MATAR: MULTI-AGENT TASK-ADAPTIVE REFINEMENT FOR MULTIMODAL DATA

Anonymous authors
Paper under double-blind review

ABSTRACT

The utility of multimodal data depends on the downstream task, yet data preparation often relies on generic quality criteria, sample selection, or fixed refinement procedures. These approaches leave limited scope to adapt record attributes and inter-record relations to task requirements. We propose MATAR, a task-adaptive multi-agent framework that uses downstream evidence to optimize executable refinement programs. Attribute and relation programs are searched independently, with task feedback selecting one final output. Task-adaptive tool optimization revises program structure and parameters and generates validated numerical feature operations when existing tools are insufficient. Paired replay evaluation compares revisions under matched training conditions, while experiential policy adaptation uses recent outcomes and an online controller to guide subsequent proposals with frozen LLM parameters. Across eight datasets spanning four downstream tasks, MATAR achieves an average relative gain of 23.9% over the strongest baseline per dataset under the evaluated protocols. Further experiments support the contributions of the three components and show that experience encoded in refinement programs can benefit other datasets. The findings suggest that effective data refinement should consider both attribute calibration and relational structure, with task-specific evaluation guiding their use and transfer.

1 INTRODUCTION

Product catalogs pair items with images, descriptions, and metadata, while content platforms combine visual posts, text, and interaction records. These sources provide complementary evidence for classification, cross-modal retrieval, and recommendation, but their quality depends on the information required by each downstream task. Web-derived image-text collections may contain mismatched descriptions, redundant examples, and uneven semantic coverage, motivating data curation rather than indiscriminate use (Cheng et al., 2026; Shechter & Carmon, 2025). Generic quality measures do not fully capture this dependence, as data sufficient for coarse classification may lack the fine-grained distinctions required for retrieval. Preparing data for a target task therefore requires considering not only which examples to retain, but also which attributes to expose and how to organize records. We study multimodal data refinement, which modifies record attributes or constructs and revises inter-record relations according to downstream task requirements.

Challenges. ❶ **Task-dependent data utility.** Generic completeness or image-text agreement measures can miss distinctions needed for retrieval but not coarse classification. Task-aware selection uses downstream feedback to score or select examples (Shechter & Carmon, 2025; Cheng et al., 2026), but does not specify how to edit attributes or relations to meet task requirements. Data quality should therefore be assessed through downstream performance, with refinement guided by the information needed for the target task. ❷ **Limited relational refinement.** Sample selection leaves attributes and topology unchanged (Cheng et al., 2026; Shechter & Carmon, 2025), and deduplication or diversity clusters remain unexposed as editable structure. Generic relations can contextualize ambiguous records but may not suit the task. Graph methods correct edges using cross-modal evidence (Xue & Wu, 2026) or suppress shortcut propagation (Wang et al., 2026b), but retain corrections within the model. Relations should therefore be exposed as revisable data artifacts and evaluated according to their utility for the target task. ❸ **Rigid refinement procedures.** Hand-designed pipelines fix operations, order, and conditions, while learned score combinations remain limited to sample scoring (Shechter & Carmon, 2025). Optimizing a procedure against generic

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

quality proxies does not establish whether its edits improve performance on the target task. Revising downstream inputs requires matched task evaluations, yet scores do not explain when changes help or how to guide decisions. Adaptation therefore requires revisable procedures, paired evaluation, and contextual experience.

In this work, we propose a task-adaptive multi-agent framework for multimodal data refinement that uses downstream evidence to guide interventions on record attributes or inter-record relations.

Our approach. Our framework, MATAR, coordinates task analysis, data diagnosis, intervention construction, and feedback interpretation to align refinement with downstream utility. Attribute programs construct or transform feature views, and relation programs construct and revise proposed edges in a separate search path. **Task-adaptive tool optimization** revises executable Code-as-Rule programs within each path and generates validated numerical feature operations when the available tools are insufficient, making the refinement procedure the object of search (Yue et al., 2026). **Paired replay evaluation** compares parent and revised procedures under matched training conditions using evaluation protocols designed for different tasks. **Experiential policy adaptation** combines recent Search experience with an online proposal controller, keeping LLM parameters fixed (Gao et al., 2026; Huang et al., 2026a). Structured feedback links measured outcomes to subsequent decisions (Feng et al., 2026). The attribute and relation searches produce separate finalists, from which Search selects one candidate for Commit evaluation with Preserve as the fallback.

Contributions. ❶ We formulate multimodal data refinement as task-guided optimization of executable attribute and relation programs in separate search paths. This formulation makes both feature views and inter-record structure editable, extending data preparation beyond sample selection. ❷ We propose MATAR, a multi-agent framework that integrates task-adaptive tool optimization, paired replay evaluation, and experiential policy adaptation. The framework revises refinement procedures and generates validated numerical feature operations, using paired downstream measurements and recent experience to guide subsequent proposals while keeping LLM parameters fixed. ❸ Across eight datasets spanning four downstream tasks, MATAR achieves an average relative gain of 23.9% over the strongest baseline per dataset. Our experiments suggest that effective refinement should consider both attribute calibration and inter-record structure, with task-specific evaluation guiding their use and cross-dataset transfer.

2 Related Work

Task-aware data selection. EcoDatum scores quality through weak supervision (Xu et al., 2025), DataTailor balances informativeness, uniqueness, and representativeness (Yu et al., 2025), TAROT matches distributions via optimal transport (Feng et al., 2025), and FLYT learns task-based scoring and controls exposure (Shechter & Carmon, 2025). TADS combines quality, relevance, and diversity (Cheng et al., 2026), ZPD matches difficulty to capability (Yang et al., 2026a), and ARDS selects perturbation-informed examples for robustness (Yang et al., 2025). Filtering benefits vary with compute (Mohri et al., 2026). These methods optimize samples rather than attributes or relations.

Task-directed data refinement. MM-Gen generates and filters annotations (Joshi et al., 2025), COMET prioritizes cleaning through corruption-based estimates (Mohammed et al., 2025), and DataOrchestra learns to discard, preserve, or clean text (Huang et al., 2026b). Ranked curation addresses modality gaps (Muthukumar et al., 2026), while audits distinguish quality scores from downstream utility (Zhang et al., 2026). ReCoG refines edges during graph learning (Xue & Wu, 2026), while GOMA controls post-encoder graph propagation without diagonal cross-modal shortcuts (Wang et al., 2026b). We instead revise executable attribute or relation programs using task feedback, with unchanged inputs as a valid outcome.

Agentic curation and executable program optimization. Data-Juicer Sandbox supports refinement feedback (Chen et al., 2025), DataFlow-Agent retrieves and repairs operators (Liang et al., 2025), and TabClean synthesizes repairs from cell-level feedback (Wang et al., 2026c). Kang et al. (2026) study coding-based curation, CurateEvo evolves policies from development-set failures (Wang et al., 2026a), DataEvolve optimizes cleaning through quality proxies (Mi et al., 2026), and EvoDS accumulates skills (Yang et al., 2026b). Surveys cover workflow optimization (Yue et al., 2026), self-evolution, and memory (Gao et al., 2026; Huang et al., 2026a). Building on program revision, MATAR combines typed programs, paired downstream evaluation, and experience with frozen LLM parameters. Functional agent roles complement studies of communication (Yan et al., 2026), while structured feedback links revisions to outcomes (Feng et al., 2026).

3 PRELIMINARIES

Multimodal data. Let $X_{\text{raw}} = \{x_i\}_{i=1}^{N}$ contain N records with up to M modalities. For modality m, the availability indicator $m_i^{(m)} \in \{0, 1\}$ equals one when the observation is present, and $z_{0,i}^{(m)} \in \mathbb{R}^{d_m}$ denotes its base feature vector. The modality masks and base representations are

$$\mathbf{m}_i = (m_i^{(1)}, \ldots, m_i^{(M)}), \qquad Z_0 = (Z_0^{(m)})_{m=1}^{M}, \qquad Z_0^{(m)} = [z_{0,1}^{(m)}, \ldots, z_{0,N}^{(m)}]^\top. \tag{1}$$

Public unlabeled observations are available for refinement, while private supervision and evaluation partitions remain with the evaluator.

Multimodal data refinement. A program P maps the public inputs to features Z_P and proposed relations $E_P \subseteq \mathcal{V} \times \mathcal{V}$, where $\mathcal{V} = \{1, \ldots, N\}$ indexes the records. Attribute and relation programs satisfy

$$P_A(X_{\text{raw}}, Z_0) = (Z_{P_A}, \varnothing), P_R(X_{\text{raw}}, Z_0) = (Z_0, E_{P_R}), \mathrm{Id}(X_{\text{raw}}, Z_0) \quad = (Z_0, \varnothing). \tag{2}$$

Here Id denotes Preserve, and $\varnothing$ means that no additional relations are proposed. Each program combines operations within one path and preserves raw observations and record membership. The evaluator constructs task inputs $I_P = \mathcal{A}_{\mathcal{Q}}(Z_P, E_P, E_{\text{ref}})$, where E_{ref} is the shared reference graph and $\mathcal{A}_{\mathcal{Q}}$ specifies feature-width adaptation and edge integration. Selection baselines may restrict training supervision through the common evaluation interface.

Task adaptivity. We represent a task as $\mathcal{T} = (\ell_{\mathcal{T}}, \mathcal{Y}_{\mathcal{T}}, s_{\mathcal{T}})$, specifying its training objective, supervision, and primary metric, with larger $s_{\mathcal{T}}$ indicating better performance. Task-aware data selection uses task representations, downstream feedback, or both to guide sample use (Cheng et al., 2026; Shechter & Carmon, 2025). Here, task utility is measured after fitting model parameters $\vartheta_{P,\xi}$ under protocol $\mathcal{Q}$ and seed ξ,

$$\vartheta_{P,\xi} = \mathrm{Fit}_{\mathcal{T},\mathcal{Q}}(I_P, \xi), \qquad m_\xi(P) \quad = s_{\mathcal{T}}\big(f_{\vartheta_{P,\xi}}(I_P), \mathcal{Y}_{\mathcal{T}}^{\text{search}}\big). \tag{3}$$

The fitting procedure follows the declared training and checkpoint-selection rules, and the score uses the designated Search queries and supervision $\mathcal{Y}_{\mathcal{T}}^{\text{search}}$. Adaptivity thus concerns the construction of Z_P or E_P according to measured task utility.

Problem formulation. Let $\mathcal{Q}$ specify the model family, training rules, splits, seeds, resource limits, and acceptance policy. The admissible space is $\Omega(\mathcal{Q}) = \Omega_A(\mathcal{Q}) \cup \Omega_R(\mathcal{Q})$, with Id available in both paths and no cross-path composition. For the common Search seed set Ξ with $K = |\Xi|$, the mean score is $\widehat{V}_{\mathcal{T},\mathcal{Q}}^{\text{search}}(P) = \frac{1}{K}\sum_{\xi\in\Xi} m_\xi(P)$, and the objective is to maximize its gain over Preserve,

$$\underset{P\in\Omega(\mathcal{Q})}{\text{maximize}} \quad \widehat{V}_{\mathcal{T},\mathcal{Q}}^{\text{search}}(P) - \widehat{V}_{\mathcal{T},\mathcal{Q}}^{\text{search}}(\mathrm{Id}). \tag{4}$$

Search approximates this objective within a finite budget using the paired ranking rule in Sec. 4.3. Frozen candidates undergo the predeclared Commit policy and final Test evaluation.

4 METHOD

4.1 OVERVIEW

Given multimodal records and a downstream task, MATAR searches independent attribute and structure paths for refinement programs (Fig. 1). Task-adaptive tool optimization uses task requirements, public data profiles, and search history to revise a parent program. Paired replay evaluation compares the downstream performance of the revision and its parent under matched training conditions to rank candidates and assess the refinement hypothesis. Experiential policy adaptation records outcomes with their revision context and updates the controller, guiding the next parent, tool, and edit choices. Search selects one finalist across both paths, and Commit compares it with Preserve, returning refined attributes, a refined graph structure, or unchanged inputs.

In task-adaptive tool optimization, the Planner selects a parent and forms a refinement hypothesis, the Primitive Analyst identifies suitable tools or missing numerical capabilities, and the Rule Generator revises the program or generates an attribute operation. In paired replay evaluation, the Rule Scorer summarizes externally measured scores and paired differences for candidate comparison. In experiential policy adaptation, the Rule Evaluator interprets revisions and outcomes, recording experience for subsequent planning and generation. All share a frozen LLM. A numerical controller adapts proposal instructions. An external evaluator computes downstream scores.

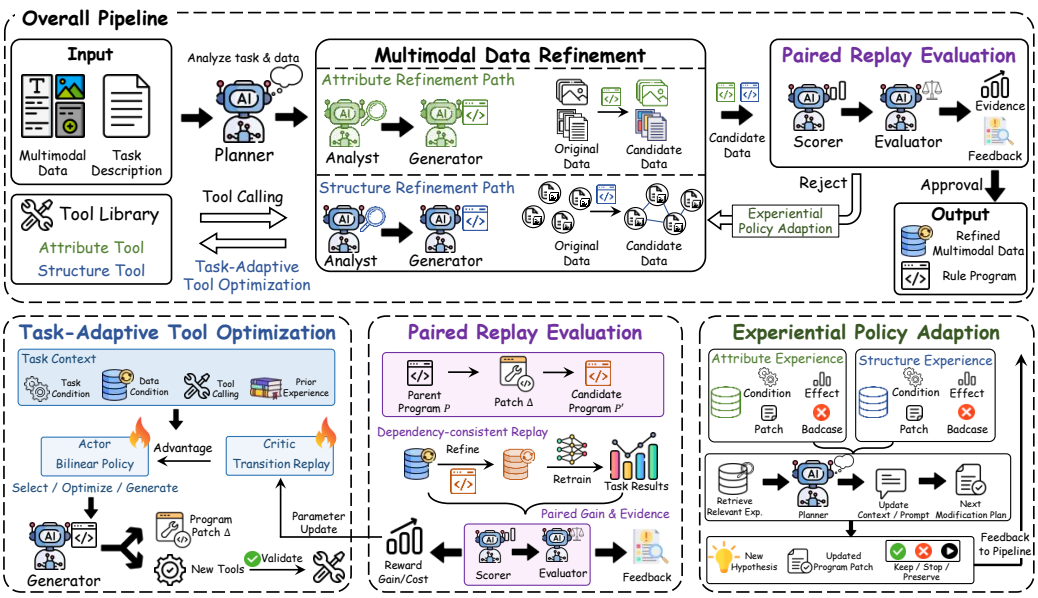


Figure 1: Overview of MATAR. Task analysis guides separate attribute and structure searches. Tool optimization constructs and revises refinement programs. Paired evaluation measures downstream utility, guiding subsequent proposals through experiential policy adaptation.

4.2 Task-Adaptive Tool Optimization

Motivation. Rigid refinement procedures restrict adaptation to the operations and execution order chosen in advance. Selecting tools or tuning parameters within such a procedure may be insufficient when a task requires a different feature transformation or a revised structure construction process. Treating the procedure as an editable program allows the search to change both the operations used and their composition within each refinement path. When available attribute tools cannot express a proposed transformation, generating a validated numerical operation further extends the search space. These considerations motivate joint optimization of program structure and parameters, with constrained tool generation to address missing capabilities.

MATAR searches bounded programs within an active path $b \in \{A, R\}$,

$$P_b = (a_1, \ldots, a_L), \qquad a_\ell = (o_\ell, \theta_\ell), \qquad 0 \le L \le L_{\max}, \tag{5}$$

where o_ℓ is a path-compatible operator and θ_ℓ contains its validated parameters. The path identifiers A and R denote attribute and structure refinement, respectively. The empty program represents Preserve. The formal configuration permits at most six operations, including multiple operations from the same path. Starting from $S_0 = (Z_0, \varnothing)$, execution produces

$$S_\ell = T_{o_\ell, \theta_\ell}(S_{\ell-1}; X_{\text{raw}}), \qquad S_L = (Z_{P_b}, E_{P_b}). \tag{6}$$

Original observations remain unchanged, and the evaluator applies the declared input adapter to the resulting features and edges. The formal operator catalog retains all training records.

Attribute operators append lexical or frozen-encoder views, rescale modalities, or apply generated numerical transformations. For example, a derived text view v_i is appended as

$$z_i^{T\prime} = [z_i^T \,;\, \gamma r_0 u(v_i)], \qquad u(v) = \frac{v}{\max(\|v\|_2, \epsilon)}, \qquad r_0 = \text{median}_i \, \|z_{0,i}^T\|_2. \tag{7}$$

The weight γ controls the derived view relative to the base feature scale. Structure operators instead construct cosine-neighbor graphs from text, image, or fused base features and revise the proposed edge set through filtering, union, or intersection. For a unit-normalized representation $\bar{z}_i$, a neighbor proposal takes the form

$$D = \{(i, j) : j \in \mathcal{N}_k(i), \ \langle \bar{z}_i, \bar{z}_j \rangle \ge \tau\}. \tag{8}$$

Union or mutual-neighbor rules convert D into undirected edges, subject to the declared edge budget.

When the existing attribute operators do not express the proposed change, the generator constructs a new operation in a restricted array-expression language. Expressions can use current features, base features, the other modality, and public feature statistics, with bounded depth and size. Registration checks shape preservation, finite outputs, permutation equivariance, and redundant behavior before admitting an operator. A generated target y_i is applied through a bounded residual update,

$$x_i' = x_i + \rho(y_i - x_i)\min\left(1, \frac{\|x_i\|_2}{\max(\|y_i - x_i\|_2, \epsilon)}\right), \qquad 0 < \rho \leq 0.5. \tag{9}$$

This bound limits the perturbation magnitude without assuming that the transformation improves task performance. Generated operators are disabled in structure sessions because their outputs are feature transformations. The Rule Generator revises a measured parent through insertion, removal, replacement, or movement of operations,

$$P_b' = \mathrm{ApplyPatch}(P_b, \delta), \qquad P_b' \in \Omega_b(\mathcal{Q}). \tag{10}$$

Validation enforces path isolation, operator dependencies, parameter bounds, and the program budget. Each revision must change the selected operator, its parameters, or its position. Invalid proposals receive bounded repair attempts, and unsuccessful slots remain part of the search budget.

4.3 Paired Replay Evaluation

Motivation. Generic quality scores do not directly establish whether a refinement benefits the target task. Even downstream score differences can reflect variation in data splits or model training rather than the effect of a program revision. Comparing a candidate with its parent under matched conditions makes the measured difference more informative about the revision, while comparison with Preserve checks its value relative to unchanged inputs. This motivates paired task evaluation as the source of feedback for candidate ranking and subsequent search decisions.

We use evaluation protocols designed for different tasks, with their metrics, input adapters, training rules, and acceptance criteria specified in the appendix. Within each protocol, a candidate and its parent are compared using the same data splits, model architecture, training budget, and Search seeds. Let $m_\xi(P)$ denote the Search score obtained after fitting the task model to the output of P with seed ξ. For K paired seeds, improvement over the parent and gain over Preserve are

$$\Delta(P', P) = \frac{1}{K}\sum_{\xi \in \Xi}[m_\xi(P') - m_\xi(P)], \qquad g(P) = \Delta(P, \mathrm{Id}). \tag{11}$$

The paired difference relates a revision to its measured downstream effect while controlling training randomness. It evaluates the complete revision and does not isolate individual operations when several changes are made together. Candidates are ranked within a path using

$$U(P) = g(P) - \lambda\,\mathrm{SE}_\xi[m_\xi(P) - m_\xi(\mathrm{Id})], \tag{12}$$

where the uncertainty penalty is a search heuristic rather than a confidence guarantee. Program hashes and output fingerprints identify duplicate proposals before another downstream evaluation. Execution records link each result to the program and data that produced it, and the selected output is checked for reproducibility. The Rule Scorer and Rule Evaluator interpret these measurements but cannot replace them with LLM-assigned scores.

After both searches finish, the evaluator selects one finalist using Search gain relative to Preserve. Commit then compares this frozen finalist with Preserve under the declared task protocol. An unsuccessful finalist falls back to Preserve, and Test evaluates only the selected output without feeding results back into the current search.

4.4 Experiential Policy Adaptation

Motivation. Reliable evaluation identifies useful candidates, but a task score alone does not explain the conditions under which a revision helps or which change to attempt next. Discarding the associated hypotheses and failure context would leave later proposals without information needed to distinguish promising directions from previously unsuccessful ones. Retaining this context allows measured outcomes to inform subsequent decisions about tool selection, program revision, and missing capabilities. We therefore combine recent experience with an adaptive proposal controller so that refinement decisions can change during search while the LLM parameters remain fixed.

After each round, MATAR associates candidate outcomes with their hypotheses, paired effects, failures, and feedback interpretations,

$$e_t = (h_t, \Delta_t, \text{status}_t, \text{error}_t, \text{interpretation}_t). \tag{13}$$

Recent records enter the context used to select parents, diagnose missing capabilities, and propose subsequent edits. Both successful and unsuccessful outcomes are retained, so later decisions can respond to observed failure conditions. The formal runs initialize this memory independently for each search session and update it within the run.

Adaptation also updates a lightweight controller that selects structured instructions for tool selection, optimization, or generation. The LLM produces the corresponding program edits while its parameters remain fixed,

$$u_t \sim \pi_{W_t}(\cdot \mid c_t), \qquad \delta_t \sim p_\psi(\cdot \mid P_t, u_t, \mathcal{L}_t, \mathcal{H}_t), \qquad \psi_{t+1} = \psi_t. \tag{14}$$

Here c_t summarizes the search state, $\mathcal{L}_t$ contains measured program records, and $\mathcal{H}_t$ contains recent experience. Controller rewards combine paired local progress and improvement of the best candidate, with penalties for failures and model calls. Fresh decisions update the controller, while replayed outcomes train its value estimator. This separates adaptation of the proposal policy from LLM training and from the evaluator that supplies authoritative task scores.

5 Experiments

Our experimental protocol addresses five questions. **Q1**. Does MATAR improve downstream performance over data selection and workflow baselines when attribute and structure refinement are evaluated separately? **Q2**. How much does each of the three core modules contribute to refinement performance? **Q3**. How sensitive is MATAR to its principal hyperparameters? **Q4**. Does refinement experience acquired on one dataset improve performance on another dataset for the same task? **Q5**. How robust is MATAR to modality and label noise?

5.1 Experimental Setup

Datasets. The main comparison uses eight datasets from OpenMAG (Wan et al., 2026), Grocery, Movies, Bili Dance, DY, Flickr30k, Bili Music, Bili Cartoon, and Bili Food, with two datasets for each downstream task. Attribute and structure refinement are evaluated independently.

Baselines. We compare MATAR with six data selection methods, FLYT (Shechter & Carmon, 2025), TADS (Cheng et al., 2026), EcoDatum (Xu et al., 2025), DataTailor (Yu et al., 2025), ARDS (Yang et al., 2025), and TAROT (Feng et al., 2025), and three workflow-based methods, DataFlow (Liang et al., 2025), COMET (Mohammed et al., 2025), and MM-Gen (Joshi et al., 2025). Preserve provides the no-refinement reference. All methods share evaluation records and the downstream protocol.

Tasks and Metrics. We evaluate entity classification (EC) using Macro-F1, link prediction (LP) using sampled pooled ROC-AUC, cross-modal retrieval (MR) using bidirectional MRR, and modality matching (MM) using mean paired cosine similarity.

Backbone. We use a MGAT model with task-specific prediction heads and training protocols for downstream evaluation. DeepSeek-V4-Flash serves as the LLM for the agent roles in MATAR.

5.2 Main Performance Comparison (Q1)

We compare MATAR with nine baselines on eight datasets, selecting one output from the separate refinement paths. Table 1 shows that MATAR achieves the highest mean on all eight datasets, averaging a 23.9% relative gain over the strongest baseline per column. The largest relative gains are 75.8% on MR Bili Music and 62.5% on MM Bili Cartoon. Relative gains are 9.2% on MR Flickr30k and 27.1% on MM Bili Food, compared with 5.9% to 7.6% on EC and 1.3% to 1.4% on LP.

The larger MM and MR gains may arise because these tasks depend on the relative positions of text and image representations, which can change substantially when refinement adjusts modality scales, adds feature views, or revises neighborhood context. Task-Adaptive Tool Optimization allows the search to explore these changes, while Paired Replay Evaluation assesses their effects on matching or ranking performance. This helps identify useful refinements even for original records already satisfying generic quality criteria. The smaller LP gains may reflect the strong signal the training graph already supplies, with attribute calibration helping the model better use available features. Gains across all four tasks support selecting refinement programs through downstream feedback, since an operation useful for modality tasks need not equally benefit classification or link prediction.

Table 1: Main Performance. The best results are in **bold**, and the second-best means are underlined.

Method \ Dataset	EC		LP		MR		MM	
	Grocery	Movies	Dance	DY	Flickr30k	Music	Cartoon	Food
FLYT	72.01 ±0.54	35.68 ±0.78	78.29 ±0.95	68.31 ±0.55	51.18 ±0.01	12.12 ±0.10	16.03 ±0.35	13.50 ±0.49
TADS	67.11 ±1.46	30.97 ±1.57	68.87 ±0.68	65.58 ±1.14	51.02 ±0.14	12.17 ±0.03	14.88 ±0.97	12.16 ±0.10
EcoDatum	43.30 ±0.54	34.92 ±0.87	77.50 ±2.28	67.37 ±2.56	50.95 ±0.07	12.25 ±0.04	18.42 ±0.54	12.41 ±3.33
DataTailor	60.98 ±0.58	23.33 ±0.83	59.57 ±1.37	61.15 ±0.99	51.29 ±0.06	12.48 ±0.06	6.46 ±1.32	7.05 ±0.44
ARDS	57.41 ±0.04	26.40 ±0.25	56.81 ±0.32	62.74 ±0.39	50.55 ±0.01	11.62 ±0.02	9.58 ±0.56	8.48 ±0.24
TAROT	49.57 ±0.61	22.88 ±0.23	63.38 ±1.64	69.70 ±1.19	50.48 ±0.03	11.86 ±0.09	21.62 ±2.05	15.70 ±2.98
DataFlow	73.37 ±0.15	38.93 ±0.14	78.02 ±0.54	68.15 ±0.20	47.74 ±0.07	11.45 ±0.08	21.28 ±0.04	13.50 ±0.02
COMET	73.63 ±0.36	37.80 ±1.37	81.74 ±2.89	68.56 ±1.39	51.17 ±0.04	11.60 ±0.26	25.07 ±8.39	13.41 ±5.83
MM-Gen	73.57 ±0.04	37.41 ±0.27	87.73 ±0.11	80.33 ±0.08	49.80 ±0.01	11.95 ±0.08	15.19 ±0.07	9.27 ±0.00
MATAR	**78.00 ±0.65**	**41.89 ±1.49**	**88.86 ±0.36**	**81.42 ±0.70**	**56.03 ±0.04**	**21.94 ±0.28**	**40.74 ±1.79**	**19.95 ±0.23**

Table 2: Ablation results on four task and dataset pairs. The best results are in **bold**.

Variant	Grocery(EC)	Bili Food(MM)	Bili Dance(LP)	Bili Music(MR)
MATAR (Full)	$\mathbf{78.00 \pm 0.65}$	$\mathbf{19.95 \pm 0.23}$	$\mathbf{88.86 \pm 0.36}$	$\mathbf{21.94 \pm 0.28}$
w/o TATO	77.37 ± 0.09	15.78 ± 0.05	85.08 ± 0.11	18.21 ± 0.10
w/o PRE	77.65 ± 0.14	17.53 ± 0.83	85.42 ± 0.07	18.32 ± 1.77
w/o EPA	77.90 ± 0.40	17.95 ± 1.77	85.25 ± 0.06	17.51 ± 2.50

5.3 Ablation Study (Q2)

We compare the full framework with w/o TATO, w/o PRE, and w/o EPA on four scenarios. Table 2 shows that the full framework achieves the highest mean on all four pairs. Removing TATO, PRE, and EPA produces mean relative decreases of 10.74%, 8.24%, and 8.61%, respectively. TATO removal has the largest effect on EC, MM, and LP, whereas EPA removal causes the largest MR decrease.

The large decrease in MM performance without TATO suggests that selecting a fixed operation is insufficient when useful refinement requires adjusting feature scales or combining neighborhood construction with edge filtering. TATO can revise operator parameters and composition, giving the search more ways to adapt the inputs to the matching objective. The decrease without PRE may result from less reliable candidate comparisons, as using independent training seeds makes it harder to distinguish the effect of a program revision from variation in model fitting. Pairing each candidate with its parent under the same seeds makes the feedback more informative for both candidate selection and subsequent search decisions. The particularly large decrease in MR performance without EPA suggests that retrieval refinement benefits from remembering which revisions improved ranking and which failed under similar conditions. Retaining these outcomes alongside the corresponding

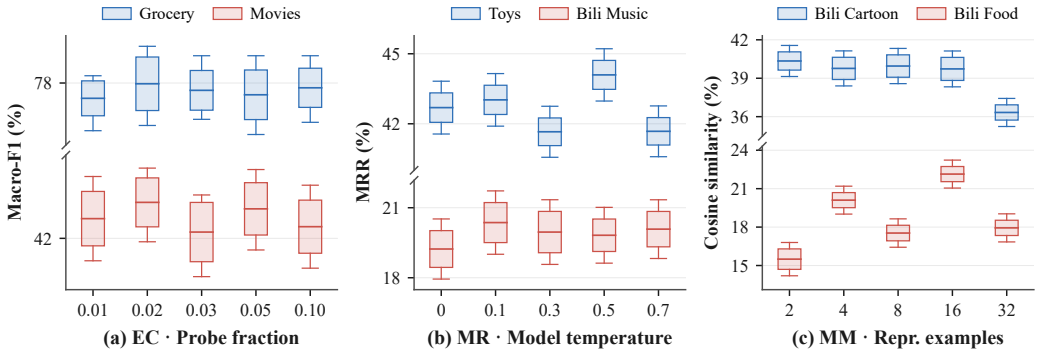


Figure 2: Hyperparameter sensitivity on EC, MR, and MM. Center lines show measured means, while rectangles and whiskers are constructed display summaries.

Table 3: Cross-dataset experience transfer through frozen source refinement programs.

Task	Source → Target	Refinement Path	Target Preserve	Transferred Score
EC	Grocery → Movies	Structure Path	39.02 ± 0.08	41.82 ± 0.59
EC	Movies → Grocery	Structure Path	74.08 ± 0.11	77.00 ± 0.26
LP	Bili Dance → DY	Attribute Path	69.59 ± 0.29	82.20 ± 0.61
LP	DY → Bili Dance	Attribute Path	83.65 ± 0.12	84.66 ± 0.29
MR	Flickr30k → Bili Music	Structure Path	12.10 ± 0.09	12.34 ± 0.07
MR	Bili Music → Flickr30k	Attribute Path	51.17 ± 0.05	52.53 ± 0.42
MM	Bili Cartoon → Bili Food	Structure Path	15.50 ± 0.21	27.27 ± 1.67
MM	Bili Food → Bili Cartoon	Structure Path	18.87 ± 0.15	23.60 ± 0.57

hypotheses allows the Planner to build on useful edits and avoid repeatedly spending the limited proposal budget on unproductive changes.

5.4 Hyperparameter Sensitivity (Q3)

We vary EC probe fraction, MR proposal temperature, and MM context size with other controls fixed. Figure 2 shows limited EC sensitivity to probe fraction, while MR favors different temperatures across datasets. MM is more sensitive to context size, with Bili Food increasing from 15.50% with 2 examples to 22.14% with 16, but neither dataset benefiting from a further increase to 32.

The limited EC sensitivity to the probe fraction may reflect the separation between initial diagnosis and candidate evaluation. The probe supplies only feature statistics to the Planner, while Paired Replay Evaluation still assesses every candidate on the complete Search split. This evaluation feedback can therefore guide later revisions even with a small initial diagnostic sample. The different preferred MR temperatures may reflect a balance between exploring new program edits and refining promising parent programs, since more varied proposals must compete for the same evaluation budget. Recent experience guides proposal generation, but how much variation is useful still depends on the dataset. The MM improvement with more context on Bili Food suggests additional records help the Planner identify recurring patterns relevant to feature or neighborhood refinement. The decline at 32 examples may result from extra records adding redundant or conflicting cues to the planning context, changing the proposed edits without correspondingly increasing useful task information.

5.5 Cross-Dataset Experience Transfer (Q4)

We transfer experience encoded in frozen source programs and retrain target models without further program search. Table 3 shows higher mean target scores in all eight transfer directions. Gains remain directional, with LP on DY increasing from 69.59% to 82.20%, compared with 83.65% to 84.66% on Bili Dance. MR also benefits in both directions, reaching 12.34% on Bili Music and 52.53% on Flickr30k, compared with Preserve at 12.10% and 51.17%, respectively.

Transfer gains may arise because frozen programs reconstruct neighborhoods or transform representations from target features without further target-side program search. For EC, reconstructed neighborhoods can contextualize ambiguous records, supporting transfer in both directions. The

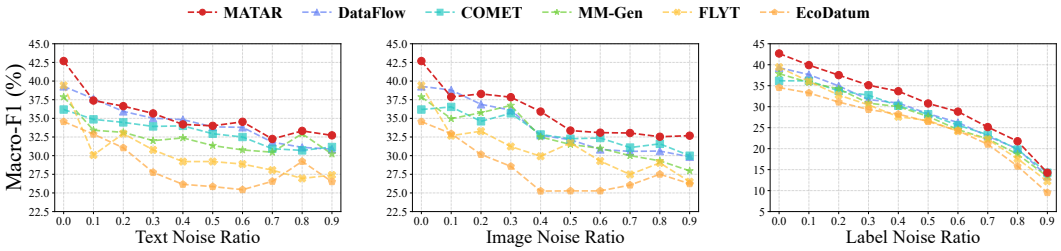


Figure 3: EC robustness on Movies under text, image, and training-label corruption.

larger LP gain on DY may reflect attribute transformations better suited to target modality-scale or feature-variance imbalances, improving feature use alongside the shared graph. MR gains through both paths suggest that revised neighborhoods or attributes help the model learn cross-modal rankings. For MM, aggregating related records before modality projection training provides shared semantic context that may bring paired text and image outputs closer.

These results suggest different data refinement priorities across downstream tasks. ❶ For EC, data quality should be assessed for a record and its neighborhood, since contextual evidence can make an otherwise ambiguous record useful for classification. This favors transferring rules for constructing informative neighborhoods across datasets. ❷ For LP, predicting structure can benefit from refining attributes even with a fixed input graph. The directional gains make compatibility between feature calibration and the target distribution an important consideration when transferring a program. ❸ For MR, a transferable refinement should preserve useful ranking relationships, whether operating on attributes or structure. This favors choosing the refinement path by its effect on target rankings rather than assigning a fixed path to retrieval tasks. ❹ For MM, data preparation can support alignment by organizing shared semantic context across records. Neighborhood construction is therefore a useful part of matching-oriented refinement alongside transformations of individual modality features.

5.6 Robustness to Modality and Label Noise (Q5)

We compare MATAR with five baselines on EC Movies under text, image, and label corruption at $r = 0, 0.1, \ldots, 0.9$. Figure 3 shows that MATAR leads at most noise levels, retaining 1.56- and 2.68-% advantages over COMET under text and image noise at $r = 0.9$. Severe label noise removes this advantage, with Macro-F1 at 14.26%, comparable to COMET at 14.27%.

The sustained advantage under text and image corruption may come from the ability to rescale modalities or construct neighborhoods from the remaining informative features. Paired replay evaluates these revisions on the corrupted inputs, and experience feedback allows later proposals to build on changes that recover classification performance under the same noise conditions. The similar scores under severe label noise may reflect a different source of error: incorrect training targets affect the supervision of the classifier even when the refined features remain informative. This explains why adapting attributes or relations can help more when corruption affects an input modality than when it affects the labels used for training.

6 Conclusion

We presented MATAR, a task-adaptive multi-agent framework that treats executable refinement procedures as the object of optimization. By searching attribute and relation programs separately, the framework makes feature views and inter-record structure editable according to downstream requirements. Task-adaptive tool optimization, paired replay evaluation, and experiential policy adaptation connect program revision with measured task utility while keeping LLM parameters fixed. Experiments across eight datasets and four tasks show an average relative improvement of 23.9% over the strongest baseline per dataset. The transfer results further indicate that source refinement programs can encode reusable experience, although their benefits vary across tasks and transfer directions. Together, these findings support evaluating attribute calibration and relational interventions according to the information consumed by each downstream task. The observed sensitivity to context choices and severe label noise also highlights the limits of refinement under constrained search and unreliable supervision. Future work can investigate experience-guided adaptation of transferred programs and evaluate these findings under broader datasets and independently fixed protocols.

AI USE STATEMENT

No large language models (LLMs) or LLM-based tools were used for manuscript drafting, revision, or language editing. However, LLMs were used as functional components of the proposed methodological framework. The authors take full responsibility for all content, references, and claims presented in this manuscript.

REFERENCES

Daoyuan Chen, Haibin Wang, Yilun Huang, Ce Ge, Yaliang Li, Bolin Ding, and Jingren Zhou. Data-Juicer Sandbox: A feedback-driven suite for multimodal data-model co-development. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 9198–9230, 2025.

Guanjie Cheng, Boyi Li, Lingyu Sun, Mengying Zhu, Yangyang Wu, Xinkui Zhao, and Shuiguang Deng. TADS: Task-aware data selection for multi-task multimodal pre-training. *arXiv preprint arXiv:2602.05251*, 2026.

Lan Feng, Fan Nie, Yuejiang Liu, and Alexandre Alahi. TAROT: Targeted data selection via optimal transport. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 16837–16852, 2025.

Xinshun Feng, Xinhao Song, Lijun Li, Gongshen Liu, and Jing Shao. SEARL: Joint optimization of policy and tool graph memory for self-evolving agents. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 24518–24535, 2026.

Huan-ang Gao, Jiayi Geng, Wenyue Hua, Mengkang Hu, Xinzhe Juan, Hongzhang Liu, Shilong Liu, Jiahao Qiu, Xuan Qi, Qihan Ren, Yiran Wu, Hongru Wang, Han Xiao, Yuhang Zhou, Shaokun Zhang, Jiayi Zhang, Jinyu Xiang, Yixiong Fang, Qiwen Zhao, Dongrui Liu, Cheng Qian, Zhenhailong Wang, Minda Hu, Huazheng Wang, Qingyun Wu, Heng Ji, and Mengdi Wang. A survey of self-evolving agents: What, when, how, and where to evolve on the path to artificial super intelligence. *Transactions on Machine Learning Research*, 2026.

Wei-Chieh Huang, Weizhi Zhang, Yueqing Liang, Yuanchen Bei, Yankai Chen, Tao Feng, Xinyu Pan, Zhen Tan, Yu Wang, Tianxin Wei, Shanglin Wu, Ruiyao Xu, Liangwei Yang, Rui Yang, Wooseong Yang, Chin-Yuan Yeh, Hanrong Zhang, Haozhen Zhang, Siqi Zhu, Henry Peng Zou, Wanjia Zhao, Song Wang, Wujiang Xu, Zixuan Ke, Zheng Hui, Dawei Li, Yaozu Wu, Langzhou He, Chen Wang, Xiongxiao Xu, Baixiang Huang, Juntao Tan, Shelby Heinecke, Huan Wang, Caiming Xiong, Ahmed Metwally, Jun Yan, Chen-Yu Lee, Hanqing Zeng, Yinglong Xia, Xiaokai Wei, Ali Payani, Yu Wang, Haitong Ma, Wenya Wang, Chenguang Wang, Yu Zhang, Xin Eric Wang, Yongfeng Zhang, Jiaxuan You, Hanghang Tong, Xiao Luo, Xue Liu, Yizhou Sun, Wei Wang, Julian McAuley, James Zou, Jiawei Han, Philip S. Yu, and Kai Shu. A survey of agent memory in the second half: Towards self-evolving and long-horizon agents. *Transactions on Machine Learning Research*, 2026a.

Zhen Huang, Yikun Wang, Shijie Xia, and Pengfei Liu. DataOrchestra: Learning to orchestrate per-example curation of pretraining data. *arXiv preprint arXiv:2607.24717*, 2026b.

Siddharth Joshi, Besmira Nushi, Vidhisha Balachandran, Varun Chandrasekaran, Vibhav Vineet, Neel Joshi, and Baharan Mirzasoleiman. MM-GEN: Principled and generalizable data curation for enhancing task performance in VLMs. *Journal of Data-centric Machine Learning Research*, 2 (26):1–28, 2025.

Feiyang Kang, Hanze Li, Adam Nguyen, Mahavir Dabas, Jiaqi W. Ma, Frederic Sala, Dawn Song, and Ruoxi Jia. Can generalist agents automate data curation? *arXiv preprint arXiv:2606.04261*, 2026. EMNLP 2026 Main Conference, as reported in arXiv v2.

Hao Liang, Xiaochen Ma, Zhou Liu, Zhen Hao Wong, Zhengyang Zhao, Zimo Meng, Runming He, Chengyu Shen, Qifeng Cai, Zhaoyang Han, Meiyi Qiang, Yalin Feng, Tianyi Bai, Zewei Pan, Ziyi Guo, Yizhen Jiang, Jingwen Deng, Qijie You, Peichao Lai, Tianyu Guo, Chi Hsu Tsai,

Hengyi Feng, Rui Hu, Wenkai Yu, Junbo Niu, Bohan Zeng, Ruichuan An, Lu Ma, Jihao Huang, Yaowei Zheng, Conghui He, Linpeng Tang, Bin Cui, Weinan E, and Wentao Zhang. DataFlow: An LLM-driven framework for unified data preparation and workflow automation in the era of data-centric AI. *arXiv preprint arXiv:2512.16676*, 2025.

Tiantian Mi, Dongming Shan, Zhen Huang, Yiwei Qin, Muhang Xie, Yuxuan Qiao, Yixiu Liu, Chenyang Zhou, and Pengfei Liu. Data darwinism part II: DataEvolve – AI can autonomously evolve pretraining data curation. *arXiv preprint arXiv:2603.14420*, 2026.

Sedir Mohammed, Felix Naumann, and Hazar Harmouch. Step-by-step data cleaning recommendations to improve ML prediction accuracy. In *Proceedings of the 28th International Conference on Extending Database Technology*, pp. 542–554, 2025. doi: 10.48786/edbt.2025.43.

Christopher Mohri, John Duchi, and Tatsunori Hashimoto. A bitter lesson for data filtering. *arXiv preprint arXiv:2605.19407*, 2026.

Pratyush Muthukumar, Harshil Kotamreddy, Sarah Amiraslani, Tomo Kanazawa, Ramani Akkati, Shaan Jain, and Andrew Mathau. Multimodal data curation through ranked retrieval. *arXiv preprint arXiv:2605.01163*, 2026.

Mikey Shechter and Yair Carmon. Filter like you test: Data-driven data filtering for CLIP pretraining. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

Chenxi Wan, Xunkai Li, Yilong Zuo, Haokun Deng, Sihan Li, Bowen Fan, Hongchao Qin, Ronghua Li, and Guoren Wang. OpenMAG: A comprehensive benchmark for multimodal-attributed graph. *arXiv preprint arXiv:2602.05576*, 2026.

Dingzirui Wang, Xuanliang Zhang, Keyan Xu, Qingfu Zhu, and Wanxiang Che. CurateEvo: Data-curation evolving for agentic post-training. *arXiv preprint arXiv:2607.06140*, 2026a.

Xu Wang, Xunkai Li, Yinlin Zhu, Rong-Hua Li, and Guoren Wang. GOMA: Toward structure-driven multimodal alignment from a graph signal smoothing perspective. *arXiv preprint arXiv:2605.15723*, 2026b.

Yibo Wang, Riteng Zhang, Yinghao He, Yongye Su, Bharat Bhargava, and Chunwei Liu. TabClean: Reusable LLM-synthesized programs for tabular data cleaning. *arXiv preprint arXiv:2606.25388*, 2026c.

Jinda Xu, Yuhao Song, Daming Wang, Weiwei Zhao, Minghua Chen, Kangliang Chen, and Qinya Li. Quality over quantity: Boosting data efficiency through ensembled multimodal data curation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 21761–21769, 2025. doi: 10.1609/aaai.v39i20.35481.

Rui Xue and Tianfu Wu. ReCoG: Reciprocal co-evolution for multimodal graph learning. *arXiv preprint arXiv:2608.22786*, 2026.

Bingyu Yan, Zhibo Zhou, Litian Zhang, Lian Zhang, Ziyi Zhou, Dezhuang Miao, Zhoujun Li, Chaozhuo Li, and Xiaoming Zhang. Beyond self-talk: A communication-centric survey of LLM-based multi-agent systems. *Frontiers of Computer Science*, 2026. doi: 10.1007/s11704-026-50857-y.

Bo Yang, Yunkui Chen, Lanfei Feng, Yu Zhang, and Shijian Li. ZPD detector: Data selection via capability-difficulty alignment for large language models. *arXiv preprint arXiv:2601.10986*, 2026a.

Xu Yang, Chen Liu, and Ying Wei. Data selection matters: Towards robust instruction tuning of large multimodal models. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-0316.

Zherui Yang, Fan Liu, Yansong Ning, and Hao Liu. EvoDS: Self-evolving autonomous data science agent with skill learning and context management. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2*, 2026b. doi: 10.1145/3770855.3818002.

Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *Transactions of the Association for Computational Linguistics*, 2:67–78, 2014.

Qifan Yu, Zhebei Shen, Zhongqi Yue, Yang Wu, Bosheng Qin, Wenqiao Zhang, Yunfei Li, Juncheng Li, Siliang Tang, and Yueting Zhuang. Mastering collaborative multi-modal data selection: A focus on informativeness, uniqueness, and representativeness. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 155–165, 2025.

Ling Yue, Kushal Raj Bhandari, Ching-Yun Ko, Dhaval Patel, Shuxin Lin, Nianjun Zhou, Jianxi Gao, Pin-Yu Chen, and Shaowu Pan. From static templates to dynamic runtime graphs: A survey of workflow optimization for LLM agents. *arXiv preprint arXiv:2603.22386*, 2026.

Kaituo Zhang, Mingzhi Hu, Hoang Anh Duy Le, Fariha Kabir Torsha, Zhimeng Jiang, Minh Khai Bui, Chia-Yuan Chang, Yu-Neng Chuang, Zhen Xiong, Ying Lin, Guanchu Wang, and Na Zou. A survey on evaluating quality and trustworthiness in LLM-generated data. *Transactions on Machine Learning Research*, 2026.

A DATASETS

The experimental protocol uses eight datasets from OpenMAG (Wan et al., 2026), with two datasets assigned to each of four downstream tasks. MATAR searches attribute and relation programs independently on the same task inputs. The attribute path keeps the reference relation input fixed, whereas the relation path keeps base features and training-record eligibility fixed. Search selects one finalist across these paths and Preserve, so the reported result is one final output rather than a composition or an average of the two paths. Baselines use their own adapted procedures without this path-selection stage.

A.1 DATASET DESCRIPTIONS

The eight datasets span e-commerce products, image–caption pairs, and short-video content. Each dataset pairs text and image content with pretrained features; the availability of category labels and a relation graph varies by dataset and determines which tasks it supports. Where a dataset has no verified classification labels it is not used for EC, and no labels are inferred from entity names or generated by language models. We describe each dataset below.

Grocery contains 17,074 entities across 20 categories of grocery and household products. Each entity pairs product descriptions and brand text with image features under CLIP/Qwen views, and co-purchase relations form the product graph. With broad category coverage and both modalities present, Grocery serves as the primary entity-classification domain. It is disclosed as a development domain: earlier development evaluations informed protocol choices, so Grocery results are reported as development evidence rather than as independent confirmation.

Movies contains 16,672 entities across 20 categories of film and home-media products. Each entity pairs textual descriptions and tags with CLIP image features, while the original product graph is unavailable. Movies starts without an observed graph, although the relation path can propose feature-based edges for downstream classification.

Bili Dance contains 2,307 entities of dance-video content. Records pair video-derived text and image features, and the dataset provides a relation graph but no verified category labels, so it is used for link prediction, where its small scale represents the short-video domain.

DY contains 8,299 entities of short-video content with a relation graph and bimodal features and no verified category labels. It is the larger of the two link-prediction datasets and extends the evaluation to a larger video graph.

Flickr30k comprises 31,783 images with five captions per image in the original release (Young et al., 2014). Our experiments use the OpenMAG-provided version with pretrained text and image features. It serves as an image–text retrieval dataset.

Bili Music contains 6,038 entities of music-video content with paired text and image features and no verified category labels. It is used for cross-modal retrieval.

Bili Cartoon contains 4,724 entities of animation content with bimodal features and no verified category labels. Modality matching requires only reliable paired records, so the dataset is usable without classification labels.

Bili Food contains 1,579 entities of food-video content with bimodal features and no verified category labels. It provides the smaller-scale matching probe, complementing Bili Cartoon.

A.2 DATA ACCESS AND SPLITS

The protocol is transductive. Public unlabeled records may support feature statistics and graph construction, while private targets remain inaccessible to refinement programs. A verified official Test split is retained where available. Otherwise, non-LP tasks use Train/Search/Commit/Test proportions of 50/20/15/15 with split seed 42. EC splits are stratified, and repeated images or entities are grouped for MR and MM. Non-LP tasks start without original input relations.

B BASELINES

The comparison includes six data selection methods and three workflow-based methods, together with Preserve as the no-refinement reference. The descriptions below follow the method summaries in Sec. 2. The experimental adaptations use a shared task interface rather than reproducing every original pretraining or data-cleaning setting.

B.1 DATA SELECTION METHODS

The six selection methods make different inductive bets about what makes a sample useful, and each is adapted by mapping its selected subset or weights onto Train supervision units.

FLYT (Shechter & Carmon, 2025) turns discrete data filtering into differentiable sample weighting so that the scoring model can be trained from downstream loss. A reference model is updated on weighted pretraining data, and the gradient through that update trains a scoring model over candidate features; the learned scores are then used to weight or filter the full candidate pool. Two instantiations cover different input views: M-FLYT combines standardized scores from pretrained CLIP and data-quality scorers through a learned linear combination, while E-FLYT learns scores directly from multimodal embeddings. Soft Cap Sampling finally organizes how many training exposures each score draws, trading unique-sample coverage against repeated exposure of high-scoring data. In our comparison FLYT represents learning-to-weight selection under downstream feedback. Its optimization object remains sample weighting and inclusion: the learned scores change which data are emphasized, but they never edit record attributes or inter-record relations, and the procedure that consumes them is not itself revised.

TADS (Cheng et al., 2026) decomposes data selection into three utility dimensions and learns a selection policy over them. A quality predictor estimates intrinsic sample quality from unimodal and cross-modal operators, task prototypes provide reference support sets for relevance estimation, and cluster-based weighting characterizes distributional diversity. A Data Value Network combines these quality, task-relevance, and diversity signals into a sample value. A proxy model trained on sampled subsets supplies multi-task evaluation feedback, and cluster-level perturbations translate this feedback into approximate update signals for the selection network without end-to-end differentiability. In our adaptation, TADS uses a ten-round proxy-feedback budget, with the feedback metric set to the task's primary metric. Its clusters are consumed internally for deduplication, density features, and feedback estimation, but they are never exposed as editable downstream structure.

EcoDatum (Xu et al., 2025) ensembles unimodal and cross-modal quality operators into a curated selection score. Weak supervision combines the operator signals, and quality-guided deduplication removes near-duplicate records before training. The ensemble targets data efficiency: with a fixed training budget, the curated subset should carry more information per sample than the full pool. In our comparison EcoDatum represents quality-based selection. Its ensembled operators measure completeness and agreement rather than task utility, so the selection can retain high-quality samples that are nevertheless redundant for the target task.

DataTailor (Yu et al., 2025) selects multimodal instruction data by balancing three collaborative criteria, informativeness, uniqueness, and representativeness, with informativeness estimated from sample representations, uniqueness from intra-cluster structure, and representativeness from inter-cluster structure. The clustering stage partitions records into coherent groups, and the three criteria jointly decide which records best carry each group's signal. In our comparison DataTailor represents criteria-driven selection over the training records available to the target task. The scoring principles are specified in advance, while their combination and selection proportions adapt to the data without iterative updates from measured downstream performance.

ARDS (Yang et al., 2025) selects visual instruction data to improve robustness. It identifies vulnerable evaluation subgroups using visual and textual perturbations, then prioritizes examples with similar multimodal representations. Our adaptation maps the selected entities onto Train supervision units under the common evaluation protocol, without adding generated relations or refined attributes. The adapter uses public modality embeddings in place of token-level conversation states and replaces the original perturbation proxy with embedding noise, masking, and cross-modal consistency. It retains the released subgroup construction and influence-based ranking procedures, so the comparison concerns this declared adaptation rather than the original instruction-tuning setup.

TAROT (Feng et al., 2025) selects data by matching the candidate distribution to a target reference through optimal transport in a whitened feature space. Whitening removes scale and correlation differences between feature channels before transport costs are computed, and the transport plan determines which candidates to retain. In our comparison TAROT represents distribution-matching selection. Distribution matching seeks to match the reference distribution, but it does not construct derived attributes or revise relations according to task requirements.

B.2 Workflow-Based Methods

The three workflow-based methods treat data preparation as a pipeline of cleaning, repair, or generation stages; they train on the full Train set and influence training through their governed data products.

DataFlow (Liang et al., 2025) maintains operator-based data workflows and extends them with an agentic loop, DataFlow-Agent, that retrieves existing operators from a library, generates missing ones when the library is insufficient, and repairs execution failures encountered by the pipeline, making the workflow itself the object of construction and revision. In our comparison DataFlow represents workflow construction under execution-level feedback. Its operator-construction loop uses execution and verification feedback, while the original paper separately evaluates downstream model performance. In our comparison, candidates are scored under the shared downstream protocol.

COMET (Mohammed et al., 2025) estimates the predictive improvement of candidate cleaning actions without executing them end-to-end: it simulates additional corruption of each feature, measures how much predictive performance degrades, and ranks cleaning actions by the estimated gains they would recover. The method prioritizes tabular cleaning and leaves the actual repairs to separate algorithms or human analysts, so its use on multimodal graphs requires a task-specific adaptation rather than treating the original method as a refiner.

MM-Gen (Joshi et al., 2025) guides multimodal annotation generation with target-task reference examples and subgroup information: the generation stage conditions on the reference context and subgroup structure, and a perplexity-based filtering stage balances annotation quality and informational value. It represents task-directed generation rather than sample selection alone, and in our comparison the generated annotations are consumed as governed features under the common downstream protocol. The generation and filtering stages follow a fixed pipeline: the procedure is not revised by measured downstream outcomes.

B.3 Common Comparison Protocol

Selection methods restrict Train supervision, whereas workflow methods retain the full Train set and modify its data products. Evaluation records and candidate sets remain identical across methods. For each task and dataset, methods share input permissions, downstream configurations, maximum optimizer-step budgets, and training seeds. Preserve retains the original inputs and serves as the internal acceptance reference.

The independent-path restriction applies to the two searches of MATAR. Attribute refinement cannot modify reference relations, and structure refinement cannot modify base attributes, representations, or training-record eligibility. Baselines retain their declared task-specific adaptations and do not perform A/R finalist selection. Complete method-specific adapter specifications and tuning grids remain to be finalized for reproducibility.

C Tasks and Metrics

All four tasks fit a downstream model to the candidate inputs under a task-specific protocol. EC and LP score category and edge predictions, while MR and MM score learned modality representations through ranking and paired cosine similarity, respectively. For MATAR, each task admits separate attribute and relation searches, followed by selection of one output. The main table reports that selected output, including Preserve fallback, and does not identify separate causal contributions from the two paths.

C.1 Task Descriptions

Entity classification (EC) is the canonical category prediction task on Grocery and Movies. The downstream model predicts each held-out category from the candidate features and graph input. Training uses Train labels, while refinement programs receive only controlled Search feedback and cannot read private labels. The primary metric is Macro-F1, which averages per-class F1 scores so that skewed category distributions do not dominate the aggregate. EC measures whether the selected features or proposed relations improve category prediction under the same training protocol.

Link prediction (LP) scores candidate edges on Bili Dance and DY. The primary metric is sampled pooled ROC-AUC, which measures how well the model distinguishes positive links from sampled negative links.

Cross-modal retrieval (MR) retrieves relevant images from text queries and relevant text from image queries on Flickr30k and Bili Music. The primary metric is bidirectional MRR, which measures how highly the learned representations rank relevant targets.

Modality matching (MM) measures agreement between paired modality representations on Bili Cartoon and Bili Food. The primary metric is the mean cosine similarity of paired text and image outputs from the trained model. A cyclic shift of image outputs supplies nonpaired similarity and pair-margin diagnostics, which are not components of the primary metric. The metric has theoretical range $[-100, 100]$ after display scaling and requires no classification labels, only reliable paired records. MM uses a trained graph encoder and modality projections with a symmetric contrastive objective on Train pairs. Both candidate features and proposed relations can affect the learned outputs, so an improvement cannot be attributed solely to direct feature alignment.

C.2 Metric Definitions

Macro-F1 is the unweighted mean of class-specific F1 scores, giving each class equal weight regardless of its frequency. With class-specific precision p_c and recall r_c over the evaluation label set $\mathcal{C}$, the metric is

$$\mathrm{MacroF1} = \frac{1}{|\mathcal{C}|} \sum_{c \in \mathcal{C}} \frac{2 p_c r_c}{p_c + r_c}. \tag{15}$$

It is the primary metric for category prediction and stresses class-boundary recovery under possibly skewed category priors. The evaluator includes every class in the declared label space and assigns zero F1 when a class has no true positives, false positives, or false negatives.

Sampled pooled ROC-AUC measures the probability that a randomly drawn positive link scores above a randomly drawn sampled negative link, pooling all scores across evaluation queries within each training seed and counting ties as half a hit. Let $\mathcal{S}^+$ and $\mathcal{S}^-$ be the corresponding score multisets. The empirical metric is

$$\mathrm{AUC} = \frac{1}{|\mathcal{S}^+||\mathcal{S}^-|} \sum_{s^+ \in \mathcal{S}^+} \sum_{s^- \in \mathcal{S}^-} \left[\mathbf{1}\{s^+ > s^-\} + \frac{1}{2} \mathbf{1}\{s^+ = s^-\} \right]. \tag{16}$$

It stresses edge-ranking quality under the shared reference graph.

Bidirectional MRR measures the average reciprocal rank of the first relevant target across queries and retrieval directions, so a relevant target at rank one contributes a full point while later ranks contribute progressively less. For direction $d \in \{T \to I, I \to T\}$, let $\mathcal{Q}_d$ be the query set and $r_{i,d}$ the rank of the first relevant result. The metric is

$$\mathrm{MRR}_{\mathrm{bi}} = \frac{1}{2} \sum_{d \in \{T \to I, I \to T\}} \frac{1}{|\mathcal{Q}_d|} \sum_{i \in \mathcal{Q}_d} \frac{1}{r_{i,d}}. \tag{17}$$

It stresses whether the refined representations place the paired target near the top of the complete gallery; the gallery and relevant-target assignments are fixed across methods.

Mean paired cosine similarity evaluates the geometric agreement of paired outputs from the trained modality projections. For the evaluation pair set $\mathcal{J}$ and nonzero projected representations $\widehat{z}_i^T$ and $\widehat{z}_i^I$, it is

$$\mathrm{Cos}_{\mathrm{pair}} = \frac{1}{|\mathcal{J}|} \sum_{i \in \mathcal{J}} \frac{\langle \widehat{z}_i^T, \widehat{z}_i^I \rangle}{\|\widehat{z}_i^T\|_2 \|\widehat{z}_i^I\|_2}. \tag{18}$$

This is an alignment score rather than classification accuracy or retrieval success. Zero-norm or nonfinite projected outputs invalidate an evaluation rather than reducing its coverage. The displayed metrics equal 100 times their mathematical definitions, with MM already scaled by 100 inside the evaluator. Paired comparisons in Eq. 11 use the native metric units consistently within a task.

C.3 Downstream Training

The evaluators use the frozen feature bundles and a shared three-layer MGAT model family. A deterministic signed projection maps augmented features back to the base input width before model fitting, preserving the downstream parameter dimensions across candidates. EC allows up to 150 epochs.

MM trains for at most 60 epochs using symmetric in-batch contrastive loss with temperature 0.07, batch size 256, and Search paired cosine for checkpoint selection. It uses patience eight, so its training-seed repetitions measure model-fitting variability.

D Refinement Setting and Evaluation Details

Data views. The convention $m_i^{(j)} = 1$ indicates an observed modality. The formal relation interface uses unweighted undirected entity pairs. Let X_0 denote the unrefined attribute view and $Z_0 = \Phi(X_0) \in \mathbb{R}^{N \times d}$ its base representation, where Φ is a fixed encoding pipeline and d is the feature dimension.

Executable refinement programs. We implement a refinement procedure P as an executable program built from a registered primitive library $\mathcal{P}_{\text{lib}}$ with typed input and output contracts. An operator has a typed parameter schema and validated numerical behavior, and a program combines operators through a bounded sequence of edits. A program organizes one or more tools within a single refinement path. Code-as-Rule denotes this constrained executable representation. The admissible set $\Omega_b(\mathcal{Q})$ enforces type, access, provenance, integrity, path-isolation, and resource constraints. Registered implementations and model versions remain fixed when a candidate is evaluated, while new attribute operators may be registered during search. Resource limits constrain admissibility, and controller rewards also penalize failed proposals and model calls. Program optimization does not require differentiable execution or updates to LLM parameters.

Independent refinement paths. Attribute refinement produces feature views Z_P while preserving a reference relation input E_{ref}. Relation refinement produces proposed edges E_P from the original records and base representations while preserving Z_0. Let $\mathcal{A}_{\mathcal{Q}}$ denote the shared input adapter and $\mathcal{G}_{\mathcal{Q}}$ the declared graph-combination rule. The resulting task inputs are

$$\begin{aligned} \mathcal{I}_A(P) &= \big(\mathcal{A}_{\mathcal{Q}}(Z_P), E_{\text{ref}}\big), \\ \mathcal{I}_R(P) &= \big(\mathcal{A}_{\mathcal{Q}}(Z_0), \mathcal{G}_{\mathcal{Q}}(E_{\text{ref}}, E_P)\big). \end{aligned} \tag{19}$$

The reference input E_{ref} is specified before attribute comparisons and may be empty. Changing attributes does not rebuild relations in the attribute path. The structure path uses a fixed relation-aware input adapter, so changing relations does not change the downstream architecture. The paths are evaluated separately, and their outputs are neither exchanged nor composed. Preserve supplies Z_0 and an empty proposed edge set, so the graph-combination rule retains the reference graph. Preserve uses the same downstream model architecture, input adapter, and training protocol as candidate programs within each path.

Evaluation protocol. The protocol $\mathcal{Q}$ fixes the model architecture, input adapter, training procedure, data partitions, access permissions, and resource limits. Train, Search, Commit, and Test have distinct evaluation roles. Access to unlabeled records is specified separately for the declared inductive or transductive setting. Training and evaluation components can read their designated labels, whereas refinement programs cannot. Search provides controlled feedback for program optimization. Commit confirms frozen candidates, and Test is reserved for final reporting. Neither Commit nor Test provides feedback for program revision or memory updates within a search run.

Downstream utility. For an evaluation split s, let $\mathcal{I}_b^s(P)$ denote its task inputs under $\mathcal{Q}$ and Y_s its supervision. Using a common set of K training seeds $\{\xi_k\}_{k=1}^K$, we define

$$f_{P,b,k} = \mathrm{Train}_{\mathcal{Q}}\big(\mathcal{I}_b^{\mathrm{tr}}(P), Y_{\mathrm{tr}}, \xi_k\big)\,,$$
$$\widehat{V}^s_{\mathcal{T},\mathcal{Q}}(P,b) = \frac{1}{K}\sum_{k=1}^{K} \mathcal{M}_{\mathcal{T}}(f_{P,b,k}, \mathcal{I}_b^s(P), Y_s)\,. \tag{20}$$

The superscript tr denotes Train, and $s \in \{\mathrm{search}, \mathrm{commit}, \mathrm{test}\}$ identifies the evaluation split. $\mathrm{Train}_{\mathcal{Q}}$ denotes finite training rather than exact optimization. Model parameters are fitted separately for each program and seed, while the architecture and training rules remain fixed. Cheaper proxy evaluations are distinguished from this reference protocol.

The gain relative to Preserve under the same path and protocol is

$$\widehat{\Delta}^s_{\mathcal{T},\mathcal{Q}}(P,b) = \widehat{V}^s_{\mathcal{T},\mathcal{Q}}(P,b) - \widehat{V}^s_{\mathcal{T},\mathcal{Q}}(\mathrm{Id},b). \tag{21}$$

The objective in Equation 4 maximizes this gain on Search. Refinement programs are optimized through measurements of downstream utility.

E Experimental Settings

This section consolidates the protocol shared by all methods and the principal MATAR settings. All methods share the same frozen splits, evaluation queries, downstream architecture, maximum optimizer-step budget, early-stopping rule, and seed discipline; checkpoints are selected on the task's search metric, and the held-out test split is scored exactly once per final output. Refinement seeds are 1103, 2207, and 3301. Q1 EC and MM use five training seeds, 601, 701, 809, 907, and 1009. For MATAR, the training-seed scores are averaged within each search repetition before computing the reported mean and sample standard deviation across search repetitions. Deterministic baselines report variation across training seeds, so their uncertainty terms have a different aggregation level.

MATAR search configuration. Attribute and structure paths are searched independently, each with 18 candidate slots. Within each path, candidates are ranked by mean paired Search gain minus half its standard error. Program validation and bounded repairs precede training, and failures remain recorded within the candidate budget. One finalist is selected across paths using relative Search gain and evaluated under the one-sided 95% paired t rule in Appendix F, with Preserve fallback included in the reported result. Table 4 lists the controls of the separately frozen Q3 sensitivity configuration. Its diagnostic sampling, prompt examples, degree cap, and temperature reference values belong to that derived configuration and are not defaults of the Q1 source snapshot.

Table 4: Controls and reference values in the Q3 sensitivity configuration.

Knob	Default	Sweep range
Diagnostic coverage (probe fraction)	0.05	[0.01, 0.10]
Structure constraint (max degree)	30	[4, 30]
LLM proposal temperature	0.1	[0, 0.7]
Planner context examples	8	[2, 32]
Candidate slots per path	18	fixed
Commit margin	0	fixed

E.1 Hyperparameter Sensitivity Protocol

Q3 uses a separately frozen derivative of the search implementation with explicit diagnostic sampling, proposal temperature, and planner context controls. We vary probe fraction over $\{0.01, 0.02, 0.03, 0.05, 0.10\}$ on EC Grocery and Movies, temperature over $\{0, 0.1, 0.3, 0.5, 0.7\}$ on MR Toys and Bili Music, and context size over $\{2, 4, 8, 16, 32\}$ on MM Bili Cartoon and Bili Food. Toys is an additional MR dataset for this study and does not replace Flickr30k in Q1. Other controls remain at the reference values in Table 4, including 18 candidate slots per path. These reference runs belong to Q3 and are not reused Q1 results.

For N public records and probe fraction p, a fixed nested permutation supplies $\min(N, 1000, \max(64, \lfloor pN \rfloor))$ records for feature-norm and paired-cosine diagnostics. This sample changes the planner profile without subsampling the complete Search split. The temperature is passed to the LLM for proposal generation, while context examples are deterministic, equally spaced public records containing text and an image-feature availability indicator. Each setting uses search seeds 1103, 2207, and 3301, with five training seeds (601, 701, 809, 907, and 1009) per selected output.

E.2 Cross-Dataset Experience Transfer Protocol

Q4 represents source experience as an executable refinement program and transfers the Q1 Search-selected program from each of three independent source refinement seeds, including programs whose source Commit decision returned Preserve. The selected attribute or relation path, operator parameters, and generated-tool definitions are frozen before target evaluation. The target model is retrained under its Q1 protocol, without transferring source model parameters or conducting target program search. For each source seed, we average target training scores under the task-specific seed configuration, then aggregate across source seeds. EC uses Macro-F1, LP uses ROC-AUC, MR uses bidirectional MRR, and MM uses paired cosine similarity, all on the $\times 100$ display scale. Metrics from different tasks are not pooled.

Table 3 reports direct target Test scores of frozen programs, rather than final scores after target Commit acceptance or Preserve fallback. Target Test scores are computed only after Commit decisions are saved and do not select programs. The reported means exceed Preserve in all eight directions, but aggregate Test scores do not determine individual Commit decisions or establish that every transferred program improves performance.

This study evaluates experience transfer through direct reuse of frozen source refinement programs, with Preserve as the no-refinement reference and no additional target program search. The EC pilot Test results for source seed 1103 were visible before the three-seed extension, making this batch exploratory. Only the source-selected path is transferred in each case, so comparisons across tasks or directions do not isolate the effects of attribute versus relation refinement. For MM, increased paired similarity must also be assessed against nonpaired similarity and representation dispersion before inferring improved alignment.

E.3 Robustness Evaluation on Movies

Q5 evaluates EC on Movies under text, image, and training-label corruption at nominal ratios $r = 0, 0.1, \ldots, 0.9$. Each condition starts from the same clean assets and retains the original Train, Search, Commit, and Test splits. For each modality, a fixed random target permutation identifies the selected records, and an independent source permutation supplies replacement feature rows. Increasing ratios use longer prefixes of these permutations, with $\lfloor rN \rfloor$ records selected from the full dataset of N entities. Text descriptions follow the replacement text feature rows, and public image paths are removed at every ratio, including $r = 0$. Label corruption selects $\lfloor rN_{\mathrm{Train}} \rfloor$ training records and independently draws replacement labels uniformly from the 20 classes, leaving Search, Commit, and Test labels unchanged. The nominal ratio measures selection rather than actual change, since a replacement can match the original feature row or label. All methods receive the same corrupted inputs at each ratio.

The five baselines are DataFlow, COMET, MM-Gen, FLYT, and EcoDatum, selected after inspecting the Q1 clean EC-Movies Test ranking. Each condition uses noise seed 42 and training seed 601, with search seed 1103 for MATAR. Attribute and relation paths are searched independently on the corrupted inputs, and the finalist is selected using Search gains relative to Preserve. The finalist and Preserve are then trained with the same seed and compared on Commit Macro-F1. The finalist is accepted only if its Commit score exceeds that of Preserve, otherwise Preserve supplies the final Test result. This Q5 acceptance rule uses a single paired training seed and does not constitute a statistical test across seeds.

The three panels reuse the same clean run for each method at $r = 0$, which is a single-seed Q5 measurement rather than the Q1 multi-seed mean. Each curve point reports one Test Macro-F1 value without an uncertainty interval, and the panels use separate vertical scales. The post hoc

baseline selection and single-run design limit the findings to the tested conditions and do not establish statistical significance or a general robustness advantage.

F Refinement Implementation and Evaluation Boundaries

The formal method uses the frozen `source_v6` adaptive implementation with independent attribute and relation sessions, an online controller, and generated residual feature operators. The underlying engine also supports a joint mode, but the main comparison uses the isolated A/R configuration. Separate joint-mode experiments do not define the main method or its ablations. Each path starts from the empty Preserve program, permits six operations, and allocates nine rounds with two proposal slots per round. Operators are represented by names and parameter dictionaries, and patches support insertion, removal, replacement, and movement subject to path and dependency checks.

Lexical augmentation uses word unigrams and bigrams or character n-grams of lengths three to five, sublinear term frequency, and at most 200,000 terms. The selected minimum document frequency lies between one and five, and the retained width lies between 16 and 512. SVD uses seed zero and seven iterations, fitting up to 512 components when the input dimensions permit and retaining the requested width. Frozen text encoders supply additional views, and each appended block is normalized and scaled by the median base-text norm and its selected weight. The evaluator maps any changed feature width back to its original dimension through a deterministic signed projection before model fitting. Feature statistics and encoder views use public unlabeled records under the declared transductive setting.

Relation construction uses exact cosine neighbors from text, image, or fused base features, with k between one and 32, a similarity threshold, and union or mutual-neighbor mode. Successive graph operations can replace, union, intersect, or filter the proposed edges. The formal engine bounds the total proposed edges at one million, but the neighbor count is not a hard bound on undirected degree after union.

Generated attribute operators use expressions bounded by depth eight and 48 nodes. They cannot read files, entity identifiers, private labels, or evaluator state. Registration tests finite, shape-preserving behavior on varied inputs, checks entity-permutation equivariance, and rejects identities, uniform scaling, constant outputs, and duplicate behavior on validation probes. The residual adapter applies Eq. 9 with a searched mixing coefficient in $[0.01, 0.5]$. These are executable-contract checks rather than a guarantee of improved downstream performance.

Search evaluates valid new outputs at full fidelity. The ranking coefficient in Eq. 12 is $\lambda = 0.5$, and the standard-error term is set to zero when only one Search seed is available. Paired statistics use the evaluator-native metric units consistently for the parent and child. The formal implementation does not use the earlier Gaussian posterior ranking or measured-panel selection override. Program fingerprints cover operator names and parameters, and output deduplication hashes the feature arrays and proposed edges. This is sufficient for the formal MATAR catalog because its candidates do not change training-record eligibility. It should not be interpreted as a general equivalence test for selection methods with different training subsets.

After each path identifies a finalist, the runner compares its mean Search gain against its own Preserve calibration. The largest gain determines the candidate for Commit, with ties resolved in the order Preserve, attribute, then relation. Only this selected candidate and Preserve are fitted on the Commit training seeds. For paired differences $d_j = m_j^{\mathrm{commit}}(P) - m_j^{\mathrm{commit}}(\mathrm{Id})$, the formal repeated-seed rule is

$$B(P) = \bar{d} - t_{0.95,n-1}\frac{s_d}{\sqrt{n}}, \qquad \mathrm{Accept}(P) = \mathbf{1}\{B(P) > 0\}. \tag{22}$$

Here n is the declared number of paired training seeds and s_d is their sample standard deviation. The runner uses one finalist, zero acceptance margin, and no three-candidate correction. This rule is distinct from the generic multi-candidate session policy exposed by the evaluation library. Q5 instead uses its explicitly declared single-seed positive-gain rule. Test scores the accepted Commit checkpoints or the Preserve checkpoints, with no further program search.

G Ablation Definitions

The Full configuration enables program revision, paired Search feedback, experience interpretation, and the numerical controller. All variants retain independent A/R sessions, the candidate-slot budget, Search-based finalist selection, and the final paired Commit comparison. Removing a Search component does not remove Commit acceptance or Preserve fallback.

The w/o TATO variant disables generated operators and restricts proposals to one operation from a fixed menu. In the revised menu, both operator parameters and the one-step structure are fixed, so the agent can select a recipe but cannot tune its parameters, compose operations, or change their order. The original menu variant allowed legal parameter selection within its single slot. These variants must be distinguished by their recorded ablation revision when associating a result with an execution manifest.

The w/o PRE variant assigns independent candidate training seeds and exposes absolute Search scores instead of paired gains. Parent-relative differences, seed-gain vectors, and replay diagnostics are withheld from the proposal and feedback contexts. Its controller reward uses the absolute best Search score with the same failure and call penalties, rather than paired local progress. The final Commit test remains paired against Preserve.

The w/o EPA variant disables experience-memory reads and writes and the feedback-interpretation call that creates lessons. It retains the current parent, direct Search measurements, and the numerical controller. This ablation therefore measures the effect of interpreted experience reuse, not removal of every adaptive mechanism or of controller learning.

H Cost and Complexity

The search cost of MATAR is dominated by downstream fits. Each valid new candidate requires a task-model fit for each declared Search seed. Each refinement path has 18 proposal slots, while invalid or duplicate outputs can consume a slot without another fit. Commit fits the single finalist and Preserve on the declared paired training seeds. Fallback reuses the Preserve checkpoints already fitted for this comparison. Language-model proposals and bounded repairs add external calls; a 12-round development run recorded 12 controller updates and 55 external calls, and cost grows roughly linearly in the number of candidate slots and search seeds. All methods share the downstream model and the optimizer-step budget, so the differences between methods are search-side fits and API calls rather than downstream architecture.