Causally Constrained Counterfactual Generation using ASP

Sopam Dasgupta[1], Farhad Shakerin[2], Elmer Salazar[1], Joaquín Arias[3] and Gopal Gupta[1]

[1]*The University of Texas at Dallas, Richardson, USA*
[2]*Microsoft, USA*
[3]*CETINIA, Universidad Rey Juan Carlos, Madrid, Spain*

Abstract

This research focuses on generating realistic and achievable *counterfactual explanations.* Given a negative outcome predicted by a machine learning model or decision system, the novel *Causally Constrained Counterfactual Generation (C3G)* approach generates (i) a counterfactual solution representing a positive outcome and (ii) identifies the necessary changes in feature values to transition from the negative to the positive outcome. The counterfactuals produced by the *Causally Constrained Counterfactual Generation (C3G)* approach respect causal constraints among features and adhere to user-defined preferences regarding feature alterations, accounting for the ease of modifying certain features over others and the associated costs. *C3G* utilizes answer set programming (ASP) and the s(CASP) goal-directed ASP system to automatically generate counterfactual explanations from rules generated by *rule-based machine learning (RBML)* algorithms. Using *rule-based machine learning (RBML)* algorithms, *C3G* models causal dependencies between features, ensuring the realism of the solutions. This paper discusses the current status of the research and presents preliminary results.

Keywords

Counterfactual Reasoning, Causality, Machine Learning, Answer Set Programming

1. Introduction

Predictive models in automated decision-making, such as job filtering or loan approval, often function as black boxes, making the reasoning behind decisions difficult to understand. Given the significant consequences of these decisions, affected individuals desire clear explanations for undesired/negative outcomes, highlighting the need for transparency. Some approaches [1] propose generating counterfactuals to explain decisions and guide users toward desired outcomes.

We introduce a framework called *Causally Constrained Counterfactual Generation (C3G)*, which generates counterfactual explanations using *rule-based machine learning (RBML)* algorithms.

4rd Workshop on Goal-directed Execution of Answer Set Programs (GDE'24), October 12, 2024
sopam.dasgupta@utdallas.edu (S. Dasgupta); fshakerin@microsoft.com (F. Shakerin); ees101020@utdallas.edu (E. Salazar); joaquin.arias@urjc.es (J. Arias); gupta@utdallas.edu (G. Gupta)
0009-0008-3594-5430 (S. Dasgupta); 0000-0003-4148-311X (J. Arias); 0000-0001-9727-0362 (G. Gupta)

Our framework addresses the question: "What changes can achieve a desired outcome from an undesired one?" by modelling two scenarios: the current state with a negative outcome and an imagined state with a positive outcome. The aim is to transition from the negative to the positive scenario while assuming a static decision-making process. This is accomplished by altering input feature values and considering their *causal dependencies*.

C3G employs commonsense reasoning through Answer Set Programming (ASP) [2], specifically using the goal-directed s(CASP) ASP system [3].

2. Background

2.1. Counterfactual Reasoning

Humans use explanations to understand decisions. Counterfactual explanations offer meaningful insights to understand a decision and guide actions to change the outcome to a desired one. For example, in the case of being denied a loan, a counterfactual explanation might state: "If John were married, his loan application would have been approved."

For a binary classifier used for prediction, given by $f : X \rightarrow \{0, 1\}$, we define a set of counterfactual explanations $\hat{x}$ for a factual input $x \in X$ as $CF_f(\hat{x}) = \{\hat{x} \in X | f(x) \neq f(\hat{x})\}$. This set of counterfactual explanations contains all the inputs ($\hat{x}$) that lead to a different prediction under f compared to the original input x.

We demonstrate how counterfactual reasoning can be performed using the s(CASP) query-driven predicate ASP system [3] while accounting for *causal dependencies* between features. By leveraging s(CASP)'s ability to compute *dual rules* (as described in Section 2.3), which enable the execution of negated queries, counterfactual explanations are naturally obtained. Given a predicate p defined as a rule in ASP, its corresponding dual rule allows us to prove $\neg$ p, where $\neg$/`not` represents *negation as failure* [4].

2.2. Causality

MINT [5] showed that ignoring causal relations in counterfactual explanations produces unrealistic results. *MINT* focused on generating counterfactual explanations through a series of interventions, providing realistic paths to change the predicted label. In earlier approaches [6, 7], assumptions were made that changes from interventions were independent across features, which may not hold in reality. Hence, causal relationships should be considered to generate realistic counterfactual explanations. For instance, changing one's marital status without considering related features like relationship status and gender might not yield realistic results. Modeling these causal relationships ensures that downstream changes are realistically represented.

2.3. ASP, s(CASP) and Common Sense Reasoning

Answer Set Programming (ASP) is a paradigm for knowledge representation and reasoning, widely used in automating commonsense reasoning [8, 9, 2]. We use ASP to encode knowledge about features, domains, properties, decision-making, and causal rules, facilitating the

automatic generation of counterfactual explanations. The s(CASP) system, a goal-directed ASP variant, operates in a top-down, query-driven manner without grounding [3, 10]. It supports commonsense and counterfactual reasoning using proof trees and adopts program completion by introducing **dual rules**- for every rule that says $p \Rightarrow q$, add a complementary rule saying $\neg p \Rightarrow \neg q$. This ensures that q is $TRUE$ *"if and only if"* p is $TRUE$.

Commonsense knowledge in ASP is represented using default rules, integrity constraints, and multiple possible worlds. For an introduction to ASP, see Gelfond and Kahl [10, 2]; for a detailed overview of s(CASP), see [11, 3]."

2.4. FOLD-SE

FOLD-SE [12] is an efficient and explainable *rule-based machine learning (RBML)* algorithm for classification tasks. It generates a set of default rules (a stratified normal logic program) from input data (numerical and categorical). The explainability obtained through FOLD-SE is scalable. It maintains a small number of rules and literals, regardless of dataset size, while offering accuracy comparable to other *RBML* approaches like RIPPER [13] and traditional models like XGBoost [14] and Multi-Layer Perceptrons (MLP), with the added advantage of explainability.

3. Overview

3.1. The Problem

When an individual (represented as a set of features) receives a negative decision (e.g., loan denial), they may seek changes to achieve a positive outcome. *C3G* identifies these changes automatically. The negative decision state is termed the *pre-intervention state* i, while the positive outcome states are *post-intervention states* $g \in G$. For instance, if John is denied a loan (i), *C3G* models various scenarios (G) where he is approved. The negative decision should hold in i (`?- reject_loan(john)` = `True`) and not in any state in G (`?- reject_loan(john)` = `False`). The goal is to determine the interventions, or feature changes, required to transition from i to $g \in G$.

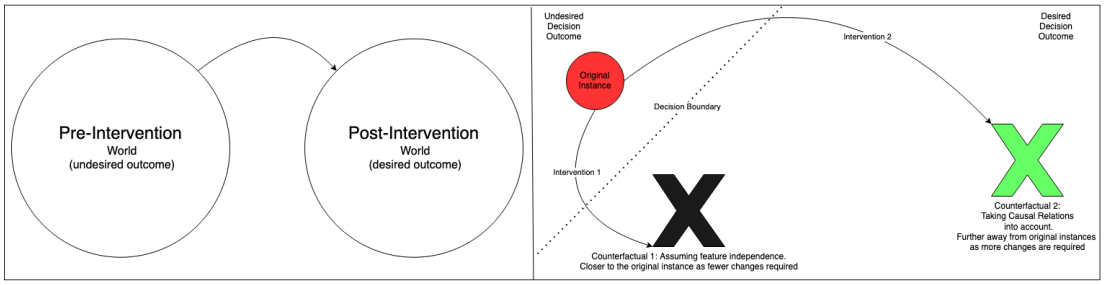

Figure 1: Left: Transition from *Pre-Intervention World* to the *Post-Intervention World.* ***Right:*** *Intervention takes the original instance to the other side of the decision boundary. With feature independence, the new counterfactual is closer to the original instance. With causal dependencies, the new counterfactual is further away as more changes are made to the original instance.*

3.2. Solution- *C3G* Approach

Given a decision query (e.g., '`?- reject_loan/1`') that succeeds (negative outcome), *C3G* identifies the state where this query fails, i.e., the query '`?- not reject_loan/1`' succeeds, representing the post-intervention state (g). In ASP terms, this involves finding the necessary feature changes while considering causal dependencies to transition from a state where the query ('`?- reject_loan/1`') is True to one where its negation ('`?- not reject_loan/1`') is True. The transition must comply with the given rules, using the s(CASP) query-driven ASP system [3], which supports negated queries constructively.

The *C3G* approach involves transitioning from a pre-intervention state (i) to a post-intervention state (g), with each state represented as feature-value pairs (e.g., John { Debt: $> \$1000$, Credit score: 600; Age: 24}). Multiple post-intervention states (G) can represent different positive outcomes. The objective is to change feature values to convert a negative decision (state i) into a positive one (state g), ensuring that '`?- not reject_loan(john)`' succeeds for $g \in G$.

Example: In a loan application scenario, John {$> \$10000$ debt, $\$40000$ balance, 599 credit score} is denied a loan because his balance is under $\$60000$ and his credit score is below 600. A naive solution might suggest raising John's balance to $\$60000$ and his credit score to 620, but directly altering the credit score is unrealistic. Considering causal dependencies, such as clearing debt to improve credit scores, *C3G* suggests a feasible intervention: John {no debt, $\$60000$ balance, 620 credit score}, resulting in loan approval. This intervention adjusts John's bank balance to $\$60000$ and clears his debt, resulting in a valid counterfactual scenario, as shown in Fig. 1."

4. Experiments

Dataset	Adult			Titanic			Dropout		
Details	# of Features Used	Training Size	Avg. Time Taken (ms)	# of Features Used	Training Size	Avg. Time Taken (ms)	# of Features Used	Training Size	Avg. Time Taken (ms)
FOLD-SE:	6	26048	291	3	891	84	4	3539	67
RIPPER:	12	26048	2869	1	891	14	16	3539	807
Dataset	Voting			Cars			Mushroom		
Details	# of Features Used	Training Size	Avg. Time Taken (ms)	# of Features Used	Training Size	Avg. Time Taken (ms)	# of Features Used	Training Size	Avg. Time Taken (ms)
FOLD-SE:	5	348	779	4	1382	807	5	6499	600
RIPPER:	2	348	75	6	1382	75	9	6499	3911

Table 1
Time for Computing the Counterfactual for Various Datasets

By using decision rules that provide the *original instance*, we use *C3G* to generate the *original instance-counterfactual* pairs, indicating possible solutions from a negative outcome to a counterfactual. It calculates the cost of reaching the counterfactual while allowing flexibility in the types of interventions permitted in the process of generating such counterfactuals. We applied our *C3G* methodology to rules generated by *RBML* algorithms, specifically FOLD-SE [12] and RIPPER [13], using datasets such as adult[15], car[16], titanic[17], dropout[18], mushroom

[19] and voting[20]. In the adult dataset, which includes demographic information with labels indicating income ('=<$50k/year' or '>$50k/year'), our *RBML* algorithms generate rules to predict income. Given decision-making rules specifying an undesired outcome ('=<$50k/year'), the goal is to find a path to a counterfactual instance where the income is '>$50k/year'. We have shown our results in Table 1.

For each dataset in Table 1, there are 3 columns denoting the following: 1) **Number of Features:** Count of the number of features that were used in generating *original instance-counterfactual* pairs. This depends on the features defined in the decision making rules and causal rules; 2) **Size of Training Data:** Size of the training data used to generate the decision making and causal rules; and 3) **Time Taken:** Average Time taken to produce a *(original instance, counterfactual)* pair.

As shown in Table 1, our *C3G* framework generates counterfactuals regardless of the *RBML* algorithm that specifies the decision making rules.

5. Discussion and Related Work

Existing approaches enhance transparency by explaining undesired outcomes through counterfactuals, using model-specific or optimization-based methods [1, 21, 22]. *Actionable Recourse* [6] focuses on actionable changes, while *MACE* [7], a model-agnostic method, generates counterfactuals considering feature immutability, avoiding unrealistic suggestions like changing 'gender' or 'age.' *CLEAR* [23] uses counterfactuals to improve model performance. However, both *Actionable Recourse* and *MACE* assume feature independence, which does not reflect real-world causal dependencies.

MINT [5] models causal dependencies for realistic counterfactuals but lacks the 'if and only if' property necessary for incorporating causal effects. Some methods [24] use Answer Set Programming (ASP) to address this but rely on grounding, which can disconnect variables. In contrast, *C3G* uses the goal-directed s(CASP) system, supporting complex logic without grounding. While some frameworks [25] generate contrastive explanations in ASP by explaining why one outcome occurred instead of another, *C3G* goes further by incorporating causal dependencies, ensuring counterfactuals are both realistic and achievable.

Our framework, *Causally Constrained Counterfactual Generation (C3G)*, utilizes Answer Set Programming (ASP) to generate counterfactuals, accommodating any *rule-based machine learning (RBML)* algorithm and allowing for user-defined rules. *C3G* ensures realistic counterfactuals by accounting for causal dependencies. This paper demonstrates how s(CASP) can model complex tasks, such as imagining possible scenarios and reasoning about counterfactual situations, to provide detailed explanations for decisions. Further details on the methodology and code can be found in the work by Dasgupta et al. [26].

Future work includes generating a path to the counterfactual by providing a step-by-step guide on the necessary changes and their causal effects [27]. We also plan to explore computing counterfactuals for image classification tasks, inspired by the work of Padalkar et al. [28, 29]

6. Conclusion

This paper addresses the challenge of generating counterfactual explanations for undesired outcomes predicted by machine learning models. Our framework, *Causally Constrained Counterfactual Generation (C3G)*, utilizes ASP and the goal-directed s(CASP) system to generate counterfactual explanations, regardless of the *RBML* algorithm used. By supporting negation as failure and dual rules, s(CASP) generates alternative worlds to find reachable counterfactuals. We methodically demonstrate how to reach counterfactuals, identify and constrain features to be altered, and measure the cost of interventions. This approach allows specifying intervention costs and integrating user-defined rules, enhancing the transparency and explainability of machine learning models.

References

[1] S. Wachter, B. D. Mittelstadt, C. Russell, Counterfactual explanations without opening the black box: Automated decisions and the GDPR, CoRR abs/1711.00399 (2017). URL: http://arxiv.org/abs/1711.00399. arXiv:1711.00399.

[2] M. Gelfond, Y. Kahl, Knowledge representation, reasoning, and the design of intelligent agents: Answer Set Programming approach, Cambridge Univ. Press, 2014.

[3] J. Arias, M. Carro, E. Salazar, K. Marple, G. Gupta, Constraint Answer Set Programming without Grounding, Theory and Practice of Logic Programming 18 (2018) 337–354. doi:10.1017/S1471068418000285.

[4] J. W. Lloyd, Foundations of logic programming, in: Symbolic Computation, 1987. URL: https://api.semanticscholar.org/CorpusID:46408498.

[5] A. Karimi, B. Schölkopf, I. Valera, Algorithmic recourse: from counterfactual explanations to interventions, in: FAccT '21, ACM, ????

[6] B. Ustun, A. Spangher, Y. Liu, Actionable recourse in linear classification, CoRR abs/1809.06514 (2018).

[7] A. Karimi, G. Barthe, B. Balle, I. Valera, Model-agnostic counterfactual explanations for consequential decisions, in: The 23rd International Conference on Artificial Intelligence and Statistics, AISTATS 2020, Proceedings of Machine Learning Research, PMLR, 2020.

[8] G. Brewka, T. Eiter, M. Truszczynski, Answer set programming at a glance, Commun. ACM 54 (2011) 92–103.

[9] C. Baral, Knowledge representation, reasoning and declarative problem solving, Cambridge University Press, 2003.

[10] G. Gupta, E. Salazar, S. C. Varanasi, K. Basu, J. Arias, F. Shakerin, R. Min, F. Li, H. Wang, Automating commonsense reasoning with asp and s(casp) *, 2022. URL: https://api.semanticscholar.org/CorpusID:251793743.

[11] J. Arias, M. Carro, Z. Chen, G. Gupta, Modeling and reasoning in event calculus using goal-directed constraint answer set programming, Theory and Practice of Logic Programming (????).

[12] H. Wang, G. Gupta, FOLD-SE: an efficient rule-based machine learning algorithm with scalable explainability 14512 (2024) 37–53.

[13] W. W. Cohen, Fast effective rule induction, in: Machine Learning, Proceedings of the Twelfth International Conference on Machine Learning, Morgan Kaufmann, ????
[14] T. Chen, C. Guestrin, Xgboost: A scalable tree boosting system, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2016.
[15] B. Becker, R. Kohavi, Adult, UCI Machine Learning Repository, 1996. DOI: https://doi.org/10.24432/C5XW20.
[16] M. Bohanec, Car Evaluation, UCI Machine Learning Repository, 1997. DOI: https://doi.org/10.24432/C5JP48.
[17] W. Cukierski, Titanic - machine learning from disaster, 2012. URL: https://kaggle.com/competitions/titanic.
[18] V. Realinho, M. Vieira Martins, J. Machado, L. Baptista, Predict students' dropout and academic success, UCI Machine Learning Repository, 2021. DOI: https://doi.org/10.24432/C5MC89.
[19] Mushroom, UCI Machine Learning Repository, 1987. DOI: https://doi.org/10.24432/C5959T.
[20] Congressional Voting Records, UCI Machine Learning Repository, 1987. DOI: https://doi.org/10.24432/C5C01P.
[21] G. Tolomei, F. Silvestri, A. Haines, M. Lalmas, Interpretable predictions of tree-based ensembles via actionable feature tweaking, in: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2017.
[22] C. Russell, Efficient search for diverse coherent explanations, in: Proceedings of the Conference on Fairness, Accountability, and Transparency, FAT* '19, Association for Computing Machinery, 2019.
[23] A. White, A. S. d'Avila Garcez, Measurable counterfactual local explanations for any classifier, in: ECAI 2020 - 24th European Conference on Artificial Intelligence, IOS Press, 2020.
[24] L. E. Bertossi, G. Reyes, Answer-set programs for reasoning about counterfactual interventions and responsibility scores for classification, in: N. Katzouris, A. Artikis (Eds.), Inductive Logic Programming - 30th International Conference, ILP 2021, Virtual Event, October 25-27, 2021, Proceedings, volume 13191 of *Lecture Notes in Computer Science*, Springer, 2021, pp. 41–56. URL: https://doi.org/10.1007/978-3-030-97454-1_4. doi:10.1007/978-3-030-97454-1_4.
[25] T. Eiter, T. Geibinger, J. Oetsch, Contrastive explanations for answer-set programs, in: Logics in Artificial Intelligence JELIA, LNCS, 2023, pp. 73–89.
[26] S. Dasgupta, F. Shakerin, J. Arias, E. Salazar, G. Gupta, Counterfactual generation with answer set programming, CoRR abs/2402.04382 (2024). URL: https://doi.org/10.48550/arXiv.2402.04382. doi:10.48550/ARXIV.2402.04382. arXiv:2402.04382.
[27] S. Dasgupta, J. Arias, E. Salazar, G. Gupta, Cogs: Causality constrained counterfactual explanations using goal-directed ASP, CoRR abs/2407.08179 (2024). URL: https://doi.org/10.48550/arXiv.2407.08179. doi:10.48550/ARXIV.2407.08179. arXiv:2407.08179.
[28] P. Padalkar, H. Wang, G. Gupta, NeSyFOLD: A framework for interpretable image classification, in: Proc. AAAI, 2024.
[29] P. Padalkar, H. Wang, G. Gupta, Using logic programming and kernel-grouping for improving interpretability of CNNs, in: Proc. PADL, LNCS, 2024.