Amazon Books	769,383	493,012	308.20	51.23
MovieLens-1M	6,038	3,524	2.33	4.30
Yelp	188,266	139,291	79.97	17.20

Table 4.9: Amazon Books Stage 1 candidate Recall@C at fixed cap ($C_{\text{cap}} = 2000$).

Configuration	Valid Recall@C	Test Recall@C
A0 Encoder-only	0.18445	0.16010
A1 Encoder+Rules	0.31015	0.24805
A2 Encoder+TFIDF (disabled)	0.18445	0.16010
A3 Hybrid (Rules+TFIDF)	0.31015	0.24805
FULL (HiX2S)	0.31354	0.25266

4.4.1 Candidate recall under fixed budgets

Amazon Books (retrieval-limited regime). Relative to A0, A1 increases Recall@C by +0.12570 (valid) and +0.08795 (test). FULL increases Recall@C by +0.12909 (valid) and +0.09256 (test). These are large ceiling lifts under a fixed cap in a sparse large-catalog setting.

MovieLens-1M (high-recall regime). Candidate recall is already high; rule expansion provides only marginal additional headroom under this budget.

Yelp (intermediate regime). Using the same Stage 1 evaluation, base candidates achieve test Recall@C of 0.4296368 and union candidates achieve 0.4850743 over 188,266 users (absolute gain +0.05544). For validation, the retained notebook output reports base 0.4783763 and union 0.5409474.

4.4.2 Retrieval ceiling interpretation

In a two-stage pipeline, Stage 2 cannot place a held-out positive into top- K if it is not present in the candidate set. Under the same user set and protocol, this

Table 4.10: MovieLens-1M Stage 1 candidate Recall@C at fixed cap ($C_{\text{cap}} = 2000$).

Configuration	Valid Recall@C	Test Recall@C
A0 Encoder-only	0.9180899	0.9051724
FULL (HiX2S)	0.9238932	0.9121353
Absolute gain (FULL – A0)	+0.0058033	+0.0069629

Table 4.11: Yelp Stage 1 candidate Recall@C under fixed capped candidates.

Configuration	Valid Recall@C	Test Recall@C
Base candidates (encoder top- C)	0.4783763	0.4296368
Union candidates (rules/attr expansion)	0.5409474	0.4850743
Absolute gain (union – base)	+0.0625711	+0.0554375

yields the practical constraint:

$$\text{HR@}K \leq \text{Recall@}C,$$

and similarly bounds bounded-candidate NDCG@K. Across datasets, the gap between Recall@C and HR/NDCG quantifies remaining headroom attributable to within-candidate ordering.

4.5 Stage-2 Reranking Performance

4.5.1 End-to-end results (all users)

Table 4.12 reports end-to-end performance under fixed candidates. Standard rerank is the main evidence-aware MLP reranker; EFE-proxy is a diagnostic scoring mode (Section 4.1.6). Near-overlap between standard and EFE-proxy indicates the proxy does not materially change ordering under the current setting.

4.5.2 Conditional reranking (candidate-hit users only)

To separate retrieval misses from ordering quality, I report a candidate-hit diagnostic on the subset of users whose held-out positive appears in the Stage 1 candidate set. For Yelp test, all-user HR@10 is 0.1774882, while candidate-hit HR@10 is 0.3659867; this difference quantifies how Stage 1 availability continues

Table 4.12: Stage-2 end-to-end ranking performance (all users) under fixed candidate budgets.

Dataset	Split	Stage-2 mode	HR@10	NDCG@10	HR@20	NDCG@20	EC@10	EC@20
Amazon Books	Test	Standard rerank	0.03880	0.020088	0.05750	0.024794	0.859786	0.801132
Amazon Books	Test	EFE-proxy rerank	0.03872	0.020028	0.05752	0.024757	0.858376	0.799680
MovieLens-1M	Test	Standard rerank	0.0437666	0.0217861	0.0747679	0.0294939	0.9977122	0.9965103
MovieLens-1M	Test	EFE-proxy rerank	0.0424403	0.0214476	0.0736074	0.0292304	—	—
Yelp	Test	Standard rerank	0.1774882	0.0948147	0.2644822	0.1167215	0.4839865	0.4833225

Table 4.13: Explanation coverage on the test split (FULL configuration).

Dataset	EC@10	EC@20
Amazon Books	0.859786	0.801132
MovieLens-1M	0.997712	0.996510
Yelp	0.483986	0.483322

to constrain end-to-end performance even when Stage 2 ordering is strong on retrieved positives.

4.6 Explainability and Evidence Diagnostics

4.6.1 Explanation coverage (EC@K)

EC@K measures explanation *availability*: the fraction of top- K recommended items with at least one valid evidence-backed explanation. It does not prove evidence influenced ranking, which I test via fidelity-drop.

4.6.2 Faithfulness via evidence removal (fidelity-drop)

To test whether the symbolic evidence channel affects scoring (not just explanation text), I cache candidates C_u , score with the full feature vector, then re-score the same candidates with the rule-evidence feature masked and recompute metrics:

$$\Delta\text{HR@}K = \text{HR@}K_{\text{full}} - \text{HR@}K_{\text{no-rule}}, \quad \Delta\text{NDCG@}K = \text{NDCG@}K_{\text{full}} - \text{NDCG@}K_{\text{no-rule}}.$$

These results support the interpretation that (at least in Amazon valid and Yelp test retained runs) rule evidence is operational in reranking decisions under fixed candidates.

Table 4.14: Retained fixed-candidate fidelity-drop diagnostics. Positive drops indicate operational dependence on rule evidence in Stage 2 scoring.

Dataset	Split/Variant	$\Delta\text{HR@10}$	$\Delta\text{NDCG@10}$
Amazon Books	Valid / rule-masked run	0.0490	0.0213
Yelp	Test / hybrid-masked run	0.0749896	0.0422126

Table 4.15: Candidate Recall@C across ablations under $C_{\text{cap}} = 2000$.

Dataset	Variant	Valid Recall@C	Test Recall@C
Amazon Books	A0	0.18445	0.16010
Amazon Books	A1	0.31015	0.24805
Amazon Books	A2 (disabled)	0.18445	0.16010
Amazon Books	A3	0.31015	0.24805
Amazon Books	FULL	0.31354	0.25266
MovieLens-1M	A0	0.9180899	0.9051724
MovieLens-1M	FULL	0.9238932	0.9121353
Yelp	A0 (base)	0.4783763	0.4296368
Yelp	FULL/Union	0.5409474	0.4850743
Yelp	A1/A2/A3 (per-variant recall)	0.54010	0.48460

4.7 Ablation Studies

I use A0/A1/A2/A3 to isolate component effects under the same split/masking/budget protocol.

4.7.1 Ablation: Candidate Recall@C

4.7.2 Ablation: End-to-end HR/NDCG at K=10

Interpretation. Amazon shows the strongest retrieval-ceiling lift (Recall@C increases sharply), MovieLens shows smaller Stage 1 headroom but meaningful Stage 2 ordering improvements, and Yelp exhibits a mixed regime where hybrid A3 is strongest in the retained ablation export.

4.8 Limitations

This chapter provides protocol-bounded offline evidence. Key limitations are:

- **Sampled evaluation:** metrics use one positive + 99 sampled negatives; absolute values are protocol-dependent.

Table 4.16: End-to-end metrics at $K = 10$ across ablations (test).

Dataset	Variant	HR@10	NDCG@10
Amazon Books	A0	0.02375	0.01274
Amazon Books	A1	0.03725	0.01897
Amazon Books	FULL	0.03880	0.02009
MovieLens-1M	A0	0.01907	0.00855
MovieLens-1M	A1 (rules)	0.04625	0.02290
MovieLens-1M	FULL	0.04377	0.02179
Yelp	A0	0.137449	0.074478
Yelp	A1	0.175794	0.095717
Yelp	A2	0.131766	0.070105
Yelp	A3	0.179135	0.096008
Yelp	FULL	0.177488	0.094815

- **Latency profiling scope:** runtime values are environment-specific snapshots; not a universal benchmark.
- **Cold-start:** cold-start users/items are filtered out by design in this evaluation protocol.
- **Baseline parity:** external retriever baselines are contextual.

4.9 Research Questions and Answers

RQ1. Do rule-based candidate expansion and evidence-aware reranking improve HR@K/NDCG@K compared to encoder-only under the same cap? Yes, but the magnitude is regime-dependent. Amazon shows clear ceiling lift and end-to-end gains; MovieLens shows smaller Stage 1 lift but meaningful ordering gains; Yelp shows hybrid improvements in the retained ablation export.

RQ2. How does Recall@C bound and explain end-to-end performance? Recall@C provides a retrieval ceiling context: if the target is not in the candidate set, HR/NDCG cannot recover it. Joint reporting of Recall@C and HR/NDCG clarifies whether gains come from Stage 1 availability or Stage 2 ordering.

RQ3. What trade-offs arise under EFE-proxy reranking? In the retained runs, EFE-proxy and standard rerank are near-overlapping on Amazon and MovieLens, indicating the proxy does not materially change ranking behavior under the chosen settings. Therefore, I interpret EFE-proxy primarily as a diagnostic policy rather than a superior replacement in this configuration.

Chapter 5

Conclusion and Future Work

5.1 Conclusion

Explainable recommendation is most meaningful when ranking quality and explanation behavior are evaluated jointly rather than as separate objectives. In two-stage systems, this requirement becomes critical because the final top- K ranking is strictly bounded by the Stage 1 candidate set. In sparse implicit-feedback settings, candidate availability becomes a structural bottleneck: if the relevant item is not retrieved, no reranker can recover it. This thesis addressed that methodological gap by explicitly analyzing retrieval quality, ranking performance, explanation coverage, and evidence dependence under a fixed candidate-budget retrieval→reranking pipeline.

To study this setting, the thesis introduced a neuro-symbolic two-stage framework that combines a LightGCN retriever, train-only symbolic candidate expansion, and an explainable MLP reranker. The contribution lies not only in the architectural integration, but in the evaluation discipline used to analyze it. By reporting Candidate Recall@ C alongside HR@ K , NDCG@ K , EC@ K , and fixed-candidate evidence-removal diagnostics, the experiments separated retrieval effects from reranking effects and tested whether symbolic evidence was operational in scoring rather than appended post-hoc. This design made it possible to interpret performance changes through a retrieval-ceiling lens under leakage-safe and budget-controlled conditions.

Empirical results across Amazon Books, MovieLens-1M, and Yelp demon-

strate regime-sensitive behavior. In retrieval-limited settings such as Amazon Books, rule-based expansion increased Candidate Recall@ C (0.3135/0.2527 valid/test) and this lift was accompanied by corresponding gains in HR@10 and NDCG@10, with EC@10 approaching 0.86. Conditional candidate-hit analysis confirmed that much of the end-to-end error arises from candidate misses rather than ordering failure alone. Fidelity-style masking further showed measurable ranking degradation in regimes where symbolic evidence was active (e.g., $\Delta\text{HR@10} = 0.0490$ on Amazon Books; larger drops on Yelp), supporting evidence dependence under fixed candidates. In higher-recall regimes such as MovieLens-1M, retrieval headroom was smaller and fidelity-drop was limited, clarifying that explanation availability and evidence dependence are related but distinct properties. Overall, the findings support a retrieval-ceiling-aware interpretation of two-stage recommendation and demonstrate that symbolic expansion and explainable reranking can improve both ranking quality and explanation coverage under realistic serving constraints.

5.2 Future Work

This thesis is intentionally bounded by a leakage-safe, fixed-budget offline evaluation design, and the conclusions are strongest within that scope. The reported findings are based on the retained experimental runs and exported diagnostics across Amazon Books, MovieLens-1M, and Yelp. While this provides clear evidence for the main contributions, some diagnostics (especially fidelity and fully matched variant-level exports) were not retained uniformly for every dataset/variant combination. These are scope boundaries of the present study rather than contradictions to the central results.

A first priority for future work is to strengthen Stage 1 candidate generation under the same serving-time budget, since the results show that Candidate Recall@ C remains the dominant bottleneck in retrieval-limited regimes. Promis-

ing directions include stronger retrievers (e.g., contrastive or sequential models), adaptive candidate allocation, and better-calibrated symbolic expansion policies that increase candidate availability without relaxing computational constraints. A second priority is to extend the symbolic layer beyond co-occurrence rules by integrating richer evidence channels such as item attributes, sequential patterns, or knowledge-graph relations, and to evaluate the semantic/text channel in a fully active, budgeted configuration with reliable metadata quality.

Future work should also extend the current offline protocol toward stronger external validity, including latency and serving-cost analysis, robustness testing, and user-centered evaluation of explanation usefulness and trust. In addition, tighter paired comparisons between standard reranking and EFE-proxy reranking under identical candidate sets, user subsets, and novelty protocols would provide a clearer estimate of any accuracy–novelty–coverage trade-offs. These extensions would build directly on the present contribution, which establishes a neuro-symbolic two-stage framework and an evaluation methodology that jointly analyzes retrieval constraints, ranking effectiveness, explanation coverage, and evidence dependence under fixed candidate budgets.

REFERENCES

- [1] K. Peffers, T. Tuunanen, M. A. Rothenberger, and S. Chatterjee, “A design science research methodology for information systems research,” *Journal of Management Information Systems*, vol. 24, no. 3, pp. 45–77, 2007.
- [2] P. Covington, J. Adams, and E. Sargin, “Deep neural networks for youtube recommendations,” in *Proceedings of the 10th ACM Conference on Recommender Systems (RecSys)*, 2016.
- [3] W.-C. Kang and J. McAuley, “Cigar: Candidate generation and re-ranking for recommendation,” *arXiv preprint arXiv:1905.04108*, 2019.
- [4] W. Krichene and S. Rendle, “On sampled metrics for item recommendation,” in *Proceedings of the 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, 2020.
- [5] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, and M. Wang, “Lightgcn: Simplifying and powering graph convolution network for recommendation,” in *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, 2020.
- [6] R. Agrawal and R. Srikant, “Fast algorithms for mining association rules,” in *Proceedings of the 20th International Conference on Very Large Data Bases (VLDB)*, 1994.
- [7] M. J. Zaki, “Scalable algorithms for association mining,” in *IEEE Transactions on Knowledge and Data Engineering*, 2000.
- [8] M. T. Ribeiro, S. Singh, and C. Guestrin, ““why should i trust you?”: Explaining the predictions of any classifier,” in *Proceedings of the 22nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, 2016.
- [9] A. Jacovi and Y. Goldberg, “Towards faithfully interpretable nlp systems: How should we define and evaluate faithfulness?,” *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2020.

- [10] J. Adebayo, J. Gilmer, M. Muelly, I. Goodfellow, M. Hardt, and B. Kim, “Sanity checks for saliency maps,” in *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.
- [11] Y. Hu, Y. Koren, and C. Volinsky, “Collaborative filtering for implicit feedback datasets,” in *Proceedings of IEEE ICDM*, 2008.
- [12] R. Pan, Y. Zhou, B. Cao, N. N. Liu, R. Lukose, M. Scholz, and Q. Yang, “One-class collaborative filtering,” in *Proceedings of ECML/PKDD*, 2008.
- [13] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, “Bpr: Bayesian personalized ranking from implicit feedback,” in *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence (UAI ’09)*, pp. 452–461, AUAI Press, 2009.
- [14] C. A. Gomez-Urbe and N. Hunt, “The netflix recommender system: Algorithms, business value, and innovation,” *ACM Transactions on Management Information Systems*, vol. 6, no. 4, pp. 13:1–13:19, 2015.
- [15] W. Liu, W. Chen, Z. Dong, R. Tang, R. Wang, R. Zhang, W. Zhang, and Y. Yu, “Neural re-ranking in multi-stage recommender systems: A review,” in *Proceedings of the 31st International Joint Conference on Artificial Intelligence (IJCAI)*, 2022.
- [16] Y. Zhang and X. Chen, “Explainable recommendation: A survey and new perspectives,” *Foundations and Trends® in Information Retrieval*, vol. 14, no. 1, pp. 1–101, 2020.
- [17] M. Ge, C. Delgado-Battenfeld, and D. Jannach, “Beyond accuracy: Evaluating recommender systems by coverage and serendipity,” in *RecSys*, 2010.
- [18] J. McAuley, C. Targett, Q. Shi, and A. van den Hengel, “Image-based recommendations on styles and substitutes,” in *Proceedings of the 38th International ACM SIGIR Conference (SIGIR)*, ACM, 2015.
- [19] Yelp, “Yelp open dataset.” <https://www.yelp.com/dataset>, 2021. Accessed: 2025-02-22.
- [20] F. M. Harper and J. A. Konstan, “The movielens datasets: History and context,” in *ACM Transactions on Interactive Intelligent Systems*, 2015.
- [21] X. Zheng, F. Shirani, Z. Chen, C. Lin, W. Cheng, W. Guo, and D. Luo, “Robust fidelity: Evaluating the robustness of explanations for machine learning,” 2024. Proceedings of the 12th International Conference on Learning Representations (ICLR).

- [22] J. Hron, N. Kilbertus, *et al.*, “On component interactions in two-stage recommender systems,” in *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.
- [23] K. Zheng, K. Gai, *et al.*, “Full stage learning to rank: A unified framework for multi-stage systems,” in *Proceedings of The Web Conference (WWW)*, 2024.
- [24] Y. Zhang and X. Chen, “Explainable recommendation: A survey and new perspectives,” *Foundations and Trends in Information Retrieval*, 2020.
- [25] M. Velmurugan, K. Aiswarya, R. Krishnan, S. J. Reddi, P. Netrapalli, S. Kumar, and P. Jain, “Through the looking glass: Evaluating post hoc explanations using transparent models,” *Machine Learning and Knowledge Extraction*, 2023.
- [26] X. Zheng, F. Shirani, Z. Chen, C. Lin, W. Cheng, W. Guo, and D. Luo, “Fidelity: A robust framework for evaluating the faithfulness of explainable ai,” 2025. *Proceedings of the 13th International Conference on Learning Representations (ICLR)*.
- [27] J. McAuley, C. Targett, J. Shi, and A. V. D. Hengel, “Inferring networks of substitutable and complementary products,” in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2015.
- [28] T. Schnabel, A. Swaminathan, A. Singh, N. Chandak, and T. Joachims, “Recommendations as treatments: Debiasing learning and evaluation,” in *Proceedings of the 33rd International Conference on Machine Learning (ICML)*, pp. 1670–1679, PMLR, 2016.
- [29] B. M. Marlin and R. S. Zemel, “Collaborative prediction and ranking with non-random missing data,” in *Proceedings of the 3rd ACM Conference on Recommender Systems (RecSys)*, pp. 5–12, ACM, 2009.
- [30] T. Joachims, A. Swaminathan, and T. Schnabel, “Unbiased learning-to-rank with biased feedback,” in *Proceedings of the 10th ACM International Conference on Web Search and Data Mining (WSDM)*, pp. 781–789, ACM, 2017.
- [31] W. Wang, F. Feng, X. He, X. Wang, and T. Chua, “Deconfounded video recommendation with causal intervention,” in *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pp. 189–198, ACM, 2020.

- [32] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, “Neural collaborative filtering,” in *Proceedings of the 26th International Conference on World Wide Web (WWW)*, pp. 173–182, ACM, 2017.
- [33] Y. Malkov and D. Yashunin, “Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 4, pp. 824–836, 2020.
- [34] J. Johnson, M. Douze, and H. Jégou, “Billion-scale similarity search with gpus,” 2017.
- [35] W. Liu, W. Chen, Z. Dong, R. Tang, R. Wang, R. Zhang, W. Zhang, and Y. Yu, “Neural re-ranking in multi-stage recommender systems: A review,” in *Proceedings of the 31st International Joint Conference on Artificial Intelligence (IJCAI)*, 2022.
- [36] Y. Koren, R. Bell, and C. Volinsky, “Matrix factorization techniques for recommender systems,” *IEEE Computer*, 2009.
- [37] X. Wang, X. He, M. Wang, F. Feng, and T. Chua, “Neural graph collaborative filtering,” in *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, 2019.
- [38] K. Mao, J. Zhu, X. Xiao, B. Lu, Z. Wang, and X. He, “Ultragcn: Ultra simplification of graph convolutional networks for recommendation,” in *Proceedings of the 30th ACM International Conference on Information and Knowledge Management (CIKM)*, 2021.
- [39] J. Wu, X. Wang, F. Feng, X. He, L. Chen, J. Lian, and X. Xie, “Self-supervised graph learning for recommendation,” in *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, 2021.
- [40] X. Cai, C. Huang, L. Xia, and X. Ren, “Lightgcl: Simple yet effective graph contrastive learning for recommendation,” in *The Eleventh International Conference on Learning Representations (ICLR 2023)*, 2023. OpenReview.
- [41] J. Yu, X. Xia, T. Chen, L. Cui, N. Q. V. Hung, and H. Yin, “Xsimgcl: Towards extremely simple graph contrastive learning for recommendation,” *IEEE Transactions on Knowledge and Data Engineering*, 2024.

- [42] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk, “Session-based recommendations with recurrent neural networks,” in *Proceedings of the 4th International Conference on Learning Representations (ICLR)*, 2016.
- [43] W. Kang and J. McAuley, “Self-attentive sequential recommendation,” in *2018 IEEE International Conference on Data Mining (ICDM)*, pp. 197–206, IEEE, 2018.
- [44] F. Sun, J. Liu, J. Wu, C. Pei, X. Lin, W. Ou, and P. Jiang, “Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer,” in *Proceedings of the 28th ACM International Conference on Information and Knowledge Management (CIKM ’19)*, pp. 1441–1450, ACM, 2019.
- [45] Z. Sun, J. Yu, H. Fang, Q. Yang, J. Liu, and Q. Qu, “Are we evaluating rigorously? benchmarking recommendation for reproducible evaluation and fair comparison,” in *Proceedings of the 14th ACM Conference on Recommender Systems (RecSys)*, pp. 23–32, Association for Computing Machinery, 2020.
- [46] Y. Zhang *et al.*, “Explainable graph neural network recommenders: Challenges and opportunities,” in *Proceedings of the 17th ACM Conference on Recommender Systems (RecSys)*, 2023.
- [47] K. Damak, S. Khenissi, and O. Nasraoui, “Debiased explainable pairwise ranking from implicit feedback,” in *RecSys*, 2021.
- [48] S. M. Lundberg and S. Lee, “A unified approach to interpreting model predictions,” in *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.
- [49] L. H. Gilpin, D. Bau, B. Z. Yuan, A. Bajwa, M. Specter, and L. Kagal, “Explaining explanations: An overview of interpretability of machine learning,” 2018.
- [50] G. Peake and J. Wang, “Explanation mining: Post hoc interpretability of latent factor models for recommendation,” in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery Data Mining*, 2018. Applies fidelity/faithfulness testing via feature perturbation in RecSys. URL: <https://doi.org>.
- [51] Y. Zhang and X. Chen, “Explainable recommendation: A survey and new perspectives,” *Foundations and Trends in Information Retrieval*, 2020.

- [52] Y. Zhao *et al.*, “A review of explainable graph-based recommender systems,” 2024.
- [53] M. Kaminskis and D. Bridge, “Diversity, novelty and serendipity in recommender systems: A survey and experimental evaluation,” *ACM Transactions on Interactive Intelligent Systems*, vol. 7, no. 1, 2016.
- [54] P. Castells, S. Vargas, and J. Wang, “Novelty and diversity metrics for recommender systems: choice, discovery and relevance,” in *ECIR*, 2011.
- [55] X. Wang, X. He, Y. Cao, M. Liu, and T. Chua, “Kgat: Knowledge graph attention network for recommendation,” in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*, pp. 950–958, ACM, 2019.
- [56] Z. Liu, J. Tan, B. Wu, G. Li, D. Yao, and J. Zhou, “Logger: A learned optimizer towards generating efficient and robust plans,” *Proceedings of the VLDB Endowment (PVLDB)*, vol. 16, no. 11, pp. 2849–2861, 2023.
- [57] K. Järvelin and J. Kekäläinen, “Cumulated gain-based evaluation of IR techniques,” *ACM Transactions on Information Systems*, vol. 20, no. 4, pp. 422–446, 2002.
- [58] J. Huang, J. Chen, J. Lin, J. Qin, Z. Feng, W. Zhang, and Y. Yu, “A comprehensive survey on retrieval methods in recommender systems,” *ACM Transactions on Information Systems*, 2025.
- [59] J. Hron, K. Krauth, M. I. Jordan, and N. Kilbertus, “On component interactions in two-stage recommender systems,” in *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.
- [60] R. Baeza-Yates and B. Ribeiro-Neto, *Modern Information Retrieval*. Addison-Wesley, 1999.
- [61] T.-Y. Liu, “Learning to rank for information retrieval,” *Foundations and Trends® in Information Retrieval*, vol. 3, no. 3, pp. 225–331, 2009.
- [62] B. Bai, X. Li, Y. Yu, P. Zheng, *et al.*, “Negative sampling in large-scale recommendation systems: A survey,” 2024.