ADAPT: AN AGENT-DRIVEN AUTONOMOUS PARADIGM FOR TRANS-SPECIES SINGLE-CELL ANNOTATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Cross-species knowledge transfer provides a powerful strategy for single-cell RNA sequencing analysis in species with limited reference annotations, while offering insights into cellular diversity and evolutionary biology. However, cross-species transfer remains challenging due to species-specific expression programs, cellular differences, and complex gene homology. Existing methods often rely on predefined adaptation strategies that require manual trial-and-error for new transfer scenarios and lack closed-loop feedback to refine these strategies based on experimental outcomes. To address this challenge, we introduce ADAPT, a feedback-driven agentic framework for cross-species single-cell annotation that automatically searches and selects candidate adaptation approaches for transferring cell-type knowledge from reference to target species. Starting from frozen scGPT embeddings, ADAPT performs iterative hypothesis-driven exploration: an outer agent proposes scientific hypotheses and experimental plans, while an inner agent implements candidate adaptation models under controlled experimental protocols. The validation results are fed back to guide subsequent search rounds. We evaluate ADAPT on two cross-species transfer settings involving knowledge transfer from Tabula Sapiens to Tabula Muris and Macaca fascicularis. ADAPT achieves the highest average shared-class Macro-F1 among the compared methods, reaching 0.81 for human-to-mouse transfer and 0.73 for human-to-macaque transfer. These results suggest the potential of feedback-driven agentic search for cross-species adaptation. The code is available at https://anonymous.4open.science/r/Cross_Species_Transfer-7647

1 INTRODUCTION

Cross-species transfer of cell-type knowledge is an important problem in single-cell RNA sequencing analysis Yuan et al. (2026), which is particularly valuable when high-quality annotations are limited for target species Tarashansky et al. (2021); Shafer (2019). However, cross-species annotation remains challenging due to species-specific gene expression programs, divergent cell-type characteristics, and complex gene homology relationships Rosen et al. (2024); Song et al. (2023). A model trained to recognize cell types in one species may therefore fail to generalize to another species. This motivates the development of computational methods that can effectively transfer cell-type knowledge across species.

Existing cross-species alignment methods typically transfer knowledge by constructing explicit correspondences between source and target species or learning shared representations Stuart et al. (2019); Tarashansky et al. (2021); Liu et al. (2023); Xu et al. (2021). However, these methods generally employ predefined modeling and adaptation mechanisms, whose performance may vary across species, tissues, and transfer settings Zhong et al. (2025), while automatic selection of downstream strategies based on experimental outcomes remains largely unexplored. Therefore, developing an automated, closed-loop framework for adaptation search may reduce the need for manual model selection across different transfer settings.

Inspired by recent advances in LLM agents for automated hypothesis testing and closed-loop search Lu et al. (2026); Baek et al. (2025); Jansen et al. (2025), we introduce ADAPT, an agent-driven

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.

framework that automatically searches for adaptation mechanisms to transfer cell-type knowledge from a labeled reference species to a target species. ADAPT targets two major limitations of current methods: their reliance on manually designed adaptation strategies across varying transfer scenarios, and their lack of closed-loop feedback to dynamically refine strategies based on experimental outcomes.

To address the first limitation, ADAPT explores adaptive downstream models built on frozen scGPT Cui et al. (2024) embeddings to obtain task-adaptive representations for cross-species cell-type annotation. Rather than applying a fixed adaptation strategy across transfer scenarios, ADAPT automatically searches for scenario-specific downstream adaptation mechanisms, including model architectures, training objectives, and optimizer configurations.

In response to the second limitation, we introduce a closed-loop agentic architecture in which an outer agent formulates hypotheses about adaptation mechanisms and designs experimental plans, while an inner agent implements candidate models. Then models are trained on labeled source and unlabeled target data, while target labels are accessed only for evaluation. The outer agent leverages complete-round results and candidate history to assess hypotheses and plan subsequent search rounds, forming a feedback-driven iterative loop. Finally, the selected model is evaluated on the sealed test set.

To evaluate the cross-species transfer capability of ADAPT, we conduct experiments under two cross-species transfer settings: (1) human-to-mouse transfer from Tabula Sapiens Consortium* et al. (2022) to Tabula Muris Schaum et al. (2018), and (2) human-to-macaque transfer from Tabula Sapiens to Macaca fascicularis Han et al. (2020). Across both settings, ADAPT achieves improved performance over existing cross-species annotation methods, demonstrating the effectiveness of feedback-driven adaptation strategy discovery. Our contributions are summarized as follows:

- We propose ADAPT, an agent-driven framework for cross-species single-cell annotation. It integrates an LLM agent framework into cross-species tasks to automatically search for downstream adaptation strategies across diverse transfer scenarios, addressing the reliance of existing methods on predefined adaptation strategies.
- We develop a feedback-driven agentic search process where an outer agent formulates hypotheses and experiment plans, while an inner agent implements candidate models. Performance feedback from each complete round is returned to the outer agent to iteratively guide subsequent search.
- We evaluate ADAPT on two cross-species transfer settings spanning mouse and macaque targets, using Tabula Sapiens as the human reference. Across both settings, ADAPT achieves improved performance over conventional cross-species annotation methods in Macro-F1 scores.

2 RELATED WORK

Cross-species Cell-type Annotation Single-cell RNA sequencing has become an essential technology for profiling the transcriptomes of large numbers of individual cells across diverse tissues, organs, and species Kolodziejczyk et al. (2015); Zheng et al. (2017). Analysis of cross-species scRNA-seq data helps investigate the evolutionary conservation of cell types while facilitating knowledge transfer from well-studied to less-studied species Zhong et al. (2025). A key task in such analyses is cell type annotation, yet manual annotation is time-consuming and partially subjective Cheng et al. (2023); Hou & Ji (2024). Moreover, cross-species annotation is further complicated by species-specific gene expression programs, divergent cell-type characteristics, and complex gene homology relationships Rosen et al. (2024); Loft et al. (2025); Haghverdi et al. (2018). These challenges motivate the development of models for transferring annotation knowledge across species.

Models for Cross-species Annotation

Existing methods typically transfer cell-type knowledge by constructing cross-species correspondences or learning shared representations. Seurat performs label transfer through anchor-based correspondences between reference and query datasets Stuart et al. (2019), while SAMap incorporates protein sequence similarity and iteratively refines gene and cell correspondences for cross-species alignment Tarashansky et al. (2021). CAME represents cells, genes, and cross-species gene relation-

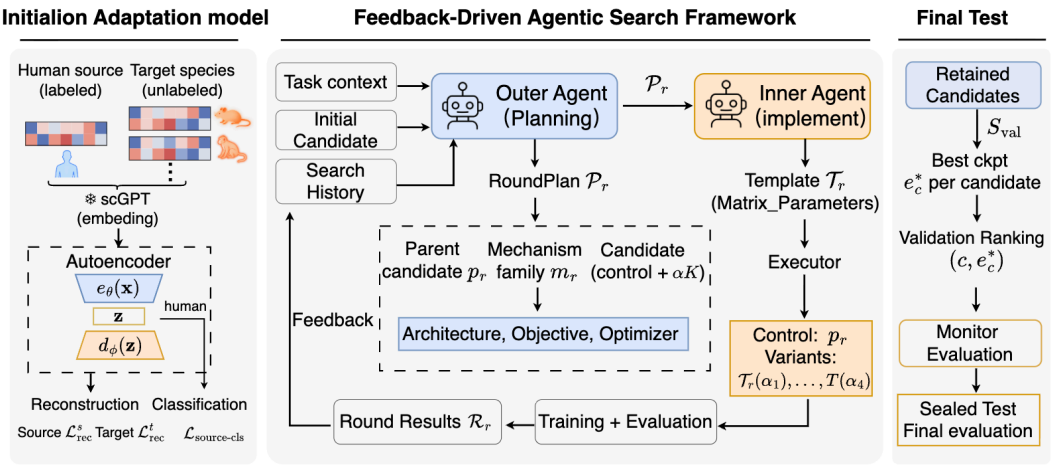

Fig. 1: Overview of ADAPT. Given labeled human source cells and unlabeled target-species cells, ADAPT first obtains fixed cell representations using a frozen scGPT encoder and initializes a downstream adaptation model. A feedback-driven agentic search framework then iteratively explores adaptation strategies through an outer planning agent and an inner implementation agent, with candidates evaluated under a standardized training and validation pipeline. The final model is selected through a deterministic post-search protocol and evaluated on sealed test data for cross-species cell-type annotation.

ships as a heterogeneous graph and learns cross-species representations through graph-based modeling Liu et al. (2023). scANVI learns a probabilistic latent representation through a semi-supervised generative model, enabling cell-type annotation transfer across datasets Xu et al. (2021); Lopez et al. (2018). These methods have improved cross-species cell-type annotation, but several limitations remain. They rely on predefined modeling strategies that may not generalize equally well across species, tissues, and cell types Zhong et al. (2025). Adapting them to new species also requires manual hypothesis design and experimentation, limiting the range of mechanisms explored. Beyond specialized cross-species annotation methods, recent advances in single-cell foundation models have enabled large-scale representation learning from diverse single-cell datasets. Models such as scGPT Cui et al. (2024) and Geneformer Theodoris et al. (2023) learn transferable cellular representations through large-scale pretraining, providing powerful embeddings for downstream analysis. However, these models typically rely on fixed pretrained representations and do not automatically optimize downstream adaptation strategies for specific cross-species transfer scenarios.

Agent-Driven Automated Scientific Search Language-model agents can formulate scientific hypotheses, design candidate methods, interpret experimental feedback, and iteratively refine the search strategies, making them well suited for automated scientific search Yao et al. (2022); M. Bran et al. (2024); Lu et al. (2026). In this work, we apply this feedback-driven search paradigm to cross-species single-cell annotation. Specifically, an outer agent proposes testable scientific hypotheses and candidate mechanisms, while an inner agent implements the candidates. A fixed executor then performs model training and validation. After each completed round, the outer agent uses the experimental results to update the search direction and iteratively refine candidate mechanisms, which enables the automatic exploration of effective cross-species adaptation strategies across target species, tissues, and cell-type compositions.

3 METHOD

3.1 PROBLEM SETUP

We transfer cell-type knowledge from human reference data to other target species. Let

$$\mathcal{G} = \{g_1, g_2, \ldots, g_G\}$$

denote the set of profiled genes. The gene expression profile of a cell is

$$r = (r(g_1), r(g_2), \ldots, r(g_G)) \in \mathbb{R}_+^G,$$

where $r(g_i)$ denotes the expression level of gene g_i. The labeled human source dataset and the unlabeled target-species dataset are defined as

$$\mathcal{D}^s = \{(\mathbf{r}_i^s, y_i^s)\}_{i=1}^{n_s}, \qquad \mathcal{D}^t = \{\mathbf{r}_j^t\}_{j=1}^{n_t},$$

where $y_i^s \in \mathcal{Y}$ denotes the cell-type label of the i-th human source cell. All cells are first encoded by a frozen scGPT encoder $\phi(\cdot)$ into a shared embedding space:

$$x_i^s = \phi(r_i^s), \qquad x_j^t = \phi(r_j^t).$$

Downstream learning uses only these embeddings. Starting from an initial adapter, an LLM searches over candidate modules f_θ; each candidate is trained on labeled source embeddings and unlabeled target-train embeddings. After training, a source-only logistic regression probe h is fitted on $\{f_\theta(\mathbf{x}_i^s)\}$. The prediction for a target cell is

$$\hat{y}_j^t = h\big(f_\theta(\mathbf{x}_j^t)\big) \in \mathcal{Y}.$$

Target cells are split into train, validation, monitor, and test partitions. Only the train split is used for unlabeled adaptation; validation guides the LLM search; monitor is reserved for post-search selection; and test remains sealed until final evaluation.

3.2 Model Overview

As shown in Figure 1, ADAPT formulates cross-species single-cell annotation as a feedback-driven automated scientific search process. Before search begins, the system extracts unified embeddings for human reference cells and target-species cells using a frozen scGPT model, and treats these embeddings as fixed inputs to all downstream candidates. It further provides an MLP autoencoder baseline that learns a cross-species adaptation representation and serves as both the starting point and performance reference for the first round of search. In the first round, the baseline is used as the round control. An outer agent combines this baseline with the task context to propose scientific hypotheses and generate a RoundPlan, while an inner agent implements the corresponding candidate variants. The control and candidate variants, together with the frozen embeddings, are then passed into a fixed pipeline for candidate-code validation, training, and validation-based evaluation, producing a complete round result. In subsequent rounds, the outer agent updates the search direction using historical candidates and complete experimental outcomes, generates a new RoundPlan, and thereby forms a feedback-driven iterative loop. After the search terminates, the system selects the final model under a fixed protocol and performs sealed test evaluation.

3.2.1 Cell Representation and Initial Adaptation Model

ADAPT focuses on cross-species transfer from labeled human source cells to unlabeled target-species cells. Source labels provide supervision for learning cell-type information, while unlabeled target-training cells expose the adaptation model to the target-species data distribution. We first encode source and target cells using a pretrained scGPT model, obtaining 512-dimensional embeddings:

$$\mathbf{z}_i^s, \mathbf{z}_j^t \in \mathbb{R}^{512}.$$

The embeddings are standardized using the mean and standard deviation from the source-training subset. scGPT remains frozen throughout downstream training, and the resulting standardized embeddings, denoted by $\tilde{\mathbf{z}}$, are treated as fixed inputs to downstream candidates.

The initial downstream adaptation model is an MLP autoencoder equipped with a source-only cell-type classification head. The encoder maps each standardized 512-dimensional embedding through a 128-dimensional ReLU hidden layer to a 64-dimensional latent representation with no activation:

$$\mathbf{u} = g_{\text{enc}}(\tilde{\mathbf{z}}) \in \mathbb{R}^{64}. \tag{1}$$

The decoder maps the latent representation through a 128-dimensional ReLU hidden layer to reconstruct the standardized 512-dimensional input:

$$\hat{\tilde{\mathbf{z}}} = g_{\text{dec}}(\mathbf{u}) \in \mathbb{R}^{512}. \tag{2}$$

Both source-training and unlabeled target-training embeddings are reconstructed through the shared autoencoder, while only source latent representations are supervised by a neural classification head using source cell-type labels. For evaluation, the neural head is discarded, and a fixed source-only logistic-regression probe is trained on labeled source representations and applied to target representations. Using the same probe protocol across candidates makes the comparison reflect representation quality rather than differences in classification heads.

The initialization objective combines source cell-type classification with reconstruction of source and target-training embeddings:

$$\mathcal{L}_{\text{init}} = \mathcal{L}_{\text{source-cls}} + 0.1\,\mathcal{L}_{\text{rec}}, \tag{3}$$

where

$$\mathcal{L}_{\text{source-cls}} = \text{CE}\big(\mathbf{s}^s, y^s\big), \tag{4}$$

and

$$\mathcal{L}_{\text{rec}} = \frac{1}{2}\left[\text{MSE}\big(\hat{\tilde{\mathbf{z}}}^s, \tilde{\mathbf{z}}^s\big) + \text{MSE}\big(\hat{\tilde{\mathbf{z}}}^t_{\text{train}}, \tilde{\mathbf{z}}^t_{\text{train}}\big)\right]. \tag{5}$$

Here, $\mathcal{L}_{\text{source-cls}}$ enables the latent representation to retain source cell-type-discriminative information, whereas $\mathcal{L}_{\text{rec}}$ enables the encoder to preserve information from both source and target embedding distributions. In this way, target-training cells contribute to representation learning without requiring target labels. The resulting model serves as the initial candidate for the subsequent agentic search.

3.2.2 Agentic Planning and Model Variant Generation

Starting from the initial downstream candidate, ADAPT performs iterative search for improved cross-species adaptation models using two cooperating agents. Both agents are instantiated with the DeepSeek-V4-Flash as the planning module Xu et al. (2026). The outer agent determines the scientific direction of each search round by formulating a hypothesis and specifying the experimental plan, while the inner agent translates this plan into executable model variants.

In the first round, the outer agent receives the task definition and the initial candidate. In later rounds, it also receives a compact summary of the search history and the complete results of the preceding round. Based on this context, it proposes a falsifiable hypothesis and constructs a *RoundPlan* specifying the parent candidate, the mechanism family to modify, the proposed modification, and the configurations to be evaluated.

Let p_r denote the parent candidate selected for round r. In the first round, p_r is the initial downstream candidate; in subsequent rounds, the outer agent selects p_r from eligible historical candidates. The control and the four generated candidates share this parent.

Each round modifies exactly one mechanism family:

$$m_r \in \{\text{architecture}, \text{objective}, \text{optimizer}\}.$$

An architecture round may modify only the model construction, encoder, and forward computation; an objective round may modify only the training objective; and an optimizer round may modify only the optimizer configuration. All remaining components are kept identical to the parent p_r, so the round tests one isolated change.

Given the *RoundPlan*, the inner agent implements the proposed modification as an executable template T_r. In our experiments, each round evaluates five candidates,

$$\mathcal{E}_r = \{c_{r,0}, c_{r,1}, c_{r,2}, c_{r,3}, c_{r,4}\}, \tag{6}$$

consisting of one control and four generated candidates. The control is an unchanged rerun of the parent implementation,

$$c_{r,0} = \text{Rerun}(p_r), \tag{7}$$

while the four generated candidates instantiate the same template using frozen configurations $\boldsymbol{\alpha}_{r,1}, \ldots, \boldsymbol{\alpha}_{r,4}$ specified by the *RoundPlan*:

$$c_{r,k} = T_r(\boldsymbol{\alpha}_{r,k}), \qquad k = 1, \ldots, 4. \tag{8}$$

The generated candidates therefore share the same implementation structure and differ only in their assigned configurations. The rerun of p_r provides a control evaluated under the same execution and training protocol as the generated candidates. Before training, every candidate must pass fixed safety and interface checks; the control is copied verbatim from the historical parent and revalidated in the same training sandbox.

3.2.3 MODEL TRAINING, EVALUATION, AND FEEDBACK

After contract validation, each candidate in the frozen *RoundPlan* is executed under a fixed training and evaluation protocol. The target-training split, source-label usage, random seeds, checkpointing, prediction generation, and evaluation procedure are held constant across candidates; only the planned mechanism family and its assigned configurations $\alpha_{r,k}$ are allowed to change.

Each trained candidate is evaluated using a common representation-based protocol. For candidate c, the trained encoder produces source and target representations

$$\mathbf{u}^s = g_c(\tilde{\mathbf{z}}^s), \qquad \mathbf{u}^t = g_c(\tilde{\mathbf{z}}^t). \tag{9}$$

A fixed logistic-regression probe is fitted exclusively on labeled source representations and then applied to target representations. Target-validation labels are used only after predictions have been generated to compute shared cell-type Macro-F1. Candidates are ranked by the source-only logistic-regression probe on shared-class Macro-F1. The neural-head Macro-F1 is recorded but not used for ranking.

After all candidates in round r have completed, the system deterministically aggregates their results and compares the generated candidates with the rerun control. Only the complete round result is released to the outer agent, preventing subsequent decisions from depending on partial outcomes. The outer agent then evaluates the round hypothesis, updates the campaign memory with the resulting evidence, and either constructs the next frozen *RoundPlan* or terminates the search. The overall search process can be summarized as

$$\mathcal{M}_r \xrightarrow{\text{planning}} \mathcal{P}_r \xrightarrow{\text{generation}} \mathcal{C}_r \xrightarrow{\text{train+eval}} \mathcal{R}_r \xrightarrow{\text{feedback}} \mathcal{M}_{r+1}, \tag{10}$$

where $\mathcal{M}_r$, $\mathcal{P}_r$, $\mathcal{C}_r$, and $\mathcal{R}_r$ denote the campaign memory, frozen round plan, candidate set, and complete round result, respectively.

In our experiments, the search budget is set to 50 rounds. The search process typically reaches a stable validation performance before exhausting the full budget, indicating that the feedback loop can efficiently identify effective adaptation strategies without requiring unlimited exploration.

3.2.4 FINAL MODEL SELECTION AND TEST EVALUATION

After search ends, we merge the top-10 candidates of each of the three replicas, ranked by search-time S_{val}, and deduplicate them. For each remaining candidate we re-evaluate the epoch checkpoints shared by all three seeds on validation and keep $e_c^* = \arg\max_e S_{\text{val}}(c, e)$. The top $K = 3$ pairs then go to monitor, which is never shown to either agent:

$$(c^*, e^*) = \arg \max_{(c,e)\in\mathcal{K}} S_{\text{monitor}}(c, e). \tag{11}$$

All scores use the same rule on the source-only logistic-regression probe, $S = \bar{F}_{\text{shared}} - 0.25 \cdot \text{std}$, with three-seed mean and sample standard deviation of shared-class Macro-F1. The selected pair is evaluated once on the sealed test set; no further tuning is allowed after test labels are opened.

4 EXPERIMENT

4.1 EXPERIMENT SETUP

Datasets We evaluate ADAPT on cross-species single-cell annotation using two transfer settings. The first setting evaluates human-to-mouse transfer from Tabula Sapiens Consortium* et al. (2022) to Tabula Muris Schaum et al. (2018), which contains mouse single-cell transcriptomic profiles with curated cell-type annotations across multiple tissues. The second setting evaluates human-to-macaque transfer from Tabula Sapiens to Macaca fascicularis Han et al. (2020), representing transfer across primate species. The source species provides labeled cells for training, while target-species train cells are used as unlabeled inputs. Target cells are split independently of labels into a fixed 6:1:1:2 train/validation/monitor/test partition, where validation guides search, monitor supports post-search selection, and test is accessed only after model freezing. Results are averaged over three folds and three training seeds.

Table 1: Shared-class and all-target Macro-F1 for human-to-mouse cell-type annotation.

Setting	Tissue	CAME	Seurat	SAMap	scANVI	ADAPT (ours)
Shared-class	Bladder	0.92 ± 0.05	$\mathbf{0.98} \pm 0.02$	0.97 ± 0.02	0.93 ± 0.13	0.97 ± 0.02
	Heart and Aorta	$\mathbf{1.00} \pm 0.00$	$\mathbf{1.00} \pm 0.01$	$\mathbf{1.00} \pm 0.00$	$\mathbf{1.00} \pm 0.00$	$\mathbf{1.00} \pm 0.01$
	Kidney	0.56 ± 0.03	0.56 ± 0.03	0.55 ± 0.04	0.43 ± 0.18	$\mathbf{0.70} \pm 0.13$
	Limb Muscle	$\mathbf{0.97} \pm 0.02$	0.96 ± 0.01	0.95 ± 0.00	0.80 ± 0.10	$\mathbf{0.97} \pm 0.02$
	Liver	0.79 ± 0.12	0.44 ± 0.13	0.42 ± 0.04	0.28 ± 0.05	$\mathbf{0.83} \pm 0.09$
	Lung	0.69 ± 0.04	0.77 ± 0.03	0.75 ± 0.02	0.62 ± 0.05	$\mathbf{0.79} \pm 0.03$
	Mammary Gland	$\mathbf{0.96} \pm 0.01$	0.95 ± 0.01	0.89 ± 0.02	0.49 ± 0.14	0.95 ± 0.01
	Marrow	0.32 ± 0.04	0.31 ± 0.03	0.32 ± 0.02	0.18 ± 0.05	$\mathbf{0.40} \pm 0.01$
	Spleen	0.68 ± 0.04	0.72 ± 0.01	0.74 ± 0.01	0.48 ± 0.13	$\mathbf{0.75} \pm 0.03$
	Thymus	0.30 ± 0.13	0.26 ± 0.04	0.17 ± 0.02	0.26 ± 0.09	$\mathbf{0.61} \pm 0.04$
	Tongue	0.59 ± 0.32	0.90 ± 0.03	$\mathbf{0.94} \pm 0.00$	0.91 ± 0.10	$\mathbf{0.94} \pm 0.02$
	Trachea	$\mathbf{0.78} \pm 0.01$	0.72 ± 0.01	0.52 ± 0.02	0.60 ± 0.07	0.77 ± 0.01
	Mean	0.71 ± 0.04	0.71 ± 0.01	0.69 ± 0.01	0.58 ± 0.02	$\mathbf{0.81} \pm 0.02$
All-target	Bladder	0.92 ± 0.05	$\mathbf{0.98} \pm 0.02$	0.97 ± 0.02	0.93 ± 0.13	0.97 ± 0.02
	Heart and Aorta	$\mathbf{0.72} \pm 0.00$	$\mathbf{0.72} \pm 0.01$	$\mathbf{0.72} \pm 0.01$	$\mathbf{0.72} \pm 0.00$	$\mathbf{0.72} \pm 0.01$
	Kidney	$\mathbf{0.15} \pm 0.01$	0.11 ± 0.01	0.11 ± 0.02	0.09 ± 0.03	$\mathbf{0.15} \pm 0.04$
	Limb Muscle	$\mathbf{0.97} \pm 0.02$	0.96 ± 0.01	0.95 ± 0.00	0.80 ± 0.10	$\mathbf{0.97} \pm 0.02$
	Liver	0.79 ± 0.12	0.44 ± 0.13	0.42 ± 0.04	0.28 ± 0.05	$\mathbf{0.83} \pm 0.09$
	Lung	0.63 ± 0.04	0.70 ± 0.02	0.69 ± 0.01	0.57 ± 0.05	$\mathbf{0.72} \pm 0.02$
	Mammary Gland	$\mathbf{0.96} \pm 0.01$	0.95 ± 0.01	0.89 ± 0.02	0.49 ± 0.14	0.95 ± 0.01
	Marrow	0.26 ± 0.03	0.25 ± 0.02	0.27 ± 0.02	0.15 ± 0.04	$\mathbf{0.33} \pm 0.01$
	Spleen	0.68 ± 0.04	0.72 ± 0.01	0.74 ± 0.01	0.48 ± 0.13	$\mathbf{0.75} \pm 0.03$
	Thymus	0.30 ± 0.13	0.26 ± 0.04	0.17 ± 0.02	0.26 ± 0.09	$\mathbf{0.61} \pm 0.04$
	Tongue	0.39 ± 0.21	0.60 ± 0.02	$\mathbf{0.63} \pm 0.00$	0.60 ± 0.07	0.62 ± 0.01
	Trachea	$\mathbf{0.78} \pm 0.01$	0.72 ± 0.01	0.52 ± 0.02	0.60 ± 0.07	0.77 ± 0.01
	Mean	0.63 ± 0.03	0.62 ± 0.01	0.59 ± 0.00	0.50 ± 0.02	$\mathbf{0.70} \pm 0.01$

Baselines We compare ADAPT with representative cross-species annotation approaches. Specifically, we include Seurat Stuart et al. (2019), which performs anchor-based label transfer; SAMap Tarashansky et al. (2021), which leverages cross-species gene relationships and iterative alignment to refine cell correspondences. CAME Liu et al. (2023), which learns cross-species representations through heterogeneous graph modeling; and scANVI Xu et al. (2021), which learns a semi-supervised probabilistic latent representation for annotation transfer. All methods are evaluated under the same cross-species transfer settings and sealed test protocol.

Metrics Since some target cell types lack human counterparts, we use shared-class Macro-F1 as the primary metric and all-target Macro-F1 as a secondary metric. Shared-class Macro-F1 is computed over cell types shared between the reference and target species, while all-target Macro-F1 additionally includes target-specific cell types. During search, candidates are ranked using validation shared-class Macro-F1. The final model is evaluated once on the sealed test split after model selection. Results are reported as mean $\pm$ standard deviation over the evaluated folds and random seeds.

4.2 EVALUATION ON TABULA MURIS

We first evaluate human-to-mouse transfer from Tabula Sapiens to Tabula Muris, where the target dataset is the Tabula Muris droplet atlas covering 12 organs. Each organ is paired with a matching Tabula Sapiens tissue and a fixed cell-type label mapping. As shown in Table 1, ADAPT achieves the highest overall shared-class Macro-F1 (0.81), outperforming all compared cross-species annotation methods in this benchmark. The improvements are particularly notable in tissues where existing methods show limited transfer performance, such as Kidney, Liver, Marrow, and Spleen, indicating that feedback-driven adaptation search can identify effective downstream representations beyond manually designed adaptation choices.

For the stricter all-target evaluation, which additionally considers target-specific cell types, ADAPT also achieves the highest overall Macro-F1 (0.70). The gap between shared-class and all-target

performance is mainly caused by the presence of target-specific cell types that are absent from the human reference. As shown in Sec. A.1, several tissues contain more target cell types than shared cell types, such as Kidney (8 vs. 3), Marrow (14 vs. 12), and Lung (13 vs. 12). These additional cell types are not covered by the source annotation space and therefore make the all-target metric a stricter assessment of cross-species transfer. Together, these results demonstrate that ADAPT can discover effective adaptation strategies across diverse cross-species transfer scenarios.

4.3 EVALUATION ON MACACA FASCICULARIS

We further evaluate ADAPT on human-to-macaque cell-type annotation using Macaca fascicularis. As shown in Table 2, ADAPT achieves the highest overall shared-class Macro-F1 score (0.73), outperforming all compared cross-species annotation methods. The improvements are particularly notable on tissues where baseline approaches show limited transfer performance, including Kidney (0.94 vs. 0.85 for Seurat) and PBMC (0.55 vs. 0.46 for CAME), demonstrating that feedback-driven adaptation search can identify effective representations for cross-species transfer.

We further evaluate all-target Macro-F1, which provides a stricter assessment by including target-specific cell types. ADAPT obtains the highest overall score (0.46), outperforming existing methods. The difference between shared-class and all-target performance mainly reflects the varying degree of label overlap across tissues. As shown in Sec. A.1, Kidney contains only two shared cell types but eleven target cell types, and Lung contains six shared cell types but ten target cell types. These additional target-specific classes increase the difficulty of all-target evaluation because they cannot be directly learned from labeled source cells.

Table 2: Shared-class and all-target Macro-F1 for human-to-macaque cell-type annotation.

Setting	Tissue	CAME	Seurat	SAMap	scANVI	ADAPT (ours)
Shared-class	Aorta	0.82 ± 0.05	0.74 ± 0.09	0.72 ± 0.05	0.57 ± 0.21	$\mathbf{0.86} \pm 0.03$
	Kidney	0.78 ± 0.25	0.85 ± 0.02	0.72 ± 0.03	0.48 ± 0.19	$\mathbf{0.94} \pm 0.01$
	Liver	0.69 ± 0.02	0.65 ± 0.03	0.36 ± 0.10	0.73 ± 0.04	$\mathbf{0.79} \pm 0.05$
	Lung	0.84 ± 0.02	0.84 ± 0.05	0.66 ± 0.05	0.76 ± 0.12	$\mathbf{0.86} \pm 0.02$
	PBMC	0.46 ± 0.05	0.46 ± 0.03	0.39 ± 0.08	0.37 ± 0.10	$\mathbf{0.55} \pm 0.05$
	Parotid	$\mathbf{0.44} \pm 0.02$	0.36 ± 0.03	0.39 ± 0.01	0.23 ± 0.04	0.42 ± 0.02
	Pancreas	0.66 ± 0.09	$\mathbf{0.71} \pm 0.02$	0.67 ± 0.02	0.42 ± 0.09	0.66 ± 0.02
	Mean	0.67 ± 0.04	0.66 ± 0.02	0.56 ± 0.01	0.51 ± 0.07	$\mathbf{0.73} \pm 0.01$
All-target	Aorta	0.51 ± 0.03	0.46 ± 0.06	0.45 ± 0.03	0.37 ± 0.16	$\mathbf{0.52} \pm 0.02$
	Kidney	$\mathbf{0.07} \pm 0.03$	0.06 ± 0.01	0.04 ± 0.00	0.04 ± 0.01	0.06 ± 0.00
	Liver	0.69 ± 0.02	0.65 ± 0.03	0.36 ± 0.10	0.73 ± 0.04	$\mathbf{0.79} \pm 0.05$
	Lung	0.40 ± 0.02	0.39 ± 0.02	0.33 ± 0.03	0.37 ± 0.05	$\mathbf{0.41} \pm 0.02$
	PBMC	0.46 ± 0.05	0.46 ± 0.03	0.39 ± 0.08	0.37 ± 0.10	$\mathbf{0.55} \pm 0.05$
	Parotid	$\mathbf{0.21} \pm 0.01$	0.17 ± 0.02	0.18 ± 0.01	0.11 ± 0.03	$\mathbf{0.20} \pm 0.01$
	Pancreas	0.66 ± 0.09	$\mathbf{0.71} \pm 0.02$	0.67 ± 0.02	0.42 ± 0.09	0.66 ± 0.02
	Mean	0.43 ± 0.01	0.41 ± 0.01	0.35 ± 0.01	0.35 ± 0.04	$\mathbf{0.46} \pm 0.01$

4.4 ABLATION STUDY

We evaluate the contributions of LLM-guided search, unlabeled target adaptation, and target reconstruction in ADAPT on the human-to-mouse setting. All variants use the same three-fold protocol and three random seeds, and are evaluated using shared-class and all-target Macro-F1. Results are shown in Figure 2. The full ADAPT model achieves the best performance, with Macro-F1 scores of 0.81 on shared classes and 0.70 on all-target classes. Removing the LLM Agent while retaining the fixed candidate and target adaptation branch reduces performance to 0.61 and 0.53. Removing both unlabeled target adaptation and LLM-guided search while retaining the fixed candidate reduces performance to 0.65 on shared classes and 0.55 on all-target classes. Removing the target reconstruction loss yields scores of 0.62 and 0.54. The scGPT + LogReg baseline directly applies logistic regression to frozen scGPT embeddings.

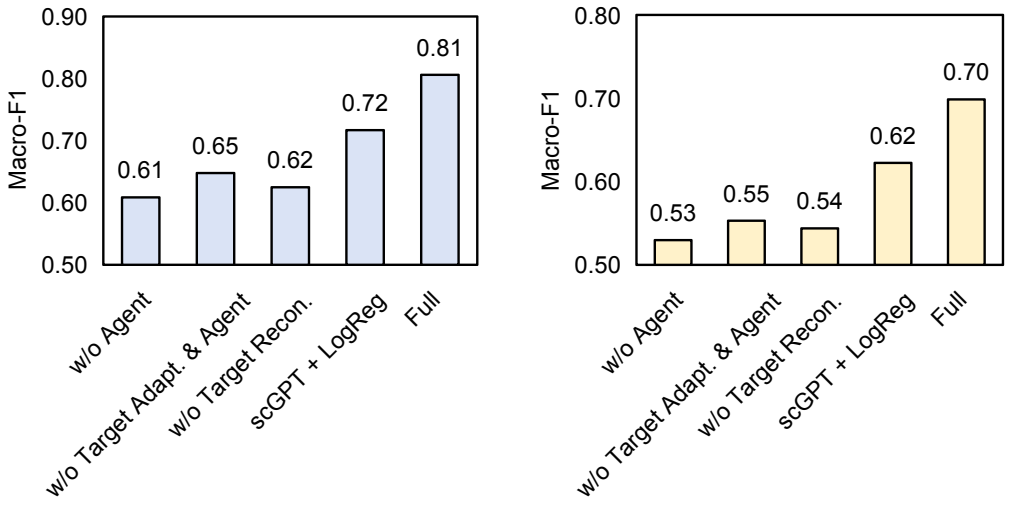

Fig. 2: The ablation experiments of ADAPT

Overall, the complete ADAPT framework achieves the strongest performance among the evaluated variants. The ablation results further suggest interactions among the searched configuration and target-side adaptation components.

5 CONCLUSION

In this work, we introduced ADAPT, a feedback-driven agentic framework for cross-species single-cell annotation that automatically discovers effective adaptation strategies. By combining an outer planning agent with an inner implementation agent, ADAPT transforms downstream adaptation design from a manually crafted process into an iterative hypothesis-driven search loop. Starting from frozen scGPT embeddings, ADAPT explores candidate architectures, objectives, and optimization strategies under a fixed evaluation protocol, while using experimental feedback to guide subsequent search rounds.

We evaluated ADAPT on human-to-mouse and human-to-macaque transfer scenarios using Tabula Sapiens as the reference dataset. Across diverse tissues in Tabula Muris and Macaca fascicularis, ADAPT improved annotation performance over existing cross-species methods, demonstrating the ability of feedback-driven search to identify transferable adaptation strategies.

AI USE STATEMENT

This work proposes an agent-driven framework for cross-species transfer, where generative AI agents are used as part of the proposed method. In addition, generative AI tools were used for LaTeX formatting assistance.

REFERENCES

Jinheon Baek, Sujay Kumar Jauhar, Silviu Cucerzan, and Sung Ju Hwang. Researchagent: Iterative research idea generation over scientific literature with large language models. In *Proceedings of the 2025 conference of the nations of the Americas chapter of the association for computational linguistics: human language technologies (volume 1: long papers)*, pp. 6709–6738, 2025.

Changde Cheng, Wenan Chen, Hongjian Jin, and Xiang Chen. A review of single-cell rna-seq annotation, integration, and cell–cell communication. *Cells*, 12(15):1970, 2023.

The Tabula Sapiens Consortium*, Robert C Jones, Jim Karkanias, Mark A Krasnow, Angela Oliveira Pisco, Stephen R Quake, Julia Salzman, Nir Yosef, Bryan Bulthaup, Phillip Brown, et al. The tabula sapiens: A multiple-organ, single-cell transcriptomic atlas of humans. *Science*, 376(6594): eabl4896, 2022.

Haotian Cui, Chloe Wang, Hassaan Maan, Kuan Pang, Fengning Luo, Nan Duan, and Bo Wang. scgpt: toward building a foundation model for single-cell multi-omics using generative ai. *Nature methods*, 21(8):1470–1480, 2024.

Laleh Haghverdi, Aaron TL Lun, Michael D Morgan, and John C Marioni. Batch effects in single-cell rna-sequencing data are corrected by matching mutual nearest neighbors. *Nature biotechnology*, 36(5):421–427, 2018.

Lei Han, Xiaoyu Wei, Chuanyu Liu, Giacomo Volpe, Zhifeng Wang, Taotao Pan, Yue Yuan, Ying Lei, Yiwei Lai, Carl Ward, et al. Single-cell atlas of a non-human primate reveals new pathogenic mechanisms of covid-19. *BioRxiv*, pp. 2020–04, 2020.

Wenpin Hou and Zhicheng Ji. Assessing gpt-4 for cell type annotation in single-cell rna-seq analysis. *Nature methods*, 21(8):1462–1465, 2024.

Peter Jansen, Oyvind Tafjord, Marissa Radensky, Pao Siangliulue, Tom Hope, Bhavana Dalvi, Bodhisattwa Prasad Majumder, Daniel S Weld, and Peter Clark. Codescientist: End-to-end semi-automated scientific discovery with code-based experimentation. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 13370–13467, 2025.

Aleksandra A Kolodziejczyk, Jong Kyoung Kim, Valentine Svensson, John C Marioni, and Sarah A Teichmann. The technology and biology of single-cell rna sequencing. *Molecular cell*, 58(4): 610–620, 2015.

Xingyan Liu, Qunlun Shen, and Shihua Zhang. Cross-species cell-type assignment from single-cell rna-seq data by a heterogeneous graph neural network. *Genome Research*, 33(1):96–111, 2023.

Anne Loft, Margo P Emont, Ada Weinstock, Adeline Divoux, Adhideb Ghosh, Allon Wagner, Ann V Hertzel, Babukrishna Maniyadath, Bart Deplancke, Boxiang Liu, et al. Towards a consensus atlas of human and mouse adipose tissue at single-cell resolution. *Nature metabolism*, 7(5): 875–894, 2025.

Romain Lopez, Jeffrey Regier, Michael B Cole, Michael I Jordan, and Nir Yosef. Deep generative modeling for single-cell transcriptomics. *Nature methods*, 15(12):1053–1058, 2018.

Chris Lu, Cong Lu, Robert Tjarko Lange, Yutaro Yamada, Shengran Hu, Jakob Foerster, David Ha, and Jeff Clune. Towards end-to-end automation of ai research. *Nature*, 651(8107):914–919, 2026.

Andres M. Bran, Sam Cox, Oliver Schilter, Carlo Baldassari, Andrew D White, and Philippe Schwaller. Augmenting large language models with chemistry tools. *Nature machine intelligence*, 6(5):525–535, 2024.

Yanay Rosen, Maria Brbić, Yusuf Roohani, Kyle Swanson, Ziang Li, and Jure Leskovec. Toward universal cell embeddings: integrating single-cell rna-seq datasets across species with saturn. *Nature methods*, 21(8):1492–1500, 2024.

Nicholas Schaum, Jim Karkanias, Norma F Neff, Andrew P May, Stephen R Quake, Tony Wyss-Coray, Spyros Darmanis, Joshua Batson, Olga Botvinnik, Michelle B Chen, et al. Single-cell transcriptomics of 20 mouse organs creates a tabula muris: The tabula muris consortium. *Nature*, 562(7727):367, 2018.

Maxwell ER Shafer. Cross-species analysis of single-cell transcriptomic data. *Frontiers in cell and developmental biology*, 7:175, 2019.

Yuyao Song, Zhichao Miao, Alvis Brazma, and Irene Papatheodorou. Benchmarking strategies for cross-species integration of single-cell rna sequencing data. *Nature Communications*, 14(1):6495, 2023.

Tim Stuart, Andrew Butler, Paul Hoffman, Christoph Hafemeister, Efthymia Papalexi, William M Mauck, Yuhan Hao, Marlon Stoeckius, Peter Smibert, and Rahul Satija. Comprehensive integration of single-cell data. *cell*, 177(7):1888–1902, 2019.

Alexander J Tarashansky, Jacob M Musser, Margarita Khariton, Pengyang Li, Detlev Arendt, Stephen R Quake, and Bo Wang. Mapping single-cell atlases throughout metazoa unravels cell type evolution. *Elife*, 10:e66747, 2021.

Christina V Theodoris, Ling Xiao, Anant Chopra, Mark D Chaffin, Zeina R Al Sayed, Matthew C Hill, Helene Mantineo, Elizabeth M Brydon, Zexian Zeng, X Shirley Liu, et al. Transfer learning enables predictions in network biology. *Nature*, 618(7965):616–624, 2023.

Anyi Xu, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, Chenchen Ling, et al. Deepseek-v4: Towards highly efficient million-token context intelligence. *arXiv preprint arXiv:2606.19348*, 2026.

Chenling Xu, Romain Lopez, Edouard Mehlman, Jeffrey Regier, Michael I Jordan, and Nir Yosef. Probabilistic harmonization and annotation of single-cell transcriptomics data with deep generative models. *Molecular systems biology*, 17(1):MSB20209620, 2021.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022.

Hao Yuan, Christopher A Mancuso, Kayla Johnson, Ingo Braasch, and Arjun Krishnan. Computational strategies for cross-species knowledge transfer. *Nature methods*, 23(2):312–327, 2026.

Grace XY Zheng, Jessica M Terry, Phillip Belgrader, Paul Ryvkin, Zachary W Bent, Ryan Wilson, Solongo B Ziraldo, Tobias D Wheeler, Geoff P McDermott, Junjie Zhu, et al. Massively parallel digital transcriptional profiling of single cells. *Nature communications*, 8(1):14049, 2017.

Huawen Zhong, Wenkai Han, David Gomez-Cabrero, Jesper Tegner, Xin Gao, Guoxin Cui, and Manuel Aranda. Benchmarking cross-species single-cell rna-seq data integration methods: towards a cell type tree of life. *Nucleic Acids Research*, 53(1):gkae1316, 2025.

A Appendix

A.1 Organ-level sample sizes and class coverage

Tables 3 and 4 summarize the organ-level statistics for the human-to-mouse and human-to-macaque transfer settings, respectively. For each organ, we report the number of shared cell types, total target cell types, and the numbers of source and target cells used in evaluation. These statistics highlight the variation in sample size and class overlap across tissues.

Table 3: Statistics of human-to-mouse cross-species annotation datasets.

Organ	Shared classes	All target classes	Target cells	Source cells
Bladder	4	4	2,500	800
Heart and Aorta	3	4	526	600
Kidney	3	8	2,781	501
Limb Muscle	6	6	3,909	1,049
Liver	4	4	1,845	800
Lung	12	13	5,404	2,203
Mammary Gland	7	7	4,481	1,215
Marrow	12	14	3,652	2,223
Spleen	5	5	9,552	823
Thymus	3	3	1,429	492
Tongue	2	3	7,538	400
Trachea	5	5	11,248	818

Table 4: Statistics of the human-to-macaque cross-species annotation datasets.

Organ	Shared classes	All target classes	Target cells	Source cells
Aorta	3	4	2,260	600
Kidney	2	11	20,647	301
Liver	6	6	7,170	1,200
Lung	6	10	11,468	1,200
PBMC	7	7	8,507	1,370
Parotid	4	7	10,531	709
Pancreas	5	5	6,046	744

A.2 MATHEMATICAL DERIVATIONS

This appendix derives three quantities used by ADAPT. Sec. A.2.1 identifies the term of the target risk that the initial objective does not control. Sec. A.2.2 specializes that term to the covariance and class-conditional surrogates admitted by an objective update. Sec. A.2.3 defines the seed-penalized score used to rank candidates and states what the penalty coefficient does not imply.

A.2.1 TARGET RISK AND THE INITIAL OBJECTIVE

Let g denote the encoder and h a probe applied to its latent, and write $f = h \circ g$. Let $R_S(f)$ and $R_T(f)$ be the expected 0–1 risks of f under the source and target distributions. For a hypothesis class $\mathcal{H}$, the population form of the domain-adaptation bound **?** states that every $f \in \mathcal{H}$ satisfies

$$R_T(f) \le R_S(f) + \frac{1}{2}\, d_{\mathcal{H}\Delta\mathcal{H}}(S,T) + \lambda, \tag{12}$$

where the $\mathcal{H}\Delta\mathcal{H}$-divergence is

$$d_{\mathcal{H}\Delta\mathcal{H}}(S,T) = 2 \sup_{f,f' \in \mathcal{H}} \Big| \Pr_S\big[f(x) \neq f'(x)\big] - \Pr_T\big[f(x) \neq f'(x)\big] \Big| \tag{13}$$

and the joint error of the best hypothesis in the class is

$$\lambda = \inf_{f^\star \in \mathcal{H}} \big[R_S(f^\star) + R_T(f^\star)\big]. \tag{14}$$

The constant λ remains when no hypothesis in $\mathcal{H}$ labels both species well. Adaptation can act on the discrepancy term. It does not remove λ.

The initial objective of Section 3,

$$\mathcal{L}_{\text{init}} = \mathcal{L}_{\text{source-cls}} + 0.1\, \mathcal{L}_{\text{rec}}, \tag{15}$$

does not place a penalty on $d_{\mathcal{H}\Delta\mathcal{H}}$. The cross-entropy is computed on the neural head. It is a convex surrogate of that head's source error, and the head is discarded before evaluation. The scored predictor is a logistic-regression probe fit on the latent, so $\mathcal{L}_{\text{source-cls}}$ is not a direct surrogate of $R_S(h \circ g)$ for the probe that enters equation 12. The reconstruction term penalizes the decoded embedding for departing from the standardized scGPT input on source cells and on unlabeled target-training cells. That penalty keeps information from both embedding distributions in the latent. It is not a discrepancy between the two class-conditional laws. With the training loss fixed at $\mathcal{L}_{\text{init}}$, the second term of equation 12 is unconstrained. An objective update can add a surrogate of that term. The bound is stated for a fixed hypothesis class. It identifies the free term in the initial objective. It does not give a rate for the learned encoder or for the outer search.

A.2.2 DISCREPANCY SURROGATES

An objective update may change the training loss while the parent architecture and optimizer stay fixed. The surrogates below are applied to the 64-dimensional latent $\mathbf{u} = g(\tilde{\mathbf{z}})$. One matches a marginal second moment. The other matches class-conditional means.

Global covariance alignment. Let $U \in \mathbb{R}^{n\times d}$ be a batch of latent vectors and let U_c be the same matrix after subtracting the feature-wise mean. The empirical covariance is

$$C(U) = \frac{1}{n-1} U_c^\top U_c. \tag{16}$$

Correlation alignment **?** penalizes the squared Frobenius distance between the source and target covariances. Up to the constant factor $\frac{1}{4}$, which is absorbed into the loss weight, the penalty used here is the mean squared entry of that residual,

$$\mathcal{L}_{\mathrm{CORAL}} = \frac{1}{d^2}\|C_S - C_T\|_F^2. \tag{17}$$

Equation equation 17 matches the marginal second moment of the two species. Class means are free to move if that movement reduces the covariance gap. A smaller value of $\mathcal{L}_{\mathrm{CORAL}}$ does not imply that class-conditional locations have been preserved.

Class-conditional mean alignment. Target labels are not available during training. Each target cell receives a pseudo-label from a detached argmax,

$$\hat{y} = \arg\max \mathrm{softmax}(\ell^t), \tag{18}$$

and $\hat{y}$ is treated as a constant in the backward pass. For a source class c that has at least two source cells and two pseudo-labeled target cells in the batch, the class means are

$$\mu_S(c) = \frac{1}{n_S(c)} \sum_{y_i^s = c} \mathbf{u}_i^s, \qquad \mu_T(c) = \frac{1}{n_T(c)} \sum_{\hat{y}_j = c} \mathbf{u}_j^t. \tag{19}$$

Let $\mathcal{C}$ be the set of classes that meet this support constraint. The alignment loss is the average squared Euclidean distance between matched means,

$$\mathcal{L}_{\mathrm{align}} = \frac{1}{|\mathcal{C}|} \sum_{c\in\mathcal{C}} \|\mu_S(c) - \mu_T(c)\|_2^2, \tag{20}$$

and equals zero when $\mathcal{C}$ is empty. For the linear kernel $k(\mathbf{u},\mathbf{v}) = \mathbf{u}^\top\mathbf{v}$, the squared maximum mean discrepancy between the two empirical class-conditional measures collapses to the same distance. Expanding the V-statistic gives

$$\begin{aligned} \mathrm{MMD}_c^2 &= \|\mu_S(c)\|_2^2 + \|\mu_T(c)\|_2^2 - 2\,\mu_S(c)^\top \mu_T(c) \\ &= \|\mu_S(c) - \mu_T(c)\|_2^2. \end{aligned} \tag{21}$$

Hence $\mathcal{L}_{\mathrm{align}}$ is the average, over classes with sufficient support, of a within-class linear-kernel MMD. Because the pseudo-label is detached, the partition of the batch is constant while the means are differentiated. A source latent in class c and a target latent assigned to class c then receive

$$\frac{\partial \mathcal{L}_{\mathrm{align}}}{\partial \mathbf{u}_i^s} = \frac{2}{|\mathcal{C}|\, n_S(c)}\big(\mu_S(c) - \mu_T(c)\big), \qquad \frac{\partial \mathcal{L}_{\mathrm{align}}}{\partial \mathbf{u}_j^t} = \frac{2}{|\mathcal{C}|\, n_T(c)}\big(\mu_T(c) - \mu_S(c)\big). \tag{22}$$

The two class means are pulled toward each other. The update on a target cell points at the source mean of its assigned class, not at the mean of all source cells. The latter is the update that matches only the marginal means. Source labels remain an anchor through $\mathcal{L}_{\text{source-cls}}$, so the source mean does not move only to reduce $\mathcal{L}_{\mathrm{align}}$.

Marginal prior. Class-conditional alignment does not control the predicted class proportions. Let π_S be the empirical frequency of source labels in the batch, and let

$$\pi_T = \frac{1}{n_T} \sum_j \mathrm{softmax}(\ell_j^t) \tag{23}$$

be the mean target predictive distribution. The marginal penalty implemented in the candidate is

$$\mathcal{L}_{\mathrm{prior}} = \mathrm{KL}(\pi_S \| \pi_T) = \sum_c \pi_S(c) \log \frac{\pi_S(c)}{\pi_T(c)}, \tag{24}$$

with a floor of 10^{-8} inside $\log \pi_T$. The direction is source against target. If $\pi_T(c) \to 0$ for a class with $\pi_S(c) > 0$, then $\mathcal{L}_{\text{prior}} \to \infty$. The penalty therefore requires the predicted target marginal to cover the source label frequencies. It does not depend on where each class sits in latent space. A batch can decrease $\mathcal{L}_{\text{align}}$ while dropping source-frequent classes from π_T. Equation equation 24 is the term that penalizes that collapse. The two penalties are not substitutes.

The strong conditional candidate uses the fixed combination

$$\mathcal{L} = \mathcal{L}_{\text{source-cls}} + 0.15\,\mathcal{L}_{\text{rec}} + 0.05\,\mathcal{L}_{\text{align}} + 0.01\,\mathcal{L}_{\text{prior}}. \tag{25}$$

Relative to $\mathcal{L}_{\text{init}}$, the reconstruction weight is 0.15 rather than 0.1. These coefficients are one configuration of the objective template. Another configuration $\boldsymbol{\alpha}$ of the same template changes the weights and keeps the functional form in equation 17–equation 24. Equation equation 25 is not claimed to be optimal on every tissue.

A.2.3 Seed-penalized selection score

Let $F(c, s)$ be the shared-class Macro-F1 of candidate c on seed $s \in \{0, 1, 2\}$, evaluated on the partition used for the decision at hand. The selection score uses the sample standard deviation,

$$\bar{F}(c) = \frac{1}{3}\sum_{s=0}^{2} F(c, s), \qquad \sigma^2(c) = \frac{1}{2}\sum_{s=0}^{2}\big(F(c, s) - \bar{F}(c)\big)^2, \tag{26}$$

$$S(c) = \bar{F}(c) - 0.25\,\sigma(c). \tag{27}$$

The same expression is S_{val} on the validation partition and S_{monitor} on the monitor partition. The divisor in $\sigma^2(c)$ is $n - 1 = 2$. Candidates are ranked by larger $S(c)$.

The coefficient 0.25 is not a confidence quantile. A one-sided 95% lower bound from a Student t statistic with $n = 3$ observations and 2 degrees of freedom uses the critical value $t_{2,0.95} = 2.920$ and the standard error $\sigma/\sqrt{3}$:

$$\bar{F}(c) - t_{2,0.95}\,\frac{\sigma(c)}{\sqrt{3}} = \bar{F}(c) - 1.686\,\sigma(c). \tag{28}$$

The factor 0.25 is about one seventh of 1.686. Written on the standard error, the penalty is

$$0.25\,\sigma(c) = \big(0.25\sqrt{3}\big)\,\frac{\sigma(c)}{\sqrt{3}} \approx 0.433\,\frac{\sigma(c)}{\sqrt{3}}, \tag{29}$$

while the one-sided 95% critical value multiplies the same standard error by 2.920. Thus $S(c)$ applies a light penalty to seed-to-seed variation when two candidates have similar mean Macro-F1. It is not a confidence interval for the expected validation score, and it does not upper-bound the risk on the sealed test partition.

A.3 Agent prompts

The nested campaign sends two kinds of messages. The outer planner receives the system prompt in Section A.3.1 and one JSON user message. The inner code planner receives the system prompt in Section A.3.3 and one JSON user message, then returns Python source. Both clients use temperature 0.2. The outer planner is allowed at most 20 attempts and $12{,}000$ completion tokens. The inner planner uses the same model, with at most 5 attempts in the nested entry point. The outer request sets `response_format` to a JSON object. The inner proposal request does not.

The user messages below are pretty-printed for reading. The client sends each object through `canonical_json`, which sorts keys and removes insignificant whitespace. String values are unchanged. Fields marked `RUNTIME` are filled from the campaign state, the organ cache, or the parent source file. Organ cell counts, class histograms, hashes, host name, and the parent `CandidateModule` source are omitted from this listing because they differ by run.

The nested inner loop does not call the diagnosis model. The executor writes the diagnosis object from the frozen round plan and passes it to the proposal prompt. A rejected response is not repaired by editing the original prompt in place. The client appends the rejected output and one of the correction messages in Section A.3.5, then requests the object or source file again from scratch.

A.3.1 OUTER SYSTEM PROMPT

```
You are the auditable outer-loop scientist for cross-species neural
    representation search. Return exactly the requested JSON object,
    never Python or markdown. Propose falsifiable hypotheses and a
    frozen group of controlled experiments. Every generated experiment
    changes one declared mechanism while holding the data split, primary
    metric, training budget and unrelated mechanisms fixed. Include one
    control rerun in every group. The neutral control has hypothesis_id
    null; comparisons connect it to every generated hypothesis
    candidate. Never invent a special undeclared baseline hypothesis ID.
    Same-round experiments are independent siblings whose parents
    already exist in campaign history. Do not use unavailable target
    truth, credentials, file paths, commands or private
    chain-of-thought. The input is a frozen scGPT embedding: candidates
    may transform that embedding but cannot fine-tune scGPT itself.
    Assign every generated candidate exactly one mechanism_family:
    architecture, objective, or optimizer. The executor freezes the
    other CandidateModule methods and rejects source that changes
    multiple families. The outer loop chooses a semantic mechanism; the
    inner loop must use one frozen code template with a top-level
    MATRIX_PARAMETERS dictionary. Across candidates in that matrix, only
    the declared MATRIX_PARAMETERS values may differ.
```

A.3.2 OUTER USER MESSAGES

Every outer call uses the system prompt above. The first round sends a plan request with empty prior evidence. After a completed round, a separate mechanism call is sent only when the exploration policy forces a direction change. The review call then returns either the next plan or a stop decision.

The task context included in every planning payload contains the following fixed strings, together with the runtime organ name, source cell count, source class counts, and label-access status.

```
scientific_goal:
  Learn a downstream representation from frozen scGPT embeddings for
  cross-species classification using a fixed source-only
  LogisticRegression probe.
feature_provider: frozen precomputed scGPT embeddings
candidate_capability:
  CandidateModule can transform the 512-d embedding and define its
  loss/optimizer, but cannot access or fine-tune the scGPT model.
mechanism_isolation:
  Each round tests one hypothesis as a controlled parameter matrix.
  All generated variants share exactly one AST-enforced family and
  either one factor with multiple levels or two factors in a complete
  Cartesian design, with all nuisance factors fixed.
matrix_design_prior:
  Use the historical control plus four variants that span a meaningful
  range, including conservative and stronger settings, so the complete
  round can reveal direction, saturation, and instability.
embedding_dim: 512
n_target_train_scope: frozen random 60% partition
fixed_priors:
  - preserve an explicit identity or residual path from the input
  - global alignment may erase minority or boundary structure
  - low entropy or prediction concentration is not adaptation evidence
  - source labels are available; target labels are unavailable to models
  - the neural classifier head is diagnostic, not the primary selector
monitor_holdout_status: isolated_and_never_exposed_to_planner
post_test_tuning_allowed: false
```

Initial plan and later plan requests. The instruction string is below. N is the frozen experiment count, 5 by default.

```
Return one complete RoundPlan JSON object for exactly one falsifiable
hypothesis and one controlled parameter matrix containing exactly
N total experiments: exactly 1 control and exactly N-1 generated.
The generated candidates must sweep unique exact assignments over
either one shared factor or a complete two-factor Cartesian matrix
within one mechanism_family. The control is included in the total,
not added on top of it.
```

The same message contains `round_plan_contract` and `planning_context`. The contract tells the model to return these fields.

```
schema_version: crossspecies_nested_campaign.v1
campaign_id: RUNTIME
round_index: RUNTIME
planner_context_sha256: RUNTIME
hypotheses:
  - hypothesis_id: path-safe ID
    statement: string
    rationale: string grounded in supplied evidence
    falsifiable_prediction: string
    success_criterion: string
    failure_criterion: string
experiments:
  - experiment_id: path-safe unique ID
    candidate_id: path-safe globally unique ID
    hypothesis_id: declared hypothesis ID for generated candidates;
                   null for the neutral control rerun
    branch_id: path-safe branch ID
    parent_candidate_id: candidate ID from snapshot
    mechanism_to_change: one mechanism, or control rerun
    mechanisms_to_hold_fixed: [at least one string]
    seeds: [0, 1, 2]
    max_epochs: RUNTIME
    tags:
      candidate_role: exactly one control; all others generated
      mechanism_family: control for the control; exactly one of
                        architecture, objective, optimizer for
                        generated candidates
      matrix_axis: control for the control; the same comma-separated
                   set of one factor or two factorial factors for
                   every generated candidate
      matrix_value: historical_parent for the control; a unique exact
                    value or level for every generated candidate
      mechanism_id: control for the control; one shared stable semantic
                    ID for the generated mechanism
      template_id: historical_parent for the control; one new shared
                   round-unique ID for the frozen generated-code
    template
comparisons:
  - comparison_id: path-safe unique ID
    left_candidate_id: generated same-round candidate ID
    right_candidate_id: control same-round candidate ID
    metric: validation_encoder_shared_Macro_f1
    purpose: controlled contrast
branch_budgets:
  - branch_id: one entry per distinct experiment branch_id
    planned_experiments: N
    remaining_after_round: 0
selection:
  primary_metric: validation_encoder_shared_Macro_f1
  direction: maximize
  seed_std_penalty: 0.25
  minimum_successful_seeds: 3
```

```
evaluation:
  search_partition: validation
  audit_partition: null
  run_audit_this_round: false
  audit_used_for_candidate_selection: false
  sealed_test_partition: test
```

The planning context binds this contract to the task context, the compact campaign memory, and the following requirements.

```
experiment_count: N
experiment_count_decomposition:
  total_including_control: N
  control_count: 1
  generated_count: N-1
one_control_rerun: true
same_round_parents_forbidden: true
one_hypothesis_per_round: true
parameter_matrix:
  shared_mechanism_family: true
  shared_matrix_axis: true
  unique_matrix_values: true
  purpose: estimate the response curve and useful range of one idea,
           rather than make unrelated one-off guesses
  axis_semantics: Use either one factor with multiple levels or two
                  factors as a complete Cartesian product; hold all
                  nuisance factors fixed
two_stage_search:
  outer_stage: choose a biologically and technically distinct
               mechanism; respect the deterministic exploration policy
  inner_stage: freeze one code template and vary only MATRIX_PARAMETERS
               over the declared factor matrix
candidate_capability:
  CandidateModule receives frozen 512-d scGPT embeddings. It can learn
  a downstream transform but cannot modify or fine-tune scGPT.
mechanism_family_rule:
  architecture: change only __init__, encode, and/or forward; preserve
                training_step, validation_step, and configure_optimizers
  objective: change only training_step; preserve architecture,
             validation, and optimizer methods
  optimizer: change only configure_optimizers; preserve architecture,
             training_step, and validation_step
  control: copy and rerun the historical parent unchanged
use_complete_group_evidence_only: true
branch_budget_rule:
  For every distinct branch_id, planned_experiments must equal the
  exact count of experiments in this plan with that branch_id.
control_hypothesis_rule:
  The control experiment must set hypothesis_id to null. Every
  generated experiment must reference a declared hypothesis.
parent_selection_guidance:
  mode: soft_prompt_guidance_not_validator_policy
  recommended_parent_candidate_id: RUNTIME
  default_behavior: Use the recommended historical validation incumbent
                    as the shared parent for the control and generated
                    matrix. If the latest round did not improve it,
                    prefer returning to that incumbent.
  exploration_exception: You may choose another eligible historical
                         parent when the supplied validation evidence
                         gives a concrete exploration reason. Explain
                         that reason in the hypothesis rationale.
  allowed_evidence:
    - development_validation_selection_score
    - complete_prior_round_validation_evidence
```

```
  forbidden_evidence: [monitor, sealed_test]
exploration_policy:
  plateau_forces_new_mechanism_and_axis: true
  maximum_consecutive_rounds_per_mechanism: 2
  mechanism_cooldown_rounds: 3
  maximum_consecutive_rounds_per_family: 2
  template_id_must_be_new_each_round: true
  semantic_change_rule: A new mechanism_id must denote a scientifically
                        distinct active computation. Adding a version
                        suffix or renaming the same mechanism does not
                        count as a mechanism change.
  candidate_freedom: Choose any scientifically justified mechanism and
                     parameter axis outside these currently prohibited
                     values.
```

Mechanism contract. When the previous round triggers a forced direction change, the outer model first receives the following instruction, and must then fill the contract underneath it.

```
Select one scientifically distinct mechanism before writing the full
experiment matrix. Return only mechanism_contract. Self-check that its
family, mechanism ID and parameter axis obey every simultaneous-change
and prohibited-value rule. Preserve freedom outside those exact
exclusions.
```

```
mechanism_family: architecture|objective|optimizer
mechanism_id: stable semantic identifier
matrix_axis: one factor or two comma-separated factors
scientific_difference:
  concise explanation of the distinct active computation and why it is
  not a rename of a recent mechanism
```

The accepted contract is then copied into the next plan request as `fixed_mechanism_contract`, with the binding instruction

```
The next_plan must exactly use fixed_mechanism_contract's family,
mechanism_id and matrix_axis. Every generated experiment must copy its
exact design_spec_sha256 and scientific_difference as the
design_spec_summary tag. The control must omit both tags.
```

Review and next plan. After every experiment in the round has a terminal result, the outer model receives

```
In one response, review the hypothesis using the complete frozen
parameter matrix, summarize the response curve and useful range, and
either return the next complete RoundPlan or stop. If a next plan is
returned, it must contain exactly N total experiments: exactly 1
control and exactly N-1 generated; the control is included in the
total. Return exactly one JSON object whose only top-level key is
decision_contract. Do not echo instruction, must_stop, validator
feedback, context, or schema descriptions outside decision_contract.
```

The required decision object is

```
decision_contract:
  review:
    reviewed_round_index: RUNTIME
    result_sha256: RUNTIME
    hypothesis_verdicts:
      RUNTIME_HYPOTHESIS_ID: supported|rejected|inconclusive
    lessons: [concise evidence-grounded string]
    branch_actions:
```

```
      RUNTIME_BRANCH_ID: continue|pause|stop|replace
  next_plan: the RoundPlan contract, or null on the final round
  stop_reason: non-empty string on the final round, otherwise null
must_stop: true only when the frozen round budget is exhausted
```

The completed round result is attached as `complete_prior_evidence`. Monitor and test metrics are not part of this object. On the final round the controller discards any proposed next plan and keeps the review.

A.3.3 INNER SYSTEM PROMPT

```
You are the model-design stage of CrossSpecies AutoResearch v3. Design
    one
PyTorch Lightning CandidateModule for cross-species representation
    learning.
Return Python source only, without markdown fences or explanatory prose.

Maximize scientific creativity inside the model-authoring boundary.
    There is no
prescribed topology, latent width, depth, residual design,
    normalization, attention
mechanism, differentiable objective, optimizer hyperparameter, or
    scheduler unless
the current controlled experiment explicitly holds that component
    fixed. You may
define import-safe helper classes, helper functions, literal constants,
    and
CandidateModule helper methods when they support the declared mechanism.

The fixed executor owns data loading, normalization, source/validation
    splitting,
Trainer settings, early stopping, deterministic seeds, fixed LogReg
    probing,
artifact writing, W&B, and target-truth isolation. These are
    infrastructure
boundaries, not restrictions on model capacity. Candidate code owns the
    neural
architecture, forward computation, differentiable losses, and optimizer.

The source file is imported by a sandbox. At module scope, emit
    definitions only:
imports from the allowlist, docstrings, classes, functions, immutable
    literal
constants, and the exact required MATRIX_PARAMETERS literal when
    supplied. Never
instantiate a model, allocate data, start training, call Trainer,
    mutate global
framework state, read or write files, inspect the environment, make
    network calls,
or place executable tests at module scope.

Keep CandidateModule, its constructor arguments, required method
    signatures,
forward output keys, and validation metric compatible with the supplied
    contract.
When a controlled experiment supplies mutable and frozen core-method
    lists, freely
redesign the mutable methods and add supporting definitions, but copy
    every frozen
core method exactly from the supplied parent. This preserves causal
    attribution;
it does not prescribe the implementation of the active research
    mechanism.
```

```
Treat source labels as training annotations. Treat target_x as strictly
    unlabeled.
Never infer that the target population is uniform. Prefer biologically
    plausible
preservation of cell identity under species/domain shift. A low-entropy
    prediction
is not evidence of adaptation and class concentration is a failure mode.

Use the validated history diagnosis. Preserve successful mechanisms,
    explicitly
repair diagnosed failures, and make a falsifiable change relative to
    the parent.
Do not evade duplicate detection with prose, names, or seed-only
    changes.
```

A.3.4 INNER USER MESSAGE

The user message is one JSON object with five fields:

```
instruction
candidate_authoring_boundary
information_packet
validated_history_diagnosis
required_design_metadata_as_class_docstring
```

Its instruction is

```
Produce exactly one complete candidate source file. The implementation
must satisfy the fixed contract, log val/source_loss, use target_x
without annotations, and implement the diagnosed next action. First
internally check the authoring-boundary checklist; do not print the
checklist or reasoning.
```

The authoring boundary sent with that instruction is

```
principle:
  Scientific freedom is broad; only the infrastructure interface and
  the current controlled-comparison boundary are fixed.
scientific_freedom:
  - freely design topology, width, depth, residual paths and attention
  - freely add import-safe helper classes, functions and module methods
  - freely design differentiable losses and optimization within the
    active controlled mechanism
fixed_infrastructure:
  - CandidateModule name, constructor and required method signatures
  - forward keys and tensor shapes
  - data split, seeds, evaluation, logging and target-label firewall
  - safe import with no module-level execution or external side effects
current_controlled_experiment:
  primary_mechanism_family: RUNTIME from the RoundPlan
  mutable_core_methods:
    architecture: __init__, encode, forward
    objective: training_step
    optimizer: configure_optimizers
  copy_exactly_from_parent: every other core method, with its parent
                            source included verbatim
  matrix_axis: RUNTIME
  exact_matrix_assignment: RUNTIME
  required_matrix_statement_copy_verbatim:
    MATRIX_PARAMETERS = {inherited parameters plus this assignment}
  matrix_instruction:
    Define one top-level literal string-to-string dictionary named
    MATRIX_PARAMETERS containing every inherited frozen parameter plus
    the declared active axis and assigned value. CandidateModule must
```

```
    continue consuming inherited parameters and must consume the active
    parameter. Every candidate in this round uses the same frozen code
    template: only active-axis values in MATRIX_PARAMETERS may differ.
    Do not substitute another assignment or modify code outside that
    dictionary.
preflight_checklist:
  - the module is safe to import and contains definitions only
  - all fixed interfaces and exact output keys are preserved
  - every frozen core method is byte-for-byte equivalent to the parent
    AST
  - the required unannotated MATRIX_PARAMETERS statement is copied
    verbatim and consumed by CandidateModule
  - all new trainable parameters receive gradients
  - target_x is used without target annotations
  - only Python source is returned, with no markdown or prose
response_format: Python source only
```

The first generated candidate in a round is sampled from this prompt. The remaining matrix entries are not new model calls. The executor substitutes the assigned `MATRIX_PARAMETERS` values into that frozen source.

The information packet included in the same user message fixes the task text below. Its numeric priors, class order, baseline prediction histogram, and incumbent source are runtime fields.

```
scientific_goal:
  Learn a cross-species representation that preserves source cell
  identity and improves unlabeled-target transfer beyond scGPT.
source_domain: human source cells with cell-type annotations
target_domain: mouse target cells with embeddings only
evaluation_views:
  - encoder representation with fixed source-only LogisticRegression
  - joint neural classifier head
success_definition:
  - retain source OOF Macro-F1
  - improve label-free adaptation proxy without concentration
  - improve development-validation target Macro-F1
  - remain stable across seeds before any sealed evaluation
train_and_evaluation_separation:
  - fit neural parameters with source annotations and target embeddings
  - select candidates with source and label-free target feedback
  - use only the random development-validation partition for feedback
  - keep the development-test partition out of prompts until freeze
  - never use sealed-organ annotations before atomic blind evaluation
label_firewall:
  candidate_batch_keys: [source_x, source_y, target_x]
  target_x_semantics: unlabeled embeddings only
  candidate_annotation_access: source annotations only
  planner_annotation_access: source annotations only
  development_feedback_policy:
    aggregate validation metrics may guide search and calibrate proxy;
    test metrics never enter search
scientific_priors:
  - scGPT coordinates already encode substantial cell identity;
    use residual or bottleneck transformations and protect the original
    representation from destructive drift
  - species shift mixes technical, orthology, and biological effects;
    align shared structure conditionally; global alignment can merge
    distinct or minority cell types
  - organ cell-type frequencies are naturally imbalanced; do not
    optimize toward uniform predictions
  - reconstruction alone may preserve nuisance variation; combine
    identity-preserving source supervision with a controlled unlabeled
    adaptation mechanism
  - target-only cell types may be absent from the source axis; avoid
```

```
    forcing every target cell into a confident source class
fixed_candidate_contract:
  class_name: CandidateModule
  constructor: CandidateModule(input_dim: int, n_classes: int)
  required_methods: [encode, forward, training_step, validation_step,
                     configure_optimizers]
  forward_exact_keys: [latent, reconstruction, logits]
  allowed_import_roots: [pytorch_lightning, torch, typing]
  allowed_optimizers: [Adam, AdamW, SGD]
  maximum_parameters: 20000000
  maximum_source_bytes: 65536
  validation_metric_name: val/source_loss
fixed_executor:
  normalization: source-training-split feature mean and standard
    deviation
  early_stopping: monitor val/source_loss, mode min, patience 5,
                  min_delta 0.0001
  representation_probe: source-only LogisticRegression, C=1.0,
                        max_iter=3000, seed=0, schema-3 frozen source
                        OOF assignments
reference_minimal_baseline:
  architecture: shared MLP 512->128->64, symmetric decoder, linear head
  loss: source cross-entropy + 0.1 * mean source/target reconstruction
  optimizer: AdamW(lr=0.0003, weight_decay=0.00001)
design_requirements:
  - keep positive source classification supervision
  - use at least one target_x-dependent unlabeled mechanism
  - protect minority and boundary structure from global over-alignment
  - include anti-concentration logic before entropy or pseudo-label
    losses
  - prefer one interpretable mechanism change per exploit iteration
```

The diagnosis object attached to this packet is written by the executor:

```
candidate_assessments:
  - candidate_index: 0
    observation: Use the supplied historical parent as the control.
    mechanism_assessment: inconclusive
    safety_assessment: The fixed executor and contract remain unchanged.
next_action:
  mode: explore
  parent_candidate_index: 0
  mechanism_to_change: RUNTIME from the RoundPlan
  mechanisms_to_preserve: RUNTIME mechanisms held fixed
  falsifiable_hypothesis: RUNTIME falsifiable_prediction
  expected_feedback_changes:
    validation_encoder_shared_Macro_f1: increase
  failure_criterion: RUNTIME failure_criterion
```

The class docstring of the returned module must record the parent, the hypothesis, the mechanism changed from the parent, the expected feedback, and the failure criterion.

A.3.5 VALIDATOR CORRECTION MESSAGES

If the outer response is not a complete JSON object, the next user message adds

```
The previous response could not be decoded. Return the complete
requested JSON object again from scratch, not a fragment or patch.
Keep it concise and omit markdown.
```

If the plan fails validation, the correction instruction is

```
Satisfy every validator error in the accumulated history, not only the
```

```
latest one. Return a complete corrected object and do not omit
unchanged required fields. The experiments array must contain exactly
N total entries: exactly 1 control plus exactly N-1 generated. Use
exactly one hypothesis; all generated entries must share one
mechanism_family and matrix_axis and must have unique matrix_value
tags. Each matrix_value must contain only the raw value (or
comma-separated raw values), never axis=value syntax. Generated
entries must also share one mechanism_id and one new template_id.
Read fixed_exploration_policy_remediation and satisfy every listed
simultaneous change and prohibited-value constraint. A version suffix
or rename of the same computation is not a new scientific mechanism.
```

A rejected mechanism contract is retried with

```
Return a fresh mechanism_contract from scratch. Do not repeat any
mechanism_id, matrix_axis, mechanism_family, or computation named in
the prohibited values or accumulated feedback. A rename or version
suffix is not a scientific change.
```

A rejected review uses the plan correction above and adds that the only top-level key must be `decision_contract`. An inner source file that fails the contract is retried with the failed program and

```
The previous source failed the import/interface preflight, not the
scientific hypothesis: RUNTIME_VALIDATOR_ERROR. Preserve its scientific
idea while repairing only the violated authoring boundary.
```

The same message then appends one of three repairs: copy the required `MATRIX_PARAMETERS` statement verbatim; copy every frozen method from the supplied parent source; or move runtime construction out of module scope. It ends with

```
Re-run the supplied preflight checklist internally. Return one complete
corrected Python source file only, without markdown.
```