TEST TIME TRAINING FOR SUPERVISED CAUSAL LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Supervised Causal Learning (SCL) has shown promise in causal discovery by framing it as a supervised learning problem. However, it suffers from significant out-of-distribution generalization challenges. We reveal three limitations of previous SCL practices: a significant performance gap between synthetic benchmarks and real-world data, fragility to distribution shifts, and failure in compositional generalization, collectively questioning its real-world applicability. To address this, we propose Test-Time Training for Supervised Causal Learning (TTT-SCL), a framework that dynamically generates training sets aligned with the current test dataset, on which a new supervised learner is trained from random initialization. We demonstrate the correlation between TTT-SCL and score-based methods, and design an efficient module for generating training sets based on the classic scoring function. Experiments on synthetic, pseudo-real, and real-world datasets demonstrate competitive performance and improvements over the compared baselines in challenging settings.

1 INTRODUCTION

Causal discovery aims to infer causal relationships from observational data (Pearl, 2009; Spirtes et al., 2000). While traditionally approached as an unsupervised problem (Fig. 1 (A)), Supervised Causal Learning (SCL) has recently emerged as a promising alternative (Dai et al., 2023; Lorch et al., 2022; Ke et al., 2022; Zhang et al., 2025). SCL treats causal discovery as a supervised learning task: a model is trained on a set of synthetic causal instances, each comprising a causal graph and its corresponding sampled dataset, and learns to map datasets to their underlying causal graphs (Fig. 1 (B)).

A pivotal factor for SCL's success is the design of its training set. What properties should these synthetic training sets possess to ensure the model performs well on a real-world, unseen test instance? Two complementary principles guide this design: **diversity** and **concentration**. Diversity seeks broad coverage of possible causal models by varying key components such as graph structures, mechanisms, and noise characteristics, thereby encouraging generalization. Concentration, in contrast, aims to align the training set closely with the specific characteristics of the test domain of interest.

Current SCL methods heavily prioritize diversity, pre-training on large, static training sets generated from wide-ranging synthetic distributions (Lorch et al., 2022; Ke et al., 2022). However, we demonstrate that this paradigm suffers from critical limitations. Through experiments, we observe a notable generalization gap where strong performance on synthetic benchmarks fails to translate to real-world data, thereby questioning the practical utility of existing SCL approaches. Within the synthetic domain, we further find that these models are fragile under distribution shifts, showing significant performance degradation when the test instance differs from the training set in graph structure, causal mechanism, or noise distribution. More fundamentally, they fail to generalize compositionally; even when trained on all individual components, they cannot handle novel combinations of these components.

These findings motivate a paradigm shift from diversity to concentration. We argue that robust generalization requires moving beyond a single, fixed training set and instead dynamically adapting to each test instance. To this end, we introduce a Test-Time Training for Supervised Causal Learning (TTT-SCL) framework. The core idea is that for a given test dataset, we dynamically generate a new,

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

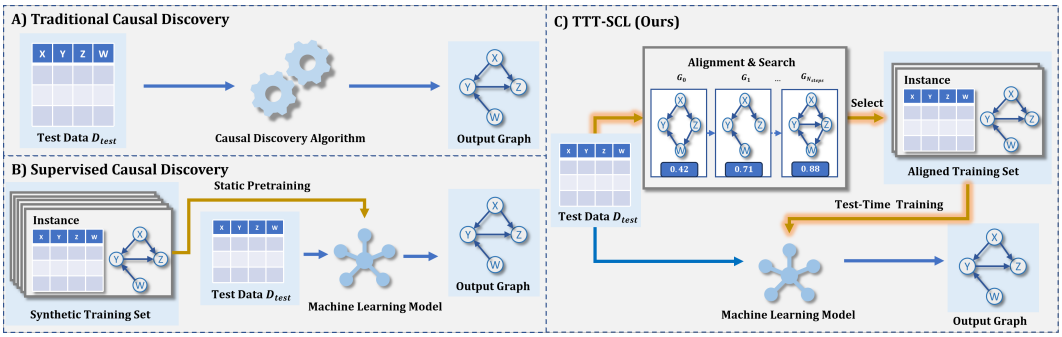

Figure 1: Comparison of classical causal discovery, static supervised causal learning, and TTT-SCL. TTT-SCL retains the final K graph states from score-guided refinement, fits SIM to generate graph-labeled training pairs, and trains a new learner for the observed dataset.

customized training set that is explicitly aligned with its distribution, train a specialized SCL model, and utilize this model to infer the causal graph (Fig. 1 (C)). TTT-SCL shares score-guided graph search with classical score-based methods, but uses the retained graphs to construct instance-specific supervised training data rather than returning a single highest-scoring graph.

Our main contributions are:

- We reveal three limitations of static SCL pre-training: a significant performance gap between synthetic benchmarks and real-world data, fragility to distribution shifts, and failure in compositional generalization, collectively questioning its real-world applicability.
- We introduce the TTT-SCL framework, enabling dynamic generation of aligned training sets at test time. We develop a score-guided procedure for constructing instance-specific training pairs and empirically evaluate the contribution of the supervised learner.
- Our experiments confirm that TTT-SCL achieves good performance across both synthetic, pseudo-real and real-world datasets.

2 Background

We begin by formalizing the core components of causal learning. A Structural Causal Model (SCM) consists of three key elements: causal graph, causal mechanisms, and noise distributions (Pearl, 2009; Peters et al., 2017). Specifically:

- Causal Graph: Let $G = (V, E)$ be a Directed Acyclic Graph (DAG) with vertex set $V = \{X_1, \ldots, X_d\}$ and edge set $E \subseteq V \times V$, where d is the number of variables. The adjacency matrix $A \in \{0,1\}^{d\times d}$ encodes edge relationships where $A_{ij} = 1$ iff $X_i \to X_j \in E$.
- Causal mechanisms and noise: Each variable X_i is generated by a causal mechanism and exogenous noise, following the Structural Causal Model (SCM) framework (Pearl, 2009). The data-generating process is characterized by the structural equations:

$$X_i := f_i(\mathbf{Pa}_G(X_i), \varepsilon_i), \tag{1}$$

  where $\mathbf{Pa}_G(X_i)$ denotes parents of X_i in G, $f_i : \mathbb{R}^{|\mathbf{Pa}_G(X_i)|} \times \mathcal{E}_i \to \mathbb{R}$ is the causal mechanism, and $\varepsilon_i \sim P_{\varepsilon_i}$ is exogenous noise from distribution P_{ε_i}. The full SCM is thus characterized by the tuple $(G, \{f_i\}_{i=1}^d, \{\varepsilon_i\}_{i=1}^d)$, which comprehensively captures the causal structure, functional relationships, and exogenous noise.

In supervised causal learning, we work with **causal instances**. A causal instance is defined by a graph G and a dataset D containing n observations $\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}\} \in \mathbb{R}^{n\times d}$, generated from the SCM $(G, \{f_i\}_{i=1}^d, \{\varepsilon_i\}_{i=1}^d)$. The **training set** comprises K such instances, denoted as $\{(D_{train}^k, G_{train}^k)\}_{k=1}^K$, where each D_{train}^k is generated from its corresponding G_{train}^k. Similarly,

at test time, we are given a single **test instance** (D_{test}, G_{test}), where D_{test} is observed but G_{test} is unknown. To avoid notation clutter, we adopt the following conventions: indices i, j refer to variable/node indices within a graph, and subscripts "train" and "test" distinguish between training and test entities.

Causal discovery aims to estimate the causal graph G_{test} from D_{test} using a model or algorithm M. Supervised causal learning (SCL) frames this as a supervised learning problem, where a model (typically a neural network) is trained on synthetic causal instances to learn a mapping from observational data to graph structures. Formally, the SCL objective is to learn:

$$\mathcal{M} : \mathbb{R}^{n \times d} \to \{0, 1\}^{d \times d}, \tag{2}$$

which maps an input data matrix (e.g., D_{test}) to an output adjacency matrix (representing G_{test}). The model is trained on synthetic pairs $\{(D^k_{train}, G^k_{train})\}_{k=1}^K$.

Previous SCL methods rely on training with synthetic data, where the generative distribution is explicitly controlled along three dimensions consistent with the SCM framework: graph structure, causal mechanisms, and noise distributions (Lorch et al., 2022; Ke et al., 2022; Froehlich & Koeppl, 2024).

3 Out-of-distribution Challenges for SCL

Supervised causal learning (SCL) relies almost entirely on synthetic data for training because real-world causal graphs are rarely available, which naturally raises the concern that performance measured on synthetic benchmarks may not faithfully reflect performance on real-world data. Our results on the Sachs dataset confirm this concern, as competitive performance on synthetic benchmarks does not translate into comparable accuracy on real-world data (**Issue 1**). A closer look within the synthetic domain further reveals why static pre-training breaks down. The training distribution is jointly characterized by three dimensions, namely graph structure, causal mechanism, and noise distribution, and a mismatch in any single dimension between training and test already causes substantial performance degradation, exposing the fragility of SCL under distribution shifts (**Issue 2**). More fundamentally, even when the model has encountered all individual settings across these dimensions during training, it still fails on unseen combinations of them, demonstrating a clear failure of compositional generalization (**Issue 3**). The experiments below illustrate these three limitations.

3.1 Experiment Setup

Datasets. To comprehensively evaluate generalization, we use both synthetic benchmarks, pseudo-real and real-world datasets.

- **Synthetic data:** We generate test instances from a factorial combination of mechanism, graph and noise distribution. We use three mechanism classes: Linear, Random Fourier Features (RFF) (Rahimi & Recht, 2007), and Chebyshev polynomials (Froehlich & Koeppl, 2024). We use two random graph models: Erdos-Renyi (ER) and Scale-Free (SF) (Gilbert, 1959; Barabási, 2009). Gaussian noise is used for RFF and Chebyshev mechanisms, while Uniform noise is used for Linear mechanisms to ensure identifiability. This yields six test settings: RFF_G_ER, RFF_G_SF, Linear_U_ER, Linear_U_SF, Chebyshev_G_ER, and Chebyshev_G_SF.
- **Pseudo-real data**: We also incorporate pseudo-real datasets generated by the SynTReN generator (Van den Bulcke et al., 2006). This generator is specifically designed to simulate synthetic transcriptional regulatory networks with biologically plausible structures and parameters, producing gene expression data that closely resembles experimental microarray data.
- **Real-world data:** We use the Sachs dataset (Sachs et al., 2005), a well-established benchmark in causal discovery. It contains 853 measurements of 11 proteins and a consensus causal graph derived from biological knowledge.

Model & Training setting.: We mainly use AVICI as the model backbone (Lorch et al., 2022), a DNN-based architecture which is currently widely followed by the community and open-sourced. To assess different generalization aspects, we compare several training settings, more detailed configurations can be found in Appendix A:

- **i.i.d**: The training data and test data are exactly the same distribution.
- **Graph/Noise/Mechanism shift**: The mechanism/graph/noise of the training data is different from that of the test data, but the other two distributions are the same.
- **Component-mixed**: This training setup contains all individual components (mechanisms, graph types, noise distributions) seen in isolation during training, but crucially excludes the specific combinations present in the test instances. This tests whether the model can perform *compositional generalization* by recombining learned components, rather than merely memorizing training configurations.
- **AVICI (scm-v0)**: This model was trained on SCM data simulated from a large variety of graph models with up to 100 nodes, both linear and nonlinear causal mechanisms, and homogeneous and heterogeneous additive noise from Gaussian, Laplace, and Cauchy distributions. It can be considered one of the strongest model of open source under the SCL paradigm [1].

Table 1: Divergent generalization patterns. Strong synthetic performance does not guarantee effectiveness on real-world data. Results are presented as AUROC (standard deviation).

	RFF_G	Linear_U	Chebyshev_G	Sachs	Syntren
PC	61.1 (4.9)	60.9 (4.7)	59.8 (6.6)	67.1	58.1
AVICI (scm-v0)	97.8 (1.3)	75.6 (13.8)	81.7 (10.5)	62.3	65.4

3.2 Limitations of Current SCL Paradigms

Our experimental results validate the three issues outlined above, collectively exposing the limitations of static pre-training in SCL.

Issue 1. The results in Table 1 question the value of synthetic benchmarks by demonstrating that strong synthetic performance fails to guarantee effectiveness on real-world data. Here, we merge the dimensions of the graph and analyze more from the perspective of the mechanism. While AVICI (scm-v0) excels on synthetic data similar to its training distribution (e.g., RFF_G, 97.8), its performance collapses on the real-world Sachs dataset (62.3). In contrast, traditional methods like PC maintain consistent, albeit lower, performance across domains. This divergence reveals that SCL models overfit to the artifacts of their synthetic training set, lacking the cross-domain consistency required for real-world applicability.

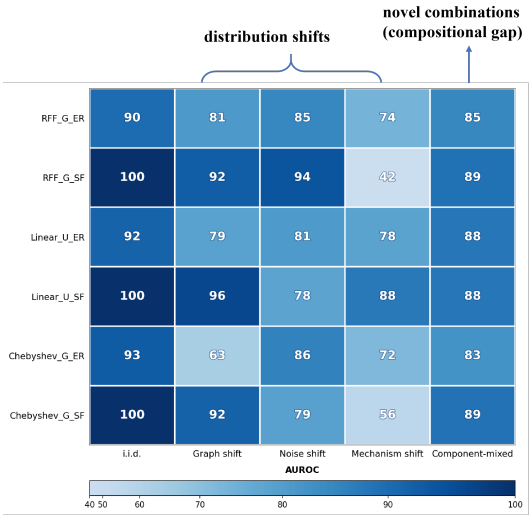

Figure 2: Two limitations of static SCL: fragility to distribution shifts and failure in compositional generalization. Each column represents a test setting, and each row shows the specific distribution of test data. The specific training distribution for the 5 test settings can be found in Appendix A.

Issue 2. The results in Fig. 2 demonstrate that distribution shifts across all three dimensions (graph structure, causal mechanism, and noise distribution) significantly degrade SCL performance. Models struggle when the test-time graph structure ("Graph shift" compared to "i.i.d"), causal mechanism ("Mechanism shift" compared to "i.i.d"), or noise distribution ("Noise shift" compared to "i.i.d") differs categorically from those seen during training. While performance drops are observed in all cases, "Mechanism shifts" emerge as particularly damaging, underscoring the profound impact of the underlying mechanism functional form on model generalization.

[1] https://github.com/larslorch/avici

Issue 3. Even when trained on data containing all individual components, the model still exhibits performance drop on unseen combinations of these components, as seen when comparing "Component-mixed" to "i.i.d" in Fig. 2. This compositional failure indicates that SCL models memorize specific (G, f, ε) configurations rather than learning a modular understanding of causal factors.

In summary, the dual failure of fragility under distribution shifts and inconsistency across domains fundamentally undermines the static pre-training paradigm. These limitations are not artifacts of a specific architecture, as validated by consistent failure patterns using the SiCL backbone (Appendix B). The results compellingly argue that robust causal discovery requires a shift from static, diversity-seeking pre-training to dynamic, test-time adaptation.

4 Test-Time Training for Supervised Causal Learning

Given an observed test dataset D_{test}, TTT-SCL infers its underlying causal graph by training a supervised model at test time on data that are causally aligned with D_{test}, as illustrated in Fig. 1(C). For each observed test dataset, we train a new learner from random initialization; we do not fine-tune a pretrained checkpoint. The same unlabeled dataset is intentionally used for scoring, SIM fitting, and final prediction, while its true graph is never used for training or model selection. The workflow proceeds as follows: a training set tailored to D_{test} is first generated under the guidance of a score function, and a supervised causal learning model is then trained on this set and applied to D_{test} for inference.

In our work, the design and acquisition of the test-time training set constitutes the core technical contribution, while the supervised causal learning module can leverage any existing architecture. Therefore, we focus on the generation of the training set in the following, and use the score as a bridge to analyze the relationship between TTT-SCL and classical score-based methods.

To generate a training set that is causally aligned with D_{test}, we need a principled measure of alignment quality, which motivates our adoption of a score function that evaluates how well a candidate graph G fits the observed data D_{test}. Formally, we adopt a score

$$\mathrm{score}(G, D_{test}) = \mathrm{AD}(G, D_{test}) - \|\mathbf{A}_G\|_0, \tag{3}$$

where the alignment-of-distribution (AD) term is defined as the average log-likelihood of the variables under the mechanisms induced by G:

$$\mathrm{AD}(G, D_{test}) \coloneqq \frac{1}{d}\sum_{i=1}^{d}\frac{1}{n}\sum_{t=1}^{n}\log p\big(x_i^{(t)} \mid f_i^G(\mathbf{x}_{\mathrm{Pa}_G(i)}^{(t)})\big). \tag{4}$$

Here f_i^G denotes the causal mechanism of variable X_i given its parents in G. For each candidate graph, Structure-Induced Mechanism (SIM) fits the node-wise mechanisms to the observed test data. Details of mechanism fitting and sampling are provided in Appendix C. The term $\|\mathbf{A}_G\|_0$ counts the number of edges and penalizes graph complexity, in the spirit of classical information criteria. Guided by this score, we construct a collection of candidate graphs.

However, finding these graphs is difficult as exhaustively searching the entire DAG space is intractable. To solve this problem, we propose an informed construction approach which combines good initializations with guided refinement. Concretely, it proceeds in three stages:

1. **Seed Initialization.** We start from an initial graph G^0 obtained by applying a traditional causal discovery method (e.g., PC, NOTEARS) on D_{test}. Note that the initial graph can also be a random DAG, while using a better initial graph provides a useful prior and accelerates the whole process.
2. **Iterative Stochastic Graph Refinement.** From the current graph G_{curr}, we iteratively propose local modifications to the graph and obtain G_{cand}. The possible modifications include single-edge additions, deletions, and reversals do not break the acyclicity. After that, the candidate G_{cand} is evaluated using the joint score function in Eq. (3) and is accepted with probability

$$\alpha = \min\{1, \exp[\mathrm{score}(G_{cand}, D_{test}) - \mathrm{score}(G_{curr}, D_{test})]\}\,.$$

Algorithm 1 Test-Time Training for Supervised Causal Learning (TTT-SCL)

Input:
Test dataset $D_{\text{test}} \in \mathbb{R}^{n \times d}$, Number of refinement steps N_{steps}, number of training graphs K, and synthetic observations per pair n_{syn}.

Output: Predicted causal graph $G_{\text{pred}} \in \{0,1\}^{d \times d}$.

$G^0 \leftarrow \text{InitialGraph}(D_{\text{test}})$ ▷ e.g., NOTEARS, or a random DAG
Initialize graph list $\mathcal{G} \leftarrow []$
$G_{\text{curr}} \leftarrow G^0$
for $t = 1$ **to** N_{steps} **do**
  $G_{\text{cand}} \leftarrow \text{Local}(G_{\text{curr}})$ ▷ Add, delete, or reverse an edge while preserving acyclicity
  $s_{\text{curr}} \leftarrow \text{Score}(G_{\text{curr}}, D_{\text{test}})$
  $s_{\text{cand}} \leftarrow \text{Score}(G_{\text{cand}}, D_{\text{test}})$
  $\alpha \leftarrow \min\big(1, \exp(s_{\text{cand}} - s_{\text{curr}})\big)$
  if $\text{random}(0,1) < \alpha$ **then**
    $G_{\text{curr}} \leftarrow G_{\text{cand}}$
  end if
  if $t > N_{\text{steps}} - K$ **then**
    $\mathcal{G}.\text{append}(G_{\text{curr}})$ ▷ Collect the last K graphs
  end if
end for
Initialize training set $\mathcal{T} \leftarrow \emptyset$
for each $G^k_{train} \in \mathcal{G}$ **do**
  $\{f_i\}_{i=1}^d \leftarrow \text{SIM}(G^k_{train}, D_{\text{test}})$ ▷ Regress mechanisms using D_{test} via SIM
  $D^k_{train} \leftarrow \text{Sample}(G^k_{train}, \{f_i\}_{i=1}^d, n_{\text{syn}})$
  $\mathcal{T} \leftarrow \mathcal{T} \cup \{(D^k_{train}, G^k_{train})\}$
end for
Train a newly initialized supervised causal learner $\mathcal{M}$ on $\mathcal{T}$ ▷ e.g., AVICI
$G_{\text{pred}} \leftarrow \mathcal{M}(D_{\text{test}})$
return G_{pred}

An improving proposal is accepted with probability one, and a worsening proposal with exponentially decreasing probability. The rule remains valid for negative scores and is invariant to additive shifts. We retain the graph states from the final K refinement iterations.

3. **Training Data Generation.** For each graph G^k_{train}, we regress its mechanism with D_{test} via SIM, and then forward-sample a synthetic dataset D^k_{train} based on the regressed mechanism. All causal instances $(D^k_{train}, G^k_{train})$ then assemble the final training sets $\{(D^k_{train}, G^k_{train})\}_{k=1}^K$.

The complete TTT-SCL workflow is formalized in Algorithm 1. By combining score and stochastic refinement, this approach realizes an efficient and directed method for searching the graph space at test time. All hyperparameter details are described in the Appendix A. A diagram illustrating the workflow of training sets generation can be found in Appendix E.

Classical score-based causal discovery methods (Chickering, 2002; Zheng et al., 2018; Lachapelle et al., 2020) find an optimal graph based on the score, while TTT-SCL gives the score a richer meaning and more flexible usage. **First**, we move from selecting a single optimal graph to building a set of candidate graphs, thus replacing a point estimate with a collection that is more informative for learning. Under finite samples, multiple graphs are often plausible due to statistical uncertainty and score approximation error. **Second**, each retained graph is paired with synthetic data generated by SIM, providing graph-conditioned training instances tailored to the observed dataset. **Third**, we do not use the score as a final decision rule; rather, we train a supervised model on these instance-specific training sets to further optimize predictive accuracy. This learned model can combine evidence across the entire candidate set and exploit patterns that are not fully captured by the scalar score alone. In Sec 5.3, our experimental results confirm the performance improvement brought about by supervised learning.

5 EXPERIMENTS

In this section, we compare the performance of TTT-SCL with multiple baseline methods on various synthetic data, pseudo-real data and real data. These datasets are consistent with the content of Section 3.1. The code we used for our experiments is available at `https://anonymous.4open.science/r/TTTSCL-FEB2/`.

Baselines: We compare against traditional causal discovery methods PC (Spirtes et al., 2000), GES (Chickering, 2002), NOTEARS (Zheng et al., 2018), RESIT (Peters et al., 2014), SCORE (Rolland et al., 2022), NoGAM (Montagna et al., 2023) and AVICI (Lorch et al., 2022), a DNN-based SCL method which is currently widely followed by the community and open source. We use the open-source pre-trained AVICI (scm-v0) model, which is trained on a vast mixture of synthetic data and represents the strongest publicly available SCL baseline.

We additionally compare with GraN-DAG (Lachapelle et al., 2020), DAG-GNN (Yu et al., 2019), and CGNN (Goudet et al., 2017), which optimize separately for each observed dataset. We use fixed public configurations without tuning on the true graph. CGNN uses a skeleton estimated by PC from the same observations, not the true skeleton. The complete results and evaluation details are in Appendix L.

Our Method (TTT-SCL): For our TTT-SCL approach, we set the number of dynamically generated training graphs to $K = 200$. The number of variables d is 10 for synthetic data, 11 for Sachs and 20 for Syntren. Each generated pair contains $n_{\text{syn}} = 200$ observations, except Sachs, where $n_{\text{syn}} = 853$. We distinguish this count from the number n of observed test rows. We evaluate two variants of our method: TTT-SCL (random) which initializes the seed graph with a random DAG, and TTT-SCL (Notears) which uses a graph estimated from D_{test} by the NOTEARS algorithm as a smarter starting point.

Evaluation metrics: We use multiple metrics to evaluate the predicted graphs, including Area Under the Receiver Operating Curve (AUROC), Area Under the Precision-Recall Curve (AUPRC), F1 score and Accuracy (ACC). In the main text, we primarily report **AUROC** for edge prediction to succinctly explore the impact of training data quality on model performance.

5.1 THE PERFORMANCE OF TTT-SCL

Table 2: TTT-SCL performance on synthetic, real and pseudo-real datasets. Results are presented as AUROC (standard deviation).

	RFF_G	Linear_U	Chebyshev_G	Sachs	Syntren
PC	61.1 (4.9)	60.9 (4.7)	59.8 (6.6)	67.1	58.1
GES	66.0 (10.6)	69.0 (10.8)	59.6 (5.9)	61.8	36.8
Notears	80.5 (4.0)	82.0 (4.6)	52.2 (3.5)	61.8	49.8
RESIT	54.3 (5.4)	54.1 (5.2)	49.8 (4.7)	62.3	64.6
SCORE	86.9 (3.2)	82.2 (18.7)	69.2 (7.6)	64.9	41.0
NoGAM	87.6 (2.9)	79.2 (18.6)	72.3 (6.4)	64.9	41.0
AVICI (scm-v0)	**97.8** (1.3)	75.6 (13.8)	81.7 (10.5)	62.3	65.4
TTT-SCL (random)	88.4 (7.0)	82.3 (7.0)	79.6 (6.7)	58.6	72.0
TTT-SCL (Notears)	91.8 (3.1)	**86.3** (4.4)	**83.0** (8.7)	**78.9**	**80.1**

The results are summarized in Table 2. TTT-SCL performs strongly across the evaluated settings, particularly on Sachs and SynTReN. AVICI (scm-v0), pretrained on RFF-like mechanisms, achieves higher AUROC on RFF_G. TTT-SCL (Notears) achieves the highest AUROC among the methods in this table on Linear_U, Chebyshev_G, Sachs, and SynTReN.

Additional results for ACC, F1-score, and AUPRC are reported in Appendix F. Appendix G presents results on four additional established benchmark causal graphs from the bnlearn repository (Asia, Cancer, Earthquake, and Survey); these are not additional independent physical experiments. Runtime measurements and limitations are detailed in Appendix J.

TTT-SCL (Notears) outperforms TTT-SCL (random) in the reported settings. Increasing the random-initialization search budget from 2,000 to 5,000 steps improves AUROC on RFF_G and Sachs; on Sachs it narrows, but does not eliminate, the gap to NOTEARS initialization (71.9 versus 78.9). These results provide partial evidence that additional search can mitigate initialization sensitivity, not evidence of initialization independence (Appendix H).

5.2 Ablation Study

We design experiments to empirically validate how these two components contribute to the quality of the generated training data. We first ablate the sparsity term in the optimization objective to isolate its effect. We compare the full TTT-SCL (Notears) method against a variant, TTT-SCL (Notears-s), where the sparsity penalty is removed, thus optimizing for AD alone. Results in Table 3 show that removing the sparsity term leads to a consistent performance drop across all test settings.

Table 3: Ablation experiment of sparsity. Results are presented as AUROC.

	RFF_G	Linear_U	Chebyshev_G	Sachs	Syntren
TTT-SCL (Notears)	91.8 (3.1)	86.3 (4.4)	83.0 (8.7)	78.9	80.1
TTT-SCL (Notears-s)	86.8 (2.9)	84.3 (7.9)	69.7 (12.4)	63.5	76.1

To further demonstrate the effectiveness of AD and the necessity of sparsity, the AD, sparsity, score of the training data obtained by different methods under different test data, as well as the AUROC on the test data were recorded in Appendix I. These ablations support the empirical usefulness of combining alignment and sparsity, but the score does not perfectly rank downstream AUROC.

5.3 Relationship with Score-based Methods

To clearly distinguish our approach from classical score-based methods, we provide a detailed stage-wise analysis comparing three key outputs: the seed graph, the highest-scoring graph found during TTT-SCL's search, and the final SCL prediction. The final learner improves over the highest-scoring graph in the reported settings; the effect of search alone varies by dataset. Specifically, we conducted experiments comparing three different outputs across four test domains (RFF, Linear, Chebyshev, and Sachs) for detailed analysis:

1. **Seed graph**: Initial graph from proxy methods
2. **Highest-score graph**: Highest-score graph found during TTT-SCL's stochastic refinement
3. **Final output**: Graph predicted by the SCL model trained on TTT-SCL-generated data

Table 4: Performance comparison (AUROC) across different stages of TTT-SCL.

	RFF_G	Linear_U	Chebyshev_G	Sachs
Seed graph	80.5	82.0	52.2	61.8
Highest-score graph in TTT-SCL search	88.9	80.1	75.8	66.6
Final graph from trained SCL model	**91.8**	**86.3**	**83.0**	**78.9**

The results in Table 4 distinguish the effects of search and learner training:

- **1→2 (Search Improvement)**: The highest-scoring graph improves over the seed on RFF_G, Chebyshev_G, and Sachs, but not on Linear_U (80.1 versus 82.0 AUROC). Thus, improving the graph score does not necessarily improve graph-recovery AUROC.
- **2→3 (Learning Improvement)**: The final SCL output improves over the highest-scoring graph on all four displayed datasets. TTT-SCL uses the retained graph states to generate training data, rather than constructing the full training set from only the highest-scoring graph. This supports the empirical value of the learner in these settings, without establishing a universal guarantee.

This two-stage process distinguishes candidate construction from final prediction. Appendix K presents additional ablation studies and experimental analyses of the relationship between TTT-SCL and score-based methods.

6 Related Work

Causal discovery has traditionally been approached as an unsupervised problem, with constraint-based methods (e.g., PC, FCI (Spirtes et al., 2000)), function-based methods (e.g., LiNGAM (Shimizu et al., 2006), ANM (Hoyer et al., 2008)), and score-based methods (e.g., GES (Chickering, 2002), NOTEARS (Zheng et al., 2018), DAG-GNN (Yu et al., 2019), GraN-DAG (Lachapelle et al., 2020)) inferring causal graphs directly from observational data. While principled, these methods often suffer from high sample complexity, sensitivity to faithfulness violations, and limited scalability.

Supervised Causal Learning (SCL) has recently emerged as a promising alternative that frames causal discovery as a supervised learning problem (Dai et al., 2023; Lorch et al., 2022; Ke et al., 2022). Prior SCL methods employ diverse architectures, including cascade classifiers for sequential independence testing (Ma et al., 2022), attention-based transformers operating on data as 3D tensors (Lorch et al., 2022; Ke et al., 2022; Froehlich & Koeppl, 2024), and pairwise attention networks (Zhang et al., 2025). Their target outputs also vary, ranging from undirected skeletons (Ma et al., 2022) and local v-structure classifications (Dai et al., 2023) to full directed adjacency matrices (Lorch et al., 2022; Ke et al., 2022), with some methods only guaranteeing recovery up to Markov equivalence (Zhang et al., 2025; Froehlich & Koeppl, 2024).

Regarding training data strategy, Montagna et al. (2024) recently investigated SCL generalization challenges but attributed performance drops primarily to unseen individual components and advocated increased pre-training diversity. In contrast, we identify a more fundamental limitation: *compositional generalization* failure, where SCL models fail on novel combinations of seen components, revealing the intractability of exhaustive static pre-training and motivating our TTT-SCL paradigm.

Finally, while test-time adaptation has shown promise in general machine learning (Liang et al., 2025; Sun et al., 2020; Wang et al., 2020; Liu et al., 2021; Sinha et al., 2023; Yuksekgonul et al., 2026), its application to causal discovery is gaining traction. Concurrently, TICL (Chen et al., 2026) addresses interventional settings with discrete data via self-augmentation, whereas our TTT-SCL provides a general framework for observational settings with continuous data, generating causally aligned training data at test time.

7 Conclusion

In this work, we identify key limitations of static supervised causal learning under distribution shifts, including failures of compositional generalization and transfer from synthetic to real-world data. We introduce TTT-SCL, which shifts the focus from static pre-training to instance-specific training at test time. By combining score-guided graph refinement, structure-induced data generation, and supervised learning, TTT-SCL constructs training data tailored to each observed dataset. Experiments across synthetic, pseudo-real, and real-world settings demonstrate its effectiveness and the benefit of learner training beyond direct score-based graph selection. These findings highlight test-time training as a promising direction for advancing supervised causal discovery.

AI use statement

Generative AI tools were used for language polishing, reference organization, manuscript formatting, and feedback on methodological descriptions and the interpretation of reported results. These tools were not used to generate experimental results. The authors reviewed the AI-assisted material and take responsibility for the final manuscript.

REPRODUCIBILITY STATEMENT

The experimental configurations, SIM procedure, and computational requirements are described in Appendices A, C, and J. Candidate-selection protocols for the supplementary ablations are given in Appendix K. The additional neural-baseline evaluation is described in Appendix L. Experiment code is linked in Section 5.

REFERENCES

Albert-László Barabási. Scale-free networks: a decade and beyond. *science*, 325(5939):412–413, 2009.

Wei Chen, Rui Ding, Bojun Huang, Yang Zhang, Qiang Fu, Yuxuan Liang, Han Shi, and Dongmei Zhang. Test-time learning of causal structure from interventional data. *arXiv preprint arXiv:2602.19131*, 2026.

David Maxwell Chickering. Optimal structure identification with greedy search. *Journal of machine learning research*, 3(Nov):507–554, 2002.

Haoyue Dai, Rui Ding, Yuanyuan Jiang, Shi Han, and Dongmei Zhang. Ml4c: Seeing causality through latent vicinity. In *Proceedings of the 2023 SIAM International Conference on Data Mining (SDM)*, pp. 226–234. SIAM, 2023.

Philipp Froehlich and Heinz Koeppl. Graph structure inference with bam: neural dependency processing via bilinear attention. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, pp. 128847–128885, 2024.

Edgar N Gilbert. Random graphs. *The Annals of Mathematical Statistics*, 30(4):1141–1144, 1959.

Olivier Goudet, Diviyan Kalainathan, Philippe Caillou, Isabelle Guyon, David Lopez-Paz, and Michèle Sebag. Causal generative neural networks. *arXiv preprint arXiv:1711.08936*, 2017. URL https://arxiv.org/abs/1711.08936.

Patrik O. Hoyer, Dominik Janzing, Joris Mooij, Jonas Peters, and Bernhard Schölkopf. Nonlinear causal discovery with additive noise models. In *Proceedings of the 22nd International Conference on Neural Information Processing Systems*, pp. 689–696, 2008.

Nan Rosemary Ke, Silvia Chiappa, Jane Wang, Anirudh Goyal, Jorg Bornschein, Melanie Rey, Theophane Weber, Matthew Botvinic, Michael Mozer, and Danilo Jimenez Rezende. Learning to induce causal structure. *arXiv preprint arXiv:2204.04875*, 2022.

Sébastien Lachapelle, Philippe Brouillard, Tristan Deleu, and Simon Lacoste-Julien. Gradient-based neural dag learning. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=rklbKA4YDS.

Jian Liang, Ran He, and Tieniu Tan. A comprehensive survey on test-time adaptation under distribution shifts. *International Journal of Computer Vision*, 133(1):31–64, 2025.

Yuejiang Liu, Parth Kothari, Bastien van Delft, Baptiste Bellot-Gurlet, Taylor Mordan, and Alexandre Alahi. Ttt++: when does self-supervised test-time training fail or thrive? In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, pp. 21808–21820, 2021.

Lars Lorch, Scott Sussex, Jonas Rothfuss, Andreas Krause, and Bernhard Schölkopf. Amortized inference for causal structure learning. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, pp. 13104 – 13118, 2022.

Pingchuan Ma, Rui Ding, Haoyue Dai, Yuanyuan Jiang, Shuai Wang, Shi Han, and Dongmei Zhang. Ml4s: Learning causal skeleton from vicinal graphs. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 1213–1223, 2022.

Francesco Montagna, Nicoletta Noceti, Lorenzo Rosasco, Kun Zhang, and Francesco Locatello. Causal discovery with score matching on additive models with arbitrary noise. In *Conference on Causal Learning and Reasoning*, pp. 726–751. PMLR, 2023.

Francesco Montagna, Max Cairney-Leeming, Dhanya Sridhar, and Francesco Locatello. Demystifying amortized causal discovery with transformers. *arXiv preprint arXiv:2405.16924*, 2024.

Judea Pearl. *Causality*. Cambridge university press, 2009.

Jonas Peters, Joris M Mooij, Dominik Janzing, and Bernhard Schölkopf. Causal discovery with continuous additive noise models. *The Journal of Machine Learning Research*, 15(1):2009–2053, 2014.

Jonas Peters, Dominik Janzing, and Bernhard Schölkopf. *Elements of causal inference: foundations and learning algorithms*. The MIT press, 2017.

Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Proceedings of the 21st International Conference on Neural Information Processing Systems*, pp. 1177–1184, 2007.

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 *International Conference on Machine Learning*, pp. 18741–18753. PMLR, 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. doi: 10.1126/science.1105809. URL https://www.science.org/doi/abs/10.1126/science.1105809.

Shohei Shimizu, Patrik O Hoyer, Aapo Hyvärinen, Antti Kerminen, and Michael Jordan. A linear non-gaussian acyclic model for causal discovery. *Journal of Machine Learning Research*, 7(10), 2006.

Samarth Sinha, Peter Gehler, Francesco Locatello, and Bernt Schiele. Test: Test-time self-training under distribution shift. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 2759–2769, 2023.

Peter Spirtes, Clark N Glymour, and Richard Scheines. *Causation, prediction, and search*. MIT press, 2000.

Yu Sun, Xiaolong Wang, Zhuang Liu, John Miller, Alexei Efros, and Moritz Hardt. Test-time training with self-supervision for generalization under distribution shifts. In *International conference on machine learning*, pp. 9229–9248. PMLR, 2020.

Tim Van den Bulcke, Koenraad Van Leemput, Bart Naudts, Piet van Remortel, Hongwu Ma, Alain Verschoren, Bart De Moor, and Kathleen Marchal. Syntren: a generator of synthetic gene expression data for design and analysis of structure learning algorithms. *BMC bioinformatics*, 7(1):43, 2006.

Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno Olshausen, and Trevor Darrell. Tent: Fully test-time adaptation by entropy minimization. *arXiv preprint arXiv:2006.10726*, 2020.

Yue Yu, Jie Chen, Tian Gao, and Mo Yu. Dag-gnn: Dag structure learning with graph neural networks. In *International conference on machine learning*, pp. 7154–7163. PMLR, 2019.

Mert Yuksekgonul, Daniel Koceja, Xinhao Li, Federico Bianchi, Jed McCaleb, Xiaolong Wang, Jan Kautz, Yejin Choi, James Zou, Carlos Guestrin, et al. Learning to discover at test time. *arXiv preprint arXiv:2601.16175*, 2026.

Jiaru Zhang, Rui Ding, Qiang Fu, Huang Bojun, zizhen Deng, Yang Hua, Haibing Guan, Shi Han, and Dongmei Zhang. Learning identifiable structures avoids bias in DNN-based supervised causal learning. In *The 28th International Conference on Artificial Intelligence and Statistics*, 2025. URL https://openreview.net/forum?id=8JwuE2HhY8.

Xun Zheng, Bryon Aragam, Pradeep Ravikumar, and Eric P. Xing. Dags with no tears: continuous optimization for structure learning. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pp. 9492–9503, 2018.

A Detailed configuration

A.1 Setting of static training data

For all static SCL training setups evaluated(including i.i.d., Graph/Noise/Mechanism shift), we use a total of $K = 2,000$ synthetic training instances. Each instance contains $n = 200$ i.i.d. observations. The specific training settings for different test instances are as follows:

Table 5: Graph/Noise/Mechanism shift training data setting

	RFF_G_ER	RFF_G_SF	Linear_U_ER	Linear_U_SF	Chebyshev_G_ER	Chebyshev_G_SF
Graph shift	RFF_G_SF	RFF_G_ER	Linear_U_SF	Linear_U_ER	Chebyshev_G_SF	Chebyshev_G_ER
Noise shift	RFF_U_ER	RFF_U_SF	Linear_L_ER	Linear_L_SF	Chebyshev_U_ER	Chebyshev_U_SF
Mechanism shift	Chebyshev_G_ER	Chebyshev_G_SF	RFF_U_ER	RFF_U_SF	RFF_G_ER	RFF_G_SF

In the Component-mixed setting, these 2,000 instances are uniformly distributed across the 6 mechanism_noise_graph combinations, resulting in approximately 330 instances per specific combination. The training data is a mixture of RFF_U_ER, RFF_U_SF, Linear_G_ER, Linear_G_SF, Chebyshev_U_ER, and Chebyshev_U_SF. This makes the model see all components, mechanism (RFF, Linear, Chebyshev), graph (ER, SF), noise (G, U), but not see the specific combination in the test instance, such as RFF_G_ER.

A.2 Hyperparameter settings

Our score combines average log-likelihood with an edge-count penalty, $\mathrm{score}(G, D_{test}) = \mathrm{AD}(G, D_{test}) - \|\mathbf{A}_G\|_0$. This reflects the fit-versus-complexity tradeoff of information criteria, but it is not AIC: edge count is not generally the number of fitted GAM parameters, and our likelihood is normalized across samples and variables. We use the score as a heuristic for candidate construction, with the edge-penalty coefficient fixed to one.

We perform $N_{\mathrm{steps}} = 2000$ stochastic refinement steps per test instance and use $K = 200$ retained states. Larger budgets may increase search coverage or the number of training pairs, but do not guarantee improved downstream accuracy; we use the fixed budgets reported here.

A.3 Test-time learner training protocol

For each observed test dataset, the learner starts from random weights and trains on $K = 200$ graph-labeled pairs. Each pair contains $n_{\mathrm{syn}} = 200$ observations, except Sachs, where $n_{\mathrm{syn}} = 853$. The AVICI-based transformer uses hidden size 128, 12 alternating attention blocks, 8 heads, widening factor 4, and dropout 0.1. We use Adam with learning rate 3×10^{-4}, batch size 10, linear warmup over 20 epochs, gradient clipping at 1.0, and no weight decay. For TTT-SCL in the supplementary experiments reported in Appendices K and L, training is fixed at 100 epochs and only the final checkpoint is evaluated; there is no validation split, early stopping, or selection using test graph labels. Each pair is independently sampled from GAM mechanisms fitted separately for each candidate graph and node, with independent $\epsilon_i \sim \mathcal{N}(0, 1)$. Training applies joint permutations of nodes and graph labels and reshuffles pairs each epoch. These augmentations do not use the unknown test graph.

B Consistency on other model backbones

To further validate the generality of the TTT-SCL framework and the observed o.o.d generalization challenges across different model architectures, we conduct experiments using the Pairwise Attention from Zhang et al. (2025) (**SiCL**) as an alternative model backbone. Unlike the AVICI transformer used in the main experiments, which predicts a full directed adjacency matrix (DAG),

SiCL incorporates pairwise attention mechanisms and is trained to predict the undirected skeleton and v-structures of the causal graph. This setup allows us to investigate whether the identified o.o.d failure patterns persist when using a fundamentally different architecture (with pairwise attention) and a different learning target (skeleton and v-structures instead of a full DAG), thereby testing the robustness of our conclusions.

B.1 Experimental Setup

Backbone Model is SiCL (Pairwise Attention Network) Zhang et al. (2025). Learning Target is Undirected graph skeleton. The training strategy for the static baseline models (i.i.d. and SiCL(mixed)) follows the same data generation procedures described in Section 3.1 and Appendix A.1, but the ground-truth labels are converted to the appropriate representation for SiCL (skeleton labels). Evaluation Metric is AUROC for edge presence in the predicted skeleton. The OOD settings are the same as those in Figure 2, with shift configurations given in Table 5: *i.i.d.*, *Graph shift*, *Noise shift*, *Mechanism shift*. The *AVICI(mixed)* is replaced with *SiCL(mixed)*, respectively.

B.2 Results and Analysis

Table 6 presents the AUROC for skeleton discovery under different distribution shifts.Consistent with the findings in Fig 2 using the AVICI backbone, the SiCL backbone—which employs a fundamentally different pairwise attention architecture and learns undirected skeletons rather than full DAGs—exhibits the same pattern of out-of-distribution generalization failure. Under i.i.d. conditions, SiCL achieves perfect or near-perfect performance. However, significant performance degradation occurs across all types of distribution shifts, with mechanism shifts proving particularly damaging (e.g., dropping to 66.5 on RFF_G_SF and 58.4 on Chebyshev_G_SF). Critically, the SiCL(mixed) variant, while trained on data containing all individual distributional components (graph types, mechanisms, and noise distributions), still fails to generalize to novel combinations of these factors. This demonstrates that SCL models struggle with compositional generalization—they memorize specific configuration patterns rather than learning modular causal representations. These results demonstrate that the OOD generalization challenge is not specific to a particular model architecture or output representation, but represents a fundamental limitation of the static pre-training paradigm in supervised causal learning. The consistent failure patterns across both transformer-based (AVICI) and pairwise-attention-based (SiCL) models strongly validate the need for test-time adaptation frameworks like TTT-SCL.

Table 6: OOD generalization performance for skeleton using the SiCL (Pairwise Attention) backbone.

	RFF_G_ER	RFF_G_SF	Linear_U_ER	Linear_U_SF	Chebyshev_G_ER	Chebyshev_G_SF
iid	82.1(6.7)	100.0(0.0)	81.4(6.9)	100.0(0.0)	94.3(2.8)	100.0(0.0)
Graph shift	66.4(9.0)	85.4(4.1)	65.8(6.9)	94.0(2.5)	73.0(5.7)	92.9(4.3)
Noise shift	60.0(8.9)	91.7(3.8)	65.3(7.4)	84.0(7.7)	88.6(5.3)	89.3(5.4)
Mechanism shift	62.1(7.4)	66.5(6.3)	59.4(4.7)	83.8(4.8)	76.1(8.9)	58.4(9.7)
SiCL(mixed)	64.4(8.0)	74.4(10.7)	66.7(7.3)	82.7(8.2)	85.6(3.7)	91.2(4.1)

C Structure-Induced Mechanism (SIM)

Intuition Given an observed test dataset $D_{test} = \{x^{(t)}\}_{t=1}^{n}$ and a candidate DAG G_k, SIM produces a synthetic dataset D_k whose distribution is induced by (G_k, Θ_k), where $\Theta_k = \{\hat{f}_{k,i}\}_{i=1}^{d}$ denotes the mechanisms fitted from D_{test}. Exogenous noises are fixed to independent standard normal variables.

Formal description. For each candidate graph and node i with parent set $\mathrm{Pa}_{G_k}(X_i)$, we fit a separate GAM mechanism by solving

$$\hat{f}_{k,i} \;=\; \arg\min_{f \in \mathcal{F}} \sum_{t=1}^{n} \ell\big(x_i^{(t)},\, f\big(x^{(t)}_{\mathrm{Pa}_{G_k}(X_i)}\big)\big),$$

where ℓ is squared error for continuous targets and $x_{\mathrm{Pa}}^{(t)}$ denotes observed parent values. We fix the exogenous noise to independent standard normal variables, rather than estimating a residual-noise distribution. To generate n_{syn} synthetic observations, iterate over nodes in topological order for each $t = 1, \ldots, n_{\mathrm{syn}}$:

$$\tilde{X}_i^{(t)} = \hat{f}_{k,i}\big(\tilde{X}^{(t)}_{\mathrm{Pa}_{G_k}(X_i)}\big) + \tilde{\epsilon}_{k,i}^{(t)}, \qquad \tilde{\epsilon}_{k,i}^{(t)} \overset{\text{iid}}{\sim} \mathcal{N}(0,1).$$

The resulting dataset is $D_k = \{\tilde{x}^{(t)}\}_{t=1}^{n_{\mathrm{syn}}}$. Here n is the number of observed test rows and n_{syn} is the number of generated rows per training pair; they need not coincide. The standard-normal assumption specifies SIM, not the noise distributions of the synthetic evaluation benchmarks.

D Implementation of AD

In the main text, we propose the Alignment of Distribution (AD) metric as a core measure of causal similarity between the generated training data D_{train} and the test instance D_{test}. While the likelihood-based implementation was used in our primary experiments, we provide alternative formulations here to accommodate different data distributions and modeling assumptions.

Likelihood-based AD. Consistent with the SIM specification in Appendix C, the likelihood in Eq. (4) uses independent Gaussian noise with fixed unit variance:

$$p\big(x_i^{(t)} \mid f_i^G(\mathbf{x}^{(t)}_{\mathrm{Pa}_G(i)})\big) = \mathcal{N}\Big(x_i^{(t)}; f_i^G(\mathbf{x}^{(t)}_{\mathrm{Pa}_G(i)}), 1\Big)\,.$$

The variance is fixed rather than estimated from the regression residuals; the same noise specification is used for synthetic sampling.

D.1 R^2-based AD

For continuous variables under additive noise models, the coefficient of determination (R^2) provides an intuitive measure of goodness-of-fit for each causal mechanism:

$$AD_{R^2}(G_{train}, D_{test}) = \frac{1}{d}\sum_{i=1}^{d}\left[\frac{1}{K}\sum_{k=1}^{K} R^2\left(f_i^k(\mathbf{Pa}^k(X_i)), X_i\right)\right]$$

This value approaches 1 when the fitted mechanisms explain the variance in D_{test} well, indicating strong alignment.

D.2 Normalized Wasserstein Distance-based AD

For multi-modal or heavy-tailed distributions, the Wasserstein distance offers a robust metric for comparing empirical distributions. We define a *Normalized Wasserstein Distance (NWD)* based AD metric as follows:

For a given variable X_i and a candidate graph G^k with its fitted mechanism f_i^k, we compute:

$$\mathrm{NWD}(f_i^k, G^k, D^{test}) := 1 - \frac{W_1\left(\{x_i\},\ \{f_i^k(\mathbf{Pa}^k(X_i))\}\right)}{\max(\mathcal{U}) - \min(\mathcal{U})}$$

where:

- $\{x_i\}$ are the observed values of X_i in D_{test}.
- $\{f_i^k(\mathbf{Pa}^k(X_i))\}$ are the values generated by applying the fitted mechanism f_i^k to the parent values in D_{test}.
- W_1 is the 1-Wasserstein distance (Earth Mover's Distance). For two equally sized, sorted collections of values $\{a^{(j)}\}$ and $\{b^{(j)}\}$, it is defined as:

$$W_1(\{a\},\{b\}) = \frac{1}{n}\sum_{j=1}^{n}|a^{(j)} - b^{(j)}|$$

- $\mathcal{U} = \{x_i\} \cup \{f_i^k(\mathbf{Pa}^k(X_i))\}$ is the union of the observed and generated values for X_i.
- The denominator, $\max(\mathcal{U}) - \min(\mathcal{U})$, is the range of the combined set, used for normalization.

The resulting NWD value lies between 0 and 1, where 1 indicates a perfect match between the generated and observed distributions for that variable. The overall AD metric is then the average NWD across all variables and generated graphs:

$$AD_{\text{NWD}}(G_{train}, D_{test}) = \frac{1}{K}\sum_{k=1}^{K}\left[\frac{1}{d}\sum_{i=1}^{d}\text{NWD}(f_i^k, G^k, D^{test})\right]$$

D.3 Selection Guidance

The **likelihood-based** AD is most natural for probabilistic models and was used in our main experiments. The R^2**-based** AD is suitable for continuous variables under additive noise assumptions, often leading to computationally efficient and intuitive scores. The **NWD-based** AD is recommended for complex, non-Gaussian, or heavy-tailed distributions where likelihood or R^2 might be less informative or robust. The TTT-SCL framework is agnostic to the specific choice of AD metric, allowing users to select the most appropriate one for their domain.

E Workflow of Training Sets Generation

To further illustrate the generation of training sets, we have drawn a workflow to visualize the process, as shown in Fig. 3.

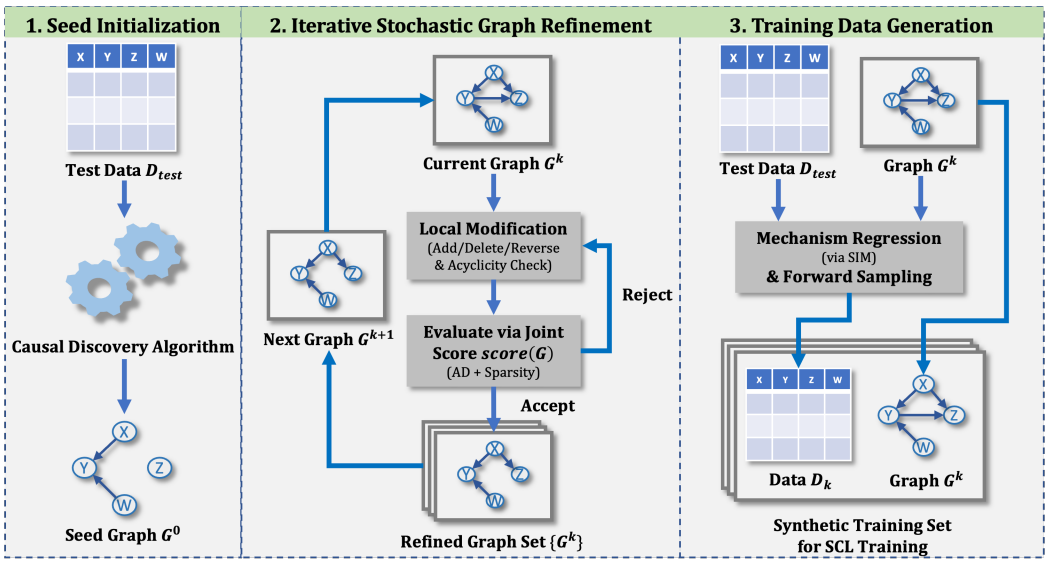

Figure 3: Workflow of training set generation. *AD + Sparsity* denotes AD minus the edge-count penalty. Refinement retains the final K states. SIM uses fitted GAM mechanisms and independent standard-normal noise. The displayed scores are illustrative.

F Performance in Other Metrics

In the main text, we primarily reported the AUROC for edge prediction to succinctly demonstrate the impact of training data quality on model performance. For a more comprehensive evaluation, we provide results on additional standard causal discovery metrics in this appendix:

- **Accuracy (ACC)**: The proportion of correctly predicted edge presence/absence across all possible edges. Higher is better. This metric can be viewed as a normalized version of the Structural Hamming Distance (SHD), where instead of counting the number of incorrect edges, it measures the proportion of correct edge predictions relative to the total possible edges.
- **F1-Score**: The harmonic mean of precision and recall for edge prediction. Higher is better.
- **Area Under the Precision-Recall Curve (AUPRC)**: Particularly informative under class imbalance (sparse graphs). Higher is better.

Table 7: Comprehensive evaluation across multiple datasets and metrics. Mean (standard deviation) over multiple runs are reported for synthetic data.

Method	RFF_G			Linear_U			Chebyshev_G			Sachs			Syntren		
	ACC↑	F1↑	AUPRC↑	ACC↑	F1↑	AUPRC↑	ACC↑	F1↑	AUPRC↑	ACC↑	F1↑	AUPRC↑	ACC↑	F1↑	AUPRC↑
PC	75.6(4.2)	39.5(9.7)	37.5(6.3)	74.0(4.4)	40.4(8.4)	37.2(6.2)	73.7(5.1)	37.4(12.6)	36.8(7.8)	84.2	45.7	30.1	84.75	16.43	6.89
GES	76.7(7.7)	49.8(16.3)	43.5(12.3)	71.9(9.8)	55.4(12.9)	45.3(9.3)	72.1(5.1)	38.5(10.5)	35.7(6.6)	82.6	36.3	24.2	65.50	1.42	4.79
NOTEARS	86.6(3.3)	73.2(6.2)	64.1(7.4)	89.1(2.9)	76.6(7.3)	69.7(8.6)	72.3(2.7)	14.5(8.7)	29.8(3.3)	82.6	36.3	24.2	94.75	0.00	5.00
AVICI(scm-v0)	93.1(1.6)	87.3(3.4)	94.9(3.1)	73.9(7.1)	41.8(18.4)	52.8(17.0)	80.6(5.4)	58.4(14.5)	69.3(14.2)	83.4	23.0	31.6	93.00	22.22	25.53
TTT-SCL (random)	83.2(5.3)	72.8(9.4)	68.8(10.8)	75.9(6.1)	59.8(11.8)	56.2(11.5)	75.5(7.0)	56.6(10.8)	60.0(10.0)	68.5	24.0	24.5	72.50	16.66	53.91
TTT-SCL (Notears)	86.8(3.5)	78.4(6.1)	76.0(8.6)	78.7(3.9)	65.4(8.0)	65.0(9.9)	77.1(6.7)	61.9(10.2)	66.0(16.3)	85.9	56.4	53.6	90.50	32.14	51.85

Table 7 presents the performance of all compared methods across three distinct synthetic data settings (RFF_G, Linear_U, and Chebyshev_G) and the real-world Sachs dataset. TTT-SCL (Notears) achieves highly competitive performance across all datasets and evaluation metrics (ACC, F1, AUPRC), demonstrating its robustness to distribution shifts. It consistently outperforms traditional methods (PC, GES, NOTEARS) and the strong pre-trained SCL baseline AVICI(scm-v0) on most settings, particularly on the challenging Chebyshev_G and real-world Sachs dataset. While AVICI(scm-v0) excels in the RFF_G setting it was trained on, its performance degrades significantly under mechanism shifts (Linear_U) and on real data, highlighting the limitation of static pre-training. The superior performance of TTT-SCL across multiple metrics confirms that its test-time training strategy generates high-quality, causally-aligned training data, leading to more accurate and reliable causal discovery.

G Additional Experiments on Benchmark Causal Graphs

To further validate TTT-SCL's performance on real-world causal structures, we conducted additional experiments on well-established benchmark causal graphs from the bnlearn repository. The scarcity of real-world causal datasets with ground truth is a fundamental challenge in causal discovery research. While most works primarily rely on synthetic data and a limited number of real datasets (e.g., Sachs), benchmark causal graphs from bnlearn provide valuable testbeds as they represent causal structures derived from real-world domains and expert knowledge.

We selected four representative graphs from bnlearn:

- **Asia:** A classic medical diagnostic network modeling the relationships between visiting Asia, smoking, tuberculosis, lung cancer, bronchitis, and various test results. This graph represents a well-known benchmark in causal inference with clear medical relevance.
- **Cancer:** A compact but meaningful graph modeling causal relationships in cancer epidemiology, including pollution, smoking, and genetic factors. Its small size belies its representativeness of real-world medical causal reasoning.
- **Earthquake:** Models causal relationships between burglary, earthquake, alarm triggers, and neighbor responses. This graph exemplifies causal reasoning in security and monitoring systems.
- **Survey:** Represents causal relationships in social science research, including age, sex, education, occupation, and transportation preferences. This graph demonstrates causal structures in sociological studies.

These benchmark graphs are representative because they: (1) capture diverse real-world domains (medical, social, security), (2) are widely recognized and validated in the causal inference literature, and (3) reflect expert-curated causal knowledge rather than purely synthetic constructions.

We parameterized these graphs using Chebyshev polynomial mechanisms to generate pseudo-real datasets, maintaining the authentic causal structures while incorporating realistic nonlinear relation-

ships. Table 8 shows that TTT-SCL consistently achieves state-of-the-art performance across all benchmark graphs, demonstrating its robustness to diverse real-world causal structures.

Table 8: Performance comparison (AUROC) on benchmark causal graphs from bnlearn repository.

Method	Asia	Cancer	Earthquake	Survey
PC	74.1	70.2	75.5	90.0
GES	46.4	85.1	80.3	88.3
NOTEARS	68.7	87.5	60.1	64.9
AVICI (scm-v0)	83.3	86.9	94.0	89.4
TTT-SCL (random)	86.8	84.5	84.5	92.7
TTT-SCL (NOTEARS)	**91.0**	**91.6**	**98.8**	**95.5**

TTT-SCL performs well on these additional benchmark graphs; these results complement, but do not replace, evaluation on real physical systems.

H Sensitivity to Seed Graph Initialization

As shown in Table 2, TTT-SCL (Notears) outperforms TTT-SCL (random) under the default budget. To study how additional search affects this gap, TTT-SCL (random-long) retains random initialization but increases the refinement budget from $N_{\text{steps}} = 2000$ to 5000.

Table 9: Effect of seed quality and search budget on AUROC.

	RFF_G	Sachs
TTT-SCL (Notears)	91.8	78.9
TTT-SCL (random)	88.4	58.6
TTT-SCL (random-long)	89.1	71.9

Increasing the random-initialization search budget from 2,000 to 5,000 steps improves AUROC in both evaluated settings: from 88.4 to 89.1 on RFF_G and from 58.6 to 71.9 on Sachs. The corresponding NOTEARS-initialized results remain higher at 91.8 and 78.9. Additional search therefore provides partial mitigation of initialization sensitivity, but these experiments do not establish that initialization is irrelevant or that its effect is solely computational.

I Ablation study

The main text evaluates the sparsity penalty in the TTT-SCL objective. This appendix provides further empirical evidence on the roles of the AD metric and the sparsity penalty.

I.1 The role of AD and sparsity

Table 10 reports AD, edge count, score, and downstream AUROC. Removing the sparsity term produces denser candidates and lower AUROC in these comparisons. Combining AD and sparsity is empirically useful, but the score does not perfectly rank downstream AUROC. For example, on Chebyshev_G, the random-initialized variant has a higher score than the NOTEARS-initialized variant (-333.3 versus -335.5) but a lower AUROC (0.796 versus 0.830). These measurements do not establish a statistical correlation or a guarantee that a higher score yields a more accurate final learner.

I.2 Control AD, change sparsity

To control sparsity independent of AD, we design a controlled experiment based on the ground-truth test graph G_{test}. For a given G_{test} and its observational data D_{test}, we generate alternative candidate training graphs G_{train} by **gradually adding extra edges** to G_{test} (while ensuring the

Table 10: AD and Sparsity characterize the quality of the training data.

Metric	Methods	RFF_G	Linear_U	Chebyshev_G
AD	TTT-SCL(random)	-370.0	-258.5	-303.5
	TTT-SCL(Notears-s)	-357.5	-217.5	-298.0
	TTT-SCL(Notears)	-363.0	-220.5	-308.0
Sparsity	TTT-SCL(random)	33.15	34.00	29.49
	TTT-SCL(Notears-s)	38.80	38.75	38.85
	TTT-SCL(Notears)	31.95	35.65	27.25
Score	TTT-SCL(random)	-403.3	-293.0	-333.3
	TTT-SCL(Notears-s)	-397.0	-256.8	-337.8
	TTT-SCL(Notears)	-395.0	-256.5	-335.5
AUROC	TTT-SCL(random)	0.884	0.823	0.796
	TTT-SCL(Notears-s)	0.868	0.843	0.697
	TTT-SCL(Notears)	0.918	0.863	0.830

Note: AD/Score/AUROC (Higher is better), Sparsity (Low is better)

resulting graph remains a DAG). This creates a series of graphs that are supergraphs of the true graph. The true graph is used only in this explanatory controlled experiment, not to construct candidates or train the full method.

- **Setting 1 (Sparse)**: Add a small number of extra edges ($|E_{add}| = m_1$).
- **Setting 2 (Medium)**: Add a medium number of extra edges ($|E_{add}| = m_2$, $m_2 > m_1$).
- **Setting 3 (Dense)**: Add a large number of extra edges ($|E_{add}| = m_3$, $m_3 > m_2$).

For each generated supergraph G_{train} in these settings, we then: 1. Parameter Fitting: Fit a separate GAM mechanism f_i for each node from D_{test} using G_{train} (via SIM), with independent fixed noise $\epsilon_i \sim \mathcal{N}(0, 1)$. 2. Forward Sampling: Generate synthetic training data D_{train} from the fitted SCM (G_{train}, f, ϵ). 3. Calculate Metrics: Compute the AD score between D_{train} and D_{test}, and the sparsity of G_{train}. 4. Train & Evaluate: For each (G_{train}, D_{train}) pair, train an SCL model (AVICI backbone) and evaluate its AUROC on recovering the *true* G_{test} from D_{test}.

This procedure is repeated for K graphs per setting. Adding edges preserves the availability of the original parent sets, but the fitted mechanisms and their finite-sample AD values need not be identical. The experiment compares downstream performance across these density settings.

Table 11 shows the results for the **RFF_ER_G** dataset, which are representative of the overall trend.

Table 11: Control AD, change sparsity

RFF_ER_G	setting	AD	sparsity	AUROC
Control AD, change sparsity	1	-375	25.91	1.0(0)
	2	-368 (+1.8%)	32.32(+24.7%)	0.972(0.017)
	3	-362(+3.4%)	36.59(+41.2%)	0.908(0.023)

The AD values in Table 11 vary by less than 4%, while the graph edge counts increase across settings. The downstream AUROC decreases from 1.000 in the sparsest setting to 0.972 in the medium-density setting and 0.908 in the densest setting. These descriptive results support the value of the sparsity penalty; no statistical-significance test is reported.

J SCALABILITY ANALYSIS AND LIMITATION

The computational complexity of TTT-SCL is dominated by the Stochastic Graph Refinement step. The search space for Directed Acyclic Graphs (DAGs) with d variables is super-exponential, ren-

dering exhaustive search intractable. Our stochastic search conducts a guided walk through this space, and its complexity is determined by the number of steps N_{steps} and the cost of evaluating the Alignment of Distribution (AD) score for each candidate graph.

The evaluation for a single candidate graph G involves:

- **Mechanism Fitting**: For each node X_i, we fit a GAM mechanism from n test observations using its encountered parent set $\mathbf{Pa}_G(X_i)$. Fitting cost depends on the parent-set size, basis representation, solver, and iteration count. The edge penalty encourages sparse candidates but does not impose a constant maximum in-degree; we therefore do not reduce the fitting cost to a guaranteed $O(nd)$ bound.
- **Likelihood Calculation**: After fitting, computing the log-likelihood for all n samples and d variables has a cost of $O(n \cdot d)$.

Runtime depends on the number of refinement steps, graph-proposal and acyclicity checks, and the mechanism-fitting and scoring cost for the encountered parent sets. SIM generation for the retained states and learner training introduce additional costs. We therefore characterize practical cost through the measured component breakdown below rather than claim guaranteed linear scaling. In the reported configuration, $N_{\text{steps}} = 2000$ and $K = 200$.

Table 12 reports the measured component costs. Serial graph refinement is the main bottleneck; the summed times are approximately 30.6, 70.0, and 126.9 minutes for 10, 20, and 30 nodes, respectively. The experiment uses an 80 GB A100 GPU; the reported usage at 10 nodes is about 4 GB, while the peak learner memory at 30 nodes is approximately 30 GB. These measurements characterize the tested regime, not guaranteed asymptotic scaling.

The per-instance cost limits high-throughput use. The tested setting may suit offline scientific analyses where longer computation is acceptable, but it does not replace amortized inference for applications requiring immediate predictions. Performance at 50–100 nodes remains unverified.

Table 12: Runtime breakdown of TTT-SCL for a test instance with varying number of nodes.

Component	10 nodes	20 nodes	30 nodes
Stochastic Graph Refinement	26 min	61 min	113 min
Training Data Generation (SIM fitting)	1.3 min	3.2 min	5.6 min
Model Training	3.3 min	5.8 min	8.3 min

K Supplementary ablations of candidate construction and learner training

Table 13 reports ablations of candidate construction and learner training. The variants use the same NOTEARS initializer and, where a learner is present, the same number of training pairs, SIM procedure, and learner configuration.

Table 13: Ablations of candidate construction and learner training. AUROC (%).

Method	RFF_G	Linear_U	Chebyshev_G	Sachs	SynTReN
Seed graph	80.5	82.0	52.2	61.8	49.8
Highest-score graph	88.9	80.1	75.8	66.6	57.9
Top K voting	91.3	83.5	81.5	65.8	55.7
Score-weighted voting	91.4	83.6	82.1	65.2	56.5
Best-only training	90.6	76.0	79.6	59.1	53.2
Random candidates + SCL	71.1	73.1	51.6	42.2	57.7
Full TTT-SCL	**91.8**	**86.3**	**83.0**	**78.9**	**80.1**

Candidate-selection protocols. Seed reports the NOTEARS graph before refinement; highest score returns the graph with the largest score encountered during refinement. Top K voting av-

erages edge indicators over the $K = 200$ highest-scoring candidates. Score-weighted voting uses those same top-K candidates with softmax weights computed from standardized graph scores. In contrast, Full TTT-SCL uses the final K graph states, retaining repeats after rejected proposals; it does not use the top-K selection rule. Thus, the two voting baselines share a candidate pool with each other, but not necessarily with the full method.

Best-only training keeps the learner and number of pairs fixed, but generates every pair from the highest-scoring graph, so all graph labels are identical. Random candidates + SCL retains the search and learner budgets but removes score-based acceptance. The highest-scoring graph improves over the seed on four of five datasets, with Linear_U the exception. The full method exceeds the compared variants in this table, with larger margins over voting on Sachs and SynTReN. These comparisons support the empirical usefulness of candidate construction and learner training, but do not establish that the learner is universally necessary or isolate a learner-only effect on an identical candidate pool.

L ADDITIONAL NEURAL CAUSAL-DISCOVERY BASELINES

GraN-DAG (Lachapelle et al., 2020), DAG-GNN (Yu et al., 2019), and CGNN (Goudet et al., 2017) are run separately for each observed dataset using fixed public configurations and no tuning based on the ground-truth graph. CGNN requires a skeleton; we estimate it using PC from the same observations and evaluate CGNN's binary graph output. No true skeleton is provided.

Table 14: Comparison with neural causal discovery baselines. AUROC (%).

Method	RFF_G	Linear_U	Chebyshev_G	Sachs	SynTReN
GraN-DAG	57.9	58.9	55.0	56.1	51.6
DAG-GNN	82.2	**92.4**	43.8	51.1	55.3
CGNN (PC skeleton)	55.6	55.3	60.0	61.5	58.6
Full TTT-SCL	**91.8**	86.3	**83.0**	**78.9**	**80.1**

Full TTT-SCL exceeds GraN-DAG and CGNN on all five settings, and DAG-GNN on four; DAG-GNN performs better on Linear_U. These results extend the comparison beyond pretrained SCL models. Because CGNN uses a PC-estimated skeleton and a binary output, its result reflects that full pipeline and is not a comparison with an oracle skeleton or a continuous edge-ranking score.