SAND2023-11084J

Test and Evaluation of Systems with Embedded Machine Learning Components

Michael R. Smith, Carianne Martinez, Joe Ingram, Mark DeBonis, Christopher Cuellar, Deepu Jose

Sandia National Laboratories, Albuquerque, New Mexico
{msmith4, cmarti5, jbingra, mjdebon, crcuell, djose}@sandia.gov

Abstract

As Machine Learning (ML) continues to advance, it is being integrated into more systems. Often, the ML component represents a significant portion of the system that reduces the burden on the end user or significantly improves task performance. However, the ML component represents an unknown complex phenomenon that is learned from collected data without the need to be explicitly programmed. Despite the improvement in task performance, the models are often black boxes. Evaluating the credibility and the vulnerabilities of ML models poses a gap in current test and evaluation practice. For high consequence applications, the lack of testing and evaluation procedures represents a significant source of uncertainty and risk. To help reduce that risk, we present considerations to evaluate systems embedded with an ML component within a red-teaming inspired methodology. We focus on (1) cyber vulnerabilities to an ML model, (2) evaluating performance gaps, and (3) adversarial ML vulnerabilities.

Introduction

Machine learning (ML) is a paradigm in which the actions taken by a computer are learned rather than explicitly programmed. This is a tremendous advance, especially in complex applications. ML is now an everyday experience ranging from innocuous applications such as recommending what movie to watch next to high consequence domains such as medical (Bradley, Korfiatis, Akkus, & Kline, 2017), critical infrastructure (Laplante, Milojicic, Serebryakov, & Bennett, 2020), and warfare (Tangredi & Galdorisi, 2021) applications. Over 160 billion US dollars was invested in ML applications in 2021 and that investment is continuing to grow exponentially (Zhang, et al., 2022). As the integration of ML is more prevalent, there have also been some disastrous results including deaths from mistakes made by self-driving vehicles (McFarland, 2022), racist chat bots (Schwartz, 2019) and image classifiers (Guynn, 2015), as well as targeted adversarial attacks against ML models (Chakraborty, Alam, Dey, Chattopadhyay, & Mukhopadhyay, 2018). Thus, establishing a process and tools to evaluate such systems is critically important. Our goal in this paper is to define an initial process for evaluating systems that have an ML component central to its operation.

As opposed to the academic evaluation of ML models, we present a system-level evaluation rather than the ML model in isolation. We outline three axes along which to evaluate an ML component:

1) Evaluating the performance of the ML component to ensure that the model functions as intended and is developed based on best practices developed by the ML community. This process entails more than simply evaluating the learned model. As the model operates on data

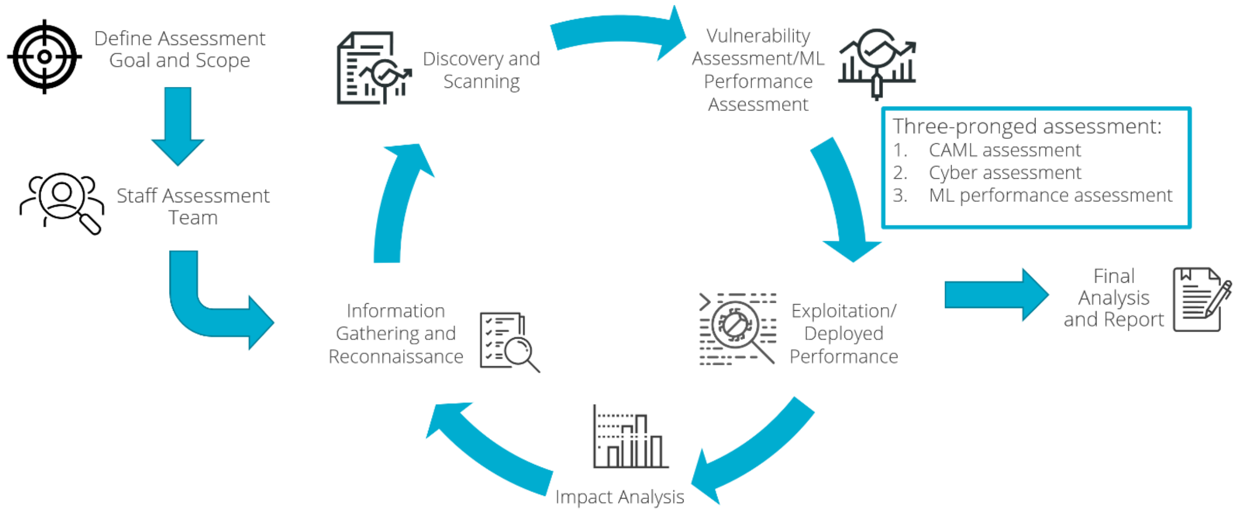

Figure 1: Overview of the methodology for assessing systems with an ML component. The key component is a three-pronged assessment: (1) an assessment of the ML component(s) to attacks that cause a failure or leak of unintended information; (2) an assessment of the infrastructure supporting the ML component(s) and how it may affect the performance of the ML component; and (3) an assessment of the performance of the ML component in contested environments.

used for training as well as perceived by the system, peripheral functions such as feature engineering and the data pipeline need to be included.

2) ML components necessitate supporting infrastructure in deployed systems. The support infrastructure may introduce additional vulnerabilities that are overlooked in traditional test and evaluation processes. Further, the ML component may be subverted by modifying key configuration files or data pipeline components.
3) ML models introduce possible vulnerabilities to adversarial attacks. The adversarial machine learning (AML) attacks could be designed to evade detection by the model, poison the model, steal the model or training data, or misuse the model to act inappropriately.

It is assumed that there will be an accompanying cyber assessment which is outside the scope of this paper. Reporting deliverables, actions, and planning should be followed according to the established guidance. This paper focuses on elements specific to the ML component that would accompany a cyber vulnerability assessment. The final product of the assessment methodology is a document outlining the risks and possible remediations related to an ML system. The document is designed to record the expected performance and uncertainties of the ML component(s), cybersecurity vulnerabilities, ML vulnerabilities, data leakage through the ML component, and the impact of these vulnerabilities on the system and application.

A typical red teaming methodology (summarized in Figure 1) comprises the following eight steps. We augment these steps for considering systems with an ML component.

- **Define Assessment Goal and Scope:** The primary objective of this step is to align the assessment to the application goal of the system and specifically outline how the ML component affects that goal. All information that is available about the system and the ML component should be provided to the assessment team, access to the system, and rules of engagement established. The scope of the assessment establishes the rules of engagement and defines the threat model(s).

- **Staff Assessment Team:** The Integrated Assessment Team (IAT) will be charged with planning and executing the assessment. This team will need to include ML and AML experts who understand the domain, system, and the ML component that is being assessed.
- **Information Gathering and Reconnaissance:** This step seeks to gather as much information as possible that documents the application objectives, the system, and the ML component. Ideally, system developers are available for interview and provide additional information as needed to cover undocumented aspects of the system. Open-source Intelligence (OSINT) should be consulted—particularly relating to the ML component and techniques to subvert it.
- **Discovery and Scanning:** The objective of discovery and scanning is to discover where in the system the ML component can be affected. Access points generally lie in a data pipeline for operating on data and outputting results. How the ML component is executed and configured is identified in this step. In cases where the ML component is not fully disclosed, discovering the ML component is also undertaken.
- **Vulnerability Assessment/ML Performance Assessment:** Given the identified touch points in data pipeline(s), the system configuration to execute the ML component, and ML model details, vulnerabilities are identified, and plans are made to exploit them. Vulnerabilities are identified relating to the infrastructure supporting the ML model as well as the model itself. Additionally, performance, reliability, and robustness of the ML model is assessed, and plans are made to test it.
- **Exploitation/Deployed Performance:** Exploitation of the identified vulnerabilities and testing of ML performance, reliability, and robustness issues are executed. Impact to the ML model and downstream system effects are recorded.
- **Impact Analysis:** The goal of the assessment is to assess the impact on the domain and how the exploitation of the identified vulnerabilities affects the objective of the application. Once the vulnerabilities have been exploited and the assumptions made by the ML component have been tested with edge cases, the impact of such adversarial attacks or data that breaks assumptions on the system are evaluated. The impact analysis consolidates the findings from the exploitation and deployed performance assessment with respect to the application impact. Possible mitigations are also provided and analyzed. The objective of this phase is to quantify the severity of any vulnerability or unexpected behavior from the ML component on the overall system. In this step, it may be recognized that there are additional gaps that need to be addressed. If so, any phase of the assessment can be repeated.
- **Final Analysis and Report:** All findings and documentation of the assessment steps are provided in a final report. It should include recommendations on how to mitigate the identified vulnerabilities and how to improve the performance of the ML component and the overall system.

This paper establishes an initial set of considerations for ML components. We first provide a high-level description of developing an ML model and then discuss the vulnerabilities associated with ML.

The Machine Learning Lifecycle

A broad overview of the ML model development and deployment cycle is provided to give context in the assessment and to motivate the need for access to data and additional information in an assessment. Figure 2 shows the steps that are often involved in developing and deploying an ML model. There are two primary components that are integrated into an ML component: a data processing module and a trained ML model. As can be seen in Figure 2, several steps and design decisions are involved which are

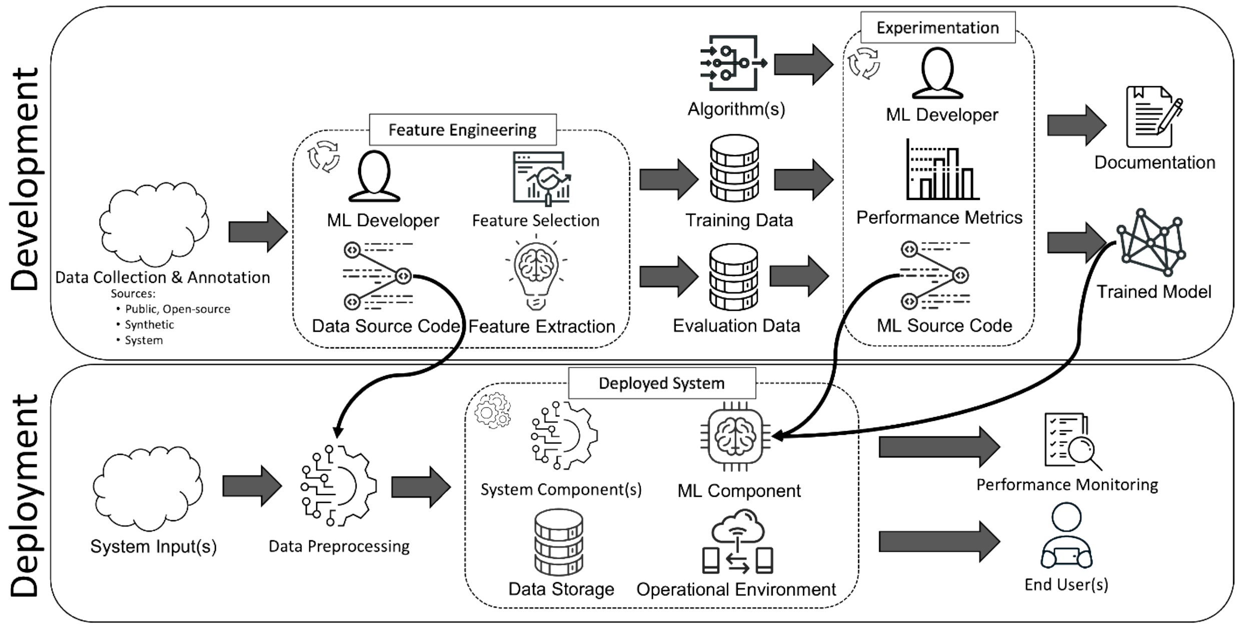

Figure 2: Process of the ML life-cycle for developing a data processing component and an ML component. While documentation and the deployed system are typically provided for an assessment, significantly more development steps are involved in developing the ML component. Ideally, an assessment would have access to intermediate steps and design decision processes.

difficult to derive from access to only the deployed system. Ideally, an assessment of an ML model would begin prior to its deployment in a final system.

The ML development life cycle is a composition of three broad phases:

1. **Data Collection and Annotation:** A key phase in ML is collecting and labelling the data. It is important that the data is representative of the task; the failure of training data to capture the statistical distribution of data in the deployment environment has proven to be a key limitation of ML (Yampolskiy, 2019). As such, several public open-source datasets are available and synthetic data generation methods are employed. There are several implications in vulnerabilities and the performance of ML that will be discussed below related to the training and evaluation data. Open-source data sets represent a possible vulnerability.
2. **Feature Engineering:** Once the raw data is collected, it often needs to be processed to make it suitable for an ML model. Common data processing techniques include filtering noise, normalizing to a standard range of values, or otherwise transforming the data to be suitable for the ML model. Different ML models have different requirements. For example, deep neural networks can operate on raw images. Other models, such as a support vector machine, may need to have features extracted from the image to operate on. Feature engineering is often an iterative process with experimentation used to discover the best representation of the data. The end product is a training dataset and often an associated evaluation dataset. For assessments, knowing the design decisions for the feature engineering are beneficial to understand what is considered important to the system and what is thrown away.
   Experimentation: The experimentation step involves an ML developer tuning the ML algorithm to optimize a performance metric on the training and evaluation datasets. This can be quite complex. In the case of deep neural networks, experimentation can involve determining the architecture, activation functions, learning rate, number of epochs, etc. The end product is a

trained model. In many cases, the internals of the learned model and training data are not exposed, assessing risks is difficult if no further information is provided.

After these steps are completed, the data processing and ML components are deployed. In some systems, the development and deployment stages are integrated such that the ML component is continually updated as additional data is received. In some cases, a human in the loop annotates data, providing feedback to the system.

The ML life cycle illustrates the chain of decisions that goes into a final model and the amount of information that would be beneficial for an assessment integrating the development cycle. Significant improvement in the assessment quality can be achieved with access to the feature engineering and experimentation components, the ML algorithm, training data, and evaluation data.

Vulnerability Assessment/ML performance Assessment

The inclusion of an ML model introduces additional possible vulnerabilities into a system. This section focuses on assessing (1) the infrastructure supporting the ML, (2) adversarial attacks against the ML component, and (3) performance of the ML component.

Cyber Attacks and Vulnerability Assessment on the ML Infrastructure

The cyber vulnerability assessment focuses on the infrastructure supporting the ML component, specifically focusing on the data access, storage, data processing, and associated configuration files that were discovered when scanning the system. Understanding how a data pipeline is generally designed and the chain of custody of data through which it flows helps define the methodology and types of attacks on the ML ecosystem. In ML, data represents a key component driving the quality of an ML model.

Generally, there exists some form of data generation or data capture from a sensor or set of sensors that provide information possibly including results from other subsystems. The data is processed eventually in preparation for the ML model. This step can happen at the same time as the algorithmic processing but does not have to. The intermediate results may be stored or can be directly transferred into the ML algorithm. The results from the ML model are often directed to storage for persistence and any other follow-on algorithmic handling of information. Eventually, these results are displayed such that strategic decisions can be made and information gleaned, or some action is taken. Each of these data flows represent interfaces that can be tested for weakness and net effect on the ML output, not all of these are unique to machine learning or ML systems; however, there should be a presence of them in many deployments.

Using the previously discovered components, the assessment team checks for and documents any known vulnerabilities. Noting which libraries are loaded may provide information about the existence and implementation of the ML component, for example, knowing if PyTorch or TensorFlow is used. Some attacks related to libraries could be if the libraries themselves are known to contain vulnerabilities, such as not maintaining and updating the operating environment of the ML component or even checking dependency chains of the libraries if they contain vulnerabilities that could be accessible. Other potential vectors of attack on a ML component could include actions such as modifying a configuration file or overwriting data in the database.

Beyond cyber vulnerabilities, an ML model could be subverted by actions including:

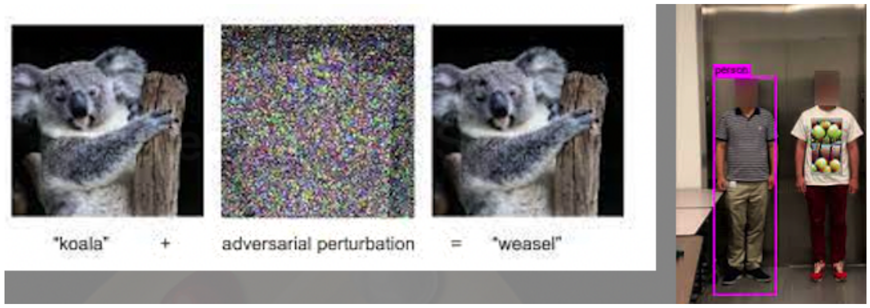

Figure 3: Left: A digital evasion attack adding imperceptible noise to an image (Goodfellow, Shiens, & Szegedy, 2014), Right: An evasion attack using a specially designed t-shirt to evade detection (Xu, et al., 2020).

1) Modifying a saved model by swapping out the entire model or changing a specific portion of the saved file. This occurs as many systems have a pre-learned model that is stored to be used rather than retraining a model each time it is used.
2) Modifying configuration settings such as thresholds of when to take an action or to retrain. These configurations can be stored in files or environment variables.
3) Directly modifying the data when the ML model is updated or when queried. Any modification poses a potential threat.

Adversarial ML Attacks

AML refers to malicious attacks on ML algorithms and the data. The information gained from the previous phases inform the types of attacks that are possible and those that are the most pertinent to the assessment. Important information includes the type of ML algorithm that is being used, the training and evaluation data, access to the ML component, the threat model, and goal of the assessment. These will dictate the type of attacks that are possible to execute. The attacks should be prioritized based on the access to the model according to the threat model and goals of the assessment/threat model. Possible attacks are outlined in the following subsections. Actual attack details will be coordinated by the AML SME on the IAT with input from domain and mission experts to best assess application impact under these attacks.

In the past decade the number of papers on this topic has grown exponentially and these attacks are both effective and alarming. These types of attacks come in several varieties: Evasion, Subversion (or Poisoning), Stealing, and Misuse.

Defense mechanisms against adversarial attacks is another consideration for an assessment. There are several proposed methods for defenses, albeit with limited success, as shown in several surveys (Tian, Cui, Liang, & Yu, 2022; Short, La Pay, & Gandhi, 2019).

Evasion

Evasion attacks involve carefully crafting inputs to an ML model to induce an error. This generally entails altering data input to avoid detection or to be misclassified. The changes made to the data are often imperceptible to humans but produce high confidence outputs from ML models that are incorrect. Figure 3 illustrates attacks by adding noise to an image (Goodfellow, Shiens, & Szegedy, 2014) and an attack that adds specially crafted noise to a shirt to avoid a person detection algorithm (Xu, et al., 2020).

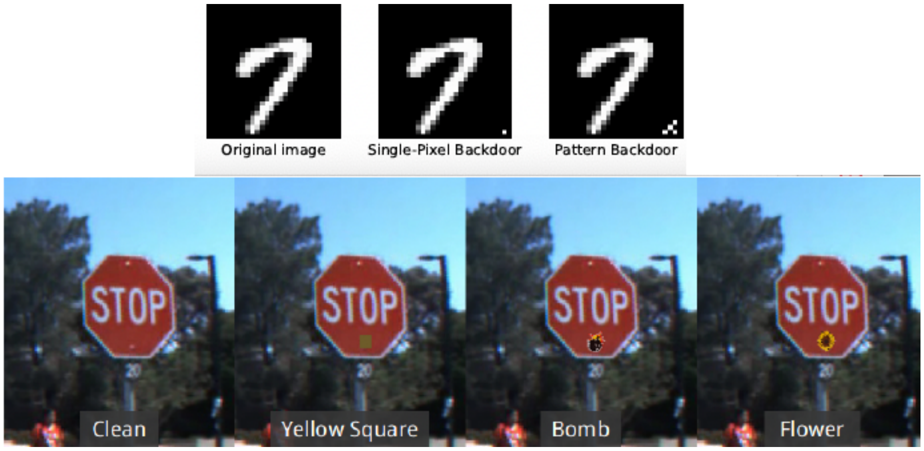

Figure 2: Top: A digital subversion (backdoor) attack to misclassify a 7 as 0, Below: A subversion (backdoor) attack that misclassifies a stop sign as a different sign (i.e. a speed limit sign) depending on the sticker that is placed on the sign (Gu, Dolan-Gavitt, & Garg, 2017).

Subversion

Subversion, or data poisoning, attacks the training data used to create the ML model. Since many data sets are obtained through open sources, one can see how such an attack is of extreme concern. This may be as simple as adjusting the labels of the training data to incorrect labels or adding a specific feature that will trigger the ML model to produce a desired output. There are several motives for such an attack. One is simply to break the ML model so that its performance is decreased. Another motive is to dictate the output of an ML model when a specified feature is present. Figure 4 illustrates subversion attacks in digital images and by altering a physical object (Gu, Dolan-Gavitt, & Garg, 2017). In each, a specified pattern is included to induce a specific output.

Stealing

This type of attack focuses on obtaining information about the ML model (**model extraction** (Atli, Szyller, Juuti, Marchal, & Asokan, 2020)) or the data that was used for training (**model inversion** (Fredrikson, et al., 2014) or **membership inference** (Shokri, Stronati, Song, & Shmatikov, 2017)). Stealing attacks are performed by careful and repeated querying of the ML model. Model extraction poses a threat by stealing the model that represents potentially large investments of intellectual property. Often the data used to train a model is sensitive and methods exists that can infer the data that was used for training an ML model. This represents a potentially critical privacy risk.

Misuse

This type of attack occurs when an attacker employs an ML model in a malicious way and not for its intended purpose. Examples include the altering of audio, imagery, or videos (**deep fakes** (Verdoliva, 2020)) for ulterior motives such as disinformation for political or financial gain.

ML Model Performance Assessment

In this stage, the IAT performs an independent assessment of the ML system and model. The intent is to ensure that the model will perform satisfactorily once deployed and to assess how it may perform when presented with novel inputs. This is a challenging portion of the assessment as it is difficult to predict how an environment may change and is an active area of research in the ML community.

Specifically, the IAT will complete as many of the following steps as possible given access to system components and resources:

1. Inspect and assess the data used for training and evaluation.
2. Compare the training and evaluation data to data sampled from the deployment domain.
3. Review the ML source code.
4. Independently train and evaluate the ML model in an environment similar to (ideally, identical to) the deployed system.
5. Review methods used to understand model behavior, such as explanations and uncertainty quantification.
6. Document findings, identify risks, and recommend mitigations.

Where possible, the actual training and evaluation datasets and deployment environment should be used for the ML assessment. However, the assessment team may use proxies when necessary.

The assessment should answer the following questions:

- Does the ML component work as intended?
- Is the component robust enough for deployed scenarios?
- Are the limits or failure modes of the model understood and documented?
- Were best practices followed during development?

The IAT should assess the ML component along the following axes: (1) representative datasets, (2) model performance, (3) deployment model performance, and (4) model trust. In the remainder of this section, we describe the assessment process, and we provide a rubric for evaluation ML model performance risk with further details in the Appendix.

Representative Datasets

As the performance of an ML model is completely dependent on the data used to train it, there are several criteria that must be met to provide high confidence in its usage. Data used for training and evaluation need to be representative of the domain that the model will be deployed against. Statistical tests to determine whether features in the training data are drawn from a distribution that is similar to that found in the deployment environment serve to assess risk associated with fielded system performance. Additionally, the manner in which the data is partitioned for training the ML model and assessing its performance must be reviewed to avoid biases with consideration of temporal, spatial, and generalization biases (Pendlebury, Pierazzi, Jordaney, Kinder, & Cavallaro, 2019; Smith, et al., 2022). Other data considerations include the size of the dataset, the coverage and appropriateness of the dataset in feature space with respect to the specific model task, and sensitivities present in the data where access control procedures must be reviewed. Data should be documented including its source and any known limitations. The Datasheets for Datasets (or similar) methodology (Gebru, et al., 2021) should be followed for concise documentation.

Model Performance

An ML model should be evaluated to ensure that it is developed and performing correctly based on several criteria.

First, the appropriateness of an ML model for the specific task should be reviewed. Given the dataset review, model complexity should also be assessed; for example, deep learning algorithms typically require large datasets and are not always appropriate for tasks with limited training data. Performance metrics should be reviewed to ensure that they capture the desired model behavior. Additionally, the process for selecting all decision thresholds within the model should be reviewed and analyzed for sensitivities, and hyperparameter tuning methods should be scrutinized to understand potential model performance variability.

Second, the model's performance should be evaluated after training. Considerations such as performance requirements, range of data values expected to be input to the model, and model stability should be reported. The IAT should ensure the model's performance in isolation is consistent with its performance as part of the full system. Special attention should be given to subgroups in the dataset that are particularly important for the model's intended use or that run the risk of being underrepresented in the dataset. Evaluation metrics should be explicitly reported for these subgroups, and mitigations should be recommended for any observed degradation of model performance within these groups.

Finally, the model should be well documented, and its performance should be reproducible. The IAT should review documentation, and ideally, methods such as model cards (Mitchell, et al., 2019) should be used for consistency. Best coding practices including version control, experiment tracking, and random number generator seeding should be verified to ensure reproducibility of model results.

Deployed Model Performance

The data and environment that an ML model operates on can vary over time and significantly differ from those that the ML model was developed on. This introduces a risk that the ML component may be irrelevant or incorrect. Over time, an ML model can become stale because historical data was used for training. The impact of an outdated model should be quantified as to how it impacts its performance over time. A model generally becomes outdated as the data is operates on changes (concept drift). Data should be reviewed periodically to detect concept drift. As concept drift is detected, methods to update ML models appropriately should be identified and scheduled. Additionally, independent data sets collected from the actual deployed environment should be used if available.

Model Trust

Recent work in the ML community has shown that models can be wrong but extremely confident in their predictions (Nguyen, Yosinski, & Clune, 2015). This is exploited by adversarial attacks. There is a need to provide trust in the model beyond good performance. Open areas of research in the machine learning field include explainability, uncertainty quantification, and the development of defenses for adversarial attacks.

Explainability is the capability of ML models to provide an explanation for how decisions are made either for the model as a whole or for individual predictions (Ribeiro, Singh, & Guestrin, 2016). Explanations are a source for increasing trust in the output of the model when working with a domain expert to ensure that the model is functioning correctly.

Another facet to understanding limitations of ML models lies in uncertainty quantification (Abdar, et al., 2021). There are many sources of uncertainty in ML models including model uncertainty (uncertainty from the model errors in approximating the true function), data uncertainty (uncertainty from noise in the data due to sensor errors or inherent noise), and distributional uncertainty (uncertainty from a mismatch between training data and data that will be encountered in deployed scenarios). Quantifying the uncertainty will help to quantify the risk associated with using the model.

Final Analysis and Report

At the end of the assessment, a final report is produced summarizing all the steps taken to come to any conclusions. It should be detailed enough to reproduce the exploitation and ML model assessment. Importantly, it should be noted what was *not* able to be assessed due to a lack of a certain resource. Recall from Figure 2 that there is a large number of steps in producing a final ML component. Lack of resources can limit the efficacy of an assessment and they should be pointed to here including the risk that is introduced by not being able to use them in the assessment. Table 1 provides a high-level summary of the necessary components to produce each section of the final report. The rows of the table represent the various components of the ML lifecycle. The columns represent assessments of interest. Each cell represents the priority level associated with the need for the component in that portion of the assessment. The scores are interpreted as follows:

1. **Low:** this component is optional at this stage.
2. **Medium:** this component is a ``nice-to-have'' during this stage of the assessment, but the assessment can still be completed successfully without it.
3. **High:** the component is needed, but the assessment can still be completed through other means. As an example, the data source code may be needed to assess the performance of the ML model, but if the training and evaluation data is already provided, a performance assessment can still be performed successfully. However, additional cost is generally needed if the component is not provided.
4. **Critical:** indicates that the relevant assessment stage cannot be completed satisfactorily without that specific component.

Table 1: Summary of necessary components for an ML assessment

	SUPPORTING INFRASTRUCTURE	**ML PERFORMANCE**	CAML	**OVERALL**
	ML DEVELOPMENT STAGE			
DOCUMENTATION	Medium	Medium	Medium	Medium
ML DEVELOPER	Medium	Medium	Medium	Medium
DATA SOURCE CODE	Medium	High	High	High
TRAINING DATA	Low	Critical	Critical	Critical
EVALUATION DATA	Low	Critical	Critical	Critical
ML SOURCE CODE	Medium	Critical	High	Critical
TRAINED MODEL	Medium	High	Critical	Critical
	ML DEPLOYMENT STAGE			
SYSTEM INPUTS	High	Medium	Medium	High
DATA PROCESSING	High	High	Medium	High
DATA STORAGE	High	High	High	High
SYSTEM COMPONENT(S)	Critical	Low	Low	Critical
OPERATIONL ENVIRONMENT	Critical	Low	Low	Critical
DEPLOYED ML COMPONENT	Critical	High	High	Critical
END USER(S)	Low	Medium	Medium	Medium

Conclusion

This paper presents considerations for doing an assessment on a system with an ML component. This is a new research field in the ML community and several toolkits exist to aide in this process. It is encouraged to take advantage of the tools and techniques provided by the ML community. Our primary motivation is bringing to the T&E community the importance of assessing ML models and providing a starting point for proper assessments.

Authors

Michael R. Smith joined Sandia National Laboratories in 2015 after receiving his PhD in Computer Science from Brigham Young University for his work on data-centric meta-learning. His work at Sandia focuses on both applied and foundational machine learning research. Most of his applied research focuses on the application of statistical models to cybersecurity and assessing systems that have a machine learning component. His foundational research centers on the credibility of data-driven models including explainability, uncertainty quantification in machine learning, out-of-distribution detection, and counter-adversarial machine learning.

Cari Martinez is a Principal Computer Scientist in the Applied Machine Intelligence Department at Sandia National Laboratories. She is a technical lead for an applied deep learning research team that supports Sandia's mission across a diverse set of science and engineering disciplines. Her research focuses on improving deep learning modeling capabilities with domain knowledge, uncertainty quantification, and explainability techniques. Cari holds a BS in Honors Mathematics and Music from the University of Notre Dame, an MS in Computer Science from the University of New Mexico and is currently a PhD candidate at Arizona State University.

Joe Ingram received his B.S. in Computer Science from the University of Illinois at Springfield in 2007 and his M.S. in Computer Science from Southern Illinois University in 2009. He is a distinguished technical staff member within the Applied Information Sciences group at Sandia National Laboratories. His expertise lies in the application of machine learning and the development of end-to-end data analysis systems. He has been a researcher at Sandia for over 13 years, leading numerous research projects as principal investigator.

Mark DeBonis received his PhD in Mathematics in 1991 from University of California, Irvine, USA. He spent some time working for the US Department of Energy and Department of Defense as an applied mathematician on applications of machine learning. Formally an Associate Professor at Manhattan College in New York City, he presently works for Sandia National Laboratory as an applied mathematician. His research interests include machine learning, statistics, and computational abstract algebra.

Christopher Cuellar received his M.S. in Computer Science from the University of Texas at El Paso in 2011. He has been a researcher at Sandia for 8 years and, prior to that, was a researcher at John's Hopkins University Applied Physics Laboratory for 3. Previously, Christopher has researched machine learning applications within Bio Surveillance and Bibliometrics. His current areas of research centers around applied machine learning and cyber security evaluations and applications.

Deepu Jose is a cybersecurity R&D engineer at Sandia National Laboratories, and lead for multi-disciplinary initiatives to develop capabilities for assessing and defending diverse cyber-physical systems of critical interest to national security. Deepu's experience spans novel embedded systems development to modeling and simulation of cyber and physical system characteristics for cyber analysis; and more recently investigating security concerns in systems with machine learning. He holds a BS in Electrical Engineering from the University of Texas at Dallas, MS in Electrical and Computer Engineering from the Georgia Institute of Technology, and an MBA from the University of New Mexico.

References

Abdar, M., Pourpanah, F., Hussain, S., Rezazadegan, D., Liu, L., Ghavamzadeh, M., . . . Nahavandi, S. (2021). A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *Information Fusion, 76*, 243-297.

Atli, B. G., Szyller, S., Juuti, M., Marchal, S., & Asokan, N. (2020). Extraction of complex dnn models: Real threat or boogeyman? *Engineering Dependable and Secure Machine Learning Systems: Third International Workshop* (pp. 42-57). New York City, NY, USA: Springer International Publishing.

Bradley, E. J., Korfiatis, P., Akkus, Z., & Kline, T. L. (2017). Machine learning for medical imaging. *Radiographics, 37*(2), 505-515.

Chakraborty, A., Alam, M., Dey, V., Chattopadhyay, A., & Mukhopadhyay, D. (2018). Adversarial attacks and defences: A survey. *arXiv preprint arXiv:1810.00069*.

Fredrikson, M., Lantz, E., Jha, S., Lin, S., Page, D., & Ristenpart, T. (2014). Privacy in pharmacogenetics: An end-to-end case study of personalized warfarin dosing. *23rd USENIX Security Symposium (USENIX Security 14)*, (pp. 17-32).

Gebru, T., Morgenstern, J., Vecchione, B., Vaughan, J. W., Wallach, H., Iii, H. D., & Crawford, K. (2021). Datasheets for datasets. *Communications of the ACM, 64*(12), 86-92.

Goodfellow, I. J., Shiens, J., & Szegedy, C. (2014). Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

Gu, T., Dolan-Gavitt, B., & Garg, S. (2017). Badnets: Identifying vulnerabilities in the machine learning model supply chain. *arXiv preprint arXiv:1708.06733*.

Guynn, J. (2015, July 1). *Google Photos labeled black people 'gorillas'*. Retrieved March 10, 2022, from https://www.usatoday.com/story/tech/2015/07/01/google-apologizes-after-photos-identify-black-people-as-gorillas/29567465/

Hond, A. A., Leeuwenberg, A. M., Hooft, L., Kant, I. M., Nijman, S. W., Os, H. J., . . . Chavannes. (2022). Guidelines and quality criteria for artificial intelligence-based prediction models in healthcare: a scoping review. *NPJ digital medicine, 5*(1), 2.

Laplante, P., Milojicic, D., Serebryakov, S., & Bennett, D. (2020). Artificial intelligence and critical systems: From hype to reality. *Computers, 53*(11), 54-52.

Lavin, A., Gilligan-Lee, C. M., Visnjic, A., Ganju, S., Newman, D., Ganguly, S., . . . Gal, Y. (2022). Technology readiness levels for machine learning systems. *Nature Communications, 13*(1), 6039.

McFarland, M. (2022, December 21). *Tesla 'full self-driving' triggered an eight-car crash, a driver tells police*. Retrieved January 18, 2023, from https://www.cnn.com/2022/12/21/business/tesla-fsd-8-car-crash/index.html

Mitchell, M., Wu, S., Zaldivar, A., Barnes, P., Vasserman, L., Hutchinson, B., . . . Gebru, T. (2019). Model cards for model reporting. *Proceedings of the conference on fairness, accountability, and transparency*, (pp. 220-229).

Nagy, B. (2022). *Level of Rigor for Artificial Intelligence Development.* Naval Air Warfare Center Weapons Division.

Nguyen, A., Yosinski, J., & Clune, J. (2015). Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. *Proceedings of the IEEE conference on computer vision and pattern recognition*, (pp. 427-436).

Orekondy, T., Schiele, B., & Fritz, M. (2019). Knockoff nets: Stealing functionality of black-box models. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition.*

Pendlebury, F., Pierazzi, F., Jordaney, R., Kinder, J., & Cavallaro, L. (2019). TESSERACT: Eliminating experimental bias in malware classification across space and time. *Proceedings of the 28th USENIX Security Symposium* (pp. 729-746). USENIX Association.

Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). "Why should i trust you?" Explaining the predictions of any classifier. *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, (pp. 1135-1144).

Schwartz, O. (2019). In 2016, Microsoft's Racist Chatbot Revealed the Dangers of Online Conversation. *IEEE Spectrum*.

Shokri, R., Stronati, M., Song, C., & Shmatikov, V. (2017). Membership inference attacks against machine learning models. *2017 IEEE symposium on security and privacy (SP)* (pp. 3-18). IEEE.

Short, A., La Pay, T., & Gandhi, A. (2019). *Defending Against Adversarial Examples.* Sandia National Labs.

Smith, M. R., Krishnakumar, R., Lubars, J., Verzi, S., Zhou, X., & Goya, A. (2022). *All models are wrong, but some(times) are useful: Evaluating when machine learning models are useful for detecting novel malware in the wild.* Sandia National Laboratories.

Tangredi, S. J., & Galdorisi, G. (2021). *Tangredi, Sam J., and George Galdorisi, eds. AI at war: How big data, artificial intelligence, and machine learning are changing naval warfare.* Naval Institute Press.

Tian, Z., Cui, L., Liang, J., & Yu, S. (2022). Comprehensive Survey on Poisoning Attacks and Countermeasures in Machine Learning. *ACM Computing Surveys, 55*(8), 1-35.

Verdoliva, L. (2020). Media forensics and deepfakes: an overview. *IEEE Journal of Selected Topics in Signal Processing, 14*(5), 910-932.

Xu, K., Zhang, G., Liu, S., Fan, Q., Sun, M., Chen, H., . . . Lin, X. (2020). Adversarial t-shirt! evading person detectors in a physical world. *European Conference on Computer Vision* (pp. 665-681). Glasgow, UK: Springer International Publishing.

Yampolskiy, R. (2019). Unpredictability of AI. *arXiv preprint arXiv:1905.13053*.

Zhang, D., Maslej, N., Brynjolfsson, E., Etchemendy, J., Lyons, T., Manyika, J., . . . Perraulty, R. (2022). *Artificial Intelligence Index Report 2022.* AI Index Steering Committee, Stanford Institute for Human-Centered AI, Stanford University.

Appendix

ML Performance Rubric

The following rubric may be used to evaluate the performance of an ML model:

	High Risk	Medium Risk	Low Risk
Representative Datasets	The training and evaluation datasets contain several high-risk attributes relating to MA • Data is not provided for evaluation OR • Data has significant biases present. • Data does not represent data that will be encountered in deployed environments • Data has not been examined and features exist which make learning inappropriate • Data is not documented	The training and evaluation datasets are partially documented, match expert assumptions but still have some sources of uncertainty and risk • Data is provided for evaluation AND • Data has moderate or no biases that significantly affect the MA of the system. • Training and evaluation data match the data that is expected to be encountered in deployed scenarios with recognized deviations and planned remediations. • Data has been at least partially reviewed and some faults are identified with appropriate remediations.	The training and evaluation datasets are documented, match expert assumptions and have low uncertainty in the above criteria • Data is provided for evaluation AND • No significant biases exist • Training and evaluation data match the expected distribution once deployed • There are enough examples for an ML algorithm to learn • Data has been reviewed by experts AND is documented

		• Data is at least partially documented.	
Model Evaluation	Most or all of the following concerns are raised • Target metric is misaligned from the mission goals • Decision thresholds are not properly set • No hyperparameter tuning was done • Model is under or overfit • Model is not numerically stable • Model performs differently once integrated into the system • No documentation on the model or development and evaluation phases • Code is not versioned • Evaluation cannot be reproduced	The ML model is properly documented and evaluated, but some concerns still persist due to the nature of the ML model and environment • Evaluation criteria may be ill defined or misaligned with the mission • The deployed environment may be highly dynamic where a representative training and evaluation data set is difficult to obtain • Model is numerically stable • Code is maintained and versioned • Evaluation results are reproducible	The ML model is properly documented and evaluated and assumptions match those in the deployed environment • Evaluation criteria is well defined • The deployed environment is well understood, and representative training and evaluation dataset are used. • Model is numerically stable • Code is maintained and versioned • Evaluation results are reproducible
Deployed Model Evaluation	There is no acknowledgement or monitoring of changes to the deployed environment • No risks are laid out AND • No processes are in place to monitor changes in the data or the retrain	The need to monitor the dynamics of the deployed environment are acknowledged but not all aspects are fully covered • Risks from concept drift are enumerated and documented • The data is not monitored for changes OR • No process is in place to adapt to changes in the environment	The risks of concept drift are understood, and mitigations are in place • Data from the system is compared with the assumptions that were used during training • Mechanisms for updating the ML model are in place

Model Trust	No aspect is addressed to ensure trustworthy outputs from an ML model • No defenses of adversarial attacks are in place or acknowledged • No explanations are provided to help vet the decision process made by the ML model • Uncertainty from the model is not accounted for	Some aspects have been addressed for trustworthy outputs from an ML model. Not all components are addressed, but those most related to MA are satisfactorily addressed • Explanations are verified by a SME OR • Defenses are in place against adversarial attacks OR • Outputs have an associated uncertainty measure	All aspects have been addressed for prediction trustworthiness • Explanations are verified by a SME AND • Defenses are in place against adversarial attacks AND • Outputs have an associated uncertainty measure

This rubric was developed after consulting several existing resources in the literature (Nagy, 2022; Lavin, et al., 2022; Hond, et al., 2022; Mitchell, et al., 2019; Gebru, et al., 2021).