GNA: GRANULAR NEIGHBOR ASSEMBLY FOR RETRIEVAL-AUGMENTED MULTIVARIATE TIME-SERIES FORECASTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Deep forecasters predict from a fixed-length lookback window, and lengthening it gives diminishing returns at a growing cost. Retrieval augmentation instead shows the model how similar past situations continued. Existing retrieval forecasters treat a retrieved past window as one unit, so every variate receives the continuation of the same past moment. In multivariate series, however, the best past match differs from variate to variate. We present GNA (Granular Neighbor Assembly), a retrieval layer for forecasting backbones that assembles neighbors at two granularities: whole past windows, which keep the variates coherent, and per-variate neighbors, in which each variate takes its future from its own best-matching past. A learned gate decides, per forecast step and variate, how much to trust these futures against a persistence forecast, next to the backbone's own forecast. Candidates come from an embedding trained to predict each window's future, and retrieval is strictly causal: a past window is used only once its future has been observed. With the same lookback for every model and one retrieval configuration for all datasets, GNA improves two Transformer backbones in 85 of 96 dataset–horizon settings, gives the lowest MSE on 8 of 12 standard benchmarks and beats its backbone in every seed on 10 of them. Both granularities are needed, and neighbors of mismatched queries are worse than none. Retrieval helps most where the lookback says least: the gate shifts trust to retrieved futures further ahead, and on monitoring data of a particle-physics experiment the gain concentrates at the start of data-taking runs. Where it fails, on hourly non-stationary series at long horizons, we trace the loss to the drifting level of the retrieved futures.

1 INTRODUCTION

Multivariate time-series forecasting predicts future values of a set of variates from their recent history. It supports decisions wherever acting early pays off: weather forecasts are valued at $31.5B per year for US households against $5.1B in costs (Lazo et al., 2009), and forecast errors are a standard signal for anomaly detection, for example in spacecraft telemetry (Hundman et al., 2018). Forecasting is hard because observations are noisy, variates interact, and the underlying system drifts, switches between operating regimes or enters situations that its recent past does not describe.

Deep forecasters (Nie et al., 2023; Liu et al., 2024b; Zeng et al., 2023) predict from a lookback window of fixed length L. Anything that happened before the window is invisible to them. Lengthening the window has diminishing returns: for PatchTST, increasing the lookback from 336 to 720 steps changes the Traffic MSE from 0.367 to 0.365 and makes ETTh1 slightly worse (Nie et al., 2023, appendix), while the cost of attention grows with L. No practical window reaches a situation last seen weeks earlier. In some applications a short lookback is also a requirement rather than a choice: after a restart or a change of operating conditions, or at the start of each data-taking run, only a short stretch of relevant past exists, and online monitoring limits latency and memory. Retrieval is a non-parametric alternative: find past windows that resemble the present one and show the model how they continued. This idea goes back to analog forecasting in meteorology (Lorenz, 1969) and underlies retrieval-augmented language models (Lewis et al., 2020).

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.

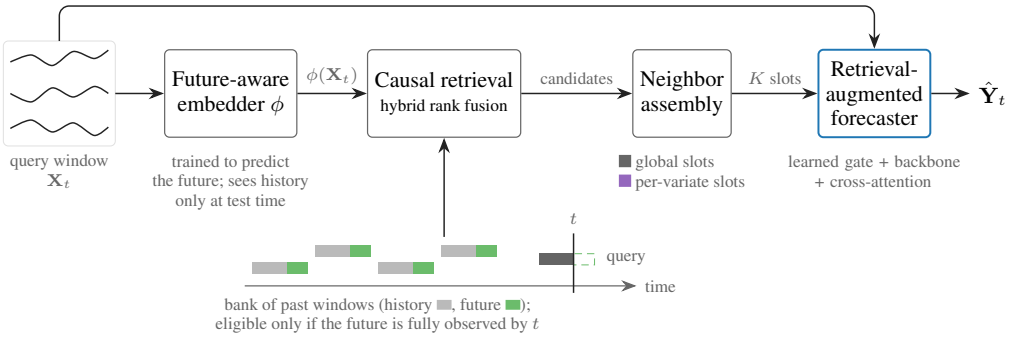


Figure 1: GNA at forecast origin t. The future-aware embedder ϕ (Sec. 3.2) embeds the query window $\mathbf{X}_t$ (lines: variates). Causal retrieval (Sec. 3.2) ranks the bank of past windows (bottom; grey: history, green: future); a window is eligible only if its future is fully observed by t (Eq. 1), and the query's own future (dashed) is unknown. Neighbor assembly (Sec. 3.3) fills K slots with whole past windows (global, grey) or with a separate choice per variate (purple). The forecaster (Sec. 3.4) combines the query window (top arrow) and the slots through a learned gate, the backbone and cross-attention into $\hat{\mathbf{Y}}_t$.

Retrieval-augmented forecasters retrieve whole past windows whose histories resemble the query and pass their futures to the model (Han et al., 2025; Liu et al., 2024a; Ning et al., 2025; Tire et al., 2026); GTR (Cao et al., 2026a) instead learns a global periodic memory. A retrieved multivariate window is used as one unit: all variates receive the continuation of the same past moment. In a network of hundreds of sensors, however, the past moment that best matches one sensor rarely matches all others, and the variates that fit the neighbor poorly receive a misleading future. Retrieved futures can also be useful for some forecast steps and misleading for others, so the model must learn when to use them.

We propose GNA (Granular Neighbor Assembly, Fig. 1), built on two ideas. First, neighbors are assembled at two granularities: *global* slots keep one past situation coherent across variates, and *per-variate* slots let each variate take its own best-matching past, so a slot can combine pieces of different past windows. Second, a learned gate weighs the slot futures against a persistence forecast per forecast step and variate, next to the backbone's own forecast, so the model can rely on the lookback for the next steps and on retrieved futures further out. Candidates come from a retrieval embedding that uses the futures of training windows as privileged information (Vapnik & Vashist, 2009): it is trained to predict each window's future, while at test time it sees only the past. The bank is strictly causal and grows during deployment without retraining. GNA needs only a backbone that produces a forecast and per-variate tokens; we use GridTST (Cheng et al., 2024) and iTransformer (Liu et al., 2024b).

Our contributions are:

- **Granular neighbors with learned trust.** A retrieval layer that assembles neighbors at two granularities, whole past windows and single variates, and weighs them per forecast step and variate against persistence and the backbone (Sec. 3). To our knowledge, no previous retrieval-augmented forecaster combines whole-window and per-variate neighbors in one multivariate model (App. H).
- **Evidence under a strict protocol.** Every model, including all baselines, uses the same lookback $L = 96$; retrieval is strictly causal; one retrieval configuration serves all datasets. GNA improves GridTST in 43 and iTransformer in 42 of 48 dataset–horizon settings, gives the lowest MSE on 8 of 12 benchmarks, and beats its backbone in all three seeds on 10 of 12. Both granularities are needed, and replacing the retrieved windows by those of mismatched queries is worse than no retrieval, so the gain comes from relevant neighbors, not from added capacity (Secs. 5–6).
- **When retrieval helps.** Retrieval helps most where the lookback says least: the gate moves trust from persistence to retrieved futures along the horizon, and on data-acquisition monitoring data the gain concentrates at the start of runs. Where GNA loses (hourly ETT data), the loss is a far-horizon effect consistent with level drift of the retrieved futures (Sec. 7).

2 Related work

Deep forecasters and the context window. Transformer forecasters differ in how they tokenize the lookback window: by time step (Zhou et al., 2021; Wu et al., 2021), by patch (Nie et al., 2023), by variate (Liu et al., 2024b), across both dimensions (Zhang & Yan, 2023; Cheng et al., 2024), or with exogenous variables (Wang et al., 2024b). Linear and MLP models (Zeng et al., 2023; Wang et al., 2024a; Han et al., 2024; Lin et al., 2024; 2025), convolutional (Wu et al., 2023; Lai et al., 2018) and state-space models (Wang et al., 2025b) are strong alternatives, and instance normalization (Kim et al., 2022) is standard. All of them condition only on the lookback window; GNA adds information from outside it and uses two of them as backbones.

Retrieval-augmented forecasting. Nearest-neighbor and analog methods predict with the continuation of similar past states (Lorenz, 1969; Yakowitz, 1987). Recent deep variants retrieve (history, future) windows by the Pearson correlation of raw histories and aggregate the top matches (RAFT; Han et al., 2025), use pre-trained encoder embeddings to condition a diffusion model (RATD; Liu et al., 2024a), or augment time-series foundation models with retrieved univariate segments (TS-RAG, RAF, TimeRAF; Ning et al., 2025; Tire et al., 2026; Zhang et al., 2024). GTR (Cao et al., 2026a) retrieves from a learned global-cycle embedding rather than from past windows. The multivariate methods treat each retrieved window as one unit, and the foundation-model methods retrieve for each univariate series on its own. GNA combines both granularities in one multivariate model, trains its retrieval embedding to predict the future, and keeps the bank strictly causal while it grows at test time (App. H compares the mechanisms).

Learning the retriever. Retrieval-augmented language models (Lewis et al., 2020) can train the retriever from a downstream signal, for example by distilling the reader's attention into the retriever (Izacard & Grave, 2021). TimeRAF (Zhang et al., 2024) carries this over to forecasting and trains its retriever to rank higher the candidates that lower the forecaster's error. Our embedding uses the forecasting target more directly: it is trained to predict the future of each window, independently of the forecaster. Since futures are available only during training, this is learning with privileged information (Vapnik & Vashist, 2009).

3 Method

GNA turns retrieved past windows into *slots*: candidate futures, one per variate, which a learned gate combines with a persistence forecast and the backbone's own forecast. Its core is how the slots are built, at two granularities (Sec. 3.3), and how much each is trusted, per forecast step and variate (Sec. 3.4). A retrieval stage supplies the candidates (Sec. 3.2). Figure 1 shows the stages for one query. The backbone sees exactly the lookback window it would see without retrieval, so GNA works with any backbone that produces a forecast and per-variate tokens.

3.1 Problem setup and causal eligibility

We observe a multivariate series $\mathbf{x}_t \in \mathbb{R}^D$ with D variates. At forecast origin t, the query window is $\mathbf{X}_t = \mathbf{x}_{t-L+1:t} \in \mathbb{R}^{L\times D}$ and the target is $\mathbf{Y}_t = \mathbf{x}_{t+1:t+H} \in \mathbb{R}^{H\times D}$, with lookback L and horizon H; $h \in \{1, \dots, H\}$ denotes the forecast step. The bank contains every window $(\mathbf{X}_s, \mathbf{Y}_s)$ of the series, indexed by its origin s. A bank window may serve query t only if

$$s \le t - \max(L, H). \tag{1}$$

Then its future ends at $s + H \le t$, so it has been fully observed at the forecast origin, and its past ends at $s \le t - L$, so it does not overlap the query's past. The rule holds for training, validation and test queries alike: a training query retrieves only earlier training windows, and at test time the bank grows as new futures are observed, without retraining (App. A, Fig. 7). We audited all banks for this rule. Table 2 (App. A) lists all symbols.

3.2 Retrieving candidates

Retrieval should return windows whose futures resemble the query's unknown future, so we train the retrieval embedding with the futures of training windows as targets. A teacher ϕ_T, an MLP on

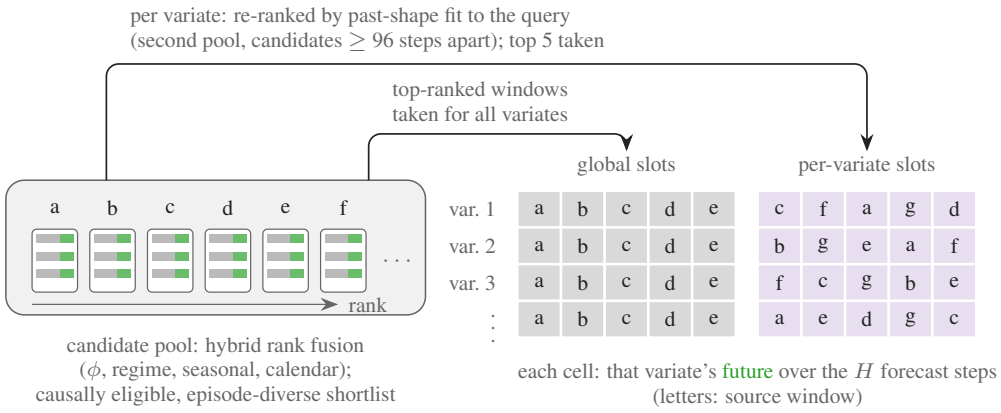


Figure 2: Neighbor assembly for one query. Left: fused candidates in rank order; letters name source windows, and each box shows one row per variate (grey: history, green: future). Middle: the global slots use the five top-ranked windows for all variates. Right: for the per-variate slots, each variate re-ranks a second, spaced candidate pool by how well its own past matches the query's past and takes its top five, so one column combines different source windows (variate 1: c, f, a, g, d). Each cell holds that variate's H-step future; the $K = 10$ columns are the slot futures $\mathbf{c}_1, \ldots, \mathbf{c}_K$ (Eq. 4).

the flattened window, maps window i to $\mathbf{z}_i = \phi_T(\mathbf{X}_i) \in \mathbb{R}^{d_z}$, and a head ψ, used only in training, predicts the window's future from it:

$$\mathcal{L}_T = \mathrm{MSE}\big(\psi(\mathbf{z}_i), \mathbf{Y}_i\big). \tag{2}$$

The embedding thus keeps what the past window says about its future. Futures appear only in the loss, so the embedder needs only the past at test time; this is learning with privileged information (Vapnik & Vashist, 2009). Retrieval uses a small Transformer ϕ distilled from the teacher (App. A, Fig. 6). For query t we search all eligible windows (Eq. 1) exactly, by cosine similarity of their ϕ embeddings. A learned embedding can miss coarse context such as the operating level or the time of day, so we combine its ranking with three simple context signals j: the similarity of low-dimensional regime features, lags at which the training series is periodic, and calendar match. With $\bar{r}(s) \in [0, 1]$ the normalized rank of candidate s under each signal, candidates are ordered by

$$\rho(s) = \bar{r}_\phi(s) + \textstyle\sum_j w_j \, \bar{r}_j(s), \tag{3}$$

lowest first, with fixed weights shared by all datasets (App. A); PEMS and Solar have no time stamps, so the calendar signal is not used there. The search keeps $C = 200$ candidates, spread over different episodes so that one episode cannot fill the list.

3.3 Granular neighbor assembly

Existing retrieval forecasters treat a retrieved window as one unit: every variate receives the future of the same past window. In a network of hundreds of sensors, however, the past moment that best matches one sensor rarely best matches all others. GNA therefore fills K *slots*, in which each variate can have its own source. Slot k assigns to every variate d an eligible source window $s_{k,d}$ and provides that variate's past and future, scaled with the mean $\mu_{t,d}$ and standard deviation $\sigma_{t,d}$ of variate d in the query window:

$$\mathbf{c}_k[:, d] = \big(\mathbf{Y}_{s_{k,d}}[:, d] - \mu_{t,d}\big)/\sigma_{t,d} \ \in \mathbb{R}^H. \tag{4}$$

We use two kinds of slots (Fig. 2). The five *global* slots take the five top-ranked candidates for all variates ($s_{k,d} = s_k$). They keep one past situation coherent across variates, such as a network-wide rush hour. The five *per-variate* slots choose $s_{k,d}$ separately for each variate. A second fused list, built as above but with candidates at least L steps apart, provides a pool of $M = 50$ candidates. Variate d ranks the candidates by how well *its own* past matches the query's past, and takes its top five. The match is a fixed multi-scale distance between the two univariate pasts (App. A). A

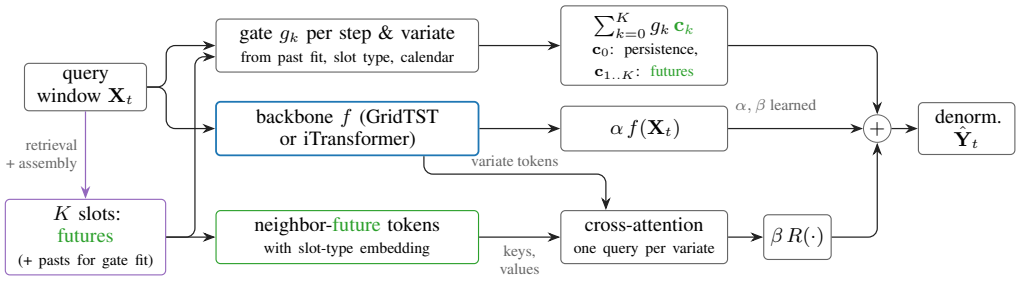


Figure 3: The retrieval-augmented forecaster (Eq. 5). Top: the gate compares each slot's past with the query's past per variate and, with slot type and calendar match, weights persistence $\mathbf{c}_0$ and the slot futures $\mathbf{c}_{1..K}$ per forecast step and variate. Middle: the backbone f forecasts from $\mathbf{X}_t$ alone, scaled by α. Bottom: each variate token of the backbone attends to its K slot-future tokens, and the result is projected to H steps and scaled by β. The sum is denormalized to $\hat{\mathbf{Y}}_t$.

per-variate slot is thus a composite that never occurred as one multivariate window. Global slots preserve cross-variate structure, while per-variate slots find a closer match when variates decouple. The spacing keeps per-variate slots from being near-copies of one episode.

3.4 Forecasting with learned trust

Let $\mathbf{c}_1, \ldots, \mathbf{c}_K \in \mathbb{R}^{H\times D}$ be the slot futures of Eq. 4 and $\mathbf{c}_0$ the persistence forecast, i.e. the query's last value repeated H times, all in the query's normalized space. The forecast is

$$\hat{\mathbf{Y}}_t = \mathrm{denorm}\Big(\sum_{k=0}^{K} \mathbf{g}_k \odot \mathbf{c}_k \;+\; \alpha\, f(\mathbf{X}_t) \;+\; \beta\, R\big(\mathbf{X}_t, \mathbf{c}_{1:K}\big)\Big), \tag{5}$$

where f is the backbone, α and β are learned scalars initialized at 0.1, and denorm inverts the query's instance normalization (Kim et al., 2022) (Fig. 3). The first term is a learned trust over the candidates. The gate weights $\mathbf{g}_k \in \mathbb{R}^{H\times D}$ sum to one over $k = 0, \ldots, K$ at every forecast step and variate. They are computed by a small MLP from how well slot k's past fits the query's past, for that variate and averaged over all variates, and from the slot type and calendar match (App. A). Persistence has its own learned logit per forecast step, so where no slot fits, the gate can fall back to it. The weights differ per forecast step, so the gate can trust persistence for the next few steps and retrieved futures further out. The second term keeps the backbone's own forecast. The third term, R, lets the backbone condition on the neighbors: each variate token of f attends to the K slot-future tokens of that variate (with a slot-type embedding), and the result is projected to H steps.

3.5 Training and cost

Training runs in four stages (Fig. 7): teacher embedder, distilled student, bank construction (embed all windows, rank the eligible candidates of every query, assemble the slots), and forecaster. The forecaster is trained end to end with MSE, with the embedder and bank fixed, using the backbone's training recipe (AdamW, weight decay 0.01, one-cycle schedule, 10 epochs, batch size 32; 8 for Traffic). Exact search costs $O(N(d_z + d_r))$ per query for a bank of N windows, so the bank for all queries is built in $O(N^2(d_z + d_r))$ time, and the index needs $O(N(d_z + d_r))$ memory. At our scales the bank is built in 3–7 minutes, the index takes 0.15–0.40 GB, training is 14–51% slower per epoch than the backbone alone, and online retrieval takes 19–25 ms per query on a CPU (App. G).

4 Experimental setup

Our experiments answer three questions: does retrieval improve forecasts under a strict same-lookback protocol (Sec. 5), which components carry the gain (Sec. 6), and when does retrieval help or fail (Sec. 7)?

Datasets. We use twelve standard benchmarks (App. C): the traffic sensor networks PEMS03/04/07/08 (as processed by Liu et al., 2022) with horizons $H \in \{12, 24, 48, 96\}$; ETTh1/h2 and ETTm1/m2 (Zhou et al., 2021); and ECL, Traffic, Weather (Wu et al., 2021) and Solar (Lai et al., 2018), all with $H \in \{96, 192, 336, 720\}$ and the standard chronological splits. We also use monitoring data from the trigger and data-acquisition (TDAQ) system of a large high-energy-physics experiment: 35 variables, one value per luminosity block (a data-taking interval of 10–60 s), organized in 195 data-taking runs of 2025; there we forecast one step ahead from $L = 5$ (App. J; 22 runs of one data-taking setting from 2024 are in App. D).

Baselines and protocol. Every model uses lookback $L = 96$. Baselines are the two backbones without retrieval, the retrieval-based GTR (Cao et al., 2026a) and RAFT (Han et al., 2025), and 14 further recent forecasters in App. D. We take published numbers at $L = 96$; where a method's own paper uses another lookback, we use $L = 96$ numbers reported by a third party (marked ‡). GridTST has no published $L = 96$ results, so we run it ourselves, with the same code, training recipe and checkpoint selection as GNA: without retrieval, Eq. 5 reduces to $\mathrm{denorm}(\mathbf{c}_0 + \alpha f(\mathbf{X}_t))$, i.e. persistence plus the scaled backbone forecast. We ran no hyperparameter search: GridTST architectures follow the official configurations, the retrieval constants (fusion and re-rank weights, K, spacing) are untuned defaults shared by all datasets, and the few per-dataset settings listed in App. B were set by hand. Tables report seed 2023; App. E repeats all datasets with three seeds.

Metrics. We report MSE and MAE on standardized data, averaged over the four horizons (per horizon in App. D), on the full test set (no dropped last batch). Every run keeps the checkpoint with the lowest validation MSE over its 10 epochs.

5 Results

Table 1 compares GNA with its backbones and with retrieval-based forecasters. With GridTST, GNA has the lowest MSE on 8 of the 12 datasets, also among all 18 baselines in App. D. The gains are largest on the PEMS sensor networks, where GNA roughly halves the MSE of the plain GridTST and is 5–44% below the best baseline. With a lookback of 96 five-minute steps (8 hours), these models see a third of the daily cycle, and retrieved windows supply how comparable past situations continued. The gain is more than a recovered daily cycle: GTR, CycleNet and TQNet model periodic structure explicitly, and GNA is 5–49% below each of them on every PEMS dataset. Likewise, ETTm windows cover a full day and ECL windows four days, yet GNA lowers the MSE of GridTST by 7% on ETTm1 and 12% on ECL. On ECL, Traffic, ETTm1 and Weather, GNA is 0.5–4.9% below the best baseline. Among retrieval-based forecasters, it is better than GTR on 8 of 12 datasets and better than RAFT on 6 of the 8 datasets RAFT reports. It is not the best on ETTm2 and Solar (2.9% and 2.1% above the best), and on ETTh1 and ETTh2 it is worse than its own backbone; Sec. 7 analyzes why.

Retrieval vs. no retrieval. The cleanest comparison holds the backbone, data and lookback fixed and changes only retrieval (App. D, Table 8). Retrieval improves GridTST in 43 and iTransformer in 42 of 48 dataset–horizon settings. Over three seeds, GNA beats the plain GridTST in every seed on 10 of 12 datasets (not on ETTh1/h2), and its seed-to-seed variation is small (median 0.3% of the MSE, vs. 4% for the plain backbone; App. E). Outside PEMS and ETTh, the seed-mean gains over GridTST are 4–21%, far above this variation. The gain is not explained by capacity: with iTransformer, GNA including the embedder has fewer parameters than the plain iTransformer in all 48 settings (App. B), and with mismatched neighbors, the same architecture performs worse than without retrieval (Sec. 6).

6 Ablations and sensitivity

We ablate on five datasets (PEMS04, PEMS08, ETTm1, ETTm2, Weather) with the GridTST backbone (Fig. 4; table in App. F). *Is the gain due to retrieval?* Feeding each query the neighbors of a different query (mismatched), with the same architecture trained and tested this way, raises the MSE by 6–247% (median 21%), more than removing retrieval (4–142%, median 8%), on all five datasets. Retrieved futures therefore help only when they are relevant, and the model cannot fully

Dataset	*GNA (this work)*				*Backbones*				*Retrieval-based*			
	w/ GridTST		w/ iTransformer		GridTST arXiv 2024, our run		iTransformer ICLR 2024b		GTR ICLR 2026a		RAFT‡ ICML 2025	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
PEMS03	**0.082**	**0.182**	0.084	0.183	0.153	0.260	0.113	0.221	0.087	0.189	—	—
PEMS04	**0.080**	**0.182**	0.083	0.184	0.174	0.283	0.111	0.221	0.087	0.189	—	—
PEMS07	**0.065**	**0.155**	0.067	0.156	0.134	0.241	0.101	0.204	0.076	0.169	—	—
PEMS08	**0.076**	**0.176**	**0.076**	**0.176**	0.184	0.275	0.150	0.226	0.142	0.222	—	—
ETTh1	0.463	0.459	0.473	0.464	0.456	0.448	0.454	0.447	0.439	0.434	**0.428**	0.433
ETTh2	0.420	0.437	0.427	0.441	0.395	0.412	0.383	0.407	0.372	0.400	0.382	0.410
ETTm1	**0.359**	**0.386**	0.365	0.389	0.386	0.402	0.407	0.410	0.367	0.389	0.381	0.400
ETTm2	0.272	0.329	0.278	0.331	0.287	0.332	0.288	0.332	0.268	0.315	0.281	0.330
ECL	**0.153**	**0.252**	0.154	0.255	0.174	0.266	0.178	0.270	0.166	0.260	0.175	0.272
Solar	0.198	0.245	0.201	**0.243**	0.228	0.278	0.233	0.262	**0.194**	0.245	0.301	0.303
Traffic	**0.397**	0.269	0.409	**0.263**	0.412	0.271	0.428	0.282	0.470	0.280	0.414	0.284
Weather	**0.235**	0.270	0.239	0.273	0.245	0.272	0.258	0.278	0.239	**0.268**	0.270	0.309
1st count	**8**	**6**	1	3	0	0	0	0	1	1	1	0

Dataset	*Other forecasters*													
	TQNet ICML 2025		Amplifier AAAI 2025		CycleNet NeurIPS 2024		SOFTS NeurIPS 2024		TimeXer NeurIPS 2024b		TimeMixer ICLR 2024a		PatchTST‡ ICLR 2023	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
PEMS03	0.097	0.203	0.131‡	0.239‡	0.118‡	0.226‡	0.104	0.210	—	—	0.167‡	0.267‡	0.180	0.291
PEMS04	0.091	0.197	0.135‡	0.249‡	0.119‡	0.232‡	0.102	0.208	—	—	0.185‡	0.287‡	0.195	0.307
PEMS07	0.075	0.171	0.122‡	0.226‡	0.113‡	0.214‡	0.087	0.184	—	—	0.181‡	0.271‡	0.211	0.303
PEMS08	0.142	0.229	0.183‡	0.271‡	0.150‡	0.246‡	0.138	0.219	—	—	0.226‡	0.299‡	0.280	0.321
ETTh1	0.441	0.434	0.430	**0.428**	0.457	0.441	0.449	0.442	0.437	0.437	0.447	0.440	0.469	0.454
ETTh2	0.378	0.402	**0.359**	**0.391**	0.388	0.409	0.373	0.400	0.367	0.396	0.364	0.395	0.387	0.407
ETTm1	0.377	0.393	0.381	0.394	0.379	0.396	0.393	0.403	0.382	0.397	0.381	0.395	0.387	0.400
ETTm2	0.277	0.323	0.276	0.323	**0.266**	**0.314**	0.287	0.330	0.274	0.322	0.275	0.323	0.281	0.326
ECL	0.164	0.259	0.171	0.265	0.168	0.259	0.174	0.264	0.171	0.270	0.182	0.272	0.205	0.290
Solar	0.198	0.256	0.241‡	0.270‡	0.210	0.261	0.229	0.256	0.237‡	0.302‡	0.216	0.280	0.270	0.307
Traffic	0.445	0.276	0.482	0.315	0.472	0.301	0.409	0.267	0.466	0.287	0.484	0.297	0.481	0.304
Weather	0.242	0.269	0.243	0.271	0.243	0.271	0.255	0.278	0.241	0.271	0.240	0.271	0.259	0.281
1st count	0	0	1	2	1	1	0	0	0	0	0	0	0	0

Table 1: Results with lookback 96, averaged over horizons (PEMS: 12/24/48/96; others: 96/192/336/720). Baselines from the respective publications, except GridTST (our run; no published lookback-96 results). ‡Lookback-96 result reported by a third party (sources in App. D). Best in red bold, second underlined.

ignore irrelevant ones (Sec. 8). *Are both granularities needed?* Using only global slots raises the MSE by 1.9–8.3% (mean 4.8%), using only per-variate slots by 0.3–15.8% (mean 7.0%), so the mixture beats either slot type alone. Measured on the retrieved futures themselves, the best per-variate slot is 17–45% closer to the true future than the best global slot on all five datasets (App. F). *How much does the retrieval stage add?* Its parts add smaller, dataset-dependent gains. Ranking with standardized raw windows instead of the learned embedding raises the MSE by 0.9–7.4% (mean 3.7%). Ranking by the embedding alone, without the context signals, raises it where calendar and regime structure is informative (ETTm1, ETTm2, Weather: $+4.1$, $+12.0$, $+4.5\%$), but not on the PEMS sensors ($+0.4\%$, -1.1%). *How many slots?* At $H = 96$, halving the number of slots to $K = 5$ raises the MSE by 4.5% on PEMS08 and 2.1% on ETTm1, while doubling it to $K = 20$ lowers it by only 0.9% and 0.6%: the default $K = 10$ is close to saturation, and even $K = 5$ keeps most of the gain (PEMS08: 71% below the plain backbone; Table 14).

7 When does retrieval help?

The results follow one pattern: retrieval helps most where the lookback says least.

Further ahead, the gate trusts neighbors more. Figure 5 shows that the gate puts substantial weight on persistence in the first forecast steps and moves it to retrieved futures further out: averaged over test windows, the persistence weight falls from steps 1–12 to steps 49–96 from 0.29 to 0.03 on

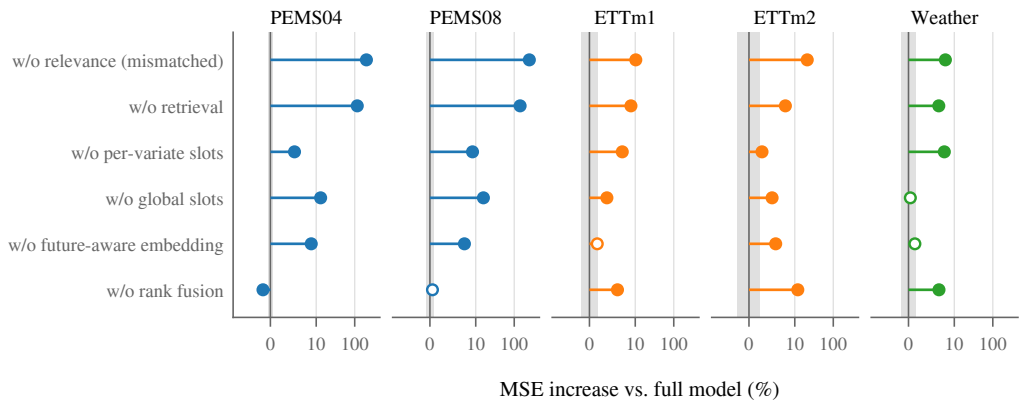


Figure 4: Ablations (GridTST backbone): change in horizon-averaged test MSE relative to the full GNA; positive = worse without the component. Filled: larger than twice the full model's seed standard deviation on that dataset (three seeds); hollow: within this noise. Symmetric-log x axis (linear within $\pm 5\%$); ablation runs use one seed.

PEMS04 and from 0.42 to 0.08 on Solar. Across all datasets, horizons and both backbones, the more weight the gate puts on neighbors, the larger the gain over the backbone (Spearman $\rho = 0.71$ for GridTST, 0.53 for iTransformer; App. I).

At run starts, retrieval helps most. On TDAQ, every run start is a discontinuity that a short lookback cannot see beyond. On the 2025 data, GNA lowers the MAE by 2.4–3.2% overall (three seeds), but the gain concentrates at run starts (App. J, Fig. 15): -20.8% for the first forecast of each run, whose lookback covers the run's first five luminosity blocks, -14.8% for forecast origins at blocks 5–9 and -9.3% at blocks 10–24, against -1 to -4% later. Pooled over the first 25 blocks of a run, the MAE change is -10.5 to -12.7% per seed, run-level bootstrap intervals exclude zero, and GNA is better in 29–30 of 36 test runs. The retrieved windows lie thousands of windows back, and 89–98% come from the query's operating regime (App. J): retrieval supplies how earlier runs evolved from the same phase.

Longer lookbacks. On ETTm1 ($H = 96$), GNA lowers the MSE of GridTST by 33% at $L = 48$ and 8% at $L = 96$; the gain falls to 1% at $L = 192$, and at $L = 336$ GNA is 5% worse (Table 14). The error of GNA changes little with the lookback (0.297–0.310), whereas the plain backbone needs a long window to reach it: retrieval substitutes for context that a short lookback lacks, but on ETTm1 it does not add to a long one. GNA is therefore most useful where the lookback has to stay short, such as at the start of data-taking runs, where its gains on TDAQ are largest (above).

Where retrieval fails: ETTh. On ETTh1 and ETTh2, GNA is worse than GridTST, and the loss is a far-horizon effect: at $H = 720$ on ETTh1, GNA is 5.5% better over steps 1–96 and 16.5% worse over steps 337–720. The evidence points to the level of retrieved futures drifting from the truth far ahead: a hindsight oracle over the same eligible windows shows that level-free futures would reduce the error by 26% (ETTh1) and 36% (ETTh2), while raw ones barely help (App. I).

8 Limitations and Future Work

GNA needs a history of windows with observed futures, so it cannot help at a cold start. The level drift of retrieved futures limits it on non-stationary series at long horizons, and the gate cannot fully ignore irrelevant neighbors: with mismatched neighbors the model is worse than without retrieval (Sec. 6), and on ETTh it keeps weight on drifting futures. On ETTm1, retrieval does not improve a long lookback ($L = 336$). The fixed fusion weights slightly hurt on PEMS04. Our training cache of assembled slots grows as $O(NKD(L + H))$ (36 GB on ECL), although the retrieval index itself is small. The main table and the ablations use one seed (three seeds in App. E), we evaluate at $L = 96$ except for one lookback study on ETTm1, and the shared retrieval settings were not tuned

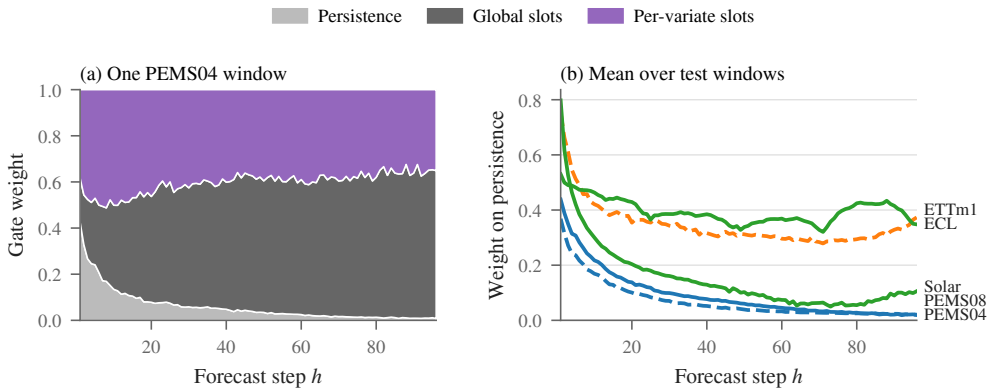


Figure 5: The gate moves weight from persistence to retrieved futures along the horizon ($H = 96$, GridTST). (a) Gate weight per forecast step for one PEMS04 test window (median gain over the backbone; the variate with the largest target spread), by candidate group. (b) Mean weight on persistence over 64 test windows and all variates, per dataset.

per dataset. Future work includes a learned level correction of retrieved futures, better per-variate retrieval (App. I), approximate search, and forecasting-based anomaly detection (App. K).

9 Conclusion

We presented GNA, a retrieval layer that assembles neighbors both as whole past windows and per variate, and learns per forecast step and variate how much to trust them. Under a strict protocol, with a fixed lookback, causal retrieval and one retrieval configuration for all datasets, it improves two Transformer backbones in most settings of twelve benchmarks and gives the lowest MSE on eight of them. Both granularities are needed, and the gains are largest where the lookback says least: far ahead, on sensor networks, and at the start of data-taking runs. Its failures on hourly non-stationary series point to level-aware use of retrieved futures as the next step.

AI use statement

In this work, we used generative AI tools (an LLM-based coding assistant) for implementation, for running and analyzing experiments, for help with result interpretation, and for help with methodology design: the assistant proposed and screened variants of individual components, while the central idea of granular neighbor assembly and the final design are the authors'. We also used them for data cleaning of the monitoring data (preprocessing code). We have not used generative AI tools for synthetic data generation, theoretical model development, mathematical formulation, proof assistance or translation, and qualitative analysis is not applicable to this work. Additionally, we used generative AI tools for figure creation, code editing, literature analysis (checking citations and related work), paper drafting and proofreading: the tools drafted parts of the text, which the authors edited and verified. We have reviewed all AI-assisted work: numbers in the text were checked against the stored result files, the causality of retrieval was audited programmatically for every retrieval bank, and every reference was checked against its source. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

Ethics statement

This work uses public benchmark datasets and technical monitoring data of a physics experiment; neither contains personal data. We see no specific ethical risks beyond those of forecasting in general.

Reproducibility statement

Sec. 3 and App. A describe the method; App. B lists all hyperparameters, App. C the datasets and splits, and App. G the hardware and cost. The eligibility rule (Eq. 1) was audited for every bank. All public-benchmark results use fixed seeds; App. E reports three seeds. Code will be released upon acceptance. The TDAQ monitoring data are not intended for release at this time.

REFERENCES

Fanpu Cao, Lu Dai, Jindong Han, and Hui Xiong. Enhancing multivariate time series forecasting with global temporal retrieval. In *The Fourteenth International Conference on Learning Representations*, 2026a. URL https://openreview.net/forum?id=QUJBPSfyui.

Fanpu Cao, Shu Yang, Zhengjian Chen, Ye Liu, and Laizhong Cui. Ister: Linear transformer for efficient multivariate time series forecasting. In *ICASSP 2026 - 2026 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3571–3575. IEEE, May 2026b. doi: 10.1109/icassp55912.2026.11463971. URL http://dx.doi.org/10.1109/ICASSP55912.2026.11463971.

Xin Cheng, Xiuying Chen, Shuqi Li, Di Luo, Xun Wang, Dongyan Zhao, and Rui Yan. Leveraging 2D information for long-term time series forecasting with vanilla transformers. *arXiv e-prints*, (arXiv:2405.13810):arXiv:2405.13810, May 2024. doi: 10.48550/arXiv.2405.13810.

Jingru Fei, Kun Yi, Wei Fan, Qi Zhang, and Zhendong Niu. Amplifier: Bringing attention to neglected low-energy components in time series forecasting. In *Proceedings of the Thirty-Ninth AAAI Conference on Artificial Intelligence and Thirty-Seventh Conference on Innovative Applications of Artificial Intelligence and Fifteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'25/IAAI'25/EAAI'25. AAAI Press, 2025. ISBN 978-1-57735-897-8. doi: 10.1609/aaai.v39i11.33267. URL https://doi.org/10.1609/aaai.v39i11.33267.

Lu Han, Xu-Yang Chen, Han-Jia Ye, and De-Chuan Zhan. SOFTS: Efficient multivariate time series forecasting with series-core fusion. In *The Thirty-Eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=89AUi5L1uA.

Sungwon Han, Seungeon Lee, Meeyoung Cha, Sercan O Arik, and Jinsung Yoon. Retrieval augmented time series forecasting. In *Forty-Second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=GUDnecJdJU.

Kyle Hundman, Valentino Constantinou, Christopher Laporte, Ian Colwell, and Tom Soderstrom. Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 387–395, 2018.

Gautier Izacard and Edouard Grave. Distilling knowledge from reader to retriever for question answering. In *ICLR 2021 - 9th International Conference on Learning Representations*, Vienna, Austria, May 2021. URL https://hal.science/hal-03463398.

Taesung Kim, Jinhee Kim, Yunwon Tae, Cheonbok Park, Jang-Ho Choi, and Jaegul Choo. Reversible instance normalization for accurate time-series forecasting against distribution shift. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=cGDAkQo1C0p.

Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long- and short-term temporal patterns with deep neural networks. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, Sigir '18, pp. 95–104, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 978-1-4503-5657-2. doi: 10.1145/3209978.3210006. URL https://doi.org/10.1145/3209978.3210006.

Jeffrey K. Lazo, Rebecca E. Morss, and Julie L. Demuth. 300 billion served: Sources, perceptions, uses, and values of weather forecasts. *Bulletin of the American Meteorological Society*, 90(6): 785–798, 2009. doi: 10.1175/2008BAMS2604.1. URL https://api.semanticscholar.org/CorpusID:17851127.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive NLP tasks. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 9459–9474. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/6b493230205f780e1bc26945df7481e5-Paper.pdf.

Shengsheng Lin, Weiwei Lin, Xinyi HU, Wentai Wu, Ruichao Mo, and Haocheng Zhong. CycleNet: Enhancing time series forecasting through modeling periodic patterns. In *The Thirty-Eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=clBiQUgj4w.

Shengsheng Lin, Haojun Chen, Haijie Wu, Chunyun Qiu, and Weiwei Lin. Temporal query network for efficient multivariate time series forecasting. In *Forty-Second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=e24CueVty2.

Jingwei Liu, Ling Yang, Hongyan Li, and Shenda Hong. Retrieval-augmented diffusion models for time series forecasting. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 2766–2786. Curran Associates, Inc., 2024a. doi: 10.52202/079017-0091. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/053ee34c0971568bfa5c773015c10502-Paper-Conference.pdf.

Minhao Liu, Ailing Zeng, Muxi Chen, Zhijian Xu, Qiuxia LAI, Lingna Ma, and Qiang Xu. SCINet: Time series modeling and forecasting with sample convolution and interaction. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 5816–5828. Curran Associates, Inc., 2022. doi: 10.52202/068431-0421. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/266983d0949aed78a16fa4782237dea7-Paper-Conference.pdf.

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 B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 11116–11140, 2024b. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/2ea18fdc667e0ef2ad82b2b4d65147ad-Paper-Conference.pdf.

Edward N. Lorenz. Atmospheric predictability as revealed by naturally occurring analogues. *Journal of the Atmospheric Sciences*, 26(4):636–646, 1969. doi: 10.1175/1520-0469(1969)26<636:APARBN>2.0.CO;2.

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.

Kanghui Ning, Zijie Pan, Yu Liu, Yushan Jiang, James Y. Zhang, Kashif Rasul, Anderson Schneider, Lintao Ma, Yuriy Nevmyvaka, and Dongjin Song. TS-RAG: Retrieval-augmented generation based time series foundation models are stronger zero-shot forecaster. In *The Thirty-Ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=PymOnHw4Ty.

Kutay Tire, Ege Onur Taga, Muhammed Emrullah Ildiz, and Samet Oymak. Retrieval Augmented Time Series Forecasting, April 2026. URL http://arxiv.org/abs/2411.08249.

Vladimir Vapnik and Akshay Vashist. A new learning paradigm: Learning using privileged information. *Neural Networks*, 22(5–6):544–557, 2009. doi: 10.1016/j.neunet.2009.06.042.

Shiyu Wang, Haixu Wu, Xiaoming Shi, Tengge Hu, Huakun Luo, Lintao Ma, James Y. Zhang, and JUN ZHOU. TimeMixer: Decomposable multiscale mixing for time series forecasting. In *The Twelfth International Conference on Learning Representations*, 2024a. URL https://openreview.net/forum?id=7oLshfEIC2.

Yulong Wang, Yushuo Liu, Xiaoyi Duan, and Kai Wang. FilterTS: Comprehensive frequency filtering for multivariate time series forecasting. In *Proceedings of the Thirty-Ninth AAAI Conference on Artificial Intelligence and Thirty-Seventh Conference on Innovative Applications of Artificial Intelligence and Fifteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'25/IAAI'25/EAAI'25. AAAI Press, 2025a. ISBN 978-1-57735-897-8. doi: 10.1609/aaai.v39i20.35438. URL https://doi.org/10.1609/aaai.v39i20.35438.

Yuxuan Wang, Haixu Wu, Jiaxiang Dong, Guo Qin, Haoran Zhang, Yong Liu, Yunzhong Qiu, Jianmin Wang, and Mingsheng Long. TimeXer: Empowering transformers for time series forecasting with exogenous variables. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 469–498. Curran Associates, Inc., 2024b. doi: 10.52202/079017-0015. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/0113ef4642264adc2e6924a3cbbdf532-Paper-Conference.pdf.

Zihan Wang, Fanheng Kong, Shi Feng, Ming Wang, Xiaocui Yang, Han Zhao, Daling Wang, and Yifei Zhang. Is Mamba effective for time series forecasting? *Neurocomputing*, 619: 129178, 2025b. ISSN 0925-2312. doi: 10.1016/j.neucom.2024.129178. URL https://www.sciencedirect.com/science/article/pii/S0925231224019490.

Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 22419–22430. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/bcc0d400288793e8bdcd7c19a8ac0c2b-Paper.pdf.

Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. TimesNet: Temporal 2D-variation modeling for general time series analysis. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=ju_Uqw384Oq.

Sidney Yakowitz. Nearest-neighbour methods for time series analysis. *Journal of Time Series Analysis*, 8(2):235–247, 1987. doi: 10.1111/j.1467-9892.1987.tb00435.x.

Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence and Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence and Thirteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'23/IAAI'23/EAAI'23. AAAI Press, 2023. ISBN 978-1-57735-880-0. doi: 10.1609/aaai.v37i9.26317. URL https://doi.org/10.1609/aaai.v37i9.26317.

Huanyu Zhang, Chang Xu, Yi-Fan Zhang, Zhang Zhang, Liang Wang, Jiang Bian, and Tieniu Tan. TimeRAF: Retrieval-augmented foundation model for zero-shot time series forecasting. *arXiv preprint arXiv:2412.20810*, 2024.

Yunhao Zhang and Junchi Yan. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=vSVLM2j9eie.

Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(12):11106–11115, May 2021. ISSN 2374-3468, 2159-5399. doi: 10.1609/aaai.v35i12.17325. URL https://ojs.aaai.org/index.php/AAAI/article/view/17325.

Yingbo Zhou, Yutong Ye, Zhiwei Ling, Shuhao Li, Rui Qian, Jian Xiong, Li Sun, and Dejing Dou. PAMNet: Cycle-aware phase-amplitude modulation network for multivariate time series forecasting, 2026. URL https://arxiv.org/abs/2605.02938.

A METHOD DETAILS

This section shows the three components of GNA in detail and the order in which they are trained and used. Figure 7 shows the stages from training to deployment.

Slots. Slot futures follow Eq. 4; slot pasts are scaled in the same way and are used only by the gate. The per-variate match score (Sec. 3.3) compares the univariate pasts of query and candidate in the query's normalization: 0.30 times their MSE over the full window, 0.25 times their MSE over the last 32 steps, 0.20 times the MSE of their first differences, 0.10 times the squared difference of the standard deviations of their first differences, and 0.15 times their MSE on every $(L/16)$-th step.

Symbol	Meaning
D, L, H	number of variates; lookback; horizon
t, s, h	origin of the query; origin of a bank window; forecast step
$\mathbf{X}_t$, $\mathbf{Y}_t$, $\hat{\mathbf{Y}}_t$	query window $\mathbf{x}_{t-L+1:t}$; target $\mathbf{x}_{t+1:t+H}$; forecast
$\mu_{t,d}$, $\sigma_{t,d}$	mean and standard deviation of variate d in the query window
ϕ_T, ψ, ϕ	teacher embedder; training-only future head; retrieval embedder (student)
$\mathbf{z}_i$, d_z	teacher embedding of window i; its size
P_i, Q_i, τ_F, τ_z, λ	target and embedding neighbor distributions; temperatures; KL weight
N, C, M	windows in the bank; fused candidates per query (200); per-variate pool (50)
$\rho(s)$, $\bar{r}_\phi$, $\bar{r}_j$, w_j	fused rank of candidate s (Eq. 3); normalized ranks; weight of signal j
w_{reg}, w_{seas}, w_{cal}, d_r	weights of the three context signals; size of the regime features (App. A)
K, $s_{k,d}$	slots per query (5 global, 5 per-variate); source window of slot k for variate d
$\mathbf{c}_0$, $\mathbf{c}_k$	persistence forecast; future of slot k, in the query's normalization (Eq. 4)
$\mathbf{g}_k$	gate weights per step and variate, summing to one over $k = 0, \ldots, K$
f, R, α, β	backbone; cross-attention term; their learned scales (Eq. 5)

Table 2: Notation.

Retrieval embedding. Figure 6 shows how the embedder is trained. All reported runs add a small auxiliary term to Eq. 2, $\lambda\,\mathrm{KL}(P_i \,\|\, Q_i)$ with $\lambda = 0.1$, which matches the embedding's neighbor distribution over a batch, $Q_i(j) = \mathrm{softmax}_{j\neq i}(\cos(\mathbf{z}_i, \mathbf{z}_j)/\tau_z)$, to one defined by future similarity, $P_i(j) = \mathrm{softmax}_{j\neq i}(-\mathrm{MSE}(\mathbf{Y}_i, \mathbf{Y}_j)/\tau_F)$, with $\tau_F = 0.5$ and $\tau_z = 0.1$. Removing it changes the test MSE of the full model on ETTm2 by -0.2% on average (-0.8 to $+1.3\%$ per horizon), so the main text presents the future-prediction objective alone. The retrieval embedder ϕ is a small Transformer over the L time steps, distilled from the frozen teacher by minimizing $\mathrm{MSE}(\phi(\mathbf{X}), \phi_T(\mathbf{X})) + 1 - \cos(\phi(\mathbf{X}), \phi_T(\mathbf{X}))$ over training windows. The student is much smaller than the teacher, whose first layer alone has $512\,LD$ weights (42M on Traffic), so embedding every window of the bank stays cheap.

Rank fusion. Eq. 3 uses three context signals. *Regime*: the Euclidean distance between fixed regime features of the two windows (a d_r-dimensional linear projection and binned statistics, fitted on training windows), weight $w_{\text{reg}} = 1$. *Seasonality*: $\bar{r}_{\text{seas}}(s) = 0$ if $t - s$ is one of up to eight lags at which the autocorrelation of the training series peaks (computed on the leading regime components) and 1 otherwise, weight $w_{\text{seas}} = 0.25$. *Calendar*: the rank of the mismatch in time of day and week, weight $w_{\text{cal}} = 0.5$. All ranks are normalized to $[0, 1]$. The shortlist takes candidates greedily in rank order at a minimum distance from those already taken.

Gate inputs. For slot k and variate d, the gate MLP receives: the MSE and MAE between the slot's past and the query's past, their correlation, the absolute gap at the last observed step, the variance of the slot's past, the slot type (global or per-variate), the MSE averaged over all variates (whether the past situation fits the whole system), and the calendar match between the slot's and the query's time stamps (ETT, ECL, Traffic and Weather). It outputs one logit per forecast step. Two learned per-step scales add $\tanh$ of the difference between $\mathbf{c}_k$ and the query's last value (the change the slot proposes) and $|\mathbf{c}_k - \bar{\mathbf{c}}|$ (disagreement with the mean slot future) to the logits. Persistence $\mathbf{c}_0$ has its own learned per-step logit, and a softmax over $k = 0, \ldots, K$ gives $\mathbf{g}_k$. The retrieval distance is not a gate input.

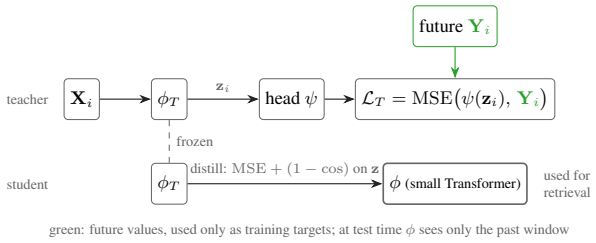


Figure 6: Training the future-aware embedder. Top: the teacher ϕ_T embeds a training window, and the head ψ predicts the window's future from the embedding $\mathbf{z}_i$ (Eq. 2); green: future values, used only as training targets. Bottom: the small Transformer ϕ used for retrieval is distilled from the frozen teacher and sees only the past window.

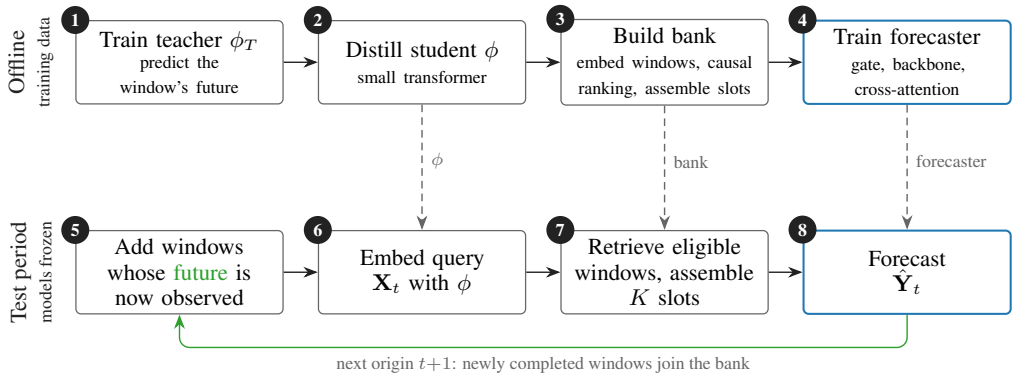


Figure 7: Order of operations. Offline, on training data only (top): (1) train the teacher embedder; (2) distill the student ϕ; (3) build the bank: embed all windows, rank the causally eligible candidates of every query and assemble their slots; (4) train the forecaster on these slots. Test period, with frozen models (bottom), for each forecast origin t: (5) add the windows whose futures have been observed by t to the bank; (6) embed the query $\mathbf{X}_t$ with ϕ; (7) retrieve eligible windows and assemble K slots; (8) forecast $\hat{\mathbf{Y}}_t$. Dashed arrows: the embedder, bank and forecaster carried over from training. Green arrow: the bank grows as t advances, without retraining.

B Implementation details and hyperparameters

Table 3 lists the settings shared by all datasets, horizons and both backbones: the embedder objective, retrieval, fusion and assembly. Tables 4 and 5 give the per-dataset backbone and embedder settings. GridTST architectures follow the configurations of the official repository where one exists. We ran no systematic hyperparameter search. Table 6 compares parameter counts: with iTransformer, GNA including the retrieval embedder has fewer parameters than the plain backbone on every dataset and horizon.

Group	Setting	Value
Protocol	Lookback L	96
	Seed (single-seed runs)	2023
	Retrieval policy	causal history index
	Candidate eligibility	non-overlapping with query
Embedder	Architecture	Transformer
	Objective	future-MSE ranking
	Future temperature τ_F	0.5
	Embedding temperature τ_z	0.1
	Ranking-loss weight	0.1
	Learning rate	0.0003
	Max epochs / patience	50 / 10
Retrieval	Neighbors per query K	10
	Global (coherent) slots	5
	Neighbor assembly	mixed (global + per-variate)
	Global slot spacing (steps)	1
	Per-variate slot spacing (steps)	96
	Candidate pool	50
	Shortlist policy	episode-diverse
	Ranking	hybrid rank fusion
	Ranked candidates (hybrid)	200
	Regime weight	1
	Seasonal weight	0.25
	Calendar weight	0.5
	Neighbor future transport	raw
Forecaster	Max epochs / patience	10 / 10

Table 3: Hyperparameters shared by all datasets, horizons and both backbones.

	GridTST backbone and training					
Dataset	batch	lr	d_{model}	layers	patch	stride
PEMS03	32	0.001	64	3	16	16
PEMS04	32	0.0005	64	3	16	16
PEMS07	32	0.001	64	3	16	16
PEMS08	32	0.001	64	3	16	16
ETTh1	32	0.001	96/96/128/256	3	48/48/48/64	48/48/96/16
ETTh2	32	0.001/0.001/0.001/0.0001	96/96/128/256	3	48/48/48/64	48/48/96/16
ETTm1	32	0.001	64/64/32/64	3	48/48/32/4	48/48/16/4
ETTm2	32	0.001	64/64/32/64	3	48/48/32/4	48/48/16/4
ECL	32	0.001	256	3	8	8
Solar	32	0.001	64	3	16	16
Traffic	8	0.001	256	5	16	8
Weather	32	0.001	128	4	16	16

Table 4: Per-dataset GridTST settings; a/b/c/d are the values for the four horizons in order.

	iTransformer				Retrieval embedder				
Dataset	batch	lr	d_{model}	layers	d_{model}	emb.	layers	heads	cal.
PEMS03	32	0.001	384	4	256	128	2	4	no
PEMS04	32	0.0005	896	4	256	128	2	4	no
PEMS07	32	0.001	64/64/256/256	3/3/4/4	256	128	2	4	no
PEMS08	32	0.001	64/64/256/256	3/3/4/4	256	128	2	4	no
ETTh1	32	0.001	64	1	32	32	1	2	yes
ETTh2	32	0.001/0.001/0.001/0.0001	64	1	32	32	1	2	yes
ETTm1	32	0.001	64	1	32	32	1	2	yes
ETTm2	32	0.001	64	1	32	32	1	2	yes
ECL	32	0.001	350	2	256	128	2	4	yes
Solar	32	0.001	200	4	256	128	2	4	no
Traffic	8	0.001	300	4	256	128	2	4	yes
Weather	32	0.001	300	2	32	128	1	2	yes

Table 5: Per-dataset iTransformer and embedder settings; a/b/c/d are the values for the four horizons in order; cal.: weight of the calendar term in the ranking.

	GridTST			iTransformer			
Dataset	Backbone	GNA	Ratio	Backbone	GNA	Ratio	Embedder
PEMS03	0.16 – 0.20	2.59 – 2.70	13.84 – 16.46	6.37 – 6.41	6.01 – 6.11	0.94 – 0.95	1.81
PEMS04	0.16 – 0.20	5.25 – 5.45	27.90 – 33.34	25.32 – 25.41	24.44 – 24.67	0.97	1.79
PEMS07	0.16 – 0.20	2.14 – 2.49	12.69 – 13.98	3.21 – 6.41	2.06 – 3.91	0.61 – 0.64	1.96
PEMS08	0.16 – 0.19	1.93 – 2.29	11.73 – 12.90	3.21 – 6.41	1.85 – 3.71	0.57 – 0.58	1.75
ETTh1	0.29 – 1.94	0.36 – 2.12	1.09 – 1.25	0.84 – 3.58	0.10 – 0.24	0.05 – 0.14	0.02
ETTh2	0.29 – 1.94	0.36 – 2.12	1.09 – 1.25	0.22 – 0.30	0.10 – 0.24	0.43 – 0.79	0.02
ETTm1	0.13 – 1.30	0.23 – 1.47	1.13 – 1.78	0.22 – 0.30	0.10 – 0.24	0.43 – 0.79	0.02
ETTm2	0.13 – 1.30	0.23 – 1.47	1.13 – 1.78	0.22 – 0.30	0.10 – 0.24	0.43 – 0.79	0.02
ECL	1.51 – 3.59	4.00 – 6.60	1.84 – 2.64	4.83 – 5.15	3.92 – 4.66	0.81 – 0.91	1.80
Solar	0.19 – 0.48	2.16 – 2.72	5.73 – 11.07	3.26 – 3.58	2.96 – 3.37	0.91 – 0.94	1.74
Traffic	2.28 – 4.20	4.77 – 7.15	1.70 – 2.09	6.41 – 6.73	4.62 – 5.27	0.72 – 0.78	1.95
Weather	0.62 – 1.18	1.14 – 2.16	1.83 – 1.84	4.83 – 5.15	1.62 – 2.26	0.33 – 0.44	0.02

Table 6: Trainable parameters in millions (ranges over horizons). GNA: forecaster plus retrieval embedder; ratio: GNA / backbone.

C Datasets

Table 7 summarizes the datasets. We use the standard chronological splits and standardize every variate with the mean and standard deviation of the training split; all errors are computed on this scale. The TDAQ series are monitoring data of the trigger and data-acquisition system of a large high-energy-physics experiment. One time step is a luminosity block, an interval of 10–60 s over which data-taking conditions are treated as constant. The 35 variables are first-level trigger rates: the total accept rate, the rates of 14 trigger items (electron, tau, jet, missing-energy and muon signatures), each counted before prescaling and after veto, and six muon-trigger sector rates. The data are organized in data-taking runs with gaps between them: 195 runs from 2025 (median 651 luminosity blocks per run; 10th to 90th percentile 95 to 955) and, as a small second set, 22 runs of one data-taking setting from 2024. We forecast one step ahead from $L = 5$ steps. MSE on TDAQ is dominated by rare spikes (the top 0.01% of points carry 90–99% of the squared error), so we report MAE there.

Dataset	Variates	Sampling	Train	Val	Test	Split	Lookback	Horizons
PEMS03	358	5 min	15,724	5,242	5,242	60:20:20	96	{12, 24, 48, 96}
PEMS04	307	5 min	10,195	3,398	3,399	60:20:20	96	{12, 24, 48, 96}
PEMS07	883	5 min	16,934	5,645	5,645	60:20:20	96	{12, 24, 48, 96}
PEMS08	170	5 min	10,713	3,571	3,572	60:20:20	96	{12, 24, 48, 96}
ETTh1/ETTh2	7	1 h	8,640	2,880	2,880	60:20:20	96	{96, 192, 336, 720}
ETTm1/ETTm2	7	15 min	34,560	11,520	11,520	60:20:20	96	{96, 192, 336, 720}
ECL	321	1 h	18,412	2,632	5,260	70:10:20	96	{96, 192, 336, 720}
Solar	137	10 min	36,792	5,256	10,512	70:10:20	96	{96, 192, 336, 720}
Traffic	862	1 h	12,280	1,756	3,508	70:10:20	96	{96, 192, 336, 720}
Weather	21	10 min	36,887	5,270	10,539	70:10:20	96	{96, 192, 336, 720}
TDAQ 2024	33	10–60 s	7,870	1,126	2,248	70:10:20	5	1
TDAQ 2025	35	10–60 s	80,633	11,519	23,038	70:10:20	5	1

Table 7: Dataset statistics. Train, Val and Test give the number of windows.

D Full results

Table 8 compares each backbone with and without retrieval. The per-horizon results and the comparison with all 18 baselines are in Tables 9 and 10. Figure 8 shows the MSE per horizon, and Fig. 9 how the gate distributes its weight on average. On the 22 TDAQ runs from 2024, retrieval lowers the MAE of iTransformer by 3.5% but raises that of GridTST by 2.0% (MSE -0.9%).

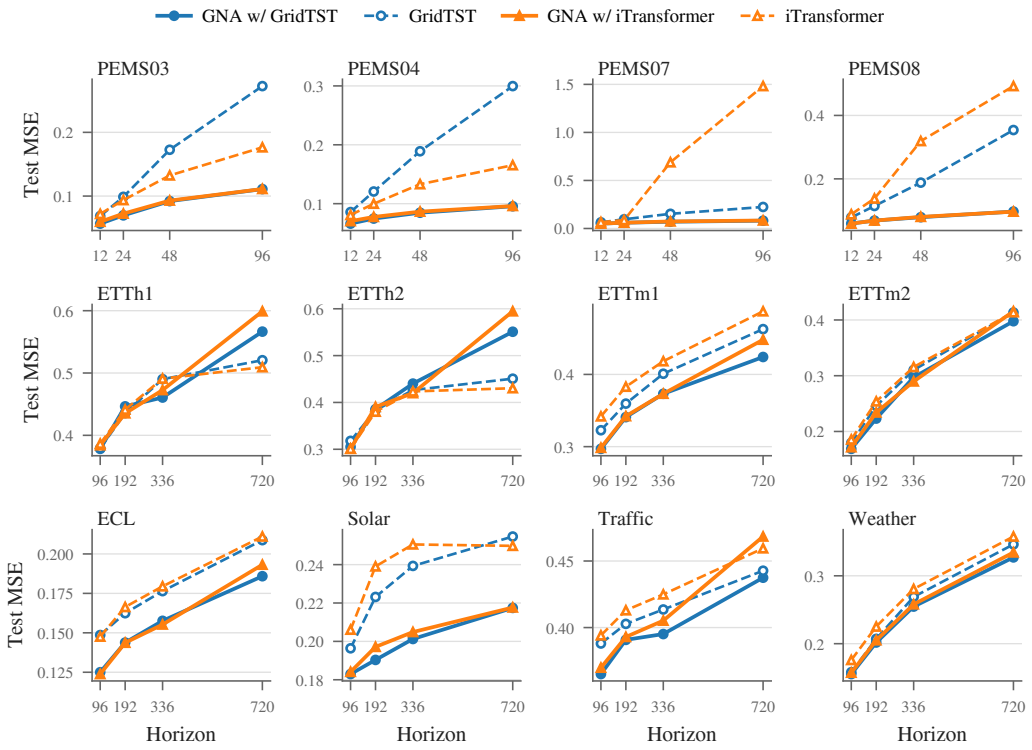


Figure 8: Test MSE per horizon H for every dataset: GNA and the plain backbones.

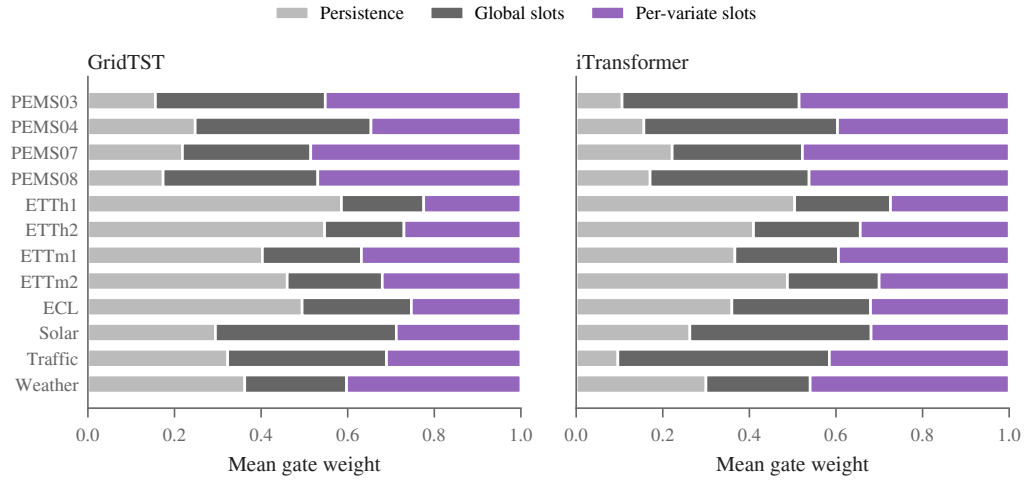


Figure 9: How the gate splits its weight between persistence ($\mathbf{c}_0$), global slots and per-variate slots, averaged over forecast steps, test windows and variates. The backbone term $\alpha f(\mathbf{X}_t)$ is added outside the gate and is not part of this split.

Dataset	Backbone only				GNA (this work)			
	iTransformer ICLR 2024b, our run		GridTST arXiv 2024, our run		iTransformer this work		GridTST this work	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Short-term forecasting								
PEMS03	0.119	0.227	0.153	0.260	0.084	0.183	**0.082**	**0.182**
PEMS04	0.120	0.231	0.174	0.283	0.083	0.184	**0.080**	**0.182**
PEMS07	0.581	0.503	0.134	0.241	0.067	0.156	**0.065**	**0.155**
PEMS08	0.259	0.312	0.184	0.275	**0.076**	**0.176**	**0.076**	**0.176**
Long-term forecasting								
ETTh1	0.457	0.449	**0.456**	**0.448**	0.473	0.464	0.463	0.459
ETTh2	**0.384**	**0.407**	0.395	0.412	0.427	0.441	0.420	0.437
ETTm1	0.408	0.412	0.386	0.402	0.365	0.389	**0.359**	**0.386**
ETTm2	0.292	0.336	0.287	0.332	0.278	0.331	**0.272**	**0.329**
ECL	0.176	0.267	0.174	0.266	0.154	0.255	**0.153**	**0.252**
Solar	0.236	0.262	0.228	0.278	0.201	**0.243**	**0.198**	0.245
Traffic	0.423	0.283	0.412	0.271	0.409	**0.263**	**0.397**	0.269
Weather	0.260	0.280	0.245	0.272	0.239	0.273	**0.235**	**0.270**
Super-short-term forecasting								
TDAQ 2024	38.603	0.766	34.480	**0.671**	34.280	0.739	**34.169**	0.684
TDAQ 2025	53.512	0.619	52.737	0.474	52.777	0.510	**52.593**	**0.462**
1st count	1	1	1	2	1	3	**12**	**9**
2nd count	1	1	3	4	10	6	0	3
Retrieval vs baseline	2	2	2	3	**12**	**12**	**12**	**11**
Per-task								
1st count	4	5	0	7	14	18	**41**	**24**
2nd count	4	4	13	8	29	19	4	19
Retrieval vs baseline	5	8	5	11	**44**	**42**	**45**	**38**

Red, bold: best in row.
Blue, underline: 2nd-best in row.

Table 8: Retrieval vs. no retrieval with the same backbone, data and lookback ($L = 96$; TDAQ: $L = 5$, $H = 1$): test MSE and MAE averaged over the four horizons, for GridTST and iTransformer without retrieval (backbone only) and with retrieval (GNA). All numbers are our runs. Our iTransformer runs are far worse than the published results on PEMS07 and PEMS08, a known reproduction issue; Table 1 uses the published numbers.

Dataset	H	*GNA (this work)* w/ GridTST MSE	*GNA (this work)* w/ GridTST MAE	*GNA (this work)* w/ iTransformer MSE	*GNA (this work)* w/ iTransformer MAE	*Baselines* GridTST arXiv 2024, our run MSE	*Baselines* GridTST arXiv 2024, our run MAE	*Baselines* iTransformer ICLR 2024b, our run MSE	*Baselines* iTransformer ICLR 2024b, our run MAE
PEMS03	12	**0.057**	**0.156**	0.060	0.160	0.069	0.177	0.072	0.177
	24	**0.070**	**0.171**	0.072	0.174	0.099	0.215	0.094	0.203
	48	**0.092**	0.193	0.093	**0.192**	0.173	0.283	0.132	0.243
	96	**0.111**	0.209	**0.111**	**0.207**	0.273	0.366	0.176	0.285
	Avg	**0.082**	**0.182**	0.084	0.183	0.153	0.260	0.119	0.227
PEMS04	12	**0.066**	**0.165**	0.070	0.171	0.086	0.194	0.081	0.188
	24	**0.075**	**0.176**	0.077	0.178	0.121	0.240	0.100	0.212
	48	**0.085**	**0.187**	0.086	0.188	0.189	0.303	0.133	0.246
	96	**0.096**	**0.199**	**0.096**	**0.199**	0.300	0.395	0.165	0.277
	Avg	**0.080**	**0.182**	0.083	0.184	0.174	0.283	0.120	0.231
PEMS07	12	**0.050**	**0.140**	0.051	0.141	0.064	0.164	0.066	0.164
	24	**0.060**	**0.150**	**0.060**	0.152	0.096	0.201	0.087	0.190
	48	**0.070**	**0.161**	0.073	0.162	0.153	0.265	0.689	0.619
	96	**0.081**	**0.168**	0.083	0.169	0.224	0.335	1.482	1.037
	Avg	**0.065**	**0.155**	0.067	0.156	0.134	0.241	0.581	0.503
PEMS08	12	**0.059**	0.157	**0.059**	**0.156**	0.078	0.182	0.088	0.193
	24	**0.068**	**0.169**	**0.068**	**0.169**	0.114	0.223	0.138	0.243
	48	**0.079**	**0.181**	0.080	0.182	0.188	0.290	0.319	0.353
	96	**0.097**	0.198	**0.097**	**0.196**	0.354	0.406	0.492	0.459
	Avg	**0.076**	**0.176**	**0.076**	**0.176**	0.184	0.275	0.259	0.312
ETTh1	96	**0.378**	0.402	0.384	0.404	0.379	**0.401**	0.387	0.405
	192	0.446	0.447	**0.435**	0.438	0.436	**0.433**	0.441	0.436
	336	**0.461**	**0.457**	0.473	0.465	0.490	0.464	0.491	0.462
	720	0.566	0.529	0.599	0.549	0.521	**0.493**	**0.509**	0.494
	Avg	0.463	0.459	0.473	0.464	**0.456**	**0.448**	0.457	0.449
ETTh2	96	0.305	0.358	**0.301**	0.357	0.318	0.356	**0.301**	**0.350**
	192	0.386	0.408	0.390	0.414	0.383	0.401	**0.380**	**0.399**
	336	0.440	0.451	**0.420**	0.444	0.426	0.433	0.424	**0.432**
	720	0.551	0.529	0.595	0.551	0.451	0.458	**0.430**	**0.447**
	Avg	0.420	0.437	0.427	0.441	0.395	0.412	**0.384**	**0.407**
ETTm1	96	**0.297**	**0.341**	0.298	**0.341**	0.323	0.363	0.342	0.377
	192	**0.341**	0.371	0.342	**0.370**	0.360	0.380	0.383	0.396
	336	**0.373**	0.400	**0.373**	**0.398**	0.401	0.412	0.418	0.418
	720	**0.424**	**0.431**	0.448	0.446	0.463	0.453	0.487	0.457
	Avg	**0.359**	**0.386**	0.365	0.389	0.386	0.402	0.408	0.412
ETTm2	96	**0.169**	**0.258**	0.172	0.261	0.179	0.262	0.186	0.272
	192	**0.223**	**0.298**	0.234	0.305	0.246	0.306	0.254	0.314
	336	0.297	0.346	**0.290**	**0.342**	0.310	0.348	0.316	0.351
	720	**0.398**	0.414	0.416	0.415	0.413	0.411	0.414	**0.407**
	Avg	**0.272**	**0.329**	0.278	0.331	0.287	0.332	0.292	0.336
ECL	96	0.125	0.223	**0.124**	**0.222**	0.149	0.240	0.148	0.239
	192	**0.144**	**0.241**	**0.144**	0.242	0.162	0.254	0.166	0.258
	336	0.158	0.257	**0.155**	**0.256**	0.176	0.270	0.179	0.273
	720	**0.186**	**0.286**	0.193	0.298	0.209	0.300	0.211	0.300
	Avg	**0.153**	**0.252**	0.154	0.255	0.174	0.266	0.176	0.267
Solar	96	**0.183**	0.217	0.184	**0.214**	0.196	0.253	0.206	0.236
	192	**0.190**	0.236	0.197	**0.232**	0.223	0.272	0.239	0.263
	336	**0.201**	**0.258**	0.205	0.259	0.239	0.289	0.251	0.274
	720	**0.217**	0.269	0.218	**0.268**	0.255	0.299	0.250	0.275
	Avg	**0.198**	0.245	0.201	**0.243**	0.228	0.278	0.236	0.262
Traffic	96	**0.365**	0.250	0.370	**0.242**	0.388	0.260	0.394	0.270
	192	**0.391**	0.264	0.393	**0.254**	0.403	0.265	0.413	0.277
	336	**0.395**	0.268	0.405	**0.262**	0.414	0.271	0.425	0.283
	720	**0.437**	0.293	0.468	0.296	0.443	**0.288**	0.460	0.301
	Avg	**0.397**	0.269	0.409	**0.263**	0.412	0.271	0.423	0.283
Weather	96	**0.155**	**0.203**	0.157	0.205	0.158	**0.203**	0.176	0.216
	192	**0.202**	0.248	0.205	0.251	0.207	**0.247**	0.225	0.257
	336	**0.255**	**0.290**	0.258	0.291	0.269	0.293	0.281	0.299
	720	**0.327**	**0.341**	0.334	0.344	0.346	0.345	0.358	0.350
	Avg	**0.235**	**0.270**	0.239	0.273	0.245	0.272	0.260	0.280
1st count		**39**	**23**	14	18	0	6	4	5

Table 9: Per-horizon results with a fixed lookback of 96 (H: forecast horizon; *Avg*: mean over the four horizons, as in the main table). Best in red bold, second underlined, per row over the columns shown. 1st count over individual horizons only.

Dataset	*GNA (this work)*				*Backbones*				*Retrieval-based*			
	w/ GridTST		w/ iTransformer		GridTST arXiv 2024, our run		iTransformer ICLR 2024b		GTR ICLR 2026a		RAFT‡ ICML 2025	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
PEMS03	**0.082**	**0.182**	0.084	0.183	0.153	0.260	0.113	0.221	0.087	0.189	—	—
PEMS04	**0.080**	0.182	0.083	0.184	0.174	0.283	0.111	0.221	0.087	0.189	—	—
PEMS07	**0.065**	**0.155**	0.067	0.156	0.134	0.241	0.101	0.204	0.076	0.169	—	—
PEMS08	**0.076**	**0.176**	**0.076**	**0.176**	0.184	0.275	0.150	0.226	0.142	0.222	—	—
ETTh1	0.463	0.459	0.473	0.464	0.456	0.448	0.454	0.447	0.439	0.434	0.428	0.433
ETTh2	0.420	0.437	0.427	0.441	0.395	0.412	0.383	0.407	0.372	0.400	0.382	0.410
ETTm1	**0.359**	0.386	0.365	0.389	0.386	0.402	0.407	0.410	0.367	0.389	0.381	0.400
ETTm2	0.272	0.329	0.278	0.331	0.287	0.332	0.288	0.332	0.268	0.315	0.281	0.330
ECL	**0.153**	0.252	0.154	0.255	0.174	0.266	0.178	0.270	0.166	0.260	0.175	0.272
Solar	0.198	0.245	0.201	0.243	0.228	0.278	0.233	0.262	**0.194**	0.245	0.301	0.303
Traffic	**0.397**	0.269	0.409	**0.263**	0.412	0.271	0.428	0.282	0.470	0.280	0.414	0.284
Weather	**0.235**	0.270	0.239	0.273	0.245	0.272	0.258	0.278	0.239	0.268	0.270	0.309
1st count	**8**	3	1	2	0	0	0	0	1	0	0	0

Dataset	*Other forecasters*													
	TQNet ICML 2025		Amplifier AAAI 2025		FilterTS AAAI 2025a		S-Mamba Neurocomputing 2025b		CycleNet NeurIPS 2024		SOFTS NeurIPS 2024		TimeXer NeurIPS 2024b	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
PEMS03	0.097	0.203	0.131‡	0.239‡	0.134‡	0.246‡	0.122	0.228	0.118‡	0.226‡	0.104	0.210	—	—
PEMS04	0.091	0.197	0.135‡	0.249‡	0.125‡	0.241‡	0.103	0.211	0.119‡	0.232‡	0.102	0.208	—	—
PEMS07	0.075	0.171	0.122‡	0.226‡	0.120‡	0.220‡	0.089	0.188	0.113‡	0.214‡	0.087	0.184	—	—
PEMS08	0.142	0.229	0.183‡	0.271‡	0.180‡	0.266‡	0.148	0.224	0.150‡	0.246‡	0.138	0.219	—	—
ETTh1	0.441	0.434	0.430	0.428	0.433	0.430	0.455	0.450	0.457	0.441	0.449	0.442	0.437	0.437
ETTh2	0.378	0.402	0.359	0.391	0.372	0.396	0.381	0.405	0.388	0.409	0.373	0.400	0.367	0.396
ETTm1	0.377	0.393	0.381	0.394	0.385	0.396	0.398	0.405	0.379	0.396	0.393	0.403	0.382	0.397
ETTm2	0.277	0.323	0.276	0.323	0.276	0.321	0.288	0.332	0.266	0.314	0.287	0.330	0.274	0.322
ECL	0.164	0.259	0.171	0.265	0.180	0.271	0.170	0.265	0.168	0.259	0.174	0.264	0.171	0.270
Solar	0.198	0.256	0.241‡	0.270‡	0.215‡	0.277‡	0.240	0.273	0.210	0.261	0.229	0.256	0.237‡	0.302‡
Traffic	0.445	0.276	0.482	0.315	0.471	0.315	0.414	0.276	0.472	0.301	0.409	0.267	0.466	0.287
Weather	0.242	0.269	0.243	0.271	0.244	0.274	0.251	0.276	0.243	0.271	0.255	0.278	0.241	0.271
1st count	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Dataset	*Other forecasters (cont.)*												*Concurrent preprints*	
	TimeMixer ICLR 2024a		PatchTST‡ ICLR 2023		TimesNet ICLR 2023		Crossformer‡ ICLR 2023		DLinear‡ AAAI 2023		Ister ICASSP 2026b		PAMNet† arXiv 2026 (preprint)	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
PEMS03	0.167‡	0.267‡	0.180	0.291	0.147‡	0.248‡	0.169	0.281	0.278	0.375	0.108	0.217	0.091	0.188
PEMS04	0.185‡	0.287‡	0.195	0.307	0.129‡	0.241‡	0.209	0.314	0.295	0.388	0.106	0.213	0.085	**0.180**
PEMS07	0.181‡	0.271‡	0.211	0.303	0.124‡	0.225‡	0.235	0.315	0.329	0.395	0.092	0.193	0.078	0.163
PEMS08	0.226‡	0.299‡	0.280	0.321	0.193‡	0.271‡	0.268	0.307	0.379	0.416	0.136	0.226	0.137	0.211
ETTh1	0.447	0.440	0.469	0.454	0.458	0.450	0.529	0.522	0.456	0.452	0.438	0.438	**0.420**	**0.424**
ETTh2	0.364	0.395	0.387	0.407	0.414	0.427	0.942	0.684	0.559	0.515	**0.349**	**0.387**	0.374	0.393
ETTm1	0.381	0.395	0.387	0.400	0.400	0.406	0.513	0.496	0.403	0.407	0.386	0.399	0.365	**0.384**
ETTm2	0.275	0.323	0.281	0.326	0.291	0.333	0.757	0.610	0.350	0.401	0.279	0.325	**0.264**	**0.308**
ECL	0.182	0.272	0.205	0.290	0.192	0.295	0.244	0.334	0.212	0.300	0.167	0.260	0.161	**0.251**
Solar	0.216	0.280	0.270	0.307	0.301‡	0.319‡	0.641	0.639	0.330	0.401	—	—	0.204	**0.228**
Traffic	0.484	0.297	0.481	0.304	0.620	0.336	0.550	0.304	0.625	0.383	0.399	0.270	0.440	**0.263**
Weather	0.240	0.271	0.259	0.281	0.259	0.287	0.259	0.315	0.265	0.317	0.243	0.271	0.240	**0.263**
1st count	0	0	0	0	0	0	0	0	0	0	1	1	2	**8**

Table 10: All compared models, lookback 96, horizon-averaged. Baseline numbers are taken from the respective publications (GridTST: our own lookback-96 run). †Concurrent preprint, not peer-reviewed. Best in red bold, second underlined, over all models shown. ‡Lookback-96 result reported by another paper, because the method's own paper uses a different lookback or omits the dataset: RAFT by GTR (Cao et al., 2026a); PatchTST, DLinear, Crossformer, and TimesNet on Solar/PEMS by iTransformer (Liu et al., 2024b); CycleNet on PEMS and TimeXer on Solar by TQNet (Lin et al., 2025); Amplifier and FilterTS on Solar/PEMS and TimeMixer on PEMS by PAMNet (Zhou et al., 2026). CycleNet: CycleNet/MLP variant.

E SEED ROBUSTNESS

We repeat all twelve datasets with seeds 2023 (the tracked runs in all other tables), 2024 and 2025, for GNA and for the plain backbone (Table 11). GNA with GridTST beats the plain backbone in every seed on 10 of 12 datasets and is worse in every seed on ETTh1 and ETTh2. Retrieval also stabilizes training. The seed-to-seed variation of GNA is a median 0.3% of the MSE (at most 0.8%), against a median 4% and up to 20% on PEMS for the plain GridTST. On PEMS, the seed-2023 backbone run used in the main table is the best of its three seeds, so the main-table comparison there is conservative. Relative to seed noise, all ablation effects in Fig. 4 exceed twice the full model's seed standard deviation except Weather without global slots; three effects are borderline.

Dataset	GridTST	GNA w/ GridTST	GNA w/ iTransformer	Reduction	Seeds won
PEMS03	0.191 ± 0.0331	0.082 ± 0.0001	0.084 ± 0.0001	$+56.9\%$	3/3
PEMS04	0.225 ± 0.0445	0.080 ± 0.0001	0.082 ± 0.0001	$+64.4\%$	3/3
PEMS07	0.166 ± 0.0281	0.065 ± 0.0000	0.067 ± 0.0003	$+60.7\%$	3/3
PEMS08	0.231 ± 0.0413	0.076 ± 0.0002	0.076 ± 0.0003	$+67.3\%$	3/3
ETTh1	0.455 ± 0.0028	0.460 ± 0.0027	0.471 ± 0.0039	-1.1%	0/3
ETTh2	0.389 ± 0.0054	0.417 ± 0.0034	0.423 ± 0.0085	-7.2%	0/3
ETTm1	0.389 ± 0.0026	0.361 ± 0.0022	0.368 ± 0.0026	$+7.2\%$	3/3
ETTm2	0.286 ± 0.0010	0.273 ± 0.0022	0.280 ± 0.0028	$+4.7\%$	3/3
ECL	0.189 ± 0.0127	0.153 ± 0.0001	0.153 ± 0.0008	$+18.9\%$	3/3
Solar	0.251 ± 0.0197	0.198 ± 0.0006	0.201 ± 0.0005	$+21.0\%$	3/3
Traffic	0.414 ± 0.0020	0.397 ± 0.0007	0.410 ± 0.0005	$+4.1\%$	3/3
Weather	0.248 ± 0.0027	0.236 ± 0.0012	0.239 ± 0.0003	$+5.0\%$	3/3

Table 11: Test MSE, mean $\pm$ standard deviation over three seeds (2023, 2024, 2025), each seed first averaged over the four horizons. Reduction: MSE of GNA with GridTST relative to the plain backbone (seed means). Seeds won: seeds in which GNA is better.

F ABLATION DETAILS AND QUALITATIVE EXAMPLE

All ablation arms use the full GNA recipe with GridTST (seed 2023, $L = 96$, 10 epochs) and change exactly one thing; the control is the tracked full model. Each arm and what it tests:

- *w/o relevance (mismatched)*: each query keeps its own window but receives the complete neighbor set (slots, slot types, masks, calendar match) of a different query. At validation and test time, that query is drawn uniformly from the training split, so all neighbors stay strictly past; during training, it lies at least $L + H$ windows away. Architecture, parameter count and training recipe are unchanged, and the swap is applied identically in training, validation and test, so only the relevance of the neighbors changes.
- *w/o retrieval*: the plain backbone.
- *w/o per-variate slots*: ten global slots, i.e. the global half of the full model doubled.
- *w/o global slots*: ten per-variate slots from the spaced pool.
- *w/o future-aware embedding*: the embedding ranking is replaced by the cosine similarity of the standardized raw lookback windows; fusion, eligibility, spacing and assembly are unchanged.
- *w/o rank fusion*: ranking by the learned embedding alone, without regime, seasonal and calendar terms.

All ablation banks were audited for causality. Table 13 lists the numbers. Fig. 10 shows a typical forecast.

Retrieved-future quality. The ablations above measure forecasts. Table 12 measures the retrieved futures themselves, before any model sees them, by comparing each slot's future with the query's true future. The learned ranking returns slot futures whose average is 0.1–4.0% closer to the truth than with raw-window ranking; the best single slot changes little. Per-variate slots contain a much closer candidate for every variate: the best per-variate slot is 17–45% closer than the best global slot

on every dataset and at every horizon. Their plain average is not always better (PEMS08: $+11\%$), so the gain depends on choosing among them, which is the role of the gate.

	Learned vs. raw-window ranking		Per-variate vs. global slots	
Dataset	Mean of slots	Best slot	Mean of slots	Best slot
PEMS04	-4.0	-0.2	-4.4	-17.2
PEMS08	-1.3	$+0.7$	$+11.0$	-24.2
ETTm1	-0.1	$+0.7$	-15.8	-24.4
ETTm2	-3.4	-0.7	-13.8	-26.7
Weather	-2.6	-1.4	-25.8	-45.2

Table 12: Quality of the retrieved futures, without any forecasting model: % change of the MSE between retrieved slot futures and the query's true future on the test set (standardized scale, mean over the four horizons; negative = closer to the truth). Mean of slots: the average of the K slot futures; best slot: per variate, the closest of the K slot futures in hindsight. Left: the full bank vs. the same bank ranked by raw-window similarity. Right: ten per-variate slots vs. ten global slots (the ablation banks).

	PEMS04	PEMS08	ETTm1	ETTm2	Weather
Full model, MSE	0.080	0.076	0.359	0.272	0.235
Full model, MAE	0.182	0.176	0.386	0.329	0.270
MSE increase vs. full model (%)					
w/o relevance (mismatched neighbors)	+201.5	+246.8	+10.3	+21.1	+5.9
w/o per-variate slots	+3.5	+8.3	+4.8	+1.9	+5.5
w/o global slots	+13.0	+15.8	+2.5	+3.4	+0.3
w/o future-aware embedding	+7.4	+5.1	+1.1	+3.9	+0.9
w/o rank fusion (embedding only)	-1.1	+0.4	+4.1	+12.0	+4.5
w/o retrieval (backbone only)	+116.5	+142.1	+7.7	+5.7	+4.4
MAE increase vs. full model (%)					
w/o relevance (mismatched neighbors)	+86.4	+96.0	+6.4	+11.1	+5.9
w/o per-variate slots	+2.5	+5.0	+2.6	+0.8	+3.0
w/o global slots	+7.3	+8.4	+1.9	+1.2	+0.5
w/o future-aware embedding	+3.8	+0.9	+0.3	+0.6	+0.1
w/o rank fusion (embedding only)	-0.7	+1.4	+2.6	+5.6	+3.1
w/o retrieval (backbone only)	+55.8	+56.3	+4.2	+0.8	+0.6

Table 13: Ablations with the GridTST backbone, horizon-averaged, single seed. Positive: worse without the component.

Slots per query K (GNA)				Lookback L (ETTm1)			
Dataset	$K = 5$	$K = 10$	$K = 20$	L	GridTST	GNA	Change
PEMS08	0.101	0.097	0.096	48	0.466	0.310	-33.4%
ETTm1	0.303	0.297	0.295	96	0.323	0.297	-8.0%
				192	0.301	0.297	-1.2%
				336	0.288	0.303	$+5.3\%$

Table 14: Sensitivity at $H = 96$ (test MSE, GridTST backbone, seed 2023). Left: number of slots per query, half global and half per-variate ($K = 10$: reported runs). Right: lookback L for the plain GridTST and GNA, each with its own embedder and bank per L; change: GNA relative to GridTST.

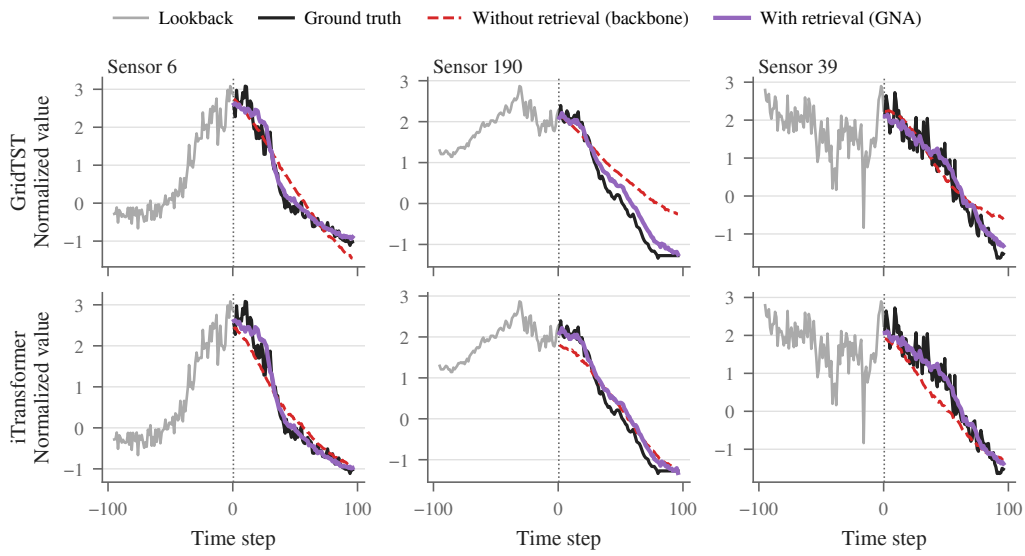


Figure 10: A typical PEMS04 test window ($H = 96$): ground truth and the forecasts of GNA and of the plain backbone, for both backbones. The window and variate were chosen by a fixed rule (median gain of GNA over the backbone; variate with the largest target spread), not by appearance.

G Computational cost

All experiments run on one NVIDIA RTX PRO 6000 Blackwell GPU (96 GB) with an AMD Ryzen Threadripper PRO 7955WX CPU (16 cores) and 124 GB RAM, using Python 3.13, PyTorch 2.12 (CUDA 13.0) and NumPy 2.4. Table 15 reports resource use at $L = H = 96$, with one stage at a time on an otherwise idle GPU. The bank is built once before training; the index holds what retrieval needs (embeddings, rankings, regime features). Our implementation also caches every query's assembled slots for fast training. This cache is about 100 times larger than the index and could be replaced by assembling the slots on the fly. Table 16 breaks down the latency of one online forecast. Retrieval takes 19–25 ms per query on the CPU and is dominated by the exact search, which approximate search would make sublinear in N.

		Retrieval bank (built once)				Train		GPU energy		GPU mem.	
		wall	CPU	index	cache	(s/epoch)		(Wh/run)		(GB)	
Dataset	D	(min)	(min)	(GB)	(GB)	w/o	with	w/o	with	w/o	with
ETTm1	7	5.2	12.4	0.15	2.0	10	15	3	5	0.8	0.8
Weather	21	5.5	12.2	0.19	5.0	13	19	6	9	1.0	1.1
PEMS08	170	3.3	5.2	0.17	12.8	14	17	10	14	4.1	4.8
ECL	321	7.0	10.4	0.40	36.3	78	89	98	111	15.1	19.2

Table 15: Resource use at $L = 96$ (GridTST backbone, batch size 32) on an otherwise idle GPU: bank construction (wall-clock and CPU time, index and training-cache size), training time per epoch, GPU energy per training run and GPU memory. w/o: the same model with retrieval disabled.

			Retrieval stages (ms)				Retrieval (ms)		Forecast
Dataset	D	N	Embed	Search	Fusion	Assembly	median	p90	(ms)†
ETTm1	7	57,601	0.7	18.9	0.6	0.9	21.2	26.7	$\leq$0.6
Weather	21	52,697	0.7	22.5	0.6	1.2	25.2	41.3	$\leq$0.7
PEMS08	170	17,665	2.4	11.7	0.4	4.4	19.1	23.1	$\leq$2.1
ECL	321	26,305	2.2	12.3	0.6	8.0	23.3	29.0	$\leq$3.2

Table 16: Latency of one forecast by stage. Retrieval runs on the CPU for one query at a time (8 threads; median and 90th percentile over 200 test queries after 20 warm-up queries): embedding the query with ϕ, exact search over the eligible bank, rank fusion, and slot assembly. D: variates; N: bank windows. †Full test-set evaluation time per window on the GPU (batch size 32, including data loading), an upper bound on the forward pass.

H Comparison with retrieval-based forecasting methods

Table 17 compares GNA with retrieval-based forecasting methods, based on each paper's description. Among these methods, only GNA trains its retrieval embedding to predict each window's own future, combines whole-window and per-variate neighbors in one multivariate model, and uses a test-time bank that grows causally. TimeRAF also learns its retriever, but from the forecaster's error on retrieved candidates; TS-RAG, RAF and TimeRAF retrieve univariate segments, i.e. for each variate on its own.

Method	Retrieved	Similarity	Assembly	How it enters the model	Bank at test time
RAFT (Han et al., 2025)	(history, future) patches of the series	Pearson correlation of offset-removed raw windows, at several downsampling periods	top-m windows, softmax-weighted sum	projected, combined with an MLP forecaster	training data
GTR (Cao et al., 2026a)	segment of a learned global-cycle embedding (no past windows)	absolute position within a fixed global cycle	one aligned segment	2D convolution, residual into the backbone	learned parameters
RATD (Liu et al., 2024a)	(history, future) windows	distance between pre-trained encoder embeddings of the histories	k nearest whole windows	guide a diffusion denoiser (reference-modulated attention)	training data
TS-RAG (Ning et al., 2025)	(context, future) pairs, univariate	Euclidean distance of frozen foundation-model encoder embeddings	top-k futures	adaptive retrieval mixer on a frozen foundation model	separate knowledge base (zero-shot)
TimeRAF (Zhang et al., 2024)	univariate segments of input length (multi-domain knowledge base)	learned retriever, trained so that candidates that lower the foundation model's forecast error score higher	top-8, averaged	channel prompting (MLP, residual) on a foundation model	separate knowledge base (zero-shot)
RAF (Tire et al., 2026)	(context, future) segments, univariate	nearest neighbour on foundation-model embeddings or raw values	best match	prepended to the input context of a foundation model	dataset-specific database
GNA (this work)	(history, future) windows	learned embedding trained to predict the future, fused with regime, seasonal and calendar ranks	global slots + per-variate re-ranked slots	learned gate over persistence and futures, backbone term, cross-attention	strictly causal, grows with the test period

Table 17: Retrieval-based forecasting methods compared by what they retrieve, how they measure similarity, how they assemble and use the retrieved information, and which data their bank contains at test time.

I Oracle-ceiling details

To separate the quality of the retrieved candidates from the model's ability to use them, we use oracles that choose past windows with hindsight, using the true future, from the same causally eligible bank. Choosing among thousands of windows with the target includes selection luck, so

absolute oracle numbers are optimistic. The comparison between raw and level-free futures is fair, since both use the same windows. Figure 11 shows the headroom between the retrieved slots and the best past windows. Figures 12 and 13 relate retrieval quality and gate trust to the forecast gain across all datasets, horizons and backbones. For ETTh (Table 18, Fig. 14), the analysis below uses full test-set predictions of the tracked runs.

ETTh diagnosis. On ETTh1 and ETTh2, GNA is worse than GridTST. The loss is a far-horizon effect (Fig. 14). At $H = 720$ on ETTh1, GNA is 5.5% better than GridTST over steps 1–96 and 16.5% worse over steps 337–720, which carry all of the excess error. ETTh2 is worse at nearly every step, with 60% of the gate weight on neighbors (32% on ETTh1). The loss grows with the hindsight level offset between the gate-weighted neighbor futures and the truth (Spearman 0.25 and 0.32), but not with the level gap at the forecast origin, which the gate can observe: retrieved futures look aligned at the start and drift further out. A hindsight oracle over the same eligible windows supports this reading (Table 18): the mean of the ten best raw past futures barely beats GridTST at $H = 720$ (-5.8%, -0.9%), whereas the same windows with level-free futures, shifted to start at the query's last value, would reduce the error by 26% and 36%. A fixed shift hurts ECL and Traffic, so a learned level correction is the natural next step.

Test MSE	ETTh1/720	ETTh2/720
GridTST (no retrieval)	0.521	0.451
GNA (this work)	0.566	0.551
Best of the K retrieved slots	0.712	0.535
Best past window(s), raw futures		
Best single	0.640	0.519
Mean of 10 best	0.491	0.447
Best past window(s), level-free futures		
Best single	0.596	0.380
Mean of 10 best	0.386	0.289

Table 18: Oracle retrieval ceiling on ETTh at $H = 720$ (diagnostic; oracle rows choose windows with the true future from the same causally eligible bank). Level-free futures are shifted to start at the query's last observed value.

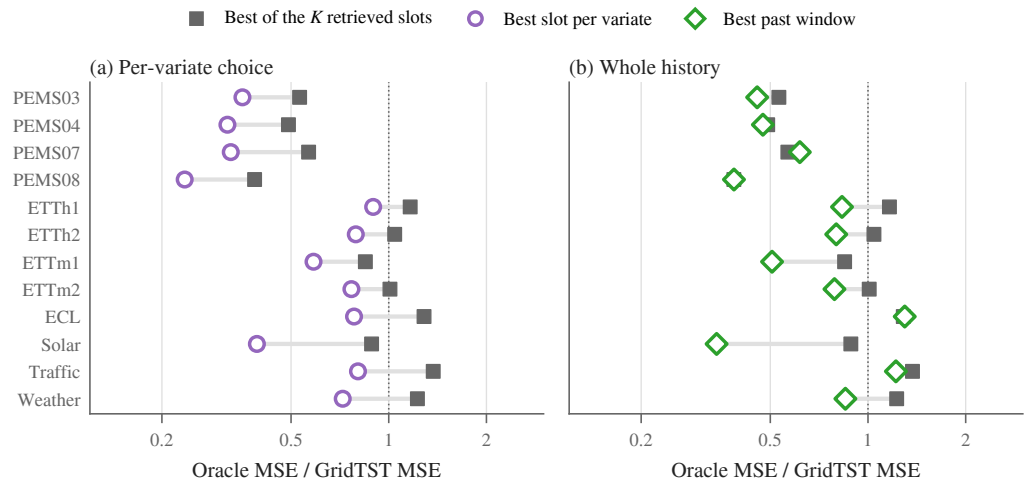


Figure 11: Retrieval headroom at $H = 96$, relative to GridTST, with oracle choices that use the true future: (a) best of the K retrieved slots vs. the best slot chosen per variate; (b) vs. the best single past window.

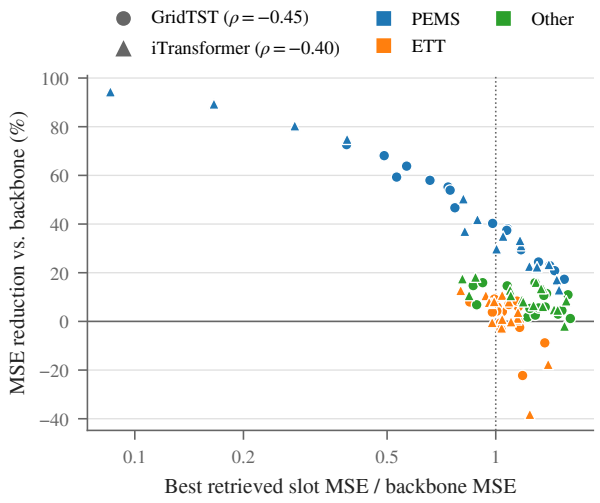


Figure 12: Retrieval quality vs. forecast gain, one point per dataset, horizon and backbone; ρ: Spearman correlation.

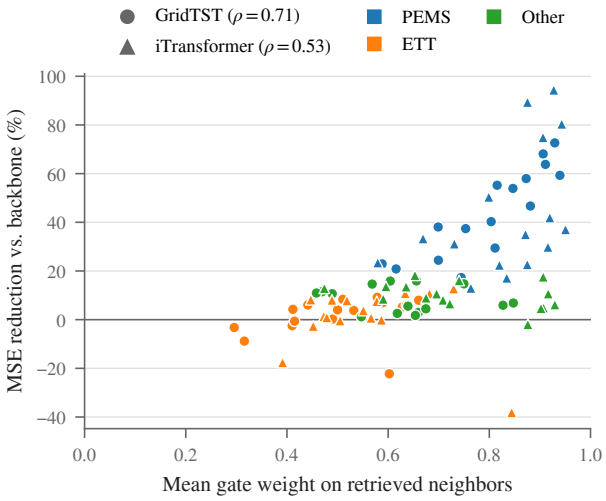


Figure 13: Learned trust vs. forecast gain, one point per dataset, horizon and backbone. x: mean gate weight on retrieved slots (the rest goes to persistence $\mathbf{c}_0$; the backbone term is added outside the gate); y: MSE reduction relative to the plain backbone; ρ: Spearman correlation.

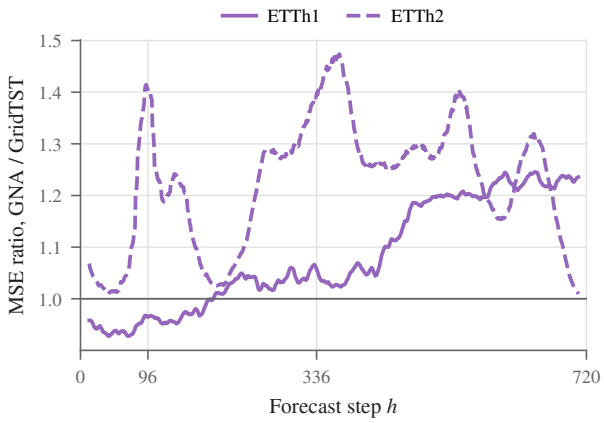


Figure 14: ETTh at $H = 720$: ratio of the test MSE of GNA to that of GridTST per forecast step (24-step moving average). Below 1: retrieval helps.

J TDAQ CASE STUDY DETAILS

The pile-up $\langle\mu\rangle$ is the mean number of inelastic proton–proton interactions per bunch crossing, which is proportional to the per-bunch luminosity. Within a run it typically rises when stable beams start, stays on a plateau while the luminosity is levelled, and declines as the beam intensity decreases (Fig. 16). We call windows on the plateau of their run *steady* (73% of test queries) and the others *transient* (ramp-up and decline, 27%). Figures 16 and 17 show that the retrieval embedding separates these regimes and that 89–98% of the retrieved neighbors come from the query's own regime, often thousands of windows back. Figure 15 uses full test-set predictions of three seed pairs (36 test runs); the run-level bootstrap resamples runs.

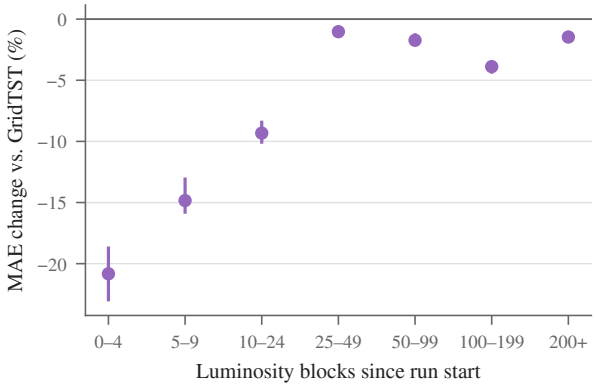


Figure 15: TDAQ ($L = 5$, $H = 1$, GridTST backbone): MAE change of GNA relative to GridTST by the position of the forecast origin within its data-taking run, in luminosity blocks since the run started; with $L = 5$, the first bin holds only the first forecast of each run. Markers: mean over three seeds; bars: range over the seeds. Negative values mean that GNA is better.

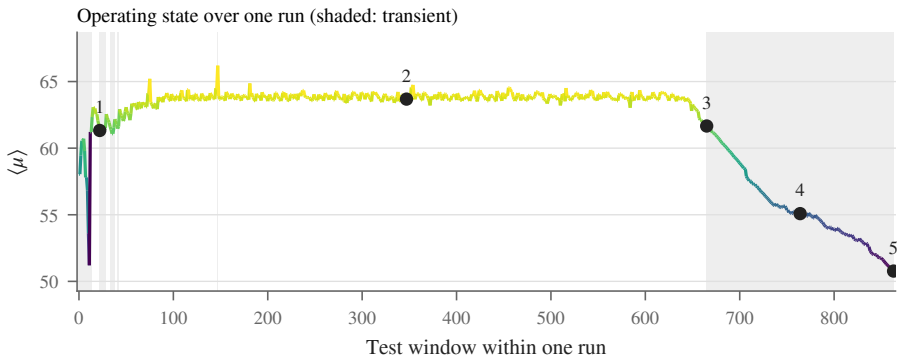


Figure 16: $\langle\mu\rangle$ over one TDAQ test run (shaded: transient regime), with the numbered query steps used in Fig. 17. Line color encodes $\langle\mu\rangle$ on the same scale as Fig. 17.

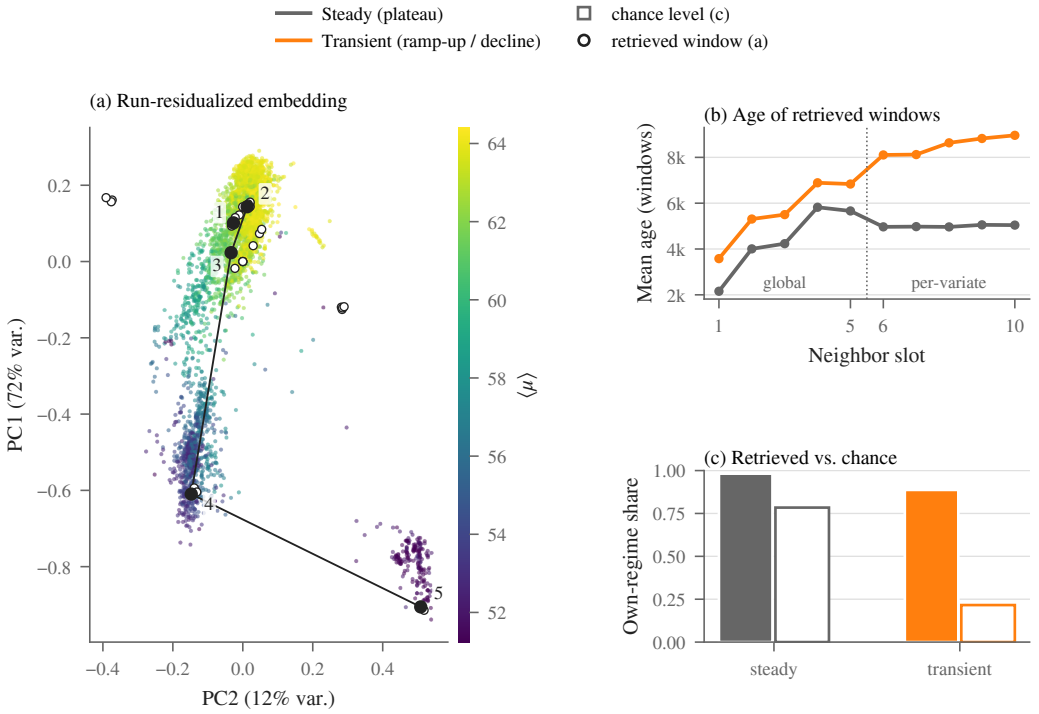


Figure 17: TDAQ operating regimes and retrieval: (a) run-residualized embedding colored by $\langle\mu\rangle$, with the query steps of Fig. 16 and their retrieved windows; (b) age of the retrieved windows per slot and regime; (c) share of neighbors from the query's own regime, against the share expected by chance.

K FORECASTING-BASED ANOMALY DETECTION

Forecasters are often used as the basis of anomaly detectors, with the forecast residual as the anomaly score (Hundman et al., 2018). We test whether the forecasting gain of GNA carries over to detection on TDAQ. We add anomalies of three shapes (level shifts, linear ramps starting from zero, and Gaussian-shaped bumps) to one randomly chosen variate with a random sign, with lengths 5, 20 and 100 and six intensities from 0.5 to 8 interquartile ranges, and ten random repeats of the injection positions (540 scenarios), and score each point by the one-step forecast residual with the corrupted context. We report pointwise average precision without point adjustment, together with the trade-off between false alerts and time to first alert (Fig. 18). GNA reaches a pointwise AP of 0.086, against 0.015 for iTransformer and 0.026 for an LSTM, and has a higher AP than each of the two baselines in 81% of the scenarios. At the same pointwise false-positive rate of 1%, it raises its first alert after 23.8 luminosity blocks, against 35.4 and 40.1 (Fig. 18c shows the full trade-off). This benchmark uses an earlier version of the retrieval model and of the TDAQ preprocessing than the rest of the paper, so its error scale differs from Table 8.

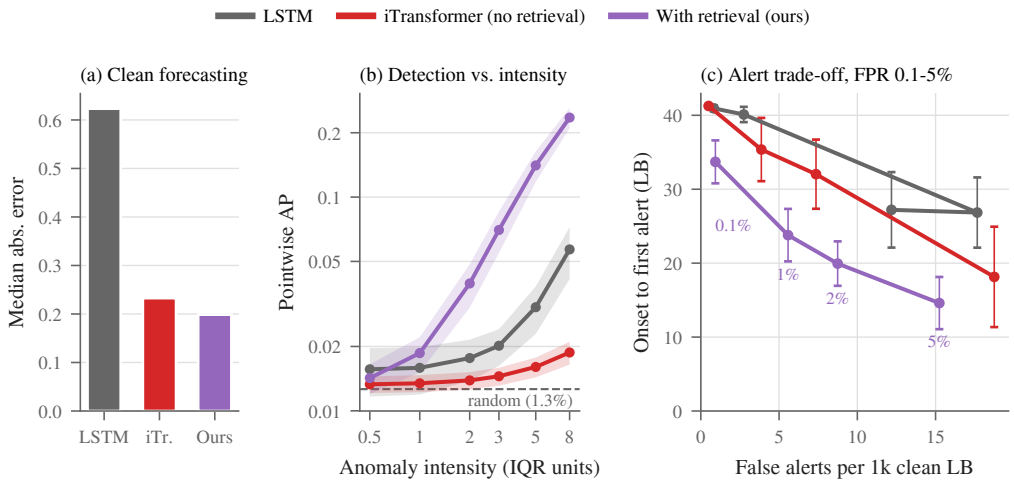


Figure 18: Forecasting-based anomaly detection on TDAQ: (a) clean forecast error; (b) pointwise average precision vs. anomaly intensity; (c) false alerts vs. time to first alert over operating points.