Distill Globally, Adapt Locally: Reasoning Distillation and Product-Type Test-Time Training for Scalable Trade-Up Recommendation

Siliang Liu[1,*], Mohammad Ghasemi[1], Sapan Patel[1] and Amin Banitalebi-Dehkordi[1]

[1]*Amazon Everyday Essentials Technologies*

Abstract

Trade-up recommendation aims to identify higher-quality alternatives that preserve a customer's purchase intent while offering upgraded benefits through improved formulation, certifications, or brand positioning. Although large language models (LLMs) can reason about these subtle distinctions, applying them directly to hundreds of millions of product pairs in e-commerce stores is operationally impractical. We introduce a *two-level* framework that distills LLM-derived reasoning into an efficient non-generative student and subsequently adapts its decision boundary to product-type-specific trade-up criteria. At **Level 1**, a retrieval-augmented few-shot LLM teacher generates both structured relation labels and natural-language rationales. These rationales are encoded and transferred to a compact, non-generative embedding-pair classifier through alignment and contrastive objectives. At inference, the student consumes only two precomputed 768-dimensional product embeddings, requiring neither LLM calls nor text generation. On a fixed human-annotated benchmark of 8,352 pairs, a 15.5M-parameter four-class reasoning-distilled student achieves an AUC of 0.924 (95% CI $[0.918, 0.929]$), improving over the four-class label-only student with the same shallow architecture (AUC 0.912); rationale supervision provides little benefit when the same task is collapsed to binary labels. At **Level 2**, we introduce product-type test-time training (PT-TTT), which uses few-shot demonstrations as gradient-based supervision to optimize lightweight category-specific adapters over the frozen student model. PT-TTT improves AUC from 0.924 to 0.941 and average precision from 0.920 to 0.940 without serving-time LLM inference. On a 100K-pair proxy catalog, inference with the distilled student on a single eight-GPU machine is approximately $5{,}000\times$ faster and has an estimated cost approximately $10{,}000\times$ lower than direct LLM inference on the same workload.

Keywords

Knowledge distillation, LLM reasoning distillation, test-time training, product recommendation, e-commerce

1. Introduction

Trade-up recommendation identifies higher-quality alternatives that preserve a customer's shopping intent while offering additional benefits in attributes such as formulation, certifications, materials, or brand positioning. At catalog scale, this requires distinguishing meaningful upgrades from same-tier substitutes, variants, pack-size changes, and incompatible products.

Given a base product b and candidate c, we predict whether c is a valid trade-up for b. Unlike substitutability, variant, and complementarity relations [1, 2, 3], trade-up is directional: the candidate must preserve the shopping mission while providing evidence of an added benefit beyond price, quantity, or other superficial variation.

Large language models (LLMs) provide useful semantic supervision for recommendation and product relations [4, 5, 6, 7], and retrieval augmentation can ground their decisions with task-specific examples [8]. However, applying LLM inference at e-commerce catalog scale is operationally impractical. For a large e-commerce store, scoring the catalog can involve hundreds of millions of pairs, costing millions of dollars and delaying recommendation availability by weeks or months.

GenAIECommerce'26: The Third Workshop on Agentic and Generative AI for E-Commerce, co-located with RecSys, September 28, 2026, Minneapolis, MN, USA

*Corresponding author.

celineli@amazon.com (S. Liu); mohamgd@amazon.com (M. Ghasemi); sapanp@amazon.com (S. Patel); aminbt@amazon.com (A. Banitalebi-Dehkordi)

0009-0007-4561-7548 (S. Liu)

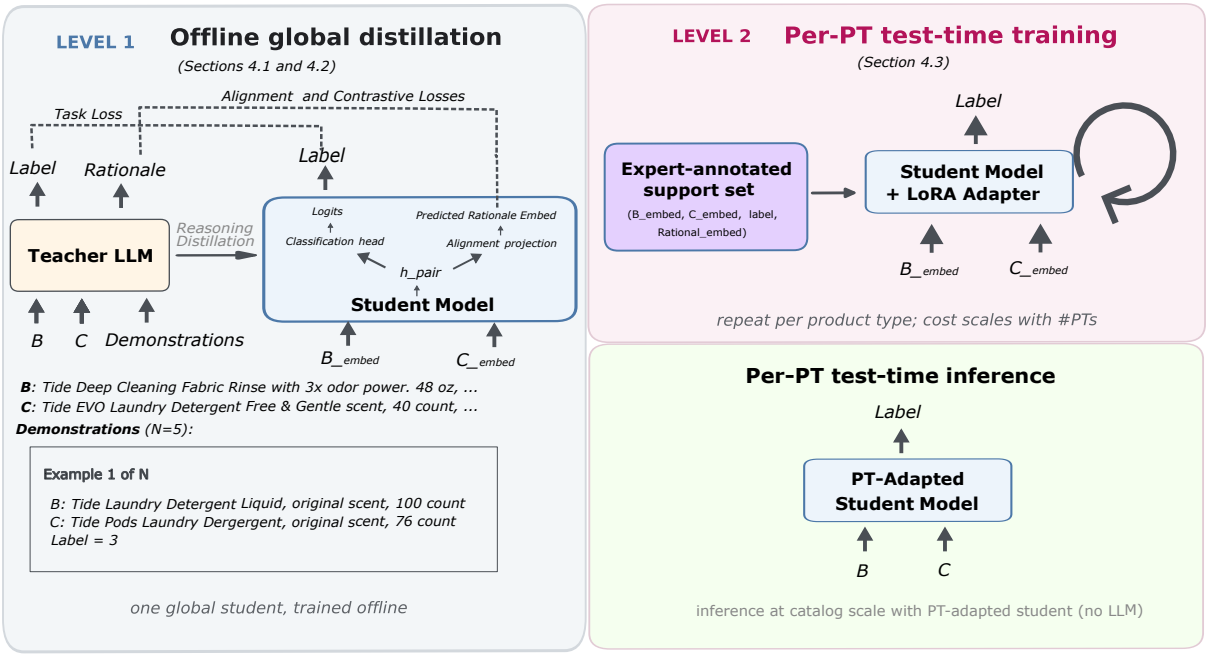

Figure 1: Two-level framework. **Level 1**: a frontier LLM teacher supervises a discriminative global student through reasoning-guided distillation. **Level 2**: the *same* per-product-type support sets that served as prompt context for the teacher are reused as test-time *gradient* supervision for a lightweight per-category adapter, applied once per product type before catalog-scale scoring. **Inference**: catalog-scale scoring with the PT-adapted student, with no LLM calls.

We propose a robust and scalable learning-based solution capable of identifying trade-up products. To achieve this, we integrate a two-level reasoning framework to distill nuanced trade-up relations across product types. Our contributions are as follows:

- We formulate trade-up identification as a directional, product-type-dependent relation classification problem designed for catalog-scale scoring without serving-time LLM inference.
- We introduce a reasoning-distillation framework that transfers fine-grained relation labels and rationale semantics from a retrieval-augmented LLM teacher to a compact, non-generative student (Sections 4.1 and 4.2).
- We introduce Product-Type Test-Time Training (PT-TTT), which uses product-type-specific demonstrations to fit lightweight adapters over the frozen student, improving predictive performance while preserving efficient large-scale inference (Section 4.3).

2. Related Work

Product relations and LLM supervision. Item-to-item relation modeling has primarily addressed substitutability, complementarity, compatibility, and product variants using behavioral, textual, and graph signals [1]; a related line uses fine-grained, ingredient-based product attributes for explainable e-commerce recommendation [9]. Formally, we model trade-up as a directional relation from a base product b to a candidate c, where c must preserve the underlying shopping mission while providing an additional customer-relevant benefit. We therefore treat trade-up identification as catalog-level relation classification upstream of candidate ranking, rather than as an end-to-end personalized recommendation problem.

LLMs have been used as recommenders, semantic feature generators, rerankers, and annotators [4]. Retrieval-augmented prompting can further condition their predictions on task-specific demonstrations [8]. In our setting, the LLM serves only as an offline semantic supervisor to produce a fine-grained

product-relation label and a textual rationale for each pair. This design targets a different operating regime from LLM-based online recommendation, as the deployed scorer must process hundreds of millions of pairs without generative inference. Moreover, the teacher is accessed as a black box, precluding conventional distillation from teacher logits or hidden states.

Rationale-guided representation distillation. Classical knowledge distillation transfers output distributions from a teacher to a compact student [10]; subsequent work matches intermediate features [11] or preserves relational geometry through contrastive objectives [12]. Reasoning-distillation methods extend the supervision signal with explanations or chain-of-thought traces. Distilling step-by-step jointly trains students on labels and rationales [13], while SCOTT introduces consistency objectives between generated reasoning and predictions [14]. These methods predominantly target generative students that emit or condition on rationale tokens.

Our student is an embedding-only pair classifier and does not decode text. We instead encode the teacher rationale into a fixed-dimensional representation and transfer it to the student pair embedding through pointwise alignment and contrastive relational matching. The projection and auxiliary losses are used only during training. This places our method between rationale distillation and representation distillation. The teacher provides natural-language supervision, but the transferred object is the geometry of the discriminative pair representation rather than a generated reasoning sequence. We use the term *rationale-guided representation distillation* because generated rationales need not faithfully expose the teacher's internal decision process [15]. Accordingly, we treat them as auxiliary semantic targets and evaluate their contribution separately from model capacity and label granularity. This distinction is consequential in our results: rationale supervision improves human-label alignment only when the student retains the teacher's fine-grained relation structure, rather than collapsing all non-trade-up cases into a single binary class.

Test-time product type-based adaptation. Test-time training adapts a trained model on the target distribution before prediction [16]; related test-time adaptation methods update only a restricted parameter subset under distribution shift [17]. Our setting is additionally connected to supervised few-shot adaptation and meta-learning [18], since each product type has a small labeled demonstration set. Recent work shows that such demonstrations can be converted from in-context examples into gradient supervision at inference time [19]. Parameter-efficient finetuning methods, including LoRA [20], make this adaptation feasible without modifying the shared backbone.

PT-TTT reuses the teacher's product-type-specific demonstrations as a support set for adapting a low-rank module over the frozen student. Because the support examples are labeled, the method is more precisely supervised few-shot test-time adaptation than conventional unsupervised test-time training. In our system, one adapter is optimized per product type and amortized over all pairs in that category, rather than optimized per query. This preserves the computational advantage of the distilled scorer while allowing category-specific trade-up criteria to modify the local decision boundary.

Overall, prior work studies semantic product relations, rationale distillation, and test-time adaptation largely as separate problems. Our framework combines them through a division of labor: rationale-guided distillation shapes a globally shared pair representation, while PT-TTT specializes its decision boundary using category-local supervision. The contribution is this coupling under catalog-scale constraints, without serving-time LLM inference or per-pair adaptation.

3. Task Setup and Data Preparation

3.1. Trade-up Definition

Given a base product b and candidate c, we predict a score $s(b, c) \in [0, 1]$ indicating whether c is a valid trade-up of b. A valid trade-up must preserve the same shopping intent while providing an additional benefit, such as stronger brand positioning, improved formulation, relevant certifications, or better materials. Differences in quantity, flavor, or packaging alone do not constitute a trade-up.

3.2. Label Sources and Corpora

Expert-annotated data and golden benchmark. We construct an expert-annotated corpus of 17,200 product pairs across 29 product types. Within each type, selected products are exhaustively paired and assigned a fixed b–c ordering under a four-class taxonomy: (1) similar/same tier; (2) b is a trade-up of c; (3) c is a trade-up of b; or (4) incompatible/not meaningfully comparable. We use 8,848 pairs as the teacher demonstration and PT-TTT support pool and reserve the remaining 8,352 pair-disjoint examples as the held-out golden benchmark. For binary evaluation, class 3 is positive and classes 1, 2, and 4 are negative. Golden labels are used only for final evaluation and analysis.

Silver supervision. Starting from 12,830 products in the Amazon–Walmart dataset [21], we construct within-product-type candidate pairs and retain semantically related pairs based on product-title and description embeddings, filtering out pairs unlikely to represent the same shopping intent. This yields 1,019,241 candidate pairs across the same 29 product types.

For each retained pair, a frontier LLM teacher, accessed as a black box, receives the product descriptions, product-type-specific trade-up criteria, and five dynamically retrieved expert demonstrations, and generates a four-class relation label and a concise rationale. These outputs constitute the silver supervision for global student training. The corpus is split 90/10 into training and validation sets, stratified by product type and relation class. Additional details on demonstration retrieval, teacher prompting, and rationale encoding are provided in Appendix A.1.1.

4. System Overview

Figure 1 shows an overview of the two-level pipeline: offline reasoning distillation into a frozen embedding-only student in Level 1 (Section 4.1 and 4.2), then per-product-type test-time training before scoring in Level 2 (Section 4.3).

4.1. Embedding-Pair Student Architecture

The student is a lightweight product-pair classifier that consumes precomputed base and candidate embeddings $e_b, e_c \in \mathbb{R}^d$, with $d = 768$. It uses separate base and candidate branches together with an ordered base-to-candidate branch, followed by an MLP classification head. No product text, rationale, or LLM output is required at inference.

The base and candidate embeddings are processed by separate parameterized branches. Because each product is represented by a single embedding, the self-attention operations act on length-one sequences and therefore have trivial attention weights. They therefore act as learned role-specific transformations rather than performing content-dependent attention over tokens. A third, ordered base-to-candidate branch provides an additional asymmetric transformation. Because each side contains a single embedding, its attention softmax is identically one and does not perform content-dependent selection. Directionality instead arises from the ordered base/candidate roles, separate branch parameters, and the ordered pair representation.

Let $h_b^{(L_s)}$ and $h_c^{(L_s)}$ denote the final outputs of the base and candidate branches, respectively, and let $z^{(L_x)}$ denote the final output of the ordered base-to-candidate branch. The three branch outputs are concatenated as

$$h_{\text{pair}} = \left[h_b^{(L_s)}; h_c^{(L_s)}; z^{(L_x)}\right] \in \mathbb{R}^{3d}, \tag{1}$$

and an MLP maps h_{pair} to binary or four-class logits.

Reasoning projection. During distillation, a training-only projection maps the pair representation to the rationale-embedding space:

$$\hat{r}_i = W_{\text{align}} h_{\text{pair}}^{(i)} + b_{\text{align}}, \qquad \hat{r}_i \in \mathbb{R}^{d_r}. \tag{2}$$

The projection is used only by the auxiliary distillation objectives and is discarded at inference. Full branch equations, layer dimensions, parameter counts, and architecture hyperparameters are provided in Appendix A.2.1.

4.2. Student Model Training Objectives

Task supervision. We consider binary and four-class supervision. In binary mode, class 3 (candidate is a trade-up of the base) is treated as positive, while classes 1, 2, and 4 are collapsed into the negative class:

$$y_i^{(2)} = \mathbb{I}[y_i^{(4)} = 3].$$

The binary model is optimized using weighted focal binary cross-entropy [22]. In four-class mode, we preserve the teacher's original relation taxonomy and optimize weighted cross-entropy. We denote either objective by $\mathcal{L}_{\text{task}}$.

Rationale alignment. Because the black-box teacher exposes neither logits nor hidden states, we use its natural-language rationales as auxiliary semantic supervision. Let $r^{(i)}$ be the encoded teacher rationale and $\hat{r}_i = W_{\text{align}} h_{\text{pair}}^{(i)} + b_{\text{align}}$ the student projection defined in Eq. 2. We minimize

$$\mathcal{L}_{\text{align}} = \frac{1}{N} \sum_{i=1}^{N} \left\| \hat{r}_i - r^{(i)} \right\|_2^2. \tag{3}$$

Contrastive distillation. Pointwise alignment does not explicitly preserve relationships among different pair–rationale examples. We therefore complement it with a contrastive objective $\mathcal{L}_{\text{con}}$ combining InfoNCE [23, 12], which contrasts each student projection against in-batch rationale embeddings, and a relational KL term [24] that transfers pairwise similarity structure.

Full objective. The global student is trained with

$$\mathcal{L} = \mathcal{L}_{\text{task}} + \lambda_{\text{align}} \mathcal{L}_{\text{align}} + \lambda_{\text{con}} \mathcal{L}_{\text{con}}. \tag{4}$$

The rationale projection and auxiliary distillation objectives are used only during training and discarded at inference. For the four-class model, the trade-up score is the softmax probability of class 3 (candidate is a trade-up of the base): $s(b, c) = \text{softmax}(g)_3$. Full loss definitions and optimization hyperparameters are provided in Appendix A.2.2.

4.3. Product-Type Test-Time Training (PT-TTT)

The globally distilled student uses a shared scoring function across product types, whereas trade-up criteria can differ substantially by category: evidence that makes a battery a trade-up differs from that of a moisturizer (formulation, certifications) or pet food (ingredient sourcing). We therefore add a second level of specialization inspired by test-time training [16, 19]. Because these support examples carry expert labels, the procedure is more precisely a supervised few-shot test-time adaptation method than conventional unsupervised test-time training.

Product-type adaptation. Let θ_0 denote the frozen global student and $S_t = \{(e_b^{(i)}, e_c^{(i)}, y^{(i)}, r^{(i)})\}_{i=1}^{K}$ the support set for product type t. We fit lightweight LoRA adapters [20] in the classification head and reasoning-projection layer while keeping the original student parameters frozen.

The adapted model is denoted $\theta_t = \theta_0 \oplus \phi_t$. The reasoning-guided adaptation objective is

$$\mathcal{L}_{\text{TTT}}(S_t) = \mathcal{L}_{\text{task}}(S_t) + \lambda_{\text{reason}} \frac{1}{K} \sum_{i=1}^{K} \left\| \hat{r}_i - r^{(i)} \right\|_2^2. \tag{5}$$

The label-only variant sets $\lambda_{\text{reason}} = 0$, whereas the reasoning-guided variant uses $\lambda_{\text{reason}} = 0.1$ and reuses the rationale-alignment signal from Eq. 3. We omit the contrastive objective during adaptation because the small, product-type-homogeneous support sets provide few reliable in-batch negatives. Algorithm 1 summarizes the complete adaptation-and-scoring procedure.

Algorithm 1 PT-TTT: test-time adaptation and scoring for product type t

```
# Frozen global student θ0; PT-specific adapter parameters φ
model = load_global_student()                          ▷ θ0 remains frozen
inject_lora(model, rank=8, alpha=16)
φ_init = save_adapter_initialization(model)

function ADAPT_AND_SCORE(support_t, query_t, use_reason)
    # Every PT starts from the same adapter initialization
    restore_adapter(model, φ_init)
    opt = AdamW(adapter_params(model), lr=1e-3)

    model.train()
    for step in 1..50 do
        logits, h_pair = model(support_t.base, support_t.cand)
        L = task_loss(logits, support_t.label)
        if use_reason then
            r_hat = alignment_proj(h_pair)
            L = L + 0.1 * mse(r_hat, support_t.reason_emb)
        end if
        opt.zero_grad()
        L.backward()
        opt.step()
    end for

    # Score unlabeled query pairs with the PT-adapted model
    model.eval()
    with no_grad():
        logits = model(query_t.base, query_t.cand)
        p = softmax(logits)[:, tradeup_class]
    return p                                           ▷ Discard φ_t; the next PT restarts from φ_init
end function
```

Each product type starts from the same global model and adapter initialization; adaptation is performed once per product type and amortized over all query pairs in that category. Additional LoRA parameterization and implementation details are provided in Appendix A.4.1.

5. Experiments

5.1. Setup

We conduct all experiments using the corpora described in Section 3. Specifically, the models are trained on 1,019,241 teacher-annotated (*silver*) pairs, partitioned using a stratified 90/10 training and validation split, and evaluated on a fixed human-annotated (*golden*) benchmark comprising 8,352 pairs. We compare shallow and deep student capacities; the shallow reasoning model contains 15.5M parameters and the corresponding deep model 65.9M. Exact layer configurations are provided in Appendix A.2.1.

Evaluation. Models and checkpoints are selected using the silver validation split; the 8,352-pair golden benchmark is held out for final evaluation. We report AUC and average precision (AP) as the primary threshold-independent metrics [25], together with F1, precision, and recall. Unless otherwise noted, thresholded metrics use operating points selected on validation data and applied unchanged to

Table 1
Golden benchmark results (n=8,352), with 95% bootstrap CIs on AUC. F1/P/R are evaluated on the golden benchmark using the best-F1 operating threshold selected on the validation set for each model. The shallow-model block provides a same-architecture comparison across label granularity (binary vs. four-class) and rationale supervision (label-only vs. +Reason). The best observed result is the *shallow* 15.5M reasoning-distilled four-class student. Row groups: teacher; shallow-architecture comparisons; larger/deep students.

Model	Params	AUC (95% CI) ↑	AP ↑	F1 ↑	Prec. ↑	Rec. ↑
LLM teacher *no demo*	—	—	—	0.724	0.918	0.597
LLM teacher *RAG, 5 demo*	—	—	—	0.749	0.967	0.610
Label-only (binary, shallow)	14.3M	0.911,[.905,.917]	0.916	0.834	0.812	0.857
+ Reason (binary, shallow)	15.5M	0.911,[.905,.918]	0.914	0.836	0.822	0.850
Label-only (4-class, shallow)	14.3M	0.912,[.906,.918]	0.916	0.836	0.820	0.853
+ Reason (4-class, shallow)	**15.5M**	**0.924**,[.918,.929]	**0.920**	**0.843**	0.829	**0.858**
Label-only (binary, deep)	64.7M	0.887,[.881,.893]	0.889	0.805	0.803	0.807
+ Reason (binary, deep)	65.9M	0.907,[.901,.913]	0.907	0.833	0.857	0.810
+ Reason (4-class, deep)	65.9M	0.911,[.905,.917]	0.902	0.832	0.806	0.860

the golden benchmark. Golden-set confidence intervals use 2,000 bootstrap resamples, with paired bootstrap resampling for model comparisons. Full evaluation and statistical details are provided in Appendix A.1.2.

5.2. Main Results

Table 1 reports performance on the fixed 8,352-pair golden benchmark for the LLM teacher, label-only students, and reasoning-distilled students. Figure 3 summarizes the principal comparison together with bootstrap confidence intervals. Additional results for global reasoning distillation are provided in Appendix A.3. Taken together, these results support three main conclusions.

(1) The best student is shallow and reasoning-distilled. The shallow four-class reasoning-distilled student achieves AUC 0.924 (95% CI $[0.918, 0.929]$), compared with 0.912 for the four-class label-only model using the same shallow architecture. It also exceeds the larger deep label-only model (AUC 0.887).

(2) Rationale and label granularity interact. Within the same shallow architecture, adding rationale supervision to the binary model leaves AUC unchanged at the reported precision (0.911 vs. 0.911), with only small changes in AP and thresholded metrics. In contrast, combining rationale supervision with four-class labels improves AUC from 0.912 to 0.924; using the unrounded predictions, the paired difference is $\Delta_{\mathrm{AUC}} = +0.013$ (95% CI $[+0.010, +0.016]$). Thus, the observed reasoning benefit depends on retaining the teacher's fine-grained relation structure.

(3) The student improves F1 relative to the evaluated teacher configuration. The retrieval-augmented teacher with five demonstrations achieves precision 0.967, recall 0.610, and F1 0.749. The distilled student reaches precision 0.829, recall 0.858, and F1 0.843 on the same benchmark, primarily by recovering recall. This comparison is specific to the evaluated retrieval-augmented teacher configuration and should not be interpreted as evidence that the student generally outperforms frontier LLMs.

5.3. PT-TTT

We use the best Level-1 model, the shallow 15.5M four-class reasoning-distilled student (AUC 0.924), as the frozen global student. We evaluate PT-TTT with support sizes $K \in \{4, 8, 16, 32\}$ under label-only and reasoning-guided adaptation. Implementation details are given in Section 4.3 and Appendix A.4.

Table 2
PT-TTT on the golden benchmark (n=8,352). K is the per-PT support size. The *global* baseline (K=0) corresponds to the frozen shallow 15.5M-parameter reasoning-distilled four-class student, evaluated without test-time adaptation, the best-performing model from Table 1, denoted + Reason (4-class). **Label-only** and **Reasoning** are the two adaptation objectives, fitting the *same* LoRA adapter without vs. with the rationale-alignment term (λ_{reason}=0 vs. > 0). Both improve threshold-independent AUC/AP monotonically over the global model; the lift is recall-driven, and the two objectives are near-identical (rationale adds little at test time). All rows report F1/precision/recall at a single fixed operating threshold (0.45) held constant across K, so the frozen global model and every adapted model are compared at the same operating point.

Adapt.	K	AUC ↑	AP ↑	F1 ↑	Prec. ↑	Rec. ↑
global	0	0.924	0.920	0.839	0.862	0.817
Label-only	4	0.929	0.924	0.839	0.841	0.837
	8	0.937	0.932	0.844	0.810	0.881
	16	0.940	0.936	0.860	0.844	0.876
	32	0.940	0.938	0.856	0.846	0.867
Reasoning	4	0.925	0.920	0.829	0.838	0.821
	8	0.937	0.931	0.847	0.818	0.878
	16	0.940	0.936	0.859	0.844	0.874
	32	**0.941** (+.017)	**0.940** (+.020)	**0.856** (+.017)	0.845 (–.017)	**0.867** (+.050)

Table 3
Matched human-supervision control for PT-TTT. The pooled and per-product-type adapters use the same expert-labeled support examples (K=32 per product type; 928 examples in total across 29 product types) and start from the same frozen global student. The pooled baseline fits one category-agnostic LoRA adapter to the union of the support examples, whereas PT-TTT independently fits one adapter per product type. F1/P/R use the same fixed threshold (0.45) as Table 2

Method	Total expert support	AUC ↑	Δ vs. global	AP ↑	F1 ↑	Prec./Rec. ↑
Global student	0	0.924	—	0.920	0.839	0.862 / 0.817
Pooled LoRA (label-only)	928	0.929	+0.005	0.926	0.845	0.852 / 0.839
PT-TTT (label-only)	928	0.940	+0.016	0.938	0.856	0.846 / 0.867
PT-TTT (+ Reason)	928	**0.941**	**+0.017**	**0.940**	0.856	0.845 / 0.867

Adapter Gains. PT-TTT improves the globally distilled student across the evaluated support budgets (Table 2). At K=32, reasoning-guided adaptation increases golden AUC from 0.924 to 0.941 (+0.017) and AP from 0.920 to 0.940 (+0.020). Label-only PT-TTT reaches nearly the same AUC (0.940). Performance largely plateaus between K=16 and K=32; the complete support-size curves are provided in Appendix A.4.2.

Matched human-supervision control. PT-TTT is directly optimized on expert-labeled support examples, whereas the global student is trained on LLM-generated silver supervision. To separate the effect of direct expert optimization from that of product-type specialization, we train a category-agnostic pooled LoRA adapter using exactly the same expert-labeled support examples as PT-TTT. At K=32, both approaches therefore use 928 support examples across the 29 product types. As shown in Table 3, pooled adaptation improves AUC from 0.924 to 0.929, whereas product-type-specific label-only adaptation reaches 0.940. Thus, direct optimization on expert labels explains part, but not all, of the PT-TTT improvement.

Within-product-type ranking control. Pooled AUC can improve if score distributions shift differently across product types, even when within-product-type discrimination remains unchanged. We therefore compute AUC independently within each product type and report the unweighted macro average across the 29 types. As shown in Table 4, macro PT-AUC increases from 0.910 for the global

Table 4
Pooled and within-product-type AUC. Pooled AUC is computed jointly over all golden benchmark pairs. Macro PT-AUC is the unweighted average of AUC computed independently within each product type. Improvement in Macro PT-AUC indicates improved within-product-type discrimination rather than only cross-category score rescaling.

Method	Pooled AUC ↑	Macro PT-AUC ↑	Δ Macro PT-AUC
Global student	0.924	0.910	—
Pooled LoRA (label-only)	0.929	0.914	+0.004
PT-TTT (label-only)	0.940	0.925	+0.015

student to 0.914 for pooled LoRA and 0.925 for PT-TTT. The larger within-type gain from PT-TTT indicates improved product-type-level discrimination rather than only cross-category score rescaling.

Rationale reuse during adaptation. Reasoning-guided and label-only PT-TTT perform similarly at moderate support sizes. At $K{=}8$ and $K{=}16$, their AUCs are identical at the reported precision, and at $K{=}32$ they reach 0.941 and 0.940, respectively. Thus, the Level-2 improvement appears to arise primarily from product-type-specific adaptation rather than from reusing rationale supervision during adaptation.

5.4. Scalability

As a small-scale proxy for full-catalog inference, we benchmark the distilled student on approximately 100K randomly sampled product pairs from the Amazon–Walmart dataset [21]. Even on this small-scale benchmark, running the distilled student on a single eight-GPU machine yields approximately a $5{,}000\times$ speedup and a $10{,}000\times$ reduction in estimated inference cost relative to direct LLM inference on the same workload. This scalability is enabled by separating training-time reasoning supervision from serving-time prediction: the rationale encoder, projection head, and auxiliary distillation objectives are used only during training, whereas inference requires only the compact pair-classification model and precomputed product embeddings. Figure 12 in Appendix A.5 additionally provides an illustrative customer-facing example showing how trade-up scores could be used to populate an "Upgrade your purchase" experience.

6. Discussion, Limitations, and Future Works

The component ablation further shows incremental silver-validation gains from rationale alignment and contrastive relational supervision (Appendix A.3.1).

Role of rationale supervision. Rationale supervision is not uniformly beneficial. Within the same shallow architecture, rationale supervision provides little improvement under collapsed binary supervision but produces a clear gain when paired with the four-class relation taxonomy (Section 5.2). This suggests that the auxiliary rationale signal is most useful when the task label space can preserve the fine-grained relational distinctions encoded by the teacher. The detailed objective ablations are provided in Appendix A.3.

Role of product-type adaptation. The Level-2 results support a category-specific adaptation effect beyond the general benefit of direct expert supervision. With the same 928 expert-labeled support examples, pooled LoRA improves AUC from 0.924 to 0.929, whereas product-type-specific adaptation reaches 0.940. Macro within-product-type AUC likewise increases from 0.910 to 0.925, indicating that the gain is not explained solely by cross-category score rescaling. At the same time, reasoning-guided and label-only PT-TTT are nearly identical at $K{=}32$ (0.941 vs. 0.940 AUC). Taken together, these

results suggest a division of labor: rationale-guided distillation is most useful for shaping the global student, whereas the Level-2 gain comes primarily from product-type-specific supervised adaptation.

Limitations and future work. Our evidence remains limited in several ways. First, evaluation covers 29 product types and does not establish generalization to unseen categories. Second, PT-TTT requires expert-labeled support examples and an explicit optimization step for each product type; performance may also depend on support-set composition and optimization randomness. Third, although the pooled-adapter and macro PT-AUC controls address two alternative explanations for the observed gain, we do not fully separate product-type adaptation from simpler category-specific calibration. Fourth, the current support/golden split is pair-disjoint but not explicitly product- or brand-disjoint; stricter entity-disjoint evaluation is therefore an important direction for future work. Fifth, the golden benchmark is designed for controlled model comparison and may have a substantially different class prevalence from production candidate streams; accordingly, its precision and F1 should not be interpreted directly as production positive predictive value. Finally, the LLM comparison reflects the fixed retrieval-augmented teacher configuration evaluated in this study rather than an exhaustive optimization over prompting, retrieval, or demonstration count. Future work should evaluate unseen-type transfer, simpler calibration and product-type-conditioned baselines, stricter entity-disjoint evaluation, and amortized adaptation that avoids per-type gradient descent.

7. Conclusion

We introduced a two-level framework for scalable trade-up identification that transfers black-box LLM supervision into an efficient embedding-pair student and then specializes it through product-type test-time training (PT-TTT). Rationale-guided global distillation reaches AUC 0.924 on the held-out human benchmark, while PT-TTT improves the model to AUC 0.941 without serving-time LLM inference. The results indicate complementary roles for the two levels: fine-grained rationale supervision is most useful during global representation learning, whereas product-type-specific expert supervision provides most of the subsequent adaptation gain. This separation enables LLM-derived semantic supervision to be used for large-scale catalog scoring without requiring generative inference at serving time.

References

[1] J. McAuley, R. Pandey, J. Leskovec, Inferring networks of substitutable and complementary products, in: Proceedings of the 21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2015, pp. 785–794. URL: https://doi.org/10.1145/2783258.2783381. doi:10.1145/2783258.2783381.

[2] P. Herrero-Vidal, Y.-L. Chen, C. Liu, P. Sen, L. Wang, Learning variant product relationship and variation attributes from e-commerce website structures, in: Workshop on Generative AI for E-Commerce (GenAIECommerce) at CIKM, 2024.

[3] C. Yamasaki, K. Sugahara, K. Okamoto, Knowledge-augmented relation learning for complementary recommendation with large language models, in: Workshop on Generative AI for E-Commerce (GenAIECommerce) at ACM RecSys, 2025.

[4] S. Geng, S. Liu, Z. Fu, Y. Ge, Y. Zhang, Recommendation as language processing: A unified pretrain, personalized prompt and predict paradigm, in: Proceedings of the 16th ACM Conference on Recommender Systems, 2022. URL: https://doi.org/10.1145/3523227.3546767. doi:10.1145/3523227.3546767.

[5] K. Bao, J. Zhang, Y. Zhang, W. Wang, F. Feng, X. He, Tallrec: An effective and efficient tuning framework to align large language model with recommendation, in: Proceedings of the 17th ACM Conference on Recommender Systems (RecSys), 2023.

[6] Y. Xi, W. Liu, J. Lin, J. Zhu, B. Chen, R. Tang, W. Zhang, R. Zhang, Y. Yu, Towards open-world

recommendation with knowledge augmentation from large language models, in: Proceedings of the 18th ACM Conference on Recommender Systems (RecSys), 2024.
[7] L. Wu, Z. Zheng, Z. Qiu, H. Wang, H. Gu, T. Shen, C. Qin, C. Zhu, H. Zhu, Q. Liu, H. Xiong, E. Chen, A survey on large language models for recommendation, World Wide Web 27 (2024).
[8] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, S. Riedel, D. Kiela, Retrieval-augmented generation for knowledge-intensive NLP tasks, in: Advances in Neural Information Processing Systems, volume 33, 2020, pp. 9459–9474. URL: https://proceedings.neurips.cc/paper/2020/hash/6b493230205f780e1bc26945df7481e5-Abstract.html.
[9] S. Liu, R. Suresh, A. Banitalebi-Dehkordi, Beauty beyond words: Explainable beauty product recommendations using ingredient-based product attributes, in: Proceedings of the 18th ACM Conference on Recommender Systems, Workshop on Strategic and Utility-aware REcommendation (SURE), 2024. ArXiv:2409.13628.
[10] G. Hinton, O. Vinyals, J. Dean, Distilling the knowledge in a neural network, arXiv preprint arXiv:1503.02531 (2015).
[11] A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, Y. Bengio, FitNets: Hints for thin deep nets, in: International Conference on Learning Representations, 2015. URL: https://arxiv.org/abs/1412.6550.
[12] Y. Tian, D. Krishnan, P. Isola, Contrastive representation distillation, in: International Conference on Learning Representations, 2020. URL: https://arxiv.org/abs/1910.10699.
[13] C.-Y. Hsieh, C.-L. Li, C.-K. Yeh, H. Nakhost, Y. Fujii, A. Ratner, R. Krishna, C.-Y. Lee, T. Pfister, Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes, in: Findings of the Association for Computational Linguistics: ACL 2023, 2023, pp. 8003–8017. URL: https://arxiv.org/abs/2305.02301.
[14] P. Wang, Z. Wang, Z. Li, Y. Gao, B. Yin, X. Ren, SCOTT: Self-consistent chain-of-thought distillation, in: Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics, 2023. URL: https://aclanthology.org/2023.acl-long.304/.
[15] M. Turpin, J. Michael, E. Perez, S. R. Bowman, Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting, in: Advances in Neural Information Processing Systems, volume 36, 2023. URL: https://proceedings.neurips.cc/paper_files/paper/2023/hash/ed3fea9033a80fea1376299fa7863f4a-Abstract-Conference.html.
[16] Y. Sun, X. Wang, Z. Liu, J. Miller, A. A. Efros, M. Hardt, Test-time training with self-supervision for generalization under distribution shifts, in: Proceedings of the 37th International Conference on Machine Learning, volume 119 of *Proceedings of Machine Learning Research*, 2020, pp. 9229–9248. URL: https://arxiv.org/abs/1909.13231.
[17] D. Wang, E. Shelhamer, S. Liu, B. Olshausen, T. Darrell, Tent: Fully test-time adaptation by entropy minimization, in: International Conference on Learning Representations, 2021. URL: https://arxiv.org/abs/2006.10726.
[18] C. Finn, P. Abbeel, S. Levine, Model-agnostic meta-learning for fast adaptation of deep networks, in: Proceedings of the 34th International Conference on Machine Learning, volume 70 of *Proceedings of Machine Learning Research*, 2017, pp. 1126–1135. URL: https://proceedings.mlr.press/v70/finn17a.html.
[19] E. Akyürek, M. Damani, L. Qiu, H. Guo, Y. Kim, J. Andreas, The surprising effectiveness of test-time training for abstract reasoning, arXiv preprint arXiv:2411.07279 (2024).
[20] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen, LoRA: Low-rank adaptation of large language models, in: International Conference on Learning Representations, 2022. URL: https://arxiv.org/abs/2106.09685.
[21] Hasso Plattner Institute, Amazon–walmart dataset, https://hpi.de/naumann/projects/repeatability/datasets/amazon-walmart-dataset.html, 2026. Accessed: 2026-07-19.
[22] T.-Y. Lin, P. Goyal, R. Girshick, K. He, P. Dollár, Focal loss for dense object detection, in: Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2017.
[23] A. van den Oord, Y. Li, O. Vinyals, Representation learning with contrastive predictive coding,

arXiv preprint arXiv:1807.03748 (2018).
[24] W. Park, D. Kim, Y. Lu, M. Cho, Relational knowledge distillation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019.
[25] V. W. Anelli, A. Bellogín, A. Ferrara, D. Malitesta, F. A. Merra, C. Pomo, F. M. Donini, T. Di Noia, Elliot: A comprehensive and rigorous framework for reproducible recommender systems evaluation, in: Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR), 2021.
[26] S. Xiao, Z. Liu, P. Zhang, N. Muennighoff, C-pack: Packaged resources to advance general Chinese embedding, 2023. arXiv:2309.07597.
[27] N. Ho, L. Schmid, S.-Y. Yun, Large language models are reasoning teachers, in: Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL), 2023.

A. Supplementary Details and Results

A.1. Data, Teacher, and Evaluation Protocol

A.1.1. Teacher Retrieval and Rationale Encoding

The teacher is a frontier LLM accessed as a black box; the interface exposes only the generated relation label and rationale text, without logits or hidden states.

For each candidate pair, the textual information of the base product b and candidate product c is concatenated and encoded using the frozen `bge-base-en-v1.5` encoder [26]. Query pairs are encoded using the same procedure. Retrieval is restricted to demonstrations from the corresponding product type, and the five examples with smallest Euclidean distance to the query representation are selected irrespective of their relation labels. Retrieved demonstrations are constrained to be distinct from the queried pair.

The retrieved examples, their expert annotations, the base and candidate product descriptions, and product-type-specific trade-up criteria are provided to the teacher as few-shot context. The teacher emits one four-class relation label and a concise free-text rationale, limited to at most two sentences.

Teacher rationales are encoded once offline using the frozen `bge-base-en-v1.5` encoder into $d_r = 768$ dimensional vectors and stored with the silver training data. The same human-annotated demonstration pool is subsequently used as the support-sampling pool for PT-TTT.

A.1.2. Dataset Splits and Evaluation Statistics

The silver corpus contains $1{,}019{,}241$ teacher-annotated product pairs, partitioned into training and validation sets using a stratified 90/10 split over product types and relation classes.

The human *golden* benchmark contains $8{,}352$ pairs from 29 product types and is pair-disjoint from both the silver corpus and the teacher's few-shot demonstration/support pool. Silver-validation metrics are used for model, checkpoint, and hyperparameter selection. The resulting fixed checkpoint is evaluated on the held-out golden benchmark, which is used only for final evaluation and analysis.

Confidence intervals are estimated using a nonparametric bootstrap with $2{,}000$ resamples of the golden benchmark. Pairwise model comparisons use a paired bootstrap with identical resample indices across models, reporting the mean difference, its 95% confidence interval, and $P(\Delta > 0)$.

AUC and AP are threshold-independent. F1, precision, and recall are reported using the best-F1 operating threshold selected on the silver validation set for each model, unless otherwise noted.

A.2. Student Architecture and Training Details

A.2.1. Architecture and Hyperparameters

The student operates on one precomputed $d = 768$ dimensional embedding per product. The base and candidate embeddings are processed by separate branches:

$$h_b^{(0)} = e_b, \qquad h_b^{(\ell)} = \mathrm{SelfAttn}_b^{(\ell)}\left(h_b^{(\ell-1)}\right), \qquad \ell = 1, \ldots, L_s, \tag{6}$$

$$h_c^{(0)} = e_c, \qquad h_c^{(\ell)} = \mathrm{SelfAttn}_c^{(\ell)}\left(h_c^{(\ell-1)}\right), \qquad \ell = 1, \ldots, L_s. \tag{7}$$

The two branches use separate parameters, allowing side-specific transformations for the base and candidate products. Because each product is represented by a single embedding, these self-attention modules operate on length-one sequences. The attention softmax therefore contains a single element and is identically one; in this implementation, the blocks act as parameterized transformations rather than content-dependent token-selection mechanisms.

In parallel, an ordered base-to-candidate branch is defined as

$$z^{(0)} = e_b, \tag{8}$$

$$z^{(m)} = \mathrm{CrossAttn}^{(m)}\left(z^{(m-1)}, e_c\right), \qquad m = 1, \ldots, L_x. \tag{9}$$

Because both query and key/value sides contain one embedding, the cross-attention softmax is likewise identically one and does not perform content-dependent selection. The branch instead contributes an additional parameterized transformation within the ordered base-to-candidate architecture. Directionality of the overall classifier arises from the asymmetric base/candidate roles, separate branch parameters, and ordered concatenation.

The branch outputs are concatenated as

$$h_{\mathrm{pair}} = \left[h_b^{(L_s)}; h_c^{(L_s)}; z^{(L_x)}\right] \in \mathbb{R}^{3d}. \tag{10}$$

The prediction MLP has hidden dimensions $[512, 256, 128]$ with ReLU activations and dropout 0.3, followed by a final linear layer producing one logit in binary mode or four logits in multiclass mode. Self- and cross-attention modules use four heads with dropout 0.1.

For rationale-guided distillation, a linear projection maps the pair representation into the $d_r = 768$ rationale-embedding space:

$$\hat{r}_i = W_{\mathrm{align}} h_{\mathrm{pair}}^{(i)} + b_{\mathrm{align}}. \tag{11}$$

This projection is used only during training and is removed at inference. Trainable weights are initialized with Xavier-uniform initialization.

We evaluate two model capacities. The shallow configuration uses one self-attention layer in each side-specific branch and one cross-attention layer (14.3M parameters label-only; 15.5M with the reasoning projection). The deep configuration uses five layers of each (64.7M label-only; 65.9M with the reasoning projection).

A.2.2. Loss Definitions and Optimization Details

Binary task loss. Let $p_i = \sigma(g_i)$. The weighted focal binary objective is

$$\mathcal{L}_{\mathrm{bin}} = -\frac{1}{N}\sum_{i=1}^{N}\left[w_{\mathrm{pos}} y_i^{(2)}(1-p_i)^{\gamma}\log p_i + (1-y_i^{(2)})p_i^{\gamma}\log(1-p_i)\right], \tag{12}$$

with $\gamma = 2.0$ and validation-selected positive-class weight w_{pos}.

Four-class task loss. The weighted cross-entropy objective is

$$\mathcal{L}_{\mathrm{multi}} = -\frac{1}{N}\sum_{i=1}^{N} w_{y_i}\log\frac{\exp(g_{i,y_i})}{\sum_{k=1}^{4}\exp(g_{i,k})}. \tag{13}$$

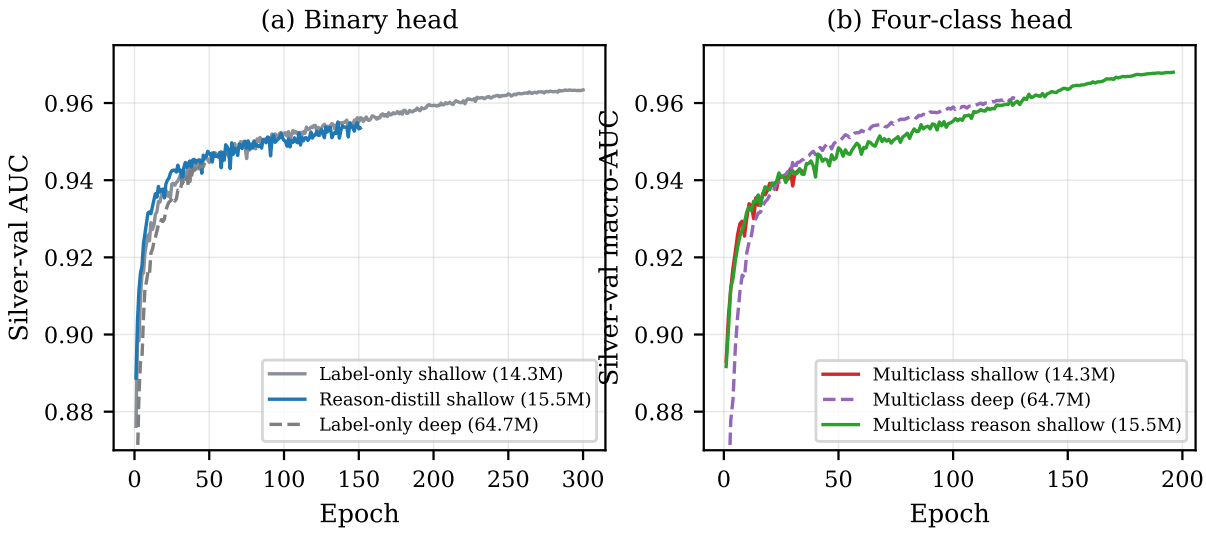

Figure 2: Per-epoch silver-validation AUC. (a) Binary head: shallow label-only and reasoning-distilled models track closely, while the deep model performs worse. (b) Four-class head: the deeper model performs better.

Contrastive distillation. Let $\tilde{r}_i$ and $\tilde{r}_i^t$ denote the ℓ_2-normalized student projection and teacher rationale embedding, respectively. We use

$$\mathcal{L}_{\text{infonce}} = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{\exp(S_{ii})}{\sum_{j=1}^{N}\exp(S_{ij})}, \qquad S_{ij} = \frac{\tilde{r}_i^\top \tilde{r}_j^t}{\tau}. \tag{14}$$

The matched pair is treated as positive and the remaining rationale embeddings in the batch as negatives. A relational KL objective transfers pairwise similarity structure:

$$\mathcal{L}_{\text{kl}} = T^2 D_{\text{KL}}\left(P^t \,\|\, P^s\right). \tag{15}$$

The complete contrastive term is

$$\mathcal{L}_{\text{con}} = \mathcal{L}_{\text{infonce}} + \lambda_{\text{kl}}\mathcal{L}_{\text{kl}}. \tag{16}$$

We use $\lambda_{\text{align}} = 0.1$, $\lambda_{\text{con}} = 0.01$, $\tau = 0.07$, $\lambda_{\text{kl}} = 0.05$, and $T = 1$.

A.2.3. Optimization Details

We use AdamW with learning rate 10^{-4}, weight decay 10^{-2}, and $\beta = (0.9, 0.999)$, with cosine learning-rate annealing to 10^{-5}, automatic mixed precision, and gradient-norm clipping at 1.0. Training uses early stopping on the monitored validation metric with patience 10.

The binary task objective uses focal BCE with $\gamma = 2.0$ and the positive-class weight used in our experiments, while the four-class task objective uses weighted cross-entropy. Reasoning-distillation hyperparameters are $\lambda_{\text{align}} = 0.1$, $\lambda_{\text{con}} = 0.01$, $\tau = 0.07$, $\lambda_{\text{kl}} = 0.05$, and $T = 1$.

A.2.4. Training Curves

Figure 2 shows per-epoch silver-validation curves for the binary and four-class heads across shallow and deep configurations.

A.3. Additional Level-1 Distillation Results

This appendix collects the supporting Level-1 figures deferred from Section 5.2 for space. Figure 3 is the headline bar chart (Table 1) with bootstrap CIs; Figure 4 contrasts the student and the conservative

teacher on the golden set; Figure 5 shows golden ROC/PR curves; Figure 6 is the silver→golden generalization gap; Figure 7 is the per-product-type breakdown; and Figure 8 shows operating-point selection and calibration.

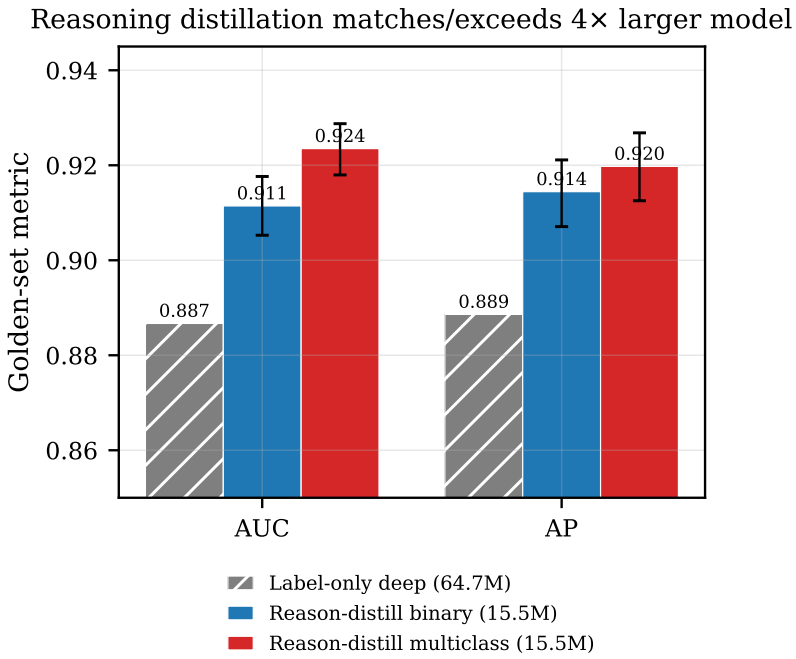

Figure 3: Golden-set AUC/AP with 95% bootstrap CIs. The shallow 15.5M reasoning-distilled multiclass student (red) exceeds the 4× larger deep label-only student (hatched grey), with non-overlapping CIs.

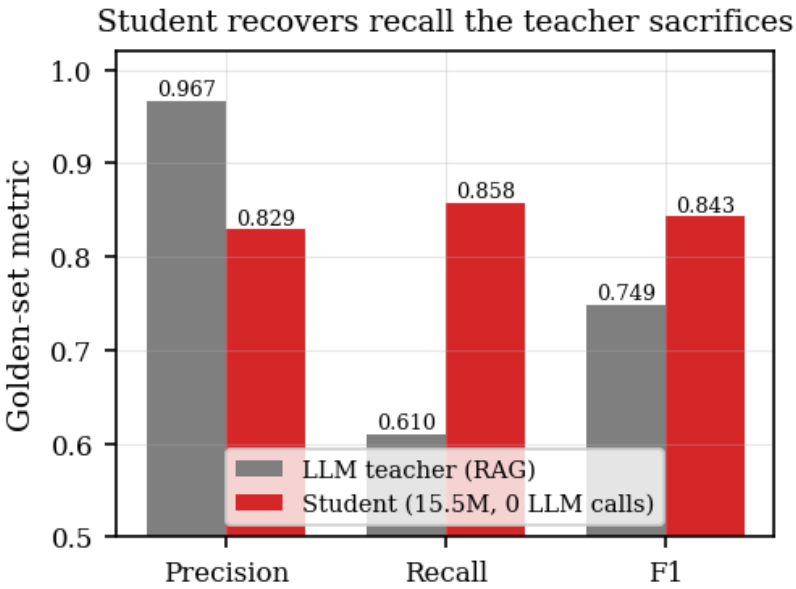

Figure 4: Student vs. teacher on the golden set. The conservative teacher trades recall for precision; the 15.5M student recovers recall (and a higher F1) at a modest precision cost, with zero LLM calls at inference.

A.3.1. Distillation Objective Ablation

The distillation objective: aligning *and* contrasting rationale embeddings. Our Level-1 objective is a core component of the system. Since the production student is embedding-only and never decodes text, we cannot supervise it on rationale *tokens* as generative rationale-distillation methods

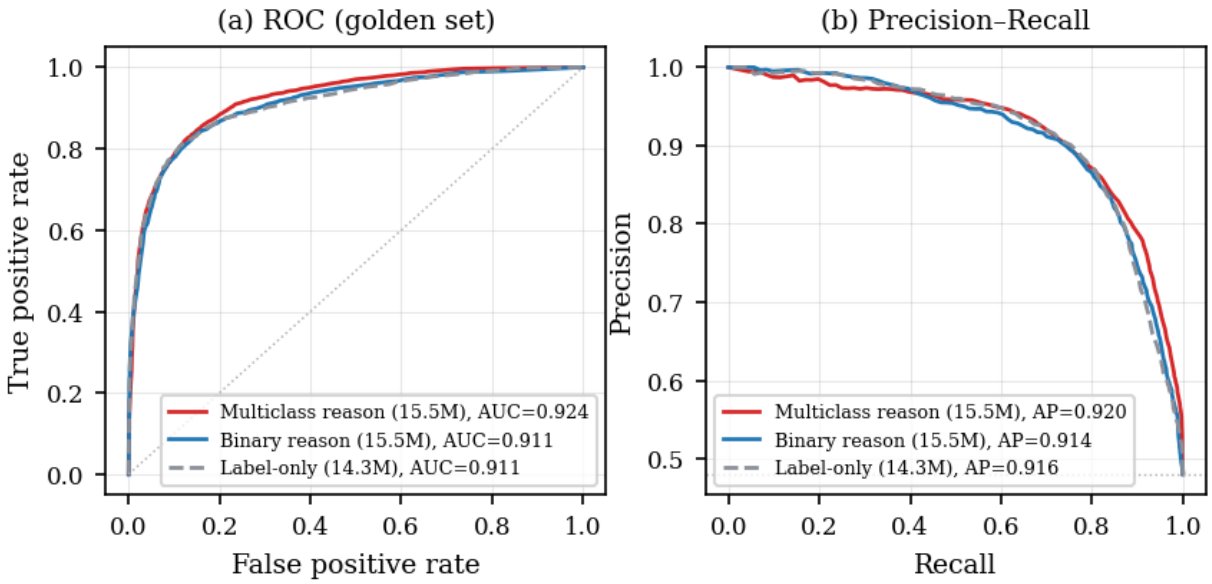

Figure 5: Golden-set ROC (a) and Precision–Recall (b). The shallow multiclass reasoning-distilled student (red) dominates the binary reason and label-only baselines across operating points.

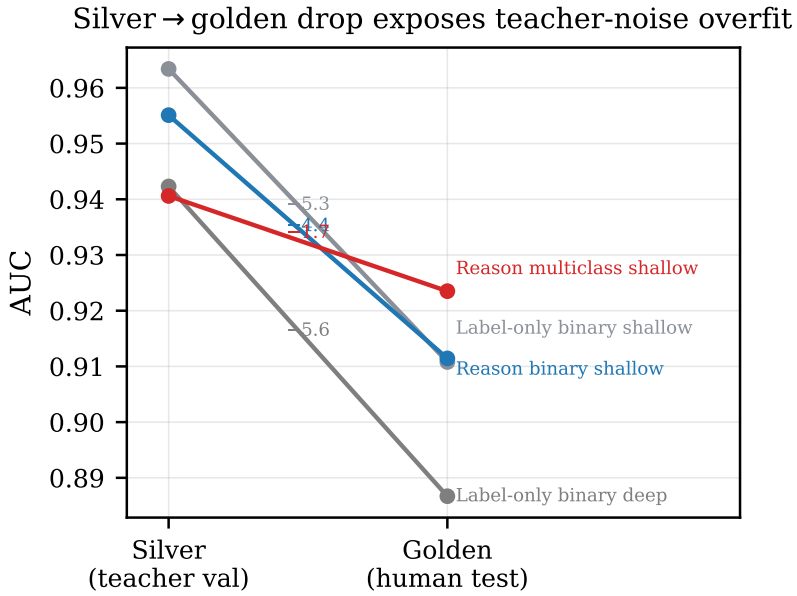

Figure 6: Silver-validation vs. golden-test AUC. The deep binary label-only model wins on silver but drops most on golden—the signature of overfitting teacher-label noise.

do [13, 27]. We instead transfer the rationale through three complementary channels on the pair representation. The *alignment* loss [11] trains the student's projected representation to predict the teacher's rationale embedding, providing a per-example target that captures the evidence relevant to each product pair. But pointwise alignment alone under-constrains the geometry: two pairs with different trade-up reasons can be pulled toward nearby rationale vectors. The *contrastive* InfoNCE term [23, 12] repairs this by making each pair's representation discriminate its own rationale from the other rationales in the batch, preserving relational structure; the *relational-KL* term [24, 10] further matches the teacher's full pairwise-similarity distribution rather than just the top match. The per-term ablation (Figure 9) confirms the channels are additive: silver AUC climbs monotonically $0.956 \rightarrow 0.964 \rightarrow 0.968$ (AP $0.947 \rightarrow 0.956 \rightarrow 0.961$) as alignment and then contrastive are added, so each earns its place. All three components are used only during training and removed at inference. As

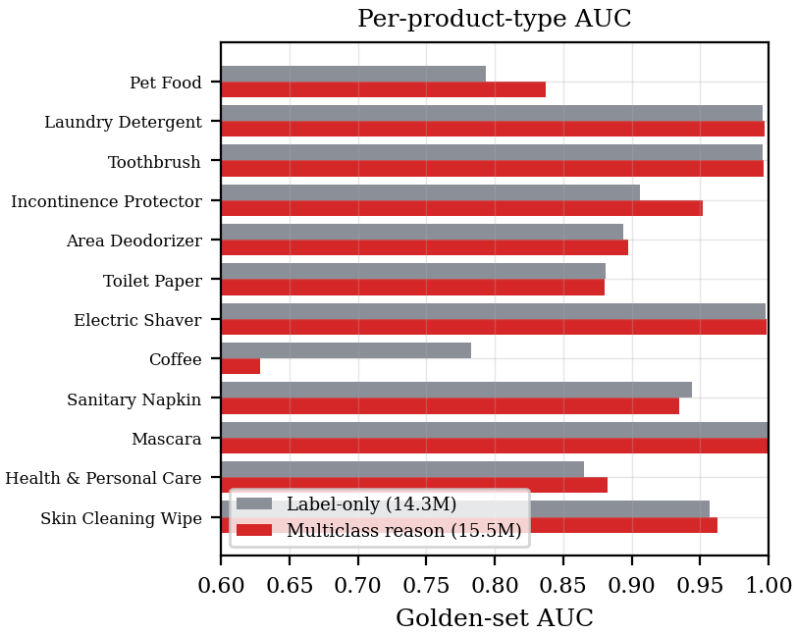

Figure 7: Per-product-type golden AUC (types with $\geq$120 pairs in golden benchmark dataset). The multiclass reasoning-distilled student (red) matches or beats the label-only baseline in nearly all types; gains are largest where trade-up evidence is subtle.

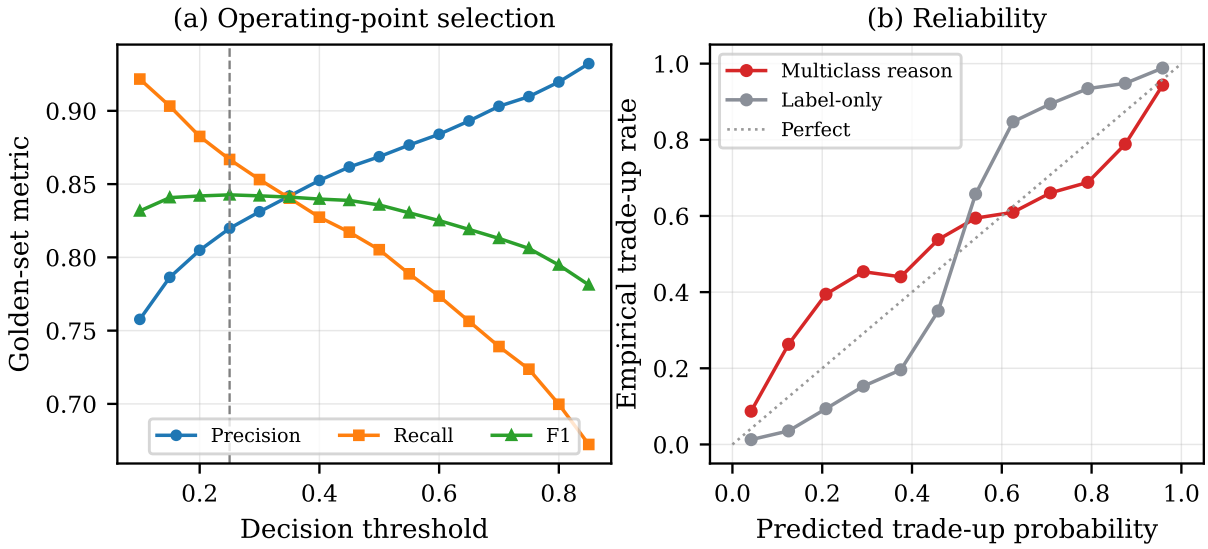

Figure 8: (a) Golden-set precision/recall/F1 vs. threshold for the multiclass student. (b) Reliability diagram: neither student is perfectly calibrated a priori, motivating post-hoc temperature scaling before threshold selection.

a result, reasoning transfer introduces no serving overhead; the deployed student retains only the improved pair representation.

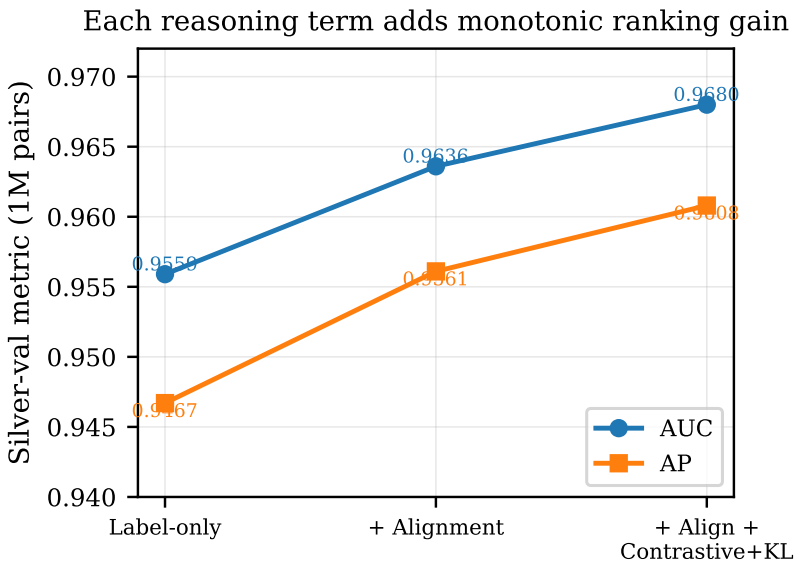

Figure 9: Per-term distillation ablation on a 1M-pair dataset; metrics on the silver-validation portion): adding the alignment loss, then the contrastive (InfoNCE+KL) loss, each yields a monotonic ranking-quality gain (AUC $0.956 \rightarrow 0.964 \rightarrow 0.968$).

A.4. Product-Type Test-Time Training Details

A.4.1. Adapter Parameterization and Support Protocol

For each adapted linear layer W, LoRA parameterizes the update as

$$W' = W + \frac{\alpha}{\rho} BA, \qquad B \in \mathbb{R}^{d_{\text{out}} \times \rho}, \quad A \in \mathbb{R}^{\rho \times d_{\text{in}}}. \tag{17}$$

We use rank $\rho = 8$ and scaling factor $\alpha = 16$. Adapters are inserted into the linear layers of the classification head and the reasoning-projection layer W_{align}, for approximately 67K trainable parameters in total. The original global-student parameters remain frozen.

Support sets are sampled in a class-balanced manner from the expert-annotated support pool. Each product type is adapted independently from the same zero-delta LoRA initialization. The label-only and reasoning-guided variants use identical support examples and adapter architecture, differing only in whether the rationale-alignment term is included.

A.4.2. Support-Size Sensitivity

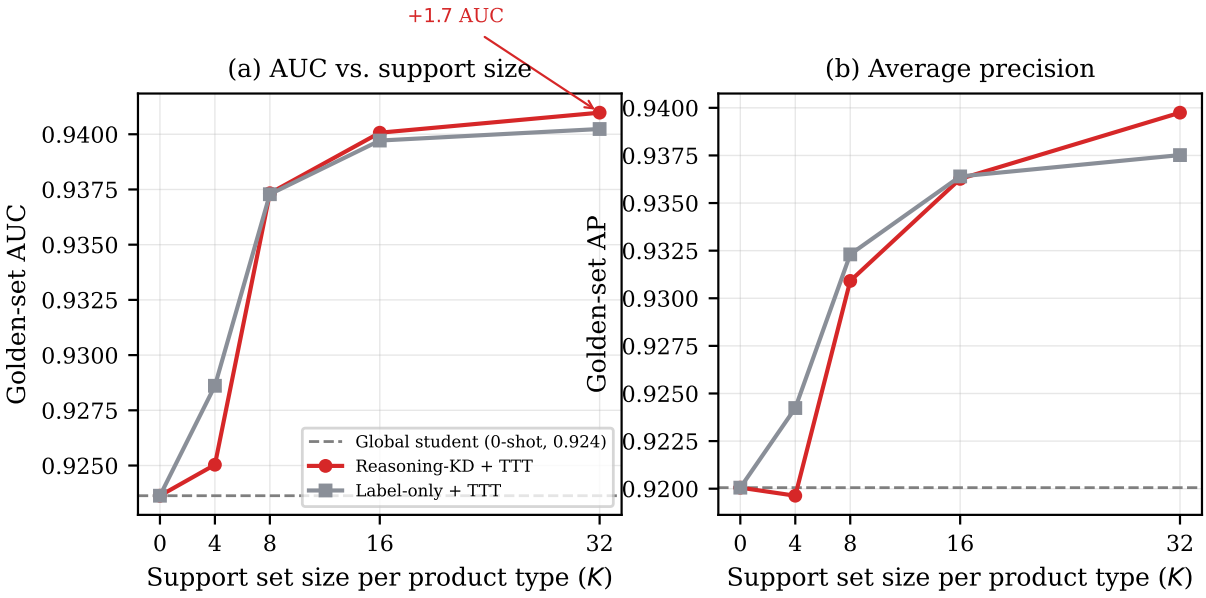

Figure 10: PT-TTT vs. support set size K. The global student without adaptation (K=0) is the dashed baseline. AUC (a) and AP (b) rise monotonically to $+0.017$/$+0.020$ at K=32.

A.4.3. Adaptation-Cost Scaling

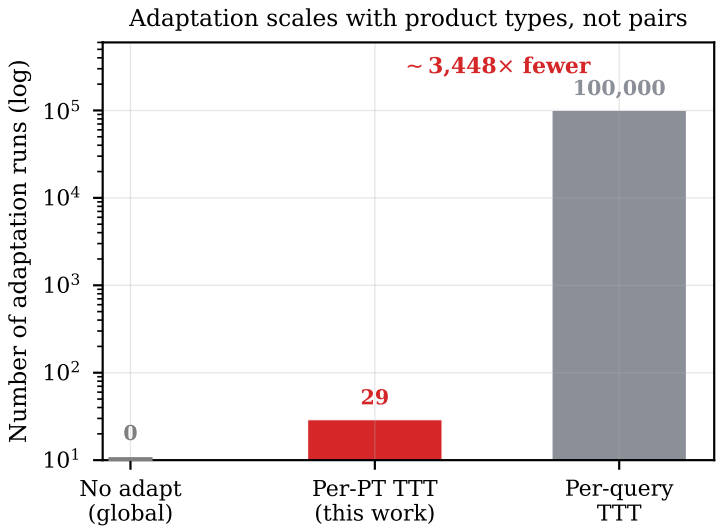

Figure 11: Adaptation-cost scaling. For an illustrative workload of Q=100,000 query pairs across P=29 product types, PT-TTT requires P=29 adapter fits, whereas per-query adaptation requires Q=100,000 fits (3,448× more). Thus, PT-TTT adaptation scales with product types rather than query pairs.

A.5. Scalability

To construct the customer-facing recommendation widget, we first group candidate pairs by base product and compute a trade-up score for each associated candidate offline. The resulting candidates are then reranked according to the desired presentation strategy. For example, candidates may be ordered by increasing trade-up strength to present more accessible upgrades first, or reranked using additional signals such as star rating, customer relevance, or price. As illustrated in Figure 12, the customer is viewing a powder laundry detergent. The "Upgrade your purchase" widget presents ranked

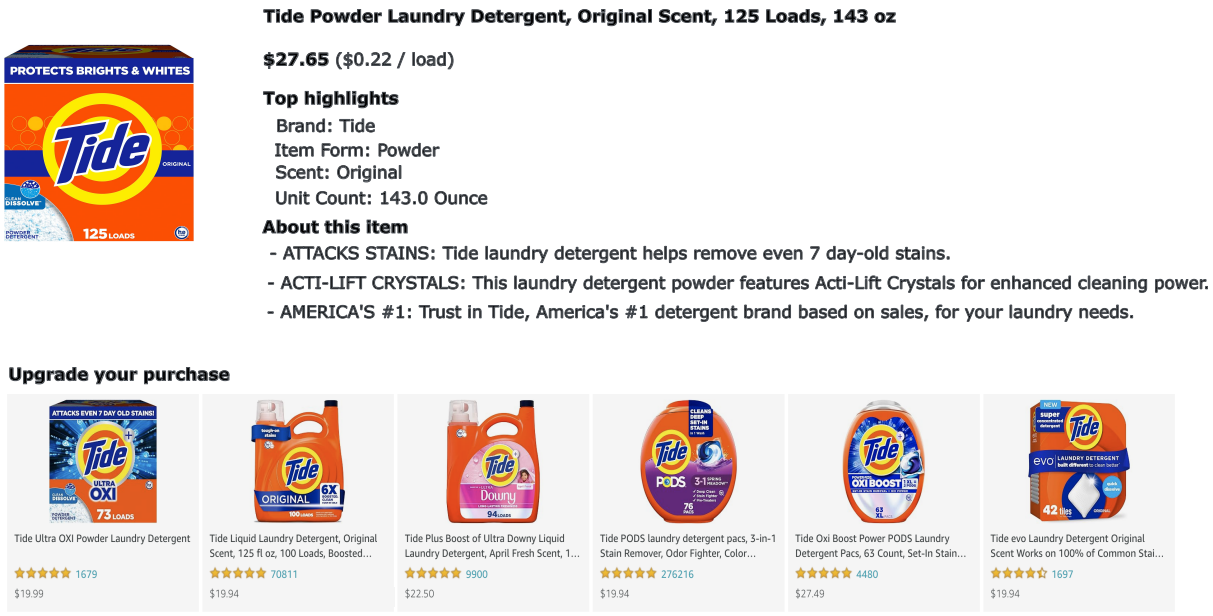

Figure 12: Customer-facing recommendation widget. Trade-Up candidate products are ranked by their predicted trade-up scores and presented as trade-up alternatives for the base product.

trade-up alternatives spanning several product formats, including enhanced-cleaning powders, liquid detergents, unit-dose pods, Power PODS, and EVO laundry tiles. All recommendations preserve the same underlying shopping mission, namely, laundering clothes, while offering potential improvements in cleaning performance, dosing convenience, formulation concentration, or sustainability-related product attributes. For example, liquid detergents may provide easier pretreatment and dispensing, whereas pods and tiles offer pre-measured dosing and reduced handling.