Sub-Quadratic Task Construction for Unsupervised Meta-Learning: Approximate kNN Graphs and OT Assignments in the Inner Loop

Liz Lemma Future Detective

January 20, 2026

Abstract

Recent work (e.g., DHM-UHT) argues that meta-learning is particularly robust to label noise and task heterogeneity, and leverages this by placing unsupervised task construction (clustering-based pseudo-labeling) inside the inner loop. A key obstacle to scaling this idea to 2026-era backbones (ViTs/VLMs) is computational: density-based clustering such as DBSCAN becomes the dominant cost in the inner loop for high-dimensional embeddings and large batch sizes. We formulate inner-loop task construction as a primitive that maps a batch of embeddings to a variable-cardinality pseudo-partition with outlier handling, and propose two scalable families of constructors: (i) approximate kNN graph construction followed by fast community detection / label propagation, and (ii) Sinkhorn-based optimal-transport (OT) soft assignments with straight-through hardening. We provide (a) time/space bounds showing near-linear per-task complexity, (b) recovery guarantees under a planted-partition noisy-embedding model, and (c) unconditional lower bounds showing that exact DBSCAN-style ε-neighborhood connectivity requires quadratic work on worst-case inputs. The resulting UHT meta-learning pipelines retain heterogeneity and noise robustness while making "meta-learn task construction" viable at foundation-model scale. (Implementations and scaling experiments would strengthen the contribution.)

Table of Contents

3. 3. Problem Formulation: formalize meta-training with a task constructor g; define desiderata (heterogeneity, outliers, stability, compute).

4. 4. Scalable Task Constructors: (A) g_{Graph} using ANN kNN graphs + community detection; (B) g_{OT} using Sinkhorn soft partitions; unified interface and head reinitialization.

5. 5. Theoretical Model: planted partition in embedding space with noise/drift; formal definitions of recovery and stability under perturbations.

6. 6. Main Guarantees: recovery theorems for g_{Graph} and g_{OT}; perturbation bounds; conditions under which cluster count is correctly estimated (up to outliers).

7. 7. Complexity and Lower Bounds: upper bounds for ANN+community detection and Sinkhorn; unconditional lower bound for exact DBSCAN neighborhood graph computation; implications for bilevel training cost.

8. 8. Experimental Protocol (recommended): apples-to-apples replacement of DBSCAN in DHM-UHT; scaling curves in (m, d); accuracy vs time; ablations for k, ANN error, and outlier thresholds.

9. 9. Discussion and Extensions: streaming/online variants, multi-source pseudo-labeling compatibility, and how constructors interact with stability/robustness diagnostics.

10. 10. Limitations and Open Questions: non-differentiability, approximation bias, worst-case failures; directions for tighter end-to-end meta-learning guarantees.

1 Introduction

Unsupervised heterogeneous task construction (UHT) has recently become a convenient mechanism for turning large unlabeled corpora into a stream of meta-learning tasks: given a minibatch T of size m, one embeds the examples by a shared body f_θ and then clusters the embeddings to obtain pseudo-labels that define a task-specific classification head. A representative instance is DHM-UHT, whose inner loop runs a density-based routine (typically DBSCAN or a close variant) on the current embeddings and then adapts a freshly initialized head (and sometimes part of θ) by cross-entropy on the resulting pseudo-labels. In this design, task construction is not a preprocessing step but an inner-loop primitive; hence its computational profile, stability properties, and failure modes directly affect the bilevel optimization.

The present work is motivated by an increasingly common regime in which DHM-UHT becomes compute-bound not by representation learning but by the repeated execution of DBSCAN-like neighborhood queries. Contemporary backbones produce embeddings of dimension $d \geq 512$ (often substantially larger), and meta-training commonly uses large per-task batch sizes $m \geq 512$ to stabilize the pseudo-labels and to amortize the forward pass. In such regimes, exact DBSCAN requires, explicitly or implicitly, access to an ε-neighborhood graph: for each point one must identify all neighbors within radius ε to determine core points and density connectivity. On worst-case inputs this graph is dense, and even writing down the adjacency can require $\Theta(m^2)$ edges. Consequently, any exact implementation inherits a quadratic-time worst-case per task, and in practice one observes an additional constant-factor penalty in high-dimensional distance computations and memory traffic. Since the constructor is invoked for every task in every outer iteration, a quadratic per-task subroutine quickly dominates wall-clock time, limiting m, limiting the number of tasks per meta-batch, and thereby constraining statistical efficiency.

Beyond cost, the density-based constructor interacts poorly with the bilevel loop when embeddings drift. Meta-training continuously perturbs θ, and thus the embeddings $Z = \{f_\theta(x_i)\}_{i=1}^m$ move from iteration to iteration. Density-based connectivity can change discontinuously under small perturbations: a single edge crossing the radius threshold may merge components, split components, or change the set of core points. Such instabilities are not merely aesthetic; they induce high-variance gradients and brittle inner-loop targets, especially when the constructor is applied on the same minibatch that is used for adaptation. Accordingly, we seek a task constructor that (i) runs in near-linear time per task, (ii) preserves the defining UHT features used by DHM-UHT (variable number of clusters, explicit outliers), and (iii) is stable under small embedding drift so that pseudo-labels do not change adversarially as θ is updated.

We propose two scalable constructors, g_{Graph} and g_{OT}, intended as drop-in replacements for DBSCAN inside UHT-style bilevel training. The graph-based constructor g_{Graph} forms a bounded-degree approximate kNN graph on the embeddings using an approximate nearest neighbor (ANN) index and then applies a community-detection or label-propagation procedure to obtain a partition into communities and an outlier set. The bounded-degree constraint (each node stores $O(k)$ neighbors) yields $|E| = \Theta(mk)$ edges and thereby enforces a near-linear per-task time and space budget for fixed k. Outliers are handled by a minimum community size threshold τ: small communities are designated as outliers and are excluded from the inner-loop cross-entropy loss, preventing noise amplification by forcing a head to fit idiosyncratic micro-clusters. The second constructor g_{OT} replaces hard graph connectivity with a soft assignment to a small set of anchors (prototypes) via entropically regularized optimal transport (Sinkhorn scaling). One then hardens assignments by an argmax-with-confidence rule; low-confidence points become outliers, and the effective number of clusters K is determined by the number of anchors that receive nontrivial mass. This yields a variable-cardinality head dimension while providing a tunable stability–fidelity trade-off through the regularization and threshold parameters.

Our contributions are organized around the computational, statistical, and bilevel-compatibility requirements imposed by UHT.

- *Scalable task construction.* We exhibit constructors whose per-task cost is $\tilde{O}(mk)$ for g_{Graph} (assuming standard ANN primitives) and near-linear in m for fixed anchor count and Sinkhorn iterations for g_{OT}. This removes the quadratic bottleneck of exact ε-neighborhood connectivity.

- *Unconditional gap for exact DBSCAN primitives.* We formalize the output-size obstruction: constructing the exact ε-neighborhood graph, the core primitive underlying DBSCAN connectivity, requires $\Omega(m^2)$ time in the worst case. Hence any approach that insists on exact DBSCAN behavior cannot asymptotically match the per-task budgets required by modern meta-training.

- *Recovery guarantees under separation.* Under a planted-partition/noisy-embedding model with separation parameter Δ and noise level σ, we provide conditions under which each constructor recovers the ground-truth partition on non-outlier points with misclustering rate at most ε, up to polylogarithmic factors in m. For g_{Graph} this follows from within-cluster connectivity and between-cluster sparsity in the approximate kNN graph; for g_{OT} it follows from concentration of Sinkhorn mass on correct anchors.

- *Stability under embedding drift.* We articulate stability in the sense relevant to bilevel training: small perturbations of Z should not cause large perturbations of the soft assignment matrix (for g_{OT}) nor rampant relabeling due to threshold effects. In particular, the Sinkhorn-based mapping admits a Lipschitz-type control under bounded perturbations, enabling smoother evolution of pseudo-labels across outer iterations.

- *Meta-compatible inner-loop integration.* Both constructors return a variable K and an explicit outlier set, enabling the inner loop to reinitialize a task-specific head of dimension K and to compute CE only on confidently assigned points. This mirrors the operational semantics of DHM-UHT while permitting substantially larger m and higher-dimensional d within the same compute envelope.

The net effect is a principled replacement of a quadratic-time, discontinuity-prone inner-loop primitive by bounded-degree graph and transport-based alternatives that are tailored to the bilevel setting. In the remainder we position these constructors within standard meta-learning formalism, clarify the role of variable cluster count and outliers in UHT, and then develop the runtime bounds, recovery analyses, and stability properties that justify their use in large-scale meta-training.

2 Background: bilevel meta-learning and UHT as inner-loop task construction

We briefly recall the bilevel viewpoint underlying gradient-based meta-learning and then explain how unsupervised heterogeneous task construction (UHT) fits into this formalism when tasks and labels are both induced from unlabeled minibatches. Our emphasis is on two features that are structurally necessary in the unsupervised setting: a variable number of classes per task and an explicit mechanism for outliers.

Bilevel meta-learning (MAML/ANIL viewpoint). In standard few-shot meta-learning, one assumes a task distribution $\mathcal{T}$ where each task T provides labeled data, typically split into a support set S and a query set Q. A shared representation (or "body") f_θ is trained so that, after a small number of inner-loop gradient steps on S, a task-specific predictor achieves low loss on Q. Abstractly, if h denotes task-specific parameters (e.g., a linear head) and Adapt denotes s steps of gradient descent on an inner loss L_{inner}, the meta-objective takes the form

$$\min_{\theta} \; \mathbb{E}_{T\sim\mathcal{T}}\Big[L_{\text{outer}}\big(\mathrm{Adapt}(h, f_\theta; T),\, T\big)\Big],$$

where L_{outer} is evaluated after adaptation, either on a query set or on a held-out subset of the same data. In MAML, the inner loop may update both h and θ; in ANIL-style variants, the inner loop updates only h while f_θ is updated only by the outer loop. For our purposes, the key property is that the outer-loop gradient flows through an *adaptation map* that depends on the data and on f_θ, so any high-variance or discontinuous dependence of the inner-loop targets on θ can destabilize training.

From labeled tasks to unlabeled minibatches. UHT replaces externally provided labeled tasks by tasks sampled as unlabeled minibatches from a dataset D. Concretely, a task T is a set of m examples $x_1, \dots, x_m \in \mathcal{X}$ sampled from D, and we form embeddings

$$z_i = f_\theta(x_i) \in \mathbb{R}^d, \qquad Z = \{z_i\}_{i=1}^m.$$

Since T arrives without labels, one introduces a *task constructor* g that maps the embedding set Z to pseudo-labels and hence to a pseudo-partition. We write

$$g(Z) \;\rightsquigarrow\; \hat{\Pi} = (C_1, \dots, C_K, O), \qquad \hat{y}_i \in \{1, \dots, K\} \cup \{\perp\},$$

where $C_1, \dots, C_K$ are discovered clusters, O is an outlier set, and $\perp$ denotes "unassigned/outlier." The inner loop then trains a freshly initialized head h (and optionally part of θ) to predict $\hat{y}_i$ from z_i. The resulting bilevel structure is thus a composition of (i) representation f_θ, (ii) discrete (or soft) pseudo-labeling via g, and (iii) adaptation via gradient descent on a cross-entropy loss defined by $\hat{y}$.

Why the constructor must live in the inner loop. In UHT, it is tempting to regard clustering as a preprocessing step. However, because f_θ changes throughout meta-training, the embedding geometry changes, and the pseudo-labels must be recomputed to remain aligned with the evolving representation. Thus g is invoked per task, per outer iteration, and its output is part of the inner-loop supervision signal. Consequently, the constructor is not an auxiliary heuristic but an algorithmic primitive whose computational cost and stability properties directly determine the feasibility of scaling the meta-training loop.

Variable number of classes and the dynamic head. A distinctive feature of unsupervised task construction is that the number of discovered clusters K is neither known a priori nor constant across tasks. Even if the underlying dataset has a fixed set of semantic categories, a minibatch may contain an unknown subset (or multiple modes within a category), and the constructor may intentionally return fewer or more clusters depending on the local geometry in Z. Therefore the task-specific head must be *dynamic*: for each task we instantiate h with output dimension K determined by $g(Z)$, and we train it on the non-outlier points. This heterogeneity is not an inconvenience but a necessary degree of freedom: forcing a fixed K across tasks either (i) induces spurious class splits/merges that inject label noise into the inner loss, or (ii) encourages representation collapse by rewarding trivial partitions that match the fixed head dimension rather than the intrinsic batch structure.

Outliers as a first-class object. Unlike supervised tasks, where every example arrives with a label, an unlabeled minibatch inevitably contains points that are ambiguous under the current representation: rare modes, boundary points between clusters, and points whose nearest neighbors are inconsistent due to noise. If such points are assigned arbitrary pseudo-labels, the resulting cross-entropy gradients can dominate the inner loop, encouraging h to fit unstable artifacts and encouraging f_θ to contort the embedding space to satisfy brittle assignments. We therefore require g to return an explicit outlier set O and we define the inner-loop loss only on assigned points:

$$L_{\text{inner}}(h;\, Z, \hat{y}) \;:=\; \frac{1}{|\{i : \hat{y}_i \neq \perp\}|} \sum_{i:\hat{y}_i \neq \perp} \mathrm{CE}\big(h(z_i), \hat{y}_i\big).$$

This exclusion mechanism is a robustness device: it allows the constructor to abstain when the geometry does not support a confident partition, preventing error amplification in the bilevel loop.

Interaction with outer-loop optimization and stability. Because pseudo-labels are computed from Z, they depend implicitly on θ. In practice one often stops gradients through the discrete output of g (treating $\hat{y}$ as constant within an outer step), yielding a tractable optimization but placing a premium on *stability*: small changes in Z induced by incremental updates of θ should not cause large changes in $\hat{\Pi}$. If $\hat{\Pi}$ fluctuates sharply, then the effective learning target of the inner loop changes erratically across outer iterations, increasing gradient variance and causing intermittent failure modes (e.g., sudden changes in K, mass outliering, or cluster merges/splits). Stability is therefore not merely a clustering desideratum; it is a meta-optimization requirement tied to the smoothness of the composed map $\theta \mapsto Z \mapsto \hat{\Pi} \mapsto \mathrm{Adapt}(\cdot)$.

Summary. In UHT, we must treat task construction as an inner-loop component of a bilevel system. This forces explicit handling of (i) heterogeneous tasks with variable K and hence dynamic heads, and (ii) outliers to control pseudo-label noise. These considerations motivate constructors that are computationally light enough to be invoked repeatedly and stable enough to define usable inner-loop supervision as f_θ evolves. In the next section we formalize this setting and state the computational and statistical desiderata that the constructor g must satisfy.

3 Problem formulation: meta-training with an inner-loop task constructor

We now formalize the unsupervised heterogeneous task construction (UHT) setting as a bilevel optimization problem in which the supervision signal is produced *within* the inner loop by a task constructor. The purpose of this section is to make explicit (i) the objects computed per task, (ii) the interface required by the meta-learner when the number of classes is task-dependent, and (iii) the algorithmic desiderata—heterogeneity, outlier handling, stability, and computational scalability—that guide the design of g.

Tasks and embeddings. Let D be an unlabeled dataset over $\mathcal{X}$, and let $\mathcal{T}$ denote the induced task distribution obtained by sampling minibatches. A task $T \sim \mathcal{T}$ is a multiset $T = \{x_i\}_{i=1}^m$ of size m. Given shared parameters θ, we form task embeddings

$$z_i \;=\; f_\theta(x_i) \in \mathbb{R}^d, \qquad Z \;=\; \{z_i\}_{i=1}^m.$$

The representation f_θ is the only component shared across tasks; all task-specific objects are re-initialized each time T is sampled.

Constructor output as a pseudo-partition with abstentions. A task constructor is a (typically non-differentiable) map

$$g:\; (\mathbb{R}^d)^m \to \big(\{1,\dots,m\} \cup \{\perp\}\big)^m$$

that assigns each embedding either a pseudo-label in $\{1,\dots,K\}$ or an outlier symbol $\perp$. Equivalently, $g(Z)$ induces a pseudo-partition

$$\hat{\Pi} \;=\; (C_1,\dots,C_K,O), \qquad O = \{i : \hat{y}_i = \perp\}, \quad C_\ell = \{i : \hat{y}_i = \ell\},$$

where $K = K(Z)$ is *discovered* and may vary across tasks. The outlier set O is not an artifact but part of the output specification: it is the mechanism by which g can decline to label ambiguous points under the current embedding geometry.

Dynamic heads and the inner loss. Given $\hat{\Pi}$, we instantiate a task-specific head h whose output dimension matches K. Concretely, one may take $h : \mathbb{R}^d \to \mathbb{R}^K$ to be a linear classifier (or shallow MLP) initialized anew for each task. We define the inner-loop loss using cross-entropy on *non-outlier* indices:

$$L_{\text{inner}}(h;\, Z, \hat{y}) \;:=\; \frac{1}{m-|O|}\sum_{i\notin O}\text{CE}\big(h(z_i), \hat{y}_i\big),$$

with the convention that if $m-|O|$ is too small (e.g. below a minimum size τ), the task can be skipped or downweighted. The adaptation operator Adapt performs s steps of gradient descent on L_{inner}, updating h and optionally a subset of θ (ANIL corresponds to updating only h). The outer loss L_{outer} is evaluated after adaptation, either on a support/query split of T or on the same batch with appropriate regularization:

$$\min_{\theta}\; \mathbb{E}_{T\sim\mathcal{T}}\Big[L_{\text{outer}}\big(\text{Adapt}(h, f_\theta, g(Z));\, T\big)\Big], \qquad Z = \{f_\theta(x) : x \in T\}.$$

In this formulation, the constructor g is an algorithmic component of the learning system: it is invoked repeatedly during meta-training and defines the targets used by the inner-loop optimizer.

Desiderata for g. We state the requirements that a constructor must satisfy to be viable at meta-training scale.

(D1) Heterogeneity (variable K). We require that K be task-dependent and that g return a *consistent labeling* of the assigned points, i.e. each $i \notin O$ receives exactly one label in $\{1, \dots, K\}$. This requirement rules out formulations that hard-code a global class count or force every minibatch into a fixed number of clusters. At the interface level, heterogeneity means that $g(Z)$ must provide both labels and a canonical reindexing of the discovered clusters so that the head dimension is well-defined.

(D2) Outlier handling (abstention). We require an explicit outlier set O so that uncertain points do not contribute to L_{inner}. From the bilevel perspective, the role of abstention is to prevent high-variance or adversarial pseudo-labels from dominating gradients. Operationally, g should mark as outliers (i) points not belonging to any sufficiently large community, (ii) points with low assignment confidence, and (iii) points whose local neighborhood evidence is inconsistent. We emphasize that outliering is not merely for evaluation convenience; it is a control knob on the noise level of the inner-loop supervision.

(D3) Stability under embedding drift. Because Z depends on θ, small outer updates change Z across iterations. Since we typically stop gradients through the discrete pseudo-labels, we require that g be *stable* in the following sense: if Z and Z' satisfy $\max_i \|z_i - z'_i\| \le \eta$ for small η, then the induced pseudo-partitions should differ on only a small fraction of indices (up to label permutation) or, in a soft-assignment variant, the assignment matrix should vary in a Lipschitz manner. Formally, for a suitable distance $\mathrm{dist}(\cdot,\cdot)$ on partitions modulo permutation, we seek bounds of the form

$$\mathrm{dist}\big(g(Z), g(Z')\big) \;\le\; c\,\eta \quad \text{or} \quad \mathrm{dist}\big(g(Z), g(Z')\big) \;\le\; \varepsilon(\eta),$$

with $\varepsilon(\eta) \to 0$ as $\eta \to 0$, away from true geometric degeneracies (e.g. vanishing inter-cluster margins). This condition is a meta-optimization requirement: it reduces outer-loop gradient variance by preventing frequent discontinuous changes in K, massive relabeling, or oscillatory merge/split behavior.

(D4) Per-task computational scalability. A constructor is called for every sampled task, so its cost must be near-linear in m under a fixed neighborhood parameter k. We therefore impose a target per-task complexity of $\tilde{O}(mk)$ time and $O(mk)$ memory (in addition to the embedding forward pass). In particular, we rule out any design that requires forming all $\binom{m}{2}$ pairwise distances or an exact ε-neighborhood graph in high dimension. This constraint is not cosmetic: when m is in the hundreds or thousands, a quadratic primitive can dominate the meta-training wall-clock time even when the backbone is large.

Meta-compatibility. Finally, we require that g be usable within SGD-based meta-training. Concretely, the constructor must be expressible as a deterministic (or controlled-randomness) procedure on Z that returns pseudo-labels and a corresponding head dimension, and whose output can be treated as fixed within an outer step. Optionally, one may allow gradients through a *soft* constructor (e.g. an assignment matrix) while still using a hardening-and-outliering step for defining K and excluding ambiguous points. The central point is that g must compose with Adapt without introducing pathological discontinuities or prohibitive cost.

Conclusion. The above formulation reduces the design of UHT to the construction of a map g that outputs a variable-cardinality pseudo-partition with abstentions, is stable under small perturbations of Z, and runs in near-linear per-task time. In the next section we instantiate two such constructors: a bounded-degree graph-based method and an entropically regularized optimal-transport method, both equipped with a unified interface supporting dynamic heads.

4 Scalable task constructors

We now instantiate two families of constructors satisfying the interface and desiderata of the preceding section while avoiding quadratic primitives. Both constructors take as input the embeddings $Z = \{z_i\}_{i=1}^m$ of a task and return a pseudo-partition with abstentions $\hat{\Pi} = (C_1, \ldots, C_K, O)$, where K is discovered from Z and may vary across tasks. In each case, we additionally specify a canonical reindexing rule so that the induced labels $\hat{y}_i \in \{1, \dots, K\} \cup \{\perp\}$ are stable under benign symmetries (e.g. permutation of samples) and so that head initialization is well-defined.

4.1 Graph-based constructor $g_{\textbf{Graph}}$: ANN kNN graph + community detection

The guiding principle is to replace dense ε-neighborhood connectivity with a bounded-degree proximity graph whose size is $\Theta(mk)$. Given Z, we build an approximate kNN graph using an ANN data structure (e.g. HNSW-style indexing, IVF-PQ, or a GPU-friendly approximate search), producing for each node i a list $\mathcal{N}_k(i)$ of k candidate neighbors. We then form an undirected graph $G = (V, E)$ with $V = \{1, \dots, m\}$ and

$$E \;=\; \big\{\{i,j\} : j \in \mathcal{N}_k(i) \text{ or } i \in \mathcal{N}_k(j)\big\}, \qquad |E| = \Theta(mk).$$

Optionally, to reduce spurious bridges in high dimension we may use a mutual-kNN rule $\{i,j\} \in E$ iff $j \in \mathcal{N}_k(i)$ and $i \in \mathcal{N}_k(j)$, at the cost of a sparser graph. We attach weights

$$w_{ij} \;=\; \exp\!\Big(-\frac{\|z_i - z_j\|^2}{\tau_w^2}\Big) \;\text{ or }\; w_{ij} = 1,$$

and we run a near-linear community detection routine to obtain connected communities that serve as pseudo-classes. Suitable choices include label propagation, Leiden/Louvain heuristics on weighted modularity, or simple pruning plus connected components; our framework does not require optimization of an explicit clustering objective, only a partition that is stable and approximately respects local neighborhoods.

Outlier handling is implemented by a minimum community size threshold $\tau \geq 2$: after community detection yields candidate components $\widetilde{C}_1, \dots, \widetilde{C}_{\widetilde{K}}$, we set

$$C_\ell \;=\; \widetilde{C}_{\pi(\ell)} \;\text{ for those } \widetilde{C}_j \text{ with } |\widetilde{C}_j| \geq \tau, \qquad O \;=\; V \setminus \bigcup_{\ell=1}^{K} C_\ell,$$

where π is a deterministic ordering of retained communities. A convenient canonical rule is to sort by decreasing size and break ties by the smallest index in the community; this makes the label set invariant to permutations of Z up to the inherent symmetry of equal-sized components. The resulting pseudo-labels are then $\hat{y}_i = \ell$ if $i \in C_\ell$ and $\hat{y}_i = \perp$ if $i \in O$.

The computational profile is immediate: ANN construction plus kNN querying yields $\widetilde{O}(mk)$ expected time for fixed k and $O(mk)$ adjacency storage, after which most community heuristics run in time near-linear in $|E|$. Importantly, this constructor never enumerates all pairs, so it remains viable for m in the hundreds or thousands. Finally, stability is mediated by locality: away from geometric degeneracies, small perturbations of Z change only a small fraction of neighbor relations, and hence only a small fraction of edges in G, which empirically yields fewer catastrophic relabelings than density thresholds that hinge on a single global radius.

4.2 OT-based constructor $g_{\mathbf{OT}}$: Sinkhorn soft partitions + hardening with abstention

Our second constructor replaces hard graph connectivity with a soft assignment of points to a small set of anchors. Fix an anchor budget $A \ll m$. Given Z, we choose anchors $\{a_1, \dots, a_A\} \subseteq Z$ (e.g. uniform subsampling, farthest-point sampling on a coarse sketch, or a memory-bank based prototype set). We compute similarities

$$S_{ia} \;=\; \exp\!\Big(-\frac{\|z_i - a_a\|^2}{\tau^2}\Big) \in (0,1], \qquad i \in [m],\; a \in [A],$$

and we form an entropically regularized transport plan $P \in \mathbb{R}_+^{m\times A}$ by approximately projecting S onto a set of matrices with prescribed marginals, using I Sinkhorn iterations. Concretely, for row marginal $r = \frac{1}{m}\mathbf{1}_m$ and column marginal $c = \frac{1}{A}\mathbf{1}_A$ (or a relaxed column marginal to allow unused anchors), Sinkhorn updates compute scaling vectors $u \in \mathbb{R}_+^m$, $v \in \mathbb{R}_+^A$ such that

$$P \;=\; \operatorname{diag}(u)\, S \operatorname{diag}(v)$$

approximately satisfies $P\mathbf{1}_A = r$ and $P^\top \mathbf{1}_m = c$. We then harden the assignment by

$$\hat{a}(i) \;=\; \arg\max_{a\in[A]} P_{ia}, \qquad \operatorname{conf}(i) \;=\; \max_a P_{ia}.$$

A confidence threshold $\beta \in (0,1)$ induces abstention: if $\operatorname{conf}(i) < \beta$, we set $\hat{y}_i = \perp$. Otherwise, i is assigned to the anchor $\hat{a}(i)$. Anchors that receive fewer than τ assigned points are discarded, and their assigned points are reclassified as outliers (or, alternatively, merged into the nearest retained anchor). After pruning, the number of surviving anchors is $K \le A$, and we reindex them canonically (again by decreasing cluster size with deterministic tie-breaking) to obtain labels in $\{1, \dots, K\}$.

This constructor is near-linear in m for fixed (A, I): computing S costs $O(mdA)$, Sinkhorn scaling costs $O(ImA)$, and the memory footprint is $O(mA)$ (which can be reduced by streaming or blockwise Sinkhorn when A is moderate). In contrast to graph partitioning, the OT construction yields an explicit soft assignment P whose variation under small perturbations of Z can be controlled by the regularization, which is advantageous for meta-optimization; moreover, one may optionally backpropagate through P while still using hardening only to define K and O.

4.3 Unified interface and dynamic heads

Both g_{Graph} and g_{OT} expose the same output: $(\hat{y}, K, O)$, with $\hat{y}_i \in \{1, \dots, K\} \cup \{\perp\}$. Given this, we instantiate a task-specific head $h : \mathbb{R}^d \to \mathbb{R}^K$ reinitialized per task. In practice we adopt a fixed initialization distribution (e.g. Gaussian with prescribed variance) and fix all constructor randomness per task (anchor sampling seed, ANN seed) to reduce outer-loop variance. We then perform inner-loop adaptation on the cross-entropy restricted to $\{i : \hat{y}_i \neq \perp\}$, and we skip or downweight tasks for which K is too small or $m - |O|$ falls below τ. The net effect is a constructor layer that is (i) heterogeneous by construction, (ii) explicit about abstention, (iii) compatible with SGD-based bilevel training, and (iv) scalable to large m without invoking an $\Omega(m^2)$ neighborhood primitive.

4.4 Theoretical model: planted partitions, noise, and drift

We formalize the statistical and perturbation regimes under which a task constructor g is expected to output a meaningful pseudo-partition and under which its output varies stably as θ (and hence f_θ) changes across outer-loop iterations. The model is intentionally minimal: it imposes separation in embedding space, allows heterogeneous tasks with variable K, and isolates a small outlier set on which we do not demand correctness.

Task generation and ground-truth partition. A task $T \sim \mathcal{T}$ is a multiset of m examples drawn from D. Conditional on T, we observe embeddings $Z = \{z_i\}_{i=1}^m$ with $z_i = f_\theta(x_i) \in \mathbb{R}^d$. We posit that, for a given task, there exists an (unobserved) ground-truth partition

$$\Pi^* = (S_1, \dots, S_{K^*}, O^*), \qquad [m] = \Big(\bigsqcup_{k=1}^{K^*} S_k \Big) \sqcup O^*,$$

where S_k are "inlier" clusters and O^* is an outlier set (e.g. background points, small spurious modes, or points near cluster boundaries). We allow K^* to be task-dependent and random under $\mathcal{T}$. To align with the constructor interface, we fix a minimum resolvable cluster size $\tau \geq 2$ and only require recovery of the *large* clusters

$$\mathcal{K}_\tau := \{k \in [K^*] : |S_k| \geq \tau\}, \qquad K^*_\tau := |\mathcal{K}_\tau|.$$

Clusters smaller than τ may be treated as outliers without penalty, since any method that discards small communities necessarily cannot guarantee their identification.

Planted-partition geometry in embedding space. Our main assumption is geometric separation with bounded within-cluster diameter. Specifically, for each task there exist radii $r_{\mathrm{in}}, r_{\mathrm{out}}$ (task-dependent but controlled in distribution) such that

$$\|z_i - z_j\| \le r_{\mathrm{in}} \quad \text{for all } i, j \in S_k \text{ and all } k \in [K^*], \tag{1}$$

$$\|z_i - z_j\| \ge r_{\mathrm{out}} \quad \text{for all } i \in S_k,\ j \in S_\ell,\ k \neq \ell, \tag{2}$$

and we define the separation margin $\Delta := r_{\mathrm{out}} - r_{\mathrm{in}} > 0$. This ball-separation condition is a convenient abstraction of more general mixture models; it is sufficient to reason about neighborhood graphs and soft assignments without committing to a particular generative distribution. Outliers are unconstrained except that they may violate (1)–(2); in particular, O^* may contain points arbitrarily close to inlier clusters.

It is often useful to refine (1)–(2) into a "noise around prototypes" model. Namely, there exist (unobserved) cluster centers $\{\mu_k\}_{k=1}^{K^*}$ such that for $i \in S_k$,

$$z_i \;=\; \mu_k + \xi_i, \qquad \|\xi_i\| \le \sigma,$$

and $\|\mu_k - \mu_\ell\| \ge \Delta_0$ for $k \neq \ell$. Then (1)–(2) holds with $r_{\mathrm{in}} \lesssim 2\sigma$ and $r_{\mathrm{out}} \gtrsim \Delta_0 - 2\sigma$, and the effective separation parameter is $\Delta \approx \Delta_0 - 4\sigma$. In this reading, σ quantifies embedding noise induced by both data variability and imperfect representation learning.

Embedding drift and perturbation model. Because the embeddings depend on θ, they change during meta-training, and we require the constructor to be stable to such changes. We model drift as a per-task perturbation of the embedding multiset: let $Z = \{z_i\}_{i=1}^m$ and $Z' = \{z_i'\}_{i=1}^m$ be embeddings of the *same* task examples under two nearby representations (e.g. successive outer-loop iterates). We assume a uniform perturbation bound

$$\max_{i \in [m]} \|z_i - z_i'\| \;\le\; \eta,$$

or, more generally, a high-probability bound under a stochastic perturbation model. The parameter η is interpreted as the embedding drift magnitude. The purpose of the model is not to predict η from optimization dynamics, but to express stability properties as explicit functions of η and separation.

Recovery metric (labels up to permutation, with abstentions). A constructor returns $\hat{\Pi} = (C_1, \ldots, C_K, O)$ and induced labels $\hat{y}_i \in \{1, \ldots, K\} \cup \{\perp\}$. Since cluster labels are only defined up to permutation, we compare $\hat{\Pi}$ to Π^* by minimizing over relabelings. Let $\mathcal{I} := \bigcup_{k \in \mathcal{K}_\tau} S_k$ denote the inlier points in large clusters. For a labeling $\hat{y}$ and a permutation π on $[K]$, define the misclustering rate on $\mathcal{I}$ by

$$\mathrm{err}(\hat{y}; \Pi^*) \;:=\; \min_{\pi} \frac{1}{|\mathcal{I}|} \left| \{ i \in \mathcal{I} :\ \hat{y}_i = \perp \text{ or } \pi(\hat{y}_i) \neq c^*(i) \} \right|,$$

where $c^*(i)$ is the (unique) index with $i \in S_{c^*(i)}$. Thus, abstaining on a true inlier counts as an error, whereas behavior on O^* is not scored. We say $\hat{\Pi}$ achieves *ε-recovery* (at size threshold τ) if $\mathrm{err}(\hat{y}; \Pi^*) \leq \varepsilon$. This definition matches the intended use in bilevel training: the inner-loop loss is computed only on non-abstained points, but we still require that, on most inliers, pseudo-labels coincide with ground truth up to permutation.

Stability metric (bounded change under perturbations). We measure stability of the constructor by comparing outputs on (Z, Z'). Let $\hat{y} = g(Z)$ and $\hat{y}' = g(Z')$ be the resulting labelings with abstentions, each canonically reindexed. We define the disagreement rate on large-cluster inliers by

$$\mathrm{stab}(\hat{y}, \hat{y}'; \Pi^*) \;:=\; \min_{\pi} \frac{1}{|\mathcal{I}|} \left|\{i \in \mathcal{I} :\; \hat{y}_i = \perp \text{ or } \hat{y}'_i = \perp \text{ or } \pi(\hat{y}_i) \neq \hat{y}'_i\}\right|.$$

We call g *(η, α)-stable* (on the task distribution) if, whenever $\max_i \|z_i - z'_i\| \leq \eta$, we have $\mathrm{stab}(\hat{y}, \hat{y}'; \Pi^*) \leq \alpha(\eta)$ with high probability over $T \sim \mathcal{T}$ and any internal randomness of g. In words: small embedding drift should not cause a large fraction of inlier points to flip pseudo-labels or fall into abstention.

Correct estimation of the number of clusters. Finally, since K is data-dependent, we require that the constructor estimate the number of *recoverable* clusters. We say that *g estimates K correctly up to outliers* if, with high probability,

$$K \;=\; K^*_\tau,$$

and moreover each recovered cluster C_ℓ corresponds (up to permutation) to one of the large true clusters S_k with at most ε relative contamination from other inlier clusters. This notion aligns with the operational constraint that the task head has output dimension K and should not be inflated by small spurious components.

The subsequent guarantees are stated in terms of these definitions: separation Δ and noise level σ control recovery, while perturbation magnitude η controls stability, both subject to the minimum size threshold τ that delineates clusters we intend to learn from versus those we intentionally abstain on.

4.5 Main guarantees: recovery, stability, and correct cluster count

We now state guarantees for the two scalable constructors g_{Graph} and g_{OT} under the separation/noise/drift model introduced above. Throughout, correctness is measured by the misclustering rate $\mathrm{err}(\hat{y}; \Pi^*)$ on the large-cluster inliers $\mathcal{I}$, and stability by $\mathrm{stab}(\hat{y}, \hat{y}'; \Pi^*)$ under perturbations of magnitude at most η.

Guarantees for $g_{\mathbf{Graph}}$ (ANN kNN + community detection). Fix a task and form an (approximate) kNN graph $G = (V, E)$ on Z, with $|V| = m$ and out-degree k, and let the community detection routine return communities $C_1, \ldots, C_K$ along with an outlier set O obtained by discarding communities of size $< \tau$. We emphasize that the specific community detection method is not critical, provided it satisfies a standard *cluster-preservation* property: if the input graph is a disjoint union of well-connected components plus a small number of inter-component edges, then the routine returns communities aligned with those components, up to a vanishing fraction of errors. This property holds for common label-propagation and modularity heuristics on graphs with strong within-cluster expansion and sparse cuts.

Theorem 4.1 (Recovery for g_{Graph} under separation). *Assume that for the large clusters S_k with $k \in \mathcal{K}_\tau$, the within-cluster neighborhoods are sufficiently dense in the sense that, for each such k, the exact kNN subgraph induced by S_k is connected and has expansion bounded below by a constant (equivalently, it contains a connected core once $k \geq c \log m$). Assume further that separation dominates noise so that, for all $i \in \mathcal{I}$, every true within-cluster neighbor of i is strictly closer than every cross-cluster inlier point by a margin proportional to Δ (e.g. $\Delta \geq c_0\sigma$ for an absolute $c_0 > 0$ in the prototype-plus-noise specialization). Let ANN return a $(1+\gamma)$-approximate kNN list with failure probability at most δ. Then, with probability at least $1 - \delta - \exp(-\Omega(k))$, the labeling $\hat{y} = g_{Graph}(Z)$ satisfies*

$$\operatorname{err}(\hat{y}; \Pi^*) \;\leq\; \varepsilon, \qquad \textit{with } \; \varepsilon = O(\exp(-c\,k))$$

for some constant $c > 0$, and the returned number of clusters obeys $K = K^*_\tau$.

The conclusion $K = K^*_\tau$ should be interpreted as "correct up to outliers": any true cluster of size $< \tau$ may be absorbed into O (or merged into another small component) without violating the guarantee, since we explicitly exclude such clusters from $\mathcal{K}_\tau$. In particular, the theorem rules out spurious inflation of K by small noisy components inside $\mathcal{I}$.

Perturbation stability for $g_{\mathbf{Graph}}$. We next state a stability statement which formalizes the requirement that small embedding drift should not induce widespread relabeling. The key observation is that, under a margin condition, the kNN identity of each inlier point is locally invariant: if the gap between the kth nearest inlier neighbor (within the true cluster) and the nearest cross-cluster inlier exceeds 2η, then any perturbation of magnitude η preserves the set of true inlier neighbors up to permutation, hence preserves the graph structure relevant for community recovery.

Theorem 4.2 (Stability of g_{Graph} under drift). *Let Z and Z' be embeddings of the same task satisfying $\max_i \|z_i - z'_i\| \leq \eta$. Suppose that for all $i \in \mathcal{I}$*

there exists a margin $M_i > 0$ such that

$$\max_{j \in S_{c^*(i)}} \|z_i - z_j\| \; + \; M_i \; \le \; \min_{\ell \neq c^*(i)} \; \min_{j \in S_\ell} \|z_i - z_j\|,$$

and that $M_i \geq 4\eta$ for all but an ε fraction of points in $\mathcal{I}$. Then, for k above the same connectivity threshold as in Theorem 4.1, the outputs $\hat{y} = g_{Graph}(Z)$ and $\hat{y}' = g_{Graph}(Z')$ satisfy

$$\mathrm{stab}(\hat{y}, \hat{y}'; \Pi^*) \; \le \; \varepsilon \; + \; \varepsilon_{\mathrm{ANN}},$$

where $\varepsilon_{\mathrm{ANN}}$ is the probability that ANN fails on either Z or Z' (typically $\varepsilon_{\mathrm{ANN}} \le 2\delta$).

The theorem makes explicit that instability is concentrated near points with small neighborhood margins (intuitively, boundary points), which are precisely the points our outlier mechanism is designed to abstain on (either because they form small, unstable communities or because they lie in low-density regions of the kNN graph).

Guarantees for g_{OT} (anchors + Sinkhorn + hardening). The OT constructor replaces discrete neighborhood connectivity by a soft assignment of points to a set of A anchors. We assume a sampling condition ensuring anchor coverage of the large clusters and a separation-to-temperature condition ensuring that the correct anchor receives dominant mass for most inlier points. Let $P \in \mathbb{R}^{m \times A}$ denote the (approximately) doubly-stochastic Sinkhorn projection of the similarity matrix S, and let hard labels be obtained by $\hat{y}_i = \arg\max_a P_{ia}$ if $\max_a P_{ia} \ge \beta$, otherwise $\hat{y}_i = \perp$. Let K be the number of anchors actually selected by at least one non-outlier point (thus K is data-dependent).

Theorem 4.3 (Recovery and correct K for g_{OT})**.** *Assume that anchors are sampled so that each large cluster S_k with $k \in \mathcal{K}_\tau$ contains at least one anchor with probability at least $1 - \delta$ (e.g. $A \gtrsim K^*_\tau \log(K^*_\tau/\delta)$ for uniform sampling when clusters are not too imbalanced). Assume further that the separation parameter satisfies $\Delta/\tau_{\mathrm{temp}} \ge c_1$ where τ_{temp} is the similarity bandwidth used in $S_{ia} = \exp(-\|z_i - a\|^2/\tau^2_{\mathrm{temp}})$, and choose the hardening threshold β below the typical correct-anchor mass but above the mass of incorrect anchors (a constant gap regime). Then, with probability at least $1 - \delta$ over anchor sampling,*

$$\mathrm{err}(\hat{y}; \Pi^*) \; \le \; \varepsilon, \qquad \textit{and} \qquad K \; = \; K^*_\tau,$$

where ε decreases as the separation margin and the Sinkhorn approximation accuracy increase.

Perturbation stability for $g_{\mathbf{OT}}$. Unlike the graph constructor, the OT pipeline admits a direct Lipschitz-type perturbation statement because the Sinkhorn operator is stable to multiplicative perturbations of the kernel matrix, and the kernel entries are smooth functions of the embeddings.

Theorem 4.4 (Lipschitz stability of OT assignments). *Let Z and Z' satisfy $\max_i \|z_i - z_i'\| \leq \eta$, and let P and P' be the corresponding Sinkhorn assignment matrices (with the same anchors, or anchors perturbed by at most η as well). Then there exists a constant L depending on τ_{temp} and the Sinkhorn regularization such that*

$$\|P - P'\|_1 \ \leq \ L\,\eta.$$

Moreover, if the row-wise assignment gaps satisfy $P_{i,a^*(i)} - \max_{a \neq a^*(i)} P_{ia} \geq 2\kappa$ for all but an ε fraction of $i \in \mathcal{I}$, and if $L\eta \leq \kappa$, then the hardened labels satisfy $\mathrm{stab}(\hat{y}, \hat{y}'; \Pi^*) \leq \varepsilon$ (up to anchor relabeling).

Consequences for bilevel meta-learning. The preceding statements isolate two principles that we use operationally: (i) correctness is guaranteed on the large-cluster inliers $\mathcal{I}$ under separation and coverage assumptions, while ambiguous points may be safely abstained; and (ii) stability is controlled by explicit margins (nearest-neighbor gaps for g_{Graph} and assignment gaps for g_{OT}), yielding small disagreement under small drift. Having established statistical correctness and perturbation robustness, we next quantify the computational cost of these constructors and contrast it with unconditional lower bounds for exact DBSCAN-style neighborhood connectivity.

4.6 Complexity and lower bounds: scalable constructors versus exact density connectivity

We quantify the per-task and per-iteration costs of the proposed constructors and contrast them with an unconditional lower bound for exact DBSCAN-style neighborhood connectivity. Our goal is to isolate the regime in which task construction is no longer the bottleneck of bilevel meta-training, while preserving the UHT requirements of heterogeneous K and principled abstention.

Upper bound for $g_{\mathbf{Graph}}$ (ANN kNN + community detection). Fix a task with embeddings $Z = \{z_i\}_{i=1}^m \subset \mathbb{R}^d$. The dominant primitive is the construction of a bounded-degree similarity graph on m nodes. Under the standard assumption that an ANN data structure supports building a (directed) $(1+\gamma)$-approximate kNN adjacency list in expected time $\tilde{O}(mk)$ and space $O(mk)$, we obtain a sparse graph with $|E| = \Theta(mk)$. Any subsequent graph routine whose work is linear (or near-linear) in the number of edges—e.g. a fixed number of label-propagation passes, greedy modularity updates, or local refinement—therefore runs in $\tilde{O}(|E|) = \tilde{O}(mk)$ time. This yields the per-task bound summarized in Theorem 1:

$$T_{\text{Graph}}(m) \;=\; \tilde{O}(mk) \;+\; T_{\text{comm}}(mk), \qquad S_{\text{Graph}}(m) \;=\; O(mk).$$

Two remarks are operationally important. First, the outlier mechanism (discard communities of size $< \tau$) is asymptotically free: it is a single pass over the community sizes and node labels, hence $O(m)$. Second, because we treat the partitioning step as discrete, we do not backpropagate through the community detection decisions; thus the gradient cost of the constructor is negligible compared to the forward pass to compute Z and the inner-loop updates of (h, θ).

Upper bound for $g_{\mathbf{OT}}$ (anchors + Sinkhorn). The OT-based constructor replaces graph construction by computing similarities to a set of A anchors. Given anchors $\{a_\ell\}_{\ell=1}^{A} \subset \mathbb{R}^d$ chosen from (or derived from) Z, forming the kernel matrix

$$S_{i\ell} = \exp\bigl(-\|z_i - a_\ell\|^2/\tau_{\text{temp}}^2\bigr)$$

costs $O(mdA)$ arithmetic operations, which is near-linear in m for fixed (A, d). Sinkhorn scaling then iteratively rescales rows and columns to obtain an approximately doubly-stochastic matrix P, with each iteration requiring $O(mA)$ time. For I iterations we obtain

$$T_{\text{OT}}(m) \;=\; O(mdA + I\,mA), \qquad S_{\text{OT}}(m) \;=\; O(mA),$$

as in Theorem 1. In memory-constrained settings we may stream the Sinkhorn updates (maintaining row/column scaling vectors and recomputing kernel blocks), reducing peak memory below $O(mA)$ at the expense of a modest constant-factor increase in time. As above, hardening and outlier rejection (thresholding by β) are $O(mA)$ operations and do not change the asymptotic bound.

Unconditional lower bound for exact DBSCAN neighborhood connectivity. We now formalize the sense in which exact DBSCAN-style task construction is intrinsically incompatible with the near-linear per-task budget when worst-case inputs are allowed. Let G_ε denote the exact ε-neighborhood graph on m points: $(i, j) \in E_\varepsilon$ iff $\|z_i - z_j\| \leq \varepsilon$. Exact DBSCAN requires, at minimum, the ability to determine for each point its ε-neighbors (or to compute equivalent core-point connectivity), which amounts to constructing G_ε or an object of comparable output size. Theorem 4 states the following output-size lower bound: there exist instances for which $|E_\varepsilon| = \Theta(m^2)$ (e.g. all pairwise distances are $\leq \varepsilon$), and therefore any algorithm that outputs G_ε must spend $\Omega(m^2)$ time simply to write down the edges. This lower bound is unconditional: it does not depend on dimension d, on any hardness conjecture, or on limitations of nearest-neighbor data structures; it is a direct consequence of potentially dense output.

This observation explains the practical scaling gap between exact DBSCAN-UHT and bounded-degree alternatives. Even if one accelerates radius queries via spatial indexing, the worst-case cost cannot be subquadratic because the output itself can be quadratic. In contrast, both g_{Graph} and g_{OT} explicitly cap per-node connectivity (by k or A), ensuring that the produced intermediate objects are of size $\tilde{O}(m)$.

Implications for bilevel meta-training cost. Let C_f denote the amortized cost of one forward+backward pass through f_θ per example, and let s denote the number of inner-loop gradient steps. A single task then incurs embedding and inner-loop cost on the order of

$$T_{\text{embed+inner}}(m) \;\approx\; O(mC_f) \;+\; O(s\,mC_f),$$

up to constants depending on whether we adapt only h or also a subset of θ. Let $C_g(m)$ denote the constructor cost. In an outer iteration with a meta-batch of B tasks, the total wall-clock cost is approximately

$$T_{\text{outer}} \;\approx\; B\Big(T_{\text{embed+inner}}(m) \;+\; C_g(m)\Big).$$

Theorem 5 makes precise a threshold phenomenon: if $C_g(m) = \Omega(m^2)$ (exact DBSCAN-like) while $C_f = \tilde{O}(1)$ per example, then for sufficiently large m the constructor dominates the entire meta-step. This is precisely the undesirable regime in which increasing task size to improve representation learning yields diminishing returns because the clustering step saturates the compute budget. Conversely, if $C_g(m) = \tilde{O}(mk)$ (Graph) or $C_g(m) = \tilde{O}(mA)$ (OT with fixed A, I), then for modern backbones with large C_f the constructor becomes a lower-order term, so scaling in m and d is limited primarily by the embedding network rather than by task construction.

Finally, we note that the above bounds are compatible with the stability requirements of bilevel training. By constraining intermediate structures to be sparse and by abstaining on ambiguous points (small communities or low-confidence assignments), we avoid the combinatorial instability associated with dense neighborhood graphs, while retaining sufficient signal for inner-loop adaptation. In the next section we specify an experimental protocol that isolates these computational effects by replacing DBSCAN in DHM-UHT with g_{Graph} or g_{OT} while controlling all other components.

4.7 Experimental protocol: apples-to-apples replacement of DBSCAN in DHM-UHT

We specify an experimental protocol whose sole purpose is to isolate the computational and statistical consequences of the task constructor. Concretely, we treat DHM-UHT with DBSCAN as the reference implementation and replace only the constructor by either g_{Graph} (ANN kNN + community detection) or g_{OT} (anchors + Sinkhorn), keeping all remaining meta-learning components fixed.

Controlled replacement and invariants. In all experiments we keep fixed: (i) the unlabeled dataset D and task sampler $\mathcal{T}$ (minibatches of size m), (ii) the backbone f_θ architecture and initialization, (iii) the inner-loop adaptation operator $\text{Adapt}(\cdot)$ (number of steps s, optimizer, learning rates, and whether θ is adapted or frozen), (iv) the outer-loop optimizer and schedule, and (v) the head parameterization (a linear classifier with output dimension K reinitialized per task). The only altered module is the mapping $Z \mapsto \hat{\Pi}$ and the induced pseudo-labels $\hat{y}_i \in \{1, \ldots, K\} \cup \{\perp\}$. Outliers ($\hat{y}_i = \perp$) are excluded from the inner-loop cross-entropy in all methods, so that differences are attributable to the constructor rather than to loss reweighting.

Datasets and downstream evaluation. We meta-train on standard unlabeled image corpora (e.g. MiniImageNet, CIFAR-FS, tieredImageNet, or a chosen ImageNet subset with labels hidden during meta-training) and evaluate the learned representation by few-shot classification on disjoint labeled classes. For few-shot evaluation we use episodic N-way S-shot tasks (e.g. $N \in \{5, 20\}$ and $S \in \{1, 5\}$), reporting mean accuracy and 95% confidence intervals over a fixed set of evaluation episodes. To decouple representation quality from any particular episodic solver, we additionally report a linear probe and a kNN classifier on frozen embeddings. The intent is not to maximize absolute accuracy but to compare constructors under identical training compute.

Time and memory measurement methodology. We measure three wall-clock components per task: (a) embedding time T_f for computing $Z = \{f_\theta(x_i)\}_{i=1}^m$, (b) constructor time T_g for producing $\hat{\Pi}$, and (c) inner-loop time T_{inner} for s gradient steps on non-outliers. We instrument GPU/CPU boundaries with CUDA events and synchronize before timing each block; data loading is excluded by prefetching. We then report

$$T_{\text{task}} \;=\; T_f + T_g + T_{\text{inner}}, \qquad \text{speedup}(g) \;=\; \frac{T_g(\text{DBSCAN})}{T_g(g)}.$$

We also report peak memory (GPU and host) attributable to the constructor, together with the retained fraction

$$\alpha \;:=\; \frac{1}{m}\big|\{i : \hat{y}_i \neq \perp\}\big|,$$

since aggressive abstention can artificially reduce inner-loop cost while degrading supervision quality.

Scaling curves in (m, d). To study scaling we vary the task size m over a geometric grid (e.g. $m \in \{64, 128, 256, 512, 1024, 2048\}$) and the embedding dimension d over a grid determined by architecture choice (e.g. ResNet widths) and/or an explicit projection head (e.g. $d \in \{128, 256, 512, 1024, 2048\}$). For each (m, d) pair we run a fixed number of outer iterations (or, alternatively, a fixed wall-clock budget) and record: (i) final few-shot accuracy, (ii) T_g and T_{task}, and (iii) stability statistics defined below. We emphasize accuracy–time trade-offs by plotting Pareto frontiers (accuracy versus constructor time) at matched training budgets; this prevents conflating speed improvements with having trained for longer.

Constructor-specific hyperparameter ablations. For g_{Graph} we ablate the neighbor parameter k (e.g. $k \in \{8, 16, 32, 64\}$) and the minimum community size τ used for outlier rejection. We additionally ablate the community detection routine (label propagation versus a greedy modularity heuristic) while keeping the same input graph. For g_{OT} we ablate the number of anchors A, the number of Sinkhorn iterations I, and the hardening/outlier threshold (e.g. declare $\perp$ when $\max_\ell P_{i\ell} < \beta$). In each case we report the induced K distribution across tasks, α, and downstream accuracy; this is essential because two constructors may achieve comparable accuracy with very different pseudo-task cardinalities.

ANN error sensitivity. To quantify the effect of approximate neighbor search, we treat the ANN index parameters as a dial that controls recall at fixed k (e.g. HNSW parameters such as `efSearch` and `M`). We estimate recall by sampling a subset of nodes and comparing ANN neighbors to exact neighbors computed on that subset, yielding an empirical recall $\widehat{\text{rec}}$. We then report accuracy and stability as a function of $\widehat{\text{rec}}$, thereby separating the effect of graph sparsity (controlled by k) from the effect of neighbor errors.

Stability and robustness diagnostics. Because bilevel training can be destabilized by rapidly changing pseudo-labels, we measure partition stability across outer iterations. For a fixed task sampled repeatedly with controlled randomness, we compute the normalized variation of information (NVI) or adjusted Rand index (ARI) between successive pseudo-partitions after matching cluster labels by permutation, restricting to non-outliers. We also measure the sensitivity of $\widehat{\Pi}$ to small embedding perturbations by injecting Gaussian noise $z_i \mapsto z_i + \xi_i$ with $\xi_i \sim \mathcal{N}(0, \sigma^2 I)$ and recording the change in assignments and in α. These diagnostics allow us to detect regimes where speed is obtained at the cost of highly discontinuous pseudo-supervision.

Reporting and statistical practice. All results are averaged over multiple random seeds (at least 3) and reported with dispersion (standard deviation across seeds for training, confidence intervals across evaluation episodes). We fix the meta-training budget either by outer iterations or by wall-clock time; both are reported, since constructor acceleration may alter the number of effective parameter updates achievable within a given time.

The outcome of this protocol is an explicit accuracy–time–stability trade-off map as (m, d) vary, together with ablations that identify which constructor knobs control this trade-off. In the subsequent discussion we use these empirical regularities to motivate streaming variants, multi-source pseudo-labeling, and principled robustness monitoring.

4.8 Discussion and extensions: streaming/online variants, multi-source pseudo-labeling, and stability diagnostics

We discuss three directions that naturally follow from the proposed scalable constructors: (i) streaming or online task construction when tasks are drawn sequentially and embeddings evolve during meta-training, (ii) compatibility with multi-source pseudo-labeling schemes that combine several weak partitioners, and (iii) explicit interactions between the constructor and stability/robustness diagnostics, including the possibility of feeding diagnostics back into training.

Streaming and online task construction. The outer loop induces a slowly drifting embedding map f_θ, hence the geometry of each task cloud $Z = \{f_\theta(x_i)\}_{i=1}^m$ changes over time. If tasks arrive as a stream, it is wasteful to rebuild neighbor indices from scratch whenever the same or similar points reappear (e.g. under reservoir sampling or repeated augmentations). A direct extension is to maintain a global memory bank $\mathcal{M}$ of recent embeddings together with an ANN index that is updated asynchronously. At outer iteration t, for a new task T we compute Z_t, insert Z_t into $\mathcal{M}$, and query the index to obtain approximate neighbors either within-task (restricted queries) or cross-task (full queries). The latter yields a bipartite (or augmented) graph in which each task node has edges to a bounded number of memory-bank nodes. We may then run a community routine on the induced subgraph on $V_T \cup V_{\mathcal{M},T}$ and subsequently restrict the resulting labeling to V_T. This construction preserves bounded degree (hence near-linear time) while allowing information reuse across tasks; it also amortizes index construction, replacing per-task indexing by incremental insert/delete operations.

A similar streaming idea applies to g_{OT}. Instead of sampling anchors anew from each task, we maintain a set of persistent prototypes $\{a_\ell\}_{\ell=1}^A$

stored in $\mathcal{M}$ and updated online (e.g. by exponential moving averages of assigned embeddings). For a task T we compute similarities $S_{i\ell} = \exp(-\|z_i - a_\ell\|^2/\tau^2)$ and run a fixed small number of Sinkhorn iterations to obtain a soft assignment matrix P. Hardening via $\hat{y}_i = \arg\max_\ell P_{i\ell}$ with abstention yields pseudo-labels, while the prototypes are updated using only high-confidence assignments. This resembles an online clustering layer, but the key constraint is that we must preserve per-task heterogeneity: we therefore interpret K as the number of prototypes actually used (i.e. those receiving at least τ confident points) and treat the rest as irrelevant to the current task. In this way, persistent prototypes serve as a shared scaffold, while task-specific heads remain variable-dimensional.

Temporal smoothing and warm-starting. Streaming emphasizes a practical tension: discrete pseudo-partitions can change abruptly even under small parameter updates. Two mitigations are immediate. First, in the graph constructor we can warm-start community detection from the previous iteration's labeling for the same task identity (or for a matched task obtained by approximate nearest neighbor search in task space), performing only a few refinement passes; this both reduces computation and discourages spurious label flips. Second, for both constructors we can explicitly smooth the embeddings used for construction by maintaining an EMA body $\bar{\theta}$ and constructing $\hat{\Pi}$ from $f_{\bar{\theta}}$ while adapting and updating using f_θ. This decouples pseudo-label stability from fast inner-loop dynamics, and it is compatible with the invariant that gradients need not pass through g.

Multi-source pseudo-labeling and compatibility. Because g is an interchangeable module, it is natural to combine multiple weak constructors. Let $g^{(1)}, \dots, g^{(R)}$ be candidate constructors producing partitions $\hat{\Pi}^{(r)}$ and pseudo-labels $\hat{y}^{(r)}$. A conservative fusion is to accept a label only when sources agree: define a point i as inlier if there exists a consensus label under an alignment between cluster identities (computed, for example, by maximum bipartite matching on overlap counts between clusterings), and otherwise set $\hat{y}_i = \perp$. This reduces effective supervision noise at the cost of lowering the retained fraction α. One may then trade off noise versus coverage by allowing partial agreement: for each point we can compute an empirical distribution over labels induced by the sources after alignment and minimize a soft cross-entropy between head predictions and this distribution, excluding points whose entropy exceeds a threshold.

A complementary approach is *heterogeneous multi-view* pseudo-labeling: run g_{Graph} on a similarity derived from cosine distance (or a learned metric), and run g_{OT} on Euclidean distance with temperature τ, then encourage agreement via a consistency term. In the OT case, since P is differentiable with respect to S (and hence to Z) when we backpropagate through

Sinkhorn, we can optionally add a regularizer that aligns P with a graph-based soft label propagation distribution $\tilde{P}$ computed on the kNN graph. This yields an objective of the schematic form

$$L_{\text{inner}} \;=\; \mathrm{CE}(\hat{y}, h(Z)) \;+\; \lambda\, \mathrm{KL}(P \,\|\, \tilde{P}),$$

with $\hat{y}$ used only for confident points. The intent is not to claim global optimality of either constructor, but to exploit their different failure modes: OT tends to be stable under small perturbations but may blur fine structure; graph methods capture local connectivity but may be sensitive to neighbor errors.

Interplay with stability and robustness diagnostics. Stability measures such as ARI/NVI across outer iterations, as well as perturbation sensitivity under $z_i \mapsto z_i + \xi_i$, can be used not only for reporting but also as a control signal. We can treat instability as a symptom of operating in a regime where f_θ has not yet separated the data or where the constructor hyperparameters are too aggressive. Concretely, in the graph constructor we may adapt k and τ online: if measured instability increases, we increase τ (reject more small communities) and/or increase k (improve within-cluster connectivity), subject to a time budget. In the OT constructor we may adjust the hardening threshold β and the temperature τ: high instability suggests that assignment margins are small, hence we either raise β (more abstention) or decrease τ (sharper affinities) depending on whether instability is due to ambiguity or due to oversmoothing.

We may also incorporate a *partition inertia* constraint. Suppose a task T is revisited (or approximated by a near-duplicate under data augmentation). Let $\hat{y}^{\text{old}}$ be a stored pseudo-labeling for its previous embedding snapshot, and let $\hat{y}^{\text{new}}$ be the current labeling. After aligning labels, we can penalize disagreement on the intersection of non-outliers:

$$L_{\text{stab}} \;=\; \frac{1}{|\mathcal{I}|} \sum_{i \in \mathcal{I}} \mathbf{1}\{\hat{y}_i^{\text{old}} \neq \hat{y}_i^{\text{new}}\}, \qquad \mathcal{I} := \{i : \hat{y}_i^{\text{old}} \neq \bot,\ \hat{y}_i^{\text{new}} \neq \bot\}.$$

For OT, an analogous differentiable inertia is $\|P - P^{\text{old}}\|_1$ on stored soft assignments. Such terms formalize the intuition that pseudo-supervision should not oscillate faster than the representation can adapt.

Failure modes suggested by diagnostics. The diagnostics also help classify constructor-specific failures. For graph methods, low ANN recall produces spurious inter-cluster edges that often manifest as sudden merges; this is detected by a sharp drop in K and a large NVI jump. For OT, an overly small anchor budget A (or poorly placed prototypes) yields persistent high abstention $\alpha \ll 1$ or systematic collapse of multiple modes onto a single anchor, visible as a heavy-tailed cluster size distribution. In both cases, monitoring (K, α) jointly with stability metrics is essential: stability alone can be achieved trivially by declaring all points outliers, while high α alone can be achieved by forcing assignments even when margins are small. The practical extension is thus a *constraint-based* tuning rule: maintain α within a target interval and minimize instability subject to time constraints by adjusting (k, τ) or (A, I, β).

These extensions preserve the core design: variable K, principled outliers, and near-linear per-task cost. They also clarify that the constructor is not merely a preprocessing step but a control point that mediates the noise–coverage–stability trade-off that ultimately governs bilevel training dynamics.

4.9 Limitations and open questions: non-differentiability, approximation bias, and worst-case behavior

Our proposed constructors g_{Graph} and g_{OT} trade exact density connectivity for scalability and meta-compatibility, and this trade induces limitations that are not merely implementational but structural. We collect the most salient issues and outline open questions toward tighter end-to-end guarantees for bilevel meta-learning with pseudo-partitions.

Non-differentiability and gradient mismatch. For the graph constructor, the mapping $Z \mapsto \hat{\Pi}$ is discrete (ANN neighbor selection, community detection, and the subsequent label map), and we typically do not backpropagate through it. Consequently, the outer-loop gradient $\nabla_\theta L_{\text{outer}}$ is computed with g treated as a stop-gradient operator, even though g depends on $Z = f_\theta(T)$. This introduces an inherent gradient mismatch: improvements to f_θ that would make g produce better pseudo-labels are only indirectly encouraged via downstream adaptation. In favorable regimes (where partitions are stable under perturbations), the mismatch may be negligible; however, near phase transitions where small changes in Z change $\hat{\Pi}$ discontinuously, the resulting training dynamics can be difficult to characterize. A concrete open problem is to formalize conditions under which the piecewise-constant dependence of $\hat{\Pi}$ on Z yields an outer objective that is "almost everywhere" smooth enough for standard stochastic optimization arguments, possibly via stability bounds on label changes (e.g. NVI control) implying bounded bias in the meta-gradient.

Approximation bias from ANN and heuristic community routines. The near-linear runtime relies on approximate neighbor search and on community detection heuristics that do not optimize a canonical objective with provable approximation guarantees in general graphs. This yields two distinct biases. First, ANN errors perturb E; even if the planted-partition assumptions hold for the true kNN graph, the approximate graph may introduce adversarial inter-cluster edges or delete crucial intra-cluster edges, breaking the recovery conditions used in our analysis. Second, common community routines (label propagation, Louvain-style modularity maximization, greedy merges) may have multiple local optima, and their outputs can depend sensitively on tie-breaking and iteration order, creating additional stochasticity across outer iterations. While one can empirically improve robustness (e.g. by ensembling, warm starts, or multiple random seeds), the theoretical question remains: can we obtain recovery and stability guarantees for *the full pipeline* (ANN + community routine) under explicit and verifiable conditions on embedding separation, ANN recall, and graph expansion? Even for benign random graph models, understanding how heuristic variability translates into pseudo-label noise is largely open.

OT-specific limitations: anchor coverage, smoothing, and capacity mismatch. The OT constructor replaces combinatorial partitioning by a softened assignment P whose stability can be controlled via temperature τ and the Sinkhorn iterations. This differentiability is partial: while P is continuous in Z for fixed anchors, the *choice* of anchors and the hardening/abstention step reintroduce non-smoothness. Moreover, OT introduces its own approximation bias: with too few anchors A, distinct modes can collapse onto a single anchor, while with too large τ the assignments can oversmooth and blur cluster boundaries. Thus, OT may be stable yet systematically wrong, and stability alone is not a sufficient diagnostic. An open direction is to quantify a *bias–variance* trade-off: increasing smoothing improves Lipschitz stability (variance reduction) but increases assignment bias, and it is unclear how this trade-off interacts with inner-loop adaptation, particularly when K is defined as the number of "active" anchors. A related question concerns head capacity: the dynamic head of size K is trained on pseudo-labels, but the representation f_θ is shared across tasks; we lack a principled characterization of when the induced per-task label spaces are compatible enough for a single body to support fast adaptation.

Worst-case failures and adversarial geometries. Our lower bound for exact DBSCAN emphasizes that avoiding $\Omega(m^2)$ work requires approximation, but approximation admits worst-case failures. In high-dimensional spaces, distance concentration and hubness can make kNN graphs unreliable: a few points become nearest neighbors of many others, and local neighborhoods cease to reflect latent clusters. Similarly, in data with heterogeneous densities, any fixed (k, τ) can be inappropriate: small dense clusters may be merged into larger ones by graph connectivity, while elongated manifolds can be fragmented by bounded-degree graphs. OT can fail in the opposite direction by enforcing approximately balanced transport to anchors, which may be misaligned with true cluster sizes. These issues suggest that no single constructor dominates uniformly, and the practical success of UHT depends on the implicit regularization induced by the constructor. A theoretical open question is to identify impossibility results (or minimax lower bounds) for variable-K partition recovery under computational constraints such as bounded-degree graphs, thereby clarifying when observed failures are unavoidable rather than artefacts of implementation.

Outliers, abstention, and identifiability in bilevel training. Outlier handling is essential to control noise amplification, yet abstention introduces a subtle selection bias: the inner loss CE is computed on a subset of points whose membership depends on Z. If the constructor preferentially retains "easy" points, the meta-learner may overfit to highly separable regions and neglect hard-but-important structure. Conversely, if abstention is too weak, pseudo-label noise may dominate, degrading both inner adaptation and outer updates. We currently lack a principled rule that links the retained fraction α (and the distribution of retained points) to guarantees on meta-learning performance. An open problem is to model abstention as a missing-data mechanism and to derive conditions under which minimizing L_{outer} on the retained set is consistent for a target objective (e.g. representation learning that supports downstream supervised tasks), possibly requiring assumptions about how the constructor's confidence correlates with correctness.

Toward end-to-end guarantees: coupling misclustering to meta-objectives. Finally, our recovery statements are phrased in terms of misclustering rates under a planted model, whereas the bilevel objective concerns post-adaptation loss. Bridging these requires a coupling argument: how does an ε-fraction of incorrect pseudo-labels affect Adapt and hence L_{outer}? The answer depends on the head architecture, regularization, inner-step count, and the margin properties induced by f_θ. A compelling direction is to develop bounds of the schematic form

$$\mathbb{E}[L_{\text{outer}}(\text{Adapt}(\cdot))] \;\le\; \mathbb{E}[L_{\text{outer}}(\text{Adapt}(\cdot;\Pi^*))] \;+\; \Psi(\varepsilon,\alpha,m,s),$$

where Ψ quantifies the degradation due to pseudo-label errors and abstention. Establishing such bounds would move beyond constructor-level correctness to an end-to-end theory of UHT that (i) incorporates approximation and non-differentiability, (ii) accounts for the stochasticity of tasks $T \sim \mathcal{T}$, and (iii) yields actionable prescriptions for choosing (k,τ) or (A,I,β) as a function of compute budgets and desired generalization behavior.