Copy, paste, infer: A robust analysis of twin networks for counterfactual inference

Logan Graham
Babylon Health,
& University of Oxford
logan@robots.ox.ac.uk

Ciarán M. Lee
Babylon Health,
& University College London
ciaran.lee@babylonhealth.com

Yura Perov
Babylon Health
yura.perov@babylonhealth.com

Abstract

Twin networks are a simple method for estimating counterfactuals, originally proposed to have several advantages over standard counterfactual inference. However, no study yet exists exploring in what contexts twin networks would be more advantageous than standard counterfactual methods in practice. We conduct an empirical and theoretical analysis of twin networks to show that in certain cases of Structural Causal Models, twin networks are faster and less memory intensive by orders of magnitude than standard counterfactual inference, in the exact setting. Further, in the approximate setting, twin networks are able to perform queries in a single inference step without requiring a potentially costly or second inference step. However, in particular cases, twin networks amount to exactly the same inference as in the standard case. We consider the implications for scenarios like Reinforcement Learning, and how twin networks can make counterfactual inference in these scenarios more efficient. We conclude that twin networks provide an elegant representation that motivate the representation of counterfactual inference depending on the nature of the counterfactual query.

1 Introduction

Answering counterfactual queries—ones that posit an unobserved *what if?* scenario—is of interest in general reasoning and applied domains. For example, counterfactual queries can powerfully answer questions of individualized treatment effect analysis [1, 2, 3, 4], legal responsibility analysis [5, 6], fairness in machine learning models [7, 8], and help with learning and planning [9, 10]. Counterfactual analysis subsumes both observation and intervention, and thus offers a powerful framework for asking and answering questions both in and outside of the capability set of typical machine learning approaches.

Counterfactual queries allow one to determine whether certain outcomes *would have* occurred had some precondition been different. Such queries concern fictional scenarios as their antecedents are necessarily counter to some observed fact. For instance, given evidence $\mathcal{E} = e$, counterfactual inference allows one to compute the probability that a different outcome $\mathcal{E} = e'$ would have been observed—*counter to the fact* $\mathcal{E} = e$—had some hypothetical intervention taken place, forcing some precondition to be different. The standard approach to calculating the probability of such a counterfactual query in the Structural Causal Model framework was introduced by Pearl [11] and is known as the *abduction-action-prediction* method. First, the distributions of the model are updated under the observed evidence $\mathcal{E}$. Second, the desired intervention is applied to the model—forcing a

Preprint. Under review.

specific set of variable to take on a fixed value. Lastly, the updated model distributions are used to make predictions in the intervened model.

Despite the impact the algorithmization of counterfactuals has had on epidemiology, medicine, and machine learning, the abduction step of the above described standard approach requires a large amount of computational resources and memory—especially as conditioning on evidence standardly induces correlations between initially uncorrelated variables. Moreover, as this step has to be repeated for every new counterfactual query, such computational resources are continually required. Balke and Pearl [12] have introduced a method that removes this computationally expensive step and reduces computing counterfactual statements to performing ordinary inference on an larger causal model, known as a *twin network*, in which the factual and counterfactual worlds are jointly represented.

However, no investigation into the specific circumstances in which twin networks are be more advantageous than the standard method have been conducted. A priori it is not clear that twin networks are preferable to abduction-action-prediction, as twin networks require inference to be performed over a causal model whose graphical structure is twice as large as the original. Given that there are many counterfactual scenarios one could consider before making a decision, it is imperative that we have computationally efficient and memory-light methods of estimating such queries.

In this work, a comprehensive empirical and theoretical analysis of twin networks is undertaken. We show that in certain situations, twin networks are faster and less memory intensive by orders of magnitude than standard counterfactual inference, in both exact and approximate settings. In particular types of Structural Causal Models however, twin networks do not provide a computational advantage over standard methods. We conclude by considering the implications of these findings for scenarios like Reinforcement Learning and bandits, and how using twin networks might make counterfactual inference in these scenarios more efficient. Our results indicate that one should choose counterfactual inference methods depending on the dependency structure of one's causal model, as well as the counterfactual question of interest.

2 Structural Causal Models and counterfactual inference

Structural Causal Models, the do-operator, and counterfactuals will now formally be defined. See chapter 7 of Pearl's "Causality" [11] for a more in depth discussion.

Definition 1 (Structural Causal Model) *A causal model specifies:*

1. *a set of latent, or* noise*, variables $U = \{u_1, \ldots, u_n\}$, distributed according to $P(U)$.*
2. *a set of observed variables $V = \{v_1, \ldots, v_n\}$,*
3. *a directed acyclic graph G, called the* causal structure *of the model, whose nodes are the variables $U \cup V$,*
4. *a collection of functions $F = \{f_1, \ldots, f_n\}$, where f_i is a mapping from $U \cup V / v_i$ to v_i. The colelction F forms a mapping from U to V. This is symbolically represented as*

$$v_i = f_i(pa_i, u_i), \text{ for } i = 1, \ldots, n,$$

where pa_i denotes the parent nodes of the ith observed variable in G.

As the collection of functions F forms a mapping from noise variables U to observed variables V, the distribution over noise variables induces a distribution over observed variables, given by

$$P(v_i) := \sum_{u | v_i = f_i(\text{pa}_i, u)} P(u), \text{ for } i = 1, \ldots, n.$$

In this manner, we can assign uncertainty over observed variables despite the fact that the underlying dynamics is deterministic. Examples of causal structures are depicted in Fig. 1, where blue nodes represent observed variables and green nodes denote latent variables.

Definition 2 (Sub model) *Let M be a causal model, X a set of variables in V, and x a particular realization of X. A submodel M_x of M is the causal model with the same same noise variables U and observed variables V as M, but where the collection of functions F is replaced with F_x, where*

$$F_x = \{f_i : V_i \notin X\} \cup \{X = x\}$$

In the above, F_X is formed by deleting from F all functions f_i corresponding to members of the set X and replacing them with the set of constant functions $X = x$.

Definition 3 (Effect of action, do-operator) *Let M be a causal model, X a set of variables in V, and x a particular value of X. The effect of action $do(X = x)$ on M is given by the sub model M_x.*

That is, the *do*-operator forces variables to take certain values, regardless of the original causal mechanism. Probabilities involving the *do*-operator, such as $P(Y = y|do(X = x))$, correspond to evaluating ordinary probabilities in the sub model M_x, in this case $P_{M_x}(Y = y)$.

2.1 Standard method of counterfactual inference

Definition 4 (Counterfactual) *Let X and Y be two subsets of variables in V. The counterfactual sentence "Y would be y (in situation U), had X been x," is the solution $Y = y$ of the set of equations F_x, succinctly denoted $Y_x(U) = y$.*

As with observed variables in definition 1, the latent distribution $P(U)$ allows one to define the probabilities of counterfactual statements.

$$P(Y_x = y) = \sum_{u|Y_x(u)=y} P(u).$$

Moreover, one can define a distribution over counterfactual statements given seemingly contradictory evidence. For instance, consider the probability of the following counterfactual query $P(Y_{x'} = y'|X = x, Y = y)$, which reads "what is probability that Y would be y' if X had been x', given that Y and X were observed to be y and x respectively?". In the case of binary variables, this query is related to the probability of necessity and sufficiency [13]. Despite the fact that this query involves contradictory evidence, it is well defined in the Structural Causal Model framework. Indeed, it was shown in Ref [11] that $P(Y_{x'} = y'|X = x, Y = y)$ can be computed as follows:

$$P(Y_{x'} = y'|X = x, Y = y) = \sum_u P(Y_{x'}(u) = y')P(u|x, y).$$

That is, first update the distribution over noise variables under the observed evidence x, y to get $P(u|x, y)$, and use this to compute the expected value of the probability that $Y_{x'} = y'$.

The ability—given knowledge of the full causal model—to mathematically answer such queries is an incredibly powerful decision making tool. The following theorem provides the general solution to computing such probabilities.

Theorem 1 (Theorem 7.1.7 from Ref. [11]) *Given a functional causal model M with latent distribution $P(U)$, the conditional probability $P(Y_x|e)$ of a counterfactual sentence "if it were x, then Y," given evidence e, can be evaluated using the three following steps:*

1. ***Abduction—** Update the distribution over noise variables with evidence e to obtain $P(U|e)$*
2. ***Action—** Apply the action $do(x)$ to obtain sub model M_x*
3. ***Prediction—** Use the sub model M_x with updated noise distribution $P(U|e)$ to compute the probability of Y, the consequence of the counterfactual.*

2.2 Counterfactual inference with twin networks

Theorem 1 provides a three step procedure for computing the probability of any counterfactual statement given arbitrary (even seemingly contradictory) evidence. The first step, updating the distribution over noise variables and storing the resulting probabilities, requires a large amount of computational resources and memory—especially as conditioning on evidence induces correlations between initially uncorrelated variables. Moreover, as this step has to be repeated for every new counterfactual query, such computational resources are continually required. In the case of k binary variables and exact inference, $|P(U|e)|$ is 2^{k-1}, which for $k = 30$ is ~ 500 million.

Balke and Pearl [12] have introduced a method that removes this computationally expensive step and reduces computing counterfactual statements to performing Bayesian inference on an associated

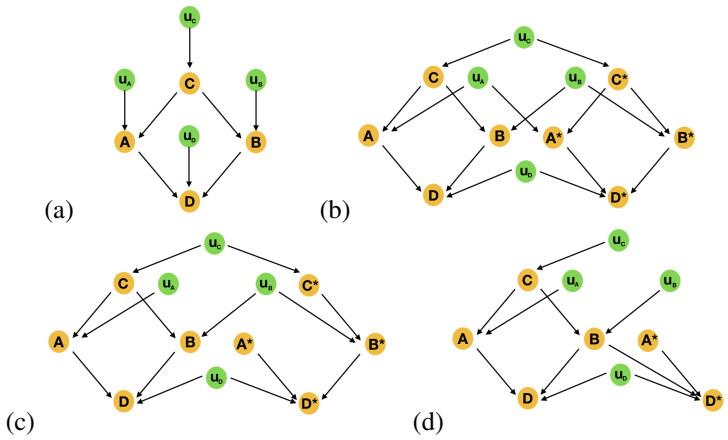

Figure 1: Orange nodes represent observed variables & green, noise variables (a) Causal structure discussed in sec 2.2 (b) Twin Network (c) Intervention in counterfactual network (d) node-merging.

causal model, known as a *twin network*. The twin network method employs two interlinked networks, one representing the real factual world and the other the counterfactual world being queried. These two networks are identical in structure and share the same noise variables—as noise variables are not modified by interventions on observed variables. The observable variables from the original model are duplicated and labeled distinctly, as they may obtain different values in the real and counterfactual networks. After the counterfactual network undergoes the desired intervention, computing a counterfactual statement in the original model is reduced to performing standard inference in the twin network, as variables in the real and counterfactual networks can be explicitly compared.

To illustrate the above discussion, a twin network for Fig. 1(a) with no interventions made in the counterfactual world is depicted in Fig. 1(b). A twin network with an intervention on the counterfactual variable A^* is depicted in Fig. 1(c). The following example will serve to explicate the above discussion. Consider the causal model with causal structure depicted in Fig. 1(a), where all variables are binary. The counterfactual statement to be computed is $P\left(D_{A=0} = 0 \mid D = 1\right)$ the probability that D would have been 0 had A been forced to be 0, given that $D = 1$ was observed.

The twin network approach to this problem first constructs the linked factual and counterfactual networks depicted in Fig. 1(b), with counterfactual nodes represented with a superscript *. The intervention $do(A^* = 0)$ is then performed in the counterfactual network. That is, all ingoing arrows from the parents of A^* to A^* are removed and A^* is set to 0. This is graphically depicted in Fig. 1(c). The above counterfactual query is reduced to the following conditional probability in the twin network: $P\left(D^* = 0 \mid D = 1\right)$, which can be computed using standard belief propagation techniques. This allows the counterfactual query to be computed without the need to update and store the joint distribution over u_C nd u_A (conditioning on $D = 1$ induces correlations between them).

Despite this, it's not a priori clear that twin networks are always preferable to standard counterfactual inference. Computing a counterfactual probability using twin networks means performing inference over a causal structure almost twice as large as the original. However, twin networks as currently described contain a lot of redundant information in their causal structure. Indeed, nodes in the counterfactual network whose parents have not been affected by an intervention are exact copies of the original node in the factual network. Consider the variable v in factual world, and its counterfactual copy v^*. As $v = f(\text{pa}_v, u)$, and the same noise variable value is fed into both v and v^*, if the parents of v^* are unaltered by interventions then $v^* = f(\text{pa}_v, u) = v$. Hence v and v^* are exact copies of one another and can hence be *merged* together in the twin network. This process is known as *node-merging* [14]. The node-merged twin network from the above example is depicted in Fig. 1(d). Depending on the network topology, node merging can substantially reduce the size of the graph on which inference has to be performed to compute counterfactual queries in the Twin Network approach.

Our extensive experiments in subsequent sections show that when node-merging and d-separation are employed, the size of the graph and location of the intervention determine when twin networks are preferable to the standard abduction-action-prediction approach to counterfactual inference.

3 Experiments: comparing counterfactual inference cost in exact inference

As even for small graphs there is a combinatorially large space of possible graphs across various characteristics, we first evaluate the empirical effect of graph characteristics on the time cost of inference. From these experiments we make theoretical inferences about the effect of graph characteristics on the advantages of twin networks.

We first consider the simple case: relatively small graphs with binary-valued random variables that admit feasible exact inference. We start by randomly generating a Structural Causal Model as defined in Def. 1, and follow the method from melancon2000 to construct G (defined by k vertices $V := \{X_i\}_i^K$, and edges E). Each noise variable u_i receives a generating function that samples $u_i \sim \text{Bern}(p); p \sim U[0,1]$; each endogeneous vertex X_i receives a generating function $x_i = \text{Bern}(f(pa_i, u_i))$ such that $f(pa_i, u_i) = \frac{(\sigma((pa_i \oplus u_i) \cdot \boldsymbol{\theta}_i)}{\sigma(\sum \boldsymbol{\theta}_i)}$, which represents a sigmoid-constrained and normalized linear combination of $\boldsymbol{x}_{PA_i}$, the parent variables of x_i, and each linear coefficient in $\boldsymbol{\theta}_i$ drawn from $\sim Beta(5,5)$. We sample from this model 1 million times to construct conditional probability tables which we use to parameterize a Bayesian network.

When constructing the twin network, we merged nodes where possible, and removed nodes that are d-separated given evidence and intervention. We then observe a set of evidence $\mathcal{E}$, perform a single randomly-chosen intervention $do(X_i = x_i)$, & query the counterfactual distribution of Y_x^*. How these nodes were chosen changes depending on the experiment. We considered the effect of the graph size (number of nodes), graph density, the size and location of evidence observed, and the location of the intervention. In our evidence and intervention topology experiments, "location" was determined by a single parameter $p \in [0,1]$, where $k \sim \text{Bin}(|V|, p)$ determined the starting point for the evidence set or the location of the intervention. Similarly, k represented the size of the evidence set in the evidence size experiment. We used the Variable Elimination [15] implementation from the *pgmpy* Python package [16] & ignored overhead time costs other than the factor elimination steps of the factor product and marginalization. We further compared results through a separate enumeration procedure to confirm the results were not implementation specific.

We took 250 draws of graphs, evidence sets, and interventions per graph characteristic parameter (e.g. size). Inference was measured as the time it took to eliminate factors using the Variable Elimination algorithm on the Bayesian network; factor order was computed via the min-fill ordering heuristic. Finally, having inferred $P(U|e)$, we sample from it many times, simulate & parameterize a new Bayesian network, effectively removing the possibly intractable noise posterior, in order to estimate the most optimistic comparison of the standard method & twin networks.

To be clear, this experimental set-up replicates perhaps the only setting where $P(U|e)$ is small enough to make exact inference tractable. For example, even trinary-valued latents on a graph with 30 endogenous nodes would be parameterized by a conditional probability table with over 200 trillion possible values. Without the reparameterization described above, which reflects a mean-field approximation, it would be clear that for any large graph inference would be computationally intractable. It should be noted that this is for the purpose of deriving insights. Outside of these restrictions, it becomes effectively impossible to use exact standard counterfactual inference; this is the main proposed benefit of twin networks, as proposed in [12].

3.1 Results

As originally speculated in [12], twin networks become significantly advantageous as the graph grows, as can be seen in Fig. 2 (a). As shown in Fig. 3, this is because the abduction step grows in complexity faster than the prediction step and begins to dominate the time cost of performing standard counterfactual inference.

We additionally see in Fig. 4(a) that the position in the topological order where the intervention occurs has a significant effect on performance of twin networks. This is because node-merging reduces the size of the graph as possible. When the intervention occurs closer to the end of the topological order, its set of non-descendents grows. In the node-merged twin network with d-separated nodes

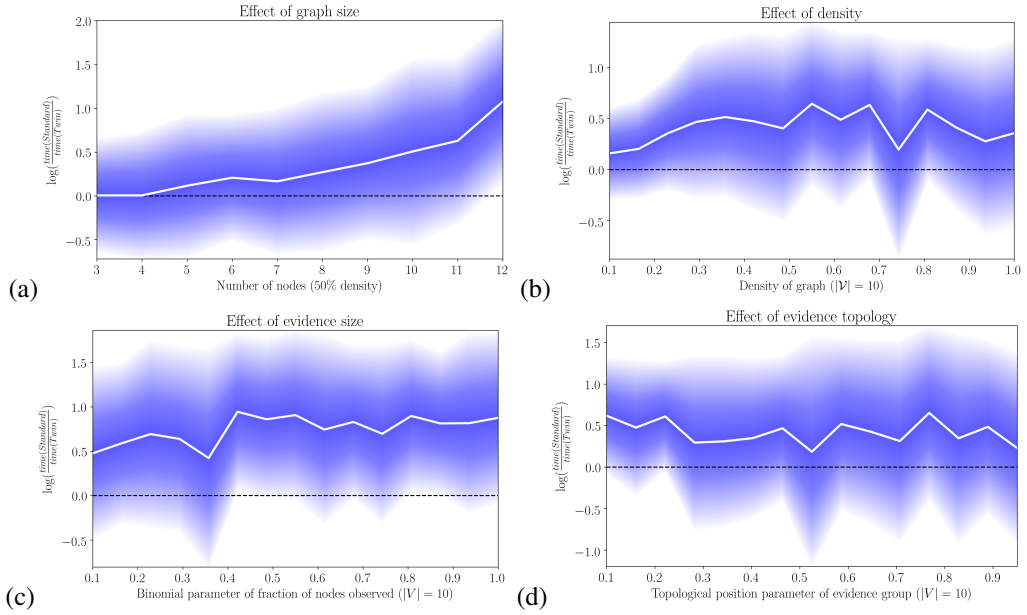

Figure 2: Here we plot the median of the $\log_{10}$ of the ratio of wall times of standard and twin network counterfactual inference. The shaded regions represent $\pm$ 2 standard deviations. The black dashed line denotes parity in the time each method takes; twin networks are superior above the black dashed line, and inferior below.

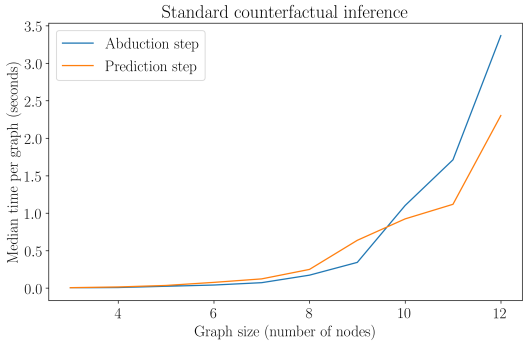

Figure 3: As the graph grows, computing $P(U|e)$ becomes costlier than the subsequent prediction.

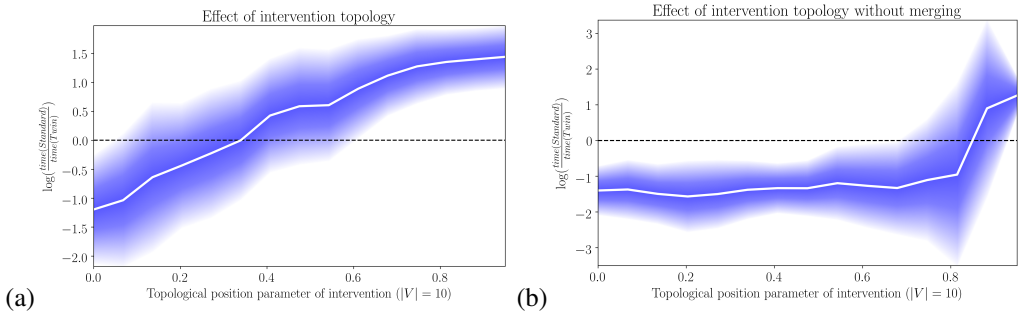

Figure 4: The effect of intervention topology on performance, with and without node merging. Twin networks without merging perform well only when the intervention is highly likely to d-separate the node of interest from a large portion of the graph, which here occurs when p is close to 1. The shaded regions represent $\pm$ 2 standard deviations. The black line denotes performance parity.

removed, there fewer factors that need to be eliminated. We validate this by observing the effect of removing node merging, as shown in Fig. 4(b). Without node merging, twin network inference begins to take significant time as in many cases we have to eliminate duplicate factors across a graph that approaches 36 nodes in our example. This is an important result: without node merging, twin network inference is not necessarily better than standard counterfactual inference.

Beyond these two effects, we see no consistent trend for density, evidence size, or evidence topology, as can be seen in Fig. 2 (b)-(d). In these cases, twin networks are consistently better, likely because they occur on a graph of size 10. This is expected as neither evidence size or topology change the nature of the counterfactual query.

In short, twin networks are valuable in this restricted, yet representative exact inference setting, as they avoid the complex abduction step, and because of the advantage of node merging. Additionally in some cases the intervention can d-separate the counterfactual node of interest from certain nodes, and twin networks can exploit this in the single inference step. These insights imply that, at least in the exact case, knowing the topology of the intervention, the complexity of noise inference, and the dependencies in the graph can guide a modeller towards saving time in inference. This theoretical grounding motivates further exploration of the real-world implications of these insights.

4 Discussion: twin networks in approximate inference

In almost all real-world cases, the graph is so large or the distributions of random variables so complex that exact inference is infeasible. Instead, one would use a sampling or variational-based approximate inference scheme to estimate variables of interest. However, in the straightforward Markovian Structural Causal Model setting (as used in Section 3), the counterfactual posterior $P(Y^*|e, do(X_i^* = x_i^*))$ will be the same via both twin network and standard counterfactual inference. This occurs whether or not one uses sampling-based approximate inference, or variational inference.

Consider importance sampling: we take the same proposal distribution Q over the k exogenous variables in both methods, and the same m samples from Q, $S_Q = \{\boldsymbol{q_m}\}_{i=1}^M$. Then, the importance weights $W_Q = \{\boldsymbol{w_i}\}_{i=1}^M$ will be the same in both methods as the likelihood is derived only from information in the factual network. This occurs because there is no additional information in the counterfactual network. This extends to Markov Chain Monte Carlo methods.

Similarly, this occurs in variational methods. The mean-field assumption in the standard method approximates the abduction step as $P(U|e) \approx Q(U) = \Pi_i^K Q(u_i)$; after one performs optimization, one must sample from $u \sim Q(U)$ in a new mutilated graph to calculate $P(Y|do(X_i = x_i; u)$. On the other hand, twin networks approximate a different query via the mean-field approximation, setting $P(Y^*|do(X_i^* = x_i^*), e, u) \approx Q(U) = \Pi_i^K Q(u_i)$. In a Structural Causal Model, with the strict assumption that all uncertainty is generated by the noise variables, then the above approximations reduce to the same approximation. In this situation, then the same approximate inference benefits of twin networks (namely parallelization and cheap sampling) apply in the variational case as well.

However, there are several cases where twin networks with approximate inference will lead to a different posterior distribution of the latents $P(U|e)$. First, in the variational case, the mean-field assumption implies that the resampling (prediction) step in standard counterfactual inference would samples from a conditionally independent posterior. This is an unlikely representation as the evidence likely makes noise variables dependent. This is avoided in twin networks as inference happens in a single step. Secondly, in the presence of "counterfactual conditioning" — setting hypothesized non-interventional evidence in the counterfactual world — the posteriors will be different as this evidence is incorporated in the first step, thus changing the likelihood. Third, for small samples or iterations, the prediction step in standard counterfactual inference involves bootstrapping from an approximate posterior, introducing approximation error.

Additionally, twin networks can leverage approximate inference speedups. First, twin networks avoid sampling from a potentially expensive $P(U|e)$. Second, propagating samples in twin networks is faster than in the standard case if parallelized. In the subgraph comprised of $\mathrm{desc}(X_{do})$ (the intervention node in the twin network of the graph, and its descendents), twin networks can propagate samples simultaneously through this subgraph and its counterpart subgraph in the factual side of the graph. Hence inference takes the amount of time the abduction step takes, while in the standard method one incurs the additional cost of propagating samples for inference in a new mutilated graph.

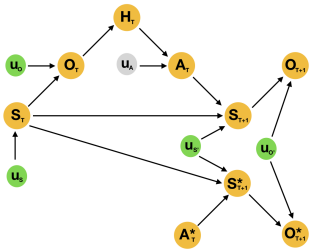

Figure 5: A merged twin network POMDP. Orange nodes are observed, green nodes are unobserved latent/noise nodes, and grey nodes are immutable observed latent nodes. By composing the counterfactual POMDP from [9] as a merged twin network, then conditioning on S_t allows one to avoid inference over previous variables as this d-separates O^*_{t+1} from nodes prior to $t-1$.

This is potentially advantageous for very large graphs. The same savings can be achieved with the standard method if one parallelizes an intervened subgraph and passes samples though it. However this exactly describes twin networks. As such, twin networks can be thought of as a way to represent standard counterfactual inference in an optimized & parallelized incarnation.

5 Twin network counterfactual inference in reinforcement learning

The result of our insights suggest that if one thinks deeply about the counterfactual intervention, inference method, and dependency structure of one's problem, one can save on inference about the noise variables. This has particular applications to sequential decision-mkaing scenaarios like Reinforcement Learning. For example, we take the scenario from [9]. Had any-time counterfactual inference been a goal from the beginning, the merged twin network representation makes clear that it suffices to maintain a belief over the state value in order to use a merged twin network to avoid the costly full-history abduction step. This can be seen in Fig. 5, as in the merged twin network conditioning on the state S_t d-separates the counterfactual node of interest O_{t+1} from all variables upstream of S_t (those with time index $\leq t-1$). This insight motivates a different approach in Reinforcement Learning to what information you keep track of if you intend to perform counterfactual inference at any point. A similar insight is made in the multi-agent case in [17].

Intuitively, this result is a sort of Markov butterfly effect: it is costly and difficult to infer the long-term impact of a change to a context far in the past. However, if a counterfactual intervention is recent, then it is easier to infer its counterfactual effect if one keeps track of the minimum beliefs necessary as shown by a merged twin network. This comes at the cost of representing and storing beliefs as the process evolves. This amounts to a trade off between persistent storage, and one-time but expensive computation. That is, an optimized twin network structure of a counterfactual problem may enable amortized inference in sequential environments.

6 Conclusion

We conducted, to our knowledge, the first empirical study of twin networks for counterfactual inference. In summary, twin networks are not a different method of counterfactual inference, but an elegant *way of representing* counterfactual inference that may avoid costly computation seen in the dominant abduction-action-prediction method. We showed that in the exact case, twin networks are computationally advantageous as the graph size increases and as the intervention topology allows for more compressed graphical representations. We discussed how twin networks reduce the computational load to perform the same counterfactual query in approximate inference under certain conditions. Last, we showed how node merging may motivate a different approach to modelling and information storage in sequential decision-making scenarios. These insights imply that those wishing to perform counterfactual inference should think about the graph size & representation, counterfactual query, dependency structure, and intervention location. Casting the inference task in a twin network may reduce unnecessary computation and extract counterfactual queries in parallel. We suggest further study on comparing counterfactual error between twin networks and the standard method in approximate inference, as well extensions to complicated noise structures.

References

[1] U. Shalit, F. D. Johansson, and D. Sontag, "Estimating individual treatment effect: generalization bounds and algorithms," *arXiv preprint arXiv:1606.03976*, 2016.

[2] F. Johansson, U. Shalit, and D. Sontag, "Learning representations for counterfactual inference," in *International Conference on Machine Learning*, pp. 3020–3029, 2016.

[3] P. Schulam and S. Saria, "Reliable decision support using counterfactual models," in *Advances in Neural Information Processing Systems*, pp. 1697–1708, 2017.

[4] J. G. Richens, C. M. Lee, and S. Johri, "Counterfactual diagnosis," *arXiv preprint arXiv:1910.06772*, 2019.

[5] H. Chockler and J. Y. Halpern, "Responsibility and blame: A structural-model approach," *Journal of Artificial Intelligence Research*, vol. 22, pp. 93–115, 2004.

[6] D. A. Lagnado, T. Gerstenberg, and R. Zultan, "Causal responsibility and counterfactuals," *Cognitive science*, vol. 37, no. 6, pp. 1036–1073, 2013.

[7] M. J. Kusner, J. Loftus, C. Russell, and R. Silva, "Counterfactual fairness," in *Advances in Neural Information Processing Systems*, pp. 4066–4076, 2017.

[8] N. Kilbertus, M. R. Carulla, G. Parascandolo, M. Hardt, D. Janzing, and B. Schölkopf, "Avoiding discrimination through causal reasoning," in *Advances in Neural Information Processing Systems*, pp. 656–666, 2017.

[9] L. Buesing, T. Weber, Y. Zwols, S. Racaniere, A. Guez, J.-B. Lespiau, and N. Heess, "Woulda, coulda, shoulda: Counterfactually-guided policy search," *arXiv preprint arXiv:1811.06272*, 2018.

[10] L. Bottou, J. Peters, J. Quiñonero-Candela, D. X. Charles, D. M. Chickering, E. Portugaly, D. Ray, P. Simard, and E. Snelson, "Counterfactual reasoning and learning systems: The example of computational advertising," *The Journal of Machine Learning Research*, vol. 14, no. 1, pp. 3207–3260, 2013.

[11] J. Pearl, *Causality (2nd edition)*. Cambridge University Press, 2009.

[12] A. Balke and J. Pearl, "Probabilistic evaluation of counterfactual queries," in *AAAI*, 1994.

[13] J. Pearl, "Probabilities of causation: three counterfactual interpretations and their identification," *Synthese*, vol. 121, no. 1-2, pp. 93–149, 1999.

[14] I. Shpitser and J. Pearl, "What counterfactuals can be tested," in *Proceedings of the Twenty-Third Conference on Uncertainty in Artificial Intelligence*, pp. 352–359, AUAI Press, 2007.

[15] D. Koller and N. Friedman, *Probabilistic Graphical Models: Principles and Techniques - Adaptive Computation and Machine Learning*. The MIT Press, 2009.

[16] A. Ankan and A. Panda, "pgmpy: Probabilistic graphical models using python," in *Proceedings of the 14th Python in Science Conference (SCIPY 2015)*, Citeseer, 2015.

[17] N. Jaques, A. Lazaridou, E. Hughes, C. Gulcehre, P. A. Ortega, D. Strouse, J. Z. Leibo, and N. de Freitas, "Social influence as intrinsic motivation for multi-agent deep reinforcement learning," 2018.