Explaining Deep Graph Networks with Molecular Counterfactuals

Danilo Numeroso
University of Pisa
danilo.numeroso@phd.unipi.it

Davide Bacciu
University of Pisa
bacciu@di.unipi.it

Abstract

We present a novel approach to tackle explainability of deep graph networks in the context of molecule property prediction tasks, named MEG (Molecular Explanation Generator). We generate informative counterfactual explanations for a specific prediction under the form of (valid) compounds with high structural similarity and different predicted properties. We discuss preliminary results showing how the model can convey non-ML experts with key insights into the learning model focus in the neighborhood of a molecule.

1 Introduction

The prediction of functional and structural properties of molecules by machine learning models for graphs is a research field with long-standing roots [1]. Much of current research on the topic relies on Deep Graph Networks (DGNs) [2, 3], as they provide a flexible and scalable means to learn effective vectorial representations of the molecules. This has resulted in a trail of works targeting increasing levels of effectiveness, breadth and performance in the prediction of chemo-physical properties [4]. The scarce intelligibility of such models and of the internal representation they develop can, however, act as a show-stopper for their consolidation, e.g. to predict safety-critical molecule properties, especially when considering well known issues of opacity in DGN assessment [5]. In this respect, attention is building towards the development of interpretability techniques specifically tailored to DGNs. While some DGN shows potential for interpretability *by-design* thanks to its probabilistic formulation [6], the majority of works in literature take a neural-based approach which requires the use of an *external* model explainer. GNNExplainer [7] is the front-runner of the model-agnostic methods providing local explanations to neural DGNs in terms of the sub-graph and node features of the input structure which maximally contribute to the prediction. RelEx [8] extends GNNExplainer to surpass the need of accessing the model gradient to learn explanations. GraphLIME [9] attempts to create locally interpretable models for node-level predictions, with application limited to single network data. This paper fits into this pioneering field of research by taking a novel angle to the problem, targeting the generation of interpretable insights for the primary use of the experts of the molecular domain. We build our approach upon the assumption that a domain expert would be interested in understanding the model prediction for a specific molecule based on differential case-based reasoning against counterfactuals, i.e. similar structures which the model considers radically different with respect to the predicted property. Such counterfactual molecules should allow the expert to understand if the structure-to-function mapping learned by the model is coherent with the consolidated domain knowledge, at least for what pertains a tight neighborhood around the molecule under study. We tackle the problem of counterfactual molecule generation by introducing an explanatory agent based on reinforcement learning (RL) [10]. This explanatory agent has access to the internal representation of the property-prediction model as well as to its output and uses this information to guide the exploration of the molecular structure space to seek for the nearest counterfactuals. Our approach is specifically thought for molecular applications and the RL agent leverages domain knowledge to constrain the generated explanations to be valid molecules.

Machine Learning for Molecules Workshop at NeurIPS 2020. https://ml4molecules.github.io

We test our explainer on DGNs tackling the prediction of toxicity (classification task) and solubility (regression task) of chemical compounds.

2 Molecular Explanation Generator (MEG)

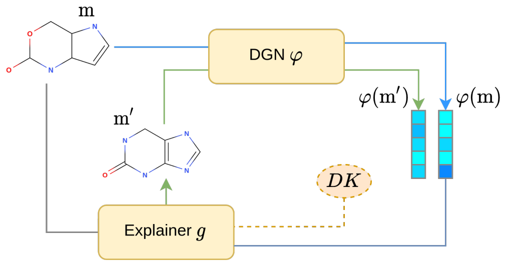


Figure 1: DGN φ is a trained molecule property predictor, whereas the Explainer g is a generative agent producing counterfactuals, constrained by prior domain knowledge DK.

The overall architecture of our explanation framework, named MEG, is depicted in Figure 1. Here we denote with $\varphi : \mathcal{I} \rightarrow \mathcal{Y}$ a DGN that is fit to solve a molecular property prediction task. $\mathcal{I}$ represents the space of (labelled) molecule structures and $\mathcal{Y}$ is the task-dependent output space. The Explainer is an RL agent implementing a generative function $g : \mathcal{I} \rightarrow \mathcal{I}$ targeting the generation of counterfactual explanations. Molecular counterfactuals ought to satisfy three properties: (i) they need to resemble the molecule under study; (ii) predicted properties on counterfactuals must differ substantially from those predicted on the input one; (iii) molecular counterfactuals need to be in compliance with chemical constraints. To this end, the agent g receives information about an input molecule m and its associated prediction score $\varphi(m)$, and generates a molecular counterfactual m', leveraging prior domain knowledge to ensure validity of the generated sample. Counterfactual generation is formalised as a maximisation problem in which, given a target molecule m with prediction $\varphi(m)$, the generator g is trained to optimize:

$$\arg\max_{\theta} \mathcal{L}\big(\varphi(m), (\varphi \circ g)(\cdot \mid \theta)\big) + \mathcal{K}\big[m, g(\cdot \mid \theta)\big]. \tag{1}$$

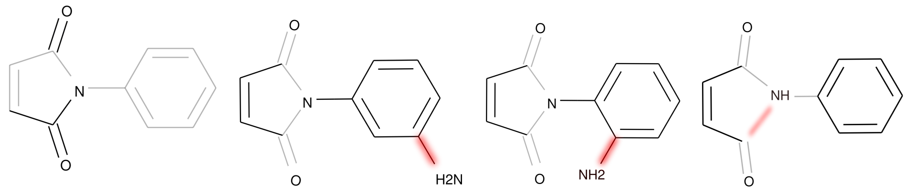
The composition $(\varphi \circ g)(\cdot \mid \theta)$ formalizes the model φ counter-predictions, made over the counterfactuals produced by g. Given the counterfactual $m' = g(\cdot \mid \theta)$ we rewrite Equation 1 as

$$\arg\max_{m'} \mathcal{L}\big(\varphi(m), \varphi(m')\big) + \mathcal{K}\big[m, m'\big] \tag{2}$$

where $\mathcal{L}$ is a measure of prediction disagreement between the molecule m and its counterfactual m', while $\mathcal{K}$ measures (m, m') similarity. In our framework, m is used to bootstrap the generative process in g which operates on the current candidate counterfactual with graph editing operations under domain knowledge constraints. Given the non-differentiable nature of the graph alterations, we model g through a multi-objective RL problem [11], that takes the form of an MDP($\mathcal{S}$, $\mathcal{A}$, $\mathcal{Q}$, π, $\mathcal{R}$, γ). Apart from well known differentiability issues of graph operations, the generator g is modeled as an RL agent for its ease in modelling and handling multi-objective optimization. This allows to easily steer towards the generation of counterfactuals optimizing several properties at a time [12, 13]. Since we are interested in generating counterfactuals that are compliant to chemical knowledge, the action space $\mathcal{A}$ is restricted so as to only retain actions that preserve the chemical validity of the molecule. To this end, we base the implementation of our agent on the MolDQN [14] model, that is an RL-based approach to molecule graph generation leveraging double Q-learning [15]. At each step, the reward function $\mathcal{R}$ exploits the prediction from φ so as to notify the agent of its current performance, emitting a scalar reward. In our design, $\mathcal{R}$ binds together a term regulating the change in prediction scores, which is inherently task-dependent, with a second term controlling similarity between the original molecule and its counterfactual, as presented in Equation 1. Currently, we have explored two formulations for the latter term. The former leverages the Tanimoto similarity over the Morgan fingerprints [16]. The latter is a φ-model dependent metric exploiting the encoding of molecules in the DGN internal representation. An advantage of using the latter approach is that it takes into account the model's own perception of structural similarity between molecules.

The leftmost term $\mathcal{L}$ in Equation 2 can be specialized for classification and regression tasks. As regards classifications, given a set of classes $\mathcal{C}$, a model φ emits a probability distribution $\varphi(\cdot) = \boldsymbol{y} = [y_0, ..., y_{|\mathcal{C}|}]$ over the predicted classes. In this case, given an input-prediction pair $\langle m, c = \arg\max_{c \in \mathcal{C}} \varphi(m) \rangle$, the generator is trained to produce counterfactual explanations m' minimising the prediction score for class c, as follows

$$\arg\max_{m'} -\alpha y_c + (1 - \alpha)\mathcal{K}[m', m] \tag{3}$$



(a) A0 (b) A1 (c) A2 (d) A3

Figure 2: Experimental results for the Tox21 sample, reported in Table 1.

where $\alpha \in [0, 1]$ is a hyper-parameter weighing the two parts. Hence, the model φ returns at each step a smooth reward, which is actually the inverse of the probability of m' belonging to class c. Differently, for a regression task, the objective function can be defined as

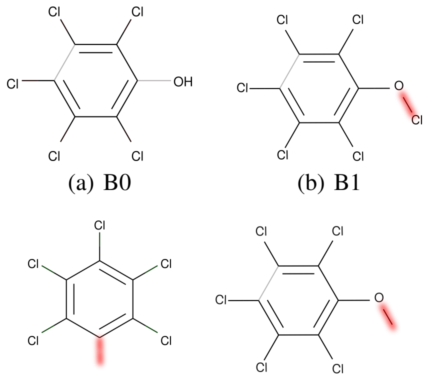
$$\arg\max_{m'} \alpha \operatorname{sgn}\left(\|s_{m'} - s\|_1 - \|s_m - s\|_1\right)\|s_{m'} - s_m\|_1 + (1-\alpha)\mathcal{K}[m', m] \tag{4}$$

where sgn is the sign function, s is the regression target, and s_m and $s_{m'}$ are the predicted values for the original molecule and its counterfactual, respectively. The sign function is needed to prevent the agent from generating molecules whose predicted scores move towards the original target, by providing negative rewards.

The main use of counterfactual explanations is to provide insights into the function learned by the model φ. In this sense, a set of counterfactuals for a molecule may be used to: (i) identify changes to the molecular structure leading to substantial changes in the properties, enabling domain experts to discriminate whether the model predictions are well founded; (ii) validate existing interpretability approaches, by running them on both the original input graph and its related counterfactual explanations. The main idea behind this latter point is that a local interpretation method may provide explanations that work well within a very narrow range of the input, but do not give a strong suggestion on a wider behaviour. To show evidence and usefulness of such a differential analysis, in the following section we use our counterfactuals to assess the quality of explanations given by GNNExplainer [7]. Given the undirected nature of the graphs in our molecular application, we restrict the original GNNExplainer model to discard the effect of edge orientation on the explanation.

3 Experimental Evaluation

We discuss a preliminary assessment of our explanations on two popular molecular property prediction benchmarks: Tox21 [17], addressing toxicity prediction as a binary classification task, and ESOL [18], that is a regressive task on water solubility of chemical compounds. Preliminarily, we scanned both datasets to filter non-valid chemical compounds. We considered structures to be valid molecules if they pass the RDKit [19] sanitization check. In the end, Tox21 comprises 1900 samples, equally distributed among the two classes, while ESOL includes 1129 compounds.



(a) B0 (b) B1

(c) B2 (d) B3

Figure 3: ESOL sample alongside its counterfactuals (B1-3). Quantitative results are reported in Table 1.

The trained DGN comprises three GraphConv [20] layers with ReLu activations, whose hidden size is 256 per layer for Tox21, and 128 for ESOL. The network builds a layer-wise molecular representation via concatenation of max and mean pooling operations, over the set of node representations. The final neural encoding of the molecule is obtained by sum-pooling of the intermediate representations. This neural encoding is then feed to a three-layer feed-forward network, with hidden sizes of [128, 64, 32], to perform the final property prediction step. The trained DGNs achieved 87% of accuracy and 0.52 MSE over the Tox21 and ESOL test sets, respectively. All experiments have been performed by using the Adam optimiser with a learning rate of $5 \cdot 10^{-4}$. During generation, we employed MEG to find the 5 best counterfactual explanations for each molecule in test, ranked according to the multi-objective score in Section 2. Ideally, we would like to observe counterfactual molecules that are structurally similar to the original compound while leading to a substantially different prediction. Due to the stringent page constraints, in the following we report two example explanation cases (one for each dataset). Further examples and results are available in the appendix.

Molecule	Target	Prediction	Similarity	Reward
A0: Figure 2	`NoTox`	`NoTox` (0.70)	-	-
A1: Figure 2	-	`Tox` (0.90)	0.76	0.80
A2: Figure 2	-	`Tox` (0.83)	0.79	0.72
A3: Figure 2	-	`Tox` (0.80)	0.68	0.66
B0: Figure 3	-4.28	-4.01	-	-
B1: Figure 3	-	-6.11	0.29	1.14
B2: Figure 3	-	-5.93	0.31	1.11
B3: Figure 3	-	-5.07	0.28	0.66

Table 1: Summary of preliminary results. A0 and B0 refers to molecules belonging to Tox21 and ESOL, respectively. Subsequent indexes refers to the related counterfactuals explanations.

We present some quantitative result in Table 1, listing the best three counterfactual explanations collected, for both tasks. We tested two similarity metrics: cosine similarity over the neural encodings, for Tox21, and the Tanimoto, for ESOL. Qualitative results are shown in Figure 2 and Figure 3. To ease the interpretation of our results, counterfactual modifications have been highlighted in red, while blurred edges represent those edges that have been masked out by GNNExplainer predictions. In other words, GNNExplainer interpretations are the sub-graphs formed by non-blurred edges. As for the Tox21 sample, we evaluate MEG against a test molecule (i.e, A0) that has been correctly classified by the DGN as being non-toxic, outputting the counterfactuals A1-3 (i.e. molecules which the model considers toxic). We can see that the addition of a carbon atom may alter the DGN prediction, as shown by A1 and A2. In fact, while A0 is classified correctly with 70% certainty, A1-2 are predicted as toxic, with certainty of 90% and 83%, respectively. Differently, A3 breaks the left side ring and achieves the lowest neural encoding similarity score among the three, giving clues about potential substructure-awareness. Furthermore, in Figure 2 we show how counterfactuals may help to detect inconsistencies in GNNExplainer predictions. In fact, although GNNExplainer identifies the substructure CC(N)O as explanation for the original sample A0, MEG counterfactuals prioritize changes to different molecule fragments. These inconsistencies suggest that the GNNExplainer interpretation is too much targeted to the input molecule (A0) and does not generalize even for minor modifications of the input graph.

We now turn our attention to ESOL results (B0-3) shown in Table 1. B0 is an organic compound named pentachlorophenol, commonly used as a pesticide or a disinfectant, and is characterized by nearly absolute insolubility in water. While the DGN achieved good predictive performance for its aqueous solubility value, the counterfactuals underlined that the φ-model predicted solubility decreases in case the oxygen atom is removed (e.g, B2), or modified somehow (e.g, B1, B3), highlighting how it is highly relevant for the DGN prediction. As in the Tox21 sample, such relation is not adequately captured by GNNExplainer explanation for B0. It is our hope that, based on our interpretability approach, an expert of the molecular domain could be able to gain a better insight into the whether the properties and patterns captured by the predictive model are meaningful from a chemical standpoint.

4 Conclusions

We have presented MEG, a novel interpretability framework that tackles explainability in the chemical domain by generation of molecular counterfactual explanations. MEG can work with any DGN model as we only exploit input-output properties of such models. As a general comment of the preliminary results, one can note that while a local approach such as GNNExplainer may give good approximations when it comes to explaining the specific prediction, it lacks sufficient breadth to characterize the model behaviour already in a near vicinity of the sample under consideration. On the other hand, our counterfactual interpretation approach may find new samples which are likely to highlight the causes of a given model prediction, providing a better approximation to a locally interpretable model, e.g. B1-3 in Figure 2. In conclusion, apart for its value in generating explanations that are well understood by a domain expert, MEG proposes itself both as a sanity checker for other local model explainers, as well as a sampling method to strengthen the coverage and validity of local interpretable explanations, such as in the original LIME method for vectorial data [21].

References

[1] Alessio Micheli, Alessandro Sperduti, and Antonina Starita. An introduction to recursive neural networks and kernel methods for cheminformatics. *Current Pharmaceutical Design*, 13(8), 2007.

[2] Jie Zhou, Ganqu Cui, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. Graph neural networks: A review of methods and applications, 2018.

[3] Davide Bacciu, Federico Errica, Alessio Micheli, and Marco Podda. A gentle introduction to deep learning for graphs. *Neural Networks*, 129:203–221, 2020.

[4] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry. volume 70 of *Proceedings of Machine Learning Research*, pages 1263–1272, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR.

[5] Federico Errica, Marco Podda, Davide Bacciu, and Alessio Micheli. A fair comparison of graph neural networks for graph classification. In *International Conference on Learning Representations*, 2020.

[6] Davide Bacciu, Federico Errica, and Alessio Micheli. Probabilistic learning on graphs via contextual architectures. *Journal of Machine Learning Research*, 21(134):1–39, 2020.

[7] Rex Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. Gnnexplainer: Generating explanations for graph neural networks, 2019.

[8] Yue Zhang, David Defazio, and Arti Ramesh. Relex: A model-agnostic relational model explainer, 2020.

[9] Qiang Huang, Makoto Yamada, Yuan Tian, Dinesh Singh, Dawei Yin, and Yi Chang. Graphlime: Local interpretable model explanations for graph neural networks, 2020.

[10] Richard S Sutton and Andrew G Barto. Reinforcement learning: an introduction cambridge. *MA: MIT Press.[Google Scholar]*, 1998.

[11] C. Liu, X. Xu, and D. Hu. Multiobjective reinforcement learning: A comprehensive overview. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 45(3):385–398, 2015.

[12] Benjamin Sanchez-Lengeling, Carlos Outeiral, Gabriel L. Guimaraes, and Alan Aspuru-Guzik. Optimizing distributions over molecular space. an objective-reinforced generative adversarial network for inverse-design chemistry (organic), Aug 2017.

[13] Mariya Popova, Olexandr Isayev, and Alexander Tropsha. Deep reinforcement learning for de novo drug design. *Science Advances*, 4(7):eaap7885, Jul 2018.

[14] Zhenpeng Zhou, Steven Kearnes, Li Li, Richard N. Zare, and Patrick Riley. Optimization of molecules via deep reinforcement learning, 2018.

[15] Hado van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning, 2015.

[16] David Rogers and Mathew Hahn. Extended-connectivity fingerprints. *Journal of Chemical Information and Modeling*, 50(5):742–754, 2010. PMID: 20426451.

[17] Kristian Kersting, Nils M. Kriege, Christopher Morris, Petra Mutzel, and Marion Neumann. Benchmark data sets for graph kernels, 2016.

[18] Zhenqin Wu, Bharath Ramsundar, Evan N. Feinberg, Joseph Gomes, Caleb Geniesse, Aneesh S. Pappu, Karl Leswing, and Vijay Pande. Moleculenet: A benchmark for molecular machine learning, 2017.

[19] Greg Landrum et al. Rdkit: Open-source cheminformatics, 2006.

[20] Christopher Morris, Martin Ritzert, Matthias Fey, William L. Hamilton, Jan Eric Lenssen, Gaurav Rattan, and Martin Grohe. Weisfeiler and leman go neural: Higher-order graph neural networks, 2018.

[21] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should I trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 1135–1144, 2016.

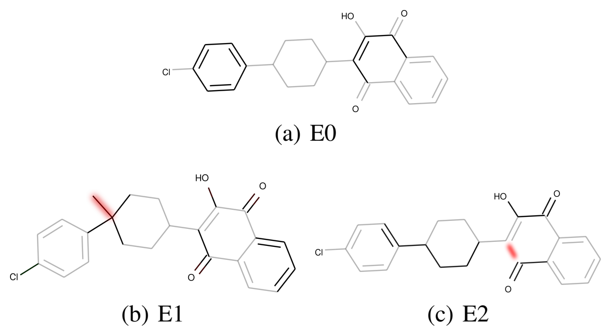
A Additional Results

Molecule	Target	Prediction	Similarity	Reward
C0: Figure 5	-4.755	-4.5195	-	1.57
C1: Figure 5	-	-2.6488	0.39	1.33
C2: Figure 5	-	-3.0170	0.65	1.12
D0: Figure 6	NoTox	NoTox (0.71)	-	-
D1: Figure 6	-	Tox (0.86)	0.90	0.78
D2: Figure 6	-	Tox (0.80)	0.91	0.73
E0: Figure 4	Tox	Tox (0.78)	-	-
E1: Figure 4	-	NoTox (0.94)	0.69	0.86
E2: Figure 4	-	NoTox (0.84)	0.89	0.73

Table 2: Summary of other preliminary results.

Table 1 provides experimental results for three compounds, one of which belongs to ESOL (C0-2) and two to Tox21 (D0-2, E0-2). Visual feedback is shown in Figure 4-5-6. As before, sharpness of graph edges indicates GNNExplainer explanations, while counterfactual modifications have been colored in red.

We seek for counterfactuals for an ESOL test compound, whose predicted solubility is close to the actual target. In this case, the atom of sulphur seems to have a negative impact on the predicted aqueous solubility. In this regard, C1 increases the compund solubility by removing, indeed, the atom of sulphur. In nature, a molecule of sulphur (i.e, S8 in SMILES encoding) is known to be insoluble. Such an analysis can drop preliminary hints about how the trained model may have learned such characteristics. Similarly to C1, C2 added an atom of oxygen causing the predicted water solubility to increase.



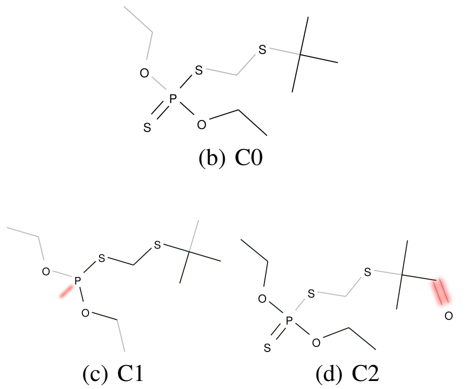
(a) E0

(b) E1

(c) E2

Figure 4: E1 modifies the cyclohexane ring, which was not considered important in the explanation provided by GNNExplainer for the original molecule E0. E2 breaks the bond highlighted in red.

Another significant examples comprises D0-2. In fact, D0 is correctly classified as a non-toxic compound. However, a simple addition of nitrogen makes the prediction change completely, resulting in classifying D1 and D2 as toxic with certainty of 86% and 80% respectively. Furthermore, sanity checks on GNNExplainer explanation for D0 emphasize that D2 updates a blurred explanation fragment (i.e, the atom of carbon attached to the atom of nitrogen nor its incident bonds have been considered important in D0). More interestingly, E0-2 present a potentially dangerous situation. In detail, starting from a toxic compound (E0), E1 achieves to be recognized as non-toxic by simply adding an atom of carbon, and so does E2 by breaking one of the rings, as shown in Figure 4. In this case, the usefulness of our counterfactuals can be exploited to the fullest, highlighting such difficulties of the model under consideration which is crucial in real-world applications.

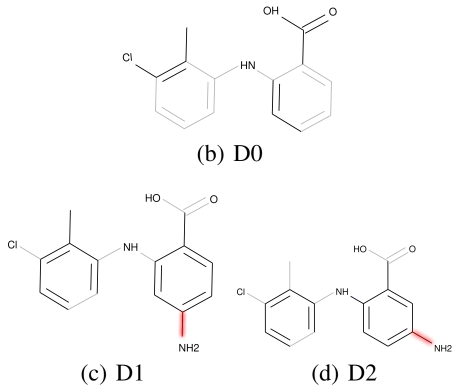


(b) C0

(c) C1

(d) C2

Figure 5: ESOL. C1 removes the atom of sulphur. C2, instead, adds a new atom of oxygen and connect it to the molecule through a double bond.



(b) D0

(c) D1

(d) D2

Figure 6: Tox21. The agent adds atoms of nitrogen to the rightmost ring.