Explaining AI-informed Network Intrusion Detection with Counterfactuals

Gang Liu
Department of Computer Science and Engineering
University of Notre Dame
Notre Dame, Indiana, USA
gliu7@nd.edu

Meng Jiang
Department of Computer Science and Engineering
University of Notre Dame
Notre Dame, Indiana, USA
mjiang2@nd.edu

***Abstract*—Artificial intelligence (AI) methods have been widely applied for accurate network intrusion detection (NID). However, the developers and users of the NID systems could not understand the systems' correct or incorrect decisions due to the complexity and black-box nature of the AI methods. This is a two-page poster paper that presents a new demo system that offers a number of counterfactual explanations visually for any data example. The visualization results were automatically generated: users just need to provide the index of a data example and do not edit anything on the graph. In the future, we will extend the detection task from binary classification to multi-class classification.**

***Index Terms*—network security, intrusion detection, explainable artificial intelligence, demo system**

I. Introduction

Network intrusion detection (NID) systems are systems that monitor network traffic for suspicious activity and alert when such activity is identified [1]. They are deployed at strategic points within the network, where they can monitor inbound and outbound traffic to and from all the devices on the network for the purposes of identifying attacks (intrusions) that passed through the network firewall.

Network intrusion detection was first introduced to the commercial market two decades ago as SNORT and quickly became a key cybersecurity control. In its first incarnation, NID systems used rules, signatures, and behavior detection engines to analyze passing traffic and match the traffic to the library of known attacks.

Last decade the scope of network attackers has changed significantly. There has been a variety of network attack types such as malware attacks, phishing emails, malvertising, worms, web attacks (e.g., SQL injection, path traversal), scan attacks, brute force attacks, and distributed denial-of-service (DDoS) attacks. To combat them, we are witnessing a wave of deep learning and AI technologies for network and application threat detection like the NSFOCUS Threat Analysis System. However, due to the black-box nature of the learning models, both AI developers and system users have to be involved in the development, deployment, and maintenance.

We use recent advances in explainable AI for NID. We expect the answer to the explanation of a decision to be given with respect to alternatives or specific unselected outcomes: "For situation X, why was the outcome Y and not Z?" A useful technique for providing such discriminative explanations is

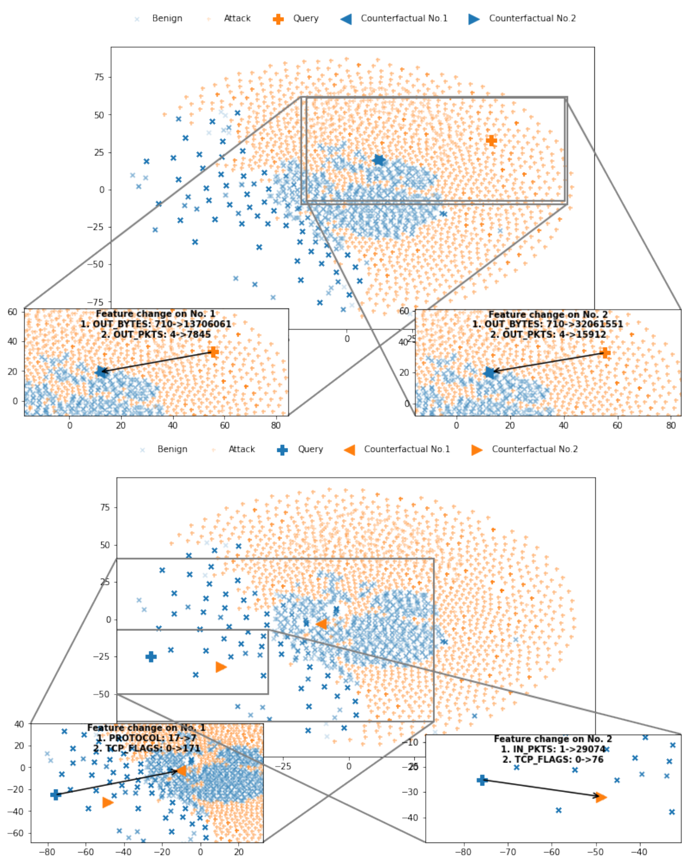

Fig. 1. Two counterfactual examples are labelled and compared with the query data example to explain the predicted label of a neural network binary classifier. The visual feature space was built by the t-SNE dimension reduction technique. The query examples are "attack" (positive) and "benign" (negative) in the top and bottom subfigures, respectively. The zoomed images shows what minimal feature changes would flip the model's binary decisions.

the counterfactual, i.e., describing what changes to the situation would have resulted in arriving at the alternative decision: "If the situation was X', then the outcome would have been Z rather than Y." [2] As intrusion detection systems achieve increasingly widespread networks, the need of the explanations to the systems' decisions is growing rapidly. Our idea is to find counterfactual examples on the continuous features: how to modify a feature's value to alter the decision of detecting the type of network intrusion and how to make sure the modification is practical. In this project we develop a system for explainable NID that creates counterfactual examples by modifying the values of a small set of features. Our system automatically generates visualizations of the explanations.

II. Counterfacutal Explanations

Suppose we have a training dataset $\{x_i, y_i\}$ where x_i has a set of attributes of the i-th example and y_i is its class label. A neural network model f_Θ was trained on a loss function l:

$$\min_{\Theta} l(f_\Theta(x_i), y_i) + \rho(\Theta), \tag{1}$$

where Θ are model parameters and ρ is a regularizer on Θ.

Counterfactual explanations can be generated to explain the decisions of the models by identifying what feature values would need to change to produce a specified output [3]. Given the classifier f_Θ and a query example x_i, we look for a counterfactual example x' as close to x_i as possible such that $f_\Theta(x')$ is equal to a new target label $y' \neq y_i$. We can find x' by holding Θ fixed and minimizing the related objective:

$$\min_{x'} \max_{\lambda} \lambda \cdot l(f_\Theta(x'), y')^2 + d(x_i, x'), \tag{2}$$

where d is a distance metric that measures how far the counterfactual x' and query x_i are. In practice, maximization over λ is done by iteratively solving for x' and increasing λ until a sufficiently close solution is found. We use Adam optimizers for first-order gradient-based optimization of the stochastic objective function. As local minima are a concern, we initialize each run with different random values for x' and select as our counterfactual the best minimizer of the Eq.(2). In practice, $\lambda = 1$ could produce satisfied counterfactual explanations in most cases. We use the random search and change at least two features each time to find a suitable x'.

III. System Design and Development

This demo system is not supposed to work independently. It serves for any AI-informed NID system which contains *a NID dataset* and *a (deep) neural network model* that was trained on the dataset to predict the class of data examples such as Benign, Reconnaissance, DDoS, DoS, Theft, and many other types of attacks. So far it has been able to process binary decisions including Benign (negative) and Attack (positive).

The system has three components. The first component is a counterfactual explanation algorithm as in the previous section. The input includes a query example in the NID dataset and the number of counterfactual explanations k. The output is the counterfactual data examples from the dataset.

The second component is the t-Distributed Stochastic Neighbor Embedding (t-SNE). It is a dimensionality reduction technique that transforms the high-dimensional NID dataset into a two-dimensional space so that we can visualize it.

The third component is a visualization tool. It plots different types of data examples into the two-dimensional space, *blue* for Benign and *orange* for Attack. In the space, it highlights the query and counterfactual examples ($+$, $\triangleleft$, $\triangleright$). Then it provides k (usually two) zoomed images to show the distance between the query and each counterfactual in the space as well as the *minimal* feature change needed from one to the other to flip the classifier's decision. This indicates why the neural model made such a prediction from the perspective of feature values.

IV. Results

We use two series of datasets [4]. One is based on NetFlow, a de-facto industry standard developed in 1996 [5]. It includes NF-BoT, NF-ToN, NF-IDS18, and NF-UQ. The other uses the CICFlowMeter tool to extract time-based features from the BoT and ToN datasets (CIC-BoT and CIC-ToN). All the datasets have more than five million examples. Their numerical attributes include timestamp, flow duration, IN_BYTES and OUT_PKTS (i.e., how many bytes/packets came in/out to the destination IP and port), and many others.

Figure 1 presents the output of two specific examples from the explainable AI-based NID system, given the NF-BoT dataset where the Benign set is much smaller than the Attack. We observe the irregular distribution of Benign (blue) and Attack (orange) examples from the top subfigure. The query example is an "Attack" and we can see that the two counterfactuals (of minimal change of original features and of the Benign label) are right at the boundary between the two classes. The features OUT_BYTES and OUT_PKTS were selected for minimal changes to generate the two counterfactuals. In the bottom subfigure, the counterfactuals select different sets of the features such as PROTOCOL and TCG_FLAGS.

V. Conclusions and Future Work

We presented a new demo system that automatically finds a number of counterfactual examples from a neural network binary classifier and visually presents their relationship with a given query example. Therefore, this system was able to provide explanations for the AI decisions which were missing in the existing AI-informed network intrusion detection systems.

In the future, we will extend the detection task from binary classification to multi-class classification. The system will be able to explain the identification of different types of attacks from the neural network models.

Acknowledgment

The authors are grateful for the support from ONR 311 Expeditionary Cyber Program N00014-22-1-2507 on this project.

References

[1] Chou, D. and Jiang, M., 2021. A survey on data-driven network intrusion detection. ACM Computing Surveys (CSUR), 54(9), pp.1-36.

[2] Liu, G., Zhang, Z., Ning, Z. and Jiang, M., 2022. On the Relationship between Counterfactual Explainer and Recommender: A Framework and Preliminary Observations. arXiv preprint arXiv:2207.04317.

[3] Barocas, S., Selbst, A.D. and Raghavan, M., 2020, January. The hidden assumptions behind counterfactual explanations and principal reasons. In Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency (pp. 80-89).

[4] Sarhan, M., Layeghy, S. and Portmann, M., 2022. Towards a standard feature set for network intrusion detection system datasets. Mobile Networks and Applications, 27(1), pp.357-370.

[5] Sarhan, M., Layeghy, S., Moustafa, N. and Portmann, M., 2021. Netflow datasets for machine learning-based network intrusion detection systems. In International Conference on Big Data Technologies and Applications, International Wireless Internet Conference (pp. 117-135). Springer.