2ND WORKSHOP
"MACHINE LEARNING & NETWORKING" (MaLeNe)
PROCEEDINGS

SEPTEMBER 4, 2023

CO-LOCATED WITH
THE 5TH INTERNATIONAL CONFERENCE ON NETWORKED SYSTEMS (NETSYS 2023)
POTSDAM, GERMANY

Demystifying User-based Active Learning for Network Monitoring Tasks

Katharina Dietz*, Nikolas Wehner*, Pedro Casas†, Tobias Hoßfeld*, Michael Seufert*
*University of Würzburg, Germany, †AIT Austrian Institute of Technology, Vienna
*{katharina.dietz, nikolas.wehner, tobias.hossfeld, michael.seufert}@uni-wuerzburg.de, †pedro.casas@ait.ac.at

I. Introduction and Problem Statement

In the past decade, Artificial Intelligence (AI), especially Machine Learning (ML), has enjoyed increasing popularity and has been widely applied in network monitoring and management. However, when it comes to sensitive topics such as network security, practical adoption has been poor, e. g., for anomaly and intrusion detection. Academic research on security-related topics lacks a holistic view [1], focusing on either human or technical aspects, while neglecting the interconnection of both [1]. Incorporating standard AI/ML approaches can further widen this gap and create barriers [2], as they take away the decision making from the experts/admins without giving any information about the confidence or severity of their decision, and provide no means to give feedback to the model, ultimately reducing the trust of the experts/admins. Thus, having an admin in the loop can be beneficial not only for the overall performance of the model by incorporating expert knowledge, but also increase its trustworthiness.

Originally, Active Learning (AL) aims to increase the performance of an ML model by manually labeling as few samples as possible via queries to an *oracle* [3], e. g., a (human) expert or a more powerful model, and add these few select samples to the (re)training data. A complementary technique to AL is self-training [4], which consists of a base classifier, which classifies the unlabeled data, but only adds the data to the (re)training on which it was the most confident on. While AL aims for the most informative data, self-training targets the most confident data to add to the model. Combining both techniques can yield great benefits [4], i. e., utilizing a base classifier to make automated decisions for obvious choices, while relaying inconfident decisions to the oracle, thus incorporating a human in the AI/ML loop.

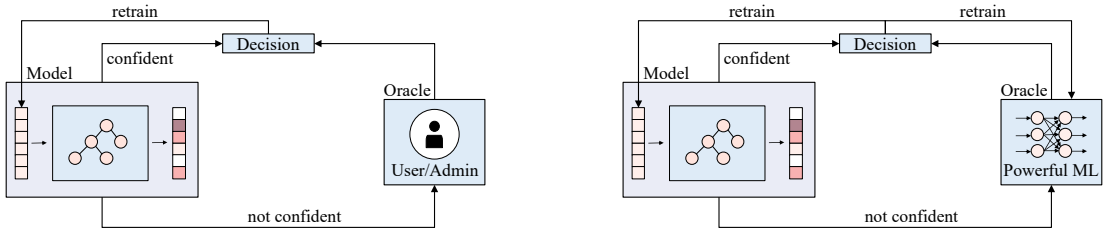
In the context of communication networks, these approaches help enabling the real-world deployment of AI/ML solutions by giving the users decisive power of critical administrative decisions, while enriching the monitored data with expertise. For example, when unseen devices, applications, or even zero-day attacks start appearing in the network (i. e., equating to potential labels the model was not trained on), the model may not be very confident in its decision and thus relay it to the expert admin for further inspection and relabeling. Ultimately, this also helps keeping the model up-to-date and adapt to network changes over time. In the following, we propose our general methodology, discuss specific use cases and related research fields, which can be incorporated in the future.

II. Background and Methodology

Active Learning. According to [3], AL can be divided into three different approaches, namely pool-based AL, stream-based selective sampling, and membership query synthesis. The pool-based approach ranks all datapoints regarding predefined criteria and relays the top ranked points to the oracle, while the latter two make an independent judgement for each sample [3]. Additionally, a membership query generates new samples on its own. The former two approaches make the most sense in context of our use case. Here, it is more of a question if we want to set a specific threshold (e. g., confidence of a decision) for every element to be relayed, which can potentially result in many requests/false alarms to the admin, or if we want to set a fixed number of queries beforehand and potentially miss out on important events. Thus, all approaches require a careful configuration of their parameters.

Querying Strategies. Regardless of the chosen AL approach, there is also a multitude of querying strategies, i. e., how to actually choose which elements to relay. In [3], the main querying strategies have been identified as diversity-based approaches, approaches based on the expected model change, and uncertainty-based approaches. The latter is a natural fit due to the self-training aspect of our use case, while the other two focus more on picking the most informative datapoints, which do not necessarily need any admin supervision.

ML Confidence/Uncertainty. To actually evaluate the confidence or uncertainty of a decision, we need predicted probabilities for all possible classes instead of just the predicted label. Luckily, many traditional ML but also Deep Learning (DL) algorithms are capable of doing so, including Random Forests (or basically any bagging/boosting algorithms due to simple majority voting), Decision Trees, Neural Networks (via softmax layers), and many more. Given these class probabilities, we may now formulate our own definitions of confidence/uncertainty. For example, we may simply set a threshold of how big the probability for the most probable class is (e. g., relay all classified samples that have no class probabilities higher than 0.8), calculate the entropy of the class probabilities (e. g., maximum entropy means that all probabilities are distributed equally and thus there is no clear winner), or calculate the distance to a uniform distribution (e. g., low Kolmogorov-Smirnov or Wasserstein distances indicate uniformly distributed probabilities thus inconfidence). How to configure these parameters and which strategy to choose is a core question.



(a) Use case 1: Admin as oracle

(b) Use case 2: Powerful ML as oracle

Fig. 1: Potential use cases for AL in network monitoring.

III. Potential Use Cases

Admin as Oracle. Figure 1a illustrates the idea behind the previously described use case. Our approach to test the impact of AL in networking may look as follows. We may utilize any existing dataset concerning network monitoring, e. g., application classification, device fingerprinting, or intrusion detection, and train a suitable ML model with them. Then we just simply utilize the label probabilities to decide if we relay the decision to an admin. As access to experts is limited, we opt for a parameterized, simulative approach, i. e., we have virtual admins that exhibit different properties with respect to accuracy and time consumption, e. g., make correct decisions in short time or take a long time and have a high probability for erroneous decisions, or any value in-between. This way we can not only evaluate if the performance improvement is significant, but also if the time consumption is even worth it. In other words, we want to find a suitable threshold via parameter studies for the uncertainty/confidence, so that the model improves, while the admin is not overburdened. We can also analyse the impact of unseen traffic (e. g., new devices/apps or zero-day attacks), by excluding one class from the training data. We expect the ML models to be inconfident on unseen labels, thus relaying these datapoints to the admin.

In-network ML. The second use case is seen in Figure 1b. The core idea is, that in-network ML (e. g., via P4 switches) can extract features and classify at line rate, which is desirable for, e. g., real-time intrusion detection/prevention. Though, models are limited in their complexity as well as features to extract due to limited operations (in regard to their type as well as their amount), potentially resulting in lower accuracy. We can leverage the previous methodology by relaying all inconfident decisions from the switches to the oracle. In this case, this may be a powerful ML server with more complex models, aiming for high accuracy. In other words, for clear decisions we can provide low/no latencies, while for harder decisions we utilize some more time and resources, thus proposing a hybrid approach. Even though we now induce a small latency overhead for the sake of higher accuracy, this still costs a lot less time than introducing a human expert into the loop, thus combining the best of both worlds. As we now have two models we deploy in the network, we also need to update and retrain both of them, making this use case more complex from a technical point of view. Especially since P4 switches cannot be updated during runtime, this may be an interesting aspect for future research with regards to downtimes etc.

IV. Related Research and Future Work

Visualization. In [5] the authors also opt for a simulation-based approach for user-based active learning in the context of image classification. Instead of just simulating "good" or "bad" experts, they develop a visual approach via dimensionality reduction and base their metrics on these visuals. This may be another methodology to implement for our use case as well, as in the end, if we want to apply this in practise, the admin/user needs some help with their decision making.

User Studies. In conjunction with visualization, we can also conduct user studies instead of having virtual admins to make our analyses more impactful and to obtain more realistic values for our admin competence and time needed to complete the tasks, shown to be fruitful in security-related topics [6]. Though, as we potentially need network experts/admins to do so, this may be a challenging task. Alternatively, we can try to simplify the admin tasks [7], e. g., by giving non-experts a small handbook/guide and make the decision tasks less complex and more well-defined. However, regardless of the users participating in potential studies, we need to design an interface first, i. e., develop a fitting visualization.

Outlier Detection. Lastly, complementary to ML uncertainty/confidence, we can also correlate outlier detection to our research. In other words, we expect there to be some kind of relationship between both approaches, as both possibly indicate that a data sample like this has not been seen before. Thus, this may add another querying strategy for our use cases.

Acknowledgment

This work was partly funded by Deutsche Forschungsgemeinschaft (DFG) in project Usernet (SE 3163/3-1, project nr. 500105691). The authors alone are responsible for the content.

References

[1] M. Vielberth, F. Böhm, I. Fichtinger, and G. Pernul, "Security operations center: A systematic study and open challenges," *IEEE Access*, vol. 8, pp. 227 756–227 779, 2020.

[2] D. Han, Z. Wang, W. Chen, Y. Zhong, S. Wang, H. Zhang, J. Yang, X. Shi, and X. Yin, "Deepaid: interpreting and improving deep learning-based anomaly detection in security applications," in *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, 2021, pp. 3197–3217.

[3] P. Ren, Y. Xiao, X. Chang, P.-Y. Huang, Z. Li, B. B. Gupta, X. Chen, and X. Wang, "A survey of deep active learning," *ACM computing surveys (CSUR)*, vol. 54, no. 9, pp. 1–40, 2021.

[4] M. S. Hajmohammadi, R. Ibrahim, A. Selamat, and H. Fujita, "Combination of active learning and self-training for cross-lingual sentiment classification with density analysis of unlabelled samples," *Information sciences*, vol. 317, pp. 67–77, 2015.

[5] C. Seifert and M. Granitzer, "User-based active learning," in *2010 IEEE International Conference on Data Mining Workshops*. IEEE, 2010, pp. 418–425.

[6] A. Beaugnon, P. Chifflier, and F. Bach, "End-to-end active learning for computer security experts," in *KDD Workshop on Interactive Data Exploration and Analytics (IDEA)*, 2018.

[7] H. Trittenbach, A. Englhardt, and K. Böhm, "Validating one-class active learning with user studies–a prototype and open challenges," in *ECML PKDD Workshop*, 2019, p. 17.