# CROSS-WINDOW RELATIONS FOR TRANSFERABILITY ESTIMATION IN TIME SERIES FORECASTING

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Time series forecasting often benefits from transferring knowledge from auxiliary datasets, yet transfer performance depends strongly on which source dataset is selected. Existing transferability estimators mainly assess source–target compatibility at the level of individual forecasting windows, overlooking the local evolution between successive overlapping windows. In this paper, we introduce the Cross-Window Relation Estimator (CWRE) for source dataset selection in time series forecasting. CWRE evaluates each candidate source using its frozen source-pretrained model on the target dataset, jointly measuring how well the learned representations explain future sequences and how well representation changes explain the corresponding future changes across adjacent windows. To reduce noise amplification introduced by differencing, CWRE separates trend and seasonal components, applying cross-window evaluation to the smoother trend component while retaining complementary predictive information through instance-wise evaluation. These relations are integrated through closed-form ridge regression, enabling efficient scoring without target-domain fine-tuning. We further evaluate transferability under a cross-architecture protocol, where scoring and downstream forecasting models are distinct. Experiments across multiple target datasets, architectures, and missing-observation settings show that CWRE achieves higher overall agreement with downstream transfer-performance rankings than distribution-alignment, task-transferability, and time-series-specific baselines. The results demonstrate that cross-window evolution provides complementary information for identifying transferable source datasets beyond conventional instance-wise compatibility. To ease reproducibility, we release our implementation code online `https://anonymous.4open.science/r/CWRE-submission-3FC1/`.

## 1 INTRODUCTION

Time series forecasting aims to predict future observations from historical sequences and has been widely applied in energy, transportation, finance, *etc.* (Lim & Zohren, 2021; Chen et al., 2023; Kim et al., 2025). In many practical scenarios, however, the target dataset may provide limited or incomplete observations, making it difficult to learn a reliable forecasting model solely from target-domain data.

Transfer learning provides a natural way to supplement target information with knowledge learned from related source datasets (Zhang et al., 2023). Its effectiveness depends critically on source selection, since an unsuitable source may lead to negative transfer (Wang et al., 2019). For example, Dong et al. (2023) report higher forecasting errors after pre-training on Weather and fine-tuning on ETTh1 than when training from scratch. Exhaustively fine-tuning models from every candidate source on the target dataset is computationally expensive. We therefore study *transferability estimation for source dataset selection*: ranking candidate source datasets according to their downstream transfer potential without target-domain fine-tuning during scoring.

Existing transferability estimators mainly assess source suitability from two perspectives. *Distribution alignment metrics*, such as CORAL (Sun et al., 2017), MMD (Gretton et al., 2012), CMD (Zellinger et al., 2017), and DTW (Ye & Dai, 2021), estimate source–target compatibility by measuring their discrepancy in a common representation space. *Task transferability metrics*, including LEEP (Nguyen et al., 2020), LogME (You et al., 2021), LinMSE (Nguyen et al., 2023), and ETran (Gholami et al.,



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.

2023), instead evaluate whether source-learned representations are informative for the target task. For time series, TEMPLATE (Zhang et al., 2025) further incorporates temporal dependencies, temporal patterns, and task adaptation into transferability estimation. Despite these advances, these methods predominantly evaluate transferability at the level of individual forecasting windows, either through source–target representation compatibility or through the relation between each target-window representation and its forecasting target.

This instance-wise perspective overlooks an important property introduced by sliding-window sampling. Adjacent forecasting windows share substantial observations and constitute successive states of the same temporal process. As a window advances, its historical context and corresponding future sequence evolve jointly. This suggests an additional criterion for source transferability: whether changes in source-learned representations correspond to the associated changes in future sequences. Such cross-window correspondence complements instance-wise compatibility by characterizing local evolution across neighboring forecasting windows.

To incorporate cross-window evidence into transferability estimation, we propose the Cross-Window Relation Estimator (CWRE). For each candidate source, we use its frozen pre-trained model to extract representations of target forecasting windows. Specifically, (1) Cross-Window Relations. We jointly assess how well these representations explain the corresponding future sequences and how well representation changes explain future changes across adjacent windows; (2) Trend-Seasonal Decomposition. Recognizing that direct differencing adjacent windows can amplify noise, we implement trend-season decomposition; and (3) Ridge Regression-Based Scoring. We formulate the scoring objective as ridge regression with a closed-form solution to prevent the linear mapping from reducing fitting errors through excessively large weights.

We further consider whether an estimated source ranking remains useful beyond the architecture used for scoring. As shown in Figure 1, a source selected using one representation model may not exhibit the same transfer behavior under another forecasting architecture. We therefore adopt a cross-architecture evaluation protocol, where source rankings are estimated using dedicated scoring architectures and validated against reference rankings obtained from distinct downstream forecasting models. Experiments across multiple target datasets, missing rates, scoring architectures, and downstream forecasting models show that CWRE achieves higher overall agreement with downstream transfer-performance rankings than existing distribution-alignment, task-transferability, and time-series-specific baselines. The results further confirm that cross-window correspondence provides complementary information beyond conventional instance-wise evaluation.

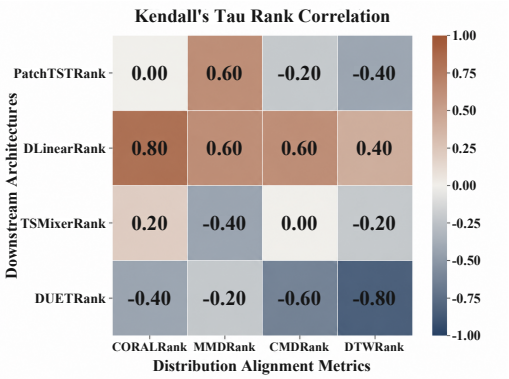



Figure 1: Agreement between distribution-alignment rankings and downstream transfer rankings. Columns correspond to rankings produced by CORAL, MMD, CMD, and DTW using linear-model embeddings, and rows to ground-truth rankings from downstream forecasting models. Each cell reports Kendall's $\tau$.

Our contributions are summarized as follows:

- We identify *cross-window evolution correspondence* as a complementary signal for source-dataset transferability in time series forecasting, extending conventional instance-wise evaluation to the local evolution between adjacent forecasting windows.
- We propose CWRE, which jointly evaluates instance-wise and cross-window relations under a shared representation-to-future mapping, together with trend–seasonal decomposition and closed-form ridge scoring for robust and efficient transferability estimation.
- We establish a cross-architecture evaluation protocol and demonstrate that CWRE provides more reliable source rankings across target datasets, scoring architectures, downstream forecasting architectures, and both complete and incomplete target observations.

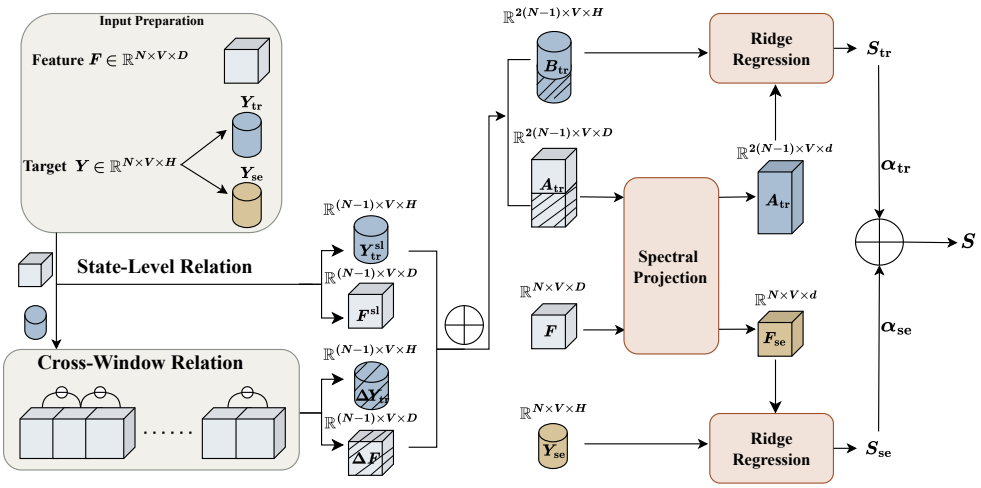



Figure 2: Framework overview of Cross-Window Relation Estimator. Given frozen source-pretrained representations $F$ and future sequences $Y$, CWRE: (a) decomposes $Y$ into trend and seasonal components; (b) evaluates the trend component using state-level relation and a cross-window relation that associates $\Delta F$ with the corresponding trend future-sequence change $\Delta Y_{\text{tr}}$ across adjacent windows, followed by concatenation, spectral projection, and ridge regression to obtain $S_{\text{tr}}$; (c) evaluates the seasonal component via spectral projection and ridge regression to obtain $S_{\text{se}}$; and (d) fuses the two scores with energy-based weights to yield the transferability score $S$.

## 2 Methodology

### 2.1 Task Definition

We study transferability estimation for source dataset selection in time series forecasting. Given $K$ candidate source datasets $\{\mathcal{D}_k^{\text{src}}\}_{k=1}^K$, we pretrain a fixed scoring architecture on each source and freeze the resulting encoder $\phi_k$. Let the chronologically ordered target forecasting windows be $\{(X_t, Y_t)\}_{t=1}^N$. For the $k$-th source, the frozen encoder produces $F_{k,t} = \phi_k(X_t)$, where $F_k \in \mathbb{R}^{N\times V\times D}$ and $Y \in \mathbb{R}^{N\times V\times H}$ denote the representations and future sequences. Here, $N$, $V$, $D$, and $H$ denote the number of forecasting windows, the number of variables, the representation dimension, and the forecasting horizon. The transferability estimator $\mathcal{T}$ assigns each candidate source a scalar score $S_k = \mathcal{T}(F_k, Y)$, where larger $S_k$ indicates higher estimated transferability, and we rank sources in descending order of $S_k$. Hereafter, we omit $k$ and write $F = F_k$ and $S = S_k$ for one candidate source.

### 2.2 Overall Framework

To estimate the transferability for source dataset selection in time series forecasting, we propose the Cross-Window Relation Estimator (CWRE), illustrated in Figure 2. CWRE evaluates frozen representations against their corresponding future sequences through three components: (1) Cross-Window Relations. We formulate the state-level relation as instance-wise evaluation, assessing whether a representation explains the corresponding future sequence, and complement it with cross-window relations that assess whether representation changes explain the corresponding future changes across adjacent windows. (2) Trend-Seasonal Decomposition. To mitigate noise amplified by differencing, we evaluate cross-window relations on the smoother trend component while retaining complementary predictive information through state-level evaluation of the seasonal component. (3) Ridge Regression-Based Scoring. To prevent fitting-error reduction through excessively large weights, we use ridge regression to constrain the linear mappings used for both relations. Its closed-form solution enables efficient scoring across candidate sources. The branch residual scores are fused using the energies of the corresponding future-sequence components.

### 2.3 CROSS-WINDOW RELATIONS

To complement instance-wise evaluation with successive-window information, we introduce cross-window difference relations that assess whether representation changes explain corresponding future changes.

We use adjacent windows by default ($s = 1$) and omit $s$ hereafter. We temporally align $F$ and $Y$, using superscripts $\mathrm{sl}$ and $\mathrm{cw}$ for the state-level and cross-window blocks. To maintain consistency with the difference samples, we retain the first $N - 1$ windows:

$$F^{\mathrm{sl}} = F_{1:N-1}, \qquad Y^{\mathrm{sl}} = Y_{1:N-1}. \tag{1}$$

Preserving temporal order, we construct the adjacent-window difference blocks:

$$\begin{aligned} \Delta F_t &= F_{t+1} - F_t, \qquad F^{\mathrm{cw}} = [\Delta F_1, \ldots, \Delta F_{N-1}], \\ \Delta Y_t &= Y_{t+1} - Y_t, \qquad Y^{\mathrm{cw}} = [\Delta Y_1, \ldots, \Delta Y_{N-1}]. \end{aligned} \tag{2}$$

The two samples $(F^{\mathrm{sl}}, Y^{\mathrm{sl}})$ and $(F^{\mathrm{cw}}, Y^{\mathrm{cw}})$ evaluate prediction within windows and changes across adjacent windows, respectively. Preserving temporal order makes each difference correspond to a local transition between adjacent windows, providing complementary evidence to instance-wise evaluations.

To limit fitting capacity, we use a linear mapping $W$ from representations to future sequences. Under the time-independent mapping $Y_t \approx F_t W$, its cross-window difference satisfies:

$$\Delta Y_t = Y_{t+1} - Y_t \approx (F_{t+1} - F_t)W = \Delta F_t W. \tag{3}$$

We therefore share $W$ across both relations, jointly constraining the state-level mapping and the calibrated cross-window mapping. Appendix A derives a lower bound on adaptation error from baseline residual variation, conditional on bounded temporal variation in prediction corrections.

### 2.4 TREND-SEASONAL DECOMPOSITION

To minimize high-frequency noise amplified by differencing (Appendix B.2), we evaluate both relations on the trend component and retain the state-level relation for the seasonal component.

For the future sequence, we apply a moving average decomposition as follows:

$$Y_{\mathrm{tr}} = MA_q(Y), \qquad Y_{\mathrm{se}} = Y - Y_{\mathrm{tr}}, \tag{4}$$

where $MA_q$ denotes a moving average of window length $q$ over the forecast horizon, yielding the trend component $Y_{\mathrm{tr}}$ and the non-trend residual, denoted as the seasonal component $Y_{\mathrm{se}}$. To construct the trend branch, we apply the level alignment and cross-window difference construction in Equations 1–2 to $Y_{\mathrm{tr}}$, obtaining $Y_{\mathrm{tr}}^{\mathrm{sl}}$ and $Y_{\mathrm{tr}}^{\mathrm{cw}}$. The seasonal branch uses $Y_{\mathrm{se}}$ without cross-window differencing.

For the frozen representations, we construct separate projection spaces for the two branches by merging the sample and variable dimensions while preserving the representation dimension. Before constructing the projection spaces, $F^{\mathrm{sl}}$, $F^{\mathrm{cw}}$, and $F$ are normalized separately. The corresponding future components are likewise normalized and reshaped using the same sample-variable ordering.

The feature-selection ratios $\rho_{\mathrm{tr}}$ and $\rho_{\mathrm{se}}$ control the numbers of eigen-directions retained for the trend and seasonal branches, respectively. We define $k_{\mathrm{tr}} = \max\{1, \lfloor D\rho_{\mathrm{tr}} \rfloor\}, k_{\mathrm{se}} = \lfloor D\rho_{\mathrm{se}} \rfloor$.

For the trend branch, we construct $G_{\mathrm{tr}} = (F^{\mathrm{sl}})^\top F^{\mathrm{sl}} + (F^{\mathrm{cw}})^\top F^{\mathrm{cw}}$. Let $\mathbf{v}_i^{\mathrm{tr}}$ be the eigenvectors ordered by decreasing eigenvalue, and define $P_{\mathrm{tr}} = [\mathbf{v}_i^{\mathrm{tr}}]_{i=1}^{k_{\mathrm{tr}}}$. We apply the same projection to both the representation block and the difference block:

$$F_{\mathrm{tr}}^{\mathrm{sl}} = F^{\mathrm{sl}} P_{\mathrm{tr}}, \qquad F_{\mathrm{tr}}^{\mathrm{cw}} = F^{\mathrm{cw}} P_{\mathrm{tr}}. \tag{5}$$

For the seasonal branch, we construct $G_{\mathrm{se}} = F^\top F$ and similarly order its eigenvectors $\mathbf{v}_i^{\mathrm{se}}$ by decreasing eigenvalue. Let $k_{\mathrm{end}} = \min\{D, k_{\mathrm{tr}} + k_{\mathrm{se}}\}$. The seasonal projection is defined as $P_{\mathrm{se}} = [\mathbf{v}_i^{\mathrm{se}}]_{i=k_{\mathrm{tr}}+1}^{k_{\mathrm{end}}}$. The seasonal representation is then $F_{\mathrm{se}} = FP_{\mathrm{se}}$. Since the two projections are derived from different Gram matrices, they do not form complementary partitions of a shared eigenbasis.

### 2.5 RIDGE REGRESSION-BASED SCORING

To evaluate these relationships without relying on excessively large linear weights to reduce the fitting error, we use ridge regression to measure how well frozen representations explain future sequences under a regularized linear mapping. Specifically, the trend branch jointly fits the state-level and cross-window relationships with shared linear weights, while the seasonal branch fits the seasonal future sequence with independent linear weights.

In the trend branch, for $\omega > 0$, we apply $\omega$ only to the standardized trend-difference targets, leaving the difference features unchanged. The shared weights are obtained by:

$$W_{\mathrm{tr}}^{*} = \arg\min_{W} \left\| Y_{\mathrm{tr}}^{\mathrm{sl}} - F_{\mathrm{tr}}^{\mathrm{sl}} W \right\|_F^2 + \left\| \omega Y_{\mathrm{tr}}^{\mathrm{cw}} - F_{\mathrm{tr}}^{\mathrm{cw}} W \right\|_F^2 + \lambda \left\| W \right\|_F^2 . \tag{6}$$

Here, $\lambda > 0$ controls the ridge regularization strength. The first term measures the fitting error of the state-level relation in the trend branch, while the second term measures the fitting error of the cross-window difference relation. The third term regularizes the magnitude of the linear weights. Both terms share the same $W$, coupling state-level fitting with the calibrated cross-window relation. We stack the two relations row-wise: $A_{\mathrm{tr}} = \begin{bmatrix} F_{\mathrm{tr}}^{\mathrm{sl}} \\ F_{\mathrm{tr}}^{\mathrm{cw}} \end{bmatrix}, B_{\mathrm{tr}} = \begin{bmatrix} Y_{\mathrm{tr}}^{\mathrm{sl}} \\ \omega Y_{\mathrm{tr}}^{\mathrm{cw}} \end{bmatrix}$.

Subsequently, the closed-form solution for the linear weights is given by:

$$W_{\mathrm{tr}}^{*} = \left( A_{\mathrm{tr}}^{\top} A_{\mathrm{tr}} + \lambda I \right)^{-1} A_{\mathrm{tr}}^{\top} B_{\mathrm{tr}}. \tag{7}$$

The closed-form solution avoids iterative optimization of the linear weight, enabling efficient evaluation across multiple candidate sources. For the detailed derivation, please refer to Appendix B.1.

The trend branch score is defined as the corresponding negative average residual:

$$S_{\mathrm{tr}} = -\frac{\| B_{\mathrm{tr}} - A_{\mathrm{tr}} W_{\mathrm{tr}}^{*} \|_F^2}{2(N-1)VH}. \tag{8}$$

The seasonal branch uses only the state-level relation with an independent mapping:

$$\begin{aligned} W_{\mathrm{se}}^{*} = \arg\min_{W} \| Y_{\mathrm{se}} - F_{\mathrm{se}} W \|_F^2 + \lambda \| W \|_F^2 = \left( F_{\mathrm{se}}^{\top} F_{\mathrm{se}} + \lambda I \right)^{-1} F_{\mathrm{se}}^{\top} Y_{\mathrm{se}}, \\ S_{\mathrm{se}} = -\frac{\| Y_{\mathrm{se}} - F_{\mathrm{se}} W_{\mathrm{se}}^{*} \|_F^2}{NVH}. \end{aligned} \tag{9}$$

The fusion weights are computed from the pre-normalization energies of the target components:

$$\alpha_{\mathrm{tr}} = \frac{\| Y_{\mathrm{tr}} \|_{\mathrm{F}}^2}{\| Y_{\mathrm{tr}} \|_{\mathrm{F}}^2 + \| Y_{\mathrm{se}} \|_{\mathrm{F}}^2 + \epsilon}, \qquad \alpha_{\mathrm{se}} = 1 - \alpha_{\mathrm{tr}}, \qquad \epsilon = 10^{-8}. \tag{10}$$

The final transferability score of CWRE is: $S = \alpha_{\mathrm{tr}} S_{\mathrm{tr}} + \alpha_{\mathrm{se}} S_{\mathrm{se}}$.

Since all candidate sources share the same target dataset and decomposition, the fusion weight is identical across candidates. Applying the procedure to all candidates yields $\{S_k\}_{k=1}^{K}$, which are ranked in descending order. Higher scores indicate smaller joint fitting residuals and thus higher estimated transferability.

## 3 EXPERIMENTS

### 3.1 EXPERIMENTAL SETUP

**Datasets.** We use ETTh1, ETTm1, ETTm2, Electricity, Exchange, Traffic, and Weather, taking ETTh1, ETTm1, and Exchange as targets and the remaining six datasets as sources for each target (statistics in Appendix C.1.1).

**Validation protocol.** We use PatchTST (Nie et al., 2023) and TimeEmb (Xia et al., 2025) for scoring, and iTransformer (Liu et al., 2024), DUET (Qiu et al., 2025), and FilterNet (Yi et al., 2024) for

Table 1: Cross-architecture ranking agreement under complete observations. $\tau_w$: mean $\pm$ sample standard deviation across three downstream architectures; $W$: concordance among their rankings and the score ranking. Higher is better. Bold/underline mark best/second-best results per target and scoring architecture, with ties retained.

| Score architecture | Method | ETTh1 | | ETTm1 | | Exchange | |
|---|---|---|---|---|---|---|---|
| | | $\tau_w$ | $W$ | $\tau_w$ | $W$ | $\tau_w$ | $W$ |
| PatchTST | CORAL | $\underline{0.1088} \pm 0.2438$ | 0.3857 | $0.2036 \pm 0.4940$ | $\underline{0.2000}$ | $0.1551 \pm 0.4563$ | 0.2643 |
| | MMD | $-0.3651 \pm 0.2830$ | 0.1571 | $\mathbf{0.2308} \pm 0.5376$ | **0.2286** | $0.1202 \pm 0.4923$ | 0.2286 |
| | CMD | $-0.0458 \pm 0.3208$ | 0.3143 | $0.1415 \pm 0.5923$ | 0.1571 | $0.0086 \pm 0.3691$ | 0.1786 |
| | DTW | $0.0848 \pm 0.3042$ | 0.3857 | $0.2036 \pm 0.4940$ | $\underline{0.2000}$ | $-0.1048 \pm 0.3392$ | 0.1357 |
| | LogME | $-0.2871 \pm 0.5180$ | 0.1857 | $-0.0975 \pm 0.3408$ | 0.0857 | $0.2875 \pm 0.4390$ | 0.3286 |
| | LinMSE | $-0.1460 \pm 0.4904$ | 0.3143 | $-0.2336 \pm 0.3925$ | 0.0357 | $0.0177 \pm 0.3263$ | 0.2500 |
| | ETran | $-0.2313 \pm 0.3955$ | 0.2214 | $0.1451 \pm 0.5932$ | 0.1929 | $0.2957 \pm 0.0727$ | 0.3214 |
| | TEMPLATE | $0.0608 \pm 0.6346$ | $\underline{0.4143}$ | $0.0082 \pm 0.5927$ | 0.1214 | $\underline{0.3116} \pm 0.2529$ | $\underline{0.3571}$ |
| | **CWRE** | $\mathbf{0.4807} \pm 0.3652$ | **0.5071** | $\underline{0.2281} \pm 0.4459$ | **0.2286** | $\mathbf{0.3891} \pm 0.0628$ | **0.3786** |
| TimeEmb | CORAL | $0.0721 \pm 0.2199$ | 0.3786 | $0.0830 \pm 0.3533$ | 0.1929 | $-0.1374 \pm 0.2390$ | 0.2214 |
| | MMD | $-0.4168 \pm 0.3783$ | 0.1857 | $0.1243 \pm 0.5355$ | 0.1643 | $\mathbf{0.3388} \pm 0.1409$ | **0.3429** |
| | CMD | $0.0590 \pm 0.0517$ | 0.3786 | $\underline{0.1737} \pm 0.5957$ | $\underline{0.2286}$ | $0.0168 \pm 0.2694$ | 0.2643 |
| | DTW | $0.0921 \pm 0.3027$ | 0.4214 | $0.1492 \pm 0.4085$ | 0.1714 | $-0.1107 \pm 0.3609$ | 0.2286 |
| | LogME | $0.2993 \pm 0.5271$ | 0.5143 | $-0.0957 \pm 0.6681$ | 0.0714 | $0.0517 \pm 0.5910$ | 0.2643 |
| | LinMSE | $0.2757 \pm 0.2725$ | 0.4929 | $0.1202 \pm 0.5229$ | 0.2000 | $-0.2939 \pm 0.4327$ | 0.1429 |
| | ETran | $\underline{0.4753} \pm 0.2484$ | $\underline{0.5429}$ | $0.1202 \pm 0.5229$ | 0.2000 | $0.2785 \pm 0.3668$ | 0.3071 |
| | TEMPLATE | $-0.4159 \pm 0.5425$ | 0.1643 | $0.0603 \pm 0.2196$ | 0.1643 | $-0.2331 \pm 0.6008$ | 0.0929 |
| | **CWRE** | $\mathbf{0.6032} \pm 0.2612$ | **0.5786** | $\mathbf{0.2141} \pm 0.2078$ | **0.2500** | $\underline{0.3224} \pm 0.4717$ | $\underline{0.3143}$ |

downstream evaluation. Each architecture is pretrained separately on each source. Frozen source-pretrained scoring models are evaluated on the target training split without target-domain fine-tuning. For each source, the maximum CWRE score over $(\omega, \rho_{\mathrm{tr}}, \rho_{\mathrm{se}})$ is used, with all other parameters fixed. Downstream models are fine-tuned on the target task. All models use a look-back window of 512 and a forecasting horizon of 96; further details are provided in Appendix C.1.2.

**Baselines.** We compare CWRE with distribution alignment metrics, including CORAL (Sun et al., 2017), MMD (Gretton et al., 2012), CMD (Zellinger et al., 2017), and DTW (Ye & Dai, 2021); task transferability metrics, including LogME (You et al., 2021), LinMSE (Nguyen et al., 2023), and ETran (Gholami et al., 2023); and the time series transferability metric TEMPLATE (Zhang et al., 2025). For distribution alignment metrics, we use the smaller source-target discrepancy from the source-specific and shared-reference protocols, where the latter uses an ETTh2-pretrained extractor. CWRE, LogME, LinMSE, ETran, and TEMPLATE use candidate-source-pretrained extractors. Details are provided in Appendix C.1.3.

**Evaluation Metrics.** Weighted Kendall's $\tau_w$ (Vigna, 2015) measures agreement with each downstream ranking, emphasizing top-ranked sources; we report its mean and sample standard deviation across the three downstream architectures. Kendall's $W$ (Zhang & Hardt, 2024) measures joint concordance between the score ranking and the three downstream rankings. Further averaging across scoring architectures is stated where applicable. Higher is better. Implementation settings are provided in Appendix C.1.4. Metric aggregation is detailed in Appendix C.2.

### 3.2 RQ1: Transferability Estimation Accuracy

We first evaluate CWRE under complete target observations in Table 1 to answer RQ1: *How accurately does CWRE estimate source-dataset transferability?*

**(a) Overall ranking.** Averaged equally over three targets and two scoring architectures, CWRE achieves the highest overall $\tau_w/W$ of $0.3729/0.3762$, exceeding the strongest baseline, ETran, by $0.1924/0.0786$. CWRE gains in both $\tau_w$ and $W$ indicate stronger agreement with individual downstream rankings and their joint concordance, rather than an advantage tied to a single evaluator.

Table 2: Ablation study with PatchTST as the scoring architecture. Results are reported using weighted Kendall's $\tau_w$ and Kendall's $W$. Avg. denotes the average over the three target datasets.

| Ablation | ETTh1 | | ETTm1 | | Exchange | | Avg. | |
|---|---|---|---|---|---|---|---|---|
| | $\tau_w$ | $W$ | $\tau_w$ | $W$ | $\tau_w$ | $W$ | $\tau_w$ | $W$ |
| **CWRE** | **0.4807** | **0.5071** | **0.2281** | **0.2286** | **0.3891** | **0.3786** | **0.3660** | **0.3714** |
| w/o Cross-Window | 0.1034 | 0.3571 | 0.0603 | 0.1643 | 0.2803 | 0.2857 | 0.1480 | 0.2690 |
| w/o Instance-wise | 0.4807 | 0.5071 | 0.0535 | 0.1714 | -0.1855 | 0.1786 | 0.1162 | 0.2857 |
| w/o Trend–Seasonal | 0.3528 | 0.4429 | 0.0771 | 0.1643 | 0.2313 | 0.3071 | 0.2204 | 0.3048 |
| w/o Temporal Order | 0.2766 | 0.4214 | 0.0603 | 0.1643 | 0.2653 | 0.2929 | 0.2007 | 0.2929 |
| w/o Shared Mapping | 0.3020 | 0.4429 | 0.0603 | 0.1643 | 0.2553 | 0.3000 | 0.2059 | 0.3024 |
| w/ Equal Weighting | 0.4807 | 0.5071 | 0.1560 | 0.2000 | 0.3891 | 0.3786 | 0.3419 | 0.3619 |

**(b) Comparison with existing methods.** Among distribution alignment metrics, CORAL achieves the highest overall average of $0.0809/0.2738$, still below CWRE. Distribution metrics measure source–target similarity without directly assessing whether source-pretrained representations support the target forecasting task, limiting their correspondence with downstream transfer performance. Table 6 compares source-specific, shared-reference, and combined distribution-alignment results with CWRE under the same complete-observation setting as Table 1. CWRE also outperforms LogME, LinMSE, ETran, and TEMPLATE on both metrics in all six target–architecture combinations. Unlike criteria based only on instance-wise task compatibility or representation statistics, CWRE also evaluates cross-window correspondence between representation and future changes, providing complementary evidence for source ranking.

**(c) Consistency across targets and scoring architectures.** Averaged over scoring architectures, CWRE achieves the highest $\tau_w$ on ETTh1, ETTm1, and Exchange. Consistent gains across targets and scoring architectures suggest that the benefit is not tied to a particular target distribution or representation space.

**(d) Exceptions.** CWRE is not uniformly best in every configuration. MMD slightly exceeds it on ETTm1 with PatchTST in $\tau_w$, with equal $W = 0.2286$, and also performs better on Exchange with TimeEmb. These cases show that source–target distribution similarity can still be informative for specific target–representation combinations. However, this advantage does not persist across the other settings, while CWRE maintains stronger overall agreement.

Overall, CWRE demonstrates effective and consistent source ranking across different targets and scoring architectures, with only configuration-specific exceptions.

### 3.3 RQ2: Effectiveness of Cross-Window Modeling

To answer RQ2: *Does cross-window modeling provide information beyond instance-wise transferability estimation*, Table 2 compares six variants under complete observations using PatchTST and the three downstream evaluators.

**(a) Complementary relations.** Removing cross-window relations lowers average $\tau_w/W$ from $0.3660/0.3714$ to $0.1480/0.2690$, with both metrics declining on every target. Removing instance-wise evaluation leaves ETTh1 unchanged but reduces $\tau_w$ from $0.2281$ to $0.0535$ on ETTm1 and from $0.3891$ to $-0.1855$ on Exchange. Thus, cross-window relations provide complementary ranking information, while instance-wise evaluations remain important for source ranking.

**(b) Temporal correspondence and shared mapping.** Shuffling temporal order or using independent mappings lowers average $\tau_w/W$ to $0.2007/0.2929$ and $0.2059/0.3024$, respectively. On ETTm1, both variants produce the same source ranking as removing cross-window relations, while the full method achieves $0.2281/0.2286$, supporting both temporal correspondence and shared mapping.

**(c) Decomposition and fusion.** Removing decomposition reduces average $\tau_w/W$ to $0.2204/0.3048$, with both metrics decreasing on all targets. Equal weighting affects only ETTm1, reducing $0.2281/0.2286$ to $0.1560/0.2000$, indicating that adaptive fusion is most useful when branch weighting changes the source ordering.

We further verify the noise effect introduced by cross-window differencing. The results show that differencing approximately doubles the injected noise, while low-rank projection substantially reduces

Table 3: Transferability estimation at MR $= 0.3$ and $0.6$. For each MR and score architecture, $\overline{\tau}_w$ and $\overline{W}$ denote averages across the three target datasets, with $\overline{\tau}_w$ additionally averaged over three evaluator architectures. Best and second-best results are bold and underlined, respectively.

| Method | MR $= 0.3$ | | | | MR $= 0.6$ | | | |
|---|---|---|---|---|---|---|---|---|
| | PatchTST | | TimeEmb | | PatchTST | | TimeEmb | |
| | $\overline{\tau}_w$ | $\overline{W}$ | $\overline{\tau}_w$ | $\overline{W}$ | $\overline{\tau}_w$ | $\overline{W}$ | $\overline{\tau}_w$ | $\overline{W}$ |
| CORAL | 0.1675 | 0.3405 | 0.0591 | 0.2857 | <u>0.3351</u> | 0.3333 | −0.2006 | 0.1452 |
| MMD | −0.0174 | 0.2714 | −0.0649 | 0.2571 | −0.1063 | 0.2190 | −0.1740 | 0.1857 |
| CMD | 0.0694 | 0.2976 | −0.0928 | 0.2500 | 0.3348 | <u>0.3429</u> | −0.1368 | 0.1881 |
| DTW | <u>0.2373</u> | <u>0.3667</u> | 0.0553 | 0.2976 | 0.1577 | 0.2810 | −0.0272 | 0.2143 |
| LogME | −0.2981 | 0.1595 | 0.0039 | 0.3095 | −0.1169 | 0.2000 | 0.1985 | 0.3429 |
| LinMSE | -0.1193 | 0.2667 | 0.2603 | 0.4024 | -0.1708 | 0.2190 | 0.1967 | 0.3452 |
| ETran | −0.1504 | 0.2095 | **0.4014** | <u>0.4405</u> | −0.0620 | 0.2238 | <u>0.3209</u> | <u>0.3714</u> |
| TEMPLATE | −0.1090 | 0.2500 | −0.2290 | 0.1810 | 0.1689 | 0.3333 | −0.1622 | 0.2024 |
| **CWRE** | **0.3879** | **0.4381** | <u>0.3799</u> | **0.4476** | **0.3881** | **0.3929** | **0.4585** | **0.4238** |

the amplified noise, with the trend-aligned projection providing an additional reduction; detailed results and analysis are provided in Appendix C.5.

Overall, the ablation results show that cross-window modeling provides complementary ranking information beyond instance-wise transferability estimation.

### 3.4 RQ3: Generalization Across Architectures and Target Conditions

To answer RQ3, *Does the estimated ranking remain useful across downstream architectures and challenging target conditions*, we evaluate this from two perspectives: cross-architecture source selection and incomplete target observations.

**(a) Cross-architecture source selection.** To assess selection utility beyond the scoring model, CWRE and time-series-specific baseline TEMPLATE select their top-ranked source for each target using PatchTST under complete observations. Figure 3 reports each selection's downstream rank among six sources and relative MAE gap to the best source under iTransformer, DUET, and FilterNet. CWRE selections rank within the top two in seven of nine cases, with a maximum relative MAE gap of $1.868\%$. Including the corresponding TimeEmb results, CWRE achieves lower downstream MAE than TEMPLATE in 9 of 18 comparisons, with 6 ties and 3 losses, showing that the advantage is not uniform across architectures. Additional comparisons with the best-performing baseline ETran, together with results under missing observations, are provided in Appendix C.6.

**(b) Incomplete target observations.** To test robustness under challenging target conditions, we follow the point-missing protocol of (Tang et al., 2025) with $\mathrm{MR} \in 0.3, 0.6$. For each missing rate, source scores and downstream MAE are averaged over three random seeds before computing the ranking metrics. Table 3 reports target-averaged results, with per-target results in Table 7. CWRE achieves the highest $\overline{\tau}_w$ in three of four MR–architecture settings and the highest $\overline{W}$ in all four. The exception occurs with TimeEmb at $\mathrm{MR} = 0.3$, where ETran obtains a higher $\overline{\tau}_w$ ($0.4014$ versus $0.3799$), while CWRE retains a higher $\overline{W}$ ($0.4476$ versus $0.4405$). From $\mathrm{MR} = 0.3$ to $0.6$, $\overline{\tau}_w$ remains nearly unchanged with PatchTST and increases with TimeEmb, whereas $\overline{W}$ decreases for both scoring architectures. Thus, the estimated rankings remain effective under incomplete observations, although higher missingness can reduce joint cross-architecture concordance.

Overall, CWRE maintains useful source-ranking information across downstream architectures and incomplete target observations, while the observed gains remain configuration-dependent.

### 3.5 RQ4: Hyperparameter Analysis and Efficiency

To answer RQ4, *How do the design choices affect robustness and efficiency*, we analyze the sensitivity to key hyperparameters and compare the scoring efficiency of CWRE with representative baselines.

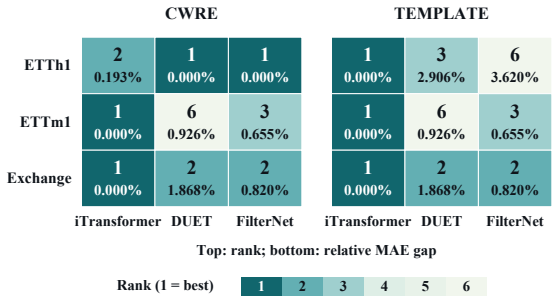



Figure 3: Cross-architecture source selection using PatchTST under complete observations.

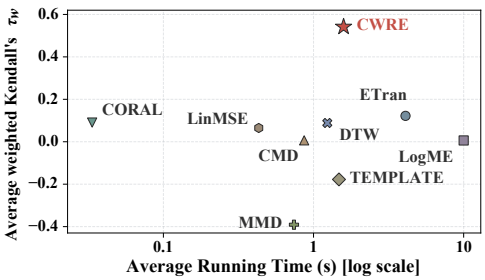



Figure 4: Scoring efficiency on ETTh1: ranking agreement versus scoring time under complete observations.

**(a) Difference step.** On ETTh1, $s = 1$ achieves the best or tied-best $\tau_w$, averaged over $MR \in \{0, 0.3, 0.6\}$, for both scoring architectures, while performance drops at $s = 32$. We therefore use adjacent windows by default. Detailed $\tau_w/W$ results for each MR and sensitivity to $\omega$, $\rho_{\mathrm{tr}}$, and $\rho_{\mathrm{se}}$ are provided in Appendix C.7.

**(b) Scoring efficiency.** Figure 4 compares average scoring time and $\tau_w$ across PatchTST and TimeEmb on ETTh1 with MR$= 0$. CWRE achieves $\tau_w = 0.5420$ in $1.586$ s, close to TEMPLATE's $1.477$ s, while ETran and LogME require $4.099$ s and $9.999$ s with lower ranking performance. These timings exclude feature extraction and hyperparameter search. CWRE therefore provides a favorable trade-off between ranking agreement and scoring cost in this setting.

## 4 Related Work

**Distribution-based source selection.** Distribution alignment methods estimate transferability by measuring the discrepancy between source and target data. Representative metrics include CORAL (Sun et al., 2017), MMD (Gretton et al., 2012), CMD (Zellinger et al., 2017), and DTW (Ye & Dai, 2021). These methods provide source-selection criteria based on source–target similarity, but distribution alignment alone does not fully characterize downstream transfer performance (Ben-David et al., 2010).

**Task transferability estimation.** Task transferability metrics instead evaluate whether source-learned representations are compatible with a target task. LEEP (Nguyen et al., 2020), LogME (You et al., 2021), LinMSE (Nguyen et al., 2023), and ETran (Gholami et al., 2023) estimate transferability using source-model outputs, frozen representations, or task-related fitting criteria. For time series, TEMPLATE (Zhang et al., 2025) further considers temporal dependencies, dominant temporal patterns, and task-adaptive representations. These approaches primarily evaluate individual target samples or properties of the learned representation. CWRE instead introduces cross-window correspondence as an additional criterion, assessing whether representation changes track the corresponding future changes across successive forecasting windows.

## 5 Conclusion

We study transferability estimation for source dataset selection in time series forecasting and propose CWRE. Beyond conventional instance-wise compatibility, CWRE evaluates whether source-pretrained representations preserve the cross-window evolution of target forecasting sequences. By jointly modeling state-level and cross-window relations, CWRE achieves higher overall agreement with downstream transfer-performance rankings across datasets and forecasting architectures. These results suggest that cross-window evolution provides a useful complementary signal for identifying transferable source datasets.

## AI USE STATEMENT

This study utilized generative AI tools to assist in interpreting the experimental results. In addition, these tools were used for literature searches, manuscript editing, and optimizing the design of figures and tables. All AI-assisted content, analyses, and recommendations were reviewed and verified by the authors, who assume full responsibility for the final content of this study.

## REFERENCES

Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Machine learning*, 79(1):151–175, 2010.

Zonglei Chen, Minbo Ma, Tianrui Li, Hongjun Wang, and Chongshou Li. Long sequence time-series forecasting with deep learning: A survey. *Information Fusion*, 97:101819, 2023.

Jiaxiang Dong, Haixu Wu, Haoran Zhang, Li Zhang, Jianmin Wang, and Mingsheng Long. Simmtm: A simple pre-training framework for masked time-series modeling. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 29996–30025. Curran Associates, Inc., 2023. doi: 10.52202/075280-1306. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/5f9bfdfe3685e4ccdbc0e7fb29cccf2a-Paper-Conference.pdf.

Mohsen Gholami, Mohammad Akbari, Xinglu Wang, Behnam Kamranian, and Yong Zhang. Etran: Energy-based transferability estimation. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 18567–18576. IEEE, 2023.

Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *The journal of machine learning research*, 13:723–773, 2012.

Jongseon Kim, Hyungjoon Kim, HyunGi Kim, Dongjun Lee, and Sungroh Yoon. A comprehensive survey of deep learning for time series forecasting: architectural diversity and open challenges. *Artificial Intelligence Review*, 58(7):216, 2025.

Bryan Lim and Stefan Zohren. Time-series forecasting with deep learning: a survey. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 379 (2194):20200209, February 2021. ISSN 1471-2962. doi: 10.1098/rsta.2020.0209. URL http://dx.doi.org/10.1098/rsta.2020.0209.

Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. itransformer: Inverted transformers are effective for time series forecasting. In *International conference on learning representations*, volume 2024, pp. 11116–11140, 2024.

Cuong Nguyen, Tal Hassner, Matthias Seeger, and Cedric Archambeau. Leep: A new measure to evaluate transferability of learned representations. In *International conference on machine learning*, pp. 7294–7305. PMLR, 2020.

Cuong N. Nguyen, Phong Tran, Lam Si Tung Ho, Vu Dinh, Anh T. Tran, Tal Hassner, and Cuong V. Nguyen. Simple transferability estimation for regression tasks. In Robin J. Evans and Ilya Shpitser (eds.), *Proceedings of the Thirty-Ninth Conference on Uncertainty in Artificial Intelligence*, volume 216 of *Proceedings of Machine Learning Research*, pp. 1510–1521. PMLR, 31 Jul–04 Aug 2023. URL https://proceedings.mlr.press/v216/nguyen23a.html.

Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=Jbdc0vTOcol.

Xiangfei Qiu, Xingjian Wu, Yan Lin, Chenjuan Guo, Jilin Hu, and Bin Yang. Duet: Dual clustering enhanced multivariate time series forecasting. In *Proceedings of the 31st ACM SIGKDD conference on knowledge discovery and data mining v. 1*, pp. 1185–1196, 2025.

Baochen Sun, Jiashi Feng, and Kate Saenko. Correlation alignment for unsupervised domain adaptation. In *Domain adaptation in computer vision applications*, pp. 153–171. Springer, 2017.

Kai Tang, Ji Zhang, Hua Meng, Minbo Ma, Qi Xiong, Fengmao Lv, Jie Xu, and Tianrui Li. Coifnet: A unified framework for multivariate time series forecasting with missing values. *arXiv preprint arXiv:2506.13064*, 2025.

Sebastiano Vigna. A weighted correlation index for rankings with ties. In *Proceedings of the 24th International Conference on World Wide Web*, WWW '15, pp. 1166–1176, Republic and Canton of Geneva, CHE, 2015. International World Wide Web Conferences Steering Committee. ISBN 9781450334693. doi: 10.1145/2736277.2741088. URL https://doi.org/10.1145/2736277.2741088.

Zirui Wang, Zihang Dai, Barnabás Póczos, and Jaime Carbonell. Characterizing and avoiding negative transfer. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11285–11294. IEEE, 2019.

Mingyuan Xia, Chunxu Zhang, Zijian Zhang, Hao Miao, Qidong Liu, Yuanshao Zhu, and Bo Yang. Timeemb: A lightweight static-dynamic disentanglement framework for time series forecasting. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=sLfMvrkn6T.

Rui Ye and Qun Dai. Implementing transfer learning across different datasets for time series forecasting. *Pattern Recognition*, 109:107617, 2021.

Kun Yi, Jingru Fei, Qi Zhang, Hui He, Shufeng Hao, Defu Lian, and Wei Fan. Filternet: Harnessing frequency filters for time series forecasting. *Advances in neural information processing systems*, 37:55115–55140, 2024.

Kaichao You, Yong Liu, Jianmin Wang, and Mingsheng Long. Logme: Practical assessment of pre-trained models for transfer learning. In *International conference on machine learning*, pp. 12133–12143. Pmlr, 2021.

Werner Zellinger, Thomas Grubinger, Edwin Lughofer, Thomas Natschläger, and Susanne Saminger-Platz. Central moment discrepancy (cmd) for domain-invariant representation learning. *arXiv preprint arXiv:1702.08811*, 2017.

Guanhua Zhang and Moritz Hardt. Inherent trade-offs between diversity and stability in multi-task benchmarks. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 58984–59002. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/zhang24u.html.

Weiyang Zhang, Xinyang Chen, Xiucheng Li, Kehai Chen, Weili Guan, and Liqiang Nie. Unified transferability metrics for time series foundation models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=atY7t0Krp6.

Zhao Zhang, Hao Yang, and Xianfeng Yang. A transfer learning–based lstm for traffic flow prediction with missing data. *Journal of transportation engineering, Part A: Systems*, 149(10):04023095, 2023.

# A Residual Variation and Its Implications for Adaptation

This section provides a conditional analysis of why cross-window mismatch can complement instance-wise fitting. We first connect cross-window mismatch to the temporal variation of prediction residuals under a shared linear weight matrix, and then show that large residual variation leads to a lower bound on the remaining prediction error when the temporal variation of prediction corrections is bounded.

**Residual decomposition.** Consider a candidate source $k$ and $N \geq 2$ chronologically ordered target windows. Let $\mathcal{Y} \in \mathbb{R}^{N\times V\times H}$ denote their corresponding future sequences, and let $\widehat{\mathcal{Y}}_k^0$ and $\widehat{\mathcal{Y}}_k^a$ denote the predictions before and after adaptation, respectively. We define the baseline residual, remaining prediction error, and prediction correction as

$$\mathcal{R}_k = \mathcal{Y} - \widehat{\mathcal{Y}}_k^0, \tag{11}$$

$$\mathcal{E}_k = \mathcal{Y} - \widehat{\mathcal{Y}}_k^a, \tag{12}$$

$$\mathcal{C}_k = \widehat{\mathcal{Y}}_k^a - \widehat{\mathcal{Y}}_k^0. \tag{13}$$

These quantities satisfy the exact decomposition

$$\mathcal{R}_k = \mathcal{E}_k + \mathcal{C}_k. \tag{14}$$

**Connection to cross-window relations.** Let $\mathbf{L} \in \mathbb{R}^{(N-1)\times N}$ denote the first-order difference operator along the window dimension, such that

$$(\mathbf{L}\mathcal{Z})_{n,:,:} = \mathcal{Z}_{n+1,:,:} - \mathcal{Z}_{n,:,:}. \tag{15}$$

Thus, $\mathbf{L}$ measures changes between consecutive forecasting windows rather than differences along the forecasting horizon.

Let $\mathcal{F}_k \in \mathbb{R}^{N\times V\times D}$ denote the frozen representations from candidate source $k$. For a shared linear weight matrix $W \in \mathbb{R}^{D\times H}$, consider the baseline prediction

$$\widehat{\mathcal{Y}}_k^0 = \mathcal{F}_k W. \tag{16}$$

Because the same $W$ is applied to all windows,

$$\mathbf{L}\mathcal{R}_k = \mathbf{L}\left(\mathcal{Y} - \mathcal{F}_k W\right) = \mathbf{L}\mathcal{Y} - (\mathbf{L}\mathcal{F}_k)W.$$

Therefore, the mismatch between future changes and the changes predicted from frozen representations

$$\mathbf{L}\mathcal{Y} - (\mathbf{L}\mathcal{F}_k)W, \tag{17}$$

is exactly the cross-window variation of the baseline residual. This relation also explains why the state and difference relations share the same linear weight matrix: if $\mathcal{Y} \approx \mathcal{F}_k W$, then the time-independent $W$ gives $\mathbf{L}\mathcal{Y} \approx (\mathbf{L}\mathcal{F}_k)W$.

**Bounded correction variation.** We assume that the prediction correction cannot vary arbitrarily between adjacent windows and satisfies

$$\|\mathbf{L}\mathcal{C}_k\|_F \leq B, \qquad B \geq 0. \tag{18}$$

This assumption places a bound on how much adaptation can change its prediction correction across neighboring windows. Under this condition, large variation in the baseline residual cannot be completely removed if it exceeds the variation that the prediction correction can provide.

**Proposition 1 (Residual-variation lower bound).** Under Equation 18, the remaining prediction error satisfies

$$\boxed{\|\mathcal{E}_k\|_F \geq \frac{\left[\|\mathbf{L}\mathcal{R}_k\|_F - B\right]_+}{\|\mathbf{L}\|_{\mathrm{op}}}}, \tag{19}$$

where $[x]_+ = \max(x, 0)$ and $\|\mathbf{L}\|_{\mathrm{op}}$ denotes the operator norm of $\mathbf{L}$.

**Proof.** Applying $\mathbf{L}$ to Equation 14 gives $\mathbf{L}\mathcal{R}_k = \mathbf{L}\mathcal{E}_k + \mathbf{L}\mathcal{C}_k$. By the triangle inequality,

$$\|\mathbf{L}\mathcal{R}_k\|_F \le \|\mathbf{L}\mathcal{E}_k\|_F + \|\mathbf{L}\mathcal{C}_k\|_F . \tag{20}$$

Using Equation 18,

$$\|\mathbf{L}\mathcal{R}_k\|_F \le \|\mathbf{L}\mathcal{E}_k\|_F + B. \tag{21}$$

By the definition of the operator norm,

$$\|\mathbf{L}\mathcal{E}_k\|_F \le \|\mathbf{L}\|_{\mathrm{op}} \|\mathcal{E}_k\|_F . \tag{22}$$

Combining the above inequalities yields

$$\|\mathcal{E}_k\|_F \ge \frac{\|\mathbf{L}\mathcal{R}_k\|_F - B}{\|\mathbf{L}\|_{\mathrm{op}}}. \tag{23}$$

Since $\|\mathcal{E}_k\|_F \ge 0$, Equation 19 follows.

**Why instance-wise fitting remains necessary.** The lower bound shows that cross-window residual variation can constrain the remaining prediction error, but cross-window variation alone does not capture the magnitude of the residual. For example, suppose that $\mathcal{R}_{k,n,:,:} = A, n = 1, \ldots, N$, where $A$ is a fixed nonzero residual. Then $\mathbf{L}\mathcal{R}_k = 0$, although the baseline prediction still contains a nonzero error. Instance-wise fitting captures this residual magnitude, whereas cross-window fitting captures how the residual changes as the forecasting window advances. The two therefore provide complementary information.

**Connection to CWRE.** Equation 17 shows that, under a shared linear weight matrix, cross-window mismatch measures the temporal variation of the residual left by state-level prediction. Together with Proposition 1, this provides a conditional motivation for evaluating cross-window mismatch in addition to instance-wise fitting. The implemented CWRE further applies branch-specific preprocessing and weighting, so the proposition motivates the use of cross-window relations rather than requiring the implemented fitting residual to be exactly identical to the quantity in the bound.

# B Methodological Details

## B.1 Closed-form Ridge Solutions

Both branches use ridge regression to fit linear readouts on frozen features. We first present the generic solution and then specify its application to the trend and seasonal branches.

**Generic ridge solution.** Given a feature matrix $X \in \mathbb{R}^{m\times d}$ and a target matrix $T \in \mathbb{R}^{m\times H}$, consider

$$W^* = \arg\min_{W\in\mathbb{R}^{d\times H}} \left\{ \|T - XW\|_{\mathrm{F}}^2 + \lambda \|W\|_{\mathrm{F}}^2 \right\}, \qquad \lambda > 0. \tag{24}$$

Setting the gradient to zero gives $(X^\top X + \lambda I_d)W^* = X^\top T$. Since $X^\top X + \lambda I_d$ is positive definite, the solution is unique:

$$W^* = (X^\top X + \lambda I_d)^{-1} X^\top T. \tag{25}$$

In implementation, we solve the corresponding linear system without explicitly forming the inverse.

**Application to the two branches.** Let $\mathrm{mat}(\cdot)$ flatten the window and variable dimensions while preserving the last dimension. All branch features and targets below are standardized as described in Appendix B.3. An overline explicitly denotes a standardized target.

For the trend branch, use

$$X_{\mathrm{tr}} = \begin{bmatrix} \mathrm{mat}(F_{\mathrm{tr}}^{\mathrm{sl}}) \\ \mathrm{mat}(F_{\mathrm{tr}}^{\mathrm{cw}}) \end{bmatrix}, \qquad T_{\mathrm{tr}} = \begin{bmatrix} \mathrm{mat}(\overline{Y_{\mathrm{tr}}^{\mathrm{sl}}}) \\ \mathrm{mat}(\omega\overline{Y_{\mathrm{tr}}^{\mathrm{cw}}}) \end{bmatrix}.$$

These matrices contain $m = 2(N - s)V$ rows. Their two blocks share the same readout. The factor $\omega$ scales only the standardized difference targets, leaving the difference features unchanged.

For the seasonal branch, use $X_{\mathrm{se}} = \mathrm{mat}(F_{\mathrm{se}}), T_{\mathrm{se}} = \mathrm{mat}(\overline{Y_{\mathrm{se}}})$, with $m = NV$ rows.

Substituting the corresponding feature and target matrices into the generic solution yields $W^*_{\mathrm{tr}}$ and $W^*_{\mathrm{se}}$, respectively.

For either branch $b \in \{\mathrm{tr}, \mathrm{se}\}$, the score is

$$S_b = -\frac{\|T_b - X_b W_b^*\|_{\mathrm{F}}^2}{m_b H}, \qquad m_{\mathrm{tr}} = 2(N-s)V, \quad m_{\mathrm{se}} = NV. \tag{26}$$

The normalization is applied only when computing the score. The ridge-fitting objective uses the unnormalized squared-error sum, and the ridge penalty is not included in the reported branch score.

### B.2 Noise Amplification Analysis

Let $\widetilde{\mathbf{R}}_n = \mathbf{R}_n + \boldsymbol{\eta}_n$, where $\boldsymbol{\eta}_n$ is a zero-mean, second-order stationary perturbation with covariance $\boldsymbol{\Sigma}_\eta = \mathrm{Cov}(\boldsymbol{\eta}_n)$. For the first-order difference $D\mathbf{R}_n = \mathbf{R}_{n+1} - \mathbf{R}_n$, we have

$$D\widetilde{\mathbf{R}}_n = D\mathbf{R}_n + (\boldsymbol{\eta}_{n+1} - \boldsymbol{\eta}_n)\,. \tag{27}$$

Before differencing, the expected noise energy is

$$\mathbb{E}\left[\|\boldsymbol{\eta}_n\|_2^2\right] = \mathrm{tr}\,(\boldsymbol{\Sigma}_\eta)\,. \tag{28}$$

After differencing, the expected noise energy becomes

$$\mathbb{E}\left[\|\boldsymbol{\eta}_{n+1} - \boldsymbol{\eta}_n\|_2^2\right] = 2\,\mathrm{tr}\,(\boldsymbol{\Sigma}_\eta) - 2\mathbb{E}\left[\boldsymbol{\eta}_{n+1}^\top \boldsymbol{\eta}_n\right]. \tag{29}$$

If adjacent perturbations are temporally independent, then

$$\mathbb{E}\left[\boldsymbol{\eta}_{n+1}^\top \boldsymbol{\eta}_n\right] = 0, \tag{30}$$

and therefore

$$\mathbb{E}\left[\|\boldsymbol{\eta}_{n+1} - \boldsymbol{\eta}_n\|_2^2\right] = 2\,\mathrm{tr}\,(\boldsymbol{\Sigma}_\eta)\,. \tag{31}$$

More generally, define the lag-one correlation coefficient as

$$\rho_1 = \frac{\mathbb{E}\left[\boldsymbol{\eta}_{n+1}^\top \boldsymbol{\eta}_n\right]}{\mathrm{tr}\,(\boldsymbol{\Sigma}_\eta)}. \tag{32}$$

Then the differenced noise energy is

$$\mathbb{E}\left[\|\boldsymbol{\eta}_{n+1} - \boldsymbol{\eta}_n\|_2^2\right] = 2\,(1-\rho_1)\,\mathrm{tr}\,(\boldsymbol{\Sigma}_\eta)\,. \tag{33}$$

Thus, for temporally independent perturbations, first-order differencing doubles the expected noise energy and increases the corresponding RMS noise magnitude by a factor of $\sqrt{2}$.

### B.3 Feature Construction and Standardization

For a candidate source dataset, let $F \in \mathbb{R}^{N\times V\times D}$ denote the frozen representations extracted from the chronologically ordered target windows by the corresponding source-pretrained model. The trend and seasonal targets are $Y_{\mathrm{tr}}, Y_{\mathrm{se}} \in \mathbb{R}^{N\times V\times H}$, where $N$, $V$, $D$, and $H$ denote the number of windows, the number of variables, the original feature dimension, and the forecasting horizon, respectively.

For an integer step size $1 \le s < N$, let $\mathbf{L}_s \in \mathbb{R}^{(N-s)\times N}$ denote the difference operator along the window dimension. For a tensor $Z \in \mathbb{R}^{N\times V\times p}$, define

$$(\Delta_s Z)_{n,:,:} = (\mathbf{L}_s Z)_{n,:,:} = Z_{n+s,:,:} - Z_{n,:,:}, \qquad n = 1, \ldots, N-s. \tag{34}$$

Consequently, $F^{\mathrm{cw}} = \Delta_s F \in \mathbb{R}^{(N-s)\times V\times D}$, $Y_{\mathrm{tr}}^{\mathrm{cw}} = \Delta_s Y_{\mathrm{tr}} \in \mathbb{R}^{(N-s)\times V\times H}$.

To align the state and difference blocks, we retain the first $N - s$ windows: $F^{\mathrm{sl}} = F_{1:N-s,:,:}, Y_{\mathrm{tr}}^{\mathrm{sl}} = Y_{\mathrm{tr},1:N-s,:,:}$.

At each index $n$, the state block corresponds to window $n$, and the difference block corresponds to the transition from window $n$ to window $n + s$.

The aligned feature blocks are concatenated along the window dimension: $\widetilde{F} = \left[F^{\mathrm{sl}} : F^{\mathrm{cw}}\right] \in \mathbb{R}^{2(N-s)\times V\times D}$, where $[\cdot : \cdot]$ denotes concatenation along the first dimension. The first $N - s$ entries form the state block, and the remaining $N - s$ entries form the difference block. No factor $\omega$ is applied to either feature block.

Before spectral projection, the feature blocks are standardized using the preprocessing rules in the main text. Denote the standardized state, difference, and seasonal inputs by $\overline{F^{\mathrm{sl}}}, \overline{F^{\mathrm{cw}}} \in \mathbb{R}^{(N-s)\times V\times D}$ and $\overline{F} \in \mathbb{R}^{N\times V\times D}$, respectively. The standardized aligned trend input is $\widetilde{\overline{F}} = \left[\overline{F^{\mathrm{sl}}} : \overline{F^{\mathrm{cw}}}\right] \in \mathbb{R}^{2(N-s)\times V\times D}$.

Following the spectral feature construction in the main text, define the projected trend and seasonal representations as

$$U_{\mathrm{tr}} = \mathrm{SpectralSplitting}_{\mathrm{tr}}\left(\widetilde{\overline{F}}\right) \in \mathbb{R}^{2(N-s)\times V\times d_{\mathrm{tr}}}, \tag{35}$$

$$U_{\mathrm{se}} = \mathrm{SpectralSplitting}_{\mathrm{se}}\left(\overline{F}\right) \in \mathbb{R}^{N\times V\times d_{\mathrm{se}}}, \tag{36}$$

where $d_{\mathrm{tr}}$ and $d_{\mathrm{se}}$ denote the resulting feature dimensions of the two branches. If the spectral projections preserve the original feature dimension, then $d_{\mathrm{tr}} = d_{\mathrm{se}} = D$.

The trend representation retains the state–difference ordering: $U_{\mathrm{tr}} = \left[U_{\mathrm{tr}}^{\mathrm{sl}} : U_{\mathrm{tr}}^{\mathrm{cw}}\right]$, $U_{\mathrm{tr}}^{\mathrm{sl}}, U_{\mathrm{tr}}^{\mathrm{cw}} \in \mathbb{R}^{(N-s)\times V\times d_{\mathrm{tr}}}$. Both trend blocks use the same spectral projection, as specified in the main text.

Denote the resulting projected features by $F_{\mathrm{tr}}^{\mathrm{sl}}, F_{\mathrm{tr}}^{\mathrm{cw}} \in \mathbb{R}^{(N-s)\times V\times d_{\mathrm{tr}}}$, $F_{\mathrm{se}} \in \mathbb{R}^{N\times V\times d_{\mathrm{se}}}$.

Similarly, let $\overline{Y_{\mathrm{tr}}^{\mathrm{sl}}}, \overline{Y_{\mathrm{tr}}^{\mathrm{cw}}}, \overline{Y_{\mathrm{se}}}$ denote the standardized trend-state targets, trend-difference targets, and seasonal targets, respectively. The overline indicates the standardized version of the entire enclosed quantity. In particular, $\overline{Y_{\mathrm{tr}}^{\mathrm{cw}}}$ denotes the result of standardizing the difference targets and does not imply that standardization commutes with differencing.

After standardization, $\omega \geq 0$ is applied only to the trend-difference targets. The concatenated trend targets used for ridge fitting are therefore $\widetilde{Y}_{\mathrm{tr}} = \left[\overline{Y_{\mathrm{tr}}^{\mathrm{sl}}} : \omega\overline{Y_{\mathrm{tr}}^{\mathrm{cw}}}\right] \in \mathbb{R}^{2(N-s)\times V\times H}$. The standardized difference features are not multiplied by $\omega$. The resulting inputs and targets have the following dimensions:

| Tensor | Dimension |
|---|---|
| $F^{\mathrm{sl}}$ | $(N-s)\times V\times D$ |
| $F^{\mathrm{cw}}$ | $(N-s)\times V\times D$ |
| $\widetilde{F}$ | $2(N-s)\times V\times D$ |
| $F_{\mathrm{tr}}^{\mathrm{sl}}$ | $(N-s)\times V\times d_{\mathrm{tr}}$ |
| $F_{\mathrm{tr}}^{\mathrm{cw}}$ | $(N-s)\times V\times d_{\mathrm{tr}}$ |
| $\overline{Y_{\mathrm{tr}}^{\mathrm{sl}}}$ | $(N-s)\times V\times H$ |
| $\omega\overline{Y_{\mathrm{tr}}^{\mathrm{cw}}}$ | $(N-s)\times V\times H$ |
| $\widetilde{Y}_{\mathrm{tr}}$ | $2(N-s)\times V\times H$ |
| $F_{\mathrm{se}}$ | $N\times V\times d_{\mathrm{se}}$ |
| $\overline{Y_{\mathrm{se}}}$ | $N\times V\times H$ |

The two branches are constructed from different inputs and should not be interpreted as complementary partitions of the same tensor. Moreover, $\omega$ calibrates the standardized difference targets; it is distinct from the final branch-fusion weights. The setting $s = 0$ used in the ablation study denotes removal of the cross-window relation and is handled separately from the difference construction above.

Table 4: Statistics of the seven datasets used in our experiments. Dim denotes the number of variables.

| Dataset | Dim | Timesteps | Interval | Domain |
|---|---|---|---|---|
| ETTh1 | 7 | 17,420 | 1 hour | Electricity transformer |
| ETTm1 | 7 | 69,680 | 15 min | Electricity transformer |
| ETTm2 | 7 | 69,680 | 15 min | Electricity transformer |
| Electricity | 321 | 26,304 | 1 hour | Electricity consumption |
| Exchange | 8 | 7,588 | 1 day | Exchange rates |
| Traffic | 862 | 17,544 | 1 hour | Transportation |
| Weather | 21 | 52,696 | 10 min | Weather |

# C Supplementary Experiments

## C.1 Experimental Setup

### C.1.1 Datasets

We use seven datasets covering electricity transformers, electricity consumption, exchange rates, transportation, and weather. ETTh1, ETTm1, and ETTm2 contain seven variables each; ETTh1 is sampled hourly, whereas ETTm1 and ETTm2 are sampled every 15 minutes. Electricity and Traffic contain 321 and 862 variables, respectively, with hourly observations. Exchange contains eight variables sampled daily, and Weather contains 21 variables sampled every 10 minutes. Table 4 summarizes their statistics.

We designate ETTh1, ETTm1, and Exchange as target datasets. For each target, the candidate source pool consists of the other six datasets, excluding the target itself. Each estimation method ranks these six candidates according to their estimated transferability to the target task.

We follow the standard chronological train/validation/test splits of each dataset, and compute all transferability scores exclusively on the target training split.

### C.1.2 Architectures and Source-Ranking Protocol

We use two scoring architectures, PatchTST (Nie et al., 2023) and TimeEmb, and three downstream forecasting architectures, iTransformer (Liu et al., 2024), DUET (Qiu et al., 2025), and FilterNet (Yi et al., 2024). This architectural split is fixed across all target datasets.

For each candidate source dataset, we pre-train each architecture on that source. During scoring, the source-pretrained scoring architectures are frozen and applied to the target training split. Both the historical input windows and their corresponding future sequences are contained within the target training interval. Each transferability estimator produces one source ranking per scoring architecture.

For downstream evaluation, each source-pretrained forecasting architecture is fine-tuned on the target training split. Checkpoints are selected using validation MSE, and the reference source ranking for each forecasting architecture is constructed by sorting test MAE in ascending order. Fine-tuning is used exclusively to establish downstream reference rankings and is not part of transferability scoring. The resulting protocol pairs two predicted rankings with three downstream reference rankings for each target setting.

All models use a look-back window of 512 time steps, a forecasting horizon of 96 time steps, and a stride of 1. Unless explicitly varied in the sensitivity analysis, we fix the moving-average window length to $q = 25$, the ridge regularization coefficient to $\lambda = 1$, and the cross-window difference step to $s = 1$. We search over $\omega$, $\rho_{\mathrm{tr}}$, and $\rho_{\mathrm{se}}$ using the corresponding grids examined in Appendix C.7. Only positive values of $\omega$ are included in the main source-scoring procedure; $\omega = 0$ is used solely for sensitivity analysis. For each target setting and scoring architecture, the same search space is applied to every candidate source. Each source $k$ is evaluated independently using its frozen source-pretrained representations $F_k$ and the future sequences $Y$ from the target training split, and its

Table 5: Source-pretraining intervals. Each cell gives the inclusive start and end timestamps. ETTh2 supplies the distance-baseline extractors.

| Dataset | Training | Validation |
|---|---|---|
| ETTh1 | 2016-07-01 00:00 | 2017-06-26 00:00 |
| | 2017-06-25 23:00 | 2017-10-23 23:00 |
| ETTm1, ETTm2 | 2016-07-01 00:00 | 2017-06-26 00:00 |
| | 2017-06-25 23:45 | 2017-10-23 23:45 |
| ETTh2 | 2016-07-01 00:00 | 2017-06-26 00:00 |
| | 2017-06-25 23:00 | 2017-10-23 23:00 |
| Electricity | 2016-07-01 02:00 | 2018-08-07 06:00 |
| | 2018-08-07 05:00 | 2018-11-24 21:00 |
| Traffic | 2016-07-01 02:00 | 2017-11-24 18:00 |
| | 2017-11-24 17:00 | 2018-02-05 21:00 |
| Weather | 2020-01-01 00:10 | 2020-09-13 05:20 |
| | 2020-09-13 05:10 | 2020-10-19 19:30 |
| Exchange | 1990-01-01 | 2004-07-17 |
| | 2004-07-16 | 2006-08-15 |

final transferability score is defined as

$$S_k^* = \max_{\substack{\omega \in \Omega_+ \\ \rho_{\mathrm{tr}} \in \mathcal{R}_{\mathrm{tr}} \\ \rho_{\mathrm{se}} \in \mathcal{R}_{\mathrm{se}}}} \mathcal{T}(F_k, Y; \omega, \rho_{\mathrm{tr}}, \rho_{\mathrm{se}}) \,. \tag{37}$$

where $\Omega_+$ contains the positive cross-window weighting values, and $\mathcal{R}_{\mathrm{tr}}$ and $\mathcal{R}_{\mathrm{se}}$ denote the corresponding feature-selection ratio grids. The maximizing configuration is selected independently for each candidate source. Candidate sources are then ranked in descending order of $S_k^*$. No downstream forecasting performance or ranking-agreement metric is used for configuration selection.

### C.1.3 Baselines

For each distribution alignment method $m \in \{\mathrm{CORAL}, \mathrm{MMD}, \mathrm{CMD}, \mathrm{DTW}\}$, candidate source $k$, and scoring architecture, we compute the source–target discrepancy under two embedding protocols.

In the source-specific protocol, a checkpoint pretrained on source $k$ is used to embed both the source training data and the target training data. In the shared-reference protocol, both datasets are embedded using a checkpoint of the same scoring architecture pretrained on ETTh2. Thus, the reference pretraining dataset is fixed across candidates, while PatchTST and TimeEmb use their respective ETTh2-pretrained checkpoints.

Let $d_{k,m}^{\mathrm{src}}$ and $d_{k,m}^{\mathrm{ett}}$ denote the discrepancies obtained under the source-specific and shared-reference protocols, respectively. The discrepancy used for source $k$ is

$$d_{k,m}^{\star} = \min\left\{d_{k,m}^{\mathrm{src}}, d_{k,m}^{\mathrm{ett}}\right\}. \tag{38}$$

The selection is performed separately for each candidate using the corresponding distribution discrepancy before constructing the source ranking. Therefore, a ranking may contain discrepancies obtained under different embedding protocols. The selection does not use downstream test MAE, $\tau_w$, or $W$.

CWRE, LogME, LinMSE, ETran, and TEMPLATE use the extractor pretrained on the candidate source being evaluated. The ETTh2 reference extractors are pretrained for 30 epochs using the training and validation intervals in Table 5, with checkpoint selection based on validation performance.

### C.1.4 Computing Environment

All experiments are conducted on Ubuntu 24.04.1 LTS using an AMD Ryzen 9 9900X CPU and two NVIDIA GeForce RTX 4090 D GPUs, each with 24 GB of memory.

### C.2 Evaluation Metrics

For a given target dataset, missing rate, and random seed, let $K = 6$ denote the number of candidate source datasets. Let $s_i^{(a)}$ denote the transferability score assigned to candidate source $i$ by the $a$-th score architecture. Let $e_i^{(b)}$ denote the fine-tuning MAE obtained using the $b$-th evaluator architecture, where $a \in \{1, 2\}$ and $b \in \{1, 2, 3\}$. All scores and errors are aligned using the same ordering of candidate source datasets.

**Weighted Kendall's $\tau_w$.** We use weighted Kendall's correlation coefficient to measure the ranking agreement between transferability scores and actual fine-tuning performance. For a given pair of scoring and evaluation architectures, we define

$$x_i = -e_i^{(b)}, \qquad y_i = s_i^{(a)}, \tag{39}$$

so that larger values indicate better transfer performance or higher estimated transferability, respectively.

We use the default settings of `scipy.stats.weightedtau`, which employ additive hyperbolic rank weights. Let $\rho_i$ denote the importance rank of candidate source $i$, starting from 0. Sources ranked closer to the top receive larger weights. The weight assigned to a source pair $(i, j)$ is

$$w_{ij}(\boldsymbol{\rho}) = \frac{1}{\rho_i + 1} + \frac{1}{\rho_j + 1}. \tag{40}$$

Given an importance ranking $\boldsymbol{\rho}$, the weighted correlation coefficient is defined as

$$\tau(\mathbf{x}, \mathbf{y}; \boldsymbol{\rho}) = \frac{\displaystyle\sum_{i<j} w_{ij}(\boldsymbol{\rho}) \operatorname{sgn}(x_i - x_j) \operatorname{sgn}(y_i - y_j)}{\sqrt{\left[\displaystyle\sum_{i<j} w_{ij}(\boldsymbol{\rho}) \mathbb{I}(x_i \neq x_j)\right] \left[\displaystyle\sum_{i<j} w_{ij}(\boldsymbol{\rho}) \mathbb{I}(y_i \neq y_j)\right]}}, \tag{41}$$

where $\operatorname{sgn}(\cdot)$ is the sign function and $\mathbb{I}(\cdot)$ is the indicator function. The numerator accumulates the weighted contributions of concordant and discordant source pairs, while the denominator accounts for ties.

The default implementation constructs two importance rankings, denoted by $\boldsymbol{\rho}^{xy}$ and $\boldsymbol{\rho}^{yx}$, using descending lexicographic orders of $(x_i, y_i)$ and $(y_i, x_i)$, respectively. Descending lexicographic ordering first compares the first value and uses the second value to resolve ties in the first. The final coefficient is the average of the two evaluations:

$$\tau_w^{(a,b)} = \frac{\tau(\mathbf{x}, \mathbf{y}; \boldsymbol{\rho}^{xy}) + \tau(\mathbf{x}, \mathbf{y}; \boldsymbol{\rho}^{yx})}{2}. \tag{42}$$

This implementation therefore accounts for highly ranked candidate sources according to both actual fine-tuning performance and predicted transferability. When the denominators are nonzero, $\tau_w \in [-1, 1]$, with larger values indicating stronger ranking agreement.

For each target setting, we first compute the six correlation coefficients obtained by pairing the two score architectures with the three evaluator architectures, and then take their mean:

$$\overline{\tau}_w = \frac{1}{6} \sum_{a=1}^{2} \sum_{b=1}^{3} \tau_w^{(a,b)}. \tag{43}$$

**Kendall's $W$.** We use Kendall's $W$ to measure the overall agreement between the ranking produced by one score architecture and the actual fine-tuning rankings produced by the three evaluator architectures.

For the $a$-th score architecture, we arrange these four rankings into a rank matrix $\mathbf{R}^{(a)} \in \mathbb{R}^{K \times 4}$. The column indexed by 0 contains the ranking obtained by sorting transferability scores in descending

order. Columns indexed by 1 through 3 contain the rankings obtained by sorting the MAE values of the respective evaluator architectures in ascending order. In all rankings, rank 1 denotes the best candidate source, and tied values are assigned average ranks.

Let $r_{ij}^{(a)}$ denote the rank of candidate source $i$ in ranking $j$. We center the ranks as follows:

$$z_{ij}^{(a)} = r_{ij}^{(a)} - \frac{K+1}{2} = r_{ij}^{(a)} - 3.5. \tag{44}$$

Consistent with our implementation, Kendall's $W$ is computed as

$$W^{(a)} = \frac{\sum_{i=1}^{K}\left(\sum_{j=0}^{3} z_{ij}^{(a)}\right)^2}{4\sum_{i=1}^{K}\sum_{j=0}^{3}\left(z_{ij}^{(a)}\right)^2}. \tag{45}$$

This formulation normalizes the squared rank sums using the actual sum of squared centered ranks. Together with the average-rank treatment of ties, it is equivalent to Kendall's $W$ with tie correction. When the denominator is nonzero, $W \in [0, 1]$, with larger values indicating stronger overall agreement among the four rankings.

Finally, we compute $W$ separately for each of the two score architectures and report their mean:

$$\overline{W} = \frac{1}{2}\sum_{a=1}^{2} W^{(a)}. \tag{46}$$

For $MR \in \{0.3, 0.6\}$, transferability scores are first averaged across three missing-mask seeds for each candidate source, while downstream MAE is averaged across the corresponding seeds and repeated runs; the evaluation metrics are then computed from these averages.

### C.3 Detailed Main Results

This section expands the main results in Section 3.2 by examining performance across target datasets, scoring architectures, and missing rates. Unless otherwise stated, paired values are reported as weighted Kendall's $\tau_w$/Kendall's $W$. Metric definitions and aggregation procedures are provided in Appendix C.2.

#### C.3.1 Effect of Embedding Protocols

Table 6 compares the results of distribution alignment metrics under the source-specific, shared-reference, and combined protocols, using the same setting as Table 1.

### C.4 Missingness Robustness

To test robustness to incomplete target observations, we follow the point-missing protocol of (Tang et al., 2025) with $\mathrm{MR} \in 0.3, 0.6$. Scoring and downstream fine-tuning use the same incomplete historical inputs, while future sequences remain complete. For each missing rate, source scores and downstream MAE are averaged over three random seeds before computing the ranking metrics. We further examine ranking performance under different levels of missing observations, as shown in Table 7. After averaging across the three target datasets and two scoring architectures, CWRE achieves $\tau_w/W$ values of $0.3729/0.3762$, $0.3839/0.4429$, and $0.4233/0.4083$ at missing rates of 0, 0.3, and 0.6, respectively. These are the highest average results among all methods for both metrics at each missing rate. Neither metric falls below its complete-observation average at the two tested nonzero missing rates, although $W$ decreases from 0.4429 at a missing rate of 0.3 to 0.4083 at a missing rate of 0.6.

Table 6: Effect of embedding protocols under complete observations ($\mathrm{MR} = 0$). Source-specific uses candidate-source-pretrained extractors; Shared-reference uses ETTh2-pretrained extractors of the same architecture. Combined selects the smaller discrepancy for each candidate before constructing the source ranking. Within each target and scoring backbone, the best and second-best results across all rows are bold and underlined, respectively, with ties retained.

| Score backbone | Method | Embedding protocol | ETTh1 | | ETTm1 | | Exchange | |
|---|---|---|---|---|---|---|---|---|
| | | | $\tau_w$ | $W$ | $\tau_w$ | $W$ | $\tau_w$ | $W$ |
| PatchTST | CORAL | Source-specific | $0.1088 \pm 0.2438$ | 0.3857 | $0.2036 \pm 0.4940$ | <u>0.2000</u> | $0.1551 \pm 0.4563$ | 0.2643 |
| | | Shared-reference | $0.1088 \pm 0.2438$ | 0.3857 | <u>0.2472</u> $\pm 0.4078$ | **0.2286** | <u>0.2639</u> $\pm 0.6026$ | <u>0.2929</u> |
| | | Combined | $0.1088 \pm 0.2438$ | 0.3857 | $0.2036 \pm 0.4940$ | <u>0.2000</u> | $0.1551 \pm 0.4563$ | 0.2643 |
| | MMD | Source-specific | $-0.4190 \pm 0.3679$ | 0.1286 | $0.2308 \pm 0.5376$ | **0.2286** | $0.1460 \pm 0.5100$ | 0.2786 |
| | | Shared-reference | $-0.3651 \pm 0.2830$ | 0.1571 | $\mathbf{0.2658} \pm 0.5881$ | **0.2286** | $0.1202 \pm 0.4923$ | 0.2286 |
| | | Combined | $-0.3651 \pm 0.2830$ | 0.1571 | $0.2308 \pm 0.5376$ | **0.2286** | $0.1202 \pm 0.4923$ | 0.2286 |
| | CMD | Source-specific | $-0.0458 \pm 0.3208$ | 0.3143 | $-0.0154 \pm 0.4605$ | 0.1000 | $0.0086 \pm 0.3691$ | 0.1786 |
| | | Shared-reference | <u>0.1950</u> $\pm 0.4058$ | <u>0.4429</u> | $0.0880 \pm 0.2065$ | 0.1714 | $-0.0585 \pm 0.3264$ | 0.2000 |
| | | Combined | $-0.0458 \pm 0.3208$ | 0.3143 | $0.1415 \pm 0.5923$ | 0.1571 | $0.0086 \pm 0.3691$ | 0.1786 |
| | DTW | Source-specific | $0.0848 \pm 0.3042$ | 0.3857 | $0.2036 \pm 0.4940$ | <u>0.2000</u> | $-0.1048 \pm 0.3392$ | 0.1357 |
| | | Shared-reference | $0.1483 \pm 0.5115$ | 0.4143 | $0.1401 \pm 0.3197$ | <u>0.2000</u> | $-0.0830 \pm 0.4969$ | 0.1714 |
| | | Combined | $0.0848 \pm 0.3042$ | 0.3857 | $0.2036 \pm 0.4940$ | <u>0.2000</u> | $-0.1048 \pm 0.3392$ | 0.1357 |
| | **CWRE** | Source-specific | $\mathbf{0.4807} \pm 0.3652$ | **0.5071** | $0.2281 \pm 0.4459$ | **0.2286** | $\mathbf{0.3891} \pm 0.0628$ | **0.3786** |
| TimeEmb | CORAL | Source-specific | $0.1025 \pm 0.3791$ | 0.4214 | $0.0431 \pm 0.3811$ | 0.1643 | $-0.1107 \pm 0.3609$ | 0.2286 |
| | | Shared-reference | $-0.5034 \pm 0.1688$ | 0.1071 | $\mathbf{0.2426} \pm 0.4677$ | 0.2071 | $0.2562 \pm 0.0983$ | 0.3071 |
| | | Combined | $0.0721 \pm 0.2199$ | 0.3786 | $0.0830 \pm 0.3533$ | 0.1929 | $-0.1374 \pm 0.2390$ | 0.2214 |
| | MMD | Source-specific | $-0.2984 \pm 0.3899$ | 0.2500 | $0.0390 \pm 0.5429$ | 0.1429 | $0.2073 \pm 0.4353$ | 0.2857 |
| | | Shared-reference | $-0.4159 \pm 0.5425$ | 0.1643 | $0.2104 \pm 0.4141$ | 0.2214 | $0.1805 \pm 0.4487$ | 0.2571 |
| | | Combined | $-0.4168 \pm 0.3783$ | 0.1857 | $0.1243 \pm 0.5355$ | 0.1643 | $\mathbf{0.3388} \pm 0.1409$ | **0.3429** |
| | CMD | Source-specific | <u>0.1660</u> $\pm 0.2398$ | <u>0.4643</u> | $0.1338 \pm 0.6815$ | 0.2000 | $0.0435 \pm 0.3895$ | 0.2714 |
| | | Shared-reference | $-0.3651 \pm 0.2830$ | 0.1571 | $0.1900 \pm 0.2772$ | <u>0.2286</u> | $0.1551 \pm 0.4563$ | 0.2643 |
| | | Combined | $0.0590 \pm 0.0517$ | 0.3786 | $0.1737 \pm 0.5957$ | <u>0.2286</u> | $0.0168 \pm 0.2694$ | 0.2643 |
| | DTW | Source-specific | $0.1025 \pm 0.3791$ | 0.4214 | $0.1492 \pm 0.4085$ | 0.1714 | $-0.1283 \pm 0.2909$ | 0.2143 |
| | | Shared-reference | $-0.3746 \pm 0.1483$ | 0.1357 | $0.1592 \pm 0.4845$ | 0.2071 | $-0.0086 \pm 0.1984$ | 0.1786 |
| | | Combined | $0.0921 \pm 0.3027$ | 0.4214 | $0.1492 \pm 0.4085$ | 0.1714 | $-0.1107 \pm 0.3609$ | 0.2286 |
| | **CWRE** | Source-specific | $\mathbf{0.6032} \pm 0.2612$ | **0.5786** | <u>0.2141</u> $\pm 0.2078$ | **0.2500** | <u>0.3224</u> $\pm 0.4717$ | <u>0.3143</u> |

### C.5 Noise Amplification and Suppression

We provide the complete noise analysis for PatchTST and TimeEmb on ETTh1. For each scoring architecture, frozen representations are independently perturbed with Gaussian noise whose standard deviation is three times the standard deviation of the corresponding feature. We first measure the change in noise MSE after first-order differencing. We then apply the trend projection used in our trend–season decoupling to the corrupted differences and reconstruct them back to the original feature space. Reconstruction error is measured against the corresponding clean differences. Projection fitting and evaluation use disjoint temporal blocks, and all nonzero-noise settings are repeated with 10 independent noise seeds.

Figures 5 and 6 report the complete results. For PatchTST, first-order differencing increases the noise MSE by $1.997\times$. The representative trace shows that the corrupted difference deviates substantially from the clean difference, while the trend-projected reconstruction is closer to the clean reference. Quantitatively, the relative difference reconstruction error decreases from $100.0\%$ without projection to $28.6\%$ with the trend projection, corresponding to a $71.41\%$ reduction. The matched random projection reaches $30.0\%$.

TimeEmb shows the same pattern. Differencing increases the noise MSE by $2.000\times$, and the trend projection reduces the relative difference reconstruction error from $100.0\%$ to $10.8\%$, corresponding to an $89.23\%$ reduction. The matched random projection gives $17.9\%$. For both scoring architectures, all 10 noise seeds yield lower reconstruction errors after applying the trend projection.

Table 7: Transferability estimation under missing observations with missing rates of (a) 0.3 and (b) 0.6. $\tau_w$ reports the mean $\pm$ sample standard deviation across iTransformer, DUET, and FilterNet. $W$ measures joint concordance between the score ranking and the three evaluator rankings. Higher is better for both metrics. Best and second-best results within each missing rate, score architecture, and dataset are bold and underlined, respectively, with ties retained.

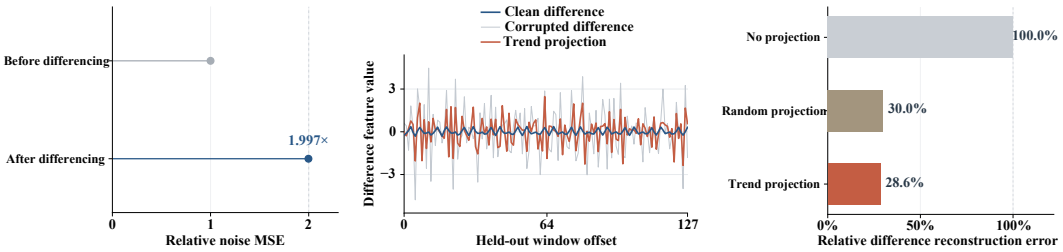
**(a) MR = 0.3**

| Score architecture | Method | ETTh1 | | ETTm1 | | Exchange | |
|---|---|---|---|---|---|---|---|
| | | $\tau_w$ | $W$ | $\tau_w$ | $W$ | $\tau_w$ | $W$ |
| PatchTST | CORAL | $\underline{0.2331} \pm 0.1135$ | $\underline{0.4143}$ | $0.2045 \pm 0.2410$ | 0.4000 | $0.0649 \pm 0.3771$ | 0.2071 |
| | MMD | $-0.2721 \pm 0.2574$ | 0.1643 | $\underline{0.3755} \pm 0.0489$ | **0.5214** | $-0.1556 \pm 0.5319$ | 0.1286 |
| | CMD | $0.2245 \pm 0.1911$ | 0.4000 | $-0.2141 \pm 0.2585$ | 0.2143 | $0.1977 \pm 0.1536$ | 0.2786 |
| | DTW | $0.1891 \pm 0.0439$ | 0.3643 | $0.2812 \pm 0.5177$ | 0.4429 | $\underline{0.2417} \pm 0.1959$ | $\underline{0.2929}$ |
| | LogME | $-0.4177 \pm 0.1723$ | 0.1429 | $-0.3338 \pm 0.3521$ | 0.1786 | $-0.1429 \pm 0.4708$ | 0.1571 |
| | LinMSE | $-0.2068 \pm 0.2299$ | 0.2643 | $-0.3156 \pm 0.2344$ | 0.2500 | $0.1646 \pm 0.8479$ | 0.2857 |
| | ETran | $-0.3655 \pm 0.2559$ | 0.1429 | $-0.2113 \pm 0.4298$ | 0.2857 | $0.1256 \pm 0.1918$ | 0.2000 |
| | TEMPLATE | $0.1964 \pm 0.5272$ | 0.4071 | $-0.3478 \pm 0.5484$ | 0.2143 | $-0.1755 \pm 0.2363$ | 0.1286 |
| | **CWRE** | $\mathbf{0.5306} \pm 0.2076$ | **0.5000** | $\mathbf{0.3778} \pm 0.6051$ | $\underline{0.5000}$ | $\mathbf{0.2553} \pm 0.4521$ | **0.3143** |
| TimeEmb | CORAL | $-0.0549 \pm 0.2012$ | 0.2643 | $\underline{0.3406} \pm 0.1528$ | 0.4714 | $-0.1084 \pm 0.7831$ | 0.1214 |
| | MMD | $-0.3859 \pm 0.3297$ | 0.1929 | $0.1202 \pm 0.4211$ | 0.3643 | $0.0712 \pm 0.4477$ | 0.2143 |
| | CMD | $-0.3002 \pm 0.0791$ | 0.1571 | $0.0789 \pm 0.2101$ | 0.4214 | $-0.0571 \pm 0.6697$ | 0.1714 |
| | DTW | $0.1569 \pm 0.5743$ | 0.3286 | $0.2848 \pm 0.6177$ | $\underline{0.4857}$ | $-0.2757 \pm 0.3238$ | 0.0786 |
| | LogME | $0.2236 \pm 0.4616$ | 0.4286 | $-0.0467 \pm 0.3749$ | 0.3214 | $-0.1651 \pm 0.2647$ | 0.1786 |
| | LinMSE | $\underline{0.5111} \pm 0.1649$ | **0.5429** | $0.0667 \pm 0.3840$ | 0.3786 | $0.2032 \pm 0.4769$ | $\underline{0.2857}$ |
| | ETran | $\mathbf{0.5542} \pm 0.3937$ | $\underline{0.5214}$ | $\mathbf{0.4195} \pm 0.3973$ | **0.5214** | $\underline{0.2304} \pm 0.6006$ | 0.2786 |
| | TEMPLATE | $-0.4200 \pm 0.2639$ | 0.1143 | $-0.0041 \pm 0.4707$ | 0.3000 | $-0.2630 \pm 0.4031$ | 0.1286 |
| | **CWRE** | $0.4308 \pm 0.3983$ | 0.5071 | $\mathbf{0.4195} \pm 0.3973$ | **0.5214** | $\mathbf{0.2893} \pm 0.4085$ | **0.3143** |

**(b) MR = 0.6**

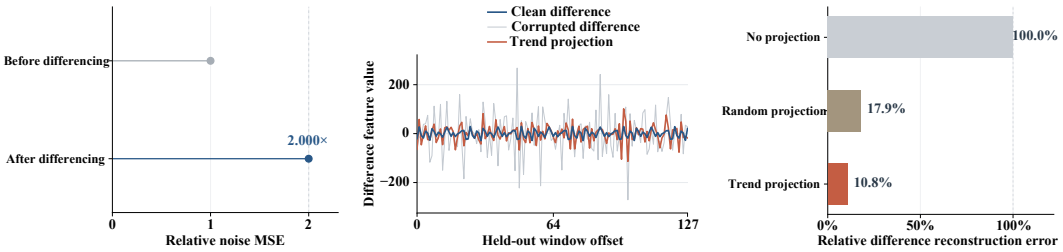
| Score architecture | Method | ETTh1 | | ETTm1 | | Exchange | |
|---|---|---|---|---|---|---|---|
| | | $\tau_w$ | $W$ | $\tau_w$ | $W$ | $\tau_w$ | $W$ |
| PatchTST | CORAL | $\underline{0.4667} \pm 0.3971$ | 0.4857 | $\mathbf{0.3995} \pm 0.3216$ | **0.3500** | $0.1392 \pm 0.4357$ | 0.1643 |
| | MMD | $-0.4694 \pm 0.1049$ | 0.1429 | $0.1111 \pm 0.6278$ | $\underline{0.3071}$ | $0.0395 \pm 0.2634$ | 0.2071 |
| | CMD | $\mathbf{0.4930} \pm 0.1999$ | **0.5143** | $0.3723 \pm 0.1749$ | **0.3500** | $0.1392 \pm 0.4357$ | 0.1643 |
| | DTW | $0.4181 \pm 0.3533$ | $\underline{0.5000}$ | $0.0499 \pm 0.6058$ | 0.2000 | $0.0050 \pm 0.1235$ | 0.1429 |
| | LogME | $-0.3179 \pm 0.7832$ | 0.1571 | $-0.1279 \pm 0.4665$ | 0.1571 | $0.0952 \pm 0.2508$ | 0.2857 |
| | LinMSE | $-0.1506 \pm 0.4409$ | 0.3071 | $-0.1088 \pm 0.2125$ | 0.2071 | $-0.2531 \pm 0.4430$ | 0.1429 |
| | ETran | $-0.3610 \pm 0.6857$ | 0.1286 | $-0.0200 \pm 0.4417$ | 0.2143 | $\underline{0.1950} \pm 0.2348$ | $\underline{0.3286}$ |
| | TEMPLATE | $0.2680 \pm 0.6831$ | 0.4286 | $0.2118 \pm 0.6038$ | 0.3000 | $0.0268 \pm 0.3864$ | 0.2714 |
| | **CWRE** | $0.4649 \pm 0.5134$ | 0.4929 | $\underline{0.3941} \pm 0.7321$ | **0.3500** | $\mathbf{0.3052} \pm 0.5066$ | **0.3357** |
| TimeEmb | CORAL | $-0.3020 \pm 0.2313$ | 0.1500 | $-0.2921 \pm 0.1052$ | 0.1000 | $-0.0077 \pm 0.5701$ | 0.1857 |
| | MMD | $-0.3138 \pm 0.4473$ | 0.2357 | $-0.0844 \pm 0.1750$ | 0.1857 | $-0.1238 \pm 0.4796$ | 0.1357 |
| | CMD | $-0.2580 \pm 0.4719$ | 0.1786 | $-0.2068 \pm 0.2063$ | 0.1571 | $0.0544 \pm 0.4742$ | 0.2286 |
| | DTW | $0.1846 \pm 0.2654$ | 0.3571 | $-0.1628 \pm 0.3021$ | 0.1357 | $-0.1034 \pm 0.0367$ | 0.1500 |
| | LogME | $0.2943 \pm 0.6111$ | 0.4429 | $\underline{0.2671} \pm 0.4157$ | $\underline{0.3143}$ | $0.0340 \pm 0.2015$ | 0.2714 |
| | LinMSE | $0.4757 \pm 0.3637$ | 0.5500 | $-0.1873 \pm 0.3795$ | 0.1643 | $0.3016 \pm 0.3007$ | 0.3214 |
| | ETran | $\underline{0.5243} \pm 0.2957$ | $\underline{0.5500}$ | $0.0644 \pm 0.2829$ | 0.2500 | $\underline{0.3741} \pm 0.5872$ | $\underline{0.3143}$ |
| | TEMPLATE | $-0.1986 \pm 0.3966$ | 0.2000 | $0.0866 \pm 0.4630$ | 0.2500 | $-0.3746 \pm 0.2007$ | 0.1571 |
| | **CWRE** | $\mathbf{0.5506} \pm 0.3358$ | **0.5643** | $\mathbf{0.4472} \pm 0.1530$ | **0.3786** | $\mathbf{0.3778} \pm 0.8136$ | **0.3286** |

These results show that first-order differencing amplifies independent perturbations, while the trend projection used in our decoupling substantially reduces the resulting noise contamination in the cross-window difference branch.




(a) Noise amplification after differencing (b) Representative difference reconstruction (c) Relative difference reconstruction error

Figure 5: Noise amplification and suppression on ETTh1 with PatchTST. (a) Relative noise MSE before and after first-order differencing. (b) Clean difference, corrupted difference, and reconstruction after trend projection for a representative held-out segment. (c) Relative difference reconstruction error over 10 independent noise seeds, normalized by the no-projection result.




(a) Noise amplification after differencing (b) Representative difference reconstruction (c) Relative difference reconstruction error

Figure 6: Noise amplification and suppression on ETTh1 with TimeEmb. (a) Relative noise MSE before and after first-order differencing. (b) Clean difference, corrupted difference, and reconstruction after trend projection for a representative held-out segment. (c) Relative difference reconstruction error over 10 independent noise seeds, normalized by the no-projection result.

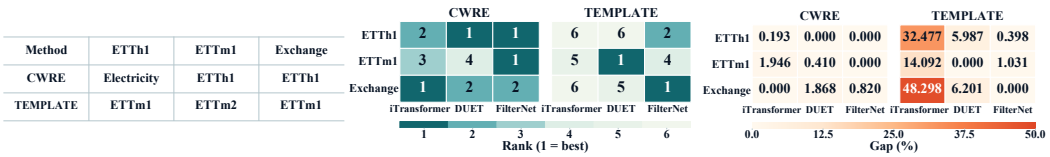
### C.6 Cross-Architecture Analysis

**Comparison with TEMPLATE.** This section provides additional cross-architecture results using TimeEmb as the scoring architecture and under different missing rates. The evaluation protocol follows the setting described in the main text.

Under complete observations, as shown in Figure 7, the TimeEmb results show a similar pattern to that observed with PatchTST in the main text. The sources selected by CWRE have relative MAE gaps below $2\%$ across all nine target–downstream combinations, whereas the sources selected by TEMPLATE show substantially larger gaps in several settings, including $32.477\%$ on ETTh1 and $48.298\%$ on Exchange with iTransformer. These results provide additional evidence that the source selected by CWRE remains competitive after changing the downstream architecture.

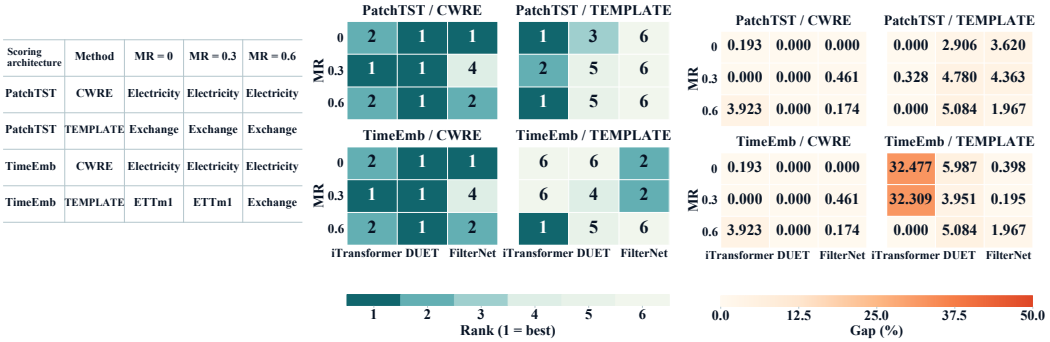
Figure 8 further examines ETTh1 under $\mathrm{MR} \in \{0, 0.3, 0.6\}$. For both scoring architectures, CWRE selects Electricity. Its downstream ranks under [iTransformer, DUET, FilterNet] are $[2, 1, 1]$, $[1, 1, 4]$, and $[2, 1, 2]$ as the missing rate increases. At $\mathrm{MR} = 0.3$, although the selected source ranks fourth under FilterNet, its MAE is only $0.461\%$ higher than that of the best source. Across the 18 paired comparisons over the three missing rates, CWRE selects a source with lower MAE than TEMPLATE in 14 cases, compared with 4 cases for TEMPLATE. Thus, the selected source generally remains competitive across downstream architectures as missingness increases.

**Comparison with ETran.** We further compare CWRE with ETran under complete observations using PatchTST as the scoring architecture; TimeEmb results are omitted because both methods select the same source for all three targets and yield identical downstream outcomes. As shown in Figure 9, the two methods select the same source on ETTm1 and Exchange, but differ on ETTh1, where CWRE selects Electricity and ETran selects Exchange. Their downstream ranks under [iTransformer, DUET,




(a) Selected sources (b) Downstream rank (c) Relative MAE gap (%)

Figure 7: Cross-architecture evaluation of source selection between CWRE and TEMPLATE using TimeEmb as the scoring architecture under complete observations.




(a) Selected sources (b) Downstream rank (c) Relative MAE gap (%)

Figure 8: Cross-architecture evaluation on ETTh1 between CWRE and TEMPLATE under MR$=0$, 0.3, and 0.6. The figure shows the selected source, its downstream rank among the six candidate sources, and its relative MAE gap to the best-performing source for each downstream architecture.

FilterNet] are $[2, 1, 1]$ and $[1, 3, 6]$, with relative MAE gaps of $[0.193, 0, 0]\%$ and $[0, 2.906, 3.620]\%$, respectively. Thus, ETran performs best under iTransformer, while the source selected by CWRE performs best under DUET and FilterNet.

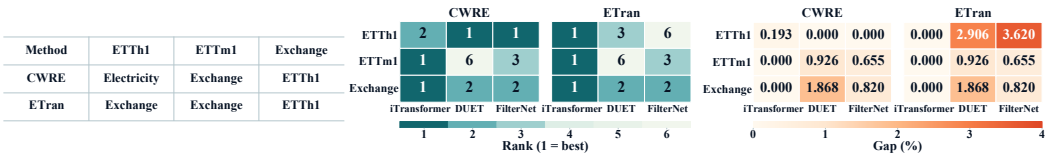
Figure 10 further compares CWRE and ETran on ETTh1 under $\mathrm{MR} \in \{0, 0.3, 0.6\}$ using PatchTST. Across all three missing rates, CWRE consistently selects Electricity, whereas ETran consistently selects Exchange. The downstream ranks and relative MAE gaps in Figure 10 show how this source-selection difference propagates to heterogeneous downstream architectures as missingness increases.

Together, these additional results support the cross-architecture analysis in the main text. The comparison with TEMPLATE shows that the sources selected by CWRE generally remain competitive under both scoring architectures and under incomplete target observations, while the comparison with ETran indicates that the main difference is concentrated in the PatchTST-based source ranking.

### C.7 Hyperparameter Sensitivity

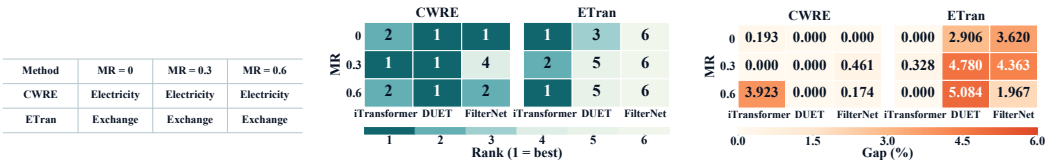
This section provides the full hyperparameter sensitivity results on ETTh1. We evaluate the difference step, dynamic weight, and trend–seasonal feature ratios using PatchTST and TimeEmb as scoring architectures under $\mathrm{MR} \in \{0, 0.3, 0.6\}$. For each setting, $\tau_w$ is the average weighted Kendall correlation between the source ranking produced by the scoring architecture and the rankings from the three downstream evaluators, while $W$ measures their joint concordance.

**Difference step.** The difference step $s$ determines the temporal scale of the cross-window relation. Figure 11 reports the results for $s \in \{0, 1, 3, 32, 95\}$. Short-range differences generally provide the most reliable ranking signal. In particular, $s = 1$ achieves strong or near-best results across the six architecture–missing-rate settings, while $s = 1$ and $s = 3$ perform similarly for PatchTST under $\mathrm{MR} = 0$ and 0.3. By contrast, increasing the step to 32 causes clear degradation in several settings, especially for TimeEmb.




(a) Selected sources (b) Downstream rank (c) Relative MAE gap (%)

Figure 9: Cross-architecture comparison between CWRE and ETran using PatchTST as the scoring architecture under complete observations. The figure shows the selected source, its downstream rank among the six candidate sources, and its relative MAE gap to the best-performing source for each downstream architecture.




(a) Selected sources (b) Downstream rank (c) Relative MAE gap (%)

Figure 10: Cross-architecture comparison between CWRE and ETran on ETTh1 under $\mathrm{MR} \in \{0, 0.3, 0.6\}$ using PatchTST as the scoring architecture. The figure shows the selected source, its downstream rank among the six candidate sources, and its relative MAE gap to the best-performing source for each downstream architecture.

The dependence on $s$ is not monotonic. For example, performance partially recovers at $s = 95$ after declining at $s = 32$ in several configurations. This suggests that increasing the temporal separation does not simply introduce progressively less useful differences: adjacent windows primarily describe local temporal evolution, whereas very large steps may capture coarser changes between more separated windows. The latter interpretation is only suggestive, as the current experiment does not explicitly separate changes at different temporal scales. PatchTST under $\mathrm{MR} = 0.6$ is also comparatively insensitive to the choice of $s$, indicating that the additional ranking information provided by the cross-window relation becomes smaller in this setting. Overall, $s = 1$ provides the most stable choice across architectures and missing rates.

**Dynamic weight.** The coefficient $\omega$ scales the cross-window target in the shared ridge regression, thereby controlling the magnitude of future changes that the shared mapping is required to explain. Figure 12 shows that its effect differs between the two scoring architectures. When $\omega = 0$, the cross-window targets no longer provide supervision for future changes. If the cross-window feature block were retained while its targets were set to zero, the objective would still contain the additional term $\|F^{\mathrm{cw}}W\|_F^2$, implicitly encouraging invariant predictions across adjacent windows. To make the zero setting a genuine instance-wise-only baseline, we therefore remove the entire cross-window fitting block when $\omega = 0$.

For PatchTST, introducing a small positive target scale produces a large improvement under $\mathrm{MR} = 0$ and $0.3$, indicating that requiring the shared mapping to explain cross-window future changes provides useful information beyond instance-wise evaluation alone. Performance remains relatively stable over a moderate range of positive values, but decreases when $\omega$ becomes large. This suggests that the cross-window target should be calibrated to complement the instance-wise target: an excessively large scale can disturb the balance between fitting individual futures and explaining their changes through the shared mapping.

Under $\mathrm{MR} = 0.6$, PatchTST shows a different pattern. The instance-wise-only baseline at $\omega = 0$ is already competitive, and introducing a small positive target scale yields only limited improvement, whereas larger values lead to a clear decrease in $\tau_w$. Together with the difference-step results, this suggests that the additional ranking information provided by cross-window changes is reduced for PatchTST under severe missingness. One possible explanation is that representation differences become less reliable when the observations are heavily missing, so enforcing a larger correspondence with future changes provides less useful guidance for source ranking.

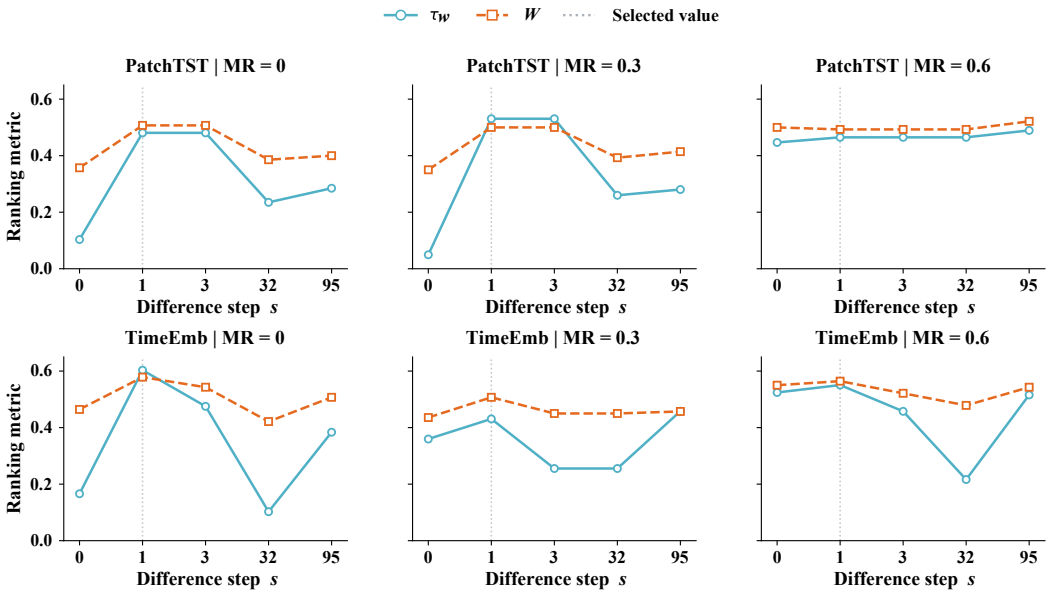



Figure 11: Sensitivity to the difference step $s$ on ETTh1. Results are reported for PatchTST and TimeEmb under $\mathrm{MR} \in \{0, 0.3, 0.6\}$ using $\tau_w$ and $W$.

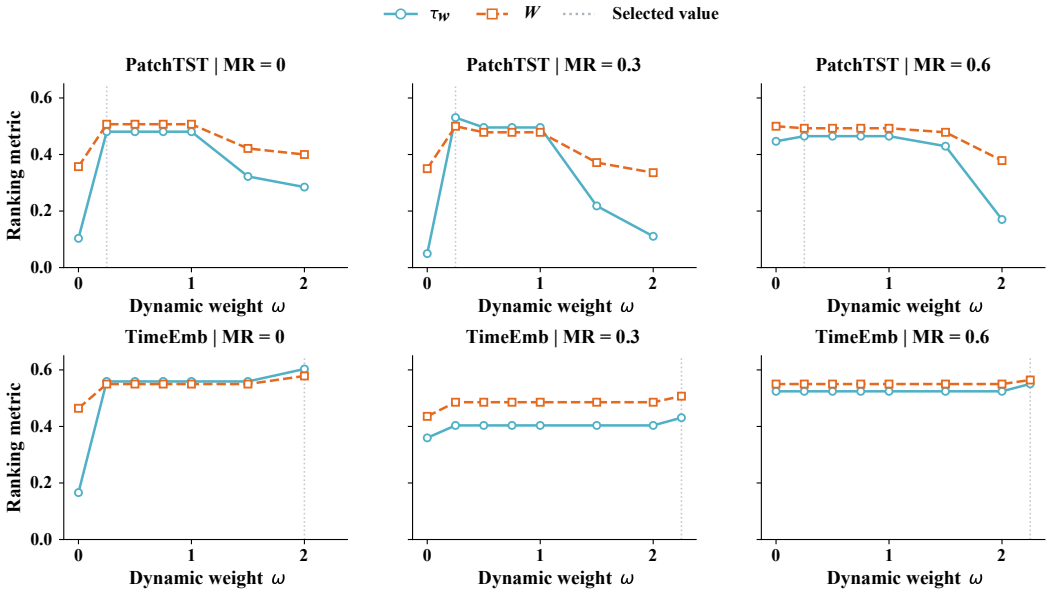



Figure 12: Sensitivity to the dynamic weight $\omega$ on ETTh1. Results are reported for PatchTST and TimeEmb under $\mathrm{MR} \in \{0, 0.3, 0.6\}$ using $\tau_w$ and $W$.

**Trend and seasonal feature ratios.** We further examine the amount of retained information from the trend and seasonal branches. For PatchTST, the trend ratio is selected from $\{0.1, 0.2, 0.3\}$ and the seasonal ratio from $\{0.2, 0.3, 0.4\}$. For TimeEmb, the tested trend ratios are $\{0.01, 0.05, 0.1, 0.3\}$ where applicable, and the seasonal ratio is selected from $\{0.3, 0.4, 0.5\}$. Figures 13 and 14 report $\tau_w$ and $W$, respectively.

Across most settings, ranking performance is more sensitive to the trend ratio than to the seasonal ratio. This is consistent with the different roles of the two branches in CWRE: trend features contribute to both state-level relations and cross-window relations, whereas seasonal features are used only in

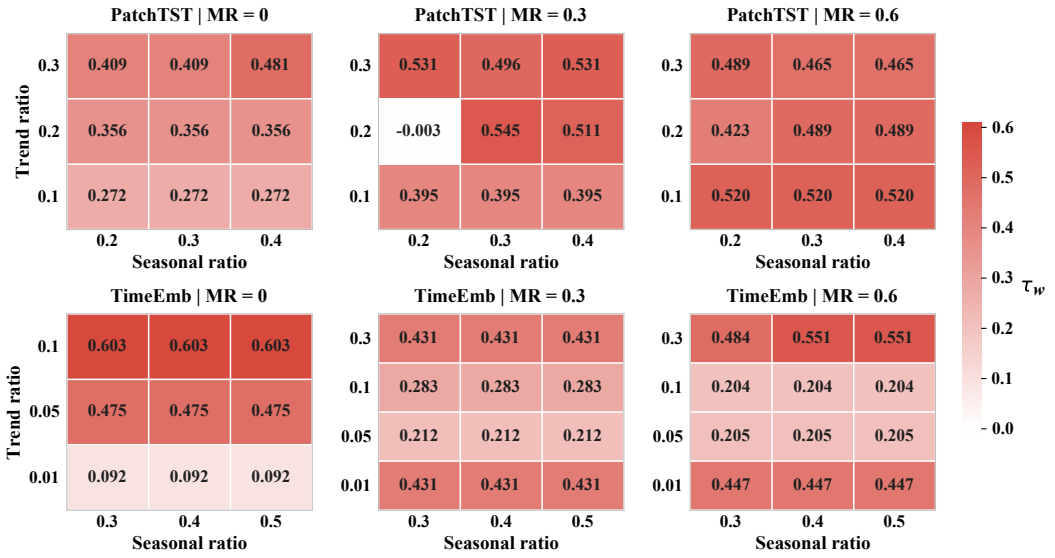



Figure 13: Sensitivity of $\tau_w$ to the trend and seasonal feature ratios on ETTh1. The top and bottom rows correspond to PatchTST and TimeEmb, respectively, under $\mathrm{MR} \in \{0, 0.3, 0.6\}$. Each cell reports $\tau_w$.

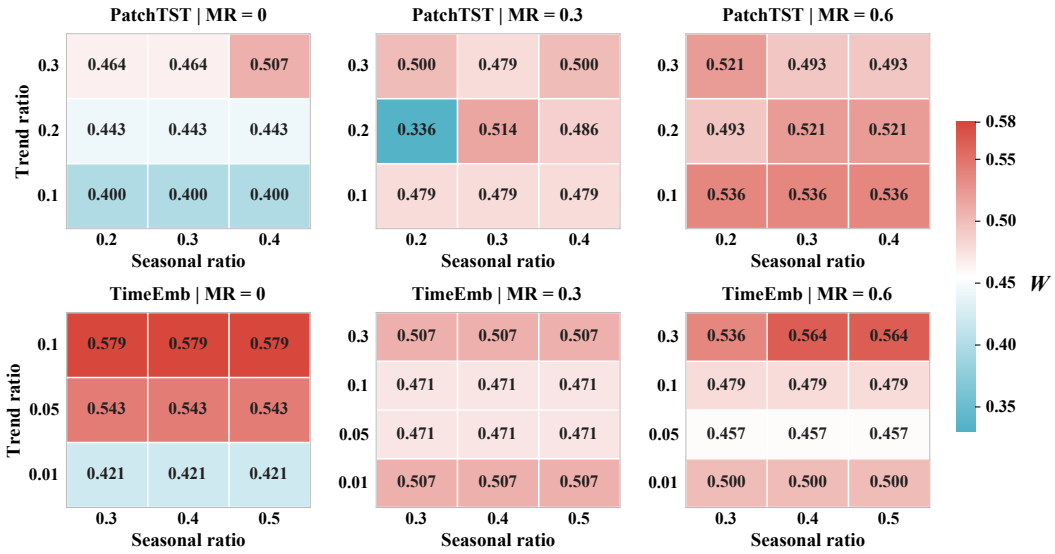



Figure 14: Sensitivity of Kendall's $W$ to the trend and seasonal feature ratios on ETTh1. The top and bottom rows correspond to PatchTST and TimeEmb, respectively, under $\mathrm{MR} \in \{0, 0.3, 0.6\}$. Each cell reports $W$.

state-level relations. For example, under $\mathrm{MR} = 0$, increasing the PatchTST trend ratio from $0.1$ to $0.3$ increases $\tau_w$ from $0.272$ to as high as $0.481$ and $W$ from $0.400$ to $0.507$. TimeEmb shows an even clearer pattern, with $\tau_w$ increasing from $0.092$ at a trend ratio of $0.01$ to $0.603$ at $0.1$, while changing the seasonal ratio produces no change within each trend-ratio row.

The preferred trend ratio nevertheless changes with both the scoring architecture and the missing rate. For PatchTST under $\mathrm{MR} = 0$, retaining a larger trend component is beneficial, whereas under $\mathrm{MR} = 0.6$, the smallest tested trend ratio of $0.1$ gives the highest values, with $\tau_w = 0.520$ and $W = 0.536$. Therefore, retaining more trend features is not universally beneficial under missing

observations. A smaller retained trend subspace may filter representation components that become less reliable under severe missingness, although the present sensitivity experiment does not directly verify this mechanism.

An isolated irregularity appears for PatchTST under $\mathrm{MR} = 0.3$ at the combination $(0.2, 0.2)$, where $\tau_w$ drops to $-0.003$ and $W$ to $0.336$. Neighboring configurations recover immediately; for example, changing only the seasonal ratio from $0.2$ to $0.3$ raises the two metrics to $0.545$ and $0.514$, respectively. Since neither a trend ratio of $0.2$ nor a seasonal ratio of $0.2$ consistently causes degradation in the surrounding settings, this point is better viewed as a configuration-specific interaction rather than evidence of a monotonic effect from either ratio alone.

TimeEmb also exhibits a non-monotonic dependence on the trend ratio under missing observations. At $\mathrm{MR} = 0.3$, the lowest and highest tested trend ratios obtain higher ranking agreement than the intermediate ratios. At $\mathrm{MR} = 0.6$, however, the larger trend ratio of $0.3$ gives the strongest results, reaching $\tau_w = 0.551$ and $W = 0.564$ for seasonal ratios of $0.4$ or $0.5$. This behavior differs from PatchTST at the same missing rate, confirming that the preferred trend allocation is architecture dependent. In contrast, the seasonal ratio has a much smaller effect in most tested settings. Overall, the sensitivity of the trend–seasonal allocation is governed mainly by the retained trend component, while its preferred value varies with both the scoring architecture and the missing condition.