Cascaded Transfer: Learning Many Tasks under Budget Constraints

Anonymous Authors[1]

Abstract

Many-Task Learning refers to the setting where a large number of related tasks need to be learned, the exact relationships between tasks are not known, and budget constraints are in place. We introduce the *Cascaded Transfer Learning*, a novel many-task transfer learning paradigm where information (e.g. model parameters) cascades hierarchically though tasks that are learned by individual models of the same class, while respecting given budget constraints. The cascade is organized as a rooted tree that specifies the order in which tasks are learned and refined. We design a cascaded transfer mechanism deployed over a minimum spanning tree structure that connects the tasks according to a suitable distance measure, and allocates the available training budget along its branches. Experiments on synthetic and real many-task settings show that the resulting method enables more accurate and cost-effective adaptation across large task collections compared to alternative approaches.

1. Introduction

Modern learning systems increasingly operate in settings where a large number of related tasks must be handled under computational or training-budget constraints. This arises in domains involving personalized, localized, or fine-grained problems, where each task comes with limited data and solving all tasks independently becomes inefficient. Exploiting relationships across tasks is essential for achieving accurate models while maintaining feasible computational cost in such a context.

Multi-Task Learning (MTL) has been specifically developed for leveraging shared information across tasks, seeking to improve performance by joint training of multiple related tasks. The main MTL method families include feature-sharing architectures, low-rank parameterizations, task clustering and other methods that learn task relationships (Zhang & Yang, 2018; 2021; Ruiz et al., 2024). Although effective, they typically require synchronized and globally coordinated optimization for training across all tasks, which lead to substantial memory or communication overhead. Moreover, relying on accurate task relations is a sensitive point for MTL, hence there are methods that try to infer those relations, but in the literature this is most usually required to be apriori known.

As the number of tasks grows, these limitations become more pronounced, giving rise to the *Many-Task Learning* (MaTL) setting. Here, the focus shifts to handling hundreds or thousands of related tasks, whose relatedness is typically unknown or only partially observable, and often under resource constraints that make joint optimization impractical. The MaTL regime appears naturally in personalized modeling, distributed sensing, or large-scale prediction environments, where one seeks scalable mechanisms for exploiting task relatedness. Energy networks, transportation, retail, where thousands of related tasks arise in parallel (He et al., 2019), are among the typical application sectors that fall within the MaTL regime. Prior work has shown that scaling to many tasks requires explicit mechanisms to control parameter sharing and interference within a *single* model, such as task-specific routing (Strezoski et al., 2019), hierarchical many-task architectures (Hashimoto et al., 2017; Liu et al., 2021), or transferability-based task selection that guides efficient knowledge reuse across large task collections (Tan et al., 2024). Studies of extreme multi-task pre-training with over one hundred tasks further highlight the benefits and the complexity of this regime (Aribandi et al., 2022). In parallel, foundation models have recently been explored for graphs, with the goal of capturing generalities across heterogeneous graph tasks. A particularly illustrative instance of these pressures is found in modern time-series forecasting. Transformer-based architectures such as PatchTST (Nie et al., 2022) and universal forecasters like Moirai (Woo et al., 2024) demonstrate that shared representations can be remarkably effective across a wide variety of series and domains. Yet these approaches typically rely on large-scale centralized pretraining and assume that inference or fine-tuning can be performed without stringent per-task budget constraints. In applications involving thousands of forecast-

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

ing tasks (each requiring lightweight, task-specific adaptation) global training becomes costly or infeasible, and the problem shifts toward designing scalable mechanisms for transferring information across many related tasks.

Thinking about sharing information from one task to assist another without requiring global joint training, *Transfer Learning* (TL) comes naturally into the discussion. TL methods are categorized to instance-based, feature-based, parameter-based and relational transfer (Pan & Yang, 2009; Weiss et al., 2016; Zhuang et al., 2020). In practice, TL can be straightforward to deploy in modern neural settings and scales well to new tasks, but it treats each source–target pair independently. This pairwise structure limits its applicability into the MaTL regime, as it provides only simple deployment schemes such as star-shaped transfers from a single source model. Beyond such configurations, classical TL offers little guidance on how to organize or coordinate transfers across a large collection of tasks, which makes it inadequate as the primary mechanism for structuring information flow in MTL, not to mention the MaTL regime. We focus on a frugal many-task regime in which each task is refined only once under a strict global budget, and propose *Cascaded Transfer Learning* (CTL) as a single-pass alternative to joint multi-task optimization when repeated task revisiting is infeasible.

Contributions. In this work, we propose a new approach that occupies the conceptual space between these two paradigms, MTL and TL. Rather than adapting a source model directly to each target task, we introduce *Cascaded Transfer Learning*, a mechanism that propagates knowledge through a sequence of tasks (see Figure 1). The key idea is that coordinated transfer across a graph structure connecting the tasks can realize gains that are not accessible through isolated, direct adaptations. To the best of our knowledge, this is the first work to provide a theoretical and algorithmic framework for budget-constrained cascaded transfer over large task collections. Our contributions are the following:

(i) We provide a theoretical analysis of cascaded transfer over trees, establishing sufficient conditions under which CTL provably improves over direct transfer. Our results characterize how the stability of local optimization, task similarity, and cascade depth jointly control error propagation, and when tree-structured transfer is beneficial.

(ii) Building on this analysis, we propose a scalable CTL algorithm that constructs a transfer tree and allocates a limited training budget across tasks. We empirically validate the approach on synthetic benchmarks, multisite forecasting, and image classification, showing consistent improvements over independent training and standard transfer baselines under equal or lower computational budgets.

Notations. Let $\mathcal{V}$ denote the set of tasks, with $|\mathcal{V}|$ its cardinality. For two tasks $u, v \in \mathcal{V}$, we write $u \to v$ to denote a transfer from task u to task v. The norms $\|\cdot\|$ and $\|\cdot\|_F$ respectively denote the Euclidean norm and the Frobenius norm. For a symmetric matrix $\mathbf{A}$, the notation $\mathbf{A} \succ 0$ indicates that $\mathbf{A}$ is positive definite. Finally, $\lceil \cdot \rceil$ denotes the ceiling function.

2. Preliminaries

For each task $v \in \mathcal{V}$, let $\boldsymbol{X}_v \in \mathbb{R}^{n \times d}$ denote the task-specific data matrix, where $n \in \mathbb{N}^*$ is the number of samples available for task v and $d \in \mathbb{N}^*$ is the feature dimension. Each task is associated with an unknown parameter vector $\boldsymbol{\theta}_v \in \mathbb{R}^d$ to be learned from data. For each task v, let $b_v \in \mathbb{N}^*$ denote the computational budget allocated to that task. Throughout this work, a *budget* denotes a fixed amount of local optimization effort, measured in refinement steps. We denote by $G_v^{b_v} : \mathbb{R}^d \to \mathbb{R}^d$ a refinement operator corresponding to b_v iterations of a gradient-based optimization algorithm applied to the task-specific parameter. Given a source task u and a target task v, a transfer $u \to v$ consists of initializing the parameters of task v using information from task u, followed by local refinement via $G_v^{b_v}$. This source–target transfer constitutes the elementary building block of cascaded transfer learning. To coordinate multiple such transfers across tasks, we introduce graph-structured dependency relations over $\mathcal{V}$, which specify how knowledge is propagated and refined throughout the task set.

In this work, we focus on *rooted trees* as the primary structure for organizing transfer. A rooted tree is a *directed acyclic graph* (DAG) in which each node has at most one parent, denoted $\mathrm{pa}(v)$. The unique node with zero in-degree is called the *root*, while nodes with zero out-degree are referred to as *leaves*. This structure induces a natural flow of information from the root toward the leaves, defining an unambiguous order in which tasks are processed. More generally, a DAG is any directed graph without cycles and admits a topological ordering such that, for each edge (u, v), node u precedes node v in the order. In this broader setting, a task may receive information from multiple parent tasks, which can be combined through an aggregation operator. While this abstraction is useful for unifying different transfer mechanisms, both our theoretical analysis and algorithmic instantiations are restricted to rooted trees, which capture the essential behavior of cascaded transfer while remaining analytically and computationally tractable.

2.1. The Cascaded Transfer Learning Paradigm

We now formalize CTL, starting from the rooted tree setting.

Definition 2.1 (Cascaded Transfer Learning)**.** Let $\mathcal{T} = (\mathcal{V}, \mathcal{E})$ be a rooted tree whose nodes correspond to learning tasks. *Cascaded Transfer Learning* is a learning process

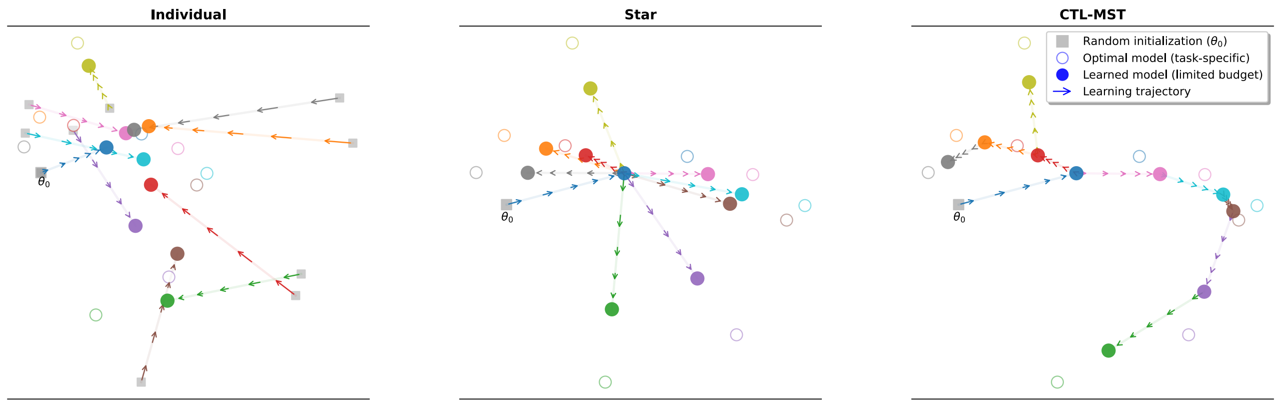


Figure 1. **Parameter-space intuition for CTL.** *Left:* Independent training. Each task is optimized from its own initialization and stops short of its optimum due to limited budget. *Middle:* Star transfer. One source task is learned first and directly transferred to all other tasks. *Right:* CTL with an MST. Tasks are learned sequentially along a minimum spanning tree, decomposing long transfers into local steps. Solid nodes denote task optima, white nodes learned parameters, and arrows optimization trajectories.

Algorithm 1 Cascaded Transfer Learning

```
1:  Input: set of tasks $\mathcal{V} = \{v_1, v_2, \dots\}$, common information space $\mathcal{I}$, total budget $B$
2:  Output: refined information $\{\widetilde{I}_v\}_{v\in\mathcal{V}}$
3:  Seed selection: choose the root task $s \in \mathcal{V}$
4:  Tree construction: construct a rooted tree $\mathcal{T} = (\mathcal{V}, \mathcal{E})$ defining the information flow
5:  Budget allocation: assign refinement budgets $\{b_v\}_{v\in\mathcal{V}}$ such that $\sum_v b_v = B$
6:  Cascaded transfer: traverse the tree in a topological (root-to-leaf) order; for each task $v$:
7:    If $v = s$, initialize $I_v^{(0)}$ arbitrarily
8:    If $v \neq s$, let $u = \mathrm{pa}(v)$ and set $I_v^{(0)} = \widetilde{I}_u$
9:    Refine: $\widetilde{I}_v = G_v^{b_v}\left(I_v^{(0)}\right)$
10: Return: refined information $\{\widetilde{I}_v : v \in \mathcal{V}\}$
```

in which tasks are trained sequentially following the tree structure: the root task is trained first, and each other task is initialized from its parent and then refined using a local optimization procedure.

Figure 1 provides an overview of the CTL framework, illustrating how, starting from a chosen seed, information is propagated from task to task along the cascade. The CTL framework involves three central design choices:

(i) *Seed selection.* The root task s that initiates the cascade.

(ii) *Graph construction.* The rooted tree defining the information flow between tasks.

(iii) *Budget allocation and refinement.* The assignment of refinement budgets $\{b_v\}_{v\in\mathcal{V}}$ and the execution of local optimization along the cascade.

Algorithm 1 summarizes the CTL process.

2.2. Cascade Construction Strategies

Cascaded Transfer Learning relies on the idea that tasks can benefit from information propagated through a predefined structure connecting them. Rather than treating transfers separately, CTL schedules task learning so that each task is refined using information inherited from neighboring tasks, under a global budget constraint. The central question is therefore not only how to define neighborhoods between tasks, but also under which conditions routing information through an intermediate task is beneficial.

At the local level, consider three tasks (S, I, T), where S and T are neighbors according to a given task distance. A direct transfer $S \to T$ is not necessarily optimal under a fixed training budget: in certain regimes, refining an intermediate task I and transferring along the path $S \to I \to T$ can lead to improved performance. Section 3 characterizes sufficient conditions under which the error introduced by intermediate optimization remains controlled and can accumulate favorably along such paths. In particular, when local optimization procedures are stable and task distances are sufficiently small along the path, information can be propagated through cascades without degrading performance. When this behavior occurs systematically across a collection of tasks, it becomes natural to schedule transfers through a global structure that connects neighboring tasks while supporting propagation through intermediates. Rooted trees provide a simple and effective way to implement this idea: each task receives information from a limited set of upstream neighbors and contributes refined information downstream, ensuring a coherent and budget-aware diffusion process.

The specific choice of tree construction depends on how task distances are estimated and on the reliability of the available information. In practical settings, task relationships are typically inferred from noisy proxies such as empirical features, gradients, or learned representations. In this context, distance-based constructions provide a robust way to organize transfer. In particular, MSTs connect neighboring tasks using pairwise distances while minimizing total edge cost, encouraging locality and limiting the accumulation of transfer error. As a result, MST-based cascades constitute a strong default choice for CTL.

Extensions. The abstract CTL framework can be naturally generalized in a number of directions that we mention here, although their proper investigation is out of the scope of this paper. First, while our presentation is restricted to a single rooted tree, the same principles apply to a *cascade forest* in which multiple seeds initiate independent cascades over disjoint subsets of tasks. Deciding when more than one seed would be beneficial and selecting those seeds, is itself a design problem that can be informed by task distances, clustering criteria, or transferability estimates. Second, CTL does not rely on supervised losses: any setting in which tasks share a common information space $\mathcal{I}$ and a local adaptation operator, can support cascaded transfer, including unsupervised, self-supervised, or representation learning scenarios. Third, Definition 2.1 can be stated more generally for DAGs, which include rooted trees. That would allow a task to fuse information from multiple parents.

3. Theoretical Analysis of Cascaded Transfer Learning

We justify CTL over rooted trees, regardless how the tree is constructed. Our analysis shows that (i) tree-structured cascades can improve over independent or pairwise transfer by routing long-range transfers through sequences of shorter ones in terms of task distance, (ii) only mild structural conditions on the tree are required, and (iii) distance-based trees such as MSTs naturally satisfy these conditions when task distances are only imperfectly observed.

3.1. Parameter-Space Analysis

For each task $v \in \mathcal{V}$, we associate a loss $\mathcal{L}_v : \mathbb{R}^d \to \mathbb{R}$ and minimizer $\boldsymbol{\theta}_v^\star \in \mathbb{R}^d$. Let $\eta \in (0,1)$ be the learning step associated to the refinement operator G_v. We assume a contraction property in parameter space: for all $v \in \mathcal{V}$, there exists $\rho_v \in (0,1)$ such that for all $(\boldsymbol{\theta}, b) \in \mathbb{R}^d \times \mathbb{N}^*$,

$$\|G_v^b(\boldsymbol{\theta}) - \boldsymbol{\theta}_v^\star\| \le \rho_v^b \|\boldsymbol{\theta} - \boldsymbol{\theta}_v^\star\|.$$

This holds, for instance, for gradient descent on strongly convex and smooth losses. We assume a latent task geometry $d(u,v) = \|\boldsymbol{\theta}_u^\star - \boldsymbol{\theta}_v^\star\|$, as a proxy of the transfer difficulty between tasks.

Let $\mathcal{T}$ be a rooted tree with root s. Given a budget allocation $\{b_v\}_v$ with $\sum_v b_v = B \in \mathbb{N}^*$, CTL refines tasks in a topological order:

$$\tilde{\boldsymbol{\theta}}_s = G_s^{b_s}(\boldsymbol{\theta}_{\text{init}}), \qquad \tilde{\boldsymbol{\theta}}_v = G_v^{b_v}(\tilde{\boldsymbol{\theta}}_{\text{pa}(v)}),\ v \ne s.$$

Proposition 3.1 (Edge-wise propagation)**.** *For any non-root node v with $u = \text{pa}(v)$, and for $j, m \in \mathbb{N}^*$ such that $S_{i,m} = \sum_{\ell=i}^m b_{v_\ell}$,*

$$\|\tilde{\boldsymbol{\theta}}_v - \boldsymbol{\theta}_v^\star\| \le \rho_v^{b_v}\Big(\|\tilde{\boldsymbol{\theta}}_u - \boldsymbol{\theta}_u^\star\| + d(u,v)\Big).$$

The recursion along the unique root-to-v path yields

$$\|\tilde{\boldsymbol{\theta}}_v - \boldsymbol{\theta}_v^\star\| \le \prod_{i=1}^m \rho_{v_i}^{b_{v_i}} \|\tilde{\boldsymbol{\theta}}_s - \boldsymbol{\theta}_s^\star\| + \sum_{i=1}^m \rho_{v_i}^{S_{i,m}} d(v_{i-1}, v_i),$$

where each edge contribution is damped by downstream refinement.

This bound shows that CTL improves over direct transfer when: (i) edges connect nearby tasks (locality), (ii) the structure is acyclic and ordered, (iii) budgets prevent error accumulation along deep paths. These conditions depend on edge lengths, not on how the tree is constructed.

MST as a default. When only pairwise dissimilarities are available, minimizing cumulative edge costs is a natural objective. Minimum spanning trees (MST) satisfy locality, are acyclic by construction, and are stable to noise in similarity estimates, making them a robust default choice.

CTL vs TL. Why Cascades? The central question is whether routing transfer through intermediate tasks can reduce the error induced by task mismatch, under a fixed refinement budget. The following result formalizes this intuition by isolating the *transfer bias* induced by task geometry.

Theorem 3.2. *Assume the seed task is solved exactly, $\tilde{\boldsymbol{\theta}}_s = \boldsymbol{\theta}_s^\star$. Let $(v_0 = s \to v_1 \to \cdots \to v_m = v)$ be a path in a CTL tree, with uniform refinement budget b on each non-root node, $\rho_{\max} = \max_{1 \le i \le m} \rho_{v_i}$, $\delta_i = d(v_{i-1}, v_i)$, and $\delta_{\max} = \max_{1 \le i \le m} \delta_i$.*

Then CTL satisfies

$$\|\tilde{\boldsymbol{\theta}}_v^{\text{CTL}} - \boldsymbol{\theta}_v^\star\| \le \sum_{i=1}^m \rho_{v_i}^{(m-i+1)b}\, \delta_i,$$

whereas a direct transfer strategy satisfies

$$\|\tilde{\boldsymbol{\theta}}_v^{\text{TL}} - \boldsymbol{\theta}_v^\star\| \le \rho_v^b d(s,v).$$

Consequently, CTL yields a strictly smaller upper bound whenever

$$\sum_{i=1}^m \rho_{v_i}^{(m-i+1)b}\, \delta_i < \rho_v^b d(s,v).$$

A sufficient condition is

$$\delta_{\max}(1-\rho_{\max}^{mb}) < d(s,v)(1-\rho_{\max}^{b}).$$

Theorem 3.2 highlights a fundamental distinction between cascaded and direct transfer. Star transfer incurs a *single* geometric error proportional to the global distance $d(s,v)$ between the seed and the target, which is discounted only once by the local refinement. In contrast, CTL decomposes this global mismatch into a sequence of local discrepancies $\delta_i = d(v_{i-1}, v_i)$, each of which is further attenuated by all downstream refinements. As a result, when tasks vary smoothly along the cascade (small δ_i), the discounted sum of local errors is strictly smaller than the discounted global jump. This explains why locality-preserving trees are effective: CTL exploits smoothness of the task geometry rather than relying on a single global alignment.

3.2. Feature-Space Analysis

We now consider the linear feature-space setting with random design and $n \geq d$. For each task $v \in \mathcal{V}$, let $\boldsymbol{X}_v$ be a random variable taking values in $\mathbb{R}^{n\times d}$, and let $\boldsymbol{y}_v$ be a random variable taking values in $\mathbb{R}^n$. We consider the quadratic objective

$$\mathcal{L}_v(\boldsymbol{\theta}) \;=\; \tfrac{1}{2}\|\boldsymbol{X}_v\boldsymbol{\theta} - \boldsymbol{y}_v\|^2, \qquad \boldsymbol{\theta}\in\mathbb{R}^d.$$

A single gradient descent step with step size $\eta > 0$ yields the affine update $G_v(\boldsymbol{\theta}) = \mathbf{M}_v\boldsymbol{\theta} + \eta\boldsymbol{X}_v^\top\boldsymbol{y}_v$, where $\mathbf{M}_v = \mathbf{I}_d - \eta\boldsymbol{X}_v^\top\boldsymbol{X}_v$.

We assume that $\boldsymbol{X}_v^\top\boldsymbol{X}_v \succ 0$ almost surely and choose a step size $\eta \in \big(0, 2/\lambda_{\max}(\boldsymbol{X}_v^\top\boldsymbol{X}_v)\big)$, where $\lambda_{\max}$ denotes the largest eigenvalue. Under this condition, the linear operator $\mathbf{M}_v = \mathbf{I}_d - \eta\boldsymbol{X}_v^\top\boldsymbol{X}_v$ satisfies $\|\mathbf{M}_v\|_2 = \rho_v < 1$, and therefore the refinement operator G_v is a contraction on $(\mathbb{R}^d, \|\cdot\|_2)$ almost surely. We further assume a realizable random-design linear model: for each task $v\in\mathcal{V}$, there exists a parameter vector $\boldsymbol{\theta}_v^\star \in \mathbb{R}^d$ such that $\boldsymbol{y}_v = \boldsymbol{X}_v\boldsymbol{\theta}_v^\star$. In this setting, gradient-based refinement contracts toward the task-specific optimum $\boldsymbol{\theta}_v^\star$, and the same edge-wise and path-wise propagation bounds as in parameter space apply directly.

Proposition 3.3 (Edge-wise propagation in feature space)**.** *With the same notations as in Proposition 3.1,*

$$\|\tilde{\boldsymbol{\theta}}_v - \boldsymbol{\theta}_v^\star\| \leq \rho_v^{b_v}\Big(\|\tilde{\boldsymbol{\theta}}_u - \boldsymbol{\theta}_u^\star\| + d(u,v)\Big).$$

Unrolling this recursion along the unique root-to-v path yields

$$\|\tilde{\boldsymbol{\theta}}_v - \boldsymbol{\theta}_v^\star\| \leq \prod_{i=1}^{m}\rho_{v_i}^{b_{v_i}}\|\tilde{\boldsymbol{\theta}}_s - \boldsymbol{\theta}_s^\star\| + \sum_{i=1}^{m}\rho_{v_i}^{S_{i,m}}\, d(v_{i-1}, v_i),$$

where each edge contribution is damped by downstream refinement.

3.3. Noisy Feature-Space Analysis

We consider a random-design linear regression model with additive observation noise. For each task $v\in\mathcal{V}$, let $\boldsymbol{X}_v$ be a random variable taking values in $\mathbb{R}^{n\times d}$, and let $\boldsymbol{y}_v$ be a random variable taking values in $\mathbb{R}^n$. We assume that $\boldsymbol{y}_v$ admits the decomposition $\boldsymbol{y}_v = \boldsymbol{X}_v\boldsymbol{\theta}_v^\star + \boldsymbol{\varepsilon}_v$, where $\boldsymbol{\theta}_v^\star \in \mathbb{R}^d$ is a deterministic task-specific parameter and $\boldsymbol{\varepsilon}_v$ is a noise random variable taking values in $\mathbb{R}^n$ with independent, mean-zero, sub-Gaussian coordinates.

We assume that the feature-space contraction condition holds for a suitable step size $\eta > 0$. As in the previous setting, gradient-based refinement induces a contraction toward a task-dependent optimum. In the presence of observation noise, however, this contraction occurs toward the empirical minimizer $\widehat{\boldsymbol{\theta}}_v = \arg\min_{\boldsymbol{\theta}\in\mathbb{R}^d} \frac{1}{2}\|\boldsymbol{X}_v\boldsymbol{\theta} - \boldsymbol{y}_v\|_2^2$, rather than toward the population parameter $\boldsymbol{\theta}_v^\star$. The discrepancy $\widehat{\boldsymbol{\theta}}_v - \boldsymbol{\theta}_v^\star$ captures statistical estimation error and perturbs the task geometry induced by the feature space. In expectation, the magnitude of this perturbation is controlled by the conditioning of the task-specific design through the linear operator $\mathbf{A}_v = (\boldsymbol{X}_v^\top\boldsymbol{X}_v)^{-1}\boldsymbol{X}_v^\top$, which governs how observation noise propagates from data space to parameter space.

Proposition 3.4 (Expected noisy propagation along a CTL tree)**.** *Under the noisy linear model and the feature-space contraction assumption, the expected error propagation along a CTL tree satisfies the following recursion. With the same notations as in Proposition 3.1,*

$$\mathbb{E}\big[\|\tilde{\boldsymbol{\theta}}_v - \boldsymbol{\theta}_v^\star\|\big] \leq \rho_v^{b_v}\Big(\mathbb{E}\big[\|\tilde{\boldsymbol{\theta}}_u - \boldsymbol{\theta}_u^\star\|\big] + d(u,v)\Big) + \sigma_v(1+\rho_v^{b_v})\|\mathbf{A}_v\|_F.$$

Unrolling this recursion along the root-to-v path ($v_0 = s \to \cdots \to v_m = v$) yields

$$\mathbb{E}\big[\|\tilde{\boldsymbol{\theta}}_v - \boldsymbol{\theta}_v^\star\|\big] \leq \Big(\prod_{j=1}^{m}\rho_{v_j}^{b_{v_j}}\Big)\mathbb{E}\big[\|\tilde{\boldsymbol{\theta}}_s - \boldsymbol{\theta}_s^\star\|\big] + \sum_{i=1}^{m}\Big(\prod_{j=i}^{m}\rho_{v_j}^{b_{v_j}}\Big)d(v_{i-1}, v_i) + \sum_{i=1}^{m}\Big(\prod_{j=i+1}^{m}\rho_{v_j}^{b_{v_j}}\Big)\sigma_{v_i}(1+\rho_{v_i}^{b_{v_i}})\|\mathbf{A}_{v_i}\|_F.$$

This bound shows that, in expectation, CTL decomposes the final error into two discounted components: a bias term, and a noise term capturing the propagation of estimation error. As in the noiseless setting, locality reduces transfer bias, while sufficient downstream refinement or shallow cascades limit noise accumulation.

4. Cascade Construction Algorithm

Building on the theoretical analysis of Section 3, we describe a simple and scalable procedure for constructing tree-structured cascades that organize information transfer under a global budget constraint. The cascade is defined by selecting a representative seed task and a rooted tree that specifies how transfer is routed across tasks.

4.1. Seed Selection and Tree Construction

The seed task s serves as the root of the cascade and influences all downstream transfers. To ensure robustness to heterogeneity and noise, we select the *medoid* of the task set, defined as the task minimizing the sum of distances to all others. Unlike centroid-based choices, the medoid is an actual task and provides a stable, parameter-free, and outlier-resistant initialization across settings.

Given a notion of task distances, the cascade structure is specified by a rooted tree that determines how information propagates from the seed to the remaining tasks. We focus on distance-based constructions, where a symmetric task distance matrix is used to connect tasks through local neighborhoods. In this setting, MSTs provide a robust default, as they favor short transfers, limit error accumulation, and remain stable under noisy distance estimates. The resulting tree is rooted at the seed and oriented away from it to define the cascade, while alternative construction strategies can be substituted within the same CTL framework when richer task information is available.

4.2. Computational Complexity

The computational cost of CTL is dominated by the construction of the cascade structure. For distance-based cascades such as MSTs, building a rooted tree over $|\mathcal{V}|$ tasks from pairwise distances requires $\mathcal{O}(|\mathcal{V}|^2 \log |\mathcal{V}|)$ time. While quadratic in the number of tasks, this cost remains tractable for the task collections considered in our experiments and is incurred only once, prior to adaptation. Once the cascade is constructed, training proceeds locally along the tree: each task is refined exactly once from its parent, and the total optimization cost scales linearly with the global refinement budget. As a result, the dominant cost during deployment is governed by the budgeted local updates rather than by joint or repeated task optimization.

5. Experiments

The goal of our experiments is twofold: (i) to demonstrate that CTL effectively exploits task-graph structure to improve performance under a fixed or reduced computational budget, and (ii) to show that these gains are not specific to a particular data modality or learning problem. We therefore evaluate CTL across a diverse set of settings, including synthetic and real-world regression, as well as image-based classification tasks. Across all experiments, learning follows the same paradigm. Each task is associated with a local objective and is optimized independently using gradient-based methods. Task interactions occur exclusively through parameter initialization: each node in the cascade is initialized from its parent in the tree, after which local refinement is performed. The task graph is constructed using training data only, while all reported results are computed on held-out test data, ensuring that task relationships are inferred without access to evaluation samples.

5.1. Datasets

Synthetic Dataset. We construct families of linear regression tasks of the form $\boldsymbol{y}_v = \boldsymbol{X}_v\boldsymbol{\theta}_v + \boldsymbol{\varepsilon}_v$, where $\boldsymbol{X}_v \in \mathbb{R}^{n\times d}$ has i.i.d. standard Gaussian rows and $\boldsymbol{\varepsilon}_v \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I}_n)$. To control task relatedness, we consider the following generative model: we first draw a global center $\boldsymbol{\theta}_0$ and K cluster shifts $\delta_c \sim \mathcal{N}(\mathbf{0}, \tau^2_{\text{between}}\mathbf{I}_d)$, every task v is assigned a cluster $c(v)$ and its parameter is generated as $\boldsymbol{\theta}_v = \boldsymbol{\theta}_0 + \delta_{c(v)} + \boldsymbol{\zeta}_v$ with $\boldsymbol{\zeta}_v \sim \mathcal{N}(\mathbf{0}, \tau^2_{\text{within}}\mathbf{I}_d)$. Varying τ or $(\tau_{\text{within}}, \tau_{\text{between}})$ controls the degree and structure of heterogeneity, enabling systematic evaluation across regimes ranging from nearly homogeneous tasks to well-separated clusters, see Figure 3.

Real Datasets. We evaluate CTL on both time-series forecasting and image classification. For time-series regression, we use the UK electricity dataset[1], which provides aggregated half-hourly residential demand from smart meters operated by two major UK distribution network operators. We focus on a subset of 100 nodes in the Oxford urban area and predict feeder-level demand using calendar features and lagged load values. To assess generality beyond regression, we also consider high-dimensional image classification tasks derived from Fashion-MNIST and CIFAR-10. Each task corresponds to a binary classification problem defined over a pair of classes, and multiple related tasks are obtained by training on different subsets of the data. This setting yields collections of closely related but non-identical tasks, making it well suited for evaluating cascaded transfer learning in classification regimes.

5.2. Experimental Settings

For all regression experiments, datasets are split into disjoint training and test sets. In the synthetic setting, we use fixed sizes of $n_{\text{train}} = 64$ and $n_{\text{test}} = 128$ per task. For the real-world time-series dataset, the training set spans from February 14, 2024, at 12:00 a.m. to February 24, 2024, at 11:30 p.m., while the test set spans from February 25, 2024,

[1]`https://weave.energy/`

at 12:00 a.m. to February 28, 2024, at 11:30 p.m. For image classification experiments, each task is trained and evaluated using fixed-size train and test splits, and all methods use the same model architecture and optimization hyperparameters across tasks. In all cases, task graphs are constructed exclusively from training data. Models are evaluated on the test set using the same metrics (root mean squared error) across all methods, and all results are averaged over 50 random seeds to account for variability due to data sampling and initialization.

5.3. Baselines

We compare our approach with several tree-structured or non-transfer baselines. All methods use identical training routines, both for seed training and and local refinement.

– *Individual models.* For each task, an individual model is trained independently for a fixed computational budget. This serves as a baseline that uses no multi-task or transfer learning.

– *Star Tree.* A seed task s is selected, and then a star graph is constructed: $(s \to v)$, for all v. Each node receives the seed parameters and performs independent refinement.

– *Random Tree.* A seed task s is selected, and a random spanning tree over all tasks is generated by sampling a Prüfer sequence uniformly (Kumar et al., 1998; Deo & Micikevicius, 2001), which yields a labeled tree drawn uniformly from the set of all spanning trees. The tree is then rooted at s by orienting all edges away from it. This baseline represents an uninformed hierarchical transfer structure that does not exploit any task-related information.

– *Minimum-Spanning Tree.* We consider several MST-based cascades that differ only in the task distance used to construct the tree. We evaluate distances based on target similarity, feature representations, optimization geometry (gradient or model-based), and distributional or representation-level discrepancies (e.g. KL, Wasserstein, MMD, CKA). Full definitions of the distance metrics are provided in the appendix.

We do not compare CTL to multi-task learning or deep neural network baselines, as these methods optimize a fundamentally different objective through repeated joint training and shared representations, typically assuming abundant compute and task revisiting. In contrast, CTL targets frugal many-task adaptation, where each task is refined once under a strict global budget. To preserve alignment between theory and experiments, we therefore focus on linear and convex models, where contraction and error propagation can be characterized explicitly and resource usage is controlled.

5.4. Results

Table 2 and Figure 2 provide a unified view of all experiments across synthetic regression, UK electricity forecasting, and image classification, enabling direct comparison of transfer structures, task geometries, and budgets.

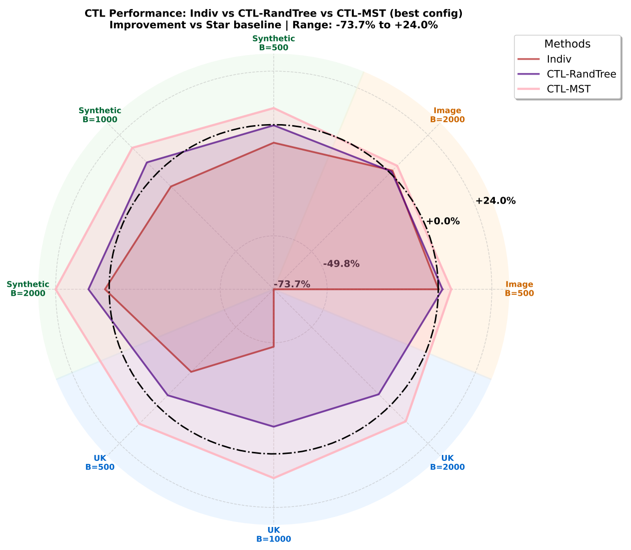


Figure 2. Radar plot of average percentage improvement over direct transfer training across datasets and budgets.

Synthetic regression. Across all regimes, cascaded transfer consistently outperforms independent training and shallow baselines, confirming that exploiting task structure remains beneficial even under increasing heterogeneity. At low to moderate variability, distances aligned with the underlying optimization or model geometry (for instance, gradient and parameter-based) yield the lowest errors across budgets, leading to improvements of 20–35% relative to independent training. As heterogeneity increases ($\tau = 10$), absolute gains diminish and variance grows, reflecting weaker task relatedness; nevertheless, optimization-aligned distances remain systematically superior to purely feature-based or uninformed measures. Distributional distances provide competitive but less stable performance, while feature-based distances degrade more rapidly with increasing dispersion.

UK electricity forecasting. Across all settings, independent training yields the highest errors, highlighting the limitations of learning tasks in isolation. Shallow star-shaped transfer improves performance but exhibits higher variance and degrades in low-budget regimes, indicating sensitivity to unfavorable source–target mismatches. In contrast, cascaded transfer consistently reduces RMSE across budgets, with distances aligned to target or optimization geometry delivering the most stable gains. At low budget, cascaded transfer reduces RMSE by more than 25% relative to independent training, while at higher budget reductions approach 50%. The aggregate results in Figure 2 confirm that geometry-aware distances consistently outperform unstructured or feature-only measures, demonstrating that cascaded transfer effectively amortizes optimization effort across re-

Table 1. Results with MST-based cascades aggregated by distance family (Feature, Target, Optimization). Values are averaged over distance variants within each family. Lower is better for RMSE and higher is better for accuracy. **Bold** and <u>underlined</u> entries denote best and second-best performance *within each row*.

	Params		Baselines			CTL-MST		
Dataset	T	B	Indiv	Star	CTL-RandTree	Feature	Target	Optim
Synthetic Regression (RMSE)								
Syn-10	200	500	913.8 ± 168.4	845.5 ± 178.8	872.5 ± 253.1	<u>840 ± 215</u>	853 ± 228	$\mathbf{825 \pm 221}$
Syn-10	200	1000	871.5 ± 141.6	802.4 ± 189.5	<u>749.2 ± 171.0</u>	752 ± 188	758 ± 176	$\mathbf{694 \pm 161}$
Syn-10	200	2000	912.8 ± 174.3	930.2 ± 262.4	887.2 ± 246.1	846 ± 228	<u>809 ± 221</u>	$\mathbf{715 \pm 178}$
UK Electricity Forecasting (RMSE, kWh)								
UK	100	500	2729.4 ± 8.8	2247.9 ± 51.8	2395.9 ± 59.4	2391 ± 51	<u>2086 ± 49</u>	$\mathbf{2040 \pm 42}$
UK	100	1000	2694.4 ± 9.5	1820.7 ± 38.7	2042.4 ± 36.4	1993 ± 38	<u>1695 ± 26</u>	$\mathbf{1675 \pm 26}$
UK	100	2000	2612.8 ± 7.3	1504.1 ± 19.0	1612.4 ± 16.3	1612 ± 19	<u>1388 ± 8</u>	$\mathbf{1375 \pm 7}$
Image Classification (Accuracy, %)								
FMNIST	200	500	90.6 ± 5.2	90.5 ± 5.2	91.5 ± 4.1	$\mathbf{93.6 \pm 3.6}$	90.4 ± 4.9	<u>92.8 ± 4.2</u>
FMNIST	200	2000	94.5 ± 3.0	94.4 ± 3.3	94.1 ± 3.2	$\mathbf{95.1 \pm 3.1}$	94.0 ± 3.7	<u>95.1 ± 3.0</u>
CIFAR	200	500	60.9 ± 4.7	64.0 ± 4.3	65.2 ± 3.9	<u>66.0 ± 3.8</u>	64.8 ± 3.8	$\mathbf{66.6 \pm 3.7}$
CIFAR	200	2000	67.7 ± 3.9	66.7 ± 3.8	67.3 ± 3.5	<u>67.9 ± 3.6</u>	67.1 ± 3.9	$\mathbf{68.6 \pm 3.7}$

lated forecasting tasks under tight computational constraints.

Image classification. On Fashion-MNIST, cascaded transfer consistently improves over independent training and shallow baselines across all budgets. Distances based on gradients or task-level feature statistics yield the most stable improvements, particularly in low-budget regimes where cascaded initialization compensates for limited task-specific optimization. In contrast, random hierarchies and star-shaped transfer provide only marginal gains, underscoring the importance of task geometry beyond simple information sharing. On CIFAR–10, improvements are smaller and variability is higher, reflecting increased task difficulty and weaker inter-task alignment. Nevertheless, cascaded transfer remains beneficial: geometry-aware distances consistently outperform independent training and unstructured baselines across most settings. While no single distance dominates uniformly, the results confirm that exploiting task geometry through cascaded transfer yields systematic accuracy gains even in challenging, high-dimensional classification regimes.

Summary. Across regression, forecasting, and classification tasks, the results confirm that organizing transfer through a tree yields robust and consistent improvements over both no-transfer and shallow-transfer baselines. Within cascaded transfer, the choice of task distance is critical: distances aligned with the underlying optimization or target geometry consistently outperform purely feature-based or uninformed measures across datasets and budgets. These findings closely match the theoretical analysis, which predicts that routing transfer through local, geometry-aware cascades mitigates the bias incurred by long-range transfer under constrained budgets.

6. Conclusion and Perspectives

We introduced Cascaded Transfer Learning, a framework for many-task learning under a strict global budget. CTL formulates knowledge transfer as a structured propagation process over a rooted tree, in which tasks are refined sequentially rather than jointly optimized. This design enables controlled information flow across tasks and departs from both independent training and shallow transfer schemes.

We provided a theoretical analysis of error propagation along cascades, establishing sufficient conditions under which routing transfer through intermediate tasks improves over direct transfer. The analysis clarifies how task distances, local optimization stability, and cascade depth jointly govern transfer bias and variance, and applies broadly across parameter and feature-space settings, including noisy regimes. Empirically, CTL consistently outperforms independent and star-shaped baselines on synthetic regression, electricity forecasting, and image classification. Simple geometry-based constructions such as minimum spanning trees emerge as robust and effective defaults, yielding reliable gains under realistic heterogeneity without additional modeling complexity.

Future work includes extending CTL to nonlinear models and deep representations, as well as to more general cascade structures such as directed acyclic graphs. Developing adaptive strategies for allocating refinement budgets based on task difficulty or uncertainty also represents a promising direction. Overall, CTL offers a principled and computationally efficient approach to structured knowledge transfer, and provides a foundation for further research on resource-aware many-task learning.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Aribandi, V., Tay, Y., Schuster, T., Rao, J., Zheng, H. S., Mehta, S. V., Zhuang, H., Tran, V. Q., Bahri, D., Ni, J., et al. ExT5: Towards extreme multi-task scaling for transfer learning. In *International Conference on Learning Representations*, 2022.

Deo, N. and Micikevicius, P. Prufer-like codes for labeled trees. *Congressus Numerantium*, pp. 65–74, 2001.

Hashimoto, K., Xiong, C., Tsuruoka, Y., and Socher, R. A joint many-task model: Growing a neural network for multiple NLP tasks. In *Conference on Empirical Methods in Natural Language Processing*, pp. 1923–1933, 2017.

He, X., Alesiani, F., and Shaker, A. Efficient and scalable multi-task regression on massive number of tasks. In *AAAI Conference on Artificial Intelligence*, volume 33, pp. 3763–3770, 2019.

Kumar, V., Deo, N., and Kumar, N. Parallel generation of random trees and connected graphs. *Congressus Numerantium*, pp. 7–18, 1998.

Liu, J., Xia, Z., Lei, Y., Li, X., and Wang, X. Multi-faceted hierarchical multi-task learning for a large number of tasks with multi-dimensional relations. *Preprint arXiv:2110.13365*, 2021.

Nie, Y., Nguyen, N. H., Sinthong, P., and Kalagnanam, J. A time series is worth 64 words: Long-term forecasting with transformers, 2022.

Pan, S. J. and Yang, Q. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22 (10):1345–1359, 2009.

Ruiz, C., Alaíz, C. M., and Dorronsoro, J. R. A survey on kernel-based multi-task learning. *Neurocomputing*, 577: 127–255, 2024.

Strezoski, G., Noord, N. v., and Worring, M. Many task learning with task routing. In *IEEE/CVF International Conference on Computer Vision*, pp. 1375–1384, 2019.

Tan, Y., Zhang, E., Li, Y., Huang, S.-L., and Zhang, X.-P. Transferability-guided cross-domain cross-task transfer learning. *IEEE Transactions on Neural Networks and Learning Systems*, 2024.

Weiss, K., Khoshgoftaar, T. M., and Wang, D. A survey of transfer learning. *Journal of Big data*, 3(1):9, 2016.

Woo, G., Liu, C., Kumar, A., Xiong, C., Savarese, S., and Sahoo, D. Unified training of universal time series forecasting transformers, 2024.

Zhang, Y. and Yang, Q. An overview of multi-task learning. *National Science Review*, 5(1):30–43, 2018.

Zhang, Y. and Yang, Q. A survey on multi-task learning. *IEEE Transactions on Knowledge and Data Engineering*, 34(12):5586–5609, 2021.

Zhuang, F., Qi, Z., Duan, K., Xi, D., Zhu, Y., Zhu, H., Xiong, H., and He, Q. A comprehensive survey on transfer learning. *Proceedings of the IEEE*, 109(1):43–76, 2020.

A. Proofs for Theoretical Analysis of Cascaded Transfer Learning

A.1. Parameter-Space

Edge-wise propagation in parameter space. By definition of the CTL update, $\tilde{\boldsymbol{\theta}}_v = G_v^{b_v}(\tilde{\boldsymbol{\theta}}_{\mathrm{pa}(v)})$. By the contractive refinement assumption,

$$\|\tilde{\boldsymbol{\theta}}_v - \boldsymbol{\theta}_v^\star\| = \|G_v^{b_v}(\tilde{\boldsymbol{\theta}}_{\mathrm{pa}(v)}) - \boldsymbol{\theta}_v^\star\| \le \rho_v^{b_v}\|\tilde{\boldsymbol{\theta}}_{\mathrm{pa}(v)} - \boldsymbol{\theta}_v^\star\|.$$

Applying the triangle inequality,

$$\|\tilde{\boldsymbol{\theta}}_{\mathrm{pa}(v)} - \boldsymbol{\theta}_v^\star\| \le \|\tilde{\boldsymbol{\theta}}_{\mathrm{pa}(v)} - \boldsymbol{\theta}_{\mathrm{pa}(v)}^\star\| + d(pa(v), v).$$

Combining the two inequalities yields the result. □

Path-wise error decomposition. Applying the above to node v_1:

$$\|\tilde{\boldsymbol{\theta}}_{v_1} - \boldsymbol{\theta}_{v_1}^\star\| \le \rho_{v_1}^{b_{v_1}}\Big(\|\tilde{\boldsymbol{\theta}}_s - \boldsymbol{\theta}_s^\star\| + d(s, v_1)\Big).$$

Same with node v_2 and substitute the previous inequality:

$$\|\tilde{\boldsymbol{\theta}}_{v_2} - \boldsymbol{\theta}_{v_2}^\star\| \le \rho_{v_1}^{b_{v_1}}\rho_{v_2}^{b_{v_2}}\|\tilde{\boldsymbol{\theta}}_s - \boldsymbol{\theta}_s^\star\| + \rho_{v_1}^{b_{v_1}}\rho_{v_2}^{b_{v_2}} d(s, v_1) + \rho_{v_2}^{b_{v_2}} d(v_1, v_2).$$

Proceeding inductively, each application introduces a multiplicative factor $\rho_{v_i}^{b_{v_i}}$ on all upstream terms and adds a new edge term. Hence, we get

$$\|\tilde{\boldsymbol{\theta}}_v - \boldsymbol{\theta}_v^\star\| \le \prod_{i=1}^{m} \rho_{v_i}^{b_{v_i}}\|\tilde{\boldsymbol{\theta}}_s - \boldsymbol{\theta}_s^\star\| + \sum_{i=1}^{m}\rho_{v_i}^{S_{i,m}} d(v_{i-1}, v_i).$$

Setting $\tilde{\boldsymbol{\theta}}_s = \boldsymbol{\theta}_s^\star$ cancels out the bias term, enforcing $b_{v_i} = b$ for all $1 \le i \le m$, plus bounding all ρ_{v_i} by $\rho_{\max}$ and δ_i by $\delta_{\max}$ yields a finite geometric series of finite ratio $\rho_{\max}^b$. □

A.2. Feature-Space

Feature-space contraction. The gradient update is $G_v(\boldsymbol{\theta}) = \boldsymbol{\theta} - \eta\boldsymbol{X}_v^\top(\boldsymbol{X}_v\boldsymbol{\theta} - \boldsymbol{y}_v) = \mathrm{M}_v\boldsymbol{\theta} + \eta\boldsymbol{X}_v^\top\boldsymbol{y}_v$. Since $\boldsymbol{y}_v = \boldsymbol{X}_v\boldsymbol{\theta}_v^\star$ and $\boldsymbol{\theta}_v^\star$ satisfies $\boldsymbol{X}_v^\top\boldsymbol{X}_v\boldsymbol{\theta}_v^\star = \boldsymbol{X}_v^\top\boldsymbol{y}_v$, then $G_v(\boldsymbol{\theta}_v^\star) = \boldsymbol{\theta}_v^\star$, so $\boldsymbol{\theta}_v^\star$ is a fixed point. Thus, $G_v(\boldsymbol{\theta}) - \boldsymbol{\theta}_v^\star = \mathrm{M}_v(\boldsymbol{\theta} - \boldsymbol{\theta}_v^\star)$. Iterating yields the claim. □

Edge-wise propagation in feature space. The above yields $\tilde{\boldsymbol{\theta}}_v - \boldsymbol{\theta}_v^\star = \mathrm{M}_v^{b_v}(\tilde{\boldsymbol{\theta}}_{\mathrm{pa}(v)} - \boldsymbol{\theta}_v^\star)$. Taking norms and using $\|\mathrm{M}_v^{b_v}\| \le \rho_v^{b_v}$,

$$\|\tilde{\boldsymbol{\theta}}_v - \boldsymbol{\theta}_v^\star\| \le \rho_v^{b_v}\|\tilde{\boldsymbol{\theta}}_{\mathrm{pa}(v)} - \boldsymbol{\theta}_v^\star\|.$$

The result follows by the triangle inequality exactly as in the parameter-space case. □

Path-wise error propagation in feature space. The argument follows verbatim from the parameter-space analysis. Unrolling the edge-wise inequality along the unique root-to-v path yields the stated bound, with each edge contribution geometrically damped by downstream refinements. □

A.3. Noisy Feature-Space

Empirical optimum. The minimizer $\hat{\boldsymbol{\theta}}_v$ of $\frac{1}{2}\|\boldsymbol{X}_v\boldsymbol{\theta} - \boldsymbol{y}_v\|^2$ satisfies the equations

$$\boldsymbol{X}_v^\top\boldsymbol{X}_v\hat{\boldsymbol{\theta}}_v = \boldsymbol{X}_v^\top\boldsymbol{y}_v = \boldsymbol{X}_v^\top(\boldsymbol{X}_v\boldsymbol{\theta}_v^\star + \varepsilon_v) = \boldsymbol{X}_v^\top\boldsymbol{X}_v\,\boldsymbol{\theta}_v^\star + \boldsymbol{X}_v^\top\boldsymbol{\varepsilon}_v.$$

Since $\boldsymbol{X}_v^\top\boldsymbol{X}_v \succ 0$,

$$\hat{\boldsymbol{\theta}}_v = \boldsymbol{\theta}_v^\star + (\boldsymbol{X}_v^\top\boldsymbol{X}_v)^{-1}\boldsymbol{X}^\top\boldsymbol{\varepsilon}_v.$$

Moreover, the gradient update is $G_v(\boldsymbol{\theta}) = \mathrm{M}_v\boldsymbol{\theta} + \eta\boldsymbol{X}^\top\boldsymbol{y}_v$, with $\mathrm{M}_v = \mathbf{I}_d - \eta\boldsymbol{X}_v^\top\boldsymbol{X}_v$. Using $\boldsymbol{X}_v^\top\boldsymbol{y}_v = \boldsymbol{X}_v^\top\boldsymbol{X}_v\,\hat{\boldsymbol{\theta}}_v$, we have that $\hat{\boldsymbol{\theta}}_v$ is a fixed point. Hence $G_v(\boldsymbol{\theta}) - \hat{\boldsymbol{\theta}}_v = \mathrm{M}_v(\boldsymbol{\theta} - \hat{\boldsymbol{\theta}}_v)$, and iterating gives $G_v^{b_v}(\boldsymbol{\theta}) - \hat{\boldsymbol{\theta}}_v = \mathrm{M}_v^{b_v}(\boldsymbol{\theta} - \hat{\boldsymbol{\theta}}_v)$. □

Expected estimation error. Let $\mathbf{A}_v = (\boldsymbol{X}_v^\top \boldsymbol{X}_v)^{-1}\boldsymbol{X}_v^\top$. From the empirical optimum lemma, $\hat{\boldsymbol{\theta}}_v - \boldsymbol{\theta}_v^\star = \mathbf{A}_v \boldsymbol{\varepsilon}_v$. Using Jensen's inequality yields $\mathbb{E}\big[\|\mathbf{A}_v\boldsymbol{\varepsilon}_v\|\big] \le \sqrt{\mathbb{E}\big[\|\mathbf{A}_v\boldsymbol{\varepsilon}_v\|^2\big]}$. Now,

$$\|\mathbf{A}_v\boldsymbol{\varepsilon}_v\|^2 = \boldsymbol{\varepsilon}_v^\top \mathbf{A}_v^\top \mathbf{A}_v \boldsymbol{\varepsilon}_v = \sum_{i,j} (\mathbf{A}_v^\top \mathbf{A}_v)_{ij} \varepsilon_{v,i}\varepsilon_{v,j}.$$

Since the coordinates are independent and mean-zero, $\mathbb{E}[\varepsilon_{v,i}\varepsilon_{v,j}] = 0$ for $i \neq j$. Therefore,

$$\mathbb{E}\big[\|\mathbf{A}_v\boldsymbol{\varepsilon}_v\|^2\big] = \sum_i (\mathbf{A}_v^\top \mathbf{A}_v)_{ii}\mathbb{E}[\boldsymbol{\varepsilon}_{v,i}^2] \le \sigma_v^2 \sum_i (\mathbf{A}_v^\top \mathbf{A}_v)_{ii} = \sigma_v^2 \mathrm{tr}(\mathbf{A}_v^\top \mathbf{A}_v) = \sigma_v^2 \|\mathbf{A}_v\|_F^2.$$

Combining yields $\mathbb{E}[\|\hat{\boldsymbol{\theta}}_v - \boldsymbol{\theta}_v^\star\|] \le \sigma_v \|\mathbf{A}_v\|_F$. □

Expected noisy edge-wise propagation. With the contraction assumption for task v, we get

$$\|\tilde{\boldsymbol{\theta}}_v - \hat{\boldsymbol{\theta}}_v\| \le \rho_v^{b_v} \|\tilde{\boldsymbol{\theta}}_u - \hat{\boldsymbol{\theta}}_v\|.$$

Then, adding and subtracting $\boldsymbol{\theta}_v^\star$,

$$\|\tilde{\boldsymbol{\theta}}_u - \hat{\boldsymbol{\theta}}_v\| \le \|\tilde{\boldsymbol{\theta}}_u - \boldsymbol{\theta}_v^\star\| + \|\hat{\boldsymbol{\theta}}_v - \boldsymbol{\theta}_v^\star\| \le \|\tilde{\boldsymbol{\theta}}_u - \boldsymbol{\theta}_u^\star\| + \|\boldsymbol{\theta}_u^\star - \boldsymbol{\theta}_v^\star\| + \|\hat{\boldsymbol{\theta}}_v - \boldsymbol{\theta}_v^\star\|.$$

Therefore,

$$\|\tilde{\boldsymbol{\theta}}_v - \boldsymbol{\theta}_v^\star\| \le \|\tilde{\boldsymbol{\theta}}_v - \hat{\boldsymbol{\theta}}_v\| + \|\hat{\boldsymbol{\theta}}_v - \boldsymbol{\theta}_v^\star\| \le \rho_v^{b_v}\big(\|\tilde{\boldsymbol{\theta}}_u - \boldsymbol{\theta}_u^\star\| + \|\boldsymbol{\theta}_u^\star - \boldsymbol{\theta}_v^\star\| + \|\hat{\boldsymbol{\theta}}_v - \boldsymbol{\theta}_v^\star\|\big) + \|\hat{\boldsymbol{\theta}}_v - \boldsymbol{\theta}_v^\star\|.$$

Taking expectations and using $\mathbb{E}\left[\|\hat{\boldsymbol{\theta}}_v - \boldsymbol{\theta}_v^\star\|\right] \le \sigma_v \|\mathbf{A}_v\|_F$ gives

$$\mathbb{E}\left[\|\tilde{\boldsymbol{\theta}}_v - \boldsymbol{\theta}_v^\star\|\right] \le \rho_v^{b_v}\Big(\mathbb{E}\left[\|\tilde{\boldsymbol{\theta}}_u - \boldsymbol{\theta}_u^\star\|\right] + \|\boldsymbol{\theta}_u^\star - \boldsymbol{\theta}_v^\star\|\Big) + \sigma_v(1 + \rho_v^{b_v})\|\mathbf{A}_v\|_F,$$

as claimed. □

Expected path-wise propagation. Let $(v_0 = s \to v_1 \to \cdots \to v_m = v)$ denote the unique root-to-v path. By the expected noisy edge-wise propagation lemma, for each $i \ge 1$,

$$\mathbb{E}\big[\|\tilde{\boldsymbol{\theta}}_{v_i} - \boldsymbol{\theta}_{v_i}^\star\|\big] \le \rho_{v_i}^{b_{v_i}}\big(\mathbb{E}\big[\|\tilde{\boldsymbol{\theta}}_{v_{i-1}} - \boldsymbol{\theta}_{v_{i-1}}^\star\|\big] + \|\boldsymbol{\theta}_{v_{i-1}}^\star - \boldsymbol{\theta}_{v_i}^\star\|\big) + \sigma_{v_i}(1 + \rho_{v_i}^{b_{v_i}})\|\mathbf{A}_{v_i}\|_F.$$

Applying this inequality to v_1 yields

$$\mathbb{E}\big[\|\tilde{\boldsymbol{\theta}}_{v_1} - \boldsymbol{\theta}_{v_1}^\star\|\big] \le \rho_{v_1}^{b_{v_1}}\Big(\mathbb{E}\big[\|\tilde{\boldsymbol{\theta}}_s - \boldsymbol{\theta}_s^\star\|\big] + \|\boldsymbol{\theta}_s^\star - \boldsymbol{\theta}_{v_1}^\star\|\Big) + \sigma_{v_1}(1 + \rho_{v_1}^{b_{v_1}})\|\mathbf{A}_{v_1}\|_F.$$

Applying the same inequality to v_2 and bounding the expectation term,

$$\begin{aligned}\mathbb{E}\big[\|\tilde{\boldsymbol{\theta}}_{v_2} - \boldsymbol{\theta}_{v_2}^\star\|\big] \le \rho_{v_2}^{b_{v_2}}\rho_{v_1}^{b_{v_1}}\mathbb{E}\big[\|\tilde{\boldsymbol{\theta}}_s - \boldsymbol{\theta}_s^\star\|\big] + \rho_{v_2}^{b_{v_2}}\rho_{v_1}^{b_{v_1}}\|\boldsymbol{\theta}_s^\star - \boldsymbol{\theta}_{v_1}^\star\| + \rho_{v_2}^{b_{v_2}}\|\boldsymbol{\theta}_{v_1}^\star - \boldsymbol{\theta}_{v_2}^\star\| + \rho_{v_2}^{b_{v_2}}\sigma_{v_1}(1 + \rho_{v_1}^{b_{v_1}})\|\mathbf{A}_{v_1}\|_F \\ + \sigma_{v_2}(1 + \rho_{v_2}^{b_{v_2}})\|\mathbf{A}_{v_2}\|_F.\end{aligned}$$

Proceeding inductively, each step multiplies all upstream terms by $\rho_{v_i}^{b_{v_i}}$ and adds a new distance term discounted by downstream contractions, together with anew noise contribution discounted only by refinements performed after node v_i.

Collecting terms yields

$$\begin{aligned}\mathbb{E}\big[\|\tilde{\boldsymbol{\theta}}_v - \boldsymbol{\theta}_v^\star\|\big] \le \Big(\prod_{j=1}^m \rho_{v_j}^{b_{v_j}}\Big)\mathbb{E}\big[\|\tilde{\boldsymbol{\theta}}_s - \boldsymbol{\theta}_s^\star\|\big] + \sum_{i=1}^m \Big(\prod_{j=i}^m \rho_{v_j}^{b_{v_j}}\Big)\|\boldsymbol{\theta}_{v_{i-1}}^\star - \boldsymbol{\theta}_{v_i}^\star\| \\ + \sum_{i=1}^m \Big(\prod_{j=i+1}^m \rho_{v_j}^{b_{v_j}}\Big)\sigma_{v_i}(1 + \rho_{v_i}^{b_{v_i}})\|\mathbf{A}_{v_i}\|_F,\end{aligned}$$

with the convention that empty products equal 1. This concludes the proof. □

Algorithm 2 CTL Adaptation

```
1: Input: task observations $\{\boldsymbol{y}_v\}_{v\in\mathcal{V}}$, optional cluster count $n_{clust}$, clustering routine findClusters(), seed selection routine
   findSeed(), seed-training budget $b$, per-cluster adaptation budget $B$, training operators $\mathcal{A}_0$ (seed training) and $\mathcal{A}_1$ (task adaptation).
2: Output: adapted models $\{f_{\widetilde{\boldsymbol{\theta}}_v}\}_{v\in\mathcal{V}}$
3: ■ (Optional) Cluster the tasks.
4: if $K$ is specified then
5:     $\{C_k\}_{k=1}^{n_{clust}} \leftarrow$ findClusters($\{\boldsymbol{y}_v\}_{v\in\mathcal{V}}, n_{clust}$)
6: else
7:     $K \leftarrow 1$;   $C_1 \leftarrow \mathcal{V}$
8: end if
9: ■ Process each cluster independently
10: for each cluster $C$ do
11:     $s \leftarrow$ findSeed($C$)                                        ▷ Select cluster seed
12:     $(\mathcal{T}, E) \leftarrow$ computeTree($C, s$)
13:     ■ Allocate adaptation budgets along the tree
14:     $\mathbf{w} \leftarrow$ extractTreeWeights($\mathcal{T}, E$)
15:     $\mathbf{w}' \leftarrow$ normalize($\mathbf{w}$)
16:     for $v \in C$ do
17:         $b_v \leftarrow \lceil w'_v \cdot B \rceil$
18:     end for
19:     ■ Train the seed model
20:     $f_{\widetilde{\boldsymbol{\theta}}_s} \leftarrow \mathcal{A}_0(f_{\boldsymbol{\theta}}, \boldsymbol{y}_s, b)$
21:     ■ Hierarchical model adaptation
22:     Initialize queue $\mathcal{Q} \leftarrow [\,s\,]$
23:     while $\neg$ isEmpty($\mathcal{Q}$) do
24:         $u \leftarrow$ dequeue($\mathcal{Q}$)
25:         for each child $v$ of $u$ in $(\mathcal{T}, E)$ do
26:             $\mathcal{Q} \leftarrow$ enqueue($\mathcal{Q}, v$)
27:             $f_{\widetilde{\boldsymbol{\theta}}_v} \leftarrow \mathcal{A}_1(f_{\widetilde{\boldsymbol{\theta}}_u}, \boldsymbol{y}_v, b_v)$        ▷ Adapt task $v$ from its parent's model
28:         end for
29:     end while
30: end for
31: return $\{f_{\widetilde{\boldsymbol{\theta}}_v}\}_{v\in\mathcal{V}}$.
```

B. Cascaded Adaptation Algorithm

Algorithm 2 implements the cascaded adaptation procedure used throughout the paper. The algorithm performs hierarchical task adaptation under a global optimization budget by propagating information along a directed tree structure. Tasks may optionally be partitioned into clusters to limit transfer across distant regimes and reduce the risk of negative transfer. Within each cluster, a tree-structured cascade is constructed, and adaptation proceeds from a designated seed task toward downstream tasks. The seed task is trained from scratch, while each remaining task is adapted exactly once from its parent in the tree.

Adaptation follows a top-down traversal of the tree, ensuring that parent tasks are fully adapted before their children. This enforces an acyclic flow of information and prevents backward or joint updates, which is consistent with the cascade analysis developed in the main text. The specific form of the local adaptation operator is left abstract, allowing instantiation with linear or nonlinear models, provided that adaptation remains task-local. A fixed global budget B is allocated across tasks using normalized tree-based weights. While the choice of weighting scheme is flexible, it reflects the intuition that tasks deeper in the cascade or further from the seed may require additional adaptation effort. In practice, discretization effects may introduce a small budget slack, which is negligible relative to B and does not affect the qualitative behavior of the algorithm.

Overall, this procedure provides a practical realization of cascaded transfer learning, emphasizing budget awareness, stability, and modularity in large collections of related tasks.

C. Distance metrics for MST-based cascades

We group the task distance metrics used to construct MST-based cascades into the following families:

Table 2. Unified experimental results across all datasets. **Synthetic regression**: Test RMSE. **UK electricity**: Test RMSE (kWh). **Image classification**: Accuracy (%). Each configuration is evaluated across 50 simulations. **Bold** and $\underline{\text{underlined}}$ entries denote best and second-best performance *within each row*.

Params			Baselines			CTL-MST variants									
Dataset	T	B	Indiv	Star	CTL-RandTree	Feature	Target	Gradient	Model	KL	Wasserstein	MMD	MeanCov	JS	CKA
Synthetic Regression (RMSE)															
Syn-2	200	500	494.2 ± 149.9	440.4 ± 155.9	455.4 ± 192.2	447.9 ± 166.0	445.5 ± 169.1	432.4 ± 153.9	**407.7 ± 143.3**	440.3 ± 154.3	453.7 ± 156.8	$\underline{417.4 \pm 130.1}$	460.8 ± 197.7	485.6 ± 254.9	458.4 ± 203.1
Syn-2	200	1000	458.9 ± 134.2	382.0 ± 139.1	386.1 ± 146.5	363.2 ± 131.7	373.3 ± 121.2	368.8 ± 143.2	$\underline{356.9 \pm 151.3}$	369.5 ± 132.6	412.8 ± 150.7	400.5 ± 156.9	**355.6 ± 130.9**	382.7 ± 114.6	385.5 ± 171.9
Syn-2	200	2000	499.2 ± 178.8	451.6 ± 181.9	409.8 ± 162.5	448.8 ± 187.7	418.4 ± 152.5	$\underline{370.1 \pm 163.8}$	**347.6 ± 144.2**	421.5 ± 181.3	433.0 ± 178.8	428.3 ± 181.5	429.0 ± 191.7	457.6 ± 215.2	403.9 ± 167.0
Syn-5	200	500	595.9 ± 166.3	547.8 ± 224.6	549.4 ± 164.9	528.9 ± 209.2	**522.1 ± 174.2**	$\underline{525.7 \pm 176.0}$	532.0 ± 165.3	543.5 ± 198.8	530.4 ± 173.5	546.5 ± 181.0	537.9 ± 189.4	546.0 ± 171.0	547.8 ± 187.3
Syn-5	200	1000	550.6 ± 130.6	**434.7 ± 142.6**	460.1 ± 136.7	449.8 ± 149.6	450.7 ± 158.0	441.4 ± 150.5	444.5 ± 150.6	460.4 ± 154.9	472.3 ± 164.9	476.2 ± 151.7	466.9 ± 144.3	$\underline{437.3 \pm 149.5}$	472.9 ± 173.6
Syn-5	200	2000	595.0 ± 185.7	541.2 ± 187.1	520.6 ± 204.2	517.9 ± 182.0	507.8 ± 190.2	**437.4 ± 152.5**	$\underline{447.1 \pm 148.3}$	511.0 ± 176.4	517.7 ± 184.0	543.5 ± 213.1	517.3 ± 194.4	517.3 ± 184.0	545.2 ± 199.5
Syn-10	200	500	913.8 ± 168.4	845.5 ± 178.8	872.5 ± 253.1	835.9 ± 189.5	857.8 ± 215.5	844.7 ± 254.5	$\underline{805.2 \pm 187.8}$	891.9 ± 262.6	860.7 ± 257.7	823.8 ± 227.4	855.7 ± 228.6	**800.4 ± 175.3**	845.9 ± 213.5
Syn-10	200	1000	871.5 ± 141.6	802.4 ± 189.5	749.2 ± 171.0	740.1 ± 200.6	768.8 ± 189.2	$\underline{711.8 \pm 169.3}$	**675.2 ± 152.7**	765.2 ± 186.3	749.1 ± 154.2	760.8 ± 195.9	753.8 ± 178.2	747.8 ± 172.4	752.1 ± 178.2
Syn-10	200	2000	912.8 ± 174.3	930.2 ± 262.4	887.2 ± 246.1	830.6 ± 218.4	814.4 ± 200.9	$\underline{722.4 \pm 175.8}$	**707.2 ± 180.1**	808.1 ± 218.3	823.9 ± 243.2	863.1 ± 222.9	858.5 ± 239.2	790.2 ± 220.9	832.2 ± 231.4
UK Electricity Forecasting (RMSE, kWh)															
UK	100	500	2729.4 ± 8.8	2247.9 ± 51.8	2395.9 ± 59.4	2420.9 ± 45.3	2156.2 ± 51.8	**1990.6 ± 41.8**	2089.7 ± 42.2	2042.8 ± 52.1	$\underline{2032.3 \pm 48.6}$	2359.4 ± 54.3	2425.8 ± 55.9	2113.3 ± 43.1	2356.5 ± 46.9
UK	100	1000	2694.4 ± 9.5	1820.7 ± 38.7	2042.4 ± 36.4	1973.4 ± 39.3	1749.8 ± 29.8	$\underline{1634.5 \pm 22.9}$	1715.5 ± 29.4	1709.2 ± 27.4	**1623.2 ± 21.8**	2032.1 ± 37.8	1976.7 ± 34.4	1698.6 ± 26.6	1989.9 ± 39.3
UK	100	2000	2612.8 ± 7.3	1504.1 ± 19.0	1612.4 ± 16.3	1604.8 ± 16.9	1412.9 ± 10.0	**1354.9 ± 6.8**	1395.9 ± 7.9	1377.1 ± 6.9	$\underline{1363.1 \pm 6.1}$	1583.4 ± 18.9	1620.7 ± 19.5	1398.2 ± 10.7	1637.9 ± 19.4
Image Classification (Accuracy, %)															
FMNIST	200	500	90.6 ± 5.2	90.5 ± 5.2	91.5 ± 4.1	93.5 ± 3.6	91.8 ± 4.0	$\underline{94.5 \pm 3.4}$	91.1 ± 5.0	89.9 ± 5.2	89.9 ± 5.2	94.1 ± 3.6	**94.7 ± 3.2**	89.9 ± 5.2	92.0 ± 4.0
FMNIST	200	2000	94.5 ± 3.0	94.4 ± 3.3	94.1 ± 3.2	95.1 ± 3.0	94.2 ± 3.0	$\underline{95.7 \pm 2.8}$	94.4 ± 3.3	94.0 ± 3.9	94.0 ± 3.9	95.2 ± 3.5	**95.8 ± 2.8**	94.0 ± 3.9	94.2 ± 3.2
CIFAR	200	500	60.9 ± 4.7	64.0 ± 4.3	65.2 ± 3.9	65.5 ± 3.8	65.4 ± 3.8	**67.7 ± 3.7**	65.4 ± 3.6	64.6 ± 3.8	64.6 ± 3.8	65.5 ± 3.4	$\underline{67.5 \pm 4.0}$	64.6 ± 3.8	65.4 ± 3.9
CIFAR	200	2000	67.7 ± 3.9	66.7 ± 3.8	67.3 ± 3.5	67.4 ± 3.7	67.3 ± 3.4	**69.7 ± 3.7**	67.5 ± 3.6	67.0 ± 4.1	67.0 ± 4.1	67.7 ± 3.7	$\underline{69.1 \pm 3.6}$	67.0 ± 4.1	67.5 ± 3.5

- **Feature-based distances.** These distances compare task feature representations $\boldsymbol{X}_v$. We consider: (i) a normalized Euclidean distance on flattened feature matrices when shapes are consistent, or a mean-variance embedding when shapes differ (*feature distance*); (ii) a kernel-based Maximum Mean Discrepancy (MMD) with RBF kernel and median bandwidth heuristic; (iii) a Gaussian approximation using differences in feature means and covariances; and (iv) a representation-level distance based on linear Centered Kernel Alignment (CKA).

- **Target-based distances.** These distances compare task outputs $\boldsymbol{y}_v$ directly. We use Euclidean distances between flattened targets (*target distance*), the symmetric KL divergence computed from smoothed histograms, the Jensen-Shannon distance, and the 1-Wasserstein (Earth Mover's) distance between empirical target distributions.

- **Optimization-geometry distances.** These distances are derived from optimization or model parameters. We consider a gradient-at-initialization proxy $g_v = \boldsymbol{X}_v^\top \boldsymbol{y}_v$ (*gradient distance*) and a distance based on ridge-regression solutions (*model distance*), the latter serving more as a diagnostic rather than as an off-the-shelf method.

D. Additional Experimental Results

This section provides complementary visualizations and detailed numerical results that support the main experimental findings. We report extended results for the synthetic regression benchmark and the UK electricity forecasting task experiments.

D.1. Synthetic Regression

Figure 3 illustrates the geometry of the synthetic tasks. Increasing the within-cluster variance τ_{within} progressively disperses task parameters while preserving the same between-cluster separation, resulting in a controlled degradation of task similarity.

Figure 4 provides a visual summary of the same sweep, highlighting the relative performance of cascaded transfer strategies across regimes.

D.2. UK Electricity Forecasting

Figure 5 provides additional context on the UK electricity benchmark. The spatial distribution of the 100 sampled smart meters shows broad geographic coverage, while the consumption histogram highlights a strongly skewed demand distribution with a heavy right tail.

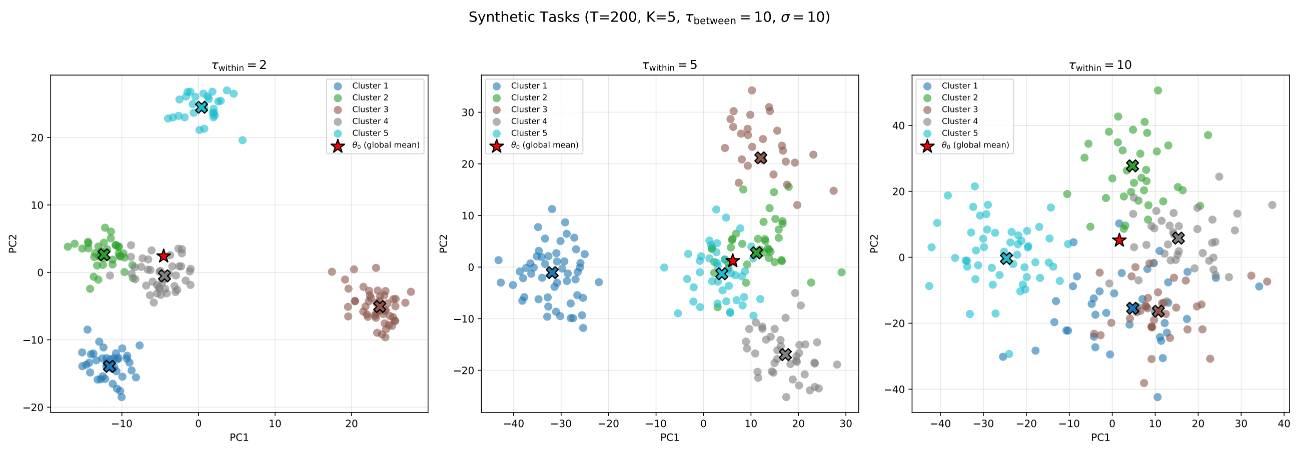


Figure 3. Two-dimensional PCA projection of task parameters for the synthetic data with increasing within-cluster variance τ_{within}.

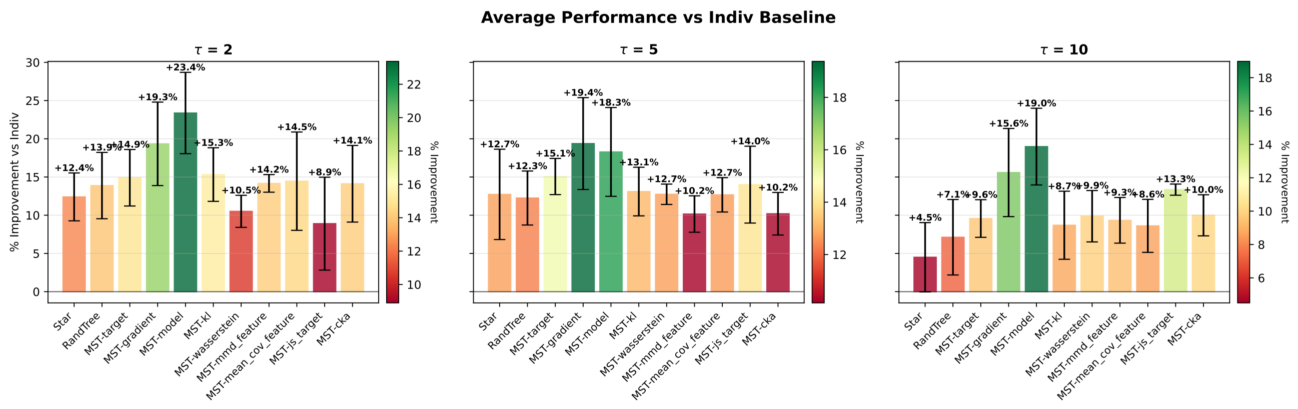
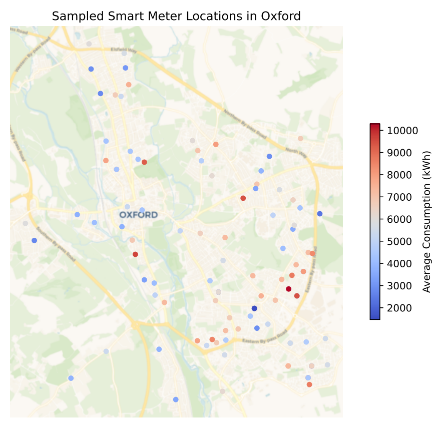
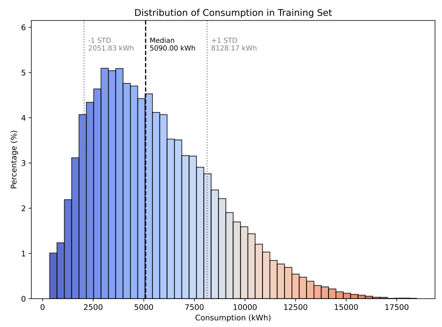


Figure 4. Test RMSE for synthetic tasks across within-cluster variability τ, number of tasks T, and global budgets B.


Sampled Smart Meter Locations in Oxford
OXFORD
Average Consumption (kWh)

(a) Spatial distribution of the sampled meters.



(b) Distribution of yearly electricity consumption.

Figure 5. Overview of spatial positions and consumption distribution of the sampled smart meters in Oxford's urban area.

E. Budget Allocation Ablation Study

This section investigates the impact of the *budget allocation strategy* in CTL. While all main experiments use a uniform allocation of the global budget across tasks, the theoretical analysis in Section 3 suggests that allocating more refinement effort to certain tasks (e.g., deeper nodes in the cascade or tasks connected by longer edges) may further improve performance. We empirically evaluate this question in a controlled synthetic setting.

CTL configuration. We evaluate CTL with minimum spanning tree construction using the gradient-based distance. The seed task is selected using the same procedure as in the main experiments, and the cascade is oriented away from the seed. Models are linear regressors trained by gradient descent. The global budget B is measured in gradient steps and is varied over $B \in \{500, 1000, 2000\}$. A fixed fraction (10%) of the budget is allocated to the seed task, and the remaining budget is distributed across non-root tasks according to the allocation scheme under consideration. Each task is refined exactly once.

Budget allocation schemes. Let $\mathrm{depth}(v)$ denote the depth of node v in the cascade tree, and let $d(u, v)$ denote the edge length between a task and its parent. We compare the following allocation strategies:

- **Uniform:** all non-root tasks receive equal budget.
- **Depth-increasing:** $b_v \propto (\mathrm{depth}(v) + 1)^{\alpha}$, allocating more budget to deeper tasks.
- **Depth-decreasing:** $b_v \propto (\mathrm{depth}(v) + 1)^{-\alpha}$, favoring tasks closer to the root.
- **Edge-length-based:** $b_v \propto (d(\mathrm{pa}(v), v) + \varepsilon)^{\beta}$, allocating more budget to tasks whose transfer edge is longer.

We consider $\alpha \in \{1, 2\}$ and $\beta \in \{1, 2\}$. In all cases, real-valued allocations are converted to integer budgets using a largest-remainder scheme, ensuring that $\sum_v b_v = B$ exactly.

Evaluation protocol. For each combination of $(\tau_{\text{within}}, B)$ and allocation scheme, we repeat the experiment over 50 independent random draws of task parameters and data. Performance is measured by the RMSE, averaged across tasks. We report mean and standard deviation over repetitions. Identical trees, seeds, learning rates, and optimization procedures are used across all allocation schemes to ensure a fair comparison.

Results. Figure 6 reports the effect of different budget allocation strategies for CTL-MST-gradient across increasing task heterogeneity (τ_{within}) and total budgets $B \in \{500, 1000, 2000\}$. Across all regimes, uniform allocation provides a strong and reliable baseline and is never severely suboptimal. In low and moderate heterogeneity settings, uniform allocation often lies within one standard deviation of the best-performing method, particularly at larger budgets, indicating that CTL is not overly sensitive to the precise allocation rule.

At smaller budgets ($B = 500$), structure-aware allocations become more influential. In several low-heterogeneity configurations, edge-length-based allocation achieves the lowest RMSE, suggesting that allocating additional refinement steps to tasks connected by longer transfer edges helps mitigate transfer bias when optimization resources are scarce. Depth-increasing allocations occasionally improve performance at intermediate budgets by compensating for accumulated upstream error at deeper nodes, whereas depth-decreasing allocations generally underperform as task heterogeneity increases, indicating that prioritizing early tasks alone is insufficient.

In the high-heterogeneity regime ($\tau_{\text{within}} = 10$), performance differences between allocation schemes narrow and variance increases. While certain structure-aware schemes still achieve the best average RMSE in individual configurations, gains over uniform allocation remain modest, suggesting that budget reallocation cannot fully compensate for weak task relatedness.

Overall, these results support uniform allocation as a robust default, while confirming that simple structure-aware schemes can provide additional gains at no extra computational cost in low-budget and low-to-moderate heterogeneity regimes, in agreement with the theoretical analysis.

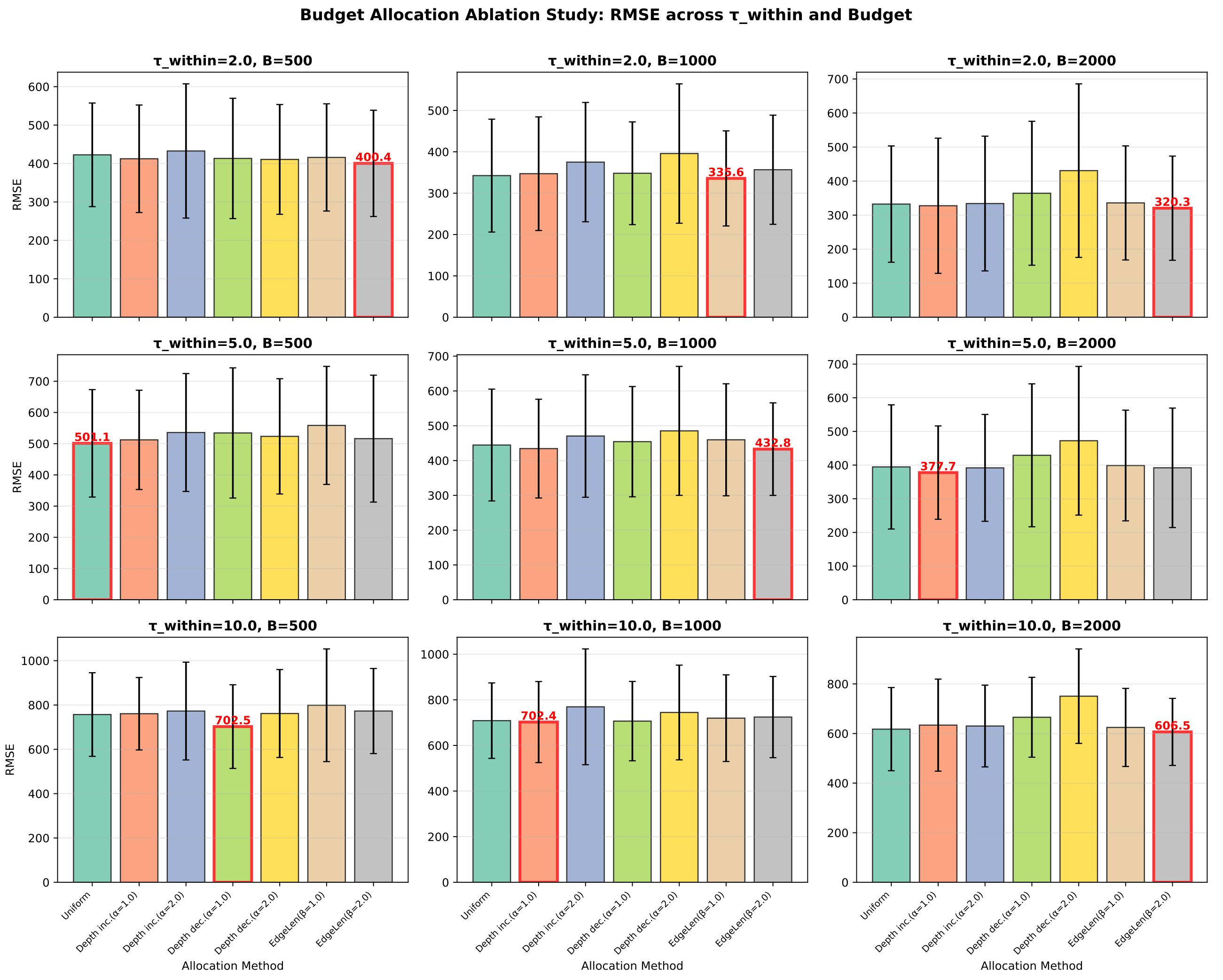


Figure 6. Budget allocation ablation for CTL-MST-gradient on clustered synthetic regression tasks. Each subplot shows mean test RMSE ($\pm$ one standard deviation over 50 runs) for a given τ_{within} and total budget B. Bars correspond to different budget allocation schemes, with the best-performing method highlighted.