One-Shot Multi-Label Causal Discovery in High-Dimensional Event Sequences

Hugo Math[1,2] **Robin Schön**[2] **Rainer Lienhart**[2]
[1] BMW Group
[2] Chair for Machine Learning and Computer Vision, Augsburg University
Augsburg, Germany
hugo.math@bmw.de robin.schoen@uni-a.de rainer.lienhart@uni-augsburg.de

Abstract

Understanding causality in event sequences with thousands of sparse event types is critical in domains such as healthcare, cybersecurity, or vehicle diagnostics, yet current methods fail to scale. We present OSCAR, a one-shot causal autoregressive method that infers per-sequence Markov Boundaries using two pretrained Transformers as density estimators. This enables efficient, parallel causal discovery without costly global CI testing. On a real-world automotive dataset with $29,100$ events and 474 labels, OSCAR recovers interpretable causal structures in minutes, while classical methods fail to scale— enabling practical scientific diagnostics at production scale.

1 Introduction

Causal discovery in event sequences is a central problem across various domains, including cybersecurity [22], healthcare [33, 13], flight operations [20], and vehicle defects [29]. These sequences of discrete events x_i recorded asynchronously over time often lead to outcomes (e.g, a diagnosed defect, a disease) denoted as labels $\boldsymbol{y}$. While becoming more available at scale, they remain challenging to interpret beyond associations. Understanding *why* specific events lead to particular outcomes is vital for effective diagnosis, prediction, and overall decision making [19, 30].

However, the majority of existing causal discovery methods remain computationally intractable in high dimensions [10, 12] with thousands of different nodes. Additionally,

practitioners frequently reason about causality *within individual unknown sequences*. For instance, "what series of events captured by diagnostics led to this vehicle failure".

We aim to solve this in a one-shot manner: given only a single unknown sequence of observed events, we directly infer the causal structure explaining its outcomes, without needing multiple repetitions or large aggregated datasets. Specifically, we seek to extract, for each label, the minimal set of causal events—its Markov Boundary.

In this work, we introduce OSCAR: the first One-Shot multi-label Causal AutoRegressive discovery method. It leverages two Transformers [40] as density estimators to extract a compact interpretable subgraph with quantified causal indicators between events and labels, providing better explainability. Unlike traditional causal discovery methods that suffer label cardinality-dependent time complexity [18, 45, 12, 10], OSCAR supports causal discovery across thousands of nodes. Thanks to its fully parallelised structure, it provides sequence-specific explainability in a matter of minutes. We validate our approach on a real-world vehicular dataset comprising 29,100 event types as diagnosis trouble codes and 474 labels as error patterns (EPs) representing vehicle defects [23].

Accepted to the NeurIPS 2025 Workshop on CauScien: Uncovering Causality in Science.

2 Related Work

Event sequences, such as diagnostic trouble codes in vehicles [29, 23] or electronic health records [33, 17], are often represented as a series of time-stamped discrete events $S = \{(t_1, x_1), \ldots, (t_L, x_L)\}$ where $0 \leq t_1 < \ldots \leq t_L$ the time of occurrence of event type $x_i \in \mathbb{X}$ drawn from a finite vocabulary $\mathbb{X}$. In multi-label settings, a binary label vector $\boldsymbol{y} \in \{0, 1\}^{|\mathbb{Y}|}$ is attached to S and denotes the presence of multiple outcome labels drawn from $\mathbb{Y}$ occuring at <u>final time step t_L</u>. Forming a multi-labeled sequence $S_l = (S, (\boldsymbol{y}_L, t_L))$.

Transformers have recently shown strong performance in high-dimensional event spaces for next-event or label prediction [40, 32, 37], including dual-architecture setups predicting both events and outcomes [23]—a structure we repurpose for causal discovery.

Neural autoregressive density estimators (NADEs) [1] factorise sequence likelihood via the chain rule, and modern Transformer-based NADEs have been applied to causal inference [9, 15] by simulating interventions or approximating Bayesian networks [9, 15].

Transformers as causal learners have been explored in sequential settings, with attention patterns interpreted as latent causal graphs [25, 24]. We extend the one-shot sequence-to-graph idea [34] to multi-label causal discovery to scale to tens of thousands of event types.

Multi-label Causal Discovery seeks to identify the Markov Boundary (**MB**) of each label—its minimal set of parents, children, and spouses—such that the label is conditionally independent of all other variables given its **MB** [38]. While classical constraint-based algorithms have shown success on low-dimensional tabular data [35, 45], their application to event sequences with multi-label outputs remains challenging due to dimensionality, sparsity, temporal dependencies, and distributional assumptions [10, 2, 45].

A comprehensive list of the notations, definitions, proofs, and assumptions used throughout the paper can be found respectively in Appendix A, B, E and C.

Working with causal structure learning from observed data requires several assumptions, notably the causal Markov assumptions [27] states that a variable is conditionally independent of its non-descendants given its parents. We assume the following: an event is allowed to influence any future events only (temporal precedence A1), event lagged effects are contained in a fixed windows (A2), the transformers model perfectly the joint probability distribution of event and labels (A4) and causal sufficiency (A3). A discussion on the impact of assumptions is provided in Appendix H.

3 Methodology

Conditional Mutual Information Estimation via Autoregressive Models. We model each multi-labeled event sequence $S_l = (S, \boldsymbol{y}_L)$ as a sequential Bayesian Network (Def. 1), over events $X_i \in \mathbb{X}$ and labels $Y_j \in \mathbb{Y}$ (Fig. 5). Our goal is to recover the **MB** of each Y_j. Specifically, we would like to assess how much additional information event X_i occurring at step i provides about label Y_j when we already know the past sequence of events $\boldsymbol{Z} = S_{<i}$. We essentially try to answer if:

$$P(Y_j|X_i, \boldsymbol{Z}) = P(Y_j|\boldsymbol{Z}) \Leftrightarrow D_{KL}(P(Y_j|X_i, \boldsymbol{Z}) \| P(Y_j|\boldsymbol{Z})) = 0$$

where D_{KL} denotes the *Kullback-Leibler divergence* [3]. The distributional difference between the conditionals $P(Y_j|X_i, \boldsymbol{Z}), P(Y_j|\boldsymbol{Z})$ is akin to Information Gain I_G [31] conditioned on past events:

$$I_G(x_i, Y_j|z_i) \triangleq D_{KL}(P(Y_j|X_i = x_i, \boldsymbol{Z} = z_i)) || P(Y_j|Z = z_i)) \tag{1}$$

Which is equals to the difference between the conditional entropies [3, 31] denoted as H:

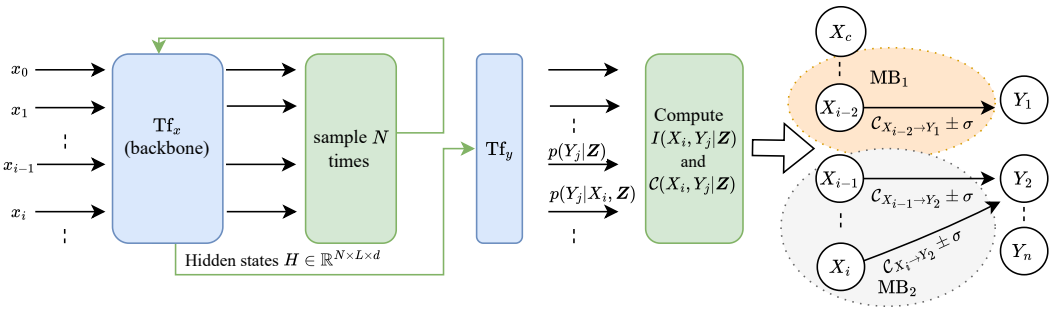
$$I_G(Y_j, x_i|z_i) = H(Y_j|z_i) - H(Y_j|x_i, z_i) \tag{2}$$

More generally, we can use the conditional mutual information (CMI) [3] as I to assess conditional independence (Def. 4). It is simply the expected value of the information gain $I_G(Y_j, x_i|z_i)$ such as:

$$I(Y_j, X_i|\boldsymbol{Z}) \triangleq H(Y_j|\boldsymbol{Z}) - H(Y_j|\boldsymbol{Z}, X_i) = \mathbb{E}_{x_i, z_i}[I_G(Y_j, X_i = x_i|\boldsymbol{Z} = z_i)]) \tag{3}$$

It can be interpreted as the expected value over all possible contexts $\boldsymbol{Z}$ of the deviation from independence of X_i, Y_j in this context. To approximate Eq. (3), a naive Monte Carlo [4] approximation

Figure 1: The overview of OSCAR: One-Shot multi-label Causal AutoRegressive discovery. d denotes the hidden dimension, L the sequence length, MB_1, MB_2 the Markov Boundary of Y_1, Y_2 respectively. All green and blue areas represent parallelised operations.



is performed where we draw N random variations of the conditioning set $z^{(l)} = \{x_0^{(l)}, \ldots, x_{i-1}^{(l)}\}$, denoting the l-th sampled particle:

$$\hat{I}_N(X_{i+1}, X_i \mid \boldsymbol{Z}) = \frac{1}{N} \sum_{l=1}^{N} I_G(X_{i+1}, X_i \mid \boldsymbol{Z} = z^{(l)}) \tag{4}$$

This estimator is unbiased because the contexts $z^{(l)}$ are sampled directly from Tf_x using a proposal Q with the same support as $P(\boldsymbol{Z})$. Since $I_G(X_{i+1}, X_i \mid \boldsymbol{Z} = z)$ is a difference between conditional entropies ((2)), it is thus bounded uniformly [3] by the log of supports such as:

$$0 < I_G(X_{i+1}, X_i \mid \boldsymbol{Z} = z^{(l)}) = H(X_{i+1}|z^{(l)}) - H(X_{i+1}|x_i, z^{(l)})) \leq H(X_{i+1}) \leq \log |\mathbb{X}|$$

Thus the posterior variance of $f_i = I_G(X_{i+1}, X_i \mid \boldsymbol{Z} = z^{(l)})$ satisfies $\sigma^2_{f_i} \triangleq \mathbb{E}_{p(z)}[f_i^2(p(z)] - I^2(f_i) < +\infty$ [4] then the variance of $\hat{I}_N(f_i))$ is equal to $var(\hat{I}_N(f_i)) = \frac{\sigma^2_{f_t}}{N}$ and from the strong law of large numbers:

$$\hat{I}_N \xrightarrow[N \to +\infty]{\text{a.s.}} \mathbb{E}_z[I_G(X_{i+1}, X_i \mid \boldsymbol{Z} = z)] \triangleq I(f_i). \tag{5}$$

Sequential Markov Boundary Recovery. In practice a per-label threshold $\theta_j \approx 0$ is applied to Eq (4) to identify conditional independence:

$$Y_j \not\perp X_i \mid \boldsymbol{Z} \quad \Leftrightarrow \quad I(X_i, Y_j \mid \boldsymbol{Z}) > \theta_j \approx 0 \tag{6}$$

θ_j is dynamically computed for each label based on the mean and standard deviation of the CMI values across the sequence such that: $\theta_j = \mu_{Y_j} + k \cdot \sigma_{Y_j}$. We analyse the effect of k in Fig. 4 as well as the effect of the proposal Q and number of particles in Appendix G.2 and G.3.

We reuse the two architectures introduced by Math et al. [23] to perform next event prediction (*CarFormer* as Tf_x) and next labels (*EPredictor* as Tf_y) with past events $\boldsymbol{Z} = (x_1, \cdots, x_{i-1})$

$$\text{Tf}_x(S_{<i}) = \textit{Softmax}(\boldsymbol{h}^x_{i-1}) = P_{\theta_x}(X_i|\boldsymbol{Z}) \tag{7}$$

$$\text{Tf}_y(S_{\leq i}) = \textit{Sigmoid}(\boldsymbol{h}^y_i) = P_{\theta_y}(Y|X_i, \boldsymbol{Z}) \tag{8}$$

Here, $\boldsymbol{h}^x_{i-1}, \boldsymbol{h}^y_i \in \mathbb{R}^d$ are the logits produced by Tf_x and Tf_y parametrized by θ_x, θ_y.

Theorem 1 (Markov Boundary Identification in Event Sequences)**.** *If S_l^k a multi-labeled sequence drawn from a dataset $D = \{S_l^1, \cdots, S_l^n\} \subset \mathbb{S}$ where two Oracle Models Tf_x and Tf_y were trained on, then under causal sufficiency (A3), bounded lagged effects (A2) and temporal precedence (A1), the Markov Boundary of each label Y_j in the causal graph $\mathbb{G}$ can be identified using conditional mutual information for CI-testing.*

Intuitively, Theorem 1 states that if our Transformers perfectly approximate the true joint distribution, then testing conditional mutual information at each step is sufficient to recover the Markov Boundary of each label sequentially. By induction, we prove that with bounded lagged effects of the previous events, we can restrict their causal influence and recover the correct **MB** of each label in the associated Bayesian Network (Fig. 5).

These guarantees rest on strong assumptions—causal sufficiency, oracle-quality density estimators, and bounded lagged effects. While these are unlikely to strictly hold in practice, they simplify identifiability and highlight where practical approximations may degrade performance (Appendix H). We particularly observed that for rare labels, the one-shot performances drastically dropped. We analyze this behavior in the experiment section in Fig. 2. One must carefully evaluate Tf_y classification performance on downstream tasks using macro metrics before applying OSCAR.

Computation. A key advantage of our approach is its scalability. Unlike traditional methods whose complexity depends on the event and label cardinality $|\mathbb{X}|$ and $|\mathbb{Y}|$ [18], our method is agnostic to both. Fig. 1 shows all parallelised steps on GPUs. CMI estimations are independently performed for all positions $i \in [c, L]$, with the sampling pushed into the batch dimension and results averaged across particles using Eq. 4. This transitions the time complexity from $\mathcal{O}(\text{BS} \times N \times L)$ to $\mathcal{O}(1)$ per batch. A Pytorch [26] implementation is given in Appendix K.

Causal Indicator. While deterministic DAGs reveal structural dependencies, they often obscure the *magnitude* and *direction* of influence between variables. Given that we can estimate conditional distributions, we define the *causal indicator* $\mathcal{C} \in [-1, 1]$ [7, 5] between an event X_i and a label Y_j under context $\boldsymbol{Z}$ that we assume fixed for every measurement [7] as:

$$\mathcal{C}(Y_j, X_i; \boldsymbol{Z}) := \mathbb{E}_Z[P(Y_j \mid X_i, Z) - P(Y_j \mid Z)] \tag{9}$$

This enables an easy interpretation, for instance, if $\mathcal{C} < 0$, then X_i inhibits the occurrence of Y_j. We employ the term causal *indicator* to separate from causal strength measures, which, if using this formulation, can be problematic [16]. Ours serves more as an indication of the rise in label likelihood after observing a certain event, rather than a strength. The causal strength, strictly speaking, is here the CMI since it serves as a CI-test to recover the correct DAG.

4 Empirical Evaluation

Comparisons. Although no existing method directly targets one-shot multi-label causal discovery [10], we benchmark OSCAR against local structure learning (LSL) algorithms that estimate global Markov Boundaries. This includes established approaches such as CMB [8], MB-by-MB [41], PCD-by-PCD [44], IAMB [39] from the *PyCausalFS* package [45], as well as the more recent, state-of-the-art MI-MCF [21]. We used a $g4dn.12xlarge$ instance from AWS Sagemaker to run comparisons, containing 4 T4 GPUs. We used a combination of F1-Score, Precision, and Recall with different averaging [46] (Appendix F.1) to perform the comparisons. The code for OSCAR, Tf_x, Tf_y and the evaluation are provided anonymously [1] as well as the anonymised version of the dataset for reproducibility purposes. An Ablation of the NADEs quality is given in G.1.

Vehicle Event Sequences Dataset. We evaluated our method on a real-world vehicular test set of $n = 300,000$ sequences. It contains $|\mathbb{Y}| = 474$ different error patterns and about $|\mathbb{X}| = 29,100$ different DTCs forming sequences of $\approx 150 \pm 90$ events. We used 105m backbones as Tf_x, Tf_y [23]. The two NADEs were not exposed to the test set during. The error patterns are manually defined by domain experts as boolean rules between DTCs in Eq (10) where (y_1) is a boolean definition based on diagnosis trouble codes (x_i):

$$y_1 = x_1 \ \& \ (x_5 \mid x_8 \ ) \ \& \ (x_{18} \mid x_{12}) \ \& \ x_3 \ \& \ (!x_{10} \mid !x_{20}) \tag{10}$$

We set the elements of this rule as the correct Markov Boundary for each label y_j in the tested sequences. It is important to note that rules are subject to changes over time by domain experts, making it more difficult to extract the true **MB**. Moreover, there is about 12% missing **MB** rules for certain Y_j.

Table 1 shows comparision with $n = 50,000$. We found out LSL algorithms failed to compute the Markov Boundaries within multiple days (3-day timeout), far exceeding practical limits for

[1] `https://github.com/Mathugo/OSCAR-One-Shot-Causal-AutoRegressive-discovery.git`

Table 1: Comparisons of **MB** retrieval with $n = 50,000$ samples, $|\mathbb{X}| = 29,100$, $|\mathbb{Y}| = 474$ averaged over $6-$folds. Classification metrics averaging is 'weighted' and shown as one-shot for OSCAR. The symbol '-' indicates that the algorithm didn't output the **MBs** under 3 days. Metrics are given in $\%$.

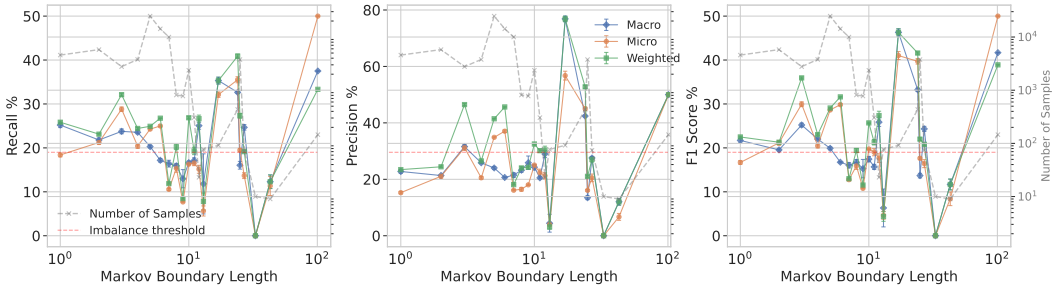
Algorithm	Precision↑	Recall↑	F1↑	Running Time (min)↓
IAMB	-	-	-	> 4320
CMB	-	-	-	> 4320
MB-by-MB	-	-	-	> 4320
PCDbyPCD	-	-	-	> 4320
MI-MCF	-	-	-	> 4320
OSCAR	$\mathbf{55.26 \pm 1.42}$	$\mathbf{31.37 \pm 0.82}$	$\mathbf{40.02 \pm 1.03}$	**11.7**

deployment. OSCAR, on the other hand, shows robust classification over a large amount of events $(29,100)$, especially 55% precision, in a matter of minutes. This behaviour highlights the current infeasibility of multi-label causal discovery in high-dimensional event sequences, since it relies on expensive global CI-testings [10]. This positions OSCAR as a more feasible approach for large-scale causal per-sequence causal reasoning in production environments.

We exemplify the explainability provided by our method for the task of explaining error patterns happening to a vehicle (Fig. 6). OSCAR's output could directly support domain expert rule refinement in diagnostics (e.g., engineers update fault detection rules), leading to a better automation of quality processes. More examples are given in the Appendix J.

Markov Boundary Length. Figure 2 reveals the classification performance depending on the number of nodes in the ground truth **MB**. On the same plot is drawn in grey the number of samples that each **MB** length contains (to account for imbalance). We observe that generally, a bigger $|\mathbf{MB}(Y_j)|$ does not imply a reduction in performance, highlighting the capability of OSCAR to retrieve complex Markov Boundaries in high-dimensional data. However, we observe that past a certain number of samples (imbalance threshold in red $\approx 7 \times 10^2$ samples), the classification metrics are directly correlated with the number of samples per $|\mathbf{MB}(Y_j)|$. This indicates that Tf_y struggles to output proper conditional probabilities, which deteriorates the CI-test when having rare classes. Therefore, when using OSCAR and more generally assumption A4, one should carefully assess class imbalance in the pretraining phase.

Figure 2: Evolution of the One-Shot Recall, Precision and F1-Score in function of the Markov Boundary length $|\mathbf{MB}(Y_j)|$ using $n = 45969$ samples.



5 Conclusion

We introduced OSCAR, a scalable one-shot causal discovery framework for high-dimensional multi-label event sequences, achieving in minutes what classical methods cannot compute in days.

We pointed out limitations, notably the oracle assumption, which might not hold for rare labels. Nevertheless, by combining pretrained autoregressive Transformers with vectorized CI-testing, OSCAR delivers, for the first time, interpretable causal graphs with quantified causal indicators for thousands of different events. This brings causal discovery closer to large-scale reasoning, making it practical for real-world sequential data.

References

[1] Y. Bengio and S. Bengio. Modeling high-dimensional discrete data with multi-layer neural networks. In S. Solla, T. Leen, and K. Müller, editors, Advances in Neural Information Processing Systems, volume 12. MIT Press, 1999. URL https://proceedings.neurips.cc/paper_files/paper/1999/file/e6384711491713d29bc63fc5eeb5ba4f-Paper.pdf.

[2] D. M. Chickering. Learning Bayesian Networks is NP-Complete, pages 121–130. Springer New York, New York, NY, 1996. ISBN 978-1-4612-2404-4. doi: 10.1007/978-1-4612-2404-4_12. URL https://doi.org/10.1007/978-1-4612-2404-4_12.

[3] T. Cover. Elements of Information Theory. Wiley series in telecommunications and signal processing. Wiley-India, 1999. ISBN 9788126508143. URL https://books.google.de/books?id=3yGJrqyanyYC.

[4] A. Doucet, N. de Freitas, and N. Gordon. An Introduction to Sequential Monte Carlo Methods, pages 3–14. Springer New York, New York, NY, 2001. ISBN 978-1-4757-3437-9. doi: 10.1007/978-1-4757-3437-9_1. URL https://doi.org/10.1007/978-1-4757-3437-9_1.

[5] E. Eells. Probabilistic Causality. Cambridge Studies in Probability, Induction and Decision Theory. Cambridge University Press, 1991.

[6] A. Fan, M. Lewis, and Y. Dauphin. Hierarchical neural story generation. In I. Gurevych and Y. Miyao, editors, Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 889–898, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1082. URL https://aclanthology.org/P18-1082/.

[7] B. Fitelson and C. Hitchcock. Probabilistic measures of causal strength. Causality in the Sciences, 01 2010. doi: 10.1093/acprof:oso/9780199574131.003.0029.

[8] T. Gao and Q. Ji. Local causal discovery of direct causes and effects. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, Advances in Neural Information Processing Systems, volume 28. Curran Associates, Inc., 2015. URL https://proceedings.neurips.cc/paper_files/paper/2015/file/fcdf25d6e191893e705819b177cddea0-Paper.pdf.

[9] S. Garrido, S. Borysov, J. Rich, and F. Pereira. Estimating causal effects with the neural autoregressive density estimator. Journal of Causal Inference, 9(1):211–228, 2021. doi: doi:10.1515/jci-2020-0007. URL https://doi.org/10.1515/jci-2020-0007.

[10] C. Gong, C. Zhang, D. Yao, J. Bi, W. Li, and Y. Xu. Causal discovery from temporal data: An overview and new perspectives. ACM Comput. Surv., 57(4), Dec. 2024. ISSN 0360-0300. doi: 10.1145/3705297. URL https://doi.org/10.1145/3705297.

[11] C. W. J. Granger. Investigating Causal Relations by Econometric Models and Cross-Spectral Methods. Econometrica, 37(3):424–438, July 1969. URL https://ideas.repec.org/a/ecm/emetrp/v37y1969i3p424-38.html.

[12] U. Hasan, E. Hossain, and M. O. Gani. A survey on causal discovery methods for i.i.d. and time series data. Transactions on Machine Learning Research, 2023. ISSN 2835-8856. URL https://openreview.net/forum?id=YdMrdhGx9y. Survey Certification.

[13] W. He, X. Mao, C. Ma, Y. Huang, J. M. Hernàndez-Lobato, and T. Chen. Bsoda: A bipartite scalable framework for online disease diagnosis. In Proceedings of the ACM Web Conference 2022, WWW '22, page 2511–2521, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450390965. doi: 10.1145/3485447.3512123. URL https://doi.org/10.1145/3485447.3512123.

[14] A. Holtzman, J. Buys, L. Du, M. Forbes, and Y. Choi. The curious case of neural text degeneration. In International Conference on Learning Representations, 2020. URL https://openreview.net/forum?id=rygGQyrFvH.

[15] D. J. Im, K. Zhang, N. Verma, and K. Cho. Using deep autoregressive models as causal inference engines, 2024. URL https://arxiv.org/abs/2409.18581.

[16] D. Janzing, D. Balduzzi, M. Grosse-Wentrup, and B. Schölkopf. Quantifying causal influences. The Annals of Statistics, 03 2012. doi: 10.1214/13-AOS1145.

[17] A. Labach, A. Pokhrel, X. S. Huang, S. Zuberi, S. E. Yi, M. Volkovs, T. Poutanen, and R. G. Krishnan. Duett: Dual event time transformer for electronic health records. In K. Deshpande, M. Fiterau, S. Joshi, Z. Lipton, R. Ranganath, I. Urteaga, and S. Yeung, editors, Proceedings of the 8th Machine Learning for Healthcare Conference, volume 219 of Proceedings of Machine Learning Research, pages 403–422. PMLR, 11–12 Aug 2023. URL https://proceedings.mlr.press/v219/labach23a.html.

[18] J. Li, K. Cheng, S. Wang, F. Morstatter, R. P. Trevino, J. Tang, and H. Liu. Feature selection: A data perspective. CoRR, abs/1601.07996, 2016. URL http://arxiv.org/abs/1601.07996.

[19] M. Liu, C.-W. Lee, X. Sun, X. Yu, Y. QIAO, and Y. Wang. Learning causal alignment for reliable disease diagnosis. In The Thirteenth International Conference on Learning Representations, 2025. URL https://openreview.net/forum?id=ozZG5FXuTV.

[20] Q. Luo, L. Zhang, Z. Xing, H. Xia, and Z.-X. Chen. Causal discovery of flight service process based on event sequence. Journal of Advanced Transportation, 2021(1):2869521, 2021. doi: https://doi.org/10.1155/2021/2869521. URL https://onlinelibrary.wiley.com/doi/abs/10.1155/2021/2869521.

[21] L. Ma, L. Hu, Y. Li, W. Ding, and W. Gao. Mi-mcf: A mutual information-based multilabel causal feature selection. IEEE Transactions on Neural Networks and Learning Systems, pages 1–15, 2025. doi: 10.1109/TNNLS.2025.3556128.

[22] L. D. Manocchio, S. Layeghy, W. W. Lo, G. K. Kulatilleke, M. Sarhan, and M. Portmann. Flowtransformer: A transformer framework for flow-based network intrusion detection systems. Expert Systems with Applications, 241:122564, 2024. ISSN 0957-4174. doi: https://doi.org/10.1016/j.eswa.2023.122564. URL https://www.sciencedirect.com/science/article/pii/S095741742303066X.

[23] H. Math, R. Lienhart, and R. Schön. Harnessing event sensory data for error pattern prediction in vehicles: A language model approach. Proceedings of the AAAI Conference on Artificial Intelligence, 39(18):19423–19431, Apr. 2025. doi: 10.1609/aaai.v39i18.34138. URL https://ojs.aaai.org/index.php/AAAI/article/view/34138.

[24] M. Nauta, D. Bucur, and C. Seifert. Causal discovery with attention-based convolutional neural networks. Machine Learning and Knowledge Extraction, 1(1):312–340, 2019. ISSN 2504-4990. doi: 10.3390/make1010019. URL https://www.mdpi.com/2504-4990/1/1/19.

[25] E. Nichani, A. Damian, and J. D. Lee. How transformers learn causal structure with gradient descent. In Proceedings of the 41st International Conference on Machine Learning, ICML'24. JMLR.org, 2024.

[26] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. PyTorch: an imperative style, high-performance deep learning library. Curran Associates Inc., Red Hook, NY, USA, 2019.

[27] J. Pearl. Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1988. ISBN 1558604790.

[28] J. Pearl. Causality: Models, Reasoning and Inference. Cambridge University Press, USA, 2nd edition, 2009. ISBN 052189560X.

[29] P. Pirasteh, S. Nowaczyk, S. Pashami, M. Löwenadler, K. Thunberg, H. Ydreskog, and P. Berck. Interactive feature extraction for diagnostic trouble codes in predictive maintenance: A case study from automotive domain. In Proceedings of the Workshop on Interactive Data Mining, WIDM'19, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450362962. doi: 10.1145/3304079.3310288. URL https://doi.org/10.1145/3304079.3310288.

[30] J. Qiao, R. Cai, S. Wu, Y. Xiang, K. Zhang, and Z. Hao. Structural hawkes processes for learning causal structure from discrete-time event sequences. In Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI '23, 2023. ISBN 978-1-956792-03-4. doi: 10.24963/ijcai.2023/633. URL https://doi.org/10.24963/ijcai.2023/633.

[31] J. R. Quinlan. Induction of decision trees. Machine Learning, 1:81–106, 1986.

[32] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever. Improving language understanding by generative pre-training. 2018.

[33] L. Rasmy, Y. Xiang, Z. Xie, C. Tao, and D. Zhi. Med-bert: pre-trained contextualized embeddings on large-scale structured electronic health records for disease prediction. NPJ Digit Med. 2021 May 20;4(1):86, abs/2005.12833, 2020. doi: 10.1038/s41746-021-00455-y.

[34] R. Y. Rohekar, Y. Gurwicz, and S. Nisimov. Causal interpretation of self-attention in pre-trained transformers. In Thirty-seventh Conference on Neural Information Processing Systems, 2023. URL https://openreview.net/forum?id=DS4rKySlYC.

[35] P. Spirtes and C. Glymour. An algorithm for fast recovery of sparse causal graphs. Social Science Computer Review, 9(1):62–72, 1991. doi: 10.1177/089443939100900106. URL https://doi.org/10.1177/089443939100900106.

[36] P. Spirtes, C. Glymour, and R. Scheines. Causation, prediction, and search, 2nd edition. In Causation, Prediction, and Search (Second Edition), 2001. URL https://api.semanticscholar.org/CorpusID:124969922.

[37] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, A. Rodriguez, A. Joulin, E. Grave, and G. Lample. Llama: Open and efficient foundation language models, 2023. URL https://arxiv.org/abs/2302.13971.

[38] I. Tsamardinos and C. F. Aliferis. Towards principled feature selection: Relevancy, filters and wrappers. In C. M. Bishop and B. J. Frey, editors, Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics, volume R4 of Proceedings of Machine Learning Research, pages 300–307. PMLR, 03–06 Jan 2003. URL https://proceedings.mlr.press/r4/tsamardinos03a.html. Reissued by PMLR on 01 April 2021.

[39] I. Tsamardinos, C. Aliferis, and A. Statnikov. Algorithms for large scale markov blanket discovery. pages 376–381, 01 2003.

[40] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, Advances in Neural Information Processing Systems, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[41] C. Wang, Y. Zhou, Q. Zhao, and Z. Geng. Discovering and orienting the edges connected to a target variable in a dag via a sequential local learning approach. Computational Statistics and Data Analysis, 77:252–266, 2014. ISSN 0167-9473. doi: https://doi.org/10.1016/j.csda.2014.03.003. URL https://www.sciencedirect.com/science/article/pii/S0167947314000802.

[42] X. Wu, B. Jiang, Y. Zhong, and H. Chen. Multi-label causal variable discovery: Learning common causal variables and label-specific causal variables. CoRR, abs/2011.04176, 2020. URL https://arxiv.org/abs/2011.04176.

[43] F. Xie, Z. Li, P. Wu, Y. Zeng, C. Liu, and Z. Geng. Local causal structure learning in the presence of latent variables. In Proceedings of the 41st International Conference on Machine Learning, ICML'24. JMLR.org, 2024.

[44] J. Yin, Y. Zhou, C. Wang, P. He, C. Zheng, and Z. Geng. Partial orientation and local structural learning of causal networks for prediction. In I. Guyon, C. Aliferis, G. Cooper, A. Elisseeff, J.-P. Pellet, P. Spirtes, and A. Statnikov, editors, Proceedings of the Workshop on the Causation and Prediction Challenge at WCCI 2008, volume 3 of Proceedings of Machine Learning Research, pages 93–105, Hong Kong, 03–04 Jun 2008. PMLR. URL http://proceedings.mlr.press/v3/yin08a.html.

[45] K. Yu, X. Guo, L. Liu, J. Li, H. Wang, Z. Ling, and X. Wu. Causality-based feature selection: Methods and evaluations. ACM Comput. Surv., 53(5), Sept. 2020. ISSN 0360-0300. doi: 10.1145/3409382. URL https://doi.org/10.1145/3409382.

[46] M.-L. Zhang and Z.-H. Zhou. A review on multi-label learning algorithms. Knowledge and Data Engineering, IEEE Transactions on, 26:1819–1837, 08 2014. doi: 10.1109/TKDE.2013.39.

A Notations

We use capital letters (e.g., X) to denote random variables, lower-case letters (e.g., x) for their realisations, and bold capital letters (e.g., $\boldsymbol{X}$) for sets of variables. Let $\boldsymbol{U}$ denote the set of all (discrete) random variables. We define the event set $\boldsymbol{X} = \{X_1, \ldots, X_n\} \subset \boldsymbol{U}$, and the label set $\boldsymbol{Y} = \{Y_1, \ldots, Y_n\} \subset \boldsymbol{U}$. When explicitly said, event $X_i^{(t_i)}$ represent the occurrence of X_i at step i and time t_i. Similarly for $Y_{i+1}^{(t_{i+1})}$.

B Definitions

Definition 1 (Bayesian Network)**.** *Pearl [27] Let P denote the joint distribution over a variable set $\boldsymbol{U}$ of a directed acyclic graph (DAG) $\mathbb{G}$. The triplet $< \boldsymbol{U}, \mathbb{G}, P >$ constitutes a BN if the triplet $< \boldsymbol{U}, \mathbb{G}, P >$ satisfies the Markov condition: every random variable is independent of its non-descendant variables given its parents in $\mathbb{G}$. Each node $X_i \in \boldsymbol{U}$ represents a random variable. The directed edge $(X_i \to X_j)$ encodes a probabilistic dependence. The joint probability distribution can be factorized $P(X_1, \cdots, X_n) = \prod_{i=1}^{n} P(X_i|X_1, \cdots, X_{i-1})$. If a variable does not depend on all of its predecessors, we can write: $P(X_i|X_1, \cdots, X_{i-1}) = P(X_i|par(X_i))$ with 'par' the parents of node X_i such that: $par(X_i) = \{X_1, \cdots, X_{i-1}\}$.*

Definition 2 (Faithfulness)**.** *Spirtes et al. [36]. Given a BN $< \boldsymbol{U}, \mathbb{G}, P >$, $\mathbb{G}$ is faithful to P if and only if every conditional independence present in P is entailed by $\mathbb{G}$ and the Markov condition holds. P is faithful if and only if there exist a DAG $\mathbb{G}$ such that $\mathbb{G}$ is faithful to P.*

Definition 3 (Markov Boundary)**.** *Tsamardinos and Aliferis [38]. In a faithful BN $< \boldsymbol{U}, \mathbb{G}, P >$, for a set of variables $\boldsymbol{Z} \subset \boldsymbol{U}$ and label $Y \in \boldsymbol{U}$, if all other variables $X \in \{\boldsymbol{X} - \boldsymbol{Z}\}$ are independent of Y conditioned on $\boldsymbol{Z}$, and any proper subset of $\boldsymbol{Z}$ do not satisfy the condition, then $\boldsymbol{Z}$ is the Markov Boundary of Y: $\boldsymbol{MB}(Y)$.*

Definition 4 (Conditional Independence)**.** *Variables X and Y are said to be conditionally independent given a variable set $\boldsymbol{Z}$, if $P(X, Y|\boldsymbol{Z}) = P(X|\boldsymbol{Z})P(Y|\boldsymbol{Z})$, denoted as $X \perp Y|\boldsymbol{Z}$. Inversely, $X \not\perp Y|\boldsymbol{Z}$ denotes the conditional dependence. Using the conditional mutual information [3] to measure the independence relationship, this implies that $I(X, Y|\boldsymbol{Z}) = 0 \Leftrightarrow X \perp Y|\boldsymbol{Z}$.*

C Assumptions

Assumption 1 (Temporal Precedence)**.** *Given a perfectly recorded sequence of events $((x_1, t_1), \cdots, (x_L, t_L))$ with labels $(\boldsymbol{y}_L, t_L)$ and monotonically increasing time of occurrence $0 \leq t_1 \leq \cdots \leq t_L$, an event x_i is allowed to influence any subsequent event x_j such that $t_i \leq t_j$ and $i < j$. Formally, the graph $\mathbb{G} = (\boldsymbol{U}, \boldsymbol{E})$, $(x_i, x_j) \in \boldsymbol{E} \implies t_i \leq t_j$ and step $i < j$*

It allows us to remove ambiguity in causal directionality. By allowing for instantaneous rates $(t_i = t_{i+1})$, our method differs from Granger [11] causal discovery.

Assumption 2 (Bounded Lagged Effects)**.** *Once we observed events up to timestamp t_i and step i as $\boldsymbol{Z}_{\leq t_i} = ((x_1, t_1), \cdots, (x_i, t_i))$, any future lagged copy of event $X_i^{(t_i+\tau)}$ is independent of Y_j conditioned on $\boldsymbol{Z}_{\leq t_i}$:*

$$Y_j \perp X_i^{(t_i+\tau)} | \boldsymbol{Z}_{\leq t_i}$$

Where $\tau = t_{i+1} - t_i$ is a finite bound on the allowed time delay for causal influence.

In other words, we allow the causal influence of event X_i on Y_j until the next event X_{i+1} is observed. We note that for data with strong lagged effects (e.g., financial transactions), this might not hold well, but it is relevant for log-based and error code-based data.

Assumption 3 (Causal Sufficiency for Labels)**.** *All relevant variables are observed, and there are no hidden confounders affecting the labels.*

Assumption 4 (Oracle Models)**.** *We assume that two autoregressive Transformer models, Tf_x and Tf_y, are trained via maximum likelihood on a dataset of multi-labeled event sequences $D_n = \{S_l^1, \cdots, S_l^n\} \subset \mathbb{S}$, and can perfectly approximate the true conditional distributions of events and labels:*

$$P(X_i|Pa(X_i)) = P_{\theta_x}(X_i|Pa(X_i)) = Tf_x(S_{<i}),\ P(Y_j|Pa(Y_j)) = P_{\theta_y}(Y_j|Pa(Y_j)) = Tf_y(S_{\leq i}) \tag{11}$$

D Lemmas

Lemma 1 (Identifiability of $\mathbb{G}$)**.** *Assuming the faithfulness condition holds for the true causal graph $\mathbb{G}$. Let Tf_x and Tf_y be oracle models that model the true conditional distributions of events and labels, respectively. The joint distribution P_{θ_x,θ_y} can then be constructed, and any conditional independence detected from the distributions estimated by Tf_x and Tf_y corresponds to a conditional independence in $\mathbb{G}$:*

$$X_i \perp_{\theta_x,\theta_y} Y_j \mid \boldsymbol{Z} \quad \Longrightarrow \quad X_i \perp_{\mathbb{G}} Y_j \mid \boldsymbol{Z}.$$

Where $\perp_{\theta_x,\theta_y}$ denotes the independence entailed by the joint probability P_{θ_x,θ_y}.

Lemma 2 (Markov Boundary Equivalence)**.** *In a multi-label event sequence S_l and under the temporal precedence assumption A1, the Markov Boundary of each label Y_j is only its parents such that $\forall X \in \{\boldsymbol{U} - Pa(Y_j)\}, X \perp Y_j|Pa(Y_j) \Leftrightarrow MB(Y_j) = Pa(Y_j)$.*

E Proofs

We provide proofs for the results described in Section 3

E.1 Proof of Lemma 1

Proof. We assume that the data is generated by the associated causal graph $\mathbb{G}$ following the sequential BN from a multi-labelled sequence S. And that the faithfulness assumption holds [27], meaning that all conditional independencies in the observational data are implied by the true causal graph $\mathbb{G}$. Given that the Oracle models Tf_x and Tf_y are trained to perfectly approximate the true conditional distributions, for any variable U_i in the graph, we have:

$$P(U_i|\text{Pa}(U_i)) = \begin{cases} P(Y_j|\text{Pa}(Y_j)) = P_{\theta_y}(Y_j|\text{Pa}(Y_j)), & \text{if } U_i \in \boldsymbol{Y} \\ P(X_i|\text{Pa}(X_i)) = P_{\theta_x}(X_i|\text{Pa}(X_i)), & \text{otherwise.} \end{cases}$$

The joint distribution P_{θ_x,θ_y} can then be constructed using the chain rule $P_{\theta_x,\theta_y}(X_1, \cdots, X_i, Y_1, \cdots, Y_c) = \prod_{k=0}^{i} P(X_k|Pa(X_k)) \prod_l^c P(Y_l|\text{Pa}(Y_l)$. By the faithfulness assumption [27], if the conditional independencies hold in the data, they must also hold in the causal graph $\mathbb{G}$:

$$X_i \perp Y_j|\boldsymbol{Z} \implies X_i \perp_{\mathbb{G}} Y_j|\boldsymbol{Z}$$

Since we can approximate the true conditional distributions, it follows that:

$$X_i \perp_{\theta_x,\theta_y} Y_j|\boldsymbol{Z} \implies X_i \perp Y_j|\boldsymbol{Z} \implies X_i \perp_{\mathbb{G}} Y_j|\boldsymbol{Z}$$

Where $\perp_{\theta_x,\theta_y}$ denotes the independence entailed by the joint probability P_{θ_x,θ_y}. Thus, the graph $\mathbb{G}$ can be identified from the observational data. □

E.2 Proof of Lemma 2

Proof. Let $< \boldsymbol{U}, \mathbb{G}, P >$ be the sequential BN composed of the events from the multi-labeled sequence $S_l = (\{(t_1, x_1, \cdots, (t_L, x_L)\}_{i=1}^{L}, (\boldsymbol{y}_L, t_L))$. Following the temporal precedence assumption A1, the labels $\boldsymbol{y_L}$ can only be caused by past events $(x_1, \cdots, x_L)$; moreover, by definition, labels do not cause any other labels. Thus, Y_j has no descendants, so no children and spouses. Therefore, together with the Markov Assumption we know that $\forall X \in \{\boldsymbol{U} - Pa(Y_j)\} : Y_j \perp X|Pa(Y_j)$. Which is the definition of the MB (Def. 3). Thus, $\mathbf{MB}(Y_j) = Pa(Y_j)$.

□

E.3 Proof of Theorem 1.

Proof. By recurrence over the sequence length L of the multi-label sequence S_l^k, we want to show that under temporal precedence A1, bounded lagged effects A2, causal sufficiency A3, Oracle Models A4 the Markov Boundary of label Y_j can be identified in the causal graph $\mathbb{G}$.

Let's define $\mathcal{M}_j^L$ as the estimated Markov Boundary of Y_j after observing L events.

Base Case: $\mathbf{L = 1}$: Consider the BN for step $L = 1$ following the Markov assumption [27] with two nodes X_1, Y_j. Using Tf_x, Tf_y as Oracle Models A4, we can express the conditional probabilities for any node U:

$$P(U|\text{Pa}(U)) = \begin{cases} P(X_1) = P_{\theta_x}(X_1|[CLS]) \text{ if } U \in \boldsymbol{X} \\ P(Y_j|X_1) = P_{\theta_y}(Y_j|X_1) \text{ otherwise} \end{cases} \tag{12}$$

Assuming that P is faithful (A2) to $\mathbb{G}$, no hidden confounders bias the estimate (A3) and temporal precedence (A1), we can estimate the CMI 3 such that iif $I(X_1, Y_j)|\emptyset) > 0 \Leftrightarrow Y_j \not\perp_{\theta_x,\theta_y} X_1 \implies Y_j \not\perp_{\mathbb{G}} X_1$ (Lemma 1).

Since we assume temporal precedence A1, we can orient the edge such that X_1 must be a parent of Y_j in $\mathbb{G}$. Using Lemma 2, we know that $Par(Y_j) = \textbf{MB}(Y_j) \implies X_1 \in \textbf{MB}(Y_j)$, thus we must include X_1 in M_j^1, otherwise not.

Heredity: For $L = i$, we obtained M_j^i with the sequential BN up to step $L = i$. Now for $L = i + 1$, the sequential BN has $i + 2$ nodes denoted as $\boldsymbol{U'} = (X_1, \cdots, X_i, X_{i+1}, Y_j)$. Using the Oracle Models A4 and following the Markov assumption [27], we can estimates the following conditional probabilities for any nodes $U \in \boldsymbol{U'}$:

$$P(U|\text{Pa}(U)) = \begin{cases} P(Y_j|\text{Pa}(Y_j)) \approx P_{\theta_y}(Y_j|\text{Pa}(Y_j)), & \text{if } U \in \boldsymbol{Y} \\ P(X|\text{Pa}(X)) \approx P_{\theta_x}(X|\text{Pa}(X)), & \text{otherwise.} \end{cases} \tag{13}$$

By bounded lagged effects (A2) we know that the causal influence of past $X_{\leq i}$ on Y_j has expired. Moreover, no hidden confounders (A3) bias the independence testing. Finally, using Eq. (3), we can estimate the CMI such that iif $I(Y_j, X_{i+1}|\boldsymbol{Z}) > 0 \Leftrightarrow Y_j \not\perp_{\theta_x,\theta_y} X_{i+1}|\boldsymbol{Z} \implies Y_j \not\perp_{\mathbb{G}} X_{i+1}|\boldsymbol{Z}$ (Lemma 1).

Since we assume temporal precedence A1, we can orient the edge so that X_{i+1} must be a parent of Y_j in $\mathbb{G}$. Using Lemma 2, we know that $Par(Y_j) = \textbf{MB}(Y_j) \implies X_{i+1} \in \textbf{MB}(Y_j)$. Thus $X_{i+1} \in M_j^{i+1}$ which represent the $\textbf{MB}(Y_j)$ for step $i + 1$.

Finally, $\mathcal{M}_j^{i+1}$ still recovers the Markov Boundary of Y_j such that

$$\forall U \in \{\boldsymbol{U'} - \mathcal{M}_j^{i+1}\}, Y_j \perp U|\mathcal{M}_j^{i+1}$$

□

F Evaluation

F.1 Metrics

The Precision, Recall, and F1-Score for Markov boundary estimation were computed as follows using the True set as the error pattern rule (True Markov Boundary) and the Inferred Markov Boundary set from OSCAR:

- **Precision** (P) measures the proportion of correctly identified causal events among all inferred events:

  $$P = \frac{|\text{Inferred} \cap \text{True}|}{|\text{Inferred}|}$$

  where $|\text{Inferred} \cap \text{True}|$ is the number of true positive causal events, and $|\text{Inferred}|$ is the total number of inferred causal events.

- **Recall** (R) captures the proportion of correctly identified causal events among all true causal events:

$$R = \frac{|\text{Inferred} \cap \text{True}|}{|\text{True}|}$$

where $|\text{True}|$ is the total number of true causal tokens.

- **F1-Score** (F_1) is the harmonic mean of precision and recall, providing a balanced measure:

$$F_1 = \frac{2 \cdot P \cdot R}{P + R}$$

F.2 PyCausalFS

Local structure learning algorithms were all used with $\alpha = 0.1$ in the associated code: `https://github.com/wt-hu/pyCausalFS/tree/master/pyCausalFS/LSL`.

F.3 MI-MCF

MI-MCF [21] was used for comparison following the official implementation at `https://github.com/malinjlu/MI-MCF` we used $\alpha = 0.05, L = 268, k_1 = 0.7, k_2 = 0.1$.

G Ablations

G.1 NADEs Quality.

We did several ablations on the quality of the NADEs and their impact on the one-shot causal discovery phase. In particular, Table 2 presents multiple Tf_x, Tf_y with respectively 90 and 15 million parameters or 34 and 4 million parameters. We also varied the context window (conditioning set $\boldsymbol{Z}$), trained on different amounts of data (Tokens), and reported the classification results on the test set of Tf_y alone. We didn't output the Running time since it was always the same for all NADEs: 1.27 minutes of 50,000 samples and 0.14 for 5000.

Table 2: Ablations of the performance of Phase 1 (One-shot **MB** retrieval) in function of different NADEs with $n = 50{,}000$ and $n = 500$ samples averaged over 5-folds. Classification metrics use weighted averaging. Metrics are given in $\%$.

Tokens	Parameters	Context	Precision (↑)	Recall (↑)	F1 Score (↑)	Tfy F1 (↑)
			For $n = 50{,}000$ samples			
1.5B	105m	$c = 4$	47.95 ± 1.05	30.65 ± 0.51	37.39 ± 0.67	88.6
1.5B	105m	$c = 12$	54.62 ± 1.03	29.88 ± 0.73	38.63 ± 0.85	90.43
1.5B	105m	$c = 15$	$\mathbf{55.26 \pm 1.42}$	$\mathbf{31.37 \pm 0.82}$	$\mathbf{40.02 \pm 1.03}$	90.57
1.5B	105m	$c = 20$	49.52 ± 1.59	$\mathbf{31.76 \pm 0.85}$	36.54 ± 1.10	91.19
1.5B	105m	$c = 30$	36.65 ± 1.18	22.75 ± 0.78	26.57 ± 0.91	**92.64**
300m	47m	$c = 20$	39.49 ± 1.77	26.30 ± 0.89	29.01 ± 1.10	83.6
			For $n = 500$ samples			
1.5B	105m	$c = 12$	54.84 ± 4.55	$\mathbf{31.45 \pm 2.23}$	$\mathbf{39.95 \pm 2.83}$	90.43
1.5B	105m	$c = 15$	55.04 ± 3.36	29.90 ± 1.78	38.74 ± 2.24	90.57
1.5B	105m	$c = 20$	48.84 ± 4.01	$\mathbf{31.65 \pm 2.37}$	36.19 ± 2.65	**91.19**
300m	47m	$c = 20$	38.23 ± 2.91	25.31 ± 2.39	27.92 ± 2.25	83.6

G.2 Sampling Type

We performed an ablation (Tab 3) on the effect of sampling methods to estimate the expected value over all possible context $\boldsymbol{Z}$. We used one A10 GPU on a sample of the test dataset (4000 random samples) composed of 205 labels with a batch size of 4 during inference. We tested top-k sampling with $k = \{20, 35\}$ [6] with and w/o a temperature scaler of T to log-probabilities $\hat{\boldsymbol{x}}$ such that

$$\hat{\boldsymbol{x}}' = \text{softmax}(\log \hat{\boldsymbol{x}}/T)$$

And a combination of top-k and a top-nucleus sampling [14] with different probability mass $p = \{0.8, 1.2\}$ and finally a permutation of token position within the context c. We fixed a dynamic threshold with z score $k = 3$ and performed 10 runs. Then, we reported the average and standard deviation of each classification metric and elapsed time (sec).

Without a surprise, sampling increases the predictive performance of OSCAR by a large margin. More interestingly, different sampling types have different effects on specific averaging. This has a 'smoothing' effect on the CMI curve when multiple labels are present in the sequence. When having no upsampling, the sensitivity of the CMI of different labels is increased, which makes it more difficult to capture a threshold and a potential cause. We can notice that globally, top-k sampling provides better results, especially with a combination of top-p=0.8 afterwards.

Sampling with the same tokens (*Permutation*) is not a good choice; sampling from the next-event prediction Tf_x yielded better results. We will choose **Top-k+p=0.8** for the increased F1 Micro and high F1 Macro, and Weighted.

Table 3: One-shot Classification performance and Elapsed Time (sec) across different sampling methods. Best results are shown in **Bold** and Best ex aequo in underline.

Sampling Method	F1 Micro (%)	F1 Macro (%)	F1 Weighted (%)	Time (sec)
w/o Sampling	14.07	12.29	16.67	$\mathbf{49.30 \pm 0.30}$
Permutation	18.22 ± 0.36	13.75 ± 0.09	19.21 ± 0.03	557.82 ± 0.13
Top-k=20	26.77 ± 0.71	23.83 ± 0.19	29.25 ± 0.07	557.4 ± 0.13
Top-k=35	26.57 ± 0.96	24.08 ± 0.23	29.30 ± 0.07	557.35 ± 0.10
Top-k=35+T=0.8	27.36 ± 0.65	23.77 ± 0.21	28.98 ± 0.07	557.45 ± 0.11
Top-k=35+T=1.2	26.59 ± 1.49	$\mathbf{24.62 \pm 0.29}$	$\mathbf{29.52 \pm 0.06}$	557.45 ± 0.12
Top-k=25+p=0.8	27.98 ± 0.67	23.82 ± 0.28	29.18 ± 0.07	558.07 ± 0.07
Top-k=35+p=0.8	$\mathbf{28.82 \pm 0.75}$	24.06 ± 0.25	29.17 ± 0.07	558.16 ± 0.14
Top-k=35+p=0.9	26.39 ± 0.99	24.12 ± 0.31	29.26 ± 0.11	558.11 ± 0.12
Top-k=35+p=0.9+T=0.9	27.63 ± 0.75	23.90 ± 0.24	29.04 ± 0.09	558.07 ± 0.12
Top-k=35+p=0.9+T=1.1	26.75 ± 1.30	$\mathbf{24.47 \pm 0.24}$	29.45 ± 0.09	558.06 ± 0.11

G.3 Sampling Number

We experimented with different numbers of n for the sampling method across different averaging (micro, macro, weighted), Fig. 3. We performed 8 different runs and reported the average, standard deviation, and elapsed time. We can say that generally, sampling with a bigger N tends to decrease the standard deviation and give more reliable Markov Boundary estimation. Moreover, as we process more samples, the model is gradually improving at a logarithmic growth until it converges to a final score. We also verify that our time complexity is linear with the number of samples N. Based on these results, we choose generally $N = 68$ as the number of samples.

G.4 Dynamic Thresholding

We performed ablations on the effect of k during the dynamic thresholding of the CMI (Eq. (6)) to access conditional independence in Fig. 4. To balance the classification metrics across the different averaging, we set $k = 2.75$.

Figure 3: Evolution of several classification metrics (one-shot) and elapsed time per sample in function of the number of samples N chosen. Results are reported using a 1-sigma error bar.

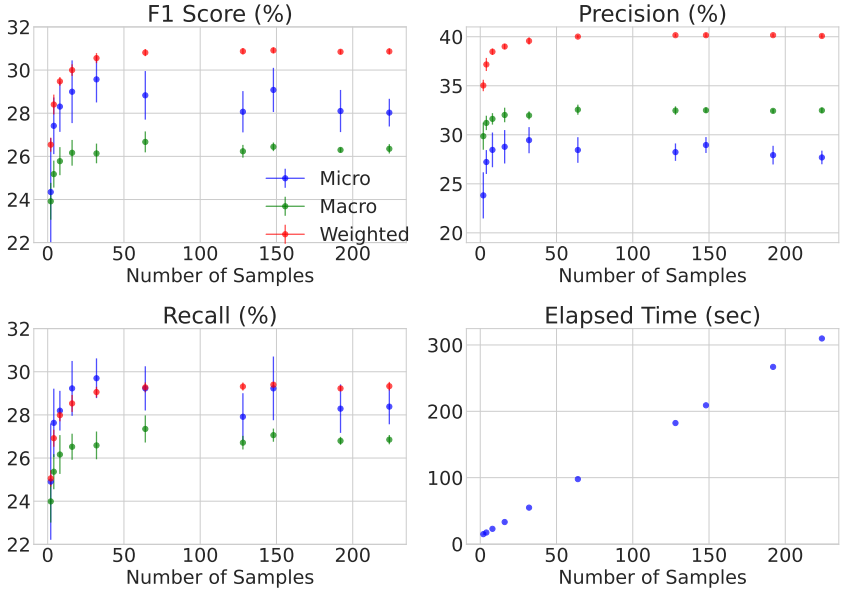
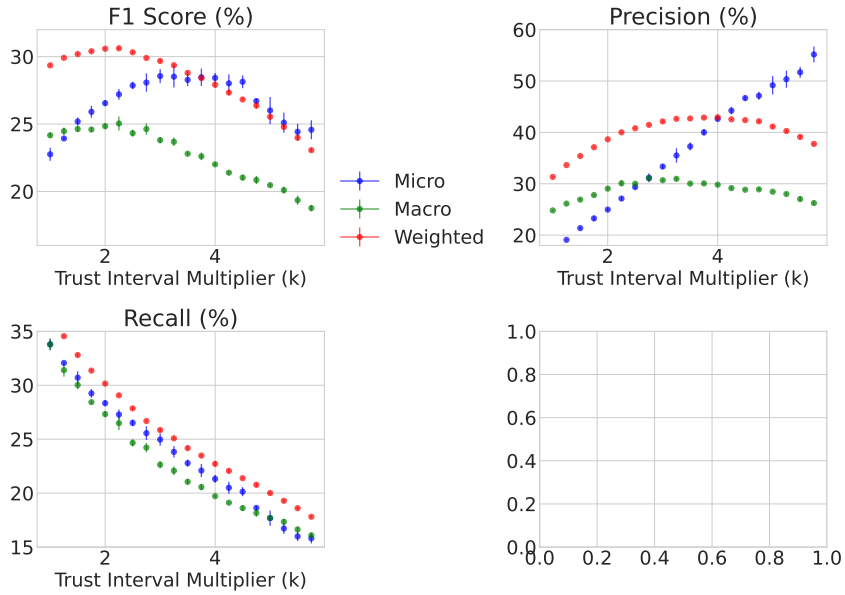


Figure 4: Evolution of one-shot F1 Score, Precision and Recall in function of coefficient k. Results are reported using 1-sigma error bar.



H Discussion on Assumptions

Our approach relies on several assumptions that enable one-shot causal discovery under practical and computational constraints.

Temporal Precedence Temporal precedence (A1) simplifies directionality and faithfulness to $\mathbb{G}$. It allows for instantaneous influence, which aligns better with log-based data in cybersecurity or vehicle diagnostics, where events can co-occur at the same timestamp. However, this places strong reliance on precise event time-stamping. Even though we only test $X_i \to Y_j$, this could falsify the conditioning test $\boldsymbol{Z}$.

Bounded Lagged Effects. The bounded lagged effects (A2) assumption enables us to restrict causal influence and recover the **MB** of each label using Theorem 1. It also makes the computation faster. In most real-world sequences where relevant history is limited, this holds empirically. Nonetheless, in highly delayed causal chains, like financial transactions, some influence may be missed.

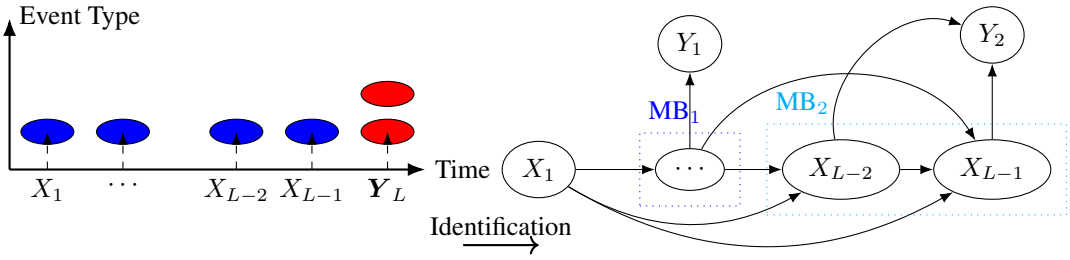
Causal Sufficiency. As with many causal discovery approaches, we assume all relevant variables are observed (A3). Although it sounds like a strong assumption, interestingly, in high-cardinality domains such as vehicle diagnostics, the volume of recorded events may reduce but not eliminate the risk of hidden confounding.

Inter-label Effects. By definition, the labels are explained solely by events. While simplifying multi-label causal discovery, this intrinsic assumption could be relaxed in future work by using the *do* operator [28] to perform interventions on common causal variables of multiple labels. For example, our current framework estimates the Markov Boundaries for each label independently. However, inter-label dependencies can exist, particularly when labels share overlapping Markov Boundaries (e.g $MB_1 = [X_1, X_3], MB_2 = [X1, X_2]$. We propose to investigate a 'Phase 2' for OSCAR, focusing on inter-label dependencies through simulated interventions. For instance, if we consider a sequence S_1 of two labels Y_1, Y_2 with the MB above, we could perform counterfactual interventions by applying $do(X_1 = 0), do(X_3 = 0)$to S_1. Then we would observe the average change in the likelihood of Y_1 which, if it is non-zero, would indicate a dependence between Y_1 and Y_2. Wu et al. [42] points out that the assumptions of these inter-label dependencies are already anchored in the Markov Boundaries; we do the same here.

NADEs. Due to the usage of flexible NADEs, we can relax common assumptions regarding data generation processes, such as Poisson Processes or SCMs. Finally, as seen in the Ablations G.1, the effectiveness of OSCAR hinges on the capacity of Tf_x and Tf_y to approximate true conditional probabilities (A4) and provide Oracle CI-test. While assuming Oracle tests are common in the literature [43, 18] and necessary to recover correct causal structures, this remains a strong assumption. And it is only valid to the extent that the models are perfectly trained. Especially for multi-label classification, performance may degrade in underrepresented regions of the data distribution, as we saw during the **MB** length comparison.

I Figures

Figure 5: An example of a causal graph extracted from a multi-label event sequence where MB_1 represents the Markov Boundary of Y_1 and MB_2 the Markov Boundary of Y_2.



J Explanation example

To enhance interpretability and illustrate the learned relationships, we present graphical explanations of error pattern occurrences based on sequences of Diagnostic Trouble Codes (DTCs). For each case, we selected representative samples that reflect diverse yet intuitive failure scenarios.

Fig. 7 depicts a clear-cut example involving a single failure label related to the emergency antenna system. In contrast, Fig. 8 captures a more intricate case where airbag and tire pressure (RDC) malfunctions co-occur. These graphs highlight the influence of preceding events, with causal contributions shown in orange and red, and inhibitory effects illustrated in pink. Such visualisations serve to provide both human-understandable insights and support for the model's reasoning process.

Figure 6: Example of a sequence of events (DTCs) that lead to a steering wheel degradation and a power limitation as outcome labels. The inhibitory strengths are shown in violet and causal strengths in orange and red depending on the magnitude.

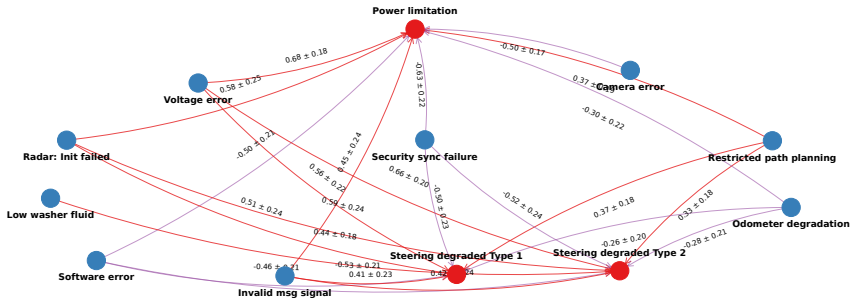


Figure 7: Example of a sequence of events (DTCs) that lead to an emergency antenna dysfunction as outcome labels. The inhibitory strengths are shown in pink and causal strengths in orange and red

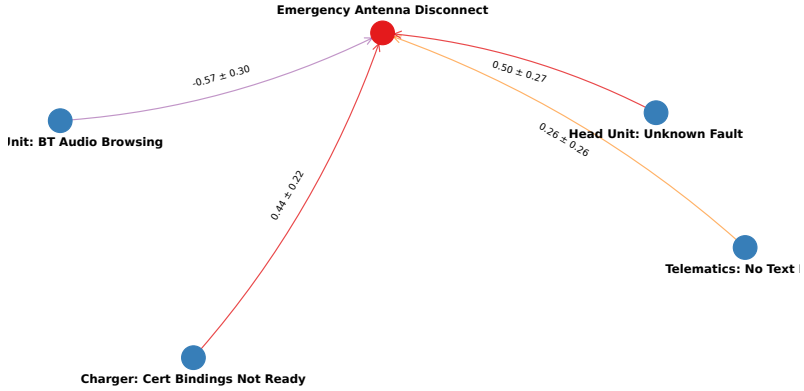
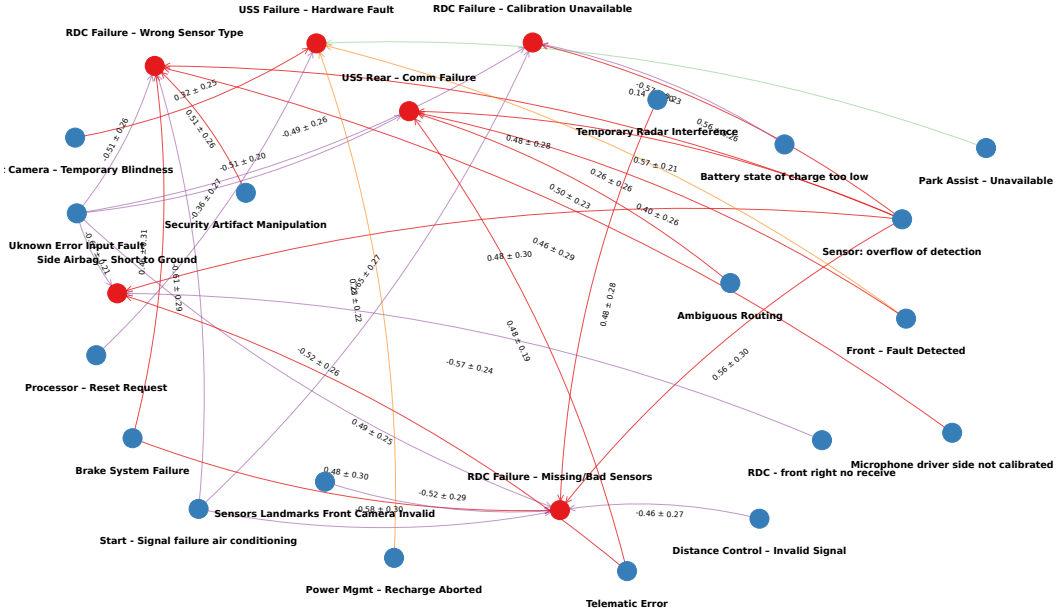


Figure 8: Example of a sequence of events (DTCs) that lead to an airbag and tire pressure malfunctions as outcome labels. The inhibitory strengths are shown in pink and causal strengths in orange and red



K Implementation

The following is the implementation of OSCAR in PyTorch [26].

```python
def topk_p_sampling(z, prob_x, c: int, n: int = 64, p: float = 0.8, k: int = 35,
                    cls_token_id: int = 1, temp: float = None):
    # Sample just the context
    input_ = prob_x[:, :c]

    # Top-k first
    topk_values, topk_indices = torch.topk(input_, k=k, dim=-1)

    # Top-p over top-k values
    sorted_probs, sorted_idx = torch.sort(topk_values, descending=True, dim=-1)
    cum_probs = torch.cumsum(sorted_probs, dim=-1)
    mask = cum_probs > p

    # Ensure at least one token is kept
    mask[..., 0] = 0

    # Mask and normalize
    filtered_probs = sorted_probs.masked_fill(mask, 0.0)
    filtered_probs += 1e-8  # for numerical stability
    filtered_probs /= filtered_probs.sum(dim=-1, keepdim=True)

    # Unscramble to match the original top-k indices
    # Need to reorder the sorted indices back to the original top-k
    reorder_idx = torch.argsort(sorted_idx, dim=-1)
    filtered_probs = torch.gather(filtered_probs, -1, reorder_idx)

    batched_probs = filtered_probs.unsqueeze(1).repeat(1, n, 1, 1)  # (bs, n, seq_len, k)
    batched_indices = topk_indices.unsqueeze(1).repeat(1, n, 1, 1)  # (bs, n, seq_len, k)

    sampled_idx = torch.multinomial(batched_probs.view(-1, k), 1)  # (bs*n*seq_len, 1)
    sampled_idx = sampled_idx.view(-1, n, c).unsqueeze(-1)

    sampled_tokens = torch.gather(batched_indices, -1, sampled_idx).squeeze(-1)
    sampled_tokens[..., 0] = cls_token_id

    # Reconstruct full sequence
    z_expanded = z.unsqueeze(1).repeat(1, n, 1)[..., c:]
    return torch.cat((sampled_tokens, z_expanded), dim=-1)

from torch import nn
def OSCAR(tfe: nn.Module, tfy: nn.Module, batch: dict[str, torch.Tensor], c: int, n: int, eps: float=1e-6, topk: int=20, k: int=2.75, p=0.8) -> torch.Tensor:
    """ tfe, tfy: are the two autoregressive transformers (event type and label)
        batch: dictionary containing a batch of input_ids and attention_mask of shape (bs, L) to explain.
        c: scalar number defining the minimum context to start inferring, also the sampling interval.
        n: scalar number representing the number of samples for the sampling method.
        eps: float for numerical stability
        topk: The number of top-k most probable tokens to keep for sampling
```

```
        k: Number of standard deviations to add to the mean for dynamic threshold calculation
        p: Probability mass for top-p nucleus
    """
    o = tfe(attention_mask=batch['attention_mask'], input_ids=batch['input_ids'])['prediction_logits'] # Infer the next event type
    x_hat = torch.nn.functional.softmax(o, dim=-1)

    b_sampled = topk_p_sampling(batch['input_ids'], x_hat, c, k=topk, n=n, p=p) # Sampling up to (bs, n, L)
    n_att_mask = batch['attention_mask'].unsqueeze(1).repeat(1, n, 1)

    with torch.inference_mode():
        o = tfy(attention_mask=n_att_mask.reshape(-1, b_sampled.size(-1)), input_ids=b_sampled.reshape(-1, b_sampled.size(-1))) # flatten and infer
        prob_y_sampled = o['ep_prediction'].reshape(b_sampled.size(0), n, batch['input_ids'].size(-1)-c, -1) # reshape to (bs, n, L-c)

        # Ensure probs are within (eps, 1-eps)
        prob_y_sampled = torch.clamp(prob_y_sampled, eps, 1 - eps)

        y_hat_i = prob_y_sampled[..., :-1, :] # P(Yj|z)
        y_hat_iplus1 = prob_y_sampled[..., 1:, :] # P(Yj|z, x_i)

        # Compute the CMI & CS and average across sampling dim
        cmi = torch.mean(y_hat_iplus1*torch.log(y_hat_iplus1/y_hat_i)+(1-y_hat_iplus1)*torch.log((1-y_hat_iplus1)/(1-y_hat_i)), dim=1)
        # (BS, L, Y)
        cs = y_hat_iplus1 - y_hat_i
        cs_mean = torch.mean(cs, dim=1)
        cs_std = torch.std(cs, dim=1)

        # Confidence interval for threshold
        mu = cmi.mean(dim=1)
        std = cmi.std(dim=1)
        dynamic_thresholds = mu + std * k

        # Broadcast to select an individual dynamic threshold
        cmi_mask = cmi >= dynamic_thresholds.unsqueeze(1)

        cause_token_indices = cmi_mask.nonzero(as_tuple=False)
        # (num_causes, 3) --> each row is [batch_idx, position_idx, label_idx]
        return cause_token_indices, cs_mean, cs_std, cmi_mask
```

Remark. *Since tfy contains tfe as backbone, in practice we need only one forward pass from tfy and extract also $\hat{x}$, so tfe is not needed. We let it to improve understanding and clarity.*