WHEN AND WHERE TO TRANSFER: LEARNING TRANSFERABLE HISTORICAL CONTEXTS FOR SHORT-HISTORY MULTIVARIATE TIME SERIES FORECASTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Related sources can improve multivariate time series forecasting when the target history is short, but their usefulness varies across source groups and historical periods. A key challenge is to identify informative historical contexts from the sources. This paper proposes Hierarchical Context Transfer (HiCT), a framework that adaptively weights source groups and their historical contexts to determine both *where* and *when* to transfer. A shared context encoder and cross-attention are used to learn weights over source groups and historical contexts. These weights guide regularized transfer estimation, followed by an optional target-specific correction. This paper establishes a target excess-risk bound for the context-weighted transfer estimator. Experiments demonstrate competitive predictive performance across four multivariate forecasting datasets and a favorable accuracy–runtime trade-off against the evaluated baselines. Source-corruption ablations illustrate the roles of group-level and within-group weighting in adapting to unreliable source histories.

1 INTRODUCTION

Multivariate time series forecasting often involves a target group with a short historical record and related source groups with substantially longer histories. Limited target observations make it difficult to estimate temporal and cross-variable dependencies reliably. Transfer learning offers a way to incorporate source information into target model estimation (Li et al., 2022a; Ma & Safikhani, 2025). Its benefit depends on whether the transferred information matches the target's current dynamics.

This match can vary across source groups and across historical periods within each group. A source group may share predictive structure with the target yet contain periods that are poorly suited to the current forecast. A single weight for an entire source group cannot capture these differences in usefulness. This motivates two related decisions: how much to borrow from each source group and which historical contexts within that group to emphasize. Both decisions should depend on the current target context.

With limited target observations, we need a criterion that connects source contributions to target prediction. Historical similarity can guide source selection (Teller et al., 2026), but does not directly measure whether a source context improves target prediction after model fitting. Validation-based weighting learns sample contributions from held-out prediction loss (Ren et al., 2018). For temporal adaptation, AdaRNN partitions historical data into periods and matches the distributions of RNN hidden-state representations across them to address temporal covariate shift (Du et al., 2021), while DDG-DA learns resampling probabilities over historical observations to adapt training data to predicted distribution changes (Li et al., 2022b). Building on prediction-based weighting, we use target prediction loss to learn source contributions across groups and historical contexts.

We propose *Hierarchical Context Transfer* (HiCT), a framework that learns *where* and *when* to transfer for each target forecast. Conditioned on the current target context, group-level weights specify how much to borrow from each source group and within-group weights distribute this contribution across the group's historical contexts. Their product gives each source context's weight in the forecasting loss. Aligned low-dimensional representations and a shared context encoder enable transferability scoring across groups with different observation dimensions.

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

Target validation loss guides the learning of a shared weighting function and hyperparameter selection. Given the current weights, we fit the predictor using target observations and weighted source contexts, then differentiate validation loss through this fit to update the weighting parameters. At test time, the weighting parameters remain fixed. In the rolling-refit implementation, the function generates source weights conditioned on each new target context. These weights guide regularized transfer estimation, followed by an optional target-specific correction using the available target history to address residual source–target mismatch.

Our contributions are summarized below.

- We propose HiCT for multivariate forecasting with short target histories. It learns *where* and *when* to transfer through target-conditioned weights over sources and their historical contexts.
- We develop a two-stage estimator that combines regularized transfer estimation with target-specific correction. We derive a target excess-risk bound for Stage 1 and show that exact Stage 2 correction does not increase empirical target loss.
- We conduct experiments on four multivariate forecasting datasets, showing that HiCT offers strong predictive performance at modest computational cost. Source-corruption ablations further illustrate the respective contributions of group-level and within-group weighting.

2 Related Work

Selective statistical transfer. Trans-Lasso combines sparse transfer estimation with a target-specific correction (Li et al., 2022a). Park et al. (2026) select informative sources for low-rank multivariate regression. RIW-TL assigns observation-level weights using residual-based density ratios (Zhao et al., 2026). Ma & Safikhani (2025) develop transfer estimators for high-dimensional vector autoregressive models. Their method combines low-rank and sparse components and selects informative auxiliary observations.

Representation learning and time-series transfer. Target-PCA uses auxiliary panels to improve target factor estimation under missing observations and weak signals (Duan et al., 2024). Zhou et al. (2023) treat historical batches as multiple source domains. They align source and target feature distributions for forecasting with limited target data. DTr-CNN selects a single source using dynamic time warping and Jensen–Shannon divergence and transfers a causal CNN across datasets, with competitive performance on financial forecasting tasks (Ye & Dai, 2021). ROSE combines a Transformer architecture with decomposed frequency learning, and register tokens to learn from multiple pre-training domains and adapt to downstream tasks (Wang et al., 2025). These methods focus on transfer across datasets or domains, without explicitly learning target-conditioned weights over historical contexts within each source group to guide forecasting-model estimation.

Historical selection and retrieval for forecasting. Teller et al. (2026) select source subsequences using dynamic time warping distances. These subsequences are combined with target data to train realized-volatility forecasting models. RAFT retrieves historical windows and uses their observed continuations to augment predictions (Han et al., 2025). TimeRAF learns a retriever for a forecasting foundation model (Zhang et al., 2025). SARAF uses stationarity information to regulate retrieval diversity and aggregation (Zhou et al., 2026). Choi et al. (2026) learn to rerank historical candidates using future-compatibility supervision during training.

Validation-based weighting and temporal adaptation. Ren et al. (2018) obtain sample weights by differentiating validation loss through a training update. Meta-Weight-Net learns a mapping from individual training loss to sample weight using a small unbiased meta-dataset (Shu et al., 2019). RAM-Net incorporates market regimes into meta-learned sample weighting (Jang et al., 2025). AdaRNN weights hidden-state distribution matching across periods (Du et al., 2021), whereas HiCT's weights determine each source context's contribution to the forecasting loss. DDG-DA learns historical-sample resampling probabilities through bilevel optimization with a weighted forecasting proxy (Li et al., 2022b). HiCT instead learns a shared function from one-step-ahead target validation loss to assign source-group and historical-context weights for each target context.

3 LEARNING TRANSFERABLE HISTORICAL CONTEXTS

HiCT learns target-conditioned weights over source groups and their historical contexts as showed in Figure 1. The parameters $\boldsymbol{\Theta}$ define the weight-generating function, whose outputs π_{ks} depend on the target context; $\mathbf{W}$ contains the prediction parameters in the weighted forecasting loss. Learning alternates predictor fitting with validation-based updates of $\boldsymbol{\Theta}$. At test time, $\widehat{\boldsymbol{\Theta}}$ is fixed, while the rolling-refit procedure regenerates weights and fits prediction parameters for each target context, with an optional target-specific correction.

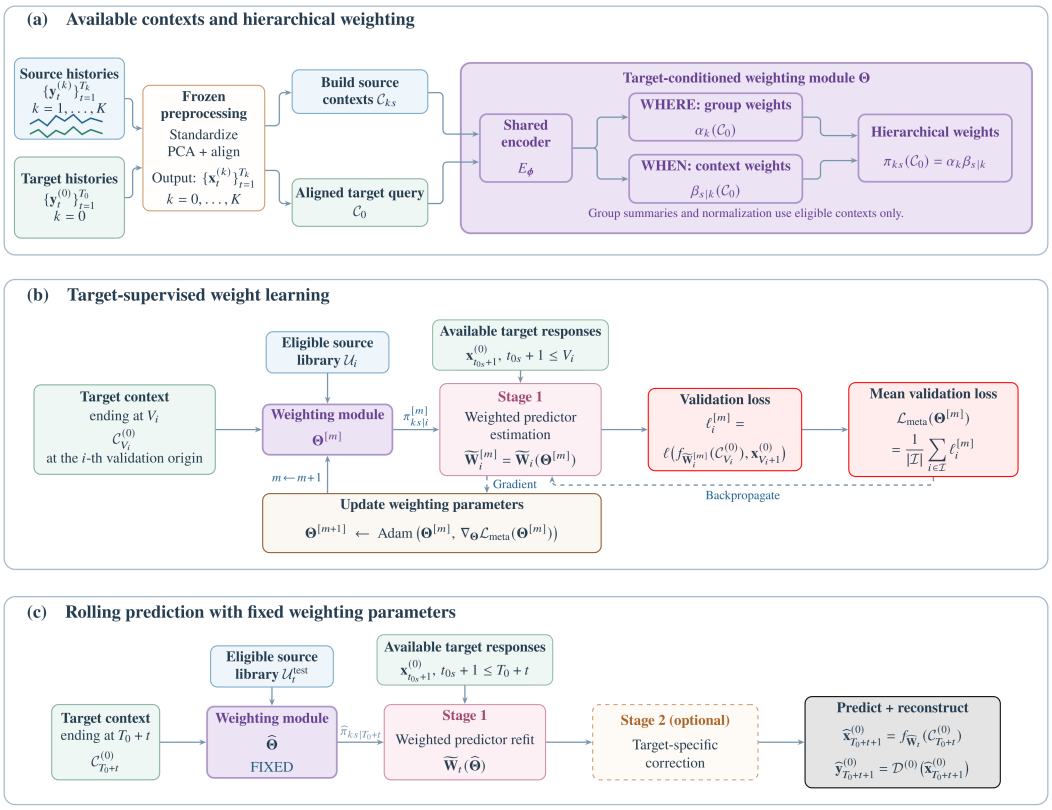

Figure 1: Overview of HiCT. (a) Aligned contexts receive target-conditioned group and within-group weights. (b) Weighting parameters are learned by differentiating the mean loss over all target validation queries through weighted predictor fits. (c) With these parameters fixed, each test context generates new weights for predictor refitting, optional target-specific correction, and one-step-ahead forecast reconstruction.

3.1 PROBLEM SETUP AND GROUP-TIME CONTEXTS

We consider one target group, indexed by $k = 0$, and K source groups, indexed by $k = 1, \ldots, K$. For each group k, let N_k denote its observation dimension. The observations from group k form a multivariate time series.

$$\mathbf{y}_t^{(k)} = \left(y_{1t}^{(k)}, \ldots, y_{N_k t}^{(k)}\right)^\top \in \mathbb{R}^{N_k}, \qquad t = 1, \ldots, T_k,$$

where T_k is the number of observed time points in group k and $y_{it}^{(k)}$ denotes the value of variable i at time t. The superscript (k) identifies the group, with (0) denoting the target group. We use t as a within-group observation index and write $\tau_t^{(k)}$ for its calendar time. We focus on the short-history setting, where the target history is substantially shorter than every source history. Specifically, we assume $T_0 \ll T_k$ for all $k = 1, \ldots, K$.

Since the group dimensions N_k may differ, we map each multivariate series to an aligned d-dimensional representation $\mathbf{x}_t^{(k)} \in \mathbb{R}^d$, as described in Section 3.2. Given the observed target history, we aim to forecast $\mathbf{y}_{T_0+1}^{(0)}$ by exploiting transferable information from the source groups. A single weight for each source group cannot distinguish the transferability of different historical contexts within that group. We therefore model transferability at both the group and temporal levels to learn *where* and *when* to transfer.

Historical contexts. To represent variation in transferability over time, we construct rolling historical contexts of length L with step size h. We take L and h to be positive integers and assume $T_k \geq L+1$ for all groups. For each group $k = 0, \ldots, K$, the context ending at an observed time $t \geq L$ is

$$\mathcal{C}_t^{(k)} = \left(\mathbf{x}_{t-L+1}^{(k)}, \ldots, \mathbf{x}_t^{(k)}\right)^\top \in \mathbb{R}^{L \times d}.$$

We sample historical contexts at $t_{ks} = L + (s-1)h$, $s = 1, \ldots, S_k$, where $S_k = 1 + \lfloor (T_k - L - 1)/h \rfloor$ is the number of context–response pairs in group k. For brevity, we write $\mathcal{C}_{ks} = \mathcal{C}_{t_{ks}}^{(k)}$. The index s identifies the sampled context, and t_{ks} is its ending time. Since $t_{ks}+1 \leq T_k$, each context $\mathcal{C}_{ks}$ has an observed one-step-ahead response $\mathbf{x}_{t_{ks}+1}^{(k)}$. For source groups $k = 1, \ldots, K$, this context–response pair forms a group–time transfer unit indexed by (k, s), with k and s specifying *where* and *when* to transfer from, respectively. At the current forecasting origin T_0, we write $\mathcal{C}_0 = \mathcal{C}_{T_0}^{(0)}$ for the target query context used to forecast $\mathbf{x}_{T_0+1}^{(0)}$.

In Figure 1(b), V_i denotes the forecasting-origin index of validation query i, whose context is $\mathcal{C}_{V_i}^{(0)}$. Section 3.4 defines the validation schedule. In panel (c), rolling test prediction starts at T_0. The target context at test step $t = 0, 1, \ldots$ is $\mathcal{C}_{T_0+t}^{(0)}$. We write $\mathcal{U}_t^{\text{test}}$ for the library of source contexts whose one-step-ahead responses are available at this forecasting origin. As new target observations become available, we extend the same sampling grid and include historical target samples whose responses satisfy $t_{0s} + 1 \leq T_0 + t$.

3.2 Hierarchical Transferability Weighting

To score contexts across groups with different numbers of observed variables, we first construct aligned d-dimensional representations. A shared Transformer encoder then maps the aligned contexts to features used by the group-level and within-group context weighting branches.

Aligned representations. For each group $k = 0, \ldots, K$, the map $\mathcal{P}^{(k)} : \mathbb{R}^{N_k} \to \mathbb{R}^d$ yields latent observations $\mathbf{z}_t^{(k)} = \mathcal{P}^{(k)}(\mathbf{y}_t^{(k)})$. These maps include preprocessing and are fitted using training data; our experiments use principal component analysis (PCA) as in Appendix C.1. A common dimension does not ensure comparable coordinates, so we align each source with the target. Let $\mathcal{M}_k^{\text{com}}$ contain training-index pairs (u, v) with $\tau_u^{(0)} = \tau_v^{(k)}$. We estimate an orthogonal matrix in $\mathbb{R}^{d \times d}$ for each source by

$$\widehat{\mathbf{A}}^{(k)} = \underset{\mathbf{A}^\top \mathbf{A} = \mathbf{I}_d}{\arg\min} \sum_{(u,v) \in \mathcal{M}_k^{\text{com}}} \left\| \mathbf{z}_u^{(0)} - \mathbf{A}\mathbf{z}_v^{(k)} \right\|_2^2, \qquad k = 1, \ldots, K,$$

where $\mathbf{I}_d$ is the $d \times d$ identity matrix. The aligned observations are $\mathbf{x}_t^{(0)} = \mathbf{z}_t^{(0)}$ and $\mathbf{x}_t^{(k)} = \widehat{\mathbf{A}}^{(k)} \mathbf{z}_t^{(k)}$ for $k \geq 1$. This construction assumes that latent coordinates are approximately comparable up to rotation and reflection.

Shared contextual encoding. A shared Transformer $E_{\boldsymbol{\phi}}$ maps the aligned target and source contexts to $\mathbf{H}_0 = E_{\boldsymbol{\phi}}(\mathcal{C}_0)$ and $\mathbf{H}_{ks} = E_{\boldsymbol{\phi}}(\mathcal{C}_{ks})$ in $\mathbb{R}^{L \times d_h}$, where d_h is the hidden dimension and the vector $\boldsymbol{\phi}$ collects the shared encoder parameters. The encoder uses an input projection, learnable positional embeddings, multi-head self-attention, and position-wise feed-forward blocks. Cross-attention then summarizes each source context using target embeddings as queries and source embeddings as keys and values:

$$\text{CrossAttn}_{\boldsymbol{\vartheta}}(\mathbf{H}_0, \mathbf{H}_{ks}) = \text{Pool}_t \left[\text{softmax}\left(\frac{(\mathbf{H}_0 \mathbf{W}_Q)(\mathbf{H}_{ks} \mathbf{W}_K)^\top}{\sqrt{d_a}} \right) \mathbf{H}_{ks} \mathbf{W}_V \right] \in \mathbb{R}^{d_a}. \quad (1)$$

Here, $\vartheta = \{\mathbf{W}_Q, \mathbf{W}_K, \mathbf{W}_V\}$ contains learnable projection matrices in $\mathbb{R}^{d_h \times d_a}$, where d_a is the attention dimension. The softmax normalizes over source time steps, and Pool_t averages over target time steps. The group-level and context-level branches use separate attention parameters, included in ϕ_g and ϕ_c, respectively. These two parameter collections also contain the corresponding scoring-function parameters; the roman superscripts g and c label the group and context branches, respectively.

Group-level transferability: where to transfer. For each source group k, we average its encoded historical contexts to obtain a group-level representation,

$$\mathbf{H}_k^{\mathrm{g}} = \frac{1}{S_k} \sum_{s=1}^{S_k} \mathbf{H}_{ks} \in \mathbb{R}^{L \times d_h}.$$

We use (1) and a scoring function to obtain a target-conditioned group summary and its relevance score,

$$\mathbf{z}_k^{\mathrm{g}} = \mathrm{CrossAttn}_{\phi_g}\left(\mathbf{H}_0, \mathbf{H}_k^{\mathrm{g}}\right), \qquad a_k = g_{\phi_g}^{\mathrm{g}}\left(\mathbf{z}_k^{\mathrm{g}}\right),$$

where $\mathbf{z}_k^{\mathrm{g}} \in \mathbb{R}^{d_a}$ and $g_{\phi_g}^{\mathrm{g}} : \mathbb{R}^{d_a} \to \mathbb{R}$ is a learnable scoring function. Normalizing these scores across source groups yields

$$\alpha_k(\mathcal{C}_0) = \frac{\exp(a_k)}{\sum_{j=1}^{K} \exp(a_j)}, \qquad k = 1, \ldots, K.$$

The weight $\alpha_k(\mathcal{C}_0)$ determines *where* to transfer by specifying how much to borrow from source group k.

Context-level transferability: when to transfer. Similarly, we compute a target-conditioned summary and relevance score for each source context,

$$\mathbf{z}_{ks}^{\mathrm{c}} = \mathrm{CrossAttn}_{\phi_c}\left(\mathbf{H}_0, \mathbf{H}_{ks}\right), \qquad b_{ks} = g_{\phi_c}^{\mathrm{c}}\left(\mathbf{z}_{ks}^{\mathrm{c}}\right),$$

where $\mathbf{z}_{ks}^{\mathrm{c}} \in \mathbb{R}^{d_a}$ and $g_{\phi_c}^{\mathrm{c}} : \mathbb{R}^{d_a} \to \mathbb{R}$ is a learnable scoring function. Normalizing these scores within group k yields

$$\beta_{s|k}(\mathcal{C}_0) = \frac{\exp(b_{ks})}{\sum_{r=1}^{S_k} \exp(b_{kr})}, \qquad s = 1, \ldots, S_k.$$

These weights determine *when* to transfer by specifying which historical contexts to emphasize within each source group.

Joint transferability weights. Combining the group-level and context-level weights gives

$$\pi_{ks}(\mathcal{C}_0) = \alpha_k(\mathcal{C}_0)\beta_{s|k}(\mathcal{C}_0).$$

These weights define a normalized distribution over source contexts,

$$\pi_{ks}(\mathcal{C}_0) \geq 0, \qquad \sum_{k=1}^{K} \sum_{s=1}^{S_k} \pi_{ks}(\mathcal{C}_0) = 1. \tag{2}$$

Write $\boldsymbol{\pi} = (\pi_{ks})$ for the joint weight vector. For brevity, we omit the dependence on $\mathcal{C}_0$ when it is clear from context.

3.3 Context-Weighted Transfer and Target Adaptation

For given transferability weights, we estimate the forecasting parameters in two stages. Let $f_{\mathbf{W}}$ map an aligned historical context to its one-step-ahead prediction in the aligned latent space. For predictor parameters $\mathbf{W}$ and a prediction loss ℓ, we define the source-context and empirical target losses as

$$L_{ks}(\mathbf{W}) = \ell\Big(f_{\mathbf{W}}(\mathcal{C}_{ks}), \mathbf{x}_{t_{ks}+1}^{(k)}\Big), \quad L_0(\mathbf{W}) = \frac{1}{S_0} \sum_{i=1}^{S_0} \ell\Big(f_{\mathbf{W}}(\mathcal{C}_{0i}), \mathbf{x}_{t_{0i}+1}^{(0)}\Big),$$

where $\mathcal{C}_{0i}$ is the i-th historical target context and $S_0 \geq 1$ is the number of available target forecasting samples. We assume that the two optimization problems below admit minimizers.

Stage 1: Context-weighted transfer estimation. For the current target context $\mathcal{C}_0$, let $\pi_{ks} = \widehat{\pi}_{ks}$ denote the weights from (A.1), with zero weights assigned to unavailable source contexts. The transferred estimate is

$$\widetilde{\mathbf{W}} \in \arg\min_{\mathbf{W}} \left\{ L_0(\mathbf{W}) + \lambda_{\mathrm{tr}} \sum_{k=1}^{K} \sum_{s=1}^{S_k} \pi_{ks} L_{ks}(\mathbf{W}) + \lambda_{\mathbf{W}} \mathcal{J}(\mathbf{W}) \right\}, \tag{3}$$

where $\lambda_{\mathrm{tr}} \geq 0$ controls the contribution of source contexts and $\lambda_{\mathbf{W}} \geq 0$ sets the strength of the model-specific regularizer $\mathcal{J}$, such as an ℓ_1 (lasso) or nuclear-norm penalty.

Stage 2: Target-specific adaptation. To address residual source–target mismatch, we fix $\widetilde{\mathbf{W}}$ and estimate a correction $\mathbf{\Delta}$ using only the target history,

$$\widehat{\mathbf{\Delta}} \in \arg\min_{\mathbf{\Delta}} \left\{ L_0\left(\widetilde{\mathbf{W}} + \mathbf{\Delta}\right) + \lambda_{\Delta} \mathcal{J}_{\Delta}(\mathbf{\Delta}) \right\}, \tag{4}$$

where $\lambda_{\Delta} \geq 0$ controls the regularization strength and $\mathcal{J}_{\Delta}$ encodes structural assumptions on the source–target discrepancy. The final estimator is $\widehat{\mathbf{W}} = \widetilde{\mathbf{W}} + \widehat{\mathbf{\Delta}}$. With $\widehat{\mathbf{\Delta}} = \mathbf{0}$ when target-specific correction is omitted, the one-step-ahead latent forecast and its target-space reconstruction are

$$\widehat{\mathbf{x}}^{(0)}_{T_0+1} = f_{\widehat{\mathbf{W}}}(\mathcal{C}_0), \qquad \widehat{\mathbf{y}}^{(0)}_{T_0+1} = \mathcal{D}^{(0)}\left(\widehat{\mathbf{x}}^{(0)}_{T_0+1}\right), \tag{5}$$

where $\mathcal{D}^{(0)} : \mathbb{R}^d \to \mathbb{R}^{N_0}$ is the reconstruction map associated with the target representation. Appendix C.1 describes reconstruction and evaluation.

Proposition 1 establishes a target excess-risk bound for Stage 1 under the stated generalization-gap bound. Proposition 2 guarantees non-increase of the empirical target loss under exact Stage 2 correction. Appendix B provides the proof of Proposition 1 and the statement and proof of Proposition 2.

Let $\mathcal{E}$ be the finite-dimensional prediction-parameter space, and let $\mathcal{W} \subseteq \mathcal{E}$ denote the parameter class. The population risks are defined as $R_0(\mathbf{W}) = \mathbb{E}[L_0(\mathbf{W})]$ and $R_{ks}(\mathbf{W}) = \mathbb{E}[L_{ks}(\mathbf{W})]$. Let $\mathbf{W}_0^*$ be a minimizer of R_0 over $\mathcal{W}$. Define $\delta_{ks} = \sup_{\mathbf{W} \in \mathcal{W}} |R_{ks}(\mathbf{W}) - R_0(\mathbf{W})|$. The functions $L_{\boldsymbol{\pi}}$ and $R_{\boldsymbol{\pi}}$ denote the target-plus-weighted-source empirical loss and population risk, respectively, both normalized by $1 + \lambda_{\mathrm{tr}}$.

Proposition 1 (Stage 1: Risk bound for context-weighted transfer estimation)**.** *Under the assumptions in Appendix B.1, let the transferability weights $\boldsymbol{\pi}$ be fixed and satisfy* (2)*, with zero weights assigned to unavailable source contexts. Suppose that*

$$\sup_{\mathbf{W} \in \mathcal{W}} |L_{\boldsymbol{\pi}}(\mathbf{W}) - R_{\boldsymbol{\pi}}(\mathbf{W})| \leq \varepsilon_{\mathrm{gen}}(\boldsymbol{\pi}), \tag{6}$$

for some finite $\varepsilon_{\mathrm{gen}}(\boldsymbol{\pi}) \geq 0$. Let $\varepsilon_{\mathrm{opt}} \geq 0$ denote the normalized objective tolerance in (B.1)*. Then any transferred estimator $\widetilde{\mathbf{W}} \in \mathcal{W}$ satisfying* (B.1) *obeys*

$$R_0(\widetilde{\mathbf{W}}) - R_0(\mathbf{W}_0^*) \leq \frac{2\lambda_{\mathrm{tr}}}{1 + \lambda_{\mathrm{tr}}} \sum_{k=1}^{K} \sum_{s=1}^{S_k} \pi_{ks} \delta_{ks} + \frac{\lambda_{\mathbf{W}}}{1 + \lambda_{\mathrm{tr}}} \mathcal{J}(\mathbf{W}_0^*) + 2\varepsilon_{\mathrm{gen}}(\boldsymbol{\pi}) + \varepsilon_{\mathrm{opt}}. \tag{7}$$

Concentrating all weight on the source context with the smallest δ_{ks} minimizes the discrepancy term in the risk bound, but can increase estimation variance. Appendix B.1 illustrates this bias–variance trade-off through a simplified weighted-mean model.

3.4 Target-Supervised Learning of Transferability Weights

We learn the weighting parameters $\mathbf{\Theta} = (\boldsymbol{\phi}, \boldsymbol{\phi}_g, \boldsymbol{\phi}_c)$ from the last T_{val} target observations, indexed by the validation set $\mathcal{I} = \{1, \ldots, T_{\mathrm{val}}\}$. For each $i \in \mathcal{I}$, set $V_i = T_0 - T_{\mathrm{val}} + i - 1$ and hold out $\mathbf{x}^{(0)}_{V_i+1}$ as the response. We assume $1 \leq T_{\mathrm{val}} \leq T_0 - L - 1$ so that each query has a preceding target context–response pair for fitting. Each weighting-parameter update uses the full validation set, with source eligibility at origin V_i determined by response availability in calendar time:

$$\mathcal{U}_i = \left\{ (k, s) : 1 \leq k \leq K,\ 1 \leq s \leq S_k,\ \tau^{(k)}_{t_{ks}+1} \leq \tau^{(0)}_{V_i} \right\}. \tag{8}$$

The module assigns nonnegative weights $\pi_{ks|i}(\boldsymbol{\Theta})$ summing to one over $\mathcal{U}_i$. Group summaries and both weighting branches use only eligible contexts; groups with no eligible contexts receive zero weight. Let $\mathcal{D}_{<i}$ contain the target pairs $(\mathcal{C}_{0s}, \mathbf{x}^{(0)}_{t_{0s}+1})$ with $t_{0s}+1 \le V_i$. The held-out response is excluded from its own predictor fit and the weighting module's direct inputs, while earlier validation responses may enter later fits once observed. Let $\widetilde{\mathbf{W}}_i(\boldsymbol{\Theta})$ minimize (3) with the target pairs in $\mathcal{D}_{<i}$ and the weighted source contexts in $\mathcal{U}_i$. The validation objective evaluates these predictors before target-specific correction:

$$\mathcal{L}_{\text{meta}}(\boldsymbol{\Theta}) = \frac{1}{|\mathcal{I}|} \sum_{i \in \mathcal{I}} \ell \left(f_{\widetilde{\mathbf{W}}_i(\boldsymbol{\Theta})}(\mathcal{C}^{(0)}_{V_i}), \mathbf{x}^{(0)}_{V_i+1} \right). \tag{9}$$

We update $\boldsymbol{\Theta}$ with Adam by differentiating through the weighted predictor fits. For iterative fitting, each $\widetilde{\mathbf{W}}_i(\boldsymbol{\Theta})$ is approximated by $M \ge 1$ differentiable prediction-parameter updates. Appendix A.1 provides the optimization details and final weight construction.

In the rolling-refit implementation, we fix $\widehat{\boldsymbol{\Theta}}$ and apply (8) at $T_0 + t$ to obtain $\mathcal{U}^{\text{test}}_t$. Each current target context generates weights normalized over this library. We refit the predictor using these weights and available observations, then forecast after optional target-specific correction with Algorithm 1 in appendix A.

4 Experiments

We evaluate forecasting performance on Stock, NY Taxi, PEMS-BAY, and Electricity, then compare computational cost before examining the weighting components through source-corruption ablations and sensitivity analysis. Dataset construction and preprocessing are detailed in Appendix C.1.

4.1 Forecasting Performance and Computational Cost

For each dataset, we split the target history chronologically into training, validation, and test periods. Preprocessing transformations are fitted on training data and held fixed thereafter. Hyperparameters are selected using validation data. We evaluate HiCT with a linear vector autoregressive predictor (HiCT-VAR) and recurrent predictors (HiCT-RNN).

We compare HiCT with three target-only baselines and two transfer methods that incorporate source data. Among the target-only baselines, PatchTST (Nie et al., 2023) uses a patch-based Transformer, TimeMixer (Wang et al., 2024) mixes temporal patterns across scales with MLPs, and AdaRNN (Du et al., 2021) performs temporal adaptation within the target history through recurrent networks. The two source-based transfer methods are DTr-CNN and ROSE. DTr-CNN (Ye & Dai, 2021) transfers a CNN from a single source to the target. For ROSE (Wang et al., 2025), we pre-train a Transformer on the available source groups and fine-tune it on target data. Appendix C.2 lists the code repositories and details the adaptations made to each baseline for our evaluation protocol. We report mean squared error (MSE) and mean absolute error (MAE) in the standardized target test set.

Table 1 reports forecasting performance on the four datasets, and Table 2 compares total fitting and forecasting time on Stock. Overall, HiCT achieves the lowest MSE and MAE on three of the four datasets, with no consistent advantage from replacing the linear predictor with a recurrent model. HiCT-VAR performs best on PEMS-BAY and Electricity, while HiCT-RNN leads on NY Taxi. DTr-CNN achieves the lowest errors on Stock but requires more computation time than HiCT-VAR, which ranks second in both metrics. The Transformer-based PatchTST and ROSE are substantially more time-consuming than HiCT, whose Transformer is used only for the weighting function rather than forecasting. Appendix C.3 describes the timing protocol and predictor updates.

Table 1: One-step-ahead errors in the full standardized target space. Bold and underlining mark the lowest and second-lowest distinct errors, respectively; ties share markings.

	Stock		NY Taxi		PEMS-BAY		Electricity	
Method	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
PatchTST	1.6798	0.9902	0.4226	0.4464	0.1073	0.1966	0.1192	0.2242
TimeMixer	1.2410	0.8291	0.3655	0.4301	0.1301	0.2264	0.1363	0.2424
AdaRNN	1.0412	0.7654	0.3216	0.3987	0.2970	0.3446	0.3360	0.3592
DTr-CNN	**0.8254**	**0.6276**	0.3285	0.4001	0.1087	0.1844	0.1916	0.2841
ROSE	1.1456	0.7835	0.3119	0.3857	0.1081	0.1870	0.1045	<u>0.2142</u>
HiCT-VAR	<u>0.8448</u>	<u>0.6649</u>	<u>0.2764</u>	<u>0.3558</u>	**0.0993**	**0.1749**	**0.0944**	**0.2107**
HiCT-RNN	0.9183	0.7065	**0.2687**	**0.3487**	<u>0.1064</u>	<u>0.1827</u>	<u>0.0989</u>	0.2148

Table 2: Total fitting and forecasting time on Stock (eight CPU threads), including any rolling refits.

Method	PatchTST	TimeMixer	AdaRNN	DTr-CNN	ROSE	HiCT-VAR	HiCT-RNN
Total (s)	2942.6	29.5	4.3	56.9	527.5	9.3	10.9

4.2 Ablation and Sensitivity Analyses

We examine group-level and within-group context weighting through two source-corruption experiments on U.S. Stock. The first replaces the entire available Industrials input sequence with independent zero-mean Gaussian noise. The second applies the same corruption to the later chronological half of each source group's available input sequence. Noise is scaled by the corresponding training-coordinate standard deviations, while target observations remain unchanged. We refer to the original condition as *unperturbed sources*.

For each source condition, we learn the weighting function using target validation loss. We then remove one weighting component while keeping the fitted scorer and predictor settings fixed. The w/o learned α variant sets $\alpha_k = 1/4$ for all four source groups and retains learned β. The w/o learned β variant uses uniform within-group weights and retains learned α. Target-only excludes the source loss while preserving the target representation and predictor penalty. These comparisons assess adaptation through weight learning under source corruption, rather than the response of a frozen model trained on unperturbed sources.

Table 3: Forecasting errors under source corruption on Stock. Bold indicates the lowest error for each metric within each condition. Parentheses report percentage reductions relative to Target-only, calculated from the displayed errors. Negative values indicate deterioration.

Source condition	Method	MSE	MAE
Unperturbed	Target-only	0.9818 (0.00%)	0.7308 (0.00%)
	HiCT	**0.8457** (13.86%)	**0.6662** (8.84%)
Industrials corrupted (entire history)	w/o α	0.8782 (10.55%)	0.6824 (6.62%)
	HiCT	**0.8696** (11.43%)	**0.6789** (7.10%)
All sources corrupted (later half)	w/o β	0.9900 (-0.84%)	0.7151 (2.15%)
	HiCT	**0.9719** (1.01%)	**0.7086** (3.04%)

When the entire Industrials input history is replaced by noise, HiCT achieves lower MSE and MAE than the equal-group weighting variant in Table 3. The lower mean weight assigned to Industrials in Figure 2(a) is consistent with reduced reliance on the corrupted source. These results support learned group weighting in this setting.

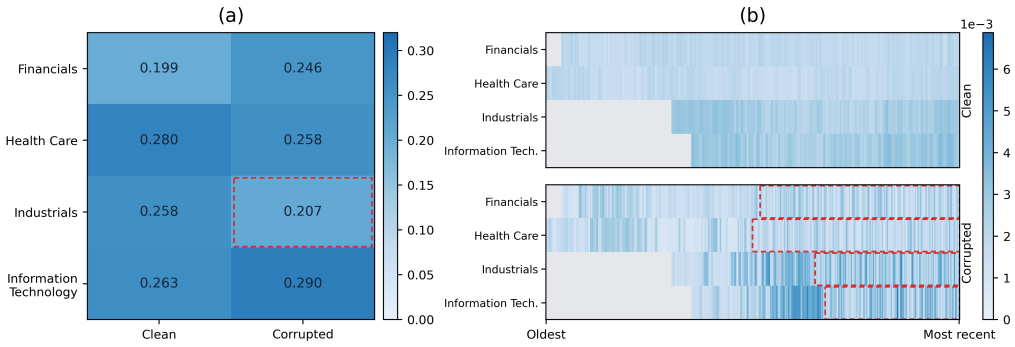

Figure 2: Mean transferability weights on Stock: (a) group weights α when the entire Industrials input history is corrupted; (b) conditional within-group weights β when the later half of each source's input history is corrupted. Weights are averaged over test forecasting origins. Red boxes mark the corrupted group or contexts; gray denotes padding. Each panel uses a common color scale across conditions.

When the later half of each source's input history is corrupted, HiCT also achieves lower errors than uniform within-group weighting. Both errors remain below Target-only, whereas removing learned β raises MSE above this baseline. Figure 2(b) shows increased weights on some uncorrupted contexts in Industrials and Information Technology. The uncorrupted half also receives greater total weight in front half of Industrials and Information Technology.

Figure 3 examines sensitivity to context length L, source-context stride h, and latent dimension d on Stock. Extending the context beyond the selected short window or using coarser source sampling increases test error, while gains from increasing d level off. Appendix C.4 provides the corresponding validation curves and discusses parameter selection. Appendix D presents heatmaps of the learned transferability weights across source groups and historical contexts.

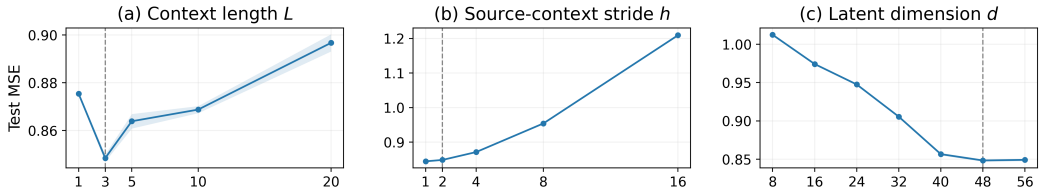

Figure 3: Test sensitivity on Stock. Each sweep varies one parameter, fixing the others at $(L, h, d) = (3, 2, 48)$. Curves and shading show the mean and one sample standard deviation over five runs; dashed lines mark validation-selected values.

5 Conclusion

We proposed HiCT for multivariate forecasting with short target histories. By learning *where* and *when* to transfer, HiCT draws on relevant information across source groups and historical contexts. Our theoretical analysis characterizes target excess risk and the role of target-specific correction. Experiments on four datasets demonstrate competitive forecasting performance, while source-corruption studies highlight the contributions of group-level and within-group weighting.

Future work will explore three directions. First, we will examine Transformer-based forecasting backbones such as PatchTST and extend HiCT to multi-step prediction. Second, inspired by online forecasting methods (Zhang et al., 2023), we will investigate updates to the weighting parameters as target dynamics evolve. Finally, we will explore efficient bilevel optimization (Ji et al., 2021) and predictor refitting to support larger source libraries.

AI USE STATEMENT

In this work, we used generative AI tools to assist with language editing, identifying related literature, and developing parts of the theoretical arguments. The authors reviewed the AI-assisted material and revised it where appropriate. The experiments were conducted by the authors. The authors take full responsibility for the final content of this paper, including its text, references, theoretical claims, proofs, and experimental results.

REPRODUCIBILITY STATEMENT

To ensure the reproducibility and completeness of this paper, we include the implementation details in the Appendix, which also contains the link to an anonymized code repository for reproducing all the results that appeared in the paper. Files for accessing baselines and the routes to access the raw datasets as well as preprocessed data used in the paper are also included. Protocols for data preprocessing are in the Appendix, reflected in the codes.

REFERENCES

Yong-Hoon Choi, Park Kwang-Hyun, and Youngjin Cho. Which histories matter for time series forecasting? learning predictive relevance with future supervision. *arXiv preprint arXiv:2608.23221*, 2026.

Yuntao Du, Jindong Wang, Wenjie Feng, Sinno Pan, Tao Qin, Renjun Xu, and Chongjun Wang. AdaRNN: Adaptive learning and forecasting for time series. *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, 21:402–411, 2021.

Junting Duan, Markus Pelger, and Ruoxuan Xiong. Target PCA: Transfer learning large dimensional panel data. *Journal of Econometrics*, https://doi.org/10.1016/j.jeconom.2023.105521, 2024.

Sungwon Han, Seungeon Lee, Meeyoung Cha, Sercan Ö. Arik, and Jinsung Yoon. Retrieval augmented time series forecasting. *Proceedings of the 42nd International Conference on Machine Learning*, 267:21774–21797, 2025.

Junkyu Jang, Taehwan Kim, Daesan Oh, and Sung-Hyuk Park. Adaptive sample weighting with regime-aware meta-learning framework for financial forecasting. *Proceedings of the 6th ACM International Conference on AI in Finance*, pp. 501–508, 2025.

Kaiyi Ji, Junjie Yang, and Yingbin Liang. Bilevel optimization: Convergence analysis and enhanced design. *Proceedings of the 38th International Conference on Machine Learning*, 139:4882–4892, 2021.

Sai Li, T. Tony Cai, and Hongzhe Li. Transfer learning for high-dimensional linear regression: Prediction, estimation and minimax optimality. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 84:149–173, 2022a.

Wendi Li, Xiao Yang, Weiqing Liu, Yingce Xia, and Jiang Bian. DDG-DA: Data distribution generation for predictable concept drift adaptation. *Proceedings of the 36th AAAI Conference on Artificial Intelligence*, 36:4092–4100, 2022b.

Mingliang Ma and Abolfazl Safikhani. Transfer learning for high-dimensional reduced rank time series models. *Proceedings of the 28th International Conference on Artificial Intelligence and Statistics*, 258:2926–2934, 2025.

Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. *Proceedings of the 11th International Conference on Learning Representations*, 2023.

Seyoung Park, Eun Ryung Lee, Hyunjin Kim, and Hongyu Zhao. Transfer learning under large-scale low-rank regression models. *Journal of the American Statistical Association*, 121:1154–1166, 2026.

Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. Learning to reweight examples for robust deep learning. *Proceedings of the 35th International Conference on Machine Learning*, 80:4334–4343, 2018.

Jun Shu, Qi Xie, Lixuan Yi, Qian Zhao, Sanping Zhou, Zongben Xu, and Deyu Meng. Meta-Weight-Net: Learning an explicit mapping for sample weighting. *Proceedings of the 33rd Conference on Neural Information Processing Systems*, 32:1919–1930, 2019.

Andreas Teller, Uta Pigorsch, and Christian Pigorsch. Realized volatility forecasting for new issues and spin-offs using multi-source transfer learning. *International Journal of Forecasting*, 42:1069–1103, 2026.

Shiyu Wang, Haixu Wu, Xiaoming Shi, Tengge Hu, Huakun Luo, Lintao Ma, James Y. Zhang, and Jun Zhou. TimeMixer: Decomposable multiscale mixing for time series forecasting. *Proceedings of the 12th International Conference on Learning Representations*, 2024.

Yihang Wang, Yuying Qiu, Peng Chen, Kai Zhao, Yang Shu, Zhongwen Rao, Lujia Pan, Bin Yang, and Chenjuan Guo. Towards a general time series forecasting model with unified representation and adaptive transfer. *Proceedings of the 42nd International Conference on Machine Learning*, 267:64127–64151, 2025.

Rui Ye and Qun Dai. Implementing transfer learning across different datasets for time series forecasting. *Pattern Recognition*, https://doi.org/10.1016/j.patcog.2020.107617, 2021.

Huanyu Zhang, Chang Xu, Yi-Fan Zhang, Zhang Zhang, Liang Wang, and Jiang Bian. TimeRAF: Retrieval-augmented foundation model for zero-shot time series forecasting. *IEEE Transactions on Knowledge and Data Engineering*, 37:5654–5665, 2025.

Yi-Fan Zhang, Qingsong Wen, Xue Wang, Weiqi Chen, Liang Sun, Zhang Zhang, Liang Wang, Rong Jin, and Tieniu Tan. OneNet: Enhancing time series forecasting models under concept drift by online ensembling. *Proceeding of the 37th International Conference on Neural Information Processing Systems*, 36:69949–69980, 2023.

Junlong Zhao, Shengbin Zheng, and Chenlei Leng. Residual importance weighted transfer learning for high-dimensional linear regression. *Journal of the American Statistical Association*, https://doi.org/10.1080/01621459.2026.2623997, 2026.

Shiqiao Zhou, Holger Schöner, Zipeng Wu, Edouard Fouché, Grant Wilson, and Shuo Wang. Stationarity-aware retrieval-augmented time series forecasting. *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2*, pp. 6875–6886, 2026.

Xiaofeng Zhou, Naiju Zhai, Shuai Li, and Haibo Shi. Time series prediction method of industrial process with limited data based on transfer learning. *IEEE Transactions on Industrial Informatics*, 19:6872–6882, 2023.

APPENDIX

Appendix A presents the algorithm and optimization details. Appendix B provides the theoretical results and proofs. Appendix C describes the datasets and group construction, and Appendix D presents heatmaps of the learned transferability weights. The code files are available at: `https://anonymous.4open.science/r/HiCT-6473`

A ALGORITHM DETAILS

Notation. Parenthesized superscripts identify groups, with (0) denoting the target group. Square-bracketed superscripts identify optimization steps: $\boldsymbol{\Theta}^{[m]}$ denotes the weighting parameters after m updates, whereas $\widetilde{\mathbf{W}}_i^{[m]} = \widetilde{\mathbf{W}}_i(\boldsymbol{\Theta}^{[m]})$ denotes the prediction parameters after m updates for validation query i in the iterative formulation of Appendix A.1. Thus, $[0]$ indicates initialization. Let $M \geq 1$ be the maximum number of weighting-parameter updates and $\eta_\Theta > 0$ their learning rate.

Validation queries and weight learning. We use the validation set $\mathcal{I} = \{1, \dots, T_{\mathrm{val}}\}$ defined in Section 3.4. For query i, the forecasting origin is $V_i = T_0 - T_{\mathrm{val}} + i - 1$. Let $\mathbf{u}_i = \mathrm{vec}(\mathcal{C}_{V_i}^{(0)}) \in \mathbb{R}^{Ld}$ and $\mathbf{v}_i = \mathbf{x}_{V_i+1}^{(0)} \in \mathbb{R}^d$ denote the vectorized context and held-out response, respectively. Here, vec stacks the context entries in the order used by the predictor. The conditioning index i in $\pi_{ks|i} = \alpha_{k|i}\beta_{s|k,i}$ identifies the query. Weights are normalized over its eligible source library $\mathcal{U}_i$, which is fixed across learning iterations. The corresponding predictor uses preceding target samples in $\mathcal{D}_{<i}$ and source contexts in $\mathcal{U}_i$; the held-out response is excluded from this fit.

For weight learning, we use a linear predictor. Its fitted coefficient matrix $\mathbf{W}_i(\mathbf{\Theta}^{[m]}) \in \mathbb{R}^{Ld\times d}$ corresponds to $\widetilde{\mathbf{W}}_i(\mathbf{\Theta})$ in the main text under this parameterization after m updates. Differentiation through an exact fit requires the selected solution map to be differentiable with respect to $\mathbf{\Theta}$. We denote the predictor's mean squared error across the d held-out latent coordinates by $\ell_i(\mathbf{\Theta})$. At each iteration, Adam updates the weighting parameters using the full-validation objective $\mathcal{L}_{\mathrm{meta}}$ in (9), differentiating through the query-specific predictor fits. The validation set remains unchanged throughout learning. Appendix A.1 describes the alternative formulation based on a finite sequence of differentiable predictor updates.

Rolling prediction. Algorithm 1 describes the linear-predictor implementation with rolling refits. After learning, we fix $\widehat{\mathbf{\Theta}}$ and generate source weights for each new target context. For a test forecasting origin $t \geq T_0$, let $\mathbf{u}_t = \mathrm{vec}(\mathcal{C}_t^{(0)})$. The eligible source library is $\mathcal{U}_{t-T_0}^{\mathrm{test}}$, and $\pi_{ks,t}$ denotes the weight assigned to source context (k, s) at that origin.

The refitted test predictor includes a coefficient matrix $\widetilde{\mathbf{W}}_t \in \mathbb{R}^{Ld\times d}$ and an intercept $\widetilde{\mathbf{b}}_t \in \mathbb{R}^d$. Unlike the validation predictor used for weight learning, this predictor therefore includes an intercept. When target-specific correction is enabled, its coefficient and intercept corrections are $\widehat{\mathbf{\Delta}}_t \in \mathbb{R}^{Ld\times d}$ and $\widehat{\mathbf{c}}_t \in \mathbb{R}^d$. The configured correction multiplier is denoted by γ_{corr}, distinguishing it from the normalized source-loss coefficient γ in the risk analysis. The latent forecast $\widehat{\mathbf{x}}_{t+1}^{(0)}$ is reconstructed as $\widehat{\mathbf{y}}_{t+1}^{(0)}$ in the target observation space. The recurrent implementation uses a fixed prediction head during testing.

A.1 Rolling validation and optimization details

The validation length T_{val} is a positive integer chosen so that every validation query has a length-L context and a nonempty set $\mathcal{D}_{<i}$ of preceding target samples for predictor fitting. Every weighting-parameter update uses all T_{val} validation responses. All source libraries used for weight normalization are nonempty. Any retained-mass filtering preserves a positive total weight. The penalty coefficients and stopping tolerance are nonnegative, and both learning rates are positive. The shared prediction-parameter initialization $\mathbf{W}_{\mathrm{init}}$ is shared across all predictor fits and is independent of $\mathbf{\Theta}$. Any data-dependent initialization uses only observations available at the earliest rolling-query origin.

Eligible source contexts and normalized weights. For each historical query $\mathcal{C}_{V_i}^{(0)}$, the eligible source library $\mathcal{U}_i$ is defined by (8). Define

$$\mathcal{S}_{k|i} = \big\{s \in \{1, \dots, S_k\} : (k, s) \in \mathcal{U}_i\big\}, \qquad k = 1, \dots, K,$$
$$\mathcal{K}_i = \big\{k \in \{1, \dots, K\} : |\mathcal{S}_{k|i}| > 0\big\}.$$

For $k \in \mathcal{K}_i$, the group summary averages the encoded contexts indexed by $\mathcal{S}_{k|i}$ only. The group-level softmax is evaluated over $\mathcal{K}_i$, and each within-group softmax is evaluated over $\mathcal{S}_{k|i}$. Thus, unavailable contexts are excluded before constructing group summaries and computing either normalization. Groups outside $\mathcal{K}_i$ receive zero group weight and zero joint weights, and no within-group normalization is performed for these groups. The resulting joint weights satisfy

$$\pi_{ks|i}(\mathbf{\Theta}) = \pi_{ks}\Big(\mathcal{C}_{V_i}^{(0)}; \mathbf{\Theta}\Big) \geq 0, \quad (k, s) \in \mathcal{U}_i,$$
$$\sum_{(k,s)\in\mathcal{U}_i} \pi_{ks|i}(\mathbf{\Theta}) = 1.$$

Algorithm 1 Learning Transferable Historical Contexts

Input: Target and source histories $\{\mathbf{y}_t^{(k)}\}_{t=1}^{T_k}$, $k = 0, \ldots, K$; context length L, stride h, and latent dimension d; maximum learning iterations M and learning rate η_Θ; validation length T_{val}; validation-selected penalty, weighting, and correction settings.

Output: Target forecasts $\widehat{\mathbf{y}}_{t+1}^{(0)}$.

Fit and freeze standardization, PCA, and Procrustes alignment using training data;
Construct target samples, the validation set $\mathcal{I}$, and source libraries using the prescribed cutoffs;
Initialize the weighting parameters $\boldsymbol{\Theta}^{[0]}$;
for $m = 0, \ldots, M-1$ **do**
    for $i \in \mathcal{I}$ **do**
        Compute $\pi_{ks|i} = \alpha_{k|i}\beta_{s|k,i}$ using $\boldsymbol{\Theta}^{[m]}$ over $\mathcal{U}_i$;
        Obtain $\mathbf{W}_i(\boldsymbol{\Theta}^{[m]})$ by differentiable weighted predictor fitting, using target samples in $\mathcal{D}_{<i}$ and source contexts in $\mathcal{U}_i$;
        Compute $\ell_i(\boldsymbol{\Theta}^{[m]}) \leftarrow d^{-1}\|\mathbf{v}_i - \mathbf{W}_i(\boldsymbol{\Theta}^{[m]})^\top \mathbf{u}_i\|_2^2$;
    end for
    Compute $\mathcal{L}_{\mathrm{meta}}(\boldsymbol{\Theta}^{[m]}) \leftarrow T_{\mathrm{val}}^{-1} \sum_{i \in \mathcal{I}} \ell_i(\boldsymbol{\Theta}^{[m]})$;
    Obtain $\boldsymbol{\Theta}^{[m+1]}$ by an Adam update with learning rate η_Θ, differentiating through the predictor fits;
end for
Freeze $\widehat{\boldsymbol{\Theta}} \leftarrow \boldsymbol{\Theta}^{[M]}$ and disable dropout;
for each test prediction context $\mathcal{C}_t^{(0)}$ **do**
    Compute context-specific weights $\pi_{ks,t}$ using $\widehat{\boldsymbol{\Theta}}$ over $\mathcal{U}_{t-T_0}^{\mathrm{test}}$;
    Apply the selected beta temperature and retained-mass rule;
    Normalize the retained nonnegative source weights to sum to one;
    Refit the Stage 1 predictor $(\widetilde{\mathbf{W}}_t, \widetilde{\mathbf{b}}_t)$ using the selected penalty;
    Set $(\widehat{\boldsymbol{\Delta}}_t, \widehat{\mathbf{c}}_t) \leftarrow (\mathbf{0}, \mathbf{0})$;
    if target-specific correction is enabled **then**
        Fit $(\widehat{\boldsymbol{\Delta}}_t, \widehat{\mathbf{c}}_t)$ to target fitting residuals;
    end if
    Set $\mathbf{u}_t \leftarrow \mathrm{vec}(\mathcal{C}_t^{(0)})$;
    Compute $\widehat{\mathbf{x}}_{t+1}^{(0)} \leftarrow \widetilde{\mathbf{b}}_t + \widetilde{\mathbf{W}}_t^\top \mathbf{u}_t + \gamma_{\mathrm{corr}}(\widehat{\mathbf{c}}_t + \widehat{\boldsymbol{\Delta}}_t^\top \mathbf{u}_t)$;
    Reconstruct $\widehat{\mathbf{y}}_{t+1}^{(0)}$ through the target PCA decoder and apply the configured prediction floor;
end for
return forecasts $\widehat{\mathbf{y}}_{t+1}^{(0)}$ for all test origins t.

Weights outside $\mathcal{U}_i$ are zero. The dependence of the weighting function on the eligible library is suppressed in the notation. The held-out response $\mathbf{x}_{V_i+1}^{(0)}$ is excluded from the weighting inputs and the predictor fit.

Prediction-parameter fitting. The predictor-fitting target loss averages over the nonempty set $\mathcal{D}_{<i}$ of preceding target context–response pairs:

$$L_{0,<i}(\mathbf{W}) = \frac{1}{|\mathcal{D}_{<i}|} \sum_{(\mathcal{C},\mathbf{x}) \in \mathcal{D}_{<i}} \ell(f_{\mathbf{W}}(\mathcal{C}), \mathbf{x}), \qquad i = 1, \ldots, T_{\mathrm{val}}.$$

The regularized predictor-fitting objective is

$$\mathcal{L}_i^{\mathrm{in}}(\mathbf{W}; \boldsymbol{\Theta}) = L_{0,<i}(\mathbf{W}) + \lambda_{\mathrm{tr}} \sum_{(k,s) \in \mathcal{U}_i} \pi_{ks|i}(\boldsymbol{\Theta}) L_{ks}(\mathbf{W}) + \lambda_{\mathbf{W}} \mathcal{J}(\mathbf{W}).$$

Assuming a minimizer exists, a selected minimizer $\widetilde{\mathbf{W}}_i(\boldsymbol{\Theta})$ defines the ideal validation objective in (9).

For computation, let $\mathcal{T}_{i,\eta_{\text{in}}}$ denote the chosen prediction-parameter update with step size η_{in}. Starting from $\mathbf{W}_i^{[0]}(\boldsymbol{\Theta}) = \mathbf{W}_{\text{init}}$, we compute

$$\mathbf{W}_i^{[m+1]}(\boldsymbol{\Theta}) = \mathcal{T}_{i,\eta_{\text{in}}}\Big(\mathbf{W}_i^{[m]}(\boldsymbol{\Theta}); \boldsymbol{\Theta}\Big), \qquad m = 0, \dots, M-1.$$

For the differentiation below, the update map is assumed differentiable with respect to its arguments along the unrolled trajectory. When the predictor-fitting objective is twice continuously differentiable in $(\mathbf{W}, \boldsymbol{\Theta})$, a gradient descent update is

$$\mathcal{T}_{i,\eta_{\text{in}}}(\mathbf{W}; \boldsymbol{\Theta}) = \mathbf{W} - \eta_{\text{in}} \nabla_{\mathbf{W}} \mathcal{L}_i^{\text{in}}(\mathbf{W}; \boldsymbol{\Theta}).$$

For a nonsmooth regularizer, an appropriate update and differentiation rule must be specified separately; ordinary gradient descent is not assumed.

Validation-based weighting-parameter updates. For the finite-step variant, the m-step estimates define the full-validation objective:

$$\mathcal{L}_{\text{meta}}^{[m]}(\boldsymbol{\Theta}) = \frac{1}{T_{\text{val}}} \sum_{i=1}^{T_{\text{val}}} \ell\Big(f_{\mathbf{W}_i^{[m]}(\boldsymbol{\Theta})}(\mathcal{C}_{V_i}^{(0)}), \mathbf{x}_{V_i+1}^{(0)}\Big).$$

Assuming differentiability along the unrolled trajectories, we compute the gradient of this finite-step objective by backpropagating through all M prediction-parameter updates. Even when the validation objective based on an exact predictor fit is differentiable, its gradient need not coincide with this finite-step gradient. At learning iteration m, an update takes the form

$$\boldsymbol{\Theta}^{[m+1]} = \boldsymbol{\Theta}^{[m]} - \eta_{\Theta} \nabla_{\boldsymbol{\Theta}} \mathcal{L}_{\text{meta}}^{[m]}\Big(\boldsymbol{\Theta}^{[m]}\Big).$$

In the finite-step variant, $\boldsymbol{\Theta}$ stores the current weighting-parameter iterate. The code-based algorithm instead differentiates through predictor fits and uses its fixed Adam update budget.

For the finite-step variant, we perform at most M weighting-parameter updates. At each iteration, the loss $v_m = \mathcal{L}_{\text{meta}}^{[m]}(\boldsymbol{\Theta}^{[m]})$ is evaluated before updating the parameters. For $m \geq 1$, optimization stops before the next update if $|v_m - v_{m-1}| \leq \epsilon_{\text{tol}}$. The returned $\widehat{\boldsymbol{\Theta}}$ is the current iterate when this condition is met or the update budget is exhausted.

Current-origin transfer and prediction. At the current forecasting origin $\tau_{T_0}^{(0)}$, define

$$\mathcal{U}_0 = \Big\{(k,s) : 1 \leq k \leq K,\ 1 \leq s \leq S_k,\ \tau_{t_{ks}+1}^{(k)} \leq \tau_{T_0}^{(0)}\Big\}.$$

Applying the same eligible-context restrictions to group summaries and both normalizations gives

$$\widehat{\pi}_{ks} = \pi_{ks}\Big(\mathcal{C}_0; \widehat{\boldsymbol{\Theta}}\Big), \qquad (k,s) \in \mathcal{U}_0, \tag{A.1}$$

where $\widehat{\pi}_{ks} \geq 0$ and $\sum_{(k,s)\in\mathcal{U}_0} \widehat{\pi}_{ks} = 1$. Unavailable contexts receive zero weight. Using these weights and all target forecasting samples available at the current origin, we fit $\widetilde{\mathbf{W}}$ using (3). If target-specific correction is enabled, we estimate $\widehat{\boldsymbol{\Delta}}$ using (4); otherwise, we set $\widehat{\boldsymbol{\Delta}} = \mathbf{0}$. The final estimator is $\widehat{\mathbf{W}} = \widetilde{\mathbf{W}} + \widehat{\boldsymbol{\Delta}}$, and its forecast is reconstructed using (5). This is the unscaled correction defined in Section 3.3. Algorithm 1 also allows an implementation-specific correction multiplier and prediction floor; these operations are separate from the estimator analyzed in Proposition 2.

B Theoretical Properties

Proposition 1 bounds the target excess risk of the regularized transfer estimator under a uniform deviation condition. Proposition 2 establishes an empirical guarantee for target-specific adaptation. We illustrate weight concentration with a weighted-mean model and the effect of the correction penalty with a simplified ridge model. These calculations explain statistical trade-offs under explicit simplifying assumptions.

B.1 CONTEXT-WEIGHTED TRANSFER ESTIMATION

Let $\mathcal{E}$ be a finite-dimensional real Euclidean space. For the transfer-risk analysis, consider a fixed nonempty compact parameter set $\mathcal{W} \subseteq \mathcal{E}$. The representation maps are treated as deterministic, and the index set of available source contexts is fixed and nonempty. This analysis does not account for uncertainty from estimating the representation maps.

We use L to denote an empirical loss, R to denote a population risk, and $\mathcal{J}$ to denote a regularization penalty. The Stage 1 penalty $\mathcal{J}(\mathbf{W})$ controls the complexity of the forecasting model. We write $Q_{\boldsymbol{\pi}}$ for the original regularized empirical objective and $\overline{Q}_{\boldsymbol{\pi}}$ for its normalized version.

Assume that the empirical losses are measurable functions of the data and are finite and continuous on $\mathcal{W}$ almost surely. The corresponding population risks are assumed to be finite and continuous on $\mathcal{W}$. The regularizer $\mathcal{J}$ is deterministic, finite, nonnegative, and continuous on $\mathcal{W}$. We restrict our analysis to transferred estimators $\widetilde{\mathbf{W}} \in \mathcal{W}$. All estimators and generalization-gap events considered below are assumed measurable.

Let $L_0(\mathbf{W})$ denote the empirical target loss and $L_{ks}(\mathbf{W})$ the prediction loss for the source context–response pair indexed by (k, s). For each fixed $\mathbf{W} \in \mathcal{W}$, define the corresponding population risks as

$$R_0(\mathbf{W}) = \mathbb{E}[L_0(\mathbf{W})], \qquad R_{ks}(\mathbf{W}) = \mathbb{E}[L_{ks}(\mathbf{W})].$$

The expectations are taken over the time-series data under the fixed setup described above. Here, R_0 is the population counterpart of the empirical loss over historical target forecasting samples. Relating this risk to prediction at the current forecasting origin requires additional assumptions on temporal distribution changes.

We measure the source–target discrepancy by

$$\delta_{ks} = \sup_{\mathbf{W} \in \mathcal{W}} \left| R_{ks}(\mathbf{W}) - R_0(\mathbf{W}) \right|.$$

Each δ_{ks} is finite under the compactness and continuity assumptions above. A smaller δ_{ks} indicates closer agreement between the source-context risk and the target risk over $\mathcal{W}$.

Let the transferability weights $\boldsymbol{\pi} = (\pi_{ks})$ satisfy

$$\pi_{ks} \geq 0, \qquad \sum_{k=1}^{K} \sum_{s=1}^{S_k} \pi_{ks} = 1,$$

with zero weights assigned to unavailable source contexts. These are the normalization conditions in (2). For fixed hyperparameters $\lambda_{\mathrm{tr}}, \lambda_{\mathbf{W}} \geq 0$, define the context-weighted population risk as

$$R_{\boldsymbol{\pi}}(\mathbf{W}) = \frac{R_0(\mathbf{W}) + \lambda_{\mathrm{tr}} \sum_{k=1}^{K} \sum_{s=1}^{S_k} \pi_{ks} R_{ks}(\mathbf{W})}{1 + \lambda_{\mathrm{tr}}}.$$

Its empirical counterpart is

$$L_{\boldsymbol{\pi}}(\mathbf{W}) = \frac{L_0(\mathbf{W}) + \lambda_{\mathrm{tr}} \sum_{k=1}^{K} \sum_{s=1}^{S_k} \pi_{ks} L_{ks}(\mathbf{W})}{1 + \lambda_{\mathrm{tr}}}.$$

Let $\mathbf{W}_0^*$ denote a target population risk minimizer,

$$\mathbf{W}_0^* \in \arg\min_{\mathbf{W} \in \mathcal{W}} R_0(\mathbf{W}).$$

Such a minimizer exists because $\mathcal{W}$ is nonempty and compact and R_0 is continuous on $\mathcal{W}$.

For fixed transferability weights, the regularized empirical objective in Stage 1 is

$$Q_{\boldsymbol{\pi}}(\mathbf{W}) = L_0(\mathbf{W}) + \lambda_{\mathrm{tr}} \sum_{k=1}^{K} \sum_{s=1}^{S_k} \pi_{ks} L_{ks}(\mathbf{W}) + \lambda_{\mathbf{W}} \mathcal{J}(\mathbf{W}).$$

Dividing this objective by $1 + \lambda_{\mathrm{tr}}$ gives the normalized objective, with normalized regularization coefficient $\rho := \lambda_{\mathbf{W}} / (1 + \lambda_{\mathrm{tr}})$,

$$\overline{Q}_{\boldsymbol{\pi}}(\mathbf{W}) = \frac{Q_{\boldsymbol{\pi}}(\mathbf{W})}{1 + \lambda_{\mathrm{tr}}} = L_{\boldsymbol{\pi}}(\mathbf{W}) + \rho \mathcal{J}(\mathbf{W}).$$

This normalization rescales both the loss and the regularization term. Since $1 + \lambda_{\mathrm{tr}} > 0$ is independent of $\mathbf{W}$, the original and normalized objectives have the same minimizers over $\mathcal{W}$.

To account for approximate optimization, assume that the transferred estimator $\widetilde{\mathbf{W}} \in \mathcal{W}$ satisfies

$$\overline{Q}_{\boldsymbol{\pi}}(\widetilde{\mathbf{W}}) \leq \inf_{\mathbf{W} \in \mathcal{W}} \overline{Q}_{\boldsymbol{\pi}}(\mathbf{W}) + \varepsilon_{\mathrm{opt}}, \tag{B.1}$$

where $\varepsilon_{\mathrm{opt}} \geq 0$ is finite and measures the optimization error in the normalized objective. Exact minimization corresponds to $\varepsilon_{\mathrm{opt}} = 0$.

Proof of Proposition 1.

Proof. Let

$$D_{\boldsymbol{\pi}} = \sum_{k=1}^{K} \sum_{s=1}^{S_k} \pi_{ks} \delta_{ks}, \qquad \gamma = \frac{\lambda_{\mathrm{tr}}}{1 + \lambda_{\mathrm{tr}}},$$

and recall that $\rho = \lambda_{\mathbf{W}} / (1 + \lambda_{\mathrm{tr}})$.

For any $\mathbf{W} \in \mathcal{W}$, the definitions of $R_{\boldsymbol{\pi}}$ and δ_{ks}, together with the nonnegativity and normalization of the weights, imply

$$\begin{aligned} |R_{\boldsymbol{\pi}}(\mathbf{W}) - R_0(\mathbf{W})| &= \gamma \left| \sum_{k=1}^{K} \sum_{s=1}^{S_k} \pi_{ks} \left[R_{ks}(\mathbf{W}) - R_0(\mathbf{W}) \right] \right| \\ &\leq \gamma \sum_{k=1}^{K} \sum_{s=1}^{S_k} \pi_{ks} \left| R_{ks}(\mathbf{W}) - R_0(\mathbf{W}) \right| \\ &\leq \gamma D_{\boldsymbol{\pi}}. \end{aligned} \tag{B.2}$$

Next, since $\mathbf{W}_0^* \in \mathcal{W}$, the approximate optimality condition (B.1) gives

$$\overline{Q}_{\boldsymbol{\pi}}(\widetilde{\mathbf{W}}) \leq \inf_{\mathbf{W} \in \mathcal{W}} \overline{Q}_{\boldsymbol{\pi}}(\mathbf{W}) + \varepsilon_{\mathrm{opt}} \leq \overline{Q}_{\boldsymbol{\pi}}(\mathbf{W}_0^*) + \varepsilon_{\mathrm{opt}}.$$

Substituting $\overline{Q}_{\boldsymbol{\pi}} = L_{\boldsymbol{\pi}} + \rho \mathcal{J}$ and rearranging yields

$$\begin{aligned} L_{\boldsymbol{\pi}}(\widetilde{\mathbf{W}}) - L_{\boldsymbol{\pi}}(\mathbf{W}_0^*) &\leq \rho \left[\mathcal{J}(\mathbf{W}_0^*) - \mathcal{J}(\widetilde{\mathbf{W}}) \right] + \varepsilon_{\mathrm{opt}} \\ &\leq \rho \mathcal{J}(\mathbf{W}_0^*) + \varepsilon_{\mathrm{opt}}, \end{aligned} \tag{B.3}$$

where the last inequality uses $\rho \geq 0$ and $\mathcal{J}(\widetilde{\mathbf{W}}) \geq 0$.

Applying (B.2) at $\widetilde{\mathbf{W}}$ and $\mathbf{W}_0^*$ gives

$$R_0(\widetilde{\mathbf{W}}) - R_0(\mathbf{W}_0^*) \leq R_{\boldsymbol{\pi}}(\widetilde{\mathbf{W}}) - R_{\boldsymbol{\pi}}(\mathbf{W}_0^*) + 2\gamma D_{\boldsymbol{\pi}}.$$

The generalization-gap bound (6) also implies

$$R_{\boldsymbol{\pi}}(\widetilde{\mathbf{W}}) - R_{\boldsymbol{\pi}}(\mathbf{W}_0^*) \leq L_{\boldsymbol{\pi}}(\widetilde{\mathbf{W}}) - L_{\boldsymbol{\pi}}(\mathbf{W}_0^*) + 2\varepsilon_{\mathrm{gen}}(\boldsymbol{\pi}).$$

Combining these inequalities with (B.3), we obtain

$$R_0(\widetilde{\mathbf{W}}) - R_0(\mathbf{W}_0^*) \leq 2\gamma D_{\boldsymbol{\pi}} + \rho \mathcal{J}(\mathbf{W}_0^*) + 2\varepsilon_{\mathrm{gen}}(\boldsymbol{\pi}) + \varepsilon_{\mathrm{opt}}.$$

Substituting the definitions of $D_{\boldsymbol{\pi}}$, γ, and ρ proves (7). □

Proposition 1 bounds the target excess risk by four contributions: weighted source–target discrepancy, a uniform generalization-gap bound, regularization, and optimization error. Shifting weight toward contexts with smaller δ_{ks} can reduce the discrepancy term. However, the generalization term may also vary with the weight allocation.

Data-dependent transferability weights. For data-dependent weights, a sufficient condition is that the generalization-gap bound holds simultaneously for every weight vector in a fixed class Π. This class must contain all admissible weights that the learning procedure may return. Let $\varepsilon_{\mathrm{gen}} : \Pi \to [0, \infty)$ be a deterministic finite-valued function. Under the same fixed setup, suppose that, for some $\zeta \in (0, 1)$,

$$\mathbb{P}\left(\sup_{\mathbf{W} \in \mathcal{W}} |L_{\boldsymbol{\pi}}(\mathbf{W}) - R_{\boldsymbol{\pi}}(\mathbf{W})| \le \varepsilon_{\mathrm{gen}}(\boldsymbol{\pi}) \quad \text{for all } \boldsymbol{\pi} \in \Pi \right) \ge 1 - \zeta.$$

On this event, the proof of Proposition 1 applies to the learned weights $\widehat{\boldsymbol{\pi}} \in \Pi$ and any corresponding estimator satisfying (B.1). Thus, the same risk bound holds with $\boldsymbol{\pi} = \widehat{\boldsymbol{\pi}}$. For each fixed $\mathbf{W} \in \mathcal{W}$, $R_{\widehat{\boldsymbol{\pi}}}(\mathbf{W})$ denotes $R_{\boldsymbol{\pi}}(\mathbf{W})$ evaluated at the learned weights. It need not equal $\mathbb{E}[L_{\widehat{\boldsymbol{\pi}}}(\mathbf{W})]$ when the weights depend on the same data.

The simultaneous bound allows the weights to depend on the data; the argument does not establish that this condition holds for the full HiCT procedure.

It retains the assumptions of fixed hyperparameters, fixed representation maps, and a fixed source-context index set. Deriving explicit rates for $\varepsilon_{\mathrm{gen}}(\boldsymbol{\pi})$ requires additional assumptions on the loss class, the weight class Π, and the temporal dependence of the data.

Comparison with uniform weighting. A comparison with uniform weighting requires separate control of the weighting-parameter learning problem. Consider a common rolling-validation protocol with the same representation maps, source availability restrictions, and predictor-fitting procedure. If the empirical rolling-validation loss uniformly approximates its finite population expectation over a fixed class containing all possible learned rules and the uniform rule, and the learned rule's empirical loss exceeds that of uniform weighting by at most a comparison tolerance, then its population rolling-prediction risk exceeds that of uniform weighting by at most twice the uniform generalization-gap bound plus that tolerance.

This conditional comparison bounds possible risk deterioration; it does not guarantee an improvement over uniform weighting. It concerns the population counterpart of the rolling-validation objective and does not directly extend to the final refitted or target-adapted predictor.

Weight concentration and temporal dependence. To isolate the effect of weighting, consider scalar source summaries $Z_{ks} = \theta_0 + b_{ks} + \epsilon_{ks}$, where θ_0 is the target mean, b_{ks} is a fixed source shift, and the stacked noise vector $\boldsymbol{\epsilon}$ has zero mean and finite covariance Σ_ϵ. Hold the representations and normalized weights $\pi_{ks} = \alpha_k \beta_{s|k}$ fixed, and define $\widehat{\theta}_{\boldsymbol{\pi}} = \sum_{k,s} \pi_{ks} Z_{ks}$. Then

$$\mathbb{E}\Big[(\widehat{\theta}_{\boldsymbol{\pi}} - \theta_0)^2\Big] = \left(\sum_{k,s} \pi_{ks} b_{ks} \right)^2 + \boldsymbol{\pi}^\top \Sigma_\epsilon \boldsymbol{\pi}.$$

This identity follows by separating the mean shift from the zero-mean noise. For independent errors with common variance $\sigma^2 > 0$, the variance term reduces to

$$\boldsymbol{\pi}^\top \Sigma_\epsilon \boldsymbol{\pi} = \sigma^2 \sum_k \alpha_k^2 \sum_s \beta_{s|k}^2.$$

In particular, assigning all weight to a single source context with a small absolute mean shift can keep the squared bias small, but it makes $\sum_{k,s} \pi_{ks}^2 = 1$ and yields variance σ^2, the largest possible variance among normalized nonnegative weights in this independent, homoscedastic model.

Under the same noise assumptions, assigning positive weight to more than one context strictly reduces the variance relative to assigning all weight to a single context. Thus, concentrating on the source context with the smallest absolute mean shift need not minimize mean squared error.

With temporally dependent or overlapping contexts, cross-context error covariances also matter, so weight concentration alone does not determine the variance. This is an illustration for a weighted-mean estimator with fixed weights, not a risk bound for the full HiCT predictor. In particular, this calculation does not provide the generalization-gap bound required by Proposition 1. HiCT learns its weighting function from target validation loss, without explicitly minimizing this illustrative variance.

B.2 TARGET-SPECIFIC ADAPTATION

We next analyze the target-specific correction while holding the transferred estimator $\widetilde{\mathbf{W}}$ fixed. For this stage, L_0 is defined on the ambient parameter space $\mathcal{E}$, and the correction is optimized over $\mathcal{E}$. Let $\mathcal{J}_\Delta$ denote the regularization penalty on the correction. For $\lambda_\Delta \geq 0$, define the second-stage objective as

$$Q_{\mathrm{ad}}(\mathbf{\Delta}) = L_0\Big(\widetilde{\mathbf{W}} + \mathbf{\Delta}\Big) + \lambda_\Delta \mathcal{J}_\Delta(\mathbf{\Delta}).$$

The correction is obtained from

$$\widehat{\mathbf{\Delta}} \in \arg\min_{\mathbf{\Delta}\in\mathcal{E}} Q_{\mathrm{ad}}(\mathbf{\Delta}), \tag{B.4}$$

and the final estimator is

$$\widehat{\mathbf{W}} = \widetilde{\mathbf{W}} + \widehat{\mathbf{\Delta}}.$$

Proposition 2 (Stage 2: Empirical guarantee for target-specific adaptation)**.** *Suppose that $L_0(\widetilde{\mathbf{W}} + \mathbf{\Delta})$ and $\mathcal{J}_\Delta(\mathbf{\Delta})$ are finite, convex, and continuous functions of $\mathbf{\Delta}$ on $\mathcal{E}$. Assume that $\mathcal{J}_\Delta(\mathbf{\Delta}) \geq 0$ for all $\mathbf{\Delta} \in \mathcal{E}$, $\mathcal{J}_\Delta(\mathbf{0}) = 0$, and $\lambda_\Delta \geq 0$. Suppose that* (B.4) *admits a minimizer $\widehat{\mathbf{\Delta}}$. Then the following properties hold.*

(i) The regularized empirical target objective and the unpenalized loss do not increase relative to the zero correction:

$$L_0(\widehat{\mathbf{W}}) \leq L_0(\widehat{\mathbf{W}}) + \lambda_\Delta \mathcal{J}_\Delta(\widehat{\mathbf{\Delta}}) \leq L_0(\widetilde{\mathbf{W}}). \tag{B.5}$$

(ii) The zero correction is optimal if and only if

$$\mathbf{0} \in \partial L_0(\widetilde{\mathbf{W}}) + \lambda_\Delta \partial \mathcal{J}_\Delta(\mathbf{0}), \tag{B.6}$$

where ∂ denotes the convex subdifferential on $\mathcal{E}$. If Q_{ad} is strictly convex, its minimizer is unique. In this case, (B.6) *is equivalent to $\widehat{\mathbf{\Delta}} = \mathbf{0}$.*

The guarantee in part (i) concerns the empirical target loss used to estimate the correction. It applies to an exact minimizer of the second-stage objective and does not imply a reduction in population or test risk.

Proof of Proposition 2.

Proof. (i) The zero correction is feasible for (B.4). By the optimality of $\widehat{\mathbf{\Delta}}$ and the condition $\mathcal{J}_\Delta(\mathbf{0}) = 0$,

$$Q_{\mathrm{ad}}(\widehat{\mathbf{\Delta}}) \leq Q_{\mathrm{ad}}(\mathbf{0}) = L_0(\widetilde{\mathbf{W}}).$$

Substituting the definition of Q_{ad} and using $\widehat{\mathbf{W}} = \widetilde{\mathbf{W}} + \widehat{\mathbf{\Delta}}$ proves (B.5). Since $\lambda_\Delta \mathcal{J}_\Delta(\widehat{\mathbf{\Delta}}) \geq 0$, it also follows that

$$L_0(\widehat{\mathbf{W}}) \leq L_0(\widetilde{\mathbf{W}}).$$

(ii) The objective Q_{ad} is convex. The convex optimality condition therefore gives

$$\mathbf{0} \in \arg\min_{\mathbf{\Delta}\in\mathcal{E}} Q_{\mathrm{ad}}(\mathbf{\Delta}) \quad \Longleftrightarrow \quad \mathbf{0} \in \partial Q_{\mathrm{ad}}(\mathbf{0}).$$

The subdifferential sum rule applies because both terms in the objective are finite, convex, and continuous on $\mathcal{E}$. Together with translation by the fixed parameter $\widetilde{\mathbf{W}}$, it yields

$$\begin{aligned}\partial Q_{\mathrm{ad}}(\mathbf{0}) &= \partial_{\mathbf{\Delta}} \left[L_0(\widetilde{\mathbf{W}} + \mathbf{\Delta})\right]\Big|_{\mathbf{\Delta}=\mathbf{0}} + \lambda_\Delta \partial \mathcal{J}_\Delta(\mathbf{0}) \\ &= \partial L_0(\widetilde{\mathbf{W}}) + \lambda_\Delta \partial \mathcal{J}_\Delta(\mathbf{0}).\end{aligned}$$

Thus, the zero correction is optimal if and only if (B.6) holds.

Finally, strict convexity implies that the minimizer, whose existence is assumed, is unique. Under this additional condition, zero is optimal if and only if $\widehat{\mathbf{\Delta}} = \mathbf{0}$. □

Proposition 2 concerns the empirical target loss used to estimate the correction. The stated guarantee applies to an exact minimizer of the second-stage objective. It does not imply a reduction in population or future prediction risk. Under strict convexity, (B.6) characterizes when the transferred estimator remains unchanged.

Bias–variance trade-off in penalized target correction. To explain why the empirical guarantee in Proposition 2 need not imply lower prediction risk, consider one output coordinate with n target samples and p predictor coefficients, where $n \geq p \geq 1$. Let the fixed design $X \in \mathbb{R}^{n\times p}$ satisfy $X^\top X/n = I_p$, where I_p is the identity matrix, and let $\mathbf{y} = X\mathbf{w}_0 + \boldsymbol{\epsilon}$ where $\mathbf{w}_0 \in \mathbb{R}^p$ is the target coefficient vector and $\boldsymbol{\epsilon} \in \mathbb{R}^n$ has zero mean and finite covariance. Treat the transferred coefficient vector $\widetilde{\mathbf{w}}$ as fixed and define its remaining error by $\mathbf{d} = \mathbf{w}_0 - \widetilde{\mathbf{w}}$. Write $\boldsymbol{\xi} = X^\top\boldsymbol{\epsilon}/n$ and $V = \mathbb{E}\|\boldsymbol{\xi}\|_2^2$.

For a fixed penalty $\lambda = \lambda_\Delta > 0$, the penalized correction and its expected squared parameter error satisfy

$$\widehat{\boldsymbol{\delta}}_\lambda = \arg\min_{\boldsymbol{\delta}\in\mathbb{R}^p} \left\{ \frac{\|\mathbf{y} - X(\widetilde{\mathbf{w}} + \boldsymbol{\delta})\|_2^2}{2n} + \frac{\lambda}{2}\|\boldsymbol{\delta}\|_2^2 \right\} = \frac{\mathbf{d} + \boldsymbol{\xi}}{1+\lambda},$$
$$\mathbb{E}\|\widetilde{\mathbf{w}} + \widehat{\boldsymbol{\delta}}_\lambda - \mathbf{w}_0\|_2^2 = \frac{\lambda^2\|\mathbf{d}\|_2^2 + V}{(1+\lambda)^2}.$$

The closed-form correction follows from the ridge normal equations. Substituting it into the estimation error gives the risk identity because $\mathbb{E}\boldsymbol{\xi} = \mathbf{0}$. For a test covariate independent of the training noise, with second-moment matrix I_p, assume that the conditional mean of the test response follows the same linear model. Then the expected squared parameter error equals the expected squared prediction error relative to that mean.

Omitting correction leaves error $\|\mathbf{d}\|_2^2$. Hence the correction strictly improves this risk precisely when $V < (1+2\lambda)\|\mathbf{d}\|_2^2$. Increasing the penalty suppresses correction noise but also leaves more of the transfer error uncorrected. If $\mathbf{d} = \mathbf{0}$, correction cannot strictly reduce this risk and increases it whenever $V > 0$. The shrinkage here is induced by the ridge penalty, without an additional blend parameter. This calculation assumes a fixed transferred estimator and an identity Gram matrix. If the transferred estimator reuses the correction responses, a cross term can remain, and the displayed identity requires additional justification.

C Experimental Details

This section describes the datasets, group construction, and preprocessing used in the experiments in Section 4.

C.1 Datasets and Group Construction

The lengths reported in the dataset tables include training, validation, and test observations; in Section 3, T_0 denotes the target history available before the first test forecast.

Each benchmark contains a short-history target group and multiple source groups with longer histories. Group construction precedes forecasting, and source contexts are filtered by calendar-time availability at each target forecasting origin.

Preprocessing and reconstruction. Each variable is transformed as specified below and standardized using statistics fitted through the target training cutoff. The source transformations use only source observations available by that cutoff. These fitted transformations remain fixed during validation and testing. Source contexts are filtered by response availability at each forecasting origin.

For HiCT, the representation and source-to-target maps in Section 3.2 are fitted on training observations. The target decoder maps latent forecasts back through the target representation. For the PCA representation, this step applies the target loading matrix and restores the fitted centering vector. We evaluate MSE and MAE against the full standardized target observations, rather than their low-dimensional projections. Thus the decoder used for these metrics stops in standardized observation space; recovering original units would additionally invert standardization and the dataset-specific transformation. This representation step is specific to the relevant HiCT configurations and does not imply that every baseline uses PCA.

C.1.1 STOCK

The Stock dataset contains daily realized-volatility series from five equity sectors, obtained through Wharton Research Data Services (`https://wrds-www.wharton.upenn.edu`). Consumer Staples serves as the target group, with 56 stocks observed over 277 trading days. Financials, Health Care, Industrials, and Information Technology serve as the four source groups, with 134, 103, 112, and 126 stocks, respectively. Table 4 summarizes the group dimensions and observation periods.

Table 4: Realized-volatility group construction. N_k is the number of stocks and T_k is the number of common daily observations.

Sector	Role	N_k	T_k	Start date	End date
Consumer Staples	Target	56	277	2024-04-05	2025-08-15
Financials	Source	134	1,100	2021-01-25	2025-08-15
Health Care	Source	103	1,143	2020-09-28	2025-08-15
Industrials	Source	112	815	2022-04-06	2025-08-15
Information Technology	Source	126	760	2022-04-25	2025-08-15

We retain stocks with at least 60 intraday observations on each included trading day. For stock i on day t, the jth five-minute log return is

$$r_{i,t,j} = \log P_{i,t,j} - \log P_{i,t,j-1},$$

where $P_{i,t,j}$ denotes the price at the jth five-minute observation. Returns are constructed within the regular trading session from 9:30 to 16:00, excluding overnight returns. The daily realized variance is

$$RV_{i,t} = \sum_{j=1}^{M_{i,t}} r_{i,t,j}^2,$$

where $M_{i,t}$ is the number of available intraday returns. We split the target history chronologically into 166 training, 55 validation, and 56 test observations. We log-transform $RV_{i,t}$ and standardize each series using its training-period mean and standard deviation.

C.1.2 NY TAXI

We use hourly pickup counts derived from the NYC Taxi and Limousine Commission trip records (`https://www.nyc.gov/site/tlc/about/tlc-trip-record-data.page`) to evaluate demand forecasting with a short target history. The target and source series are aligned by their observation timestamps, with source availability enforced at each forecasting origin.

The panel records hourly pickup counts by taxi zone. Staten Island is the target group, and the other four boroughs provide 242 source zones (Table 5). The target history is split chronologically into 864 training, 288 validation, and 288 test observations. We apply a $\log(1 + x)$ transformation to accommodate zero counts, then standardize each zone series using its training-period mean and standard deviation.

Table 5: NY Taxi group construction. N_k is the number of zones and T_k is the number of hourly observations.

Borough	Role	N_k	T_k	Start date	End date
Staten Island	Target	20	1,440	2025-11-02	2025-12-31
Bronx	Source	43	8,760	2025-01-01	2025-12-31
Brooklyn	Source	61	8,760	2025-01-01	2025-12-31
Queens	Source	69	8,760	2025-01-01	2025-12-31
Manhattan	Source	69	8,760	2025-01-01	2025-12-31

C.1.3 PEMS-BAY

The PEMS-BAY panel contains traffic-speed measurements from 325 sensors sampled every five minutes during June 2017. The source dataset is available at `https://zenodo.org/records/5724362`. Each source group retains 8,640 observations, while the target group retains the final 2,016 observations. We project the sensor coordinates onto a local Cartesian coordinate system and apply capacity-constrained K-means to obtain five geographical groups of 65 sensors each. Clustering uses fixed initialization settings and does not depend on forecasting outcomes.

The group whose centroid is closest to the centroid of the complete sensor network is designated as the target group. The remaining groups are ordered by centroid longitude and used as source groups. The target history covers June 24–30, 2017, corresponding to seven days, while each source history covers June 1–30. We split the target history chronologically into 1,209 training, 403 validation, and 404 test observations. We apply the transformation $x \mapsto \log(1 + x)$ to each sensor series, followed by standardization using training-period statistics.

Table 6: PEMS-BAY group construction. Each geographical group contains 65 sensors.

Group	Role	N_k	T_k	Observation period
Region 5	Target	65	2,016	June 24–30, 2017
Regions 1–4, each	Source	65	8,640	June 1–30, 2017

C.1.4 ELECTRICITY

The Electricity panel contains hourly consumption series from 321 customers over 2012–2014, with 26,304 observations per customer. The source dataset is available at `https://zenodo.org/records/4656140`. The target group consists of the final 60 customers, T262–T321. We rank the remaining 261 customers by the distance of their standardized training-period trajectories from the mean target trajectory and divide this ranking into three groups of 87 customers.

The target retains the final 1,440 hourly observations, which are split chronologically into 864 training, 288 validation, and 288 test observations. No logarithmic transformation is applied, and each customer series is standardized using training-period statistics.

Source histories are aligned with the target by calendar time. At each forecasting origin, transfer is restricted to source contexts whose one-step-ahead responses are available.

Table 7: Electricity group construction. N_k is the number of customers and T_k is the number of hourly observations. Source groups are formed by training-period trajectory similarity; the target row retains the original customer identifiers.

Customers	Role	N_k	T_k	Start date	End date
T262–T321	Target	60	1,440	2014-11-02	2014-12-31
Group 1	Source	87	26,304	2012-01-01	2014-12-31
Group 2	Source	87	26,304	2012-01-01	2014-12-31
Group 3	Source	87	26,304	2012-01-01	2014-12-31

C.2 BASELINE CODE AND EVALUATION ADAPTATIONS

We adapt the baseline implementations to the target dimensions, chronological splits, preprocessing, and one-step-ahead rolling evaluation used in this study. Rolling evaluation advances the input context through the test period; it does not imply that every baseline is retrained at each origin. The target-only neural adapters select settings using validation data, fit before test evaluation, and use the resulting fixed predictors for successive test contexts. Test responses are not used for hyperparameter selection.

Public code. We use the following upstream architectures:

- PatchTST (Nie et al., 2023): https://github.com/yuqinie98/PatchTST.
- TimeMixer (Wang et al., 2024): https://github.com/kwuking/TimeMixer.
- AdaRNN (Du et al., 2021): https://github.com/jindongwang/transferlearning/tree/master/code/deep/adarnn.
- ROSE (Wang et al., 2025): https://github.com/decisionintelligence/ROSE.

Our adaptation and reproduction code is available at https://anonymous.4open.science/r/HiCT-6473. The upstream links identify model sources; the following implementation differences define the evaluated variants.

Source use and implementation scope. PatchTST and TimeMixer use only the target panel. The AdaRNN adapter uses the upstream recurrent architecture on the target panel. The ROSE adapter uses the upstream Transformer encoder and prediction head for supervised one-step-ahead pre-training across the available source groups, followed by target fine-tuning. It learns the source pre-training checkpoint within our data splits instead of loading an external multi-domain checkpoint. The number of source pre-training and target fine-tuning epochs is selected using target validation loss. This is a local adaptation to our forecasting protocol, rather than a reproduction of the original ROSE pre-training procedure.

DTr-CNN is a local, simplified implementation guided by Ye & Dai (2021); an official author-maintained repository was not identified. It selects a source using DTW distances between standardized group-mean series, pre-trains a shared CNN on the selected source, and adapts it to target data.

C.3 Runtime Measurement

We measure runtime on Stock over 56 test forecasting origins, running one benchmark at a time under an eight-thread CPU budget. Table 8 separates fitting from forecasting. Hyperparameter and epoch selection, preprocessing, file I/O, startup, and independent prediction verification are excluded.

Train includes parameter learning and predictor fitting. For HiCT-VAR, it also includes source-weight generation and predictor refitting at all test origins; it is therefore not solely offline training time. ROSE includes source pre-training and target fine-tuning. The updated AdaRNN measurement includes temporal domain characterization, warm-up, temporal distribution matching, and boosting. *Forecast* measures forward prediction and target-space reconstruction with fitted predictors; it excludes the rolling refits already counted in Train. Total time includes both stages.

Table 8: Runtime breakdown on Stock in seconds.

Method	Train	Forecast	Total
PatchTST	2940.3	2.379	2942.6
TimeMixer	29.5	0.058	29.5
AdaRNN	4.3	0.027	4.3
DTr-CNN	56.9	0.020	56.9
ROSE	526.8	0.718	527.5
HiCT-VAR	9.3	0.001	9.3
HiCT-RNN	10.9	0.001	10.9

C.4 Validation Sensitivity and Parameter Selection

Figure 4 reports the validation curves for the sweeps in Figure 3. Validation selects $(L, h, d) = (3, 2, 48)$. The validation curves favor short contexts and a small source stride. Validation error rises again at the largest latent dimension. These selections need not minimize test error: the densest source sampling gives lower test error despite higher validation error.

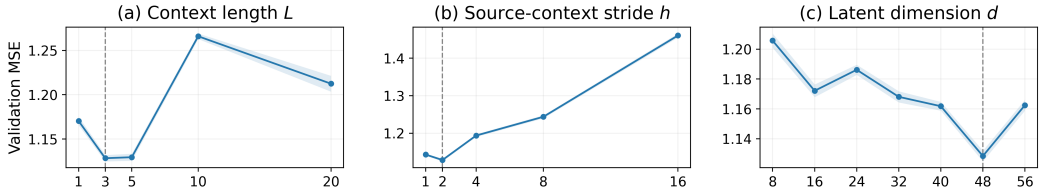

Figure 4: Validation sensitivity on Stock for the same sweeps as Figure 3. Curves and shading show the mean and one sample standard deviation over five runs; dashed lines mark validation-selected values.

D VISUALIZATION OF TRANSFERABILITY WEIGHTS

Figure 5 summarizes the mean joint weights $\pi_{ks} = \alpha_k \beta_{s|k}$ over target test forecasting origins. Rows correspond to source groups, and columns to historical contexts ordered from older to more recent within each group, with the most recent contexts placed at the right edge. Gray cells indicate padding for shorter source histories. These averages summarize source allocation; they do not display its variation from one forecasting origin to the next.

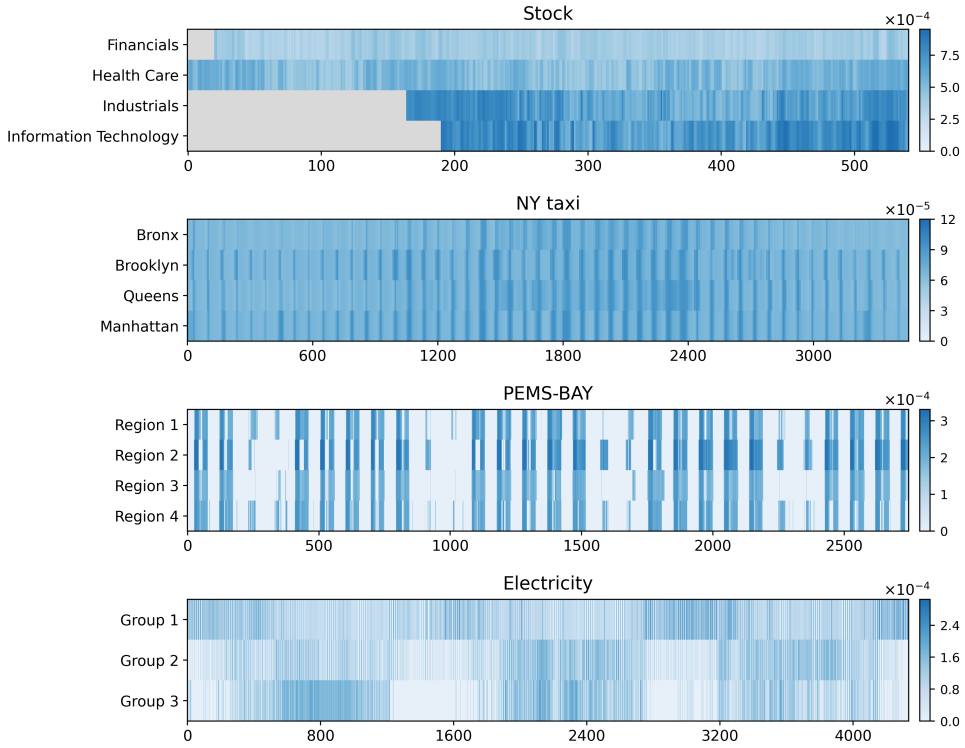

Figure 5: Heatmap visualization of transferability weights for Stock, NY Taxi, PEMS-BAY, and Electricity. Colors show the mean joint weight $\alpha_k \beta_{s|k}$ over test forecast origins. Rows represent source groups, and columns index right-aligned source contexts in their original chronological order; gray indicates padding. Each dataset uses its own color scale.