Recent Advances in Underwater Basket Weaving Under the Extreme Pressure of the Mariana Trench

André Lauren Benjamin[1], Calvin Cordozar Broadus Jr.[2,3] (✉), and Antwan André Patton[1][0000−1111−2222−3333]

[1] Fictional Southern University, Savannah GA 31404, USA
{a.l.benjamin,a.a.patton}@fsu.fake

[2] Fictional West Coast University, Long Beach CA 90840, USA ccb@fwcu.fake

[3] Secondary European Affiliation, Tiergartenstr. 17, 69121 Heidelberg, Germany
lncs@springer.com

Abstract. This document provides a basic paper template and submission guidelines. Abstracts must be a single paragraph, ideally between 4–6 sentences long. Gross violations will trigger corrections at the camera-ready phase.

Keywords: First keyword · Second keyword · Another keyword.

1 Related Literature

1.1 Background on Counterfactual Explanations

[11, 5, 2]

1.2 Learning Representations

For example, joint-energy models

1.3 Generalization and Robustness

[9] generate counterfactual images for MNIST and ImageNet through independent mechanisms (IM): each IM learns class-conditional input distributions over a specific lower-dimensional, semantically meaningful factor, such as *texture*, *shape* and *background*. The demonstrate that using these generated counterfactuals during classifier training improves model robustness. Similarly, [1] argue that counterfactuals represent potentially useful training data in machine learning, especially in supervised settings where inputs may be reasonably mapped to multiple outputs. They, too, demonstrate the augmenting the training data of image classifiers can improve generalization.

[10] propose an approach using counterfactuals in training that does not rely on data augmentation: they argue that counterfactual pairs typically already

exist in training datasets. Specifically, their approach relies on, firstly, identifying similar input samples with different annotations and, secondly, ensuring that the gradient of the classifier aligns with the vector between pairs of counterfactual inputs using the cosine distance as a loss function (referred to as *gradient supervision*) (***this might be useful for our task as well***). In the natural language processing (NLP) domain, counterfactuals have similarly been used to improve models through data augmentation: [12], propose POLYJUICE, a general-purpose counterfactual generator for language models. They demonstrate empirically that augmenting training data through POLYJUICE counterfactuals improves robustness in a number of NLP tasks.

1.4 Link to Adversarial Training

[3] propose two definitional differences between Adversarial Examples (AE) and Counterfactual Explanations (CE): firstly, and more importantly according to the authors, the term AE implies missclassification, which is not the case for CE (***this might be a useful notion for use to distinguish between adversarials and explanations during training***); secondly, they argue that closeness plays a more critical role in the context of CE but confess that even counterfactuals that are not close might be relevant explanations. [7] show that CE and AE are equivalent under certain conditions and derive upper bounds on the distances between them.

1.5 Closely Related

[4] are the first to propose end-to-end training pipeline that includes counterfactual explanations as part of the training prodeduce. In particular, they propose a specific network architecture that includes a predictor and CE generator network (***akin a GAN?***), where the parameters of the CE generator network are learnable. Counterfactuals are generated during each training iteration and fed back to the predictor network (***here we are aligned***). In contrast, we impose no restrictions on the neural network architecture at all. (***to ensure the one-hot encoding of categorical features is maintained, they simple use softmax (might be interesting for CE.jl)***) Interestingly, the authors find that their approach is sensitive to the choice of the loss function: only MSE seems to lead to good performance. They also demonstrate theoretically, that the objective function is difficult to optimize due to divergent gradients and suffers from poor adversarial robustness. (***because partial gradients with respect to the classification loss component and the counterfactual validity component point in opposite directions***). To mitigate these issues, the authors use block-wise gradient descent: they first update with respect to classification loss and then use a second update with respect to the other loss components (***this might be useful for our task as well***). [8] propose a way to train models that are guaranteed to provide recourse for individuals with high probability. The approach builds on adversarial training (***here we are aligned***), where in this context adversarial examples are actively encouraged to exist, but

only target attacks with respect to the positive class. The proposed method allows for imposing a set of actionable recourse ex-ante: for example, users can impose mutability constraints for features (***here we are aligned***). (***To solve their objective function more efficiently, they use a first-order Taylor approximation to approximate the recourse loss component (might be applicable in our case)***)

[6] introduce Counterfactual Adversarial Training (CAT) with intention of improving generalization and robustness of language models. Specifically, they propose to proceed as follows: firstly, identify training samples that are subject to high predictive uncertainty (entropy); secondly, generate counterfactual explanations for those samples; and, finally, finetune the model on the augmented dataset that includes the generated counterfactuals.

Acknowledgments. A bold run-in heading in small font size at the end of the paper is used for general acknowledgments, for example: This study was funded by X (grant number Y).

Disclosure of Interests. It is now necessary to declare any competing interests or to specifically state that the authors have no competing interests. Please place the statement with a bold run-in heading in small font size beneath the (optional) acknowledgments, for example: The authors have no competing interests to declare that are relevant to the content of this article. Or: Author A has received research grants from Company W. Author B has received a speaker honorarium from Company X and owns stock in Company Y. Author C is a member of committee Z.

Bibliography

[1] Abbasnejad, E., Teney, D., Parvaneh, A., Shi, J., van den Hengel, A.: Counterfactual vision and language learning. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 10041–10051 (2020). https://doi.org/10.1109/CVPR42600.2020.01006

[2] Altmeyer, P., Farmanbar, M., van Deursen, A., Liem, C.C.: Faithful model explanations through energy-constrained conformal counterfactuals. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 38, pp. 10829–10837 (2024)

[3] Freiesleben, T.: The intriguing relation between counterfactual explanations and adversarial examples. Minds and Machines **32**(1), 77–109 (2022)

[4] Guo, H., Nguyen, T.H., Yadav, A.: Counternet: End-to-end training of prediction aware counterfactual explanations. In: Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. p. 577–589. KDD '23, Association for Computing Machinery, New York, NY, USA (2023). https://doi.org/10.1145/3580305.3599290, https://doi.org/10.1145/3580305.3599290

[5] Joshi, S., Koyejo, O., Vijitbenjaronk, W., Kim, B., Ghosh, J.: Towards realistic individual recourse and actionable explanations in black-box decision making systems (2019)

[6] Luu, H.L., Inoue, N.: Counterfactual adversarial training for improving robustness of pre-trained language models. In: Proceedings of the 37th Pacific Asia Conference on Language, Information and Computation. pp. 881–888 (2023)

[7] Pawelczyk, M., Agarwal, C., Joshi, S., Upadhyay, S., Lakkaraju, H.: Exploring counterfactual explanations through the lens of adversarial examples: A theoretical and empirical analysis. In: Camps-Valls, G., Ruiz, F.J.R., Valera, I. (eds.) Proceedings of The 25th International Conference on Artificial Intelligence and Statistics. Proceedings of Machine Learning Research, vol. 151, pp. 4574–4594. PMLR (28–30 Mar 2022), https://proceedings.mlr.press/v151/pawelczyk22a.html

[8] Ross, A., Lakkaraju, H., Bastani, O.: Learning models for actionable recourse. In: Proceedings of the 35th International Conference on Neural Information Processing Systems. NIPS '21, Curran Associates Inc., Red Hook, NY, USA (2024)

[9] Sauer, A., Geiger, A.: Counterfactual generative networks (2021), https://arxiv.org/abs/2101.06046

[10] Teney, D., Abbasnedjad, E., van den Hengel, A.: Learning what makes a difference from counterfactual examples and gradient supervision. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part X 16. pp. 580–599. Springer (2020)

[11] Wachter, S., Mittelstadt, B., Russell, C.: Counterfactual explanations without opening the black box: Automated decisions and the GDPR. Harv. JL & Tech. **31**, 841 (2017). https://doi.org/10.2139/ssrn.3063289

[12] Wu, T., Ribeiro, M.T., Heer, J., Weld, D.: Polyjuice: Generating counterfactuals for explaining, evaluating, and improving models. In: Zong, C., Xia, F., Li, W., Navigli, R. (eds.) Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers). pp. 6707–6723. Association for Computational Linguistics, Online (Aug 2021). https://doi.org/10.18653/v1/2021.acl-long.523, https://aclanthology.org/2021.acl-long.523

2 Appendix

2.1 Initial Grid Search

Generator Params

Linearly Separable

Moons

Circles

Table 1. Results for Linearly Separable data by energy penalty.

Objective	**Lambda Energy (exper)**	**Generator Type**	**Value**	**Std**
full	0.5	*ECCo*	-11.7601	9.83205
full	0.5	*Generic*	-1.064e18	2.41995e18
full	**0.5**	**Omniscient**	**-2.5405**	**0.122789**
full	0.5	*REVISE*	-15.0277	12.5588
vanilla	0.5	*ECCo*	-4.39923	3.65268
vanilla	0.5	*Generic*	-4.38184	3.48393
vanilla	0.5	*Omniscient*	-5.24831	4.61237
vanilla	0.5	*REVISE*	-4.94731	4.2233
full	1.0	*ECCo*	-11.5401	11.0622
full	1.0	*Generic*	-1.70667e11	3.88205e11
full	**1.0**	**Omniscient**	**-2.58956**	**0.117255**
full	1.0	*REVISE*	-15.7258	13.2676
vanilla	1.0	*ECCo*	-4.27742	3.50817
vanilla	1.0	*Generic*	-4.44409	3.4741
vanilla	1.0	*Omniscient*	-5.11353	4.4628
vanilla	1.0	*REVISE*	-4.91409	4.24885
full	5.0	*ECCo*	-3.99166	3.12284
full	5.0	*Generic*	-4.88333e17	1.11064e18
full	**5.0**	**Omniscient**	**-2.5325**	**0.117196**
full	5.0	*REVISE*	-14.5887	12.1265
vanilla	5.0	*ECCo*	-4.39614	3.64978
vanilla	5.0	*Generic*	-4.37909	3.48341
vanilla	5.0	*Omniscient*	-5.24668	4.60676
vanilla	5.0	*REVISE*	-4.94655	4.22198
full	**10.0**	**ECCo**	**-2.30721**	**0.73475**
full	10.0	*Generic*	-1.69667e11	3.85893e11
full	10.0	*Omniscient*	-2.53433	0.116736
full	10.0	*REVISE*	-15.5346	13.0245
vanilla	10.0	*ECCo*	-4.28116	3.50992
vanilla	10.0	*Generic*	-4.4428	3.47049
vanilla	10.0	*Omniscient*	-5.11933	4.46099
vanilla	10.0	*REVISE*	-4.91285	4.24407
full	**15.0**	**ECCo**	**-2.00576**	**0.48751**
full	15.0	*Generic*	-4.91e17	1.11683e18
full	15.0	*Omniscient*	-2.52833	0.11602
full	15.0	*REVISE*	-14.3763	11.7494
vanilla	15.0	*ECCo*	-4.3957	3.65194
vanilla	15.0	*Generic*	-4.38497	3.48359
vanilla	15.0	*Omniscient*	-5.24893	4.60484
vanilla	15.0	*REVISE*	-4.94518	4.22746

Table 2. Results for Moons data by energy penalty.

Objective	Lambda Energy (exper)	Generator Type	Value	Std
full	0.5	*ECCo*	-7.08577	7.51393
full	0.5	*Generic*	-1.1064e31	2.53239e31
full	**0.5**	**Omniscient**	**-4.58057**	**4.8256**
full	0.5	*REVISE*	-1187.61	2643.72
vanilla	0.5	*ECCo*	-15.4966	17.1932
vanilla	0.5	*Generic*	-11.7071	12.8003
vanilla	0.5	*Omniscient*	-12.3897	14.1104
vanilla	0.5	*REVISE*	-11.2965	13.1122
full	1.0	*ECCo*	-6.06278	6.32519
full	1.0	*Generic*	-1.57758e33	3.59342e33
full	**1.0**	**Omniscient**	**-4.66436**	**4.88547**
full	1.0	*REVISE*	-1157.26	2585.3
vanilla	1.0	*ECCo*	-15.4915	17.2592
vanilla	1.0	*Generic*	-10.8969	11.888
vanilla	1.0	*Omniscient*	-12.6685	14.4499
vanilla	1.0	*REVISE*	-11.2874	13.1369
full	**5.0**	**ECCo**	**-2.56504**	**2.06543**
full	5.0	*Generic*	-1.16971e28	2.66145e28
full	5.0	*Omniscient*	-4.28955	4.30748
full	5.0	*REVISE*	-530.204	1163.55
vanilla	5.0	*ECCo*	-15.4763	17.1877
vanilla	5.0	*Generic*	-11.6655	12.7364
vanilla	5.0	*Omniscient*	-12.3937	14.1141
vanilla	5.0	*REVISE*	-11.2976	13.0533
full	**10.0**	**ECCo**	**-1.76439**	**0.973615**
full	10.0	*Generic*	-1.54318e33	3.51163e33
full	10.0	*Omniscient*	-4.44467	4.56008
full	10.0	*REVISE*	-1515.03	3402.96
vanilla	10.0	*ECCo*	-15.5074	17.275
vanilla	10.0	*Generic*	-10.9077	11.8867
vanilla	10.0	*Omniscient*	-12.6771	14.4225
vanilla	10.0	*REVISE*	-11.2735	13.1031
full	**15.0**	**ECCo**	**-1.36625**	**0.3652**
full	15.0	*Generic*	-5.32108e28	1.21152e29
full	15.0	*Omniscient*	-4.34376	4.38045
full	15.0	*REVISE*	-473.027	1034.8
vanilla	15.0	*ECCo*	-15.4703	17.1898
vanilla	15.0	*Generic*	-11.6941	12.7669
vanilla	15.0	*Omniscient*	-12.3895	14.0956
vanilla	15.0	*REVISE*	-11.2868	13.0587

Table 3. Results for Circles data by energy penalty.

Objective	Lambda Energy (exper)	Generator Type	Value	Std
full	**0.5**	**ECCo**	**-1.12388**	**0.216889**
full	0.5	*Generic*	-1.20782	0.352005
full	0.5	*Omniscient*	-5.09228	5.1182
full	0.5	*REVISE*	-5.97244e27	1.36572e28
vanilla	0.5	*ECCo*	-9.35338	7.30155
vanilla	0.5	*Generic*	-8.89415	6.91671
vanilla	0.5	*Omniscient*	-8.67963	6.9307
vanilla	0.5	*REVISE*	-8.52507	6.74796
full	**1.0**	**ECCo**	**-1.099**	**0.163365**
full	1.0	*Generic*	-1.49485	0.726287
full	1.0	*Omniscient*	-5.15975	5.20449
full	1.0	*REVISE*	-3.09069e26	7.22344e26
vanilla	1.0	*ECCo*	-9.33801	7.36386
vanilla	1.0	*Generic*	-8.8619	6.85196
vanilla	1.0	*Omniscient*	-8.69785	6.89941
vanilla	1.0	*REVISE*	-8.69498	6.85371
full	5.0	*ECCo*	-1.75204	0.154399
full	**5.0**	**Generic**	**-1.21285**	**0.362686**
full	5.0	*Omniscient*	-5.13516	5.16338
full	5.0	*REVISE*	-1.09598e28	2.50339e28
vanilla	5.0	*ECCo*	-9.36397	7.32382
vanilla	5.0	*Generic*	-8.88498	6.90503
vanilla	5.0	*Omniscient*	-8.70333	6.9289
vanilla	5.0	*REVISE*	-8.51631	6.72565
full	10.0	*ECCo*	-1.01708e6	2.31516e6
full	**10.0**	**Generic**	**-1.48827**	**0.701741**
full	10.0	*Omniscient*	-5.13432	5.15897
full	10.0	*REVISE*	-3.74376e26	9.08858e26
vanilla	10.0	*ECCo*	-9.31463	7.32684
vanilla	10.0	*Generic*	-8.87348	6.86388
vanilla	10.0	*Omniscient*	-8.7046	6.89274
vanilla	10.0	*REVISE*	-8.68653	6.83497
full	15.0	*ECCo*	-3.31332e13	7.53714e13
full	**15.0**	**Generic**	**-1.21817**	**0.370377**
full	15.0	*Omniscient*	-5.19548	5.23317
full	15.0	*REVISE*	-9.01467e27	2.0592e28
vanilla	15.0	*ECCo*	-9.37662	7.34277
vanilla	15.0	*Generic*	-8.86149	6.8695
vanilla	15.0	*Omniscient*	-8.69488	6.95691
vanilla	15.0	*REVISE*	-8.50583	6.72685