BEYOND CANDIDATE VOTING: TRANSFORMER AGGREGATION MEETS OBSERVATIONAL EVIDENCE

Anonymous authors
Paper under double-blind review

ABSTRACT

Causal discovery algorithms often return different graphs because they rely on different functional, distributional, and conditional-independence assumptions. Candidate-only ensembles can reduce this instability, but their predictions remain limited to information retained in the submitted graphs. We introduce CAFUFO – a [CA]usal [FU]sion trans[FO]rmer, a discovery framework comprising progressively richer variants – globally learned candidate weighting, edge-adaptive transformer aggregation, candidate-anchored observational correction, an independent data-driven expert, and a full hybrid that routes between candidate-anchored and data-driven expert. Our primary candidate pool contains PC, GES-BIC, ICA-LiNGAM, NOTEARS, and DAGMA. On 90 locked tasks, the full hybrid model obtains a directed F1 of 0.661, compared with 0.612 for a shared-budget adaptation of Causal-Bayes-Ensemble (CBE) (Vo et al. (2026)). The framework also exposes complementary operating points: uniform candidate voting provides the strongest skeleton recovery and normalized structural Hamming distance, while our global-weighting variant provides the best calibration. On continuous structural-equation data generated over unseen Asia, CHILD, and ALARM graph topologies, the full hybrid achieves the highest macro directed F1. On Sachs, our global candidate-weighting variant ties matched full-data PC at 0.439 directed F1, while the full hybrid matches the CBE adaptation. Together, these results demonstrate CAFUFO as a unified framework for combining heterogeneous causal graph candidates with observational evidence, improving directed recovery beyond candidate-only Bayesian voting while offering complementary variants that balance accuracy, calibration, and robustness under distribution shift (Code: https://anonymous.4open.science/r/CFF-1852/README.md).

1 INTRODUCTION

Causal discovery seeks to recover causal structure from observational data. Despite substantial methodological progress, different causal discovery algorithms often return markedly different graphs when applied to the same dataset. This disagreement is not merely an optimization artefact. Constraint-based methods such as PC rely on conditional-independence tests and faithfulness assumptions (Spirtes et al., 2000); score-based methods such as GES search over equivalence classes using a chosen model score (Chickering, 2002); ICA-LiNGAM exploits linearity and non-Gaussian disturbances (Shimizu et al., 2006); and continuous-optimization methods such as NOTEARS and DAGMA impose differentiable characterizations of acyclicity (Zheng et al., 2018; Bello et al., 2022). Because these assumptions are difficult to verify from observational data alone, therefore, selecting a single algorithm before observing the true graph is inherently risky.

Causal discovery ensembles address this instability by combining multiple candidate structures. Simple approaches use majority voting or fixed weighted combinations, while recent work develops more principled competence-aware aggregation. In particular, Causal-Bayes-Ensemble (CBE) learns algorithm-specific edge-state competence and prior information before applying Bayesian or voting-based aggregation (Vo et al., 2026). This provides a principled way to combine heterogeneous causal discovery algorithms. However, two opportunities remain. *First, an algorithm's reliability need not be constant across edges*: PC may be reliable for one adjacency, GES-BIC for another orientation, and the meaning of either prediction may depend on whether the remaining candidates agree or conflict. A transformer provides a natural mechanism for modelling these interactions, treating the candidates

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.

for each ordered node pair as a set and learning edge-specific reliability weights from their identities, predictions, and patterns of agreement. *Second, CBE and related candidate-only approaches operate entirely on the generated graphs*. Once those graphs have been produced, the original observations no longer directly influence the aggregation decision. If every candidate omits the same edge or agrees on an incorrect orientation, candidate-only aggregation has no direct observational evidence with which to reconsider that consensus. These limitations motivate our central question: **Can edge-adaptive transformer fusion of heterogeneous candidate graphs, augmented with direct observational evidence, improve causal structure recovery beyond candidate-only voting?** In this work, we introduce CAFUFO, a framework that organizes several increasingly expressive approaches to causal graph aggregation. First, the simplest learned variant assigns one global weight to each causal discovery algorithm. Second, a transformer-based variant replaces these fixed candidate-specific weights with edge-specific estimates, allowing the contributions of PC, GES-BIC, ICA-LiNGAM, NOTEARS, and DAGMA to vary across node pairs. Third, a candidate-anchor variant augments the transformer-weighted candidate prediction with a *bounded observation-derived correction*, thereby retaining the candidate ensemble as its reference point. Fourth, an independent data-only expert estimates edge probabilities from observation-derived features without access to candidate edge states. Finally, the complete hybrid uses an observable, regret-aware router to adaptively combine the candidate-anchor and data-only predictions (framework variants summarized in Appendix A).

Contributions. Our contributions are threefold: *(i)* We introduce an edge-adaptive transformer fusion framework for heterogeneous causal graph candidates. For each ordered node pair, self-attention models agreement and conflict across algorithms and resampled runs, allowing candidate reliability to vary across edges. **(ii)** We complement candidate-based prediction with direct observational evidence through two mechanisms: a *bounded observation-derived correction* to the candidate-aggregated prediction, and a separate *data-only expert* that predicts directly from the current observation. These two components are subsequently combined within a *hybrid strategy* that balances candidate-based and observation-driven predictions. *(iii)* We conduct an extensive evaluation comprising a one-shot assessment on 90 previously untouched locked tasks, transfer experiments using continuous structural-equation data generated over unseen Asia, CHILD, and ALARM topologies, and evaluation on the Sachs observational benchmark. Together, these contributions establish edge-adaptive transformer aggregation with observational correction as a flexible extension of candidate-only causal discovery ensembles, while retaining simpler framework variants for settings in which calibration or transfer characteristics are more important than maximum directed-edge recovery.

2 RELATED WORK

Prior ensemble methods combine graphs obtained from multiple algorithms, resampled datasets, or optimization runs using majority voting, stability selection, graph-distance objectives, or constrained consensus optimization. E.g., Aslani and Mohebbi (2023) formulate heterogeneous DAG aggregation as an optimization problem that preserves information contributed by the input graphs. Such methods can improve stability, but their aggregation rules are generally fixed globally and operate on the generated graph structures. The closest work is Causal-Bayes-Ensemble (CBE) (Vo et al., 2026), which develops a voting-theoretic account of causal graph aggregation and models the competence and diversity of causal discovery experts. CBE combines candidate edge states using learned Bayesian or voting-based aggregation. The distinction between CBE and CAFUFO lies in both their aggregation mechanisms and the information available during aggregation. CBE is candidate-only. In contrast, CAFUFO uses transformer to model agreement and conflict among the candidate graphs separately for each ordered node pair, with observation-derived pairwise features informing the resulting edge-specific reliability weights.

Amortized causal discovery learns a mapping from datasets to causal structures across a distribution of training tasks rather than fitting a new discovery model independently for every dataset (Löwe et al., 2022). Transformer-based models such as CSIvA directly encode observational or interventional samples and autoregressively predict graph structure (Ke et al., 2023); subsequent work has examined how such models transfer from synthetic training distributions to unseen observational settings (Montagna et al., 2025). These approaches use neural networks primarily as direct graph predictors. Our transformer instead operates at the aggregation level: its tokens represent the competing directional predictions of constituent algorithms, together with their metadata and observation-derived pair features. AVICI amortizes causal structure learning by training a variational model on simulator-

generated datasets to predict edge probabilities directly from previously unseen observational or interventional data, avoiding a separate graph search for each dataset (Lorch et al., 2022). DiBS instead performs differentiable Bayesian structure learning through a continuous latent representation of DAGs (Lorch et al., 2021). Unlike CAFUFO, both methods infer structure directly from observations rather than fusing a heterogeneous pool of candidate graphs, making them complementary data-driven comparators.

3 METHOD

Let $\mathbf{X} = \left[\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}\right]^\top \in \mathbb{R}^{n \times d}$ contain n observations of d continuous variables generated according to an unknown directed acyclic graph (DAG) $G^\star$. Its adjacency matrix is denoted by $\mathbf{A}^\star \in \{0,1\}^{d \times d}$, where $A^\star_{ij} = 1$ indicates the directed edge $X_i \to X_j$. Given $\mathbf{X}$, our objective is to estimate a directed adjacency matrix $\widehat{\mathbf{A}}$ that recovers the structure of $G^\star$. Instead of relying on a single causal discovery algorithm, we apply M constituent algorithms with different assumptions and inductive biases. Each algorithm is evaluated once on the complete dataset and $B-1$ times on bootstrap-resampled datasets. This produces the heterogeneous candidate ensemble: $\mathcal{C} = \{\widehat{\mathbf{A}}^{(m,b)} : m \in \{1, \ldots, M\},\ b \in \{0, \ldots, B-1\}\}$, where $b = 0$ denotes the full-data estimate. In our primary configuration, $M = 5$ and $B = 3$, yielding 15 candidate graphs ($K = MB$). The five constituent algorithms are PC, GES-BIC, ICA-LiNGAM, NOTEARS, and DAGMA.

Common Candidate Representation The constituent algorithms do not all return the same type of graph. Constraint-based methods such as PC can produce unoriented adjacencies, score-based methods such as GES-BIC can return partially directed structures, and continuous-optimization methods such as NOTEARS and DAGMA produce weighted directed edges. We, therefore, convert all outputs into a common directional-support representation. For candidate $k = (m, b)$ and ordered pair (i, j), let $q^{(k)}_{ij} \in [0, 1]$ denote its support for $X_i \to X_j$. For every candidate, the model receives the pair $(q^{(k)}_{ij}, q^{(k)}_{ji})$, containing its support for the proposed direction $i \to j$ and the competing direction $j \to i$. For PC and GES-BIC, which can return unoriented adjacencies – no adjacency, $i \to j$, $j \to i$, and $i - j$ are encoded as $(0, 0)$, $(1, 0)$, $(0, 1)$, and $(0.5, 0.5)$, respectively. For ICA-LiNGAM, NOTEARS, and DAGMA, the pair contains the normalized directed edge strengths in the two directions. Each candidate is additionally described by an algorithm-family indicator: $\mathbf{e}^{(k)}_{\text{alg}}$, its predicted graph density: ρ_k, and its distinct-observation fraction: f_k. The resulting representation is $\mathbf{c}^{(k)}_{ij} = [q^{(k)}_{ij}, q^{(k)}_{ji}, \mathbf{e}^{(k)}_{\text{alg}}, \rho_k, f_k]$. It allows the transformer to compare forward and reverse evidence while accounting for algorithm identity, graph sparsity, and resampling diversity. Exact definitions and examples are provided in Appendix B.

Observation-Derived Features At inference time, all observation-derived features are computed solely from the observed data $\mathbf{X}$. For each ordered node pair (i, j), we compute $\boldsymbol{\phi}_{ij} = \Phi(X_i, X_j; \mathbf{X}) \in \mathbb{R}^8$, summarizing symmetric and directional associations between X_i and X_j. For each node i, we additionally compute $\boldsymbol{\psi}_i = \Psi(X_i; \mathbf{X}) \in \mathbb{R}^6$, containing normalized marginal distributional statistics. These features enter the four CAFUFO variants differently. The two candidate-based variants take the candidate representations together with $\boldsymbol{\phi}_{ij}$. The data-only expert instead predicts edge probabilities directly from $\boldsymbol{\phi}_{ij}$, without access to candidate representations. Finally, the hybrid-routing variant operates on a task-level descriptor: $\mathbf{u} = U(\mathbf{X}, \mathcal{C}) \in \mathbb{R}^{38}$, constructed from summaries of $\boldsymbol{\phi}_{ij}$ and $\boldsymbol{\psi}_i$, candidate support and disagreement, predictions from the candidate-anchor and data-only experts, and problem-size descriptors. Complete feature definitions, normalization rules, and all 38 router coordinates are provided in Appendix C.

Neural Architecture Details In our design, f_{pair}, f_{token}, f_{rel}, f_{corr}, and f_{route} denote learned neural maps implemented as shallow feed-forward networks, while f_{data} denotes the complete data-only expert. Full architectural specifications are provided in Appendix D.

3.1 GLOBAL CANDIDATE WEIGHTING

Our simplest learned variant assigns one global reliability weight to each candidate slot. Let $\gamma_k \geq 0$, with $\sum_{k=1}^{K} \gamma_k = 1$, denote the learned weight of candidate $k = (m, b)$, where m identifies the constituent algorithm and b identifies its full-data or bootstrap member. The predicted probability

of $X_i \to X_j$ is: $p_{ij}^{\text{global}} = \sum_{k=1}^{K} \gamma_k q_{ij}^{(k)}$. For the transformer-based variants introduced below, these slot-level weights are aggregated into an algorithm-family prior: $\alpha_m = \sum_{k:\, m(k)=m} \gamma_k$, where $m(k)$ denotes the algorithm that generated candidate k. Because the candidate slots partition into algorithm families, $\sum_m \alpha_m = 1$. This model is deliberately simple: it can learn that one candidate slot is generally more reliable than another, but it cannot change its weighting across edges. We do not present it as a new ensemble principle in isolation. Within CAFUFO, it serves as a lightweight candidate-only variant, supplies the family-level prior used by the richer transformer variants, and provides an important control for determining whether edge-adaptive aggregation is necessary.

3.2 EDGE-ADAPTIVE TRANSFORMER AGGREGATION

A single global weight cannot capture that the same constituent algorithm may be reliable for one edge but unreliable for another. *We, therefore, use a transformer to estimate candidate reliability separately for every ordered node pair.* For an ordered pair (i, j), the eight-dimensional observation-derived feature vector $\boldsymbol{\phi}_{ij}$ is first mapped to an embedding: $\mathbf{z}_{ij} = f_{\text{pair}}\left(\boldsymbol{\phi}_{ij}\right)$. For candidate k, we then construct the transformer input: $\mathbf{x}_{ij}^{(k)} = \left[\mathbf{z}_{ij}, q_{ij}^{(k)}, q_{ji}^{(k)}, \mathbf{m}^{(k)}\right], \mathbf{m}^{(k)} = \left[\mathbf{e}_{\text{alg}}^{(k)}, \rho_k, f_k\right]$. The corresponding initial candidate token is: $\mathbf{t}_{ij}^{(k)} = f_{\text{token}}\left(\mathbf{x}_{ij}^{(k)}\right)$. The K candidate tokens for the ordered pair are processed jointly: $\left[\mathbf{h}_{ij}^{(1)}, \ldots, \mathbf{h}_{ij}^{(K)}\right] = \text{TransformerEnc}\left(\left[\mathbf{t}_{ij}^{(1)}, \ldots, \mathbf{t}_{ij}^{(K)}\right]\right)$. Here, $\mathbf{h}_{ij}^{(k)}$ is the contextualized representation of candidate k after self-attention across all candidates. Consequently, the representation captures both the candidate's own evidence and its agreement or conflict with the remaining candidates. E.g., an edge supported by several algorithm families can be treated differently from an edge appearing in only one unstable resampled candidate. Let $\overline{\mathbf{h}}_{ij} = \frac{1}{K}\sum_{\ell=1}^{K} \mathbf{h}_{ij}^{(\ell)}$ denote the mean-pooled representation of all candidates for the ordered pair. The model computes a bounded, candidate- and edge-specific reliability adjustment: $\delta_{ij}^{(k)} = \delta_{\max} \tanh\Big(f_{\text{rel}}\left(\left[\mathbf{h}_{ij}^{(k)}, \overline{\mathbf{h}}_{ij}, \mathbf{z}_{ij}\right]\right)\Big)$, where $\delta_{\max} = 0.75$ limits how far the edge-specific reliability may depart from its family-level prior.

Let $\alpha_m > 0$, with $\sum_{m=1}^{M} \alpha_m = 1$, denote the family-level prior for algorithm m, obtained by aggregating the globally learned candidate-slot weights as described in Section 3.1. The unnormalized reliability score is: $a_{ij}^{(k)} = \log \alpha_{m(k)} + \delta_{ij}^{(k)}$, and the edge-specific candidate weights are obtained by normalizing these scores across all K candidates: $w_{ij}^{(k)} = \frac{\exp\left(a_{ij}^{(k)}\right)}{\sum_{\ell=1}^{K} \exp\left(a_{ij}^{(\ell)}\right)}$. The transformer-weighted prediction is:

$$p_{ij}^{\text{weight}} = \sum_{k=1}^{K} w_{ij}^{(k)} q_{ij}^{(k)}. \tag{1}$$

Because p_{ij}^{weight} is a convex combination of the candidate support values, it cannot move outside the range of support provided by the candidates. E.g., if every candidate assigns little or no support to an edge, the observational features can change which candidate is trusted but cannot correct their shared error. This limitation motivates the candidate-anchored variant, which retains the transformer-weighted prediction as an anchor while permitting a bounded observation-derived correction.

3.3 CANDIDATE-ANCHORED OBSERVATIONAL CORRECTION

The candidate-anchor variant extends edge-adaptive transformer aggregation by permitting a bounded correction informed by the observational pair features and aggregated candidate context. We compute the raw correction score $r_{ij} = f_{\text{corr}}\left(\left[\mathbf{z}_{ij}, \overline{\mathbf{h}}_{ij}\right]\right)$, where $\overline{\mathbf{h}}_{ij}$ is the mean of the contextualized candidate embeddings. The correction magnitude is controlled by a learned global scale: $\lambda_{\text{corr}} = \lambda_{\max}\sigma(\eta)$, where η is a trainable scalar and $\lambda_{\max} = 1$ in our experiments. The bounded edge-specific correction is $c_{ij} = \lambda_{\text{corr}} \tanh(r_{ij})$, so that $|c_{ij}| \leq \lambda_{\text{corr}}$. We apply this correction to the transformer-weighted candidate probability on the logit scale:

$$p_{ij}^{\text{anchor}} = \sigma(\text{logit}(p_{ij}^{\text{weight}}) + c_{ij}), \tag{2}$$

The correction can strengthen or weaken an individual directed-edge probability using observational evidence, while its bounded logit magnitude prevents unrestricted departure from the transformer-

aggregated candidate prediction. The candidate-anchor variant, therefore, provides an intermediate operating point between candidate-only aggregation and independent data-driven prediction.

3.4 DATA-ONLY EXPERT

The data-only expert is a separately trained neural predictor that receives the observation-derived pair vector $\boldsymbol{\phi}_{ij}$ but no candidate graph states or candidate-specific information. Let $\ell_{ij}^{\text{data}} = f_{\text{data}}(\boldsymbol{\phi}_{ij})$ denote its predicted edge logit, where f_{data} represents the complete data-only prediction network. The corresponding edge probability is:

$$p_{ij}^{\text{data}} = \sigma\left(\ell_{ij}^{\text{data}}\right) = \sigma\left(f_{\text{data}}\left(\boldsymbol{\phi}_{ij}\right)\right). \tag{3}$$

Unlike the candidate-weighted variants, the data-only expert can assign high probability to an edge absent from every candidate or low probability to an edge supported by all candidates. This flexibility also makes it more vulnerable to distribution shift, because its predictions depend entirely on observational patterns learned from the synthetic training tasks. Implementation details are provided in Appendix D.4.

3.5 OBSERVABLE HYBRID ROUTING

The hybrid CAFUFO model combines the candidate-anchor and data-only predictions using a single task-level gate. The observable descriptor $\mathbf{u}$ is processed by the routing network: $z_{\text{route}} = f_{\text{route}}(\mathbf{u})$. Larger values of z_{route} indicate a stronger preference for the data-only prediction. We transform this score into a bounded data gate: $g_{\text{data}} = g_{\max}^{\text{data}}\sigma\left(\frac{z_{\text{route}}-a}{T_{\text{route}}}\right)$, where $g_{\max}^{\text{data}} = 0.9$ limits the contribution of the data-only prediction, $a = 1$ is an activation offset, and $T_{\text{route}} = 0.5$ controls the sharpness of the routing decision. The corresponding candidate-anchor gate is $g_{\text{anchor}} = 1 - g_{\text{data}}$. Consequently, $0 < g_{\text{data}} < 0.9$ and $0.1 < g_{\text{anchor}} < 1$, ensuring that the candidate-anchor prediction always retains at least a 10% mixture weight. The final probability for edge $X_i \to X_j$ is:

$$p_{ij}^{\text{hybrid}} = g_{\text{anchor}} p_{ij}^{\text{anchor}} + g_{\text{data}} p_{ij}^{\text{data}}. \tag{4}$$

The same task-level gate is applied to every ordered pair. During router training, the candidate-anchor and data-only variants are held fixed, so the router learns only how strongly to combine their predictions. At inference time, the routing network receives summaries of the observed data, candidate support and disagreement, the two variant predictions, and task size.

3.6 TRAINING OBJECTIVES AND ROUTER SUPERVISION

Training follows the dependency structure of the progressive framework. We first learn the global candidate weights. The transformer-aggregated, candidate-anchor, and data-only variants are then trained separately, and the candidate-anchor and data-only variants are finally held fixed while training the hybrid router. Let: $\mathcal{O} = \{(i,j) : i \neq j\}$ denote the ordered non-diagonal node pairs, let $A_{ij}^{\star}$ be the true edge indicator, and let P_{ij} be a predicted edge probability. All edge-prediction variants share the prediction objective:

$$\mathcal{L}_{\text{pred}}\left(\mathbf{P}, \mathbf{A}^{\star}\right) = \mathcal{L}_{\text{cBCE}}\left(\mathbf{P}, \mathbf{A}^{\star}\right) + \lambda_{\text{cal}} \frac{1}{|\mathcal{O}|} \sum_{(i,j)\in\mathcal{O}} \left(P_{ij} - A_{ij}^{\star}\right)^2, \tag{5}$$

where $\mathcal{L}_{\text{cBCE}}$ is class-balanced binary cross-entropy and the second term is a Brier-style calibration penalty. We use $\lambda_{\text{cal}} = 0.1$. We learn one non-negative global weight γ_k for each candidate by minimizing ordinary edgewise BCE:

$$\mathcal{L}_{\text{global}} = \mathcal{L}_{\text{BCE}}(\sum_{k=1}^{K} \gamma_k \mathbf{Q}^{(k)}, \mathbf{A}^{\star}), \qquad \gamma_k \geq 0, \qquad \sum_{k=1}^{K} \gamma_k = 1. \tag{6}$$

For the transformer-based variants, let $\mathcal{R}_{\text{KL}}$ regularize the edge-specific candidate weights toward the global candidate weights, $\mathcal{R}_{\text{dens}}$ penalize changes in mean edge probability relative to the corresponding global-weighted candidate aggregate, and $\mathcal{R}_{\text{corr}}$ penalize large observation-derived

corrections in the candidate-anchored variant. The three variant objectives are:

$$\mathcal{L}_{\text{weight}} = \mathcal{L}_{\text{pred}}\left(\mathbf{P}^{\text{weight}}, \mathbf{A}^{\star}\right) + \lambda_{\text{KL}}\mathcal{R}_{\text{KL}} + \lambda_{\text{dens}}\mathcal{R}_{\text{dens}}, \tag{7}$$

$$\mathcal{L}_{\text{anchor}} = \mathcal{L}_{\text{pred}}\left(\mathbf{P}^{\text{anchor}}, \mathbf{A}^{\star}\right) + \lambda_{\text{KL}}\mathcal{R}_{\text{KL}} + \lambda_{\text{dens}}\mathcal{R}_{\text{dens}} + \lambda_{\text{reg,corr}}\mathcal{R}_{\text{corr}}, \tag{8}$$

$$\mathcal{L}_{\text{data}} = \mathcal{L}_{\text{pred}}\left(\mathbf{P}^{\text{data}}, \mathbf{A}^{\star}\right). \tag{9}$$

For hybrid routing variant – the trained candidate-anchored variant and data-only expert variants are held fixed. For router-training task t, let $R^{(t)}_{\text{anchor}}$ and $R^{(t)}_{\text{data}}$ denote their class-balanced edge-prediction risks. We define the relative advantage of the data-only prediction as $\Delta_t = \frac{R^{(t)}_{\text{anchor}} - R^{(t)}_{\text{data}}}{\max\left\{\epsilon, \frac{1}{2}\left(R^{(t)}_{\text{anchor}} + R^{(t)}_{\text{data}}\right)\right\}}, \epsilon = 10^{-6}$. Thus, $\Delta_t > 0$ indicates that the *Data-Only Expert* has lower risk, while the magnitude of Δ_t reflects the normalized risk gap and hence the potential routing regret from favoring the inferior variant. We convert this advantage into the soft routing target $y_t = \sigma\left(\frac{\Delta_t - \mu}{T_{\text{target}}}\right)$, where μ specifies the minimum relative advantage that the *data-only expert* must achieve before the router is encouraged to favor the data-only prediction, and T_{target} controls the softness of the target. Let $z_t = f_{\text{route}}(\mathbf{u}^{(t)})$ be the router output and $g^{(t)}_{\text{data}}$ the corresponding data-only gate. The router objective is:

$$\begin{aligned}\mathcal{L}_{\text{hybrid}} = \mathcal{L}_{\text{pred}}\left(\mathbf{P}^{\text{hybrid}}, \mathbf{A}^{\star}\right) + \lambda_{\text{route}}\,\text{BCEWithLogits}\left(z_t, y_t\right) \\ + \lambda_{\text{false}} g^{(t)}_{\text{data}}\left[\mu - \Delta_t\right]_+ + \lambda_{\text{use}}\left(g^{(t)}_{\text{data}}\right)^2,\end{aligned} \tag{10}$$

where $[x]_+ = \max(x, 0)$. The first term optimizes the final hybrid edge prediction. The second trains the router from the relative risks of the two fixed variants. The third reduces avoidable routing regret by penalizing data-only routing when the *data-only expert* does not exceed the required advantage margin, while the final term mildly discourages unnecessary reliance on it. Only the router is updated at this stage. Full regularizer definitions and coefficients are provided in Appendix E.

3.7 Calibration and DAG Decoding

We calibrate each method's ordered-edge probabilities using scalar temperature scaling: $\widetilde{P}_{ij} = \sigma(\frac{\text{logit}(P_{ij})}{\tau})$, where the temperature $\tau > 0$ and decision threshold η are selected using only the designated calibration split. To guarantee acyclicity, we assign node i the score: $s_i = \sum_{\ell \neq i} \widetilde{P}_{\ell i} - \sum_{\ell \neq i} \widetilde{P}_{i\ell}$, and sort the nodes in ascending score order, using deterministic tie-breaking. We retain $i \to j$ only when $\widetilde{P}_{ij} \geq \eta$ and node i precedes node j in this ordering. Because every retained edge follows the same total node order, the resulting graph is a DAG by construction. All probability-based methods use the same calibration and decoding protocol (details are provided in Appendix F).

4 Experiments

We evaluate whether combining learned candidate-graph aggregation with direct observational evidence improves directed causal recovery. Our primary candidate pool comprises PC, GES-BIC, ICA-LiNGAM, NOTEARS, and DAGMA. Each algorithm contributes one graph estimated from the complete dataset and two graphs estimated from bootstrap resamples, yielding a shared evidence budget of 15 candidate graphs per task. We evaluate five progressively structured variants of the CaFuFo framework: global candidate weighting, transformer-aggregated weights only, the candidate-anchored, the data-only expert, and the observable hybrid, which routes between the candidate-anchor and data-only predictions. Candidate-only comparators include uniform voting, the best constituent selected on validation tasks, each constituent algorithm's three-member aggregate, and a shared-budget adaptation of the released Causal-Bayes-Ensemble models (Vo et al., 2026). We also compare against AVICI and DiBS-BGe, where DiBS is instantiated with the Bayesian Gaussian equivalent (BGe) score, as representative amortized and Bayesian differentiable causal-discovery methods, respectively.

Our primary metrics are directed F1 and area under the precision–recall curve (AUPRC). We additionally report skeleton F1, normalized structural Hamming distance (nSHD), expected calibration

Table 1: Results over 90 locked tasks. Bold and underline denote the best and second-best values, respectively.

Group	Method	Directed F1 ↑	AUPRC ↑	Skeleton F1 ↑	nSHD ↓	ECE ↓	Edge ratio
CAFUFO	Observable hybrid	**0.661**	**0.759**	0.767	0.613	0.066	1.063
	Candidate anchor	<u>0.659</u>	<u>0.722</u>	0.781	0.556	0.039	0.989
	Transformer weights only	0.647	0.708	0.764	0.563	0.038	0.939
	Global candidate weights	0.608	0.665	0.767	0.536	**0.027**	0.756
	Data-only expert	0.315	0.370	0.497	2.502	0.309	2.855
Candidate-only comparators	CBE released-model adapter	0.612	0.664	0.777	<u>0.514</u>	0.034	0.747
	Uniform vote	0.651	0.702	**0.826**	**0.469**	0.032	0.851
	Best validation constituent	0.571	0.543	0.711	0.688	0.076	0.940
Standalone baselines	AVICI	0.630	0.707	0.710	0.851	0.055	1.242
	DiBS-BGe	0.576	0.576	<u>0.817</u>	0.547	0.030	0.874
Individual candidate aggregates	GES-BIC	0.571	0.543	0.711	0.688	0.076	0.940
	NOTEARS	0.522	0.487	0.694	0.853	0.030	1.099
	DAGMA	0.499	0.452	0.694	0.891	<u>0.029</u>	1.133
	ICA-LiNGAM	0.438	0.560	0.634	1.363	0.066	1.829
	PC	0.433	0.407	0.691	0.728	0.036	0.688

error (ECE), and predicted-to-true edge ratio. After data generation, the resulting tasks are partitioned into six disjoint sets with distinct roles: *training*, *validation*, *calibration*, *development*, *external-transfer*, and *locked-test*. Tasks in the training partition are used to fit model parameters; the validation partition is used for early stopping and model selection; and the calibration partition is reserved for temperature scaling and edge-threshold selection. The development partition is used only for iterative evaluation and analysis during method development, while the external-transfer partition evaluates generalization to shifted or unseen problem settings. The locked-test partition is held out until final evaluation. Ground-truth graphs from the development, external-transfer, and locked-test partitions are never used for parameter fitting, model selection, or calibration. For each learned CAFUFO variant, we train three independent seeds. For each seed, temperature scaling and edge-threshold selection are performed separately using the calibration partition, after which the resulting edge probabilities are decoded independently. For each task, metrics are averaged across the three seeds.

We report results in three complementary evaluation settings: (i) one-shot evaluation on 90 previously untouched tasks from the *locked-test* partition; (ii) *external-transfer* evaluation on continuous structural-equation data generated over the unseen Asia, CHILD, and ALARM graph topologies; and (iii) evaluation on the observational Sachs protein-signalling benchmark. Together, these settings assess performance on held-out synthetic tasks, transfer to unseen causal structures, and generalization to real observational data, respectively.

4.1 ONE-SHOT EVALUATION ON LOCKED TASKS

Our principal evaluation comprises 90 previously untouched tasks, distributed equally across three challenging regimes: noisy, post-nonlinear scale-free graphs; large nonlinear Erdős–Rényi graphs with very small samples; and noisy linear scale-free graphs with small samples. The candidate budget, calibration rules, task partitions, and evaluation protocol were frozen before the locked tasks were generated. Full protocol and definitions appear in Appendix G. Table 1 reports the results. Among the pre-specified methods, the CAFUFO observable hybrid achieves the highest directed F1 (0.661) and AUPRC (0.759). Relative to the shared-budget CBE adapter, it improves directed F1 by 0.050 (95% CI $[0.022, 0.077]$, $p = 0.009$) and AUPRC by 0.094 (95% CI $[0.071, 0.118]$, $p < 10^{-4}$). Uniform voting instead provides the strongest skeleton F1 and nSHD, while global candidate weighting gives the lowest ECE. The observable hybrid, therefore, improves directed-edge recovery and probability ranking, but does not dominate every structural or calibration metric.

The candidate anchor nearly matches the observable hybrid in directed F1 (0.659 versus 0.661); their difference is not significant. The observable hybrid nevertheless improves AUPRC over the anchor by 0.036 ($p < 10^{-4}$) and improves directed F1 and AUPRC over transformer weights only by 0.015 ($p = 0.030$) and 0.051 ($p < 10^{-4}$), respectively. Thus, the two observational mechanisms play complementary roles: the candidate-anchored correction brings thresholded graph recovery close to the hybrid level, while hybrid routing provides a substantial and statistically significant additional gain in edge ranking. Overall, the hybrid achieves the strongest improvement over transformer-only aggregation, particularly in AUPRC.

Table 2: Continuous structural-equation datasets generated over unseen Asia, CHILD, ALARM graphs.

Group	Method	Directed F1 ↑	AUPRC ↑	Skeleton F1 ↑	nSHD ↓	ECE ↓	Edge ratio
CaFuFo	Observable hybrid	**0.676**	**0.848**	0.876	0.426	0.050	0.909
	Candidate anchor	0.658	0.848	0.907	0.436	0.050	0.994
	Transformer weights only	0.633	0.832	0.863	0.500	0.046	0.976
	Global candidate weights	0.669	0.840	0.883	0.423	**0.029**	0.898
	Data-only expert	0.267	0.406	0.560	1.647	0.262	1.670
Candidate-only comparators	Uniform vote	0.668	0.770	**0.912**	**0.420**	0.041	0.988
	Best validation constituent	0.625	0.708	0.812	0.571	0.068	1.031
Individual candidate aggregates	GES-BIC	0.625	0.708	0.812	0.571	0.068	1.031
	PC	0.506	0.590	0.837	0.612	0.044	0.862
	NOTEARS	0.499	0.422	0.831	0.630	0.046	0.881
	DAGMA	0.461	0.383	0.818	0.712	0.051	0.948
	ICA-LiNGAM	0.449	0.594	0.777	0.944	0.064	1.423

All candidate-using learned variants – global candidate weighting, transformer weights only, the candidate anchor, and the observable hybrid – outperform the best validation-selected constituent. The data-only expert does not, illustrating the importance of retaining candidate evidence.

AVICI attains 0.630 directed F1 and 0.707 AUPRC, while DiBS-BGe attains 0.576 and 0.576, respectively. The observable hybrid significantly improves AUPRC over AVICI ($+0.051$, $p = 0.020$), although their directed-F1 difference is not significant ($+0.031$, $p = 0.078$). It significantly improves both directed F1 and AUPRC over DiBS-BGe. Complete paired statistics, per-regime results, runtime settings, and baseline configuration details are reported in Appendix G.

4.2 Transfer to Unseen Graph Topologies

We next evaluate the *external-transfer* partition. Here, tasks are generated from the published Asia, CHILD, and ALARM Bayesian-network structures. Using each of these unseen topology, we generate continuous observations from linear and nonlinear structural equation models. For every topology–mechanism combination, we generate five independently seeded datasets of $1{,}000$ observations, yielding 30 external-transfer tasks. These experiments, therefore, test transfer to previously unseen causal structures rather than additional samples from the training topology distribution. They are semi-synthetic topology-transfer experiments and should not be interpreted as evaluations of the original discrete Bayesian-network datasets. All learned CaFuFo parameters, model-selection choices, temperature-scaling parameters, and edge-decision thresholds remain fixed from the preceding training, validation, and calibration stages. On each external-transfer task, only the task-specific candidate predictions and observation-derived features are recomputed from the new observational data; no CaFuFo parameter fitting, recalibration, or threshold tuning is performed.

As shown in Table 2, the CaFuFo observable hybrid achieves the highest directed F1 (0.676); it exceeds the candidate anchor by 0.017 (95% CI $[0.006, 0.031]$, $p = 0.005$). The observable hybrid and candidate anchor both attain 0.848 AUPRC. Uniform voting provides the strongest skeleton F1 and nSHD, whereas global candidate weighting gives the lowest ECE. Thus, observational routing improves directed recovery under topology shift, while simpler candidate-only variants remain competitive on structural and calibration metrics. Per-topology, per-mechanism, and paired statistical results are provided in Appendix G.

4.3 Evaluation on Sachs

We evaluate transfer to the Sachs protein-signalling benchmark, which contains $7{,}466$ measurements of 11 variables. No Sachs labels are used for training, calibration, threshold selection, or model selection. Table 3 separates two complementary comparisons. The left panel evaluates the proposed CaFuFo variants and candidate-only ensemble comparators. The PC three-member aggregate combines one full-data PC graph with two bootstrap PC graphs. Matched full-data PC (PC (Right Table)) is repeated in this panel as a single-algorithm reference point.

In the left panel, global candidate weighting and matched full-data PC obtain the highest directed F1 (0.439), while global weighting achieves the highest AUPRC (0.418). Among the edge-adaptive CaFuFo variants, the candidate anchor provides the strongest directed F1 (0.397), whereas transformer

Table 3: Sachs evaluation. Left: proposed framework variants and candidate-only ensemble comparators. Right: matched constituent algorithms run once on the complete dataset.

Method	Dir. F1 ↑	AUPRC ↑	Skel. F1 ↑	nSHD ↓
Global candidate weights	**0.439**	**0.418**	**0.439**	1.353
PC three-member aggregate	0.410	0.311	0.410	1.353
Candidate anchor	0.397	0.303	0.397	1.157
Transformer weights only	0.355	0.344	0.403	**1.020**
Uniform vote	0.324	0.338	0.378	1.412
Observable hybrid	0.258	0.260	0.325	1.275
CBE released-model adapter	0.258	0.285	0.323	1.294
Data-only expert	0.156	0.195	0.322	1.373
PC (Right Table)	**0.439**	0.309	**0.439**	1.353

Algorithm	Dir. F1	AUPRC	Skel. F1	nSHD	Edge ratio
PC	**0.439**	**0.309**	0.439	1.353	1.412
ICA-LiNGAM	0.211	0.205	0.526	1.412	1.235
GES-BIC	0.200	0.158	**0.560**	1.824	1.941
NOTEARS	0.087	0.180	0.261	**1.118**	0.353
DAGMA	0.077	0.157	0.231	1.294	0.529

weights only obtains the lowest nSHD (1.020). The observable hybrid and CBE adapter both reach 0.258 directed F1. The weak data-only result suggests that observation–edge relationships learned from the synthetic training distribution do not transfer reliably to Sachs; routing toward this expert, therefore, reduces the performance of the candidate anchor. *Developing richer synthetic training distributions that better capture the structural and observational characteristics of real systems such as Sachs is, therefore, an important direction for future work*. The right panel shows that full-data PC is the strongest constituent for directed F1 and AUPRC, whereas GES-BIC provides the strongest skeleton F1 and NOTEARS the lowest nSHD.

Scalability. The current implementation evaluates candidate pools on graphs of up to 37 nodes. Our analysis on 56-node HAILFINDER topology indicated that candidate construction, rather than transformer fusion, becomes the principal computational bottleneck: unbounded GES-BIC did not complete its first full-data run within 120 minutes, while the remaining constituents completed substantially faster. Extending CAFUFO to larger graphs will, therefore, require scalable or locally restricted constituent algorithms, candidate screening, or decomposed graph construction.

5 CONCLUSION

We introduced CAFUFO to address two limitations of existing candidate-fusion approaches. First, rather than assigning each candidate a single global reliability across all edges, our *edge-adaptive transformer aggregation* learns edge-specific candidate reliabilities from candidate representations and observation-derived pairwise features. The *candidate-anchored observational correction* further relaxes the resulting candidate-only constraint by allowing a bounded, data-derived adjustment. Second, existing aggregation methods remain fundamentally limited by the information contained in their constituent algorithms and do not directly exploit the observed data as an independent predictive source. Our *data-only expert* addresses this limitation by predicting directly from observation-derived features, while *observable hybrid routing* combines the predictions of the *candidate-anchored observational correction* and the *data-only expert*. Together, these components allow CAFUFO to exploit both candidate-based and observation-driven evidence.

In the one-shot evaluation on 90 previously untouched locked-test tasks, the observable hybrid improves directed F1 and AUPRC over the shared-budget CBE adapter models. It also improves directed-edge recovery over *Global Candidate Weighting* and the *Best Validation Constituent*, while remaining competitive when transferred without adaptation to unseen Asia, CHILD, and ALARM graph topologies. The Sachs results expose an important limitation. The weak transfer of the *data-only expert* suggests that observation–edge relationships learned from the synthetic training distribution may not necessarily transfer to real observational systems; consequently, routing toward this expert can reduce the performance of the candidate-anchor under strong synthetic-to-real shift.

Two directions are particularly important for future work. First, richer and more realistic synthetic task distributions are needed to reduce the synthetic-to-real gap and improve the transferability. Second, a promising direction is to make the hybrid router more explicitly robust to observational distribution shift, improving its ability to identify when the data-only prediction remains reliable and when greater weight should be placed on the candidate-anchored prediction. These directions offer a path toward causal-discovery systems that adapt both to variation among candidate algorithms and to changes in the observational regime itself.

AI USE STATEMENT

In this work, we used generative AI tools, including ChatGPT and Codex, to provide feedback on experimental design, assist with implementation and debugging, support literature search and result interpretation, and draft and edit portions of the manuscript. All experiments were executed by the authors. The authors reviewed and verified the AI-assisted code, analyses, numerical claims, citations, and text, and take responsibility for the final content of the work.

REFERENCES

Lars Lorch, Scott Sussex, Jonas Rothfuss, Andreas Krause, and Bernhard Schölkopf. Amortized inference for causal structure learning. In *Advances in Neural Information Processing Systems*, 2022.

Lars Lorch, Jonas Rothfuss, Bernhard Schölkopf, and Andreas Krause. DiBS: Differentiable Bayesian structure learning. In *Advances in Neural Information Processing Systems*, 2021.

Kevin Bello, Bryon Aragam, and Pradeep Ravikumar. DAGMA: Learning DAGs via M-matrices and a log-determinant acyclicity characterization. In *Advances in Neural Information Processing Systems*, 2022.

David Maxwell Chickering. Optimal structure identification with greedy search. *Journal of Machine Learning Research*, Volume 3, Pages: 507–554, 2002.

Bradley Efron and Robert Tibshirani. *An Introduction to the Bootstrap (1st Ed)*. Chapman and Hall/CRC, 1994.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In *Proceedings of the International Conference on Machine Learning*, 2017.

Paul Rolland, Volkan Cevher, Matthäus Kleindessner, Chris Russell, Dominik Janzing, Bernhard Schölkopf, and Francesco Locatello. Score matching enables causal discovery of nonlinear additive noise models. In *Proceedings of the International Conference on Machine Learning*, 2022.

Karen Sachs, Omar Perez, Dana Pe'er, Douglas A. Lauffenburger, and Garry P. Nolan. Causal protein-signaling networks derived from multiparameter single-cell data. *Science*, 308(5721):523–529, 2005.

Shohei Shimizu, Patrik O. Hoyer, Aapo Hyvärinen, and Antti Kerminen. A linear non-Gaussian acyclic model for causal discovery. *Journal of Machine Learning Research*, 7:2003–2030, 2006.

Peter Spirtes, Clark Glymour, and Richard Scheines. *Causation, Prediction, and Search*. MIT Press, second edition, 2000.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017.

Vy Vo, Haoxuan Li, and Mingming Gong. Causal discovery in the wild: A voting-theoretic ensemble approach. In *International Conference on Learning Representations*, 2026.

Xun Zheng, Bryon Aragam, Pradeep K. Ravikumar, and Eric P. Xing. DAGs with NO TEARS: Continuous optimization for structure learning. In *Advances in Neural Information Processing Systems*, 2018.

Babak Aslani and Shima Mohebbi. Ensemble framework for causality learning with heterogeneous Directed Acyclic Graphs through the lens of optimization. *Computers & Operations Research*, 152, 2023. doi:10.1016/j.cor.2023.106148.

Sindy Löwe, David Madras, Richard Zemel, and Max Welling. Amortized causal discovery: Learning to infer causal graphs from time-series data. In *Proceedings of the First Conference on Causal Learning and Reasoning*, volume 177 of *Proceedings of Machine Learning Research*, pp. 509–525, 2022. URL https://proceedings.mlr.press/v177/lowe22a.html.

Nan Rosemary Ke, Silvia Chiappa, Jane X. Wang, Jörg Bornschein, Anirudh Goyal, Melanie Rey, Théophane Weber, Matthew Botvinick, Michael C. Mozer, and Danilo J. Rezende. Learning to induce causal structure. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=hp_RwhKDJ5.

Francesco Montagna, Max Cairney-Leeming, Dhanya Sridhar, and Francesco Locatello. Demystifying amortized causal discovery with transformers. *Transactions on Machine Learning Research*, 2025. URL https://mlanthology.org/tmlr/2025/montagna2025tmlr-demystifying/.

Table 4: Variants of CAFUFO and the information used by each variant. Here, $\mathbf{c}_{ij}$ denotes the candidate representation for ordered pair (i, j), $\boldsymbol{\phi}_{ij}$ denotes the observation-derived pair features, and $\mathbf{u}$ denotes the task-level descriptor used by the hybrid router.

Variant	Information used	Prediction mechanism
Global Candidate Weighting	$\mathbf{c}_{ij}$	Learns one task-invariant global weight γ_k for each candidate and returns the corresponding global-weighted candidate aggregate.
Edge-Adaptive Transformer Aggregation	$\mathbf{c}_{ij},\ \boldsymbol{\phi}_{ij}$	Uses candidate representations and observation-derived pair features to learn edge-specific candidate reliabilities. The prediction remains a convex combination of candidate directional-support values.
Candidate-Anchored Observational Correction	$\mathbf{c}_{ij},\ \boldsymbol{\phi}_{ij}$	Starts from the edge-adaptive candidate aggregate and adds a bounded observation-derived correction, allowing the prediction to move beyond the support provided by the candidates alone.
Data-Only Expert	$\boldsymbol{\phi}_{ij}$	Predicts edge probabilities directly from the observation-derived pair features, without access to $\mathbf{c}_{ij}$.
Observable Hybrid Routing	$p_{ij}^{\mathrm{anchor}},\ p_{ij}^{\mathrm{data}},\ \mathbf{u}$	Combines the predictions of the *Candidate-Anchored Observational Correction* and the *Data-Only Expert* using the learned hybrid router.

A FRAMEWORK VARIANTS

The five evaluated CAFUFO variants form a progression from global candidate weighting to edge-adaptive candidate aggregation, direct observational correction, and hybrid combination with an independent data-only prediction. Table 4 summarizes the information available to each variant and the corresponding prediction mechanism. The variants isolate four progressively richer capabilities. First, *Global Candidate Weighting* tests whether learning a global reliability for each candidate improves over uniform aggregation. Second, *Edge-Adaptive Transformer Aggregation* tests whether candidate reliability should vary across ordered edges rather than remain fixed globally. Third, *Candidate-Anchored Observational Correction* tests whether direct observational evidence can improve predictions beyond reweighting the candidate ensemble alone. Finally, *Observable Hybrid Routing* tests whether the complementary predictions of the candidate-anchored variant and the independent *Data-Only Expert* can be combined adaptively. *Global Candidate Weighting* serves as a lightweight framework variant and control; weighted ensembling itself is not claimed as a new general principle. The principal methodological contributions are *Edge-Adaptive Transformer Aggregation*, *Candidate-Anchored Observational Correction*, and *Observable Hybrid Routing*.

B CANDIDATE REPRESENTATION AND ITS USE ACROSS FRAMEWORK VARIANTS

This appendix defines the common representation used to reconcile the heterogeneous outputs of the constituent causal discovery algorithms. It then explains how candidate information is progressively incorporated by global candidate weighting, edge-adaptive transformer aggregation, candidate-anchored observational correction, and full hybrid routing.

B.1 CANDIDATE INDEXING AND EVIDENCE BUDGET

Let: $\mathcal{C} = \{\mathbf{Q}^{(k)}\}_{k=1}^{K}$ denote the candidate set. Candidate $k = (m, b)$ is identified by its algorithm family m and data member b. In the primary (CBE-matched5 pool), we have: $m \in \{\mathrm{PC}, \mathrm{GES\text{-}BIC}, \mathrm{ICA\text{-}LiNGAM}, \mathrm{NOTEARS}, \mathrm{DAGMA}\}$. Each algorithm contributes one full-data

estimate, indexed by $b = 0$, and two estimates fitted to independently resampled datasets, indexed by $b \in \{1, 2\}$. The primary evidence budget therefore contains $K = 5 \times 3 = 15$ candidate graphs. Each candidate is aligned to the common source–target convention in which matrix entry (i, j) describes the proposed edge $X_i \to X_j$.

B.2 Directional-Support Values

Each candidate k is represented by a directional-support matrix:

$$\mathbf{Q}^{(k)} = \left[q_{ij}^{(k)}\right]_{i,j=1}^{d}, \qquad q_{ij}^{(k)} \in [0, 1],$$

where $q_{ij}^{(k)}$ measures candidate k's support for $X_i \to X_j$. By "support for $X_i \to X_j$," we mean simply the degree to which candidate k endorses the directed edge $X_i \to X_j$ in its output representation. These values are common support scores rather than necessarily calibrated causal probabilities. For PC and GES-BIC, no adjacency, $i \to j$, $j \to i$, and an unoriented adjacency $i - j$ are represented by: $(0, 0), (1, 0), (0, 1), (0.5, 0.5)$, respectively, where each ordered pair denotes: $\left(q_{ij}^{(k)}, q_{ji}^{(k)}\right)$. Consequently, an unoriented adjacency retains symmetric support rather than being assigned an arbitrary direction.

ICA-LiNGAM, NOTEARS, and DAGMA return weighted directed graphs. After aligning their matrices to the source–target convention, coefficients below the algorithm-specific threshold are set to zero. The remaining absolute magnitudes are normalized within each candidate:

$$q_{ij}^{(k)} = \begin{cases} \dfrac{\left|w_{ij}^{(k)}\right|}{\max_{r \neq s} \left|w_{rs}^{(k)}\right|}, & \max_{r \neq s} \left|w_{rs}^{(k)}\right| > 0, \\ 0, & \text{otherwise.} \end{cases}$$

The coefficient sign is, therefore, not used as an edge direction; direction is determined by the matrix position, while the normalized absolute coefficient supplies the support magnitude. For every ordered pair (i, j), both $q_{ij}^{(k)}$ and $q_{ji}^{(k)}$ are retained. This allows the learned aggregation model to distinguish forward support, reverse support, absence, and an unoriented adjacency.

B.3 Candidate Metadata

Suppose the active candidate pool contains M algorithm families. Candidate $k = (m, b)$ receives an algorithm-family indicator:

$$\mathbf{e}_{\text{alg}}^{(k)} \in \{0, 1\}^M,$$

whose m-th coordinate equals one. Two additional candidate-level descriptors are included. The first is the candidate-density statistic:

$$\rho_k = \frac{1}{d(d-1)} \sum_{i \neq j} \mathbf{1}\left[q_{ij}^{(k)} \geq 0.5\right].$$

The second is the distinct-observation fraction f_k. For the full-data member, $f_k = 1$, whereas for a resampled member:

$$f_k = \frac{\text{number of distinct observations in the resample}}{n}.$$

Thus, f_k records the effective diversity of the fitting sample rather than only indicating whether resampling occurred. The metadata vector is:

$$\mathbf{m}^{(k)} = \left[\mathbf{e}_{\text{alg}}^{(k)}, \rho_k, f_k\right] \in \mathbb{R}^{M+2}.$$

For ordered pair (i, j), the complete candidate representation is:

$$\mathbf{c}_{ij}^{(k)} = \left[q_{ij}^{(k)}, q_{ji}^{(k)}, \mathbf{e}_{\text{alg}}^{(k)}, \rho_k, f_k\right].$$

Therefore, for the primary five-algorithm pool, this vector has: $2 + 5 + 2 = 9$ coordinates.

C Observation-Derived Features and Hybrid-Routing Descriptor

This appendix defines the observation-derived features used by edge-adaptive transformer aggregation, candidate-anchored observational correction, the independent data-driven expert, and full hybrid routing. No feature contains the true adjacency matrix or an oracle indicator of the graph family, structural-equation mechanism, noise regime, or evaluation domain. Let:

$$\mathbf{X} = [X_1, \dots, X_d] \in \mathbb{R}^{n \times d}$$

denote a dataset containing n observations of d variables.

C.1 Preprocessing

Each variable is standardized independently to have zero empirical mean and unit empirical variance before candidate generation and feature extraction. For Sachs, missing values, if present, are first replaced by their column medians. The same feature construction is subsequently applied to synthetic, topology-transfer, and observational datasets.

C.2 Node-Level Features

For node i, we initially compute

$$\widetilde{\boldsymbol{\psi}}_i = \left[\widehat{\mu}_i,\, \widehat{\sigma}_i,\, \widehat{\mathrm{skew}}(X_i),\, \widehat{\mathrm{kurt}}(X_i),\, \min(X_i),\, \max(X_i)\right] \in \mathbb{R}^6,$$

where skewness is the bias-corrected sample skewness and kurtosis is the bias-corrected excess kurtosis.

Each of the six coordinates is standardized across the d nodes within the current task:

$$\psi_{ir} = \frac{\widetilde{\psi}_{ir} - \mathrm{Mean}_\ell\, \widetilde{\psi}_{\ell r}}{\mathrm{Std}_\ell\, \widetilde{\psi}_{\ell r}}, \qquad r = 1, \dots, 6.$$

Coordinates having zero variance across nodes are mapped to zero. Undefined or non-finite values are replaced by finite defaults before use. The resulting representation is:

$$\boldsymbol{\psi}_i = [\psi_{i1}, \dots, \psi_{i6}] \in \mathbb{R}^6.$$

Because the input variables have already been standardized, their means and standard deviations are approximately constant.

C.3 Ordered-Pair Features

For every ordered pair $i \neq j$, we compute:

$$\boldsymbol{\phi}_{ij} = [\phi_{ij}^{(1)}, \dots, \phi_{ij}^{(8)}] \in \mathbb{R}^8.$$

The representation contains both symmetric and directional statistics, so $\boldsymbol{\phi}_{ij}$ need not equal $\boldsymbol{\phi}_{ji}$. Let:

$$r(a, b) = \mathrm{corr}(a, b),$$

with $r(a, b) = 0$ whenever either argument has empirical standard deviation below 10^{-8}. Define the no-intercept regression coefficients:

$$\widehat{\beta}_{j|i} = \frac{X_i^\top X_j}{X_i^\top X_i + 10^{-8}}, \qquad \widehat{\beta}_{i|j} = \frac{X_j^\top X_i}{X_j^\top X_j + 10^{-8}},$$

and residual variances:

$$v_{j|i} = \mathrm{Var}\left(X_j - \widehat{\beta}_{j|i} X_i\right) + 10^{-8},$$
$$v_{i|j} = \mathrm{Var}\left(X_i - \widehat{\beta}_{i|j} X_j\right) + 10^{-8}.$$

Finally, let:

$$\widetilde{\sigma}_i = \widehat{\sigma}_i + 10^{-6}.$$

The eight pairwise features are:

$$\begin{aligned}
\phi_{ij}^{(1)} &= r(X_i, X_j), \quad & \phi_{ij}^{(2)} &= |r(X_i, X_j)|, \\
\phi_{ij}^{(3)} &= r(X_i^2, X_j), \quad & \phi_{ij}^{(4)} &= r(X_i, X_j^2), \\
\phi_{ij}^{(5)} &= \log \frac{v_{j|i}}{v_{i|j}}, & \phi_{ij}^{(6)} &= \frac{\widehat{\mu}_i - \widehat{\mu}_j}{\widetilde{\sigma}_i + \widetilde{\sigma}_j}, \\
\phi_{ij}^{(7)} &= \log \frac{\widetilde{\sigma}_i}{\widetilde{\sigma}_j}, & \phi_{ij}^{(8)} &= \frac{\log(1+n)}{10}.
\end{aligned}$$

The first two coordinates describe symmetric linear association. The third and fourth capture asymmetric second-order associations. The fifth compares the residual variances obtained under the two regression directions. The sixth and seventh compare marginal location and scale, while the eighth records sample size. Diagonal entries are fixed to: $\boldsymbol{\phi}_{ii} = \mathbf{0}$ and are excluded from training losses, routing summaries, and evaluation.

C.4 Observable Hybrid-Routing Descriptor

The full hybrid CaFuFo's variant uses one task-level routing gate shared across all ordered node pairs in a dataset. Its input is:

$$\mathbf{u} = [\mathbf{u}_{\text{node}}, \mathbf{u}_{\text{pair}}, \mathbf{u}_{\text{cand}}, \mathbf{u}_{\text{anchor}}, \mathbf{u}_{\text{data}}, u_{\text{gap}}, \mathbf{u}_{\text{size}}] \in \mathbb{R}^{38}.$$

Let: $\mathcal{O} = \{(i,j) : i \neq j\}$ denote the ordered off-diagonal pairs, and let K be the number of candidates. In the primary CBE-matched5 pool, $K = 15$.

Node-feature summary. For each of the six node-feature coordinates, the router receives its mean and population standard deviation across nodes:

$$\mathbf{u}_{\text{node}} = [\text{Mean}_i(\boldsymbol{\psi}_i), \text{Std}_i(\boldsymbol{\psi}_i)] \in \mathbb{R}^{12}.$$

Pair-feature summary. For each of the eight pair-feature coordinates, the router receives its mean and population standard deviation over ordered off-diagonal pairs:

$$\mathbf{u}_{\text{pair}} = \left[\text{Mean}_{(i,j)\in\mathcal{O}}\left(\boldsymbol{\phi}_{ij}\right), \text{Std}_{(i,j)\in\mathcal{O}}\left(\boldsymbol{\phi}_{ij}\right)\right] \in \mathbb{R}^{16}.$$

Candidate-support summary. For each ordered pair, candidate disagreement is:

$$d_{ij} = \text{Std}_{k=1}^{K}\left(q_{ij}^{(k)}\right).$$

The candidate component is:

$$\mathbf{u}_{\text{cand}} = \left[\text{Mean}_{k,(i,j)\in\mathcal{O}}\, q_{ij}^{(k)}, \text{Mean}_{(i,j)\in\mathcal{O}}\, d_{ij}, \text{Std}_{(i,j)\in\mathcal{O}}\, d_{ij}\right] \in \mathbb{R}^3.$$

Variant-prediction summaries. The candidate-anchor prediction is summarized by:

$$\mathbf{u}_{\text{anchor}} = \left[\text{Mean}_{(i,j)\in\mathcal{O}}\, p_{ij}^{\text{anchor}}, \text{Std}_{(i,j)\in\mathcal{O}}\, p_{ij}^{\text{anchor}}\right] \in \mathbb{R}^2,$$

and the independent data-driven prediction by:

$$\mathbf{u}_{\text{data}} = \left[\text{Mean}_{(i,j)\in\mathcal{O}}\, p_{ij}^{\text{data}}, \text{Std}_{(i,j)\in\mathcal{O}}\, p_{ij}^{\text{data}}\right] \in \mathbb{R}^2.$$

Their mean absolute discrepancy is:

$$u_{\text{gap}} = \text{Mean}_{(i,j)\in\mathcal{O}}\left|p_{ij}^{\text{anchor}} - p_{ij}^{\text{data}}\right| \in \mathbb{R}.$$

Problem-size descriptors. The final two coordinates are:

$$\mathbf{u}_{\text{size}} = \left[\frac{d}{20}, \frac{K}{10}\right] \in \mathbb{R}^2.$$

The total dimension is: $12 + 16 + 3 + 2 + 2 + 1 + 2 = 38$. All standard deviations in the routing descriptor use the population definition.

Although the routing vector contains summaries of the two variant predictions, these predictions are deterministic functions of $(\mathbf{X}, \mathcal{C})$. We therefore use the compact notation

$$\mathbf{u} = U(\mathbf{X}, \mathcal{C}).$$

C.5 Information Available at Inference Time

At inference time, the framework has access only to the observed dataset, the generated candidate graphs, and quantities computed from them. It does not receive the true adjacency matrix, graph-family label, structural-equation mechanism, noise scale, or an indicator identifying whether a task is synthetic, topology-transfer, or observational.

Ground-truth graphs are used only during synthetic training to supervise edge prediction and to construct the regret-aware routing targets. Once training is complete, the hybrid gate is determined exclusively from the observable 38-dimensional descriptor above.

D Neural Architecture Details

This appendix specifies the neural architecture underlying the progressive CaFuFo variants: edge-adaptive transformer aggregation, candidate-anchored observational correction, the independent data-driven expert, and full hybrid routing. Candidate and observation-derived inputs are defined in Appendices B and C, respectively.

D.1 Pairwise Feature Encoder

For every ordered pair (i, j), the eight-dimensional observation-derived vector $\boldsymbol{\phi}_{ij}$ is mapped to:

$$\mathbf{z}_{ij} = f_{\text{pair}}\left(\boldsymbol{\phi}_{ij}\right) \in \mathbb{R}^h.$$

The encoder architecture is:

$$\text{Linear}_{8\to h} \to \text{GELU} \to \text{LayerNorm}_h \to \text{Linear}_{h\to h} \to \text{GELU}\,.$$

The edge-adaptive transformer, candidate-anchor, and data-driven variants each use an encoder of this form. The architecture is shared across these variants, but the fitted parameters are learned separately.

D.2 Edge-Adaptive Transformer Aggregation

For candidate k and ordered pair (i, j), the transformer input combines the pairwise embedding with the common candidate representation:

$$\mathbf{x}_{ij}^{(k)} = \left[\mathbf{z}_{ij}, q_{ij}^{(k)}, q_{ji}^{(k)}, \mathbf{m}^{(k)}\right],$$

where:

$$\mathbf{m}^{(k)} = \left[\mathbf{e}_{\text{alg}}^{(k)}, \rho_k, f_k\right] \in \mathbb{R}^{M+2}.$$

The initial candidate token is:

$$\mathbf{t}_{ij}^{(k,0)} = f_{\text{token}}\left(\mathbf{x}_{ij}^{(k)}\right),$$

where:

$$f_{\text{token}} : \text{Linear}_{h+M+4\to h} \to \text{GELU} \to \text{LayerNorm}_h\,.$$

Thus, f_{token} is a single affine projection followed by GELU and layer normalization. For each ordered pair, the K candidate tokens are processed jointly:

$$\left(\mathbf{t}_{ij}^{(1)}, \ldots, \mathbf{t}_{ij}^{(K)}\right) = \text{TransformerEnc}\left(\mathbf{t}_{ij}^{(1,0)}, \ldots, \mathbf{t}_{ij}^{(K,0)}\right).$$

The transformer contains one pre-normalized encoder layer with four attention heads, hidden dimension 48, feed-forward dimension 96, GELU activation, and dropout 0.1. Self-attention is applied across candidate tokens for a fixed ordered node pair. It, therefore, models agreement and conflict among algorithms and resampled members; it is not applied across observations or graph nodes.

The contextualized candidate tokens are summarized by:

$$\bar{\mathbf{t}}_{ij} = \frac{1}{K}\sum_{k=1}^{K} \mathbf{t}_{ij}^{(k)}.$$

For candidate k, the reliability network receives:

$$\left[\mathbf{t}_{ij}^{(k)}, \bar{\mathbf{t}}_{ij}, \mathbf{z}_{ij}\right] \in \mathbb{R}^{3h}.$$

Its architecture is:

$$f_{\text{rel}} : \text{Linear}_{3h\to h} \to \text{GELU} \to \text{Linear}_{h\to 1}\,.$$

The resulting edge-specific reliability adjustment is bounded:

$$\delta_{ij}^{(k)} = \delta_{\max} \tanh\left(f_{\text{rel}}\left(\left[\mathbf{t}_{ij}^{(k)}, \bar{\mathbf{t}}_{ij}, \mathbf{z}_{ij}\right]\right)\right), \qquad \delta_{\max} = 0.75.$$

The final edge-specific candidate weights are:

$$w_{ij}^{(k)} = \frac{\exp\left(\log \pi_k + \delta_{ij}^{(k)}\right)}{\sum_{\ell=1}^{K} \exp\left(\log \pi_\ell + \delta_{ij}^{(\ell)}\right)},$$

and the transformer-aggregated prediction is:

$$p_{ij}^{\text{weight}} = \sum_{k=1}^{K} w_{ij}^{(k)} q_{ij}^{(k)}.$$

The final linear layer of f_{rel} is initialized to zero. Consequently, $\delta_{ij}^{(k)} = 0$ initially, and training begins from the global candidate-family prior rather than arbitrary edge-specific weights.

D.3 Candidate-Anchored Observational Correction

The candidate-anchor variant retains p_{ij}^{weight} as its reference prediction and applies a bounded observational correction. The correction network receives the pairwise embedding and transformer-pooled candidate context:

$$r_{ij} = f_{\text{corr}}\left(\left[\mathbf{z}_{ij}, \bar{\mathbf{t}}_{ij}\right]\right).$$

Its architecture is:

$$f_{\text{corr}} : \text{Linear}_{2h\to h} \to \text{GELU} \to \text{Linear}_{h\to 1}\,.$$

The applied correction is:

$$\kappa_{ij} = \sigma(\gamma)\,\kappa_{\max}\,\tanh(r_{ij}), \qquad \kappa_{\max} = 1,$$

where γ is a learned scalar initialized to -2. The candidate-anchor probability is:

$$p_{ij}^{\text{anchor}} = \sigma\left(\text{logit}\left(\text{clip}\left(p_{ij}^{\text{weight}}, 10^{-4}, 1-10^{-4}\right)\right) + \kappa_{ij}\right).$$

The final linear layer of f_{corr} is initialized to zero. The initial candidate-anchor prediction, therefore, equals the transformer-aggregated prediction. The edge-adaptive transformer-only variant uses the same candidate aggregation architecture but sets $\kappa_{\max} = 0$, disabling observational correction.

D.4 Independent Data-Driven Expert

The notation:

$$p_{ij}^{\text{data}} = f_{\text{data}}\left(\boldsymbol{\phi}_{ij}\right)$$

denotes the complete data-driven expert rather than a single feed-forward layer. This variant uses the pairwise encoder from Appendix D.1, but receives no actual candidate edge states or metadata. The implementation preserves the edge-wise backbone used during model development by replacing all candidate directional-support values with 0.5 and all candidate-metadata vectors with zero:

$$q_{ij}^{(k)} = q_{ji}^{(k)} = 0.5, \qquad \mathbf{m}^{(k)} = \mathbf{0}.$$

These neutralized slots contain no information about algorithm identity, graph density, resampling, or predicted edges. The resulting tokens are processed by a two-layer pre-normalized transformer

encoder with hidden dimension 48, four attention heads, feed-forward dimension 96, GELU activation, and dropout 0.1. Because every neutral directional-support value equals 0.5, its logit is zero and contributes no directional evidence.

An observational evidence network maps the pairwise representation to:

$$e_{ij}^{\text{data}} = f_{\text{evidence}}\left(\mathbf{z}_{ij}\right),$$

where:

$$f_{\text{evidence}} : \text{Linear}_{h\to h} \to \text{GELU} \to \text{Linear}_{h\to 1}\,.$$

A reliability network assigns weight to this observational evidence relative to the neutralized candidate slots. The residual reliability network has architecture:

$$\text{Linear}_{2h\to h} \to \text{GELU} \to \text{Linear}_{h\to 1},$$

while each neutral-slot reliability is computed using:

$$\text{Linear}_{3h\to h} \to \text{GELU} \to \text{Linear}_{h\to 1}\,.$$

All reliability values are normalized jointly with a softmax.

Since the neutral candidate logits equal zero, the resulting data-driven logit simplifies to:

$$\ell_{ij}^{\text{data}} = \omega_{ij}^{\text{obs}} e_{ij}^{\text{data}},$$

where ω_{ij}^{obs} is the reliability assigned to the observational branch. The data-driven probability is:

$$p_{ij}^{\text{data}} = \sigma\left(\ell_{ij}^{\text{data}}\right).$$

Thus, although the implementation retains neutral candidate slots for architectural consistency, the prediction is determined entirely by observation-derived pairwise features and does not depend on the generated candidate graphs.

D.5 Full Hybrid Routing

The full hybrid-routing variant combines p_{ij}^{anchor} and p_{ij}^{data}. The router receives the 38-dimensional task descriptor $\mathbf{u}$ defined in Appendix C.4. Its network is:

$$\begin{aligned} f_{\text{route}} : \quad & \text{LayerNorm}_{38} \to \text{Linear}_{38\to 24} \to \text{GELU} \to \text{Dropout}(0.2) \\ & \to \text{Linear}_{24\to 24} \to \text{GELU} \to \text{Dropout}(0.1) \to \text{Linear}_{24\to 1}\,. \end{aligned}$$

Let:

$$r = f_{\text{route}}\left(\mathbf{u}\right)$$

denote its scalar output. The data-driven gate is conservatively bounded as:

$$g_{\text{data}} = 0.90\,\sigma\left(\frac{r-1}{0.5}\right),$$

and the candidate-anchor gate is:

$$g_{\text{anchor}} = 1 - g_{\text{data}}.$$

The final edge probability is:

$$p_{ij}^{\text{hybrid}} = g_{\text{anchor}} p_{ij}^{\text{anchor}} + g_{\text{data}} p_{ij}^{\text{data}}.$$

Because the gate is task-level, the same pair of mixture weights is applied to every ordered edge within a dataset. The final router layer is initialized to zero. Hence, at initialization, $r = 0$, giving:

$$g_{\text{data}} = 0.90\,\sigma(-2) \approx 0.107, \qquad g_{\text{anchor}} \approx 0.893.$$

Training therefore begins from a candidate-dominant mixture rather than an equal combination.

The candidate-anchor and independent data-driven variants are trained before the router and held fixed during router training. The router consequently learns only how strongly to combine their predictions at the task level; it does not modify either underlying edge predictor.

E Detailed Training Objectives

E.1 Training Sequence and Data Isolation

Training follows four stages:

1. Learn the global candidate-slot weights on the synthetic training tasks and aggregate them into algorithm-family priors.
2. Train the transformer-weights, candidate-anchor, and data-only variants, using validation tasks for early stopping.
3. Freeze the selected anchor and data-only experts and train the router on the dedicated router-training tasks, using the router-validation tasks for early stopping.
4. Fit probability calibration and graph-extraction parameters using only the designated validation and calibration data.

Three model seeds are trained, and each seed is calibrated and decoded independently; task-level metrics are then averaged across seeds.

F Probability Calibration and DAG Decoding

The evaluated methods assign a probability to every ordered edge. Independent thresholding can select both $i \to j$ and $j \to i$ or produce directed cycles. We, therefore, separate probability estimation, calibration, and DAG construction.

F.1 Temperature Scaling

For an uncalibrated edge probability P_{ij}, we compute:

$$\widetilde{P}_{ij} = \sigma\left(\frac{\operatorname{logit}(P_{ij})}{\tau}\right),$$

where $\tau > 0$ is a scalar temperature. Probabilities are clipped to $[10^{-6}, 1 - 10^{-6}]$ before applying the logit transformation. For each method and model seed, τ is fitted on the dedicated calibration tasks by minimizing edgewise negative log-likelihood:

$$\tau^\star = \arg\min_{\tau>0} -\frac{1}{N_{\text{cal}}} \sum_t \sum_{i \neq j} \left[A_{ij}^{\star(t)} \log \widetilde{P}_{ij}^{(t)} + (1 - A_{ij}^{\star(t)}) \log(1 - \widetilde{P}_{ij}^{(t)})\right].$$

The scalar search is restricted to $\tau \in [0.25, 4]$.

F.2 Threshold Selection

After temperature scaling, we select a threshold η using the same designated calibration tasks. For every value in the prespecified threshold grid, the calibrated probabilities are decoded into DAGs and evaluated by directed F1. We select:

$$\eta^\star = \arg\max_{\eta} \frac{1}{N_{\text{cal}}} \sum_t \mathrm{F1}_{\text{dir}}\left(\widehat{\mathbf{A}}_\eta^{(t)}, \mathbf{A}^{\star(t)}\right).$$

If multiple thresholds obtain the same mean directed F1, the largest threshold is selected. This deterministic tie-breaking rule favours the sparser solution among tied configurations.

F.3 Order-Based DAG Decoder

Given calibrated probability matrix $\widetilde{\mathbf{P}}$, we compute the outgoing and incoming probability masses of node i:

$$s_i^{\text{out}} = \sum_{\ell \neq i} \widetilde{P}_{i\ell}, \qquad s_i^{\text{in}} = \sum_{\ell \neq i} \widetilde{P}_{\ell i}.$$

Table 5: Development performance of the observable hybrid under the three candidate pools.

Pool	Directed F1	AUPRC	Skeleton F1	nSHD
Core4	0.786	0.905	0.881	0.306
CBE-matched5	**0.813**	**0.921**	**0.919**	**0.267**
Diverse6	0.791	0.911	0.898	0.300

The node-order score is:

$$s_i = s_i^{\text{in}} - s_i^{\text{out}}.$$

Nodes with relatively stronger outgoing evidence therefore receive smaller scores and are placed earlier. We sort the nodes in ascending order of s_i, using stable sorting to resolve exact ties, and let r_i denote the resulting rank.

The decoded adjacency matrix is:

$$\widehat{A}_{ij} = \mathbb{I}\left[\widetilde{P}_{ij} \geq \eta^\star\right] \mathbb{I}\left[r_i < r_j\right], \qquad i \neq j,$$

with $\widehat{A}_{ii} = 0$.

Every retained edge points from an earlier node to a later node in a single total ordering. A directed cycle would require returning from a later node to an earlier node and is therefore impossible. The decoder consequently returns a DAG by construction. It also retains at most one direction for any node pair, even if both ordered-edge probabilities exceed the threshold.

G Experimental Protocol and Additional Results

G.1 Experimental Scope

All experiments in this study use continuous data. The primary evaluation uses the CBE-matched5 candidate pool, comprising PC, GES-BIC, ICA-LiNGAM, NOTEARS, and DAGMA. Each algorithm contributes one estimate from the complete dataset and two estimates from bootstrap-resampled datasets, yielding 15 candidate graphs per task. The two bootstrap index sets are generated once per task and shared across all constituent algorithms.

We consider three nested candidate pools:

$$\begin{aligned} \text{Core4} &= \{\text{PC}, \text{GES-BIC}, \text{NOTEARS}, \text{DAGMA}\}, \\ \text{CBE-matched5} &= \text{Core4} \cup \{\text{ICA-LiNGAM}\}, \\ \text{Diverse6} &= \text{CBE-matched5} \cup \{\text{SCORE-Stein}\}. \end{aligned}$$

Their development-set performance under the observable hybrid is reported in Table 5. CBE-matched5 was designated as the primary candidate pool because it permits a shared-budget comparison with Causal-Bayes-Ensemble Vo et al. (2026). As a development diagnostic, Table 5 also shows that this pool provides the strongest performance among the three candidate sets across directed F1, AUPRC, skeleton F1, and nSHD. Core4 and Diverse6 are retained only as candidate-pool diagnostics and do not contribute to the headline locked-test comparison. The CBE-matched5 pool was fixed before the locked-test evaluation.

G.2 Synthetic Graph and Data Generation

Synthetic tasks use Erdős–Rényi (ER) or scale-free (SF) DAGs with expected degree two. ER graphs are generated by sampling edges in an upper-triangular adjacency matrix and then randomly permuting the nodes. SF graphs are obtained from a scale-free graph, restricted to the target edge budget, oriented according to an acyclic ordering, and randomly permuted.

For every present edge, its coefficient magnitude is sampled uniformly from $[0.5, 2]$, with sign chosen uniformly from $\{-1, +1\}$. Variables are generated in topological order. For parental linear predictor

Table 6: Synthetic task partitions. Router partitions contain the 11 regimes in Table 7. All seed ranges are mutually disjoint.

Partition	Tasks	Seed range	Nodes	Samples	Mechanisms	Role
Training	30	410000–410029	8, 10, 12	500	Linear/nonlinear	Model fitting
Validation	12	420000–420011	8, 10, 12	500	Linear/nonlinear	Early stopping
Calibration	12	460000–460011	8, 10, 12	500	Linear/nonlinear	Temperature and threshold
Development	20	430000–430019	8, 10, 12	500	Linear/nonlinear	Post-selection diagnosis
Router training	66	Regime-specific blocks from 440000	8–20	150 or 500	Linear/nonlinear/post-nonlinear	Router fitting
Router validation	22	Regime-specific blocks from 470000	8–20	150 or 500	Linear/nonlinear/post-nonlinear	Router early stopping
Locked test	90	900000–902029	16 or 20	100–300	Linear/nonlinear/post-nonlinear	One-shot evaluation

Table 7: Regimes used for router supervision. ER and SF denote Erdős–Rényi and scale-free graph families, respectively.

Graph family	Mechanism	Nodes	Samples	Noise scale
ER	Linear	8, 10, 12	500	1
ER	Nonlinear	8, 10, 12	500	1
ER	Post-nonlinear	8, 10, 12	500	1
SF	Linear	8, 10, 12	500	1
SF	Nonlinear	8, 10, 12	500	1
ER	Linear	16, 20	500	1
ER	Nonlinear	16, 20	500	1
ER	Linear	8, 10, 12	500	2
ER	Nonlinear	8, 10, 12	500	2
ER	Linear	8, 10, 12	150	1
ER	Nonlinear	8, 10, 12	150	1

h_j, the three mechanisms are

$$X_j = \begin{cases} h_j + \epsilon_j, & \text{linear}, \\ \tanh(h_j) + 0.2 \sum_{i \in \mathrm{pa}(j)} X_i^2 + \epsilon_j, & \text{nonlinear}, \\ \mathrm{sign}(h_j)\sqrt{|h_j| + 10^{-6}} + \epsilon_j, & \text{post-nonlinear}, \end{cases}$$

where ϵ_j is independent Gaussian noise with the regime-specific standard deviation. Each generated variable is standardized to zero mean and unit variance. Task seeds are disjoint across all partitions.

G.3 Task Partitions

Table 6 summarizes the task partitions used for fitting, selection, calibration, diagnosis, and evaluation. The ordinary training, validation, calibration, and development partitions contain ER graphs with 8, 10, or 12 nodes, 500 observations, unit noise scale, and either a linear or nonlinear mechanism.

The router partitions contain six training tasks and two validation tasks from each of the 11 regimes in Table 7. These partitions expose the router to variation in graph family, mechanism, graph size, sample size, and noise without revealing any locked-test task.

G.4 Candidate Generation and Constituent Implementations

For each task, candidate member zero is estimated from the complete standardized dataset. Members one and two use independent bootstrap samples having the same number of rows as the original dataset. The bootstrap index sets are generated once per task and supplied to every constituent algorithm, preventing any method from receiving a systematically more favourable resample. Each candidate additionally records its algorithm identity, predicted graph density, and fraction of distinct sampled observations.

PC and GES-BIC use the `causal-learn` implementations. PC uses stable-PC with Fisher's z-test and significance level 0.05; GES uses the continuous BIC local score. ICA-LiNGAM uses `lingam.ICALiNGAM` with at most $10{,}000$ FastICA iterations. DAGMA uses the released linear DAGMA implementation with squared-error loss, $\lambda_1 = 0.03$, $T = 4$, 2,000 warm iterations, 4,000 maximum iterations, and edge threshold 0.3. NOTEARS follows the original linear augmented-Lagrangian objective with squared-error loss, $\lambda_1 = 0.03$, at most 100 outer iterations, and edge threshold 0.3. SCORE-Stein is used only in the Diverse6 development pool and is not part of the primary locked evaluation.

Table 8: Calibration-only comparison of the two CBE aggregation orders.

Aggregation order	Temp.	Threshold	Dir. F1	AUPRC	ECE
Majority then Bayes	2.892	0.46	**0.538**	**0.623**	**0.064**
Bayes then majority	4.000	0.45	0.533	0.413	0.069

Table 9: Fidelity audit for the CBE released-model adapter.

Audit item	Result
Compared candidate edge states	23,460
Released edge-state encoding mismatch	0
Maximum tensor aggregation difference	5.96×10^{-8}
Native majority-first state mismatch	0.0182
Shared candidate budget	$5 \times 3 = 15$ graphs
Training epochs / repetitions	500 / 30
Exact native reproduction	No

G.5 Training, Selection, and Seed Aggregation

Global candidate weights are fitted first on the 30 training tasks. The transformer-weights-only, candidate-anchor, and data-only variants are then trained on these tasks and early-stopped using the 12 validation tasks. The candidate anchor and data-only expert are frozen before the observable router is trained on the 66 router-training tasks and early-stopped using the 22 router-validation tasks.

We train neural variants with seeds 7, 17, and 27. Candidate and data experts are trained for at most 80 epochs with patience 12; router training uses at most 100 epochs with patience 15. Optimization uses AdamW with learning rate 2×10^{-3} and weight decay 10^{-5}.

The three neural seeds are evaluated independently. Each seed receives its own temperature and decision threshold from the common calibration tasks, after which its probability matrix is decoded into a DAG. Metrics are first averaged across seeds for each task and then averaged across tasks. Probabilities are not averaged across seeds before decoding.

G.6 CBE Released-Model Adapter

We use the released Causal-Bayes-Ensemble repository at revision: `c2b278a89cb875b4094bf6659eb092b11da02214`. The adapter imports the released Forwarder and Backwarder models, loss object, and alternating training routine. It consumes the same cached CBE-matched5 candidates as CaFuFo and never reruns a constituent algorithm. PC and GES-BIC unoriented states are preserved. The released models are trained for 500 epochs and averaged across 30 repetitions. We evaluate two aggregation orders on the calibration split:

O1: majority aggregation within each algorithm family, followed by released Bayesian aggregation across families;

O2: released Bayesian aggregation across candidates, followed by majority aggregation.

Table 8 reports the calibration-only selection. O1 is retained for all subsequent comparisons. Table 9 summarizes the fidelity audit. The adapter exactly matches the released edge-state encoder and agrees numerically with the corresponding tensor aggregation. However, the shared 15-graph budget differs from the repository's nominal 50-graphs-per-method configuration, and a 1.82% discrepancy remains between one intermediate majority-first tensor and a literal reconstruction of the complete native pipeline. We, therefore, use the label "CBE released-model adapter" rather than claiming an exact or official reproduction.

G.7 AVICI and DiBS-BGe Comparisons

AVICI uses the released pretrained checkpoint. DiBS uses 20 particles and 1,000 inference steps. Both methods use the same 12 calibration tasks and the common DAG decoder, without tuning on locked labels. Their selected calibration settings and locked-task runtimes are shown in Table 10. Runtime is measured per task on the evaluation machine and is therefore hardware-dependent.

Table 10: Baseline configurations and locked-task runtime.

Method	Temperature	Threshold	Mean time (s)
AVICI	0.773	0.56	0.67
DiBS-BGe	0.491	0.22	173.68

Table 11: Paired locked-test comparisons for the primary metrics. Differences are aligned so that positive values favour the observable hybrid.

Comparator	Metric	Mean difference	95% CI	Win rate	p-value
CBE adapter	Directed F1	0.0498	[0.0224, 0.0768]	0.611	0.0091
	AUPRC	0.0944	[0.0714, 0.1176]	0.578	$< 10^{-4}$
Candidate anchor	Directed F1	0.0028	[-0.0089, 0.0141]	0.589	0.6665
	AUPRC	0.0364	[0.0267, 0.0466]	0.600	$< 10^{-4}$
Transformer weights only	Directed F1	0.0149	[0.0017, 0.0276]	0.667	0.0296
	AUPRC	0.0507	[0.0411, 0.0605]	0.822	$< 10^{-4}$
Global candidate weights	Directed F1	0.0530	[0.0271, 0.0773]	0.633	0.0014
	AUPRC	0.0934	[0.0725, 0.1134]	0.633	$< 10^{-4}$
Uniform vote	Directed F1	0.0103	[-0.0112, 0.0315]	0.467	0.5351
	AUPRC	0.0564	[0.0362, 0.0761]	0.489	0.0105
Best validation constituent	Directed F1	0.0901	[0.0712, 0.1092]	0.733	$< 10^{-4}$
	AUPRC	0.2156	[0.1958, 0.2360]	0.933	$< 10^{-4}$
Data-only expert	Directed F1	0.3460	[0.3273, 0.3643]	0.933	$< 10^{-4}$
	AUPRC	0.3891	[0.3674, 0.4097]	0.922	$< 10^{-4}$
AVICI `scm-v0`	Directed F1	0.0313	[-0.0027, 0.0656]	0.533	0.0784
	AUPRC	0.0515	[0.0097, 0.0951]	0.533	0.0202
DiBS-BGe	Directed F1	0.0852	[0.0525, 0.1185]	0.689	0.0001
	AUPRC	0.1825	[0.1429, 0.2226]	0.800	$< 10^{-4}$

G.8 Primary-Metric Locked Comparisons

Table 11 reports paired directed-F1 and AUPRC comparisons. The observable hybrid improves both metrics over CBE, global candidate weighting, and DiBS-BGe. Its AUPRC advantage over AVICI is significant, whereas its directed-F1 advantage is not.

G.9 Secondary-Metric Locked Comparisons

Table 12 reports the complementary structural and calibration comparisons. The observable hybrid does not improve skeleton recovery over CBE and is significantly worse than CBE, global weighting, and uniform voting on nSHD and ECE.

G.10 External Benchmark Manifest

Table 13 summarizes the frozen external evaluation. Sachs is a real biological benchmark whose pooled measurements are treated as observational by our methods; intervention labels are not used. Asia, CHILD, and ALARM supply graph topologies only and are not treated as real-data evaluations.

G.11 Continuous Topology-Transfer Protocol

Asia, CHILD, and ALARM structures are downloaded from the bnlearn Bayesian-network repository. Their original conditional probability tables and observations are not used. We retain only their DAG topologies and generate continuous data using new structural equations.

Table 12: Secondary paired comparisons on the locked tasks. Differences are aligned so that positive values favour the observable hybrid, including for nSHD and ECE.

Comparator	Metric	Mean difference	95% CI	Win rate	p-value
CBE adapter	Skeleton F1	-0.0101	[-0.0283, 0.0081]	0.467	0.3062
	nSHD	-0.0989	[-0.1387,-0.0587]	0.322	$< 10^{-4}$
	ECE	-0.0317	[-0.0339,-0.0295]	0.122	$< 10^{-4}$
Candidate anchor	Skeleton F1	-0.0136	[-0.0245,-0.0032]	0.467	0.0295
	nSHD	-0.0566	[-0.0825,-0.0310]	0.500	0.0094
	ECE	-0.0268	[-0.0288,-0.0249]	0.378	$< 10^{-4}$
Transformer weights only	Skeleton F1	0.0030	[-0.0089,0.0147]	0.589	0.6404
	nSHD	-0.0498	[-0.0778,-0.0230]	0.533	0.0330
	ECE	-0.0274	[-0.0295,-0.0253]	0.378	$< 10^{-4}$
Global candidate weights	Skeleton F1	0.0002	[-0.0199,0.0189]	0.489	0.9871
	nSHD	-0.0776	[-0.1194,-0.0405]	0.378	0.0008
	ECE	-0.0389	[-0.0414,-0.0365]	0.022	$< 10^{-4}$
Uniform vote	Skeleton F1	-0.0592	[-0.0731,-0.0449]	0.178	$< 10^{-4}$
	nSHD	-0.1437	[-0.1762,-0.1117]	0.167	$< 10^{-4}$
	ECE	-0.0338	[-0.0363,-0.0314]	0.122	$< 10^{-4}$

Table 13: Frozen external evaluation manifest.

Benchmark	Rows	Nodes	Edges	Tasks
Sachs	7,466	11	17	1
Asia	1,000/task	8	8	10
CHILD	1,000/task	20	25	10
ALARM	1,000/task	37	46	10

For each topology, five linear and five nonlinear datasets are generated, each with $1{,}000$ observations and unit Gaussian noise. Edge magnitudes are sampled uniformly from $[0.5, 1.5]$, with random signs. The linear signal is normalized by the square root of the number of parents. For the nonlinear condition, the signal is:

$$\tanh(h_j) + 0.30\,\mathrm{mean}_{i\in \mathrm{pa}(j)}\sin(w_{ij}X_i).$$

Variables are standardized after simulation.

G.12 Topology-Transfer Results by Network

Table 14 reports directed F1 and AUPRC by topology and mechanism. The macro advantage of the observable hybrid does not imply that it wins every setting: uniform voting performs especially strongly on CHILD, whereas the observable hybrid is strongest on both ALARM conditions.

Relative to the candidate anchor, the observable hybrid improves directed F1 by 0.0173 (95% CI $[0.0057, 0.0315]$, $p = 0.0049$). Its AUPRC difference is only 0.0002 and is not significant ($p = 0.594$). The candidate anchor has higher skeleton F1 by 0.0304 ($p < 10^{-4}$).

For completeness, Table 15 reports the matched full-data GES-BIC and PC baselines.

G.13 Sachs Data and Reference Graph

The Sachs protein-signalling benchmark is loaded using `cdt.data.load_dataset(''sachs'')` from CDT version 0.6.0 (Sachs et al., 2005). The returned data contain $7{,}466$ rows and 11 measured variables. The pooled measurements are treated as observational, and intervention labels are not used. Any missing value would be replaced by its column median, after which all columns are standardized. The CDT reference graph contains 18 directed edges and the cycle:

$$\mathrm{plcg} \to \mathrm{PIP2} \to \mathrm{PIP3} \to \mathrm{plcg}.$$

Table 14: Directed F1 and AUPRC by unseen topology and mechanism. Each entry is "directed F1 / AUPRC" over five datasets.

Topology	Mechanism	Observable hybrid	Candidate anchor	Global weights	Uniform vote
ALARM	Linear	0.795 / 0.931	0.779 / 0.931	0.753 / 0.908	0.592 / 0.688
ALARM	Nonlinear	0.719 / 0.858	0.716 / 0.858	0.699 / 0.831	0.683 / 0.763
Asia	Linear	0.540 / 0.848	0.544 / 0.848	0.532 / 0.840	0.507 / 0.611
Asia	Nonlinear	0.677 / 0.915	0.604 / 0.916	0.606 / 0.910	0.678 / 0.815
CHILD	Linear	0.691 / 0.799	0.685 / 0.798	0.765 / 0.816	0.772 / 0.843
CHILD	Nonlinear	0.632 / 0.739	0.622 / 0.738	0.660 / 0.734	0.777 / 0.902
Macro average	—	0.676 / 0.848	0.658 / 0.848	0.669 / 0.840	0.668 / 0.770

Table 15: Matched full-data constituent baselines on the 30 topology-transfer tasks.

Method	Dir. F1	AUPRC	Skel. F1	nSHD	ECE	Edge ratio
GES-BIC	0.633	0.602	0.855	0.559	0.045	1.133
PC	0.557	0.542	0.882	0.546	0.031	0.944

Under the frozen external-data protocol, we remove the cycle-closing edge PIP2 $\rightarrow$ PIP3, producing a 17-edge DAG.

No Sachs-specific model fitting, early stopping, temperature calibration, or threshold selection is performed. Because Sachs supplies one reference graph, all comparisons are descriptive rather than inferential.

G.14 Complete Sachs Results

Table 16 provides the complete metric set. Global candidate weighting and matched full-data PC tie at 0.439 directed F1. The candidate anchor is the strongest edge-adaptive framework variant, whereas the observable hybrid and CBE adapter obtain essentially identical directed F1 after rounding. The data-only expert transfers poorly to Sachs, yet the observable router assigns it nonzero influence. Candidate-dominated variants, therefore, outperform the observable hybrid in this setting.

G.15 Candidate-Budget and Resampling Ablation

The primary CaFuFo configuration receives 15 candidate graphs: each of PC, GES-BIC, ICA-LiNGAM, NOTEARS, and DAGMA contributes one graph estimated from the complete dataset and two graphs estimated from bootstrap-resampled datasets. We conduct a matched ablation to determine whether the performance of CaFuFo is primarily explained by this enlarged candidate budget. We construct a five-candidate configuration containing only the full-data graph from each constituent algorithm. These five graphs are the same full-data candidates included in the corresponding 15-candidate pool; no causal-discovery algorithm is rerun. We then train the five-candidate framework independently using the same training, validation, router-training, and calibration tasks as the primary model. The two configurations use identical model seeds, architectures, optimization settings, early-

Table 16: Complete frozen evaluation on Sachs. "Aggregate" denotes one full-data candidate and two bootstrap candidates.

Method	Directed F1 ↑	AUPRC ↑	Skeleton F1 ↑	nSHD ↓	ECE ↓	Edge ratio
Global candidate weights	**0.439**	**0.418**	0.439	1.353	0.183	1.412
Matched full-data PC	**0.439**	0.309	0.439	1.353	0.170	1.412
PC three-member aggregate	0.410	0.311	0.410	1.353	0.180	1.294
Candidate anchor	0.397	0.303	0.397	1.157	0.145	0.941
Transformer weights only	0.355	0.344	0.403	**1.020**	0.128	0.647
Uniform vote	0.324	0.338	0.378	1.412	**0.105**	1.176
GES-BIC three-member aggregate	0.304	0.211	**0.522**	1.588	0.264	1.706
Observable hybrid	0.258	0.260	0.325	1.275	0.187	0.804
CBE released-model adapter	0.258	0.285	0.323	1.294	0.163	0.824
Data-only expert	0.156	0.195	0.322	1.373	0.229	0.804

Table 17: Matched candidate-budget ablation over the 90 locked tasks. The five-candidate configuration uses one full-data graph per algorithm, whereas the 15-candidate configuration additionally uses two bootstrap graphs per algorithm. Higher is better for directed F1, AUPRC, and skeleton F1; lower is better for nSHD and ECE. Bold denotes the better result within each method pair.

Method	Candidate budget	Directed F1 ↑	AUPRC ↑	Skeleton F1 ↑	nSHD ↓	ECE ↓	Edge ratio
Observable hybrid	15: full + two bootstrap	**0.661**	**0.759**	**0.767**	**0.613**	**0.066**	1.063
	5: full data only	0.647	0.756	0.748	0.741	0.067	1.238
Candidate anchor	15: full + two bootstrap	**0.659**	**0.722**	0.781	0.556	0.039	0.989
	5: full data only	0.647	0.693	**0.794**	**0.544**	**0.036**	0.969
Transformer weights only	15: full + two bootstrap	**0.647**	**0.708**	0.764	**0.563**	0.038	0.939
	5: full data only	0.632	0.663	**0.765**	0.587	**0.035**	0.971
Global candidate weights	15: full + two bootstrap	**0.608**	**0.665**	**0.767**	**0.536**	**0.027**	0.756
	5: full data only	0.597	0.646	0.754	0.554	0.033	0.746
Uniform vote	15: full + two bootstrap	**0.651**	**0.702**	**0.826**	**0.469**	**0.032**	0.851
	5: full data only	0.585	0.679	0.716	0.546	0.035	0.642

Table 18: Paired effect of increasing the candidate budget from five to 15 graphs. Each entry reports the mean aligned improvement followed by the paired randomisation-test p-value in parentheses. Positive values favour the 15-candidate configuration; signs for nSHD and ECE are reversed so that positive values consistently denote improvement.

Method	Δ Directed F1	Δ AUPRC	Δ Skeleton F1	Δ nSHD	Δ ECE
Observable hybrid	$+0.014$ (0.067)	$+0.002$ (0.757)	$+0.019$ (0.0002)	$+0.128$ $(< 10^{-4})$	$+0.002$ (0.260)
Candidate anchor	$+0.011$ (0.291)	$+0.029$ (0.0005)	-0.014 (0.027)	-0.012 (0.410)	-0.003 (0.027)
Transformer weights only	$+0.015$ (0.146)	$+0.045$ $(< 10^{-4})$	-0.001 (0.917)	$+0.024$ (0.115)	-0.003 (0.020)
Global candidate weights	$+0.011$ (0.142)	$+0.019$ (0.0002)	$+0.013$ (0.029)	$+0.019$ (0.036)	$+0.006$ $(< 10^{-4})$
Uniform vote	$+0.067$ $(< 10^{-4})$	$+0.023$ (0.0009)	$+0.111$ $(< 10^{-4})$	$+0.077$ $(< 10^{-4})$	$+0.003$ (0.0008)

stopping rules, calibration procedure, and DAG decoder. The five-candidate models are, therefore, retrained for their intended input budget rather than being created by deleting candidates from an already trained 15-candidate model.

Both configurations are evaluated on the same 90 locked tasks. Table 17 shows that the five-candidate observable hybrid remains competitive, reaching 0.647 directed F1 and 0.756 AUPRC. Adding the ten bootstrap candidates raises directed F1 to 0.661, but the paired improvement of 0.014 is not conventionally significant (95% CI $= [-0.001, 0.029]$, $p = 0.067$). The corresponding AUPRC improvement is only 0.002 ($p = 0.757$). Thus, the primary hybrid's directed-edge ranking cannot be attributed simply to receiving three times as many candidate graphs. The bootstrap candidates have a clearer effect on structural stability. For the observable hybrid, skeleton F1 improves by 0.019 ($p = 0.0002$), while aligned nSHD improves by 0.128 ($p < 10^{-4}$). Its edge ratio also decreases from 1.238 to 1.063, suggesting that the additional candidates primarily help the model suppress unsupported edges and control graph density. Calibration is essentially unchanged. The effect of resampling differs across aggregation strategies. Transformer weights only and the candidate anchor obtain significant AUPRC gains from the enlarged candidate pool, although neither shows a significant directed-F1 improvement. Global candidate weighting improves AUPRC, skeleton F1, nSHD, and ECE, but again not directed F1. The largest and most consistent benefit occurs for uniform voting, which improves directed F1 by 0.067, skeleton F1 by 0.111, and aligned nSHD by 0.077, with $p < 10^{-4}$ for all three comparisons.

These results indicate that bootstrap resampling is particularly valuable for stabilizing simple candidate voting. The learned CAFUFO variants rely less strongly on the enlarged candidate budget because they already estimate candidate reliability from the full-data graphs and observable task information. Nevertheless, the additional bootstrap candidates improve the complete hybrid's structural recovery and substantially reduce its tendency to overpredict edges. Resampling should, therefore, be viewed as a structural-stabilization component of CAFUFO, rather than the sole source of its directed-recovery performance.