CausalBench: A Unifying Framework for Benchmarking Causal Learning Models

Ahmet Kapkiç, Pratanu Mandal, Abhinav Gorantla, Shu Wan, Ertuğrul Çoban, Paras Sheth, Huan Liu, K. Selçuk Candan*
{akapkic,pmandal5,agorant2,swan,ecoban,psheth5,huanliu,candan}@asu.edu
Arizona State University, Tempe, AZ

Abstract

Due to the critical role causality plays in decision-making, the state-of-the-art in machine learning for causality is rapidly evolving. With rapid development and deployment of new models, datasets, and metrics, it is increasingly difficult for researchers and practitioners to identify the most suitable approach for their problem. Models exhibit different performance when they train on different data, and even different hardware/software platforms, making it challenging for users to select the appropriate setup pertinent to their problem. It is therefore increasingly critical to fairly benchmark algorithms through a unified platform across different metrics, software, and hardware. To address these shortcomings, we present CausalBench- a comprehensive benchmarking tool for causal machine learning that facilitates accurate and reproducible benchmarking of causal models across metrics and deployment contexts as per the user's needs. CausalBench provides a platform for researchers to utilize its collaborative nature to create benchmarks that are transparent, flexible, and reproducible. It serves, not only as a benchmarking platform for causal machine learning models, but also as a resource that can explain benchmarking results across different metrics, software, and hardware setups. In this paper, we introduce the various key features of CausalBench, within the context of real-world use cases on static and temporal causal discovery tasks.

Keywords

Causal machine learning, causal discovery, benchmarking

1 Introduction

Machine learning models focus on maximizing association between features and outcomes [38]. Recent research has emphasized learning causal relationships to aid in establishing a direct and consistent link between features and outcomes, and are directly reflective of the problem being modeled [37, 85, 86].

In critical areas like public health, grasping these causal connections is vital. For instance, while modeling epidemics, it is essential to capture causally complex interplay of entities in a multi-layer network, including (a) individuals and their social interactions, (b) physical short-range and long-range networks of mobility, (c) parameters of disease models (such as infection rate, average length of recovery, and impact of treatment), and (d) intervention decisions (such as school closures or restrictions on mobility) [96, 102]. Unfortunately, traditional methods like randomized controlled trials (RCTs [93]) are often impractical or unethical in such contexts. Fortunately, the availability of extensive observational data enables

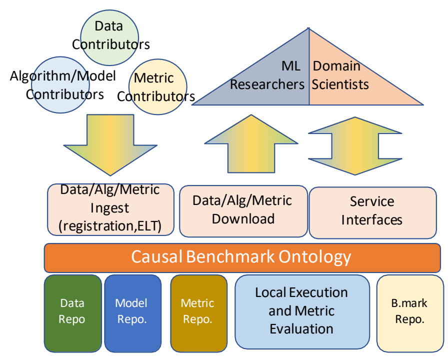

Figure 1: Overview of CausalBench (causalbench.org)

the approximation of causal relationships through data analytics, facilitating the discovery of meaningful patterns and informing effective decision-making. Causal learning from observational data offers a promising alternative to correlation-based learning [12].

1.1 Difficulties Facing the Research Community

Inferring causal relationships from data refers to the task of causal discovery [92]. Given the complexity of real-world problems, accurately identifying causal relationships is challenging. Researchers across different disciplines have made significant efforts to develop algorithms that allow for discovering causal relationships across different contexts (e.g. static, temporal, and spatio-temporal) [6, 13, 25, 81, 87, 91]. However, due to the volume of methods, datasets and metrics there is a lack of a unified platform that allows users to benchmark different algorithms to identify the most suitable ones for their use case. Moreover, existing tools exhibit different performance when they train different types of causal models on different hardware platforms, making it more challenging for users to select the appropriate setup pertinent to their problem.

Standardized evaluation played a major role in ML development and contributed to the impressive impact of ML in scientific innovation. Successful early benchmarking efforts, such as UCI ML and UCI KDD repositories [8], not only helped guide the development of efficient and effective ML algorithms, but also encouraged collaborative research and paved the way for the recent breakthroughs in deep learning. For example, to evaluate an image classifier, we have a widely used set of metrics (e.g., accuracy, F1 score and ROC-AUC), procedures (e.g., cross validation [94]) and datasets (e.g., MNIST [23], CIFAR10 [49] and ImageNet [22]). By developing a unified platform facilitating researchers to benchmark different causal discovery techniques across different levels (e.g. metrics,

*The first two authors have equal contributions. This research is funded by NSF Grant 2311716, "CausalBench: A Cyberinfrastructure for Causal-Learning Benchmarking for Efficacy, Reproducibility, and Scientific Collaboration".

datasets, hardware) can aid the causal learning community to discover areas where further efforts are needed, and aid in identifying the most suitable causal discovery setup for evaluating against their problems, and provide a better causally driven understanding of different problems under different contexts.

1.2 CausalBench

Arguing that this goal can only be achieved through systematic, objective, and transparent evaluation of causal learning models and algorithms, we present *CausalBench* [1, 47], a platform of publicly available benchmarks and consensus-building standards for the evaluation of causal learning models and algorithms from observational data[1]. Therefore, CausalBench aims to serve as a transparent, fair, and easy-to-use evaluation platform, with benchmark data, metrics, and procedures, as well as state-of-the-art baselines, in order to help establish trust in causal learning's innovation, collaboration, and applications (Figure 1):

- **Objective 1:** Universally adopted metrics, procedures and datasets. This involves conducting an extensive identification of existing datasets, performance metrics, and procedures used in the evaluation of state-of-the-art causal learning algorithms, and developing an "ontology" for benchmarking to standardize the evaluation methodology, improve transparency, and promote collaboration to efficiently advance causal learning.
- **Objective 2:** A standard and convenient way for the community to contribute data and models. Different from datasets for conventional machine learning, it is often difficult to obtain ground truth of the causal relations among observed variables, not to mention the potential existence of unobserved variables, as in many cases we have to work with datasets with incomplete causal knowledge. How to ascertain that disparate datasets can be integrated in a standard way is an open challenge.
- **Objective 3:** Evaluation of algorithms for novel problems. Novel problems of causal learning are emerging as the topic of causal learning attracts increasing attention. Researchers identify and formulate novel problems that are relevant in data intensive applications. To quantify the progress in the active research area of causal learning in a scientific way, it will be necessary to define evaluation standards.

In brief, CausalBench, publicly available both as a website and a python package aims to assist researchers and developers in easily applying and effectively evaluating (a) causal inference, (b) causal discovery, and (c) causal interpretability algorithms with a variety of standard metrics, procedures, and large-scale datasets.

2 Related Work

2.1 Causality

The study of causality has a long-standing history, yet defining causal relationships—and more so, uncovering them from data—remains an unresolved challenge [37]. Early approaches predominantly relied on statistical methodologies. For instance, the widely used Granger causality [7, 36, 53] is inherently statistical. Fisher [9, 28] and followers advocated a statistical perspective on causality, emphasizing randomized controlled trials (or, at minimum, quasi-randomized experiments [48, 75]) as a means to mitigate confounding effects. Rubin [78] advanced the "potential outcomes" framework and the counterfactual approach to defining causality [77, 79], interpreting causal inference as a missing-data problem where imputation offers a feasible solution. This school of thoughts accelerated significant advancements in data-driven causal inference, such as structural equation models [40, 71]. However, its applicability hinges on the validity of the "ignorability" assumption, which asserts that no unobserved confounders influence the causal mechanism [67, 72, 73, 76].

In contrast to these statistical approaches, Wright [104] arguedat causal conclusions cannot be inferred solely frofrom thee data without incorporating causal hypotheses. This point of view led to the development of highly effective practical methodologies, including path analysis [24, 30, 104], structural equation modeling (SEM [40, 71]), and Bayesian Networks [69], all of which leverage directed graphs to represent contextual knowledge, although not necessarily causally. Pearl introduced structural causal models (SCMs [70, 74]), which employ directed graphs to explicitly encode causal assumptions, enabling hypothesis testing and validation. Pearl and colleagues demonstrated that simple causal graphs can mitigate many common errors encountered in statistical causal analysis, offering a principled approach to handling colliders, confounders, and other sources of flawed causal reasoning [67, 71]. In this framework, learning causality requires rigorous methods that simultaneously infer structural causal hypotheses, represented as latent causal graphs, and estimate causal effects [52, 101].

2.2 ML Benchmarks – A Success Story

Conventional machine learning had maverick beginnings and its success is largely due to grassroots efforts to enable performance evaluation [29]. ML algorithms seek to build a mathematical model based on sample data, known as "training data", in order to make predictions or decisions without being explicitly programmed to perform the task [61]. When an ML algorithm discovers some patterns (a.k.a. a model), it is not guaranteed that the model actually works as designed. Thus, objective and fair performance evaluation is necessary to enable (a) if a new algorithm works better than an old one, (b) identify strengths and weaknesses of different algorithms, and (c) to enable reproducibility. The UC Irvine (UCI) machine learning repository [8] is one of the largest and earliest benchmarking efforts for ML research and is a collection of databases, domain theories, and data generators that the ML community uses for the empirical analysis of ML algorithms. ImageNet is another recent successful example of benchmark data [22], containing more than 14 million hand-annotated images and providing a standard by which the accuracy of image recognition software can be measured. ML research has advanced through efforts to standardize transparent evaluation. Scientific challenges like TREC [98] and CLEF [19], industrial crowdsourcing such as the Netflix challenge [27], ML-centric platforms like Kaggle [43], CodaLab [68], and TopCoder [50], along with evaluation-as-a-service platforms [39], have contributed significantly. Several specialized benchmarking tools have emerged to enhance machine learning model evaluation. The Ludwig Benchmarking Toolkit [64] offers a lightweight,

[1]Documentation and a video explaining how to use CausalBench are available at https://docs.causalbench.org.

customizable framework for deep learning assessment. MLModelScope [20] unifies benchmarking across hardware platforms, focusing on latency and throughput. OpenML [97] facilitates dataset and model sharing for collaboration and reproducibility. AMLB [32] evaluates AutoML systems across tasks and frameworks. Weights and Biases (*W&B*) [10] integrates experiment tracking, real-time visualization, and hyperparameter optimization.

2.3 Causal Benchmarks

One of the earliest attempts to standardize causal discovery benchmarking was the Tübingen Cause-Effect Pairs dataset [62]. This dataset contains 100 real-world cause-effect variable pairs spanning domains such as biology, economics, and physics. CauseMe [63] is an online system for benchmarking causal discovery methods. It offers both synthetic and real-world datasets with known causal structures, allowing researchers to evaluate causal discovery algorithms in controlled and real-world scenarios. More recently, OCDB (Open Causal Discovery Benchmark) [106] was proposed as a more structured benchmarking framework. However, it remains limited to static causal discovery and does not extend to effect estimation or temporal inference. Addressing the need for temporal causal benchmarking, CausalTime [14] introduced a dataset generation pipeline that creates realistic time-series data with ground-truth causal graphs. CausalRivers [31] represents an effort to scale causal discovery benchmarking to real-world time-series data. It consists of a large dataset of river discharge measurements spanning multiple years with fine-grained temporal resolution.

3 CausalBench Framework

Despite the efforts outlined in the previous section, the field still lacks a unified, publicly available, and configurable platform that supports all major causal inference tasks, including causal discovery, causal effect estimation, and causal inference.

3.1 CausalBench Desiderata

We first introduce the desiderata driving the design of CausalBench:

- **Reproducibility** - As outlined in the introduction, one of the critical challenges faced in the research community is reproducibility of experiments. Even with the current best efforts to provide detailed experiment setups, a change in a single driver or library can cause significant differences in the results. An ideal benchmarking platform would document every aspect of an experiment, from the data to the hardware/software configuration of the system used for running the experiments.
- **Ease of use** - Providing and matching every aspect of an experimental setup is costly. Therefore, while thoroughly documenting the setup and results, specifying and benchmarking contexts should be straightforward and seamlessly integrated into regular experimental workflows without adding significant overhead.
- **Transparency** - For the community to trust the results included in the benchmark, the platform should provide a transparent mechanism to track and log an experiment, whether on the data, the model, or the experiment context itself.
- **Fairness and Explainability** - Perfect replication of an experiment is an unattainable ideal. No matter how meticulous two research teams are, they cannot perfectly replicate hardware, software, and hyperparameter configurations. Therefore, when comparing experimental results, it is crucial to identify and explain any differences in experimental setups that can explain differences in the outcomes.

3.2 CausalBench – Formal Underpinnings

CausalBench includes several core components. These include **datasets**, $\mathcal{D}$, which are data files and configuration files that describe the properties of the data in the data files; **models**, $\mathcal{M}$, which are algorithms written in Python that take in a dataset and execute a particular model, producing outputs based on the tasks and models; and **metrics** $\mathcal{A}$, which are Python implementation of metric calculations that take in the outputs provided by the model and output a numerical value, based on its configuration. CausalBench follows a flexible approach, where datasets, models, and metrics can be re-used for different causal machine learning tasks. The set of all causal machine learning tasks available at CausalBench is denoted as $\mathcal{T}$. Given the above, a **benchmark context**, C, includes a subset (denoted by the subscript $_P$) of datasets $\mathcal{D}$, models $\mathcal{M}$ and accuracy metrics $\mathcal{A}$, along with the appropriate parameter and hyperparameter settings:

$$C = \{(\mathcal{D}_P, \mathcal{M}_P, \mathcal{A}_P, \mathcal{H}_P), \mathcal{D}_P \subseteq \mathcal{D},\ \mathcal{M}_P \subseteq \mathcal{M},\ \mathcal{A}_P \subseteq \mathcal{A}\}.$$

Above, $\mathcal{H}_P$ denotes the set of **parameter and hyperparameter settings** applicable to the execution or training of the models.

Note that the benchmark context can equivalently be seen as a set of **benchmark scenarios**:

$$C = \{(d, m, \mathcal{A}_P, h) \mid d \in \mathcal{D}_P,\ m \in \mathcal{M}_P,\ h \in \mathcal{H}_P\}.$$

An **instrumented context**, $\mathcal{I}$, is a coupling of these benchmark scenarios with a particular user hardware/software system, s:

$$\mathcal{I}(C, s) = \{(d, m, \mathcal{A}_P, h, s) \mid d \in \mathcal{D}_P,\ m \in \mathcal{M}_P, ; h \in \mathcal{H}_P\}.$$

A **benchmark run**, $\mathcal{R}(\mathcal{I}(C, s))$, then, is the recording of the outputs of the execution of the benchmark scenarios in an instrumented context, $\mathcal{I}$:

$$\{(A, T, S; d, m, h, s) \mid (d, m, \mathcal{A}_P, h, s) \in \mathcal{I}(C, s)\},$$

where A is a set of key-value pairs recording the value for each accuracy metric $a \in \mathcal{A}_P$. T is a set of key-value pairs recording the timing values for each timing metrics, such as *CPU-time*, *GPU-time*; and S is a set of key-value pairs recording the system usage values for each resource metrics, such as *CPU-memory*, *GPU-memory*. Noting that the timing metrics T and resource metrics S are measured for each benchmark scenario.

3.3 CausalBench System Modules

CausalBench stores datasets, models, and metrics along with authenticated benchmark runs of its users in public or private repositories (Figure 1):

- A web-based dataset, model, and metric registration module provides a guided interface through which a provider registers a dataset, a model, or a metric with CausalBench. Registration involves the systematic acquisition of metadata needed for the discovery, access, and use of data and models.

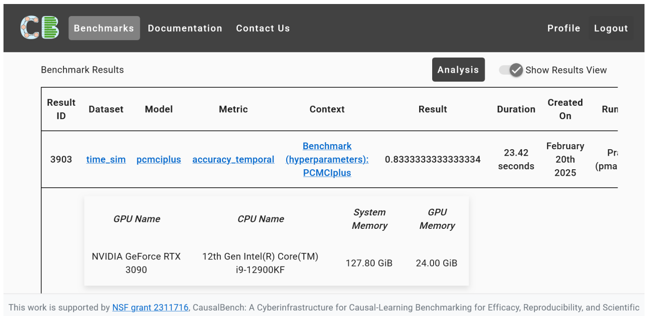

Figure 2: CausalBench runs page

- A data, model, and metric repository manages metadata associated with all registered datasets, models, and metrics and ensures that these persist and are accessible. The repository further stores (a) benchmark contexts and experiment setups consisting of data, model, and metric components and (b) authenticated performance results of benchmark runs and the associated metadata (e.g., hyperparameters, hardware/software setups).
- A benchmark runs page[2] (Figure 2) where performance results of runs, including results, system information, and a DOI attached to each benchmark run, is displayed. Experiment results are in a tabular format that can be sorted and filtered.
- A CausalBench console-based Python package supports the execution of causal machine learning experiments. The package enables quantitative evaluation of the models (for accuracy and efficiency) based on datasets in the repository using local CPU and GPU resources.
- A web interface supports browsing through repositories of datasets, models, metrics and benchmark contexts, exploring (slice-and-dice) experiments across the runs executed through CausalBench. In addition to providing data download links and data descriptions, the platform also offers accessible APIs of evaluation metrics and service interfaces.

In order to enable reproducible research on causal machine learning, once a dataset, model, or a metric is declared as public and is included in at least one public run, it becomes permanent in the system and cannot be removed. Benchmark runs that are made public are registered with an open-access repository, Zenodo [26], and are associated with a unique document object identifier (DOI).

3.4 CausalBench User Experience

CausalBench features two major components, a CausalBench library written in Python that handles execution and submission of benchmarks, and a repository that has a web front-end that provides users the existing datasets, models, metrics, benchmark contexts and the results of benchmark runs. Thanks to these two components, the user has several options available on launch: downloading/uploading a component, declaring and executing a benchmark run, and exploring existing benchmarks.

3.4.1 Uploading Data, Models, and Metrics. CausalBench repository[3] allows users to share their datasets. While these can be uploaded in the form of a manually packaged file, the CausalBench Python library, written on Python 3.10 and hosted on PyPI under the package name *causalbench-asu*[4], provides features which allow the users to tie-in the tasks of dataset publishing to their existing workflow; instead of manually bundling the dataset and uploading it as a file on the CausalBench repository, the package makes it easier for the user to upload datasets to CausalBench immediately after they have processed and cleaned their data.

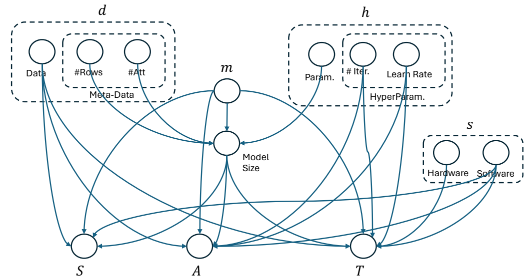

Figure 3: Outline of the causal graph enabling the causally-informed exploration and analysis of a benchmark

3.4.2 Exploring Data, Model, and Metric Repositories. Users can browse the CausalBench repository for available datasets, models, and metrics. Each component is visualized as a card, providing an overview of the relevant statistics of the components. Clicking on a card provides details and allows downloading the component. The cards corresponding to the versions of the same component are clustered and stacked.

3.4.3 Execution and Registration of Benchmark Runs. A benchmark run is essentially a benchmark scenario (a combination of datasets, models, and metrics) instrumented and executed on the user's local resources. The UI helps the user in the process of creating benchmark scenarios by filtering out incompatible components and highlighting suitable ones as the user starts declaring aspects of the benchmark scenario. This suggestion feature works based on the inputs and the outputs of each component and their task type.

CausalBench Python library, referred to earlier, enables the users to interact with the CausalBench ecosystem by executing benchmarks and submitting its results. Executing a benchmark run includes creating an instance of the benchmark scenario with current system and environment configurations on the local machine, running configurations for each combination of the core components, and uploading the execution results, including the corresponding resource usage, back to CausalBench. Once declared public, these results are registered as permanent and associated with DOIs. The Context module along with the permanancy inherent in each benchmark run generated on the CausalBench platform enables reproducibility of research findings[5].

3.4.4 Exploration of Benchmark Runs. A user can visualize and explore a benchmark run, consisting of multiple benchmark scenarios, instrumented and executed on the same hardware by the same user. The user can also visualize and explore benchmark runs

[2]Screenshots of CausalBench Runs Page and others can be found in the Appendix.
[3]https://causalbench.org/
[4]https://pypi.org/project/causalbench-asu/
[5]Results published in CausalBench can be verified in as few as four lines of Python code.

generated from the context executed on different hardware. This involves slicing and dicing a benchmark run based on the datasets and models and comparing the different metric results and resource consumption. The entire benchmark run or its various subsets can be downloaded by the user for external analysis and visualization. In addition, the user can create *virtual* benchmark runs by declaring a new benchmark context and collecting all compatible benchmark scenarios that have been instrumented, executed, and recorded in CausalBench at different times, potentially by different users. This enables the user to explore the performance of the models on different hardware/software settings.

3.4.5 Causally-Informed Explanation and Recommendation of Benchmark Runs (CausalBench-ER). Since accuracy, timing, and resource usage of the models may be impacted by the properties of the data, underlying parameter/hyper-parameter settings, as well as hardware/software configurations, CausalBench provides causally-informed services to (a) disaggregate, de-bias, and explain the various factors impacting accuracy, time, and/or resource performance of the benchmark runs, as well as (b) propose new scenarios to execute to obtain a more robust understanding of the model performance. The causally-informed exploration and analysis services provided by CausalBench includes the following:

- *Causal explanations (impact and sensitivity analysis):* The benchmark data are analyzed through a causal effect discovery algorithm [84] to quantify the impacts of various factors on the target accuracy, time, or resource usage in a given context.
- *Causal recommendations:* CausalBench aggregates the above impact analysis, ranking, and prediction services into a causally-informed recommendation service, which recommends additional benchmark configurations to execute.

Services that are not currently public, but will be included in future versions of CausalBench includes,

- *Causal ranking and exploration:* Given a set of potentially conflicting decision parameters, the causal graph can be used to identify a non-dominating (pareto-optimal) subset of the runs that best highlight/explain the underlying trade-offs.
- *Causal prediction (with knowledge transfer):* Given a causal model and a benchmark of runs, CausalBench can provide causally-informed performance predictions under new settings [15, 103]. CausalBench will tackle data sparsity through causally-informed knowledge transfer across simulation contexts, by disaggregating shareable and non-shareable information relying on the underlying causal structure.

Figure 3 provides the outline of the causal graph that forms the basis of these causally-informed explanation and recommendation services. As outlined earlier, CausalBench collects detailed profiling information during the benchmark execution step to provide transparency and enhance reproducibility across experiments. These include both easily quantifiable data, such as available memory, but also less quantifiable information, such as CPU or GPU models or package versions. Whenever possible, CausalBench-ER relies on established hardware benchmarks, such as Geekbench and Passmark [2, 3], to map these latter category of context information onto numerical performance scores for causal impact analysis.

Table 1: Summary of selected datasets for the static causal discovery case study

Dataset	Instances (Rows)	Features (Columns)	Origin
sim15-7	200	6	Simulated
sim4-47	200	51	Simulated
sim9-49	5000	6	Simulated
Abalone	4000	10	Real Life
Sachs	7000	12	Real Life

Estimating causal effects, which involves quantifying the influence of a treatment variable on an outcome, is a central challenge in causal learning. Treatments may be binary, categorical, or continuous. In this work, we focus on continuous treatments, as they are commonly encountered in benchmarking scenarios — for instance, when treatments represent hardware resource allocations or hyperparameter values. To estimate the average treatment effect (ATE), CausalBench-ER adopts linear regression with backdoor adjustment, under the assumption that a valid adjustment set is available. Figure 3 provides the outline of the causal graph that forms the basis of these causally-informed exploration and analysis services. More specifically, CausalBench-ER leverages a priori causal knowledge, described in the form of a causal graph, to boost the representational ability and achieve better explanations and recommendations. Given a causal graph (Figure 3), enriched by data-driven causal impact analysis describing the underlying causal relationships among the various factors impacting performance, CausalBench-ER provides explanations that are causally-robust.

To help obtain further insights, CausalBench-ER also provides causally-informed recommendations to its users: These recommendations can include suggestions of new experiments/scenarios to be considered (a) to strengthen the statistical strength of the current analyses or (b) to validate or refute specific hypotheses. CausalBench-ER can also (c) recommend an execution context (such as hyper-parameter settings) given a dataset, instrumentation context, and target metrics. A specific mechanism by which CausalBench-ER recommendations can improve experimentations and benchmarking relies on causal effect strengths obtained by the causal analysis process against a selected performance metric. In this case, given a dataset and a target metric, along with a target number of new benchmark runs to be executed, CausalBench-ER identifies the most causally impactful scenario settings (hyperparameters, instrumentation context parameters, etc.) and proposes a grid-search strategy that is informed by the strengths of the causal effects: scenario settings that have larger causal impacts on the target metric are more finely experimented, while avoiding new experiments that are close to existing benchmark runs. Other potential strategies include causally-informed factorization machine [51] style recommendations for new execution contexts, given a target metric to minimize or maximize.

3.5 Case Study #1 – Static Causal Discovery

In this section, we demonstrate the use of CausalBench on static causal discovery tasks.

3.5.1 Datasets. CausalBench currently boasts 1400 static datasets from different sources, including real and simulated datasets. To showcase benchmarking of static causal discovery, in this case study, we experiment with three different data contexts across five datasets; two real world datasets, Abalone and Sachs, and three simulations from the NetSim dataset to compare the performance across real world and synthetic datasets:

- *Abalone* [65] is a real life dataset that includes measurements about a group of abalones, including their length, diameter, age, and more. The ground truth is provided at [16].
- *Sachs* [83] is a dataset derived from multiparameter single-cell data. The underlying causal relationships are provided at [16].
- *NetSim* [89] provides 1440 datasets from 28 simulations of FMRI data. The underlying ground truth information is provided as adjacency matrices between nodes for each dataset. For this case study, we specifically chose sim15-7, sim4-47, and sim9-49 simulations from the NetSim dataset, as they provide the largest variance between selected datasets in terms of feature and instance sizes.

Table 1 provides a summary of selected datasets in terms of instances, features and data origin.

3.5.2 Models. We apply two causal discovery models ailable avin CausalBench on the selected datasets.

- PC [90] (Peter-Clark) algorithm is a widely known constraint-oriented causal discovery method. PC algorithm works by finding (undirected) causal relationships between variables, then directs the edges and provides a PDAG (partially directed acyclic graph) or a DAG (directed acyclic graph). PC assumes Causal Markov condition, faithfulness - no conditional independence without Markov condition is met -, no hidden confounders and cycles in the causal graph. CausalBench's PC algorithm implementation supports several conditional independence tests and *original*, *stable*, and *parallel* variants of the algorithm as hyperparameters.
- GES [60] (Greedy Equivalence Search), is a score based causal discovery algorithms that works with a forward and a backward phase where new edges are added and removed to maximize a scoring function, and returns a DAG. CausalBench's GES implementation includes BIC and BDeu scores as hyperparameters.

Both PC and GES algorithms on CausalBench are based on [105].

3.5.3 Metrics. In this case study, we employ four accuracy metrics from CausalBench to evaluate the model outputs. Specifically, we formulate causal graph evaluation as a classification task, where the presence or absence of an edge is treated as a binary classification problem. The four metrics used—accuracy, precision, recall, and F1 score—are adapted from traditional classification tasks to assess the correctness of predicted causal edges [11]. While we do not consider in this case study, CausalBench also provides other accuracy metrics, such as the *structural hamming distance (SHD)* [21].

3.5.4 Sample Results. In this section, we provide a sample subset of results from the static causal discovery benchmark with regards to metrics listed above. To replicate the results included here, the reader is advised to install the latest CausalBench repository and call the context with *module_id = 5, version = 1*:

```
from causalbench.modules import Context, Run
context_static: Context = Context(module_id=5, version=1)
run: Run = context_static.execute()
print(run)
```

Table 2: CausalBench results for static causal discovery (CBench Run ID: 34, results are recorded at [45])

Dataset	Model	Accuracy	CBench Res.ID	F1-Score	CBench Res.ID
abalone	GES	0.6049	505	**0.2727**	506
	PC	**0.6543**	509	0.2222	510
NetSim-sim15-7	GES	0.68	521	0.4285	522
	PC	**0.76**	525	**0.5714**	526
NetSim-sim4-47	GES	0.9636	529	0.4678	530
	PC	**0.9712**	533	**0.4929**	534
NetSim-sim9-49	GES	0.68	537	0.5	538
	PC	**0.76**	541	0.5	542
sachs	GES	0.6611	513	0.3050	514
	PC	**0.7768**	517	**0.4255**	518

Table 3: Sample CausalBench results for different hyperparameters (PC model, Abalone dataset) – static causal discovery (CBench Run ID: 62, results are recorded at [54])

Hyperparameters			CBench		CBench
alpha	**variant**	**Accuracy**	**Res.ID**	**F1-score**	**Res.ID**
0.0010	original	0.6914	3103	0.2424	3104
0.0010	stable	0.7037	3107	0.2500	3108
0.0531	original	0.6543	3183	0.2222	3184
0.0531	stable	0.6543	3187	0.1765	3188
0.1000	original	0.6667	3255	0.2286	3256
0.1000	stable	0.6667	3259	0.1818	3260
…	…	…	…	…	…

Table 4: Causal effects of hyperparameters on the Accuracy and F1-score metrics (PC model) - static causal discovery task

Dataset	Hyperparameter	Causal Effect on Accuracy	Causal Effect on F1-score
NetSim-sim15-7	alpha	-0.2857	-0.6171
	variant	0.0140	0.0154
NetSim-sim4-47	alpha	-0.1382	-0.8823
	variant	0.0008	-0.0025
NetSim-sim9-49	alpha	-1.1082	-2.3088
	variant	0.0000	0.0000
abalone	alpha	-0.2049	-0.3803
	variant	0.0000	-0.0389
sachs	alpha	0.0006	0.3231
	variant	0.0054	0.0060

All results reported in this section are published as benchmark records at Zenodo as a CausalBench feature [18, 45, 100]. In the tables, CausalBench execution results are identified using either a Result ID or a Run ID. A CausalBench Result ID represents the outcome of a single dataset, model, and metric on a system, providing a finegrained view of individual executions. In contrast, a CausalBench Run ID groups multiple ResultIDs within a benchmark context, encompassing an execution that includes various datasets, models, and metrics on a system.

Benchmarking Models across Datasets. As the benchmark scenario includes five datasets, two models, and four metrics, we have

Table 5: Sample CausalBench results for different system configurations (dataset: NetSim-sim4-47, model: PC, metric: Accuracy) – static causal discovery task; results are recorded at [18, 35, 45, 46, 59, 100]

CBench Res.ID	CPU	GPU	CPU (GeekBench Score) Single Core	Multi Core	Memory Available (total)	Model (used)	Metric (used)	Metric Accuracy	Time (seconds)
193	AMD Ryzen 9 7940HS	NVIDIA GeForce RTX 4070	1829	17497	31.21 GiB	0.28 MiB	0.35 MiB	0.9712	35.04
333	Apple M1 Pro	None	2387	12346	16.00 GiB	0.30 MiB	0.35 MiB	0.9712	21.55
453	Apple M3	None	1904	10454	16.00 GiB	0.30 MiB	0.35 MiB	0.9712	17.30
533	Intel(R) Core(TM) i9-12900KF	NVIDIA GeForce RTX 3090	2609	15132	127.80 GiB	0.29 MiB	0.37 MiB	0.9712	24.72
5931	AMD Ryzen 5 5625U	AMD Radeon (TM) Graphics	1372	8135	15.35 GiB	0.29 MiB	0.37 MiB	0.9712	21.29
5971	Intel(R) Core(TM) i5-8250U	NVIDIA GeForce RTX 4070	900	3091	15.52 GiB	0.28 MiB	0.35 MiB	0.9712	8.68
…	…	…	…	…	…	…	…	…	…

Table 6: Treatment effect estimation of hardware features over experiment times – static causal discovery

Hardware Feature	Causal Effect on Run Time
Used Memory	-0.0004
Single-Core Performance[7]	2.3697
Multi-Core Performance	-8.4630

a total of 40 metric evaluations. Table 2 reports F1 scores across all 5 datasets and 2 models within this static benchmark context:

- Both models perform better in simulated datasets.
- For simulated datasets, PC performs better than or equal to GES; GES performs worse in data with lower numbers of instances, but performs closer to PC as the number of instances increases.
- For real-life datasets, there is no winner between PC and GES.

Benchmarking the Effects of Hyperparamaters. CausalBench also enables us to run models with different hyperparameters. Table 3 presents sample results from multiple executions of the PC model on the Abalone dataset, each using different hyperparameter configurations. As outlined in Section 3.4.5, CausalBench enables us to analyze the causal relationships between hyperparameters and metric scores – the analysis results are presented in Table 4. The causal analysis aligns with and supports the observed trends in accuracy and F1-scores:

- The causal analysis shows that the *alpha* parameter negatively impacts the accuracy related metric scores, since it makes the model stricter, leading to overfitting.
- The choice of the *original* or *stable variants* of the PC algorithm does not have a significant causal effect on the metric scores.

Benchmarking the Effects of Computational Resources. As illustrated in Table 5, CausalBench additionally reports profiling information regarding hardware, software, and resource usage during benchmark execution[6]. CausalBench also enables causal treatment effect estimation of hardware configuration over metrics, such as experiment times. Table 6 reports sample results calculated using DoWhy[84] causal estimation model with linear regression. Here, a negative effect denotes a decrease in time. As it can be observed from the table, according to these CausalBench results, memory has a minimal effect over execution times, whereas multi-core resources have significant impact on the time complexity of these tasks.

[6]Detailed profiling information files for each execution can be accessed from the runs page at CausalBench.
[7]Single and Multi-Core performances are calculated using GeekBench 6 scores of reported CPUs across benchmarks.

Table 7: CausalBench results for temporal causal discovery (CBench Run ID: 48, results are recorded at [56])

Dataset	Model	Accur.	CBench Res.ID	F1-score	CBench Res.ID	SHD	CBench Res.ID
STLF (Panama)	VAR-LiNGAM	**0.5683**	580	**0.3122**	581	**110.5**	584
	PCMCIplus	0.4472	575	0.1705	576	141.5	579
Time Sim	VAR-LiNGAM	**0.9375**	570	0.0	571	**1.0**	574
	PCMCIplus	0.7916	565	**0.0666**	566	3.3	569

3.6 Case Study#2 - Temporal Causal Discovery

In the second case study, we demonstrate the benchmarking capabilities of CausalBench for causal discovery on time-series data, seeking a DAG describing the underlying temporal causal structure, which may include causal relationships with time lags.

3.6.1 Datasets. For this case study, we consider a real (Panama [5]) and a synthetic (Time Sim [66]) dataset:

- *Short-term electricity load forecasting (Panama)* [5] - This data is framed on predicting the short-term electricity load forecasting (STLF) problem for the Panama power system. The forecasting horizon is one week, with hourly steps, with a total of 168 hours. The dataset includes historical load, a vast set of weather variables, holidays, and historical load weekly forecast features.
- *Time Sim* [66] - This is a simulated time-series dataset comprising 4 variates and 999 tuples. The data is generated based on an pre-specified underlying causal structure. This dataset is highly useful for evaluating Causal Discovery models, since it is difficult to ascertain the veracity of the provided ground truth for the underlying causal structure in time-series data.

3.6.2 Models. Models performing causal discovery on time-series data must account for temporal dependencies, time lags, and potential feedback loops. Several classes of models have been developed for this task, leveraging techniques from graphical modeling, structural equation modeling, and deep learning [33]. In this case study, we consider two causal discovery algorithms for time-series data, VAR-LiNGAM [41] and PCMCIplus [80]:

- *VAR-LiNGAM* [41], Vector Autoregressive Linear Non-Gaussian Acyclic Model, is an extension of the LiNGAM (Linear Non-Gaussian Acyclic Model) [88] framework, designed for causal discovery in time-series data. It combines elements of vector autoregression (VAR) with non-Gaussianity assumptions to infer causal relationships between variables.

Table 8: Sample CausalBench results for different system configurations (dataset: STLF (Panama), model: PCMCIplus, metric: Accuracy) – temporal causal discovery task; results are recorded at [17, 34, 44, 56, 57, 99]

CBench	CPU	GPU	CPU (GeekBench Score)		Memory			Metric	Time
Res.ID			Single Core	Multi Core	Available (total)	Model (used)	Metric (used)	Accuracy	(seconds)
575	AMD Ryzen 5 5625U	AMD Radeon (TM) Graphics	1372	8135	15.35 GiB	44.26 MiB	0.16 MiB	0.4472	54.21
595	AMD Ryzen 9 7940HS	NVIDIA GeForce RTX 4070	1829	17497	31.21 GiB	44.25 MiB	0.17 MiB	0.4472	90.06
615	Intel(R) Core(TM) i9-12900KF	NVIDIA GeForce RTX 3090	2609	15132	127.80 GiB	44.35 MiB	0.16 MiB	0.4472	49.18
635	Intel(R) Core(TM) i5-8250U	NVIDIA GeForce RTX 4070	900	3091	15.52 GiB	44.76 MiB	0.15 MiB	0.4472	87.06
663	Apple M3	None	1904	10454	16.00 GiB	44.76 MiB	0.15 MiB	0.4472	47.91
683	Apple M1 Pro	None	2387	12346	16.00 GiB	44.82 MiB	0.15 MiB	0.4472	59.19
…	…	…	…	…	…	…	…	…	…

Table 9: Treatment effect estimation of hardware features over execution times – temporal causal discovery task

Hardware Feature	Causal Effect on Run Time
Used Memory	0.0016
Single-Core Performance	-14.6802
Multi-Core Performance	-3.1700

Table 10: Sample CausalBench results for different hyperparameters for PCMCIplus model on the STLF (Panama) dataset – temporal causal discovery task (CBench Run ID: 63, results are recorded at [55])

Hyperparameters			CBench		CBench		CBench
alpha	max_conds_dim	Accur.	Res.ID	F1-score	Res.ID	SHD	Res.ID
0.0062	6	0.4551	4978	0.1860	4979	139.5	4982
0.0166	1	0.4648	5053	0.2559	5054	137.0	5057
0.0323	10	0.4512	5248	0.1944	5249	140.5	5252
0.0479	6	0.4512	5378	0.2043	5379	140.5	5382
0.0583	6	0.4492	5478	0.2032	5479	141.0	5482
0.0635	9	0.4492	5543	0.2000	5544	141.0	5547
0.0792	9	0.4512	5693	0.2043	5694	140.5	5697
0.0896	1	0.4707	5753	0.2664	5754	135.5	5757
…	…	…	…	…	…	…	…

Table 11: Causal analysis capturing the causal effect of hyperparameters on metrics for PCMCIplus model – temporal causal discovery task

Dataset	Hyperparameter	Causal Effect		
		on Accuracy	on F1-score	on SHD
STLF (Panama)	alpha	-0.0030	0.3113	0.7648
	max_conds_dim	-0.0011	-0.0064	0.2727
Time Sim	alpha	-0.4119	-0.1323	6.5897
	max_conds_dim	0.0000	0.0000	0.0000

- *PCMCIplus* [80] – Peter-Clark Momentary Conditional Independence (PCMCI [82]) is a model that uses conditional independence tests to infer causal relationships in time-series data. It focuses on detecting direct (momentary) dependencies between variables at each time point. PCMCIplus (Peter-Clark Momentary Conditional Independence Plus) is an extension of PCMCI that improves its scalability and robustness.

3.6.3 Metrics. We evaluate the performance of the models based on five metrics. To measure graph similarity of the discovered causal graph with the original graph, we use accuracy, precision, recall, F1 score, and the average SHD metric [4, 95] – note that, unlike the static scenarios, causal discovery on time-series data yields multiple adjacency matrices, one for each time lag; therefore, causal metrics need to account these lags.

3.6.4 Sample Results. In this section, we provide a sample subset of results from the temporal causal discovery benchmark with regards to metrics listed above. This benchmark can be replicated by installing the latest `causalbench-asu` python package and executing the context with `module_id=6` and `version=3` as follows:

```
from causalbench.modules import Context, Run
context1: Context = Context(module_id=6, version=3)
run: Run = context1.execute()
print(run)
```

As before, results reported in this section are published as benchmark records at Zenodo as a CausalBench feature [18, 45, 100].

Benchmarking Models across Datasets A subset of the CausalBench results are reported in Table 7:

- We observe that VAR-LiNGAM outperforms PCMICIplus for both Panama and Time Sim datasets. This is in line with our expectations – the datasets have small number of features and tuples and the underlying causal relationships are linear, which is better captured by VAR-LiNGAM.
- We also observe that PCMCIplus is slower than VAR-LiNGAM. This also corraborates prior research [42]: PCMCIplus is a more complex model and performs conditional independence tests at multiple time lags and for each pair of variables, which can be computationally expensive.

Benchmarking the Effects of Hyperparameters As before, CausalBench enables us to study the effects of hyperparameters on selected metrics. Sample results are presented in Tables 10 and 11:

- We observe that increasing the *alpha* hyperparameter results in worse accuracy scores. This is expected, as the *alpha* parameter controls the significance level for conditional independence tests and increasing the value of *alpha* makes the model less stringent, allowing for more causal edges to be retained.
- We also observe that the *max_conds_dim* parameter does not have any significant effect on the accuracy results.

Table 12, then, provides a sample of additional benchmark run recommendations suggested by CausalBench-ER, based on the causal analysis in Table 11 for dataset STFL and target metric *Accuracy*: the numbers of new experiments recommended for parameters *alpha* and *max_conds_dim* are proportional to their causal effects on the target metric, *Accuracy*.

Table 12: Sample experiment recommendations from CausalBench-ER, on Dataset: STLF, Model: PCMCIplus, and Metric: accuracy_temporal

DS. ID	DS. Version	Model. ID	Model. Version	Metric. ID	Metric. Version	HP. alpha	HP.max _conds_dim
1447	2	4	1	2	2	0.0	6
1447	2	4	1	2	2	0.0	12
1447	2	4	1	2	2	0.0	17
1447	2	4	1	2	2	0.1	6
1447	2	4	1	2	2	0.1	12
1447	2	4	1	2	2	0.1	17
1447	2	4	1	2	2	0.2	1
...	...	...	...	...	...	...	...

Benchmarking the Effects of Computational Resources CausalBench records various other hardware information, such as CPU usage, GPU usage, memory usage, and disk usage. Table 8 presents sample results. Causal analysis of the effect of hardware configurations on the execution time based on the causal graph in Section 3.4.5 is captured in Table 9:

- We observe that the hardware configuration has no impact on the accuracies, but that they impact the execution times.
- We once again see that while memory does not appear to impact the execution time performance, both single- and multi-core CPU performance have significant effects on the execution times. In particular, single-core performance out-weights multi-core performance in terms of their impacts on execution times. These results indicate that the implementations of both VAR-LiNGAM and PCMCIplus models used in these experiments are primarly CPU-bound and are not well optimized to utilize multiple cores.

4 Conclusions

In this paper, we showcased CausalBench, a benchmarking platform facilitating open-source, adaptable, flexible, and scalable assessment of causal discovery methods. It allows users to create, execute, and publish benchmarks across various datasets, metrics, and hardware. In our future work, we aim to expand CausalBench by incorporating causal inference, causality aware machine learning downstream tasks, and a more extensive causally-informed experiment design and benchmark exploration and analysis tools. Enhancements will include causal explanations for benchmarking insights and streamlined user experience via web-based and console applications. We also plan to introduce scalability improvements and community-driven benchmarking tools to foster collaboration. CausalBench aspires to become the standard platform for causal learning evaluation, driving advancements in data-driven decision-making across critical domains.

References

[1] [n. d.]. CausalBench: Benchmarking Causal Inference Methods. http://causalbench.org. Accessed: 2025-06-10.
[2] [n. d.]. Home - Geekbench — browser.geekbench.com. https://browser.geekbench.com/. [Accessed 13-06-2025].
[3] [n. d.]. PassMark Software - CPU Benchmarks — cpubenchmark.net. https://www.cpubenchmark.net/. [Accessed 13-06-2025].
[4] Silvia Acid and Luis M de Campos. 2003. Searching for Bayesian network structures in the space of restricted acyclic partially directed graphs. *Journal of artificial intelligence research* 18 (2003), 445–490.
[5] Ernesto Aguilar Madrid and Nuno Antonio. 2021. Short-Term Electricity Load Forecasting with Machine Learning. *Information* 12, 2 (2021). https://www.mdpi.com/2078-2489/12/2/50
[6] Sahara Ali, Uzma Hasan, Xingyan Li, Omar Faruque, Akila Sampath, Yiyi Huang, Md Osman Gani, and Jianwu Wang. 2024. Causality for Earth Science–A Review on Time-series and Spatiotemporal Causality Methods. *arXiv preprint arXiv:2404.05746* (2024).
[7] Andrew Arnold, Yan Liu, and Naoki Abe. 2007. Temporal causal modeling with graphical granger methods. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*. 66–75.
[8] Arthur Asuncion, David Newman, et al. 2007. UCI machine learning repository.
[9] Debabrata Basu*. 2011. Randomization analysis of experimental data: the Fisher randomization test. *Selected Works of Debabrata Basu* (2011), 305–325.
[10] Lukas Biewald. 2020. Experiment Tracking with Weights and Biases. https://www.wandb.com/ Software available from wandb.com.
[11] David C Blair. 1979. Information retrieval, cj van rijsbergen. london: Butterworths; 1979: 208 pp. *Journal of the American Society for Information Science* 30, 6 (1979), 374–375.
[12] Stuart W Card. 2024. Bridging the AI/ML gap with explainable symbolic causal models using information theory. In *Disruptive Technologies in Information Sciences VIII*, Vol. 13058. SPIE, 1305802.
[13] Lu Cheng, Ruocheng Guo, Raha Moraffah, Paras Sheth, K. Selçuk Candan, and Huan Liu. 2022. Evaluation Methods and Measures for Causal Learning Algorithms. *IEEE Transactions on Artificial Intelligence* 3, 6 (2022), 924–943. doi:10.1109/TAI.2022.3150264
[14] Yuxiao Cheng, Ziqian Wang, Tingxiong Xiao, Qin Zhong, Jinli Suo, and Kunlun He. 2023. CausalTime: Realistically Generated Time-series for Benchmarking of Causal Discovery. *arXiv preprint arXiv:2310.01753* (2023).
[15] Yoonhyuk Choi, Jiho Choi, Taewook Ko, Hyungho Byun, and Chong-Kwon Kim. 2022. Based Domain Disentanglement without Duplicate Users or Contexts for Cross-Domain Recommendation. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 293–303.
[16] cmu phil. [n. d.]. example-causal-datasets/real/abalone at main · cmu-phil/example-causal-datasets — github.com. https://github.com/cmu-phil/example-causal-datasets/tree/main/real/abalone. [Accessed 15-02-2025].
[17] Ertugrul Coban. 2025. Benchmark run results by Ertugrul Coban, on benchmark context Benchmark: VAR-LiNGAM, PCMCIplus v3. doi:10.5281/zenodo.14880474
[18] Ertugrul Coban. 2025. Benchmark run results by Ertugrul Coban, on benchmark context CB-StaticDiscovery v1. doi:10.5281/zenodo.14876460
[19] Conference and Labs of the Evaluation Forum. [n. d.]. CLEF 2025. https://clef2025.clef-initiative.eu/. Accessed: 2025-02-15.
[20] Abdul Dakkak, Cheng Li, Jinjun Xiong, and Wen mei Hwu. 2020. MLModelScope: A Distributed Platform for Model Evaluation and Benchmarking at Scale. arXiv:2002.08295 [cs.DC] https://arxiv.org/abs/2002.08295
[21] Martijn de Jongh and Marek J Druzdzel. 2009. A comparison of structural distance measures for causal Bayesian network models. *Recent advances in intelligent information systems, challenging problems of science, computer science series* (2009), 443–456.
[22] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 248–255.
[23] Li Deng. 2012. The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE signal processing magazine* 29, 6 (2012), 141–142.
[24] Otis Dudley Duncan. 1966. Path analysis: Sociological examples. *American journal of Sociology* 72, 1 (1966), 1–16.
[25] Imme Ebert-Uphoff and Yi Deng. 2014. Causal discovery from spatio-temporal data with applications to climate science. In *2014 13th International Conference on Machine Learning and Applications*. IEEE, 606–613.
[26] European Organization For Nuclear Research and OpenAIRE. 2013. Zenodo. doi:10.25495/7GXK-RD71
[27] Andrey Feuerverger, Yu He, and Shashi Khatri. 2012. Statistical significance of the Netflix challenge. (2012).
[28] Ronald A Fisher. 1922. On the mathematical foundations of theoretical statistics. *Philosophical transactions of the Royal Society of London. Series A, containing papers of a mathematical or physical character* 222, 594-604 (1922), 309–368.
[29] Peter Flach. 2019. Performance evaluation in machine learning: the good, the bad, the ugly, and the way forward. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 33. 9808–9814.
[30] David A Freedman. 1987. As others see us: A case study in path analysis. *Journal of educational statistics* 12, 2 (1987), 101–128.
[31] Stein Gideon, Shadaydeh Maha, Penzel Niklas, and Joachim Denzler. 2025. CausalRivers - Scaling up benchmarking of causal discovery for real-world time-series. In *The Thirteenth International Conference on Learning Representations*. https://openreview.net/forum?id=wmV4cIbgl6
[32] Pieter Gijsbers, Marcos L. P. Bueno, Stefan Coors, Erin LeDell, Sébastien Poirier, Janek Thomas, Bernd Bischl, and Joaquin Vanschoren. 2023. AMLB: an AutoML

Benchmark. arXiv:2207.12560 [cs.LG] https://arxiv.org/abs/2207.12560

[33] Chang Gong, Chuzhe Zhang, Di Yao, Jingping Bi, Wenbin Li, and YongJun Xu. 2024. Causal discovery from temporal data: An overview and new perspectives. *Comput. Surveys* 57, 4 (2024), 1–38.

[34] Abhinav Gorantla. 2025. Benchmark run results by Abhinav Gorantla, on benchmark context Benchmark: VAR-LiNGAM, PCMCIplus v3. doi:10.5281/zenodo.14880142

[35] Abhinav Gorantla. 2025. Benchmark run results by Abhinav Gorantla, on benchmark context CB-StaticDiscovery v1. doi:10.5281/zenodo.14912185

[36] Clive WJ Granger. 1969. Investigating causal relations by econometric models and cross-spectral methods. *Econometrica: journal of the Econometric Society* (1969), 424–438.

[37] Ruocheng Guo, Lu Cheng, Jundong Li, P Richard Hahn, and Huan Liu. 2020. A survey of learning causality with data: Problems and methods. *ACM Computing Surveys (CSUR)* 53, 4 (2020), 1–37.

[38] Mark A Hall. 1999. *Correlation-based feature selection for machine learning*. Ph. D. Dissertation. The University of Waikato.

[39] Allan Hanbury, Henning Müller, Krisztian Balog, Torben Brodt, Gordon V Cormack, Ivan Eggel, Tim Gollub, Frank Hopfgartner, Jayashree Kalpathy-Cramer, Noriko Kando, et al. 2015. Evaluation-as-a-service: Overview and outlook. *arXiv preprint arXiv:1512.07454* (2015).

[40] Rick H Hoyle. 2012. *Handbook of structural equation modeling*. Guilford press.

[41] Aapo Hyvärinen, Kun Zhang, Shohei Shimizu, and Patrik O. Hoyer. 2010. Estimation of a Structural Vector Autoregression Model Using Non-Gaussianity. *Journal of Machine Learning Research* 11, 56 (2010), 1709–1731. http://jmlr.org/papers/v11/hyvarinen10a.html

[42] Ziyang Jiao, Ce Guo, and Wayne Luk. 2024. Optimizing VarLiNGAM for Scalable and Efficient Time Series Causal Discovery. *arXiv preprint arXiv:2409.05500* (2024).

[43] Kaggle. [n. d.]. Kaggle 2025. https://www.kaggle.com/. Accessed: 2025-02-15.

[44] Ahmet Kapkic. 2025. Benchmark run results by Ahmet Kapkic, on benchmark context Benchmark: VAR-LiNGAM, PCMCIplus v3. doi:10.5281/zenodo.14879931

[45] Ahmet Kapkic. 2025. Benchmark run results by Ahmet Kapkic, on benchmark context CB-StaticDiscovery v1. doi:10.5281/zenodo.14873295

[46] Ahmet Kapkic. 2025. Benchmark run results by Ahmet Kapkic, on benchmark context CB-StaticDiscovery v1. doi:10.5281/zenodo.14876438

[47] Ahmet Kapkiç, Pratanu Mandal, Shu Wan, Paras Sheth, Abhinav Gorantla, Yoonhyuk Choi, Huan Liu, and K Selçuk Candan. 2024. Introducing CausalBench: A Flexible Benchmark Framework for Causal Analysis and Machine Learning. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*. 5220–5224.

[48] Yongnam Kim and Peter Steiner. 2016. Quasi-experimental designs for causal inference. *Educational psychologist* 51, 3-4 (2016), 395–405.

[49] Alex Krizhevsky, Geoffrey Hinton, et al. 2009. Learning multiple layers of features from tiny images. (2009).

[50] Karim R Lakhani, David A Garvin, and Eric Lonstein. 2010. Topcoder (a): Developing software through crowdsourcing. *Harvard Business School General Management Unit Case* 610-032 (2010).

[51] Mao-Lin Li, K Selçuk Candan, and Maria Luisa Sapino. 2024. Causally Informed Factorization Machines. In *2024 IEEE International Conference on Big Data (BigData)*. 448–455. doi:10.1109/BigData62323.2024.10825754

[52] Christos Louizos, Uri Shalit, Joris M Mooij, David Sontag, Richard Zemel, and Max Welling. 2017. Causal effect inference with deep latent-variable models. *Advances in neural information processing systems* 30 (2017).

[53] Aurélie C Lozano, Naoki Abe, Yan Liu, and Saharon Rosset. 2009. Grouped graphical Granger modeling for gene expression regulatory networks discovery. *Bioinformatics* 25, 12 (2009), i110–i118.

[54] Pratanu Mandal. 2025. Benchmark run results by Pratanu Mandal, on benchmark context Benchmark (hyperparameters): PC v3. doi:10.5281/zenodo.14898152

[55] Pratanu Mandal. 2025. Benchmark run results by Pratanu Mandal, on benchmark context Benchmark (hyperparameters): PCMCIplus v2. doi:10.5281/zenodo.14898476

[56] Pratanu Mandal. 2025. Benchmark run results by Pratanu Mandal, on benchmark context Benchmark: VAR-LiNGAM, PCMCIplus v3. doi:10.5281/zenodo.14879924

[57] Pratanu Mandal. 2025. Benchmark run results by Pratanu Mandal, on benchmark context Benchmark: VAR-LiNGAM, PCMCIplus v3. doi:10.5281/zenodo.14880037

[58] Pratanu Mandal. 2025. Benchmark run results by Pratanu Mandal, on benchmark context Benchmark: VAR-LiNGAM, PCMCIplus v3. doi:10.5281/zenodo.14920545

[59] Pratanu Mandal. 2025. Benchmark run results by Pratanu Mandal, on benchmark context CB-StaticDiscovery v1. doi:10.5281/zenodo.14912072

[60] Christopher Meek. 1997. Graphical Models: Selecting causal and statistical models. (1 1997). doi:10.1184/R1/22696393.v1

[61] Tom M Mitchell and Tom M Mitchell. 1997. *Machine learning*. Vol. 1. McGraw-hill New York.

[62] Joris M Mooij, Jonas Peters, Dominik Janzing, Jakob Zscheischler, and Bernhard Schölkopf. 2016. Distinguishing cause from effect using observational data: methods and benchmarks. *Journal of Machine Learning Research* 17, 32 (2016), 1–102.

[63] J Munoz-Marí, G Mateo, J Runge, and G Camps-Valls. 2020. CauseMe: An online system for benchmarking causal discovery methods. *In Preparation* (2020).

[64] Avanika Narayan, Piero Molino, Karan Goel, Willie Neiswanger, and Christopher Ré. 2021. Personalized Benchmarking with the Ludwig Benchmarking Toolkit. arXiv:2111.04260 [cs.LG] https://arxiv.org/abs/2111.04260

[65] Warwick J Nash, Tracy L Sellers, Simon R Talbot, Andrew J Cawthorn, and Wes B Ford. 1994. The population biology of abalone (haliotis species) in tasmania. i. blacklip abalone (h. rubra) from the north coast and islands of bass strait. *Sea Fisheries Division, Technical Report* 48 (1994), p411.

[66] Meike Nauta, Doina Bucur, and Christin Seifert. 2019. Causal Discovery with Attention-Based Convolutional Neural Networks. *Machine Learning and Knowledge Extraction* 1, 1 (2019), 312–340.

[67] Leland Gerson Neuberg. 2003. Causality: models, reasoning, and inference, by judea pearl, cambridge university press, 2000. *Econometric Theory* 19, 4 (2003), 675–685.

[68] Adrien Pavao, Isabelle Guyon, Anne-Catherine Letournel, Dinh-Tuan Tran, Xavier Baro, Hugo Jair Escalante, Sergio Escalera, Tyler Thomas, and Zhen Xu. 2023. Codalab competitions: An open source platform to organize scientific challenges. *Journal of Machine Learning Research* 24, 198 (2023), 1–6.

[69] Judea Pearl. 1985. Bayesian netwcrks: A model cf self-activated memory for evidential reasoning. In *Proceedings of the 7th conference of the Cognitive Science Society, University of California, Irvine, CA, USA*. 15–17.

[70] Judea Pearl. 1995. Causal diagrams for empirical research. *Biometrika* 82, 4 (1995), 669–688.

[71] Judea Pearl. 1998. Graphs, causality, and structural equation models. *Sociological Methods & Research* 27, 2 (1998), 226–284.

[72] Judea Pearl. 2009. Causal inference in statistics: An overview. (2009).

[73] Judea Pearl. 2009. *Causality*. Cambridge university press.

[74] Judea Pearl and Thomas S Verma. 1995. A theory of inferred causation. In *Studies in Logic and the Foundations of Mathematics*. Vol. 134. Elsevier, 789–811.

[75] Charles S Reichardt. 2002. Experimental and quasi-experimental designs for generalized causal inference.

[76] Paul R Rosenbaum and Donald B Rubin. 1983. The central role of the propensity score in observational studies for causal effects. *Biometrika* 70, 1 (1983), 41–55.

[77] Donald B Rubin. 1974. Estimating causal effects of treatments in randomized and nonrandomized studies. *Journal of educational Psychology* 66, 5 (1974), 688.

[78] Donald B Rubin. 1976. Inference and missing data. *Biometrika* 63, 3 (1976), 581–592.

[79] Donald B Rubin. 2005. Causal inference using potential outcomes: Design, modeling, decisions. *J. Amer. Statist. Assoc.* 100, 469 (2005), 322–331.

[80] Jakob Runge. 2020. Discovering contemporaneous and lagged causal relations in autocorrelated nonlinear time series datasets. In *Proceedings of the 36th Conference on Uncertainty in Artificial Intelligence (UAI) (Proceedings of Machine Learning Research, Vol. 124)*, Jonas Peters and David Sontag (Eds.). PMLR, 1388–1397. https://proceedings.mlr.press/v124/runge20a.html

[81] Jakob Runge, Peer Nowack, Marlene Kretschmer, Seth Flaxman, and Dino Sejdinovic. 2019. Detecting and quantifying causal associations in large nonlinear time series datasets. *Science advances* 5, 11 (2019), eaau4996.

[82] Jakob Runge, Peer Nowack, Marlene Kretschmer, Seth Flaxman, and Dino Sejdinovic. 2019. Detecting and quantifying causal associations in large nonlinear time series datasets. *Science Advances* 5, 11 (2019), eaau4996. doi:10.1126/sciadv.aau4996 arXiv:https://www.science.org/doi/pdf/10.1126/sciadv.aau4996

[83] Karen Sachs, Omar Perez, Dana Pe'er, Douglas A. Lauffenburger, and Garry P. Nolan. 2005. Causal Protein-Signaling Networks Derived from Multiparameter Single-Cell Data. *Science* 308, 5721 (2005), 523–529. doi:10.1126/science.1105809 arXiv:https://www.science.org/doi/pdf/10.1126/science.1105809

[84] Amit Sharma and Emre Kiciman. 2020. DoWhy: An End-to-End Library for Causal Inference. *arXiv preprint arXiv:2011.04216* (2020).

[85] Paras Sheth, Raha Moraffah, K Selçuk Candan, Adrienne Raglin, and Huan Liu. 2022. Domain Generalization–A Causal Perspective. *arXiv preprint arXiv:2209.15177* (2022).

[86] Paras Sheth, Raha Moraffah, Tharindu S Kumarage, Aman Chadha, and Huan Liu. 2024. Causality guided disentanglement for cross-platform hate speech detection. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*. 626–635.

[87] Paras Sheth, Reepal Shah, John Sabo, K Selçuk Candan, and Huan Liu. 2022. Stcd: A spatio-temporal causal discovery framework for hydrological systems. In *2022 IEEE International Conference on Big Data (Big Data)*. IEEE, 5578–5583.

[88] Shohei Shimizu, Patrik O. Hoyer, Aapo Hyv228;rinen, and Antti Kerminen. 2006. A Linear Non-Gaussian Acyclic Model for Causal Discovery. *Journal of Machine Learning Research* 7, 72 (2006), 2003–2030. http://jmlr.org/papers/v7/shimizu06a.html

[89] Stephen Smith, Karla Miller, Gholamreza Salimi-Khorshidi, Matthew Webster, Christian Beckmann, Thomas Nichols, Joseph Ramsey, and Mark Woolrich.

2011. Network modeling methods for FMRI. *NeuroImage* 54 (01 2011), 875–91. doi:10.1016/j.neuroimage.2010.08.063

[90] Peter Spirtes and Clark Glymour. 1991. An Algorithm for Fast Recovery of Sparse Causal Graphs. *Social Science Computer Review* 9, 1 (1991), 62–72. doi:10.1177/089443939100900106 arXiv:https://doi.org/10.1177/089443939100900106

[91] Peter Spirtes, Clark Glymour, and Richard Scheines. 2001. *Causation, prediction, and search*. MIT press.

[92] Peter Spirtes and Kun Zhang. 2016. Causal discovery and inference: concepts and recent methodological advances. In *Applied informatics*, Vol. 3. Springer, 1–28.

[93] Harald O Stolberg, Geoffrey Norman, and Isabelle Trop. 2004. Randomized controlled trials. *American Journal of Roentgenology* 183, 6 (2004), 1539–1544.

[94] Mervyn Stone. 1974. Cross-validatory choice and assessment of statistical predictions. *Journal of the royal statistical society: Series B (Methodological)* 36, 2 (1974), 111–133.

[95] Ioannis Tsamardinos, Laura E Brown, and Constantin F Aliferis. 2006. The max-min hill-climbing Bayesian network structure learning algorithm. *Machine learning* 65 (2006), 31–78.

[96] Meliksah Turker and Haluk O Bingol. 2023. Multi-layer network approach in modeling epidemics in an urban town. *The European Physical Journal B* 96, 2 (2023), 16.

[97] Joaquin Vanschoren, Jan N. van Rijn, Bernd Bischl, and Luis Torgo. 2013. OpenML: networked science in machine learning. *SIGKDD Explorations* 15, 2 (2013), 49–60. doi:10.1145/2641190.2641198

[98] Ellen M Voorhees and Donna K Harman. 2005. The text retrieval conference. *TREC: Experiment and evaluation in information retrieval* (2005), 3–19.

[99] Shu Wan. 2025. Benchmark run results by Shu Wan, on benchmark context Benchmark: VAR-LiNGAM, PCMCIplus v3. doi:10.5281/zenodo.14881319

[100] Shu Wan. 2025. Benchmark run results by Shu Wan, on benchmark context CB-StaticDiscovery v1. doi:10.5281/zenodo.14873503

[101] Shu Wan, Reepal Shah, Qi Deng, John Sabo, Huan Liu, and K Selçuk Candan. 2024. Spatio-temporal Causal Learning for Streamflow Forecasting. In *2024 IEEE International Conference on Big Data (BigData)*. IEEE, 6161–6170.

[102] Lijing Wang, Aniruddha Adiga, Jiangzhuo Chen, Adam Sadilek, Srinivasan Venkatramanan, and Madhav Marathe. 2022. Causalgnn: Causal-based graph neural networks for spatio-temporal epidemic forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 36. 12191–12199.

[103] Song Wei, Hanyu Zhang, Ronald Moore, Rishikesan Kamaleswaran, and Yao Xie. 2023. Transfer Learning for Causal Effect Estimation. *arXiv preprint arXiv:2305.09126* (2023).

[104] Sewall Wright. 1921. Correlation and causation. *Journal of agricultural research* 20, 7 (1921), 557.

[105] Keli Zhang, Shengyu Zhu, Marcus Kalander, Ignavier Ng, Junjian Ye, Zhitang Chen, and Lujia Pan. 2021. gCastle: A Python Toolbox for Causal Discovery. arXiv:2111.15155 [cs.LG]

[106] Wei Zhou, Hong Huang, Guowen Zhang, Ruize Shi, Kehan Yin, Yuanyuan Lin, and Bang Liu. 2024. OCDB: Revisiting Causal Discovery with a Comprehensive Benchmark and Evaluation Framework. *arXiv preprint arXiv:2406.04598* (2024).

Appendix – Sample Screenshots from the CausalBench Framework

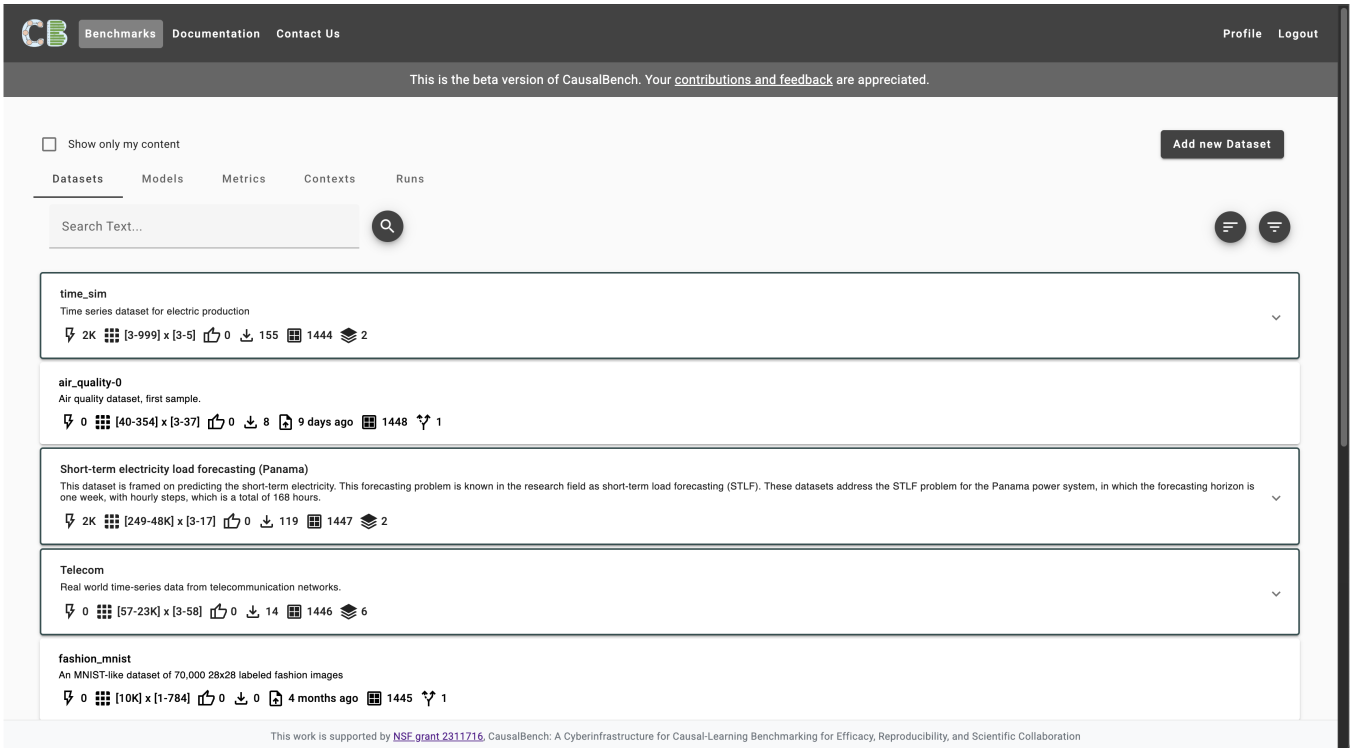

Figure 4: CausalBench "Repositories" page

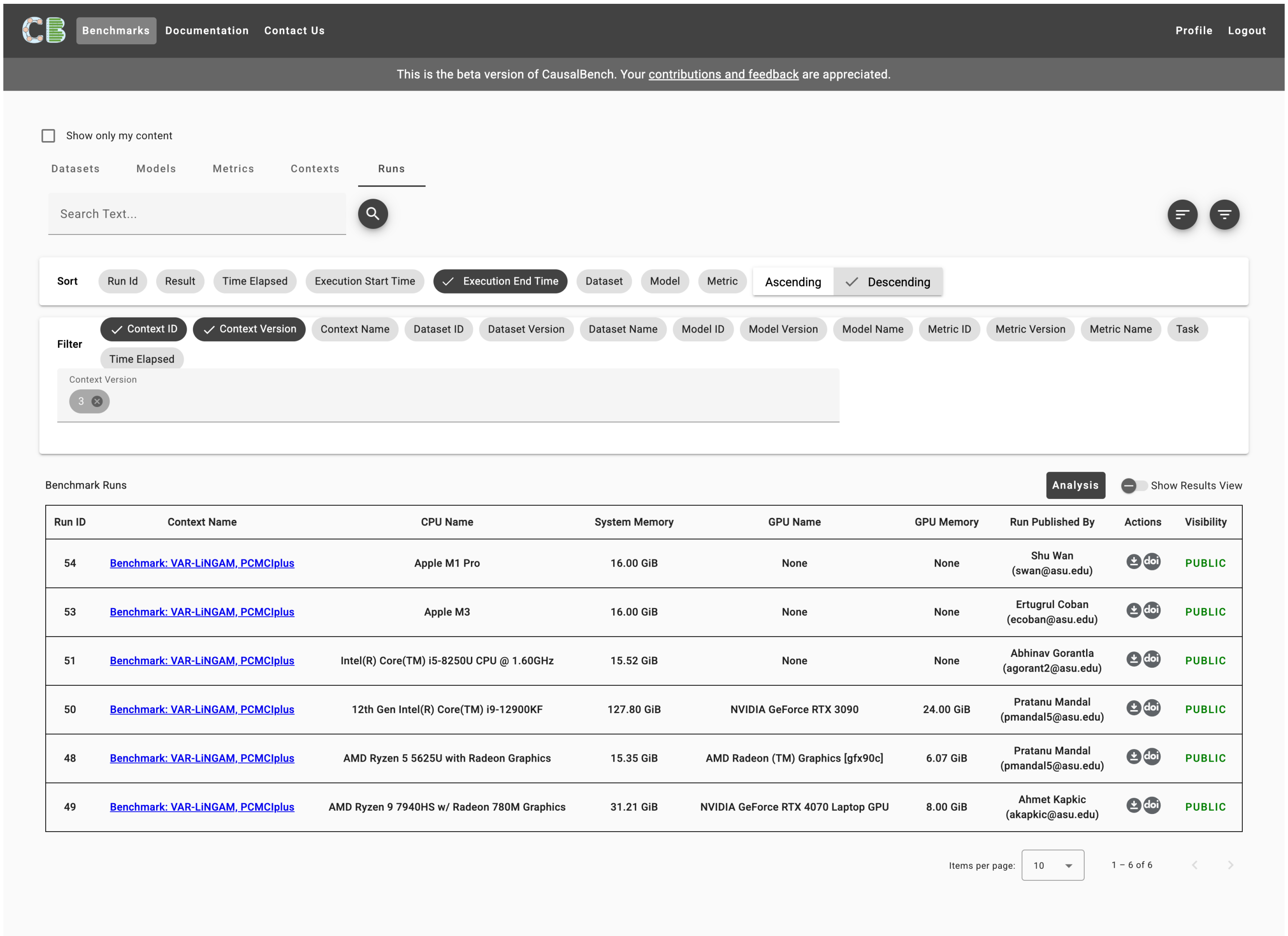

Figure 5: CausalBench "Results" page showcasing the results of a context run by multiple users

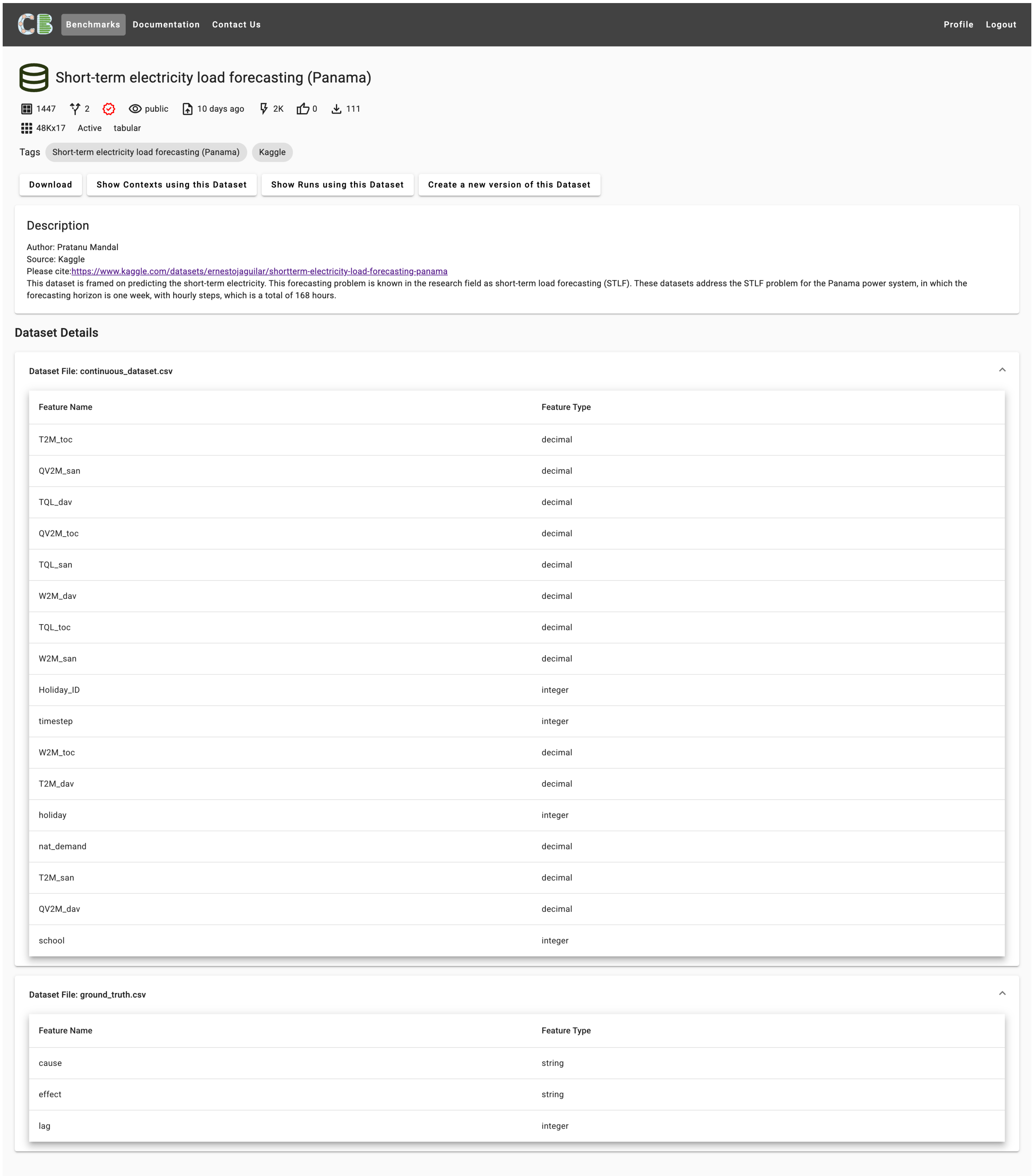

Figure 6: Details view for the Short-Term Electricity Load Forecasting (Panama) data set used in Section 3.6

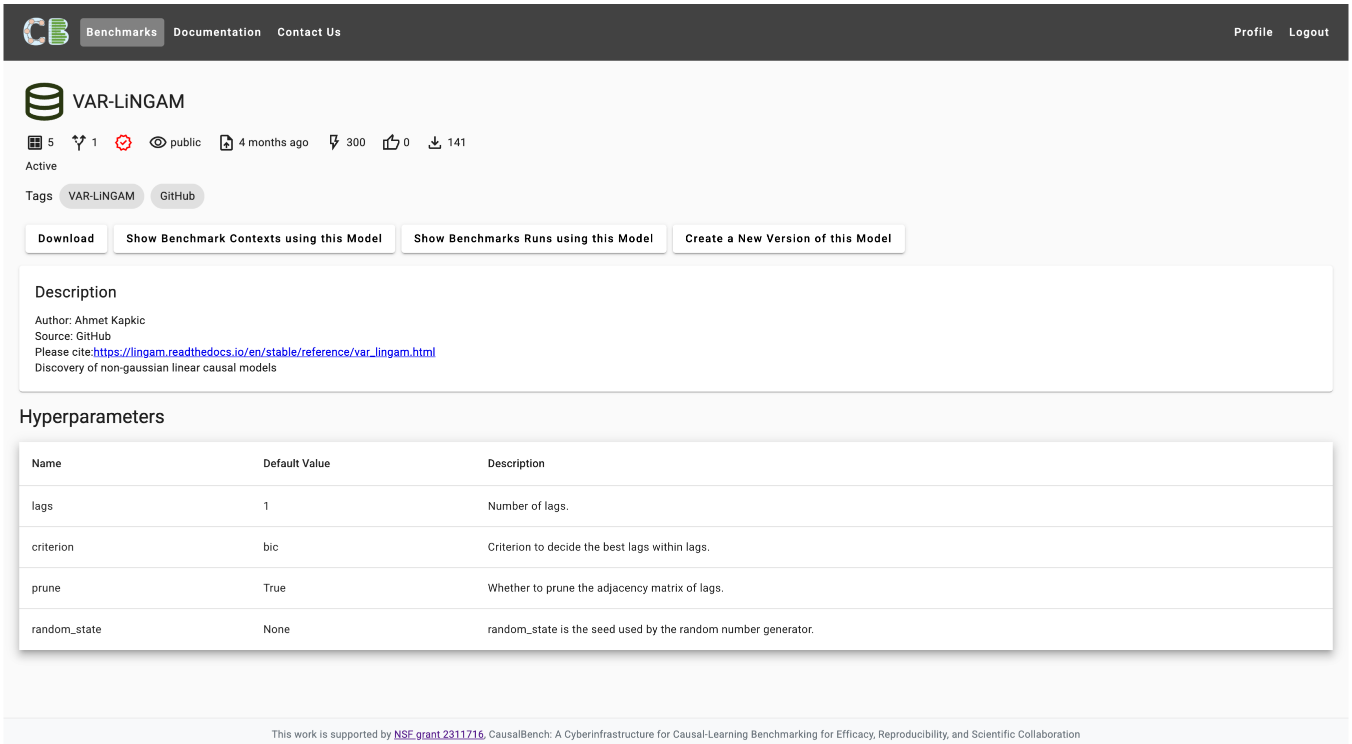

Figure 7: Details view for VAR-LiNGAM model used in Section 3.6

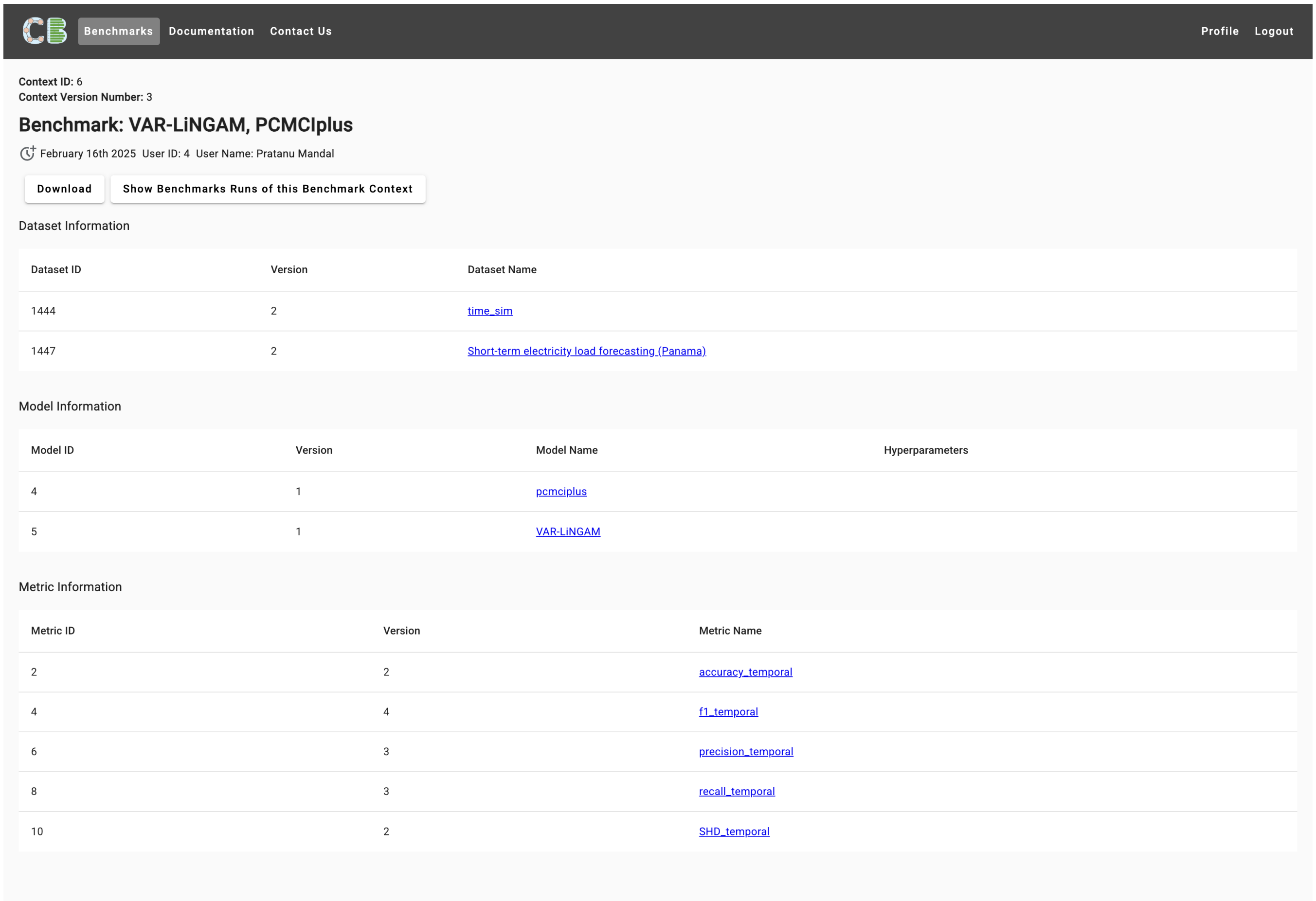

Figure 8: Details view for the benchmarking context used for the case study in Section 3.6

```yaml
context:
  id: 6
  name: 'Benchmark: VAR-LiNGAM, PCMCIplus'
  version: 3
profiling:
  cpu:
    architecture: X86_64
    name: Intel(R) Core(TM) i5-8250U CPU @ 1.60GHz
  disk:
    sda:
      fusion: null
      mediatype: SSD
      model: SanDisk X400 2.5
      usage:
        free: 43559714816
        total: 64240631808
        used: 17421107200
    sdb:
      fusion: null
      mediatype: SSD
      model: Samsung SSD 860
      usage:
        free: 0
        total: 0
        used: 0
  gpu: {}
  memory_total: 16660094976
  platform:
    architecture: 64bit
    name: Linux-6.8.0-53-generic-x86_64-with-
glibc2.39
  storage_total: 61075263488
results:
- dataset:
    id: 1444
    name: time_sim
    version: 2
  metrics:
  - hyperparameters:
      binarize: true
    id: 2
    name: accuracy_temporal
    output:
      score: '0.7916666666666666'
    profiling:
      disk:
        sda:
          read_bytes: 0
          write_bytes: 0
        sdb:
          read_bytes: 0
          write_bytes: 0
      gpu: {}
      imports:
        numpy: 1.26.4
      memory: 108464
      python: 3.11.11
    time:
      duration: 25768550
      end: 1739746789741584892
      start: 1739746789715816342
    version: 2
  - hyperparameters:
      binarize: true
    id: 4
    name: f1_temporal
    output:
      score: '0.06666666666666667'
    profiling:
      disk:
        sda:
          read_bytes: 0
          write_bytes: 0
        sdb:
          read_bytes: 0
          write_bytes: 0
      gpu: {}
      imports:
        numpy: 1.26.4
      memory: 73780
      python: 3.11.11
    time:
      duration: 24473053
      end: 1739746790842329671
      start: 1739746790817856618
    version: 4
  - hyperparameters:
      binarize: true
    id: 6
    name: precision_temporal
    output:
      score: '0.037037037037037035'
    profiling:
      disk:
        sda:
          read_bytes: 0
          write_bytes: 0
        sdb:
          read_bytes: 0
          write_bytes: 0
      gpu: {}
      imports:
        numpy: 1.26.4
      memory: 73732
      python: 3.11.11
    time:
      duration: 23907937
      end: 1739746791944269932
      start: 1739746791920361995
    version: 3

                    . . .
```


Figure 9: Partial view of sample benchmark execution results, encoded in the form of a YAML file (Context Run ID 51 [34])

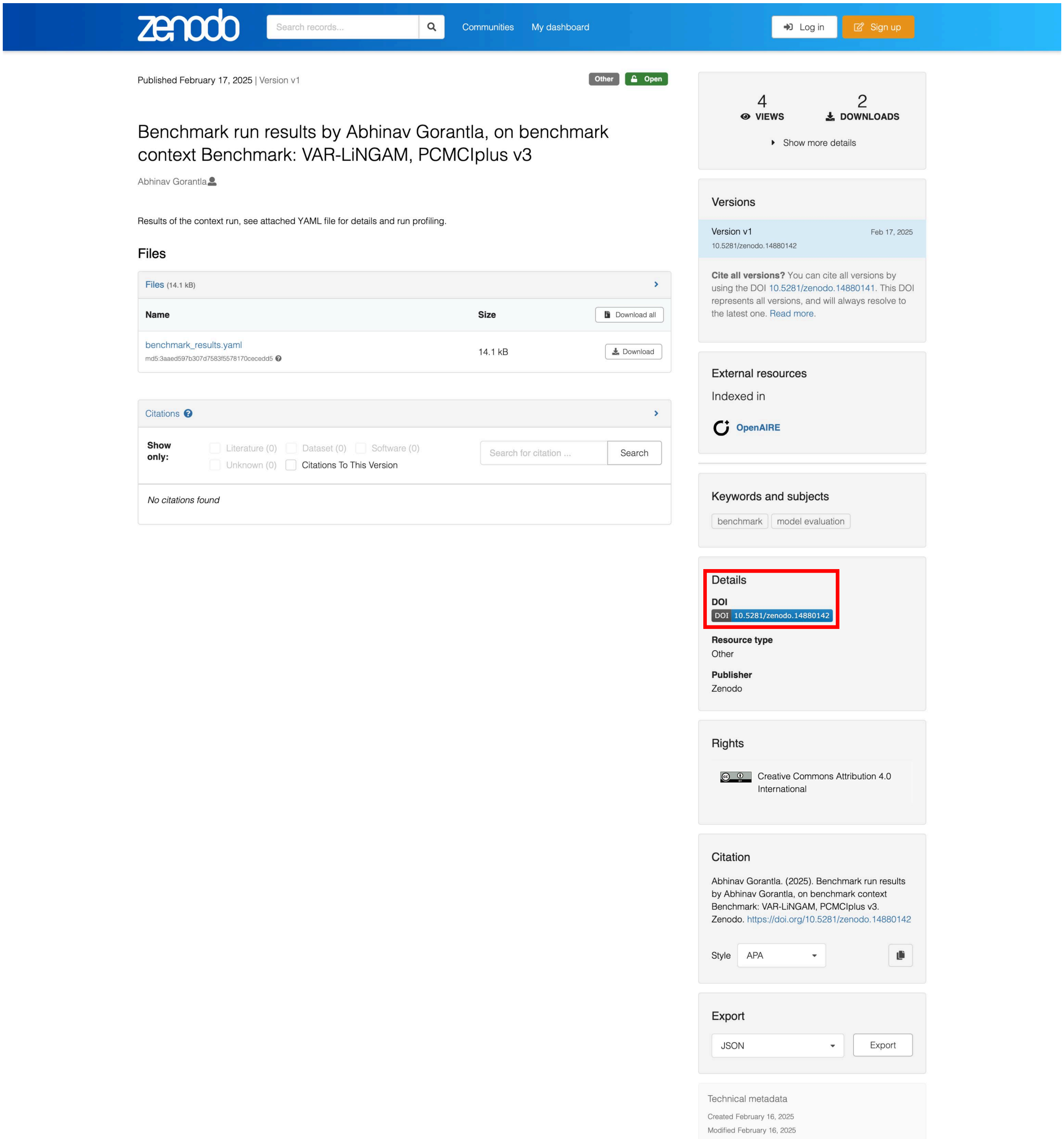

Figure 10: Results in Figure 9 uploaded for permanent storage to zenodo.org [26] – the results are assigned a permanent DOI 10.5281/zenodo.14880142

```
(causalbench) PS C:\Users\pmandal5.ASURITE\Desktop\CB> python temporal.py
Fetched context with module_id=6 and version=3
Fetched task with module_id=discovery.temporal
Fetched dataset with module_id=1444 and version=2
Fetched dataset with module_id=1447 and version=2
Fetched model with module_id=4 and version=1
Fetched model with module_id=5 and version=1
Fetched metric with module_id=2 and version=2
Fetched metric with module_id=4 and version=4
Fetched metric with module_id=6 and version=3
Fetched metric with module_id=8 and version=3
Fetched metric with module_id=10 and version=2
================================================================================
Task: discovery.temporal
--------------------------------------------------------------------------------
Dataset: time_sim
Model: pcmciplus
    tau_min: 1
    tau_max: 1
    alpha: 0.01
    contemp_collider_rule: majority
    conflict_resolution: True
    reset_lagged_links: False
    max_conds_dim: None
    max_combinations: 1
    max_conds_py: None
    max_conds_px: None
    max_conds_px_lagged: None
    fdr_method: none
Metrics:
    accuracy_temporal: 0.7916666666666666
        binarize: True
    f1_temporal: 0.06666666666666667
        binarize: True
    precision_temporal: 0.037037037037037035
        binarize: True
    recall_temporal: 0.3333333333333333
        binarize: True
    SHD_temporal: 3.3333333333333335
        binarize: True
--------------------------------------------------------------------------------
...
```

Figure 11: Screenshot of a sample context execution and partial view of the corresponding benchmark results (Context Run ID 67 [58])