Cloud-based XAI Services for Assessing Open Repository Models Under Adversarial Attacks

Zerui Wang
Department of Electrical and Computer Engineering
Concordia University
Montréal, Québec, Canada
zerui.wang@concordia.ca

Yan Liu
Department of Electrical and Computer Engineering
Concordia University
Montréal, Québec, Canada
yan.liu@concordia.ca

***Abstract*—The opacity of AI models necessitates both validation and evaluation before their integration into services. To investigate these models, explainable AI (XAI) employs methods that elucidate the relationship between input features and output predictions. The operations of XAI extend beyond the execution of a single algorithm, involving a series of activities that include preprocessing data, adjusting XAI to align with model parameters, invoking the model to generate predictions, and summarizing the XAI results. Adversarial attacks are well-known threats that aim to mislead AI models. The assessment complexity, especially for XAI, increases when open-source AI models are subject to adversarial attacks, due to various combinations. To automate the numerous entities and tasks involved in XAI-based assessments, we propose a cloud-based service framework that encapsulates computing components as microservices and organizes assessment tasks into pipelines. The current XAI tools are not inherently service-oriented. This framework also integrates open XAI tool libraries as part of the pipeline composition. We demonstrate the application of XAI services for assessing five quality attributes of AI models: (1) computational cost, (2) performance, (3) robustness, (4) explanation deviation, and (5) explanation resilience across computer vision and tabular cases. The service framework generates aggregated analysis that showcases the quality attributes for more than a hundred combination scenarios.**

***Index Terms*—Software Engineering, Artificial Intelligence, Explainable AI, Adversarial Attacks, Cloud Services, Evaluation Frameworks**

I. Introduction

Artificial intelligence models are increasingly accessible through the open community [1] and facilitate advancements in software applications across various domains. These advancements, represented by innovations in deep-learning models such as ViT, ConvNeXt, CVT, Swin Transformers, SegFormer, and ResNet [2]–[7], underscore the rapid evolution of AI capabilities. However, the integration of these models into software applications necessitates a strict evaluation [8] of their quality attributes. A noted gap in current practices is the absence of a comprehensive service framework that facilitates an understanding of model explainability [9], [10].

The importance of explainable AI (XAI) has been increasingly recognized [11], driven by the need to build trust and ensure fairness within AI systems. XAI techniques, which aim to make the decision-making processes of AI models transparent [12], are essential in AI-enabled applications [13]. Additionally, adversarial attacks target vulnerabilities of AI models. They are modifications to input data that are less visible to humans but can make AI models give incorrect inferences or predictions [14], [15]. The rising threats [16] targeting security-sensitive models introduce significant challenges to the deployment of AI in software services. Therefore, effective software quality assurance requires a comparative analysis to guide the development and refinement of AI models, ensuring their robustness and explainability across diverse applications.

The integration of XAI techniques into AI models introduces additional computational layers [17]. The operational complexity of XAI evaluation [18] arises from multiple factors of managing diverse data types, integrating explanation methods with various AI models, evaluating explanations, and summarizing the results. These factors lead to evaluation scenarios that require dozens to hundreds of experiments. In addition, the complexity of assessment is further scaled by the product of multiple kinds of adversarial attacks. Comparative analysis between XAI and adversarial attacks increases the evaluation workload by necessitating the exploration of interactions between models and explanations under various adversarial conditions.

The goal is to address these complexities by automating the evaluation pipeline. We propose a cloud-based service framework that encapsulates computing components as microservices and organizes assessment tasks into pipelines. This framework also integrates open XAI tool libraries, which are originally not inherently service-oriented, as part of the pipeline composition.

In this study, we assess six vision models against three types of adversarial attacks employing five XAI methods, resulting in a total of ninety distinct combinations. Moreover, we evaluate three transformer-based tabular models using two XAI methods across three datasets, resulting in eighteen combinations. We set pipelines to assess quality attributes for every combination scenario, including (1) computational cost, (2) performance, (3) robustness, (4) explanation deviation, and (5) explanation resilience. We assessed these attributes across a range of model and XAI method combinations. Additionally, the evaluation results indicate that higher explanation deviation requires more computational costs. Finally, the study demonstrates the impact of adversarial attacks on model performance

and their explanation.

II. Background and Related Work

In this section, we introduce the XAI methods. We review the current XAI tools and frameworks. Then, the taxonomy of adversarial attacks.

A. Introduction of XAI Methods

Mainstream post-hoc XAI methodologies can be categorized into two distinct types [18]: model-specific and model-agnostic methods. Model-specific methods [19]–[23] derive feature importance values from the internal parameters of the model itself, offering insights directly linked to the model's internal mechanisms. In contrast, model-agnostic methods [24] establish feature importance by analyzing the relationship between the model's inputs and outputs without requiring the model's internal parameters and layers.

XAI in computer vision often involves generating visual explanations for the decisions made by AI models. The model-specific methods are computationally efficient, rather than the model-agnostic methods [25]. For demonstration, Figure 1 presents selected examples derived from the Vision Transformer model [3] through our XAI service framework. Significant variations in results are observed from the same input data. These results indicate the need for establishing systematic XAI evaluations within the XAI service framework, which is notably absent in previous tools and frameworks.

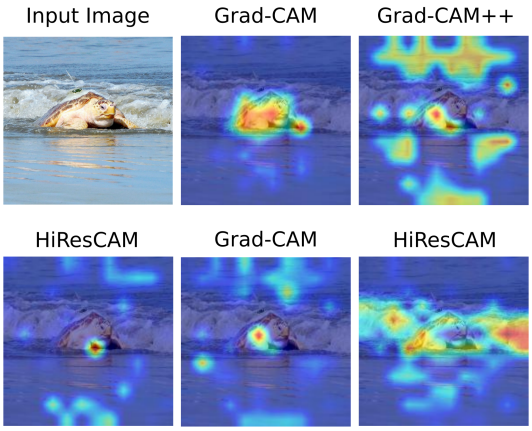

Fig. 1. Five CAM-based Visual Explanations from Vision Transformer Model with One Image Example.

Grad-CAM [19] utilizes the gradients of the target label flowing into the final convolutional layer to produce a coarse localization map highlighting important regions for prediction. According to an optimization work, Grad-CAM++ [20] extends Grad-CAM by considering the weight of each pixel in the feature maps, allowing better handling of images with multiple occurrences of the same object. HiResCAM [21] generates high-resolution class activation maps, allowing for finer detailed visual explanations with higher computational cost. XGrad-CAM [22] focuses on increasing the linearity of the saliency maps, providing improvements. LayerCAM [23] introduces a layer-wise relevance propagation mechanism, which enables the feature importance across different network layers for more details.

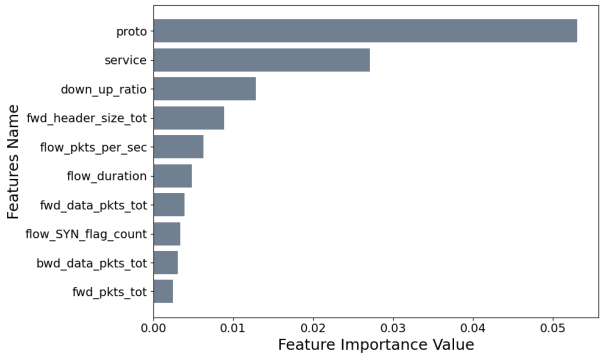

Fig. 2. Top 10 out of 83 SHAP Feature Importance Explanations from FT Transformer on RT-IoT Cybersecurity Threats Dataset.

We also apply XAI methods involving structured data within the tabular models. Mean Centroid Prediff [26] and SHAP (SHapley Additive exPlanations) [24] stand out for widespread acceptance as the baseline. These methods quantify feature importance by assessing the impact of masking each feature on model output. The methods provide a score for each feature, indicating its contribution to model predictions. Figure 2 illustrates an example of the top ten out of eighty-three global feature importance explanation from the RT-IoT2022 cybersecurity classification dataset [27]. We carry out XAI to determine the impact of the network logs features on the classification of threats. We provide data-driven explanation deviation metrics to evaluate XAI methods in the section V scenario studies.

B. XAI Tools Libraries and Frameworks

The role of XAI in providing feature contribution explanations establishes trust in AI models [26]. Before introducing XAI as a service, we review the published tools and libraries. The Explainability 360 toolkit by IBM [28] integrates their explanation techniques within the toolkit package. Microsoft's InterpretML [29] offers support for eight tabular models' XAI approaches. The recent framework OmniXAI [30] offers a broad range of techniques for XAI. However, based on our tests and usages, the existing XAI frameworks have the following limitations:

Expertise-Dependent Usage: The use of these tools often demands specific XAI knowledge expertise, which limits accessibility [31] to software engineers.

Restricted Methods Support: Each library support limited numbers of different XAI methods [28]–[30].The disparity in the number of methodologies supported by these tools complicates the selection process and necessitates additional preprocessing.

Evaluation Procedures Deficiency: The comprehensive study [32] that performed quantitative evaluations on saliency methods is relevant. However, the current XAI tools lack

standardized procedures for evaluating explanation results, which limits the selection and enhancement of these XAI tools [10], [33].

Cloud Service Support Limitations: Explicit support for cloud AI services is rare among these tools, affecting their applicability across cloud service environments.

These limitations underline the need for a service architecture that addresses these gaps, making XAI accessible and effective for diverse AI-enabled applications.

C. Adversarial Attacks Types

Adversarial attacks are designed to mislead models, and pose challenges to AI system security [34]. The adversarial attacks are briefly categorized into white-box and black-box attacks, each with distinct methodologies and implications. The taxonomy of adversarial attacks with example works are listed in Figure 3.

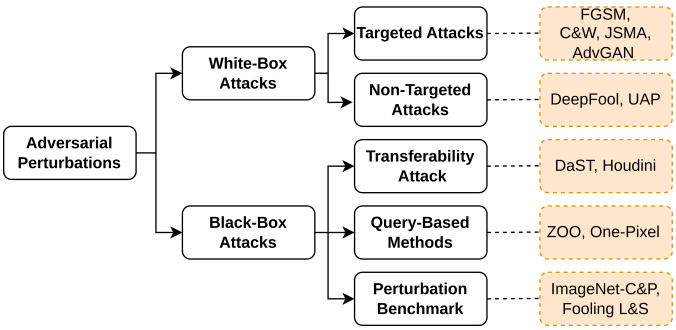

Fig. 3. Taxonomy of Adversarial Attacks. References: FGSM [35], C&W [36], JSMA [37], AdvGAN [34], DeepFool [38], UAP [39], DaST [40], Houdini [41], ZOO [42], One-Pixel [43], ImageNet-C [44], ImageNet-P [44], Fooling LIME and SHAP [45].

In white-box attacks, the attacker needs certain knowledge of the model, including its architecture and parameters. Targeted attacks [34]–[37] aim to manipulate the model to produce a specific, incorrect output. Non-targeted attacks [38], [39] aim to generate any incorrect response from the model.

In black-box attacks, where the attacker has no information about the model's internals, the focus shifts to transferability attacks, query-based methods, and perturbation benchmarks. The perturbation benchmarks refer to algorithmically induced modifications applied to the data with the intent to mislead the model. The benchmark modifications can affect models and further be employed to measure the model's robustness [44]. The study [44] presents corruption and perturbation as two methodologies for modifying images. Corruption typically refers to modifications that simulate natural or environmental degradation. Perturbation denotes generated changes in sequence. We merge these two groups algorithmically modify methods in the study [44], under the broader category of adversarial perturbations. In the assessing scenarios, we adopt the ImageNet-C benchmark [44]. The benchmark [44] contains fifteen types of algorithmically generated corruptions from categories such as noise, blur, weather, and digital. The weather category contains corruptions that demonstrate an excessive diversity. In the assessment scenarios, we select a representative algorithm from each of the three main categories. We apply nine distinct corruption perturbations across three levels of severity. The Fooling LIME and SHAP [45] method is a recent study using data perturbation to attack LIME [46] and SHAP [24]. We also develop and launch multiple levels of perturbation attacks to the numerical features in selected tabular datasets for scenarios.

III. Overview of AI System's Quality Attributes

AI-based system refers to a software system that adopts the AI models as components [47]. Software quality is the ability of a software product to meet requirements during its operation in designated conditions [48]. Software quality assurance involves an evaluation process of how well a software product satisfies its stated needs [49]. However, the AI models are distinct from conventional software components. Their core characteristic is being data-centric [50]. Additionally, these components are dynamic and continually evolving as they are exposed to new data over time [50]. Inadequate, inaccurate, or unsuitable training data can result in unreliable models and biased decisions [51]. To address quality assurance, the study [52] presents the concept of early adoption of XAI into model development.

We define quality attributes of AI models and explanations output by XAI analysis in terms of `model performance` and `explanation deviation`. Then, when AI models are under adversarial attacks, the quality attributes are `model robustness` and `explanation resilience` accordingly. Besides, `computation cost` is related to deployment. Overall, metrics defined for these quality attributes capture the combination states of AI models, XAI methods, adversarial approaches, and datasets. The results can be plotted on a radar map, indicating each quality attribute transforms into the normalized value.

Computational Cost. Computational cost measures the resources required to execute an algorithm. The computational cost includes measuring the utilization of CPU, GPU, and Memory. We record the computational cost attribute that encompasses the runtime and energy consumption of AI models and XAI techniques. These metrics are measured in seconds (s) for runtime and watt-hours (Wh) for energy consumption. The CodeCarbon library [53] supports the program to track resource utilization by monitoring hardware-specific parameters. In addition, the CodeCarbon estimates the environmental carbon footprints based on energy consumption.

Model Performance. The evaluation of model performance contains a range of metrics. The typical metrics are Top-N accuracy metrics [54], precision, recall, and F1 score [55]. In the multi-class context, we aggregate True Positives (TP), False Positives (FP), True Negatives (TN), and False Negatives (FN) across all classes. In addition, the Area Under the Receiver Operating Characteristics Curve (AUC-ROC) is also a calibrating metric to assess a model's ability to distinguish between classes [56].

Model Robustness. Robustness evaluation measures a model's performance degradation under specific adversarial conditions. For computer vision models, the use of an image adversarial perturbation benchmark [44] assesses the model under adversarial conditions. Mean Corruption Error (mCE) represents the model robustness in a previous study [44]. This metric aggregates the normalized error rates for a model across corruption types and their respective severity levels. The formulation for mCE [44] is detailed below:

$$mCE = \frac{1}{S_c} \sum_{s=1}^{S_c} \left(\frac{E_s^c(f)}{E_s^c(f_{\text{ref}})} \right) \tag{1}$$

Here, f denotes the model under evaluation. For corruption type c, S_c indicates the severity levels. The variable $E_s^c(f)$ is the error rate of the model f, while $E_s^c(f_{\text{ref}})$ specifies the error rate of a reference model, such as Alexnet according to the study [44].

Instead of comparing with a reference model, we set Kolmogorov-Smirnov (K-S) statistic [57] D_{ks} as a quantifiable metric. It represents the comparison of the distribution of the model outputs between two datasets:

$$D_{ks} = \sup_X |F(X_{\text{orig}}) - F(X_{\text{adv}})| \tag{2}$$

Here, $F(X_{\text{orig}})$ and $F(X_{\text{adv}})$ symbolize the model probabilities distribution of the original and adversarial datasets, respectively. The usage of $\sup_x$ targets the maximum divergence between these two model inference distributions. By incorporating the K-S statistic, we redefine the robustness against adversarial attacks. Equation 3 represents the assessment. A smaller value indicates better model robustness:

$$Robustness = \frac{1}{S_c} \sum_{s=1}^{S_c} D_{ks} \tag{3}$$

For the tabular model, we generate feature perturbations to datasets. Specifically, in the scenario, we make a random perturbation to the feature with the numerical severity factor. Similar to our approach, the adversarial attack [45] introduces a designed biased perturbation, instead of our random perturbation, to attack the SHAP [24] and the LIME [46] methods. The robustness Equation 3 can be used in tabular cases.

Explanation Deviation. Explanation deviation is an assessment of the impact of the feature importance explanation on the model's outputs. At its core, explanation deviation measures the discrepancy between a model's predictions when all features are considered versus when only those important features are emphasized. Therefore, measuring explanation deviation is a means to validate the actual influence of presumed features on the model's outputs.

In vision scenarios, the saliency map is applied to assess how important each pixel is to a model's output, creating visual importance maps. Grad-CAM [19] is commonly accepted in CNN-based vision model [2] explanation. However, whether these XAI perform as effectively for more transformer-based models remains under evaluation [58]. Explanation deviation is determined by the prediction change score, which indicates the impact of the saliency map area on the model's predictions. The masked image is calculated as the element-wise multiplication of the original image X and the normalized, three-dimensional saliency mask $M_{\text{norm 3D}}$. Here, P is defined as the model's prediction class probability values:

$$Deviation = 1 - (P(X_{\text{orig}}) - P(X_{\text{orig}} \cdot M_{\text{norm 3D}})) \tag{4}$$

A smaller change in probability value, then the deviation close to one. Extending to the whole dataset, the overall explanation deviation is the statistical analysis of their median value. In tabular scenarios, the XAI method computes feature importance values for each data sample. The explanation deviation is represented by evaluating the consistency in the feature importance order, according to the study [18].

Explanation Resilience. Resilience to adversarial attacks is measured by the difference in explanation deviation between non-adversarial and adversarial conditions. Therefore, the explanation resilience is quantified by the Equation 5:

$$\begin{aligned} Resilience = &(P(X_{\text{orig}}) - P(X_{\text{orig}} \cdot M_{\text{norm 3D}})) \\ &- (P(X_{\text{adv}}) - P(X_{\text{adv}} \cdot M_{\text{norm 3D}})) \end{aligned} \tag{5}$$

The resilience attribute indicates that the explanation deviation metric decreases for adversarial reasons. For tabular data, resilience can be calculated by subtracting the explanation deviation of adversarial data from the original data. Comparing the original and perturbed situation allows for resilience assessment: The smaller the resilience, the better the explanation can resist the impact of adversarial attacks.

IV. Pipelines of XAI Centric Assessment of Quality Attributes

In this section, we define the pipeline for obtaining quality attributes. Then, we illustrate the cloud-based service architecture. The operational overhead for XAI service is compared with using the existing framework.

A. Define Pipelines for Each Quality Attribute

This study integrates five quality attribute evaluations. Figure 4 shows the multiple parallel processes for accessing quality attributes. This multifaceted assessment evaluates quality attributions under both original and adversarial conditions.

Computational Cost Pipeline: This pipeline records the computational resources used during model inferences and XAI method executions. The average resource consumption is logged by the `Coordination Center`, when processing a significant volume of data inputs. The record covers AI models, XAI methods and evaluations.

Model Performance Pipeline: To assess model performance, the pipeline begins with the datasets provided by the `Data Processing Microservice`. These datasets are sequentially fed into the `Models Microservice`, which encapsulates the pre-trained model. The outcomes are extracted and persisted. The `Evaluation Microservice` derives performance metrics.

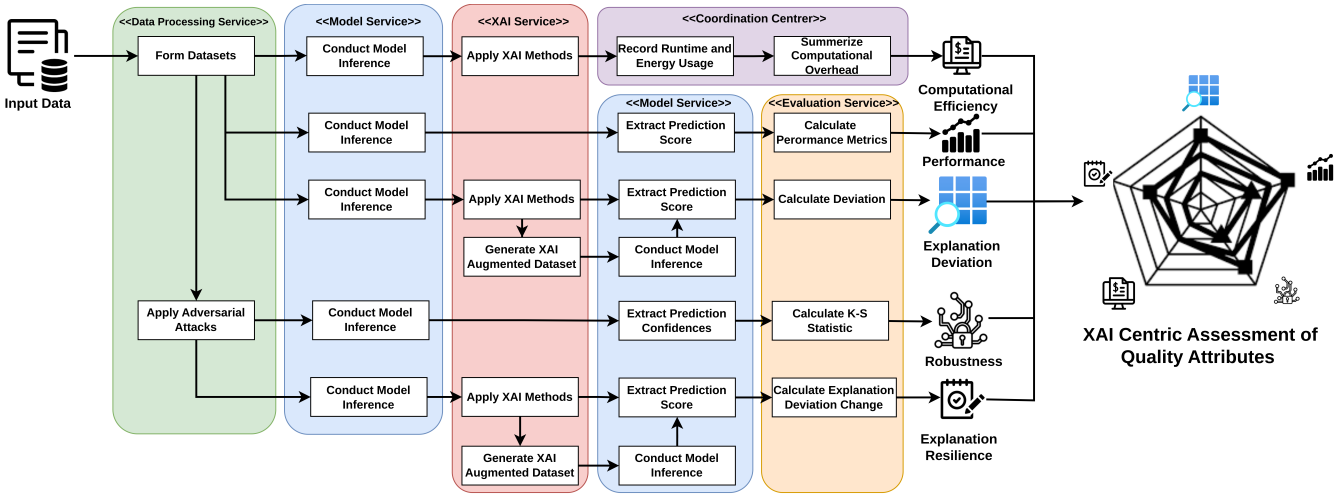

Fig. 4. Assessment Pipelines for Open-source AI Model Quality Attributes.

Explanation Deviation Pipeline: This pipeline calculates explanation deviation. `XAI Methods Microservice` are applied to generate explanations. For computer vision tasks, the explanation results are fed back into the model to calculate the prediction confidence drop, thereby calculating explanation deviation. The prediction changes value within consistency metric [18] can also be directly derived from tabular models.

Robustness Pipeline: To assess model performance decline under adversarial attacks, the pipeline begins with the `Data Processing Microservice` preparing perturbed data. The `Models Microservice` then processes both the original and perturbed datasets, yielding two sets of results. The `Evaluation Microservice` measures shifts in the model's performance between the original and perturbed datasets.

Explanation Resilience Pipeline: Similar to the robustness pipeline but with a focus on XAI methods, this pipeline starts with the `Data Processing Microservice` preparing perturbed datasets. These datasets are then subjected to selected models and XAI methods. The `Evaluation Microservice` calculates the changes in explanation deviation.

The individual pipeline is configurable for AI models, XAI methods, and adversarial datasets. They can be composed into singular or combined quality attributes assessment scenarios. Their deployment and execution need a runtime service-oriented architecture, which is defined in the next subsection.

B. Define Services Architecture for the Pipeline Configuration

We introduce a cloud service architecture designed to develop XAI assessment pipelines. This architecture enables analysis and comparison of various combinations of AI models, XAI methods, datasets, and adversarial attack approaches, as shown in Figure 5.

Coordination Center: The microservice executes unit operations based on configuration, communicates with other

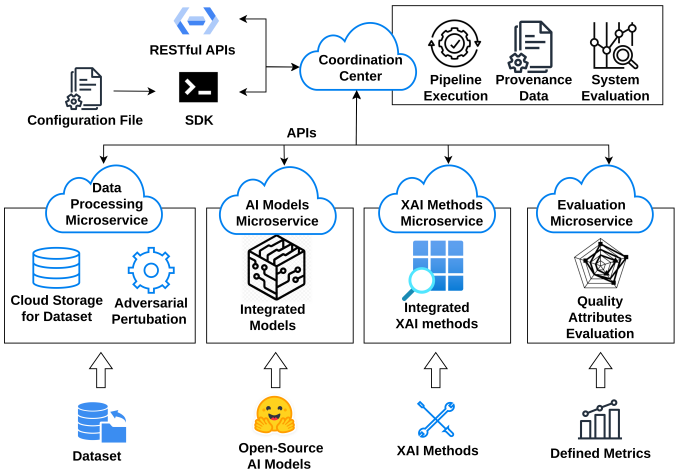

Fig. 5. Cloud-based XAI Service Architecture.

microservices as per task setup, and records provenance data for transparency and reproducibility.

Data Processing Microservice: This microservice ensures data is correctly formatted and meets XAI algorithm requirements. It also applies adversarial attack conditions.

AI Model Microservice: This service encapsulates and deploys pre-trained AI models, including open-source models from the HuggingFace Community, within its framework.

XAI Method Microservice: This service offers XAI that can generate explanations. XAI algorithms [26] and tool libraries [30] can be encapsulated into the service.

Evaluation Microservice: This service aggregates results and systematically evaluates defined quality attributes.

Collectively, these components create a cloud-based architecture supporting the complete evaluation process. The architecture's flexibility allows for switching services to test various AI models or XAI methods, facilitating extensive investigation into numerous combinations. Additionally, the deployed services are reusable across multiple pipelines.

```
"xai_config": {
  "base_url": "xaiport.ddns.net:8003", //"address"
  "datasets": {
    "t1024_gaussian_2": {     //"datasets id"
      "model_name": "resnet", //"model name"
      "algorithms": [
        "GradCAM", //"XAI methods name"
        "HiResCAM",
        "GradCAMPlusPlus",
        "XgradCAM",
        "LayerCAM"
      ]
    }
  }
}
```

Fig. 6. Sample JSON Template for XAI Methods Configuration

In addition, the JSON-based configuration template, as present in Figure 6, specifies how the service executes the pipeline according to the user's inputs. The template allows users to define the interaction between different services and customize the evaluation process to suit specific requirements. The execution of pipelines is through the use of coordination centers, each configured with its JSON-based configuration file. Upon receiving a configuration file, a coordination center systematically accesses the specified microservices to execute each pipeline step.

C. Comparative Evaluation and Service Integration of Existing XAI Frameworks

The architecture enables the encapsulation and customisation of the external XAI algorithms and frameworks. XAI microservices can encapsulate not only XAI methods but also published libraries and frameworks. For instance, OmniXAI [30] aggregates enriched XAI methods for various data types. We import the OmniXAI package and employ the related functionalities to compute explanations in the XAI methods microservices. The rest of the required units can continue to adopt our service architecture. To verify the integration of the external tool framework, We compare our XAI service and the recently published OmniXAI framework [30]. We focus on differences in task reproducibility, metrics, and ease of use.

Reproducible: Reproducible ensures that results can be reliably validated. Other frameworks offer explanation methods but lack detailed provenance data recording. Validating and reproducing an XAI process requires obtaining the original dataset and editing the source code. This demands effort to reproduce the complex XAI process manually. Our service framework makes the evaluation pipeline reproducible using provenance data. Users can execute the pipeline with a single command using provenance data as the configuration file.

Evaluation Metrics: Other XAI frameworks do not provide evaluations for their methods. Without rigorous evaluation, the effectiveness of the explanations provided by different XAI remains undetermined. Our framework introduces the calculation of XAI consistency metrics [18]. We also define the XAI-centric assessment quality attributes.

TABLE I
OPERATIONAL OVERHEAD COMPARISON BETWEEN XAI SERVICE AND TOOL FRAMEWORK

Steps	**Our Service**	**OmniXAI [30] Framework**
Data Preparation	Automate data upload with formatted templates.	Script data transformation and insertion.
Environment Setup	Automate environment setup through Docker container.	Manage dependency installation, library configuration, and compatibility.
Configuration	Utilize JSON for configuration with defined rules.	Configure interactions and dataset linkages manually.
Pipeline Execution	Execute pipeline with a single SDK command.	Load datasets, inferences, and compute XAI manually.
Results Analysis	Provide built-in metrics and visualization.	Summarize results with custom coding.
Adjustments	Support adjustments via JSON editing.	Modify code and restart processes manually.

Ease of Use: Our framework provides a streamlined SDK interface that simplifies pipeline execution. Table I provides a described comparison of the operational steps. The JSON-based configuration template allows users to select datasets, models, and XAI methods in a structured manner. By automating several steps such as data preparation and environment setup, our service minimizes the operational overhead for systematic XAI processes.

Result Values Comparison: We compared the explanation deviation results of our service framework with those of existing tools using the ImageNet dataset [44], [59] test set, containing ten thousand images.

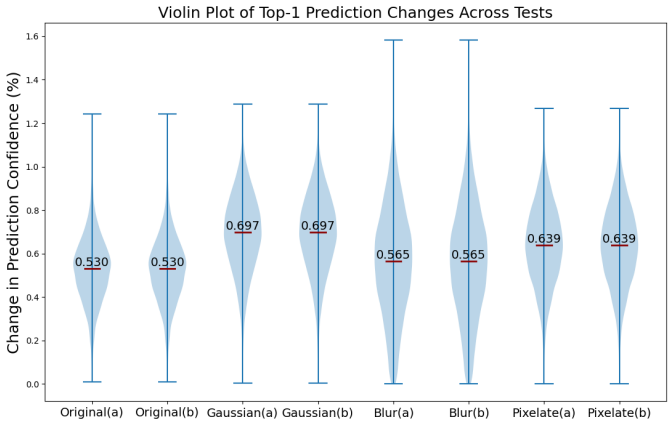

Fig. 7. Explanation Deviation Analysis for XAI Service (a) Versus OmniXAI (b) Using GradCAM on ResNet

Our XAI service includes multiple CAM-based methods [19]–[23]. The recent tool framework OmniXAI [30] employs only Grad-CAM [19] as the XAI method for vision models. Figure 7 presents a comparative analysis of the results from implementing Grad-CAM algorithms in our service framework versus OmniXAI. The results show the same prediction confidence distribution. However, our XAI service framework significantly streamlines operations.

Additionally, our framework is containerized and suitable for cloud platform environments. The RESTful API design enables easy integration with external cloud model services.

V. Assessment Scenarios and Quality Attribute Analysis

We introduce experimental scenarios to demonstrate the service's functionality in scenarios. By running the pipelines defined by the service framework, we aim to investigate the research questions as follows:

- ***RQ1****: Are the explanation deviation generated by XAI methods variable across models with different structures?*
- ***RQ2****: What is the relationship between computational cost and explanation deviation in model-XAI combinations?*
- ***RQ3****: Considering the known impacts of adversarial perturbations on model performance metrics, how do these perturbations influence the explanation deviation?*

The related source code and experimental results can be found in the GitHub repository. [1] Experiments are conducted in a controlled environment to ensure consistency. The experimental evaluation used a local setup with an Nvidia RTX 4090 GPU based on the AD102 graphics processor. This GPU features a core clock speed of 2.52 GHz and 24 GB GDDR6X VRAM. All experiments were conducted on a Linux-6.2.0 system, Ubuntu 22.04 LTS, Python 3.8.18 with CUDA 12.1, and PyTorch 2.1.0 to leverage the GPU's capabilities.

A. The Assessment Scenario of Vision Models

Process Configuration: Our dataset comprised 10,000 images from ImageNet [59], covering one thousand classes. We introduced three types of adversarial perturbations in the ImageNet-C [44]: Gaussian Noise, Defocus Blur, and Pixelate, with each type occurring at three severity levels.

TABLE II
Selected State-of-the-Art Vision Models from the Huggingface Repository

Vision Model	Publisher	Huggingface Repository
ViT	Google	google/vit-large-patch32
ConvNeXt	Meta	facebook/convnext-tiny
CVT	Microsoft	microsoft/cvt-13
Swin	Microsoft	microsoft/swin-large-patch4
SegFormer	Nvidia	nvidia/mit-b0
ResNet	Microsoft	microsoft/resnet50

As list in Table II, the models selected for evaluation are Vision Transformer (ViT) [3], Convolutional Neural Networks Next (ConvNeXt) [6], Convolutions Vision Transformer (CVT) [7], Swin Transformer (Swin) [4], Semantic Segmentation with Transformers (SegFormer) [60], and Residual Networks (ResNet) [2]. These models are top-performance sourced from the Huggingface open repository. The publisher has pre-trained these models. We select the ResNet [2] as a baseline. Additionally, our experiments employ various CAM-based methods [19]–[23] to generate saliency maps for vision models.

[1]The GitHub repository link will be made available in the camera-ready version due to the double-blind review process.

Computational Cost Analysis: Evaluating computational costs is essential to understand the practical implications of deploying XAI techniques in applications. The assessment covers processing time and energy consumption for different AI models using XAI techniques.

TABLE III
Analysis of Computational Costs for Vision Models Based on per One Thousand Images

Model	Inference Time (s)	Mean XAI Time (s)	Pipeline Energy (Wh)
ViT	13.35	62.50	5.58
ConvNext	5.73	41.19	2.67
CVT	16.91	69.76	5.86
ResNet	10.37	45.34	3.32
Swin	**32.38**	**137.24**	**10.07**
SegFormer	13.62	65.08	5.73

Table III illustrates each model's combined inference time, XAI time, and total energy in the pipelines. The results indicate significant variation in total processing time and energy consumption across models. The Swin Transformer shows the highest total process time and energy consumption, indicating a significant computational demand. The CVT model consumes the second most energy and runtime. Conversely, ConvNext has the lowest energy consumption and a relatively shorter total processing time. This is reasonable because the employed ConvNext model [6] is a relatively tiny-sized model. This conclusion suggests that CNN-based models, such as ConvNext and ResNet, are still potentially more viable options for applications where energy efficiency and faster processing are crucial.

In summary, these findings indicate significant variations in both time and energy consumption across different models. XAI consumes significantly higher computational costs compared to model inference.

Model Performance Analysis: The objective is to evaluate and compare the performance of leading open-source vision models from the Huggingface repository, applying a range of adversarial perturbations.

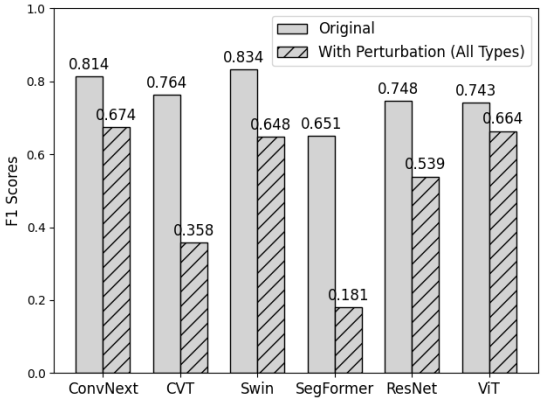

Fig. 8. Vision Models F1 Scores: Original Dataset and Adversarial Dataset Averages.

Figure 8 displays the F1 scores for six computer vision models under both original and adversarial perturbation conditions, emphasizing their performance. Initially, all models exhibit high performance on the original dataset. However, adversarial perturbations have varying effects on the performance of each model. Models indicate performance degradation under adversarial perturbations. Specifically, CVT and SegFormer show obvious vulnerability against adversarial perturbations.

Model Robustness Analysis: This analysis evaluates model robustness against different levels of adversarial perturbations. The aim is to quantitatively evaluate and compare the robustness by assessing their ability to maintain prediction accuracy. To analyze changes in prediction distributions under adversarial perturbations and further assess model robustness, we employ the introduced Kolmogorov-Smirnov (K-S) statistic. Initially, we extract the probability values from the model inference of the original dataset. Next, we extract prediction probabilities for each perturbed dataset and compute the K-S statistic. A higher K-S value indicates a greater shift in the model's outputs, implying a significant reduction in robustness.

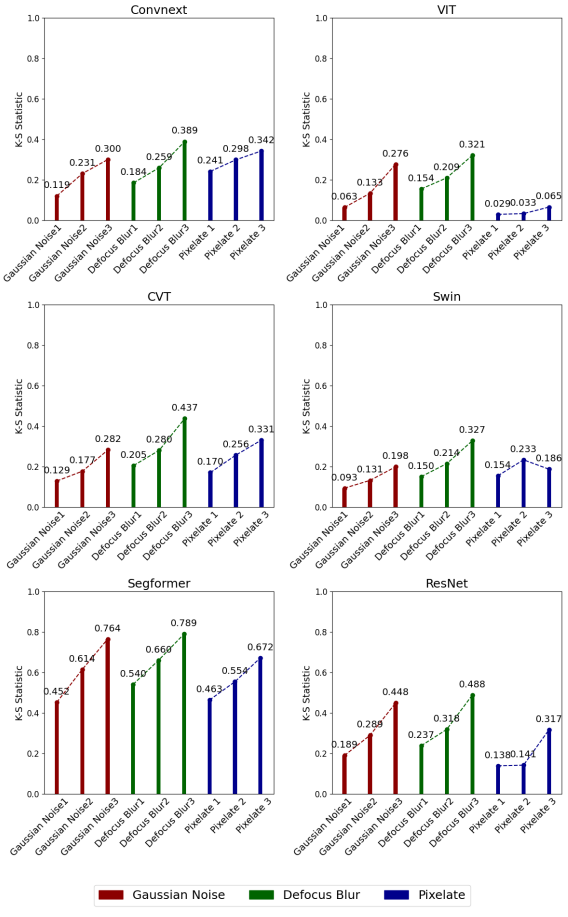

Fig. 9. Comparison of K-S Statistics to Assess Model Robustness under Three Levels and Types of Perturbations (0: Identical, 1: Highly Divergent).

Figure 9 shows the K-S values offer perspectives on model robustness. SegFormer consistently exhibits higher K-S values across all perturbations, indicating significant prediction shifts under adversarial attacks. This suggests vulnerability in maintaining prediction consistency against such perturbations. Conversely, models such as ViT and Swin exhibit lower K-S values, implying more robust performance under adversarial conditions. The bar plots in Figure 9 also display K-S statistic values for each perturbation type (Gaussian Noise, Defocus Blur, Pixelate) across three levels (1, 2, 3). Each bar's height indicates the deviation of the model's prediction distribution from its original, caused by a specific perturbation. Different colors for each perturbation type enhance visual distinction.

As a result, this quantitatively evaluates the robustness of various models under diverse adversarial perturbations. The Vision Transformer and Swin Transformer models exhibit relatively higher robustness.

Explanation Deviation Analysis: The objective is to evaluate the impact of adversarial perturbations on the explanation deviation. The utilization of saliency maps to annotate images assists developers in identifying the reason for model inaccuracies, thereby enhancing model performance.

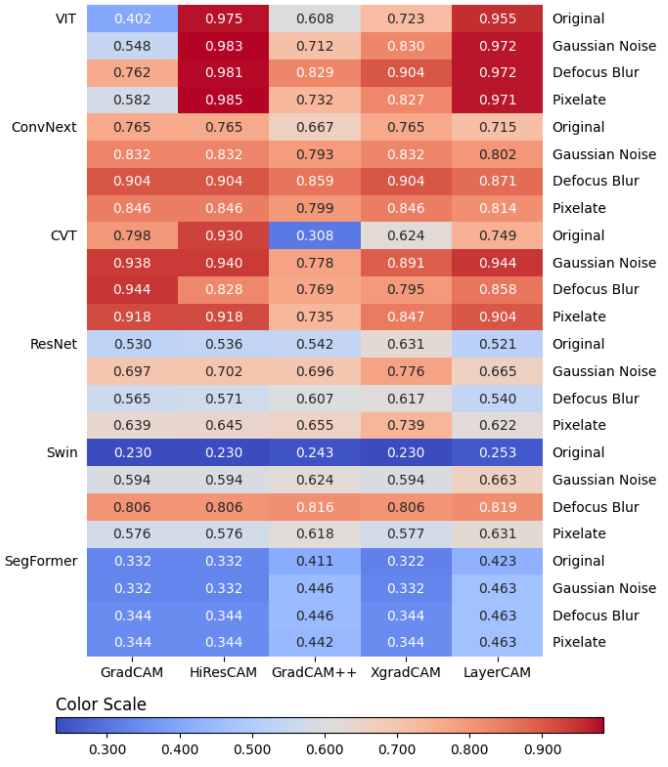

Fig. 10. Heatmaps Illustrating Median Prediction Change Percentage for Original and Adversarial Perturbed Images. Lower Values Indicate Better Explanation Deviation.

Figure 10 shows prediction change percentages on a heatmap. The explanation deviation attributes can be calculated as one minus the value shown in the figure. The figure is 3-dimensional: the left rows label six vision models, while the right rows specify the original dataset and three types of adversarial perturbation. The labels at the bottom represent various XAI methods. The value in each cell reflects the

summarized percentile of prediction value changes observed between the original input and the explanation inputs. Lower values mean the model's inference is less affected by irrelevant feature masking, indicating XAI makes a closer approximation of relevant features. Consequently, a lower summarized prediction change percentage values in Figure 10, correlates with better XAI explanation deviation.

Significant variations are observed across different model-XAI combinations. The results reveal that the Swin Transformer model maintains consistent explanation deviation across varied XAI methods and adversarial perturbation types, as confirmed through extensive cross-validation.

Explanation Resilience Analysis: The impact of adversarial perturbations in XAI scenarios remains an under-explored question. This analysis seeks to determine if adversarial perturbations contribute to misleading explanations.

Figure 10 shows the prediction change percentages under adversarial perturbations, which are used to calculate explanation resilience attributes. Additionally, Table IV offers a concise summary of the explanation resilience results. The smaller the resilience value, the better the explanation can resist the impact of adversarial attacks.

TABLE IV
SUMMARY OF ORGINAL DEVIATION, ADVERSARIAL DEVIATION AND EXPLANATION RESILIENCE FOR VISION MODELS

Model	Original	Adversarial	Resilience
VIT	0.267	0.160	0.107
ConvNext	0.265	0.155	0.110
CVT	0.310	0.133	0.177
ResNet	0.448	0.351	0.097
Swin	0.763	0.327	**0.436**
SegFormer	0.636	0.614	0.022

The analysis shows that the explanation resilience attribute does not align with performance and robustness attributes. The findings reveal that models, such as Swin Transforme, retain consistent explanation deviation, whereas show a significant decrease when encountering adversarial perturbations.

B. The Assessment Scenario of Tabular Models

In the case of structured data, we present the experimental results for all quality attributes.

Process Configuration: Our study evaluates tabular models using datasets from various domains. COMPAS [61] Recidivism Risk Score Data and Analysis dataset is instrumental in assessing the predictive accuracy of recidivism, offering a rich source of sensitive real-world data. RT-IoT2022 [27], the 2022 Real-Time Internet of Things (IoT) dataset [27] for cybersecurity threats case, is a collection of network traffic data. PriceRunner [62] Product Classification and Clustering dataset provides a scenario in the e-commerce domain.

The TabTransformer [63], TabNet [64], and FT Transformers [65] are employed as the models for tabular data. The TabTransformer [63], a model inspired by the Transformer architecture, is adapted for tabular data by encoding categorical features into embeddings. TabNet [64] utilizes sequential attention to choose features for each decision step. The FT Transformer [65] optimizes the transformer model to handle numerical features. In terms of XAI methods, Mean Centroid Prediff [26] and SHAP [24] are applied.

Computational Cost Analysis: For the computational cost analysis, we test the time and energy consumption for the three model inference, Mean centroid prediff [26], and SHAP [24] method. Table V presents the recorded time and energy consumption for every thousand samples using the algorithms.

TABLE V
ANALYSIS OF COMPUTATIONAL COSTS FOR TABULAR MODELS BASED ON PER THOUSAND ROWS

Model and XAI	Time (s)	Energy (Wh)
TabTransformer	16.04	0.49
TabNet	19.39	0.56
FT Transformers	15.62	0.47
Mean Centroid Prediff	1112.28	53.87
SHAP	768.13	26.74

The costs for the three transformer-based tabular models are comparable. However, XAI takes significantly higher computational costs than model inference. It is observed that the Mean Centroid Prediff [26] has higher time and energy consumption compared to SHAP [24].

Model Performance and Robustness Analysis: We evaluated the performance of various transformer-based tabular models [63]–[65] against adversarial perturbations. The performance metrics before and after the adversarial perturbations are shown in Table VI.

TABLE VI
ANALYSIS OF MODEL PERFORMANCE AND XAI DEVIATION IN TABULAR SCENARIOS

Dataset	**Model**	**Model Performance**		**XAI Deviation**	
		Original	**Adv. Changes**	**Original**	**Adv. Changes**
COMPAS	TabTransformer	0.683	-0.118	0.965	-0.105
	TabNet	0.674	-0.114	0.989	-0.087
	FT Transformers	0.690	-0.103	0.965	-0.071
RT-IoT	TabTransformer	0.921	-0.435	0.987	-0.060
	TabNet	0.883	-0.364	0.985	-0.078
	FT Transformers	0.950	-0.603	0.984	-0.088
PriceRunner	TabTransformer	0.994	-0.175	0.977	-0.063
	TabNet	0.991	-0.179	0.973	-0.062
	FT Transformers	0.997	-0.174	0.973	-0.049

The adversarial perturbation causes the model performance to decrease. Among the tests, the performance results decrease remarkably, especially for the RT-IoT dataset, which primarily consists of numerical data derived from sensors.

Explanation Deviation and Resilience Analysis: We employ the Mean Centroid Prediff [26] and SHAP [24] methodologies to calculate feature importance values. These explanation deviations are also shown in Table VI. The average adversarial decrement in explanation deviation for Mean Centroid Prediff stands at 0.047, in contrast to a 0.099 reduction for SHAP. This reveals that Mean Centroid Prediff exhibits superior resilience compared to SHAP; however, it comes at the cost of increased computational demands.

C. Conclusions and Insights on Research Questions

The five quality attributes are designed to comprehensively evaluate the open-source models. Figure 11 presents a radar chart visualizing the comparative analysis of selected vision and tabular models across the quality attributes. It is noted that the chart's values have been normalized, where each value closer to one indicates better the attributes. This evaluation aims to support engineers in selecting and developing explainable models in AI-enabled software.

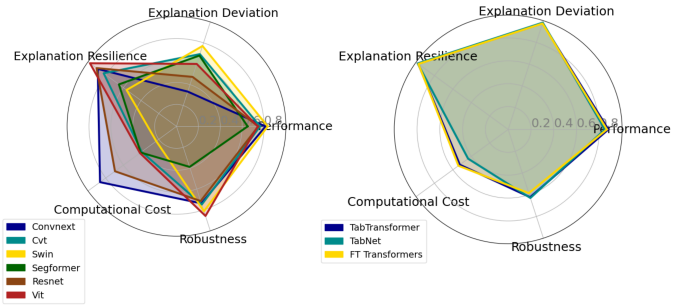

Fig. 11. Comprehensive Overview of Multiple Quality Attributes Assessment for the Selected Models.

Response to RQ1: The evaluation results show variability in the explainability metrics of XAI methods across various models. The scenario studies encompassed various models, ranging from CNN-based to Transformer-based and from vision to tabular data. When combined with various models, XAI methods offer diverse explanation utilities, as detailed in Figure 10. This variability highlights the importance of assessing compatibility between models and XAI methods. This study introduces a cloud-based XAI service that automates the evaluation pipeline, thereby streamlining the assessment of explainability metrics across diverse models and methods. By providing ease of customization, this service enables microservice to be reusable and the pipeline reproducible.

Response to RQ2: The investigation into the computational costs of XAI shows a notable burden, especially when compared to the costs for model inferences. The experimental XAI pipelines, as summarized in Table VII, show a correlation between a model's explanation deviation and its computational demands. The Swin Transformer model, offering the highest model performance and explanation deviation, also incurs the highest computational costs, as indicated by processing times and energy consumption metrics. This correlation showcases a trade-off between achieving desirable levels of explanation deviation and the associated computational costs, suggesting the need to evaluate the model-XAI combination and optimize this balance.

TABLE VII
EXPLANATION DEVIATION AND ENERGY CONSUMPTION OF THE SELECTED MODELS

Models(Vision/Tabular)	Explanation Deviation ↓	Energy (Wh) ↓
Swin	0.770	10.07
CVT	0.692	5.86
SegFormer	0.678	5.73
ViT	0.598	5.58
ResNet	0.474	3.32
ConvNeXt	0.333	2.67
TabNet	0.983	0.56
TabTransformer	0.977	0.49
FT Transformers	0.974	0.47

Response to RQ3: The experiments demonstrate that adversarial perturbations significantly affect both model performance and explanations, as detailed in Table VIII.

TABLE VIII
MODEL PERFORMANCE AND EXPLANATION DEVIATION CHANGES WITH ADVERSARIAL ATTACKS, RESULTS FROM 108 XAI PIPELINES

Attribute	Significant ($p > 0.05$)	Non significant ($p \leq 0.05$)
Performance	88.89%	11.11%
Deviation	69.44%	30.56%

In addition to p-value, we employ Cliff's Delta analysis [66] to compare the impact of adversarial attacks. A Cliff's Delta of 0.129 for model performance suggests models are more accurate without adversarial attacks. A Cliff's Delta of 0.428 for explanation deviation indicates greater accuracy of XAI results without adversarial attacks. Our cloud-based XAI service offers an automated framework for assessing model-XAI combinations and acquiring quality attributes via executing designed pipelines.

VI. CONCLUSION

This study proposes an XAI service framework designed to streamline operational complexities in XAI evaluation. The framework provides API and SDK interfaces for operation and enables deployment on cloud platforms. It facilitates data preprocessing, encompassing the processing of datasets, data transformations, and the application of adversarial perturbations. The service encapsulates AI models and XAI methods, enabling flexible combinations arranged by the task configuration. We develop and implement evaluation pipelines to assess five key quality attributes: computational cost, performance, robustness, XAI deviation, and XAI resilience.

The findings lead us to conclude with three research questions. First, we observed the variability in metrics results across models and XAI methods. This shows the necessity of evaluation before selecting the XAI method for AI models. This ensures effective explanations are provided to stakeholders. Second, our analysis summarizes the results of computational cost and the explanation metric. High explanation deviation often requires the expense of increased computational resources. Finally, we demonstrate that adversarial perturbations affect both the model and the XAI, thereby emphasizing the importance of incorporating robust model and XAI methods. These multidimensional quality attributes guide researchers and practitioners in making informed decisions in AI-based software development and deployment.

REFERENCES

[1] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz *et al.*, "Huggingface's transformers: State-of-the-art natural language processing," *arXiv preprint arXiv:1910.03771*, 2019.

[2] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[3] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[4] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 10 012–10 022.

[5] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo, "Segformer: Simple and efficient design for semantic segmentation with transformers," *Advances in Neural Information Processing Systems*, vol. 34, pp. 12 077–12 090, 2021.

[6] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie, "A convnet for the 2020s," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 11 976–11 986.

[7] H. Wu, B. Xiao, N. Codella, M. Liu, X. Dai, L. Yuan, and L. Zhang, "Cvt: Introducing convolutions to vision transformers," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 22–31.

[8] G. X. Yu, Y. Gao, P. Golikov, and G. Pekhimenko, "Habitat: A runtime-based computational performance predictor for deep neural network training," in *USENIX Annual Technical Conference*, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:236992542

[9] F. Doshi-Velez and B. Kim, "Towards a rigorous science of interpretable machine learning," *arXiv preprint arXiv:1702.08608*, 2017.

[10] P. Lopes, E. Silva, C. Braga, T. Oliveira, and L. Rosado, "Xai systems evaluation: A review of human and computer-centred methods," *Applied Sciences*, vol. 12, no. 19, p. 9423, 2022.

[11] D. Gunning and D. Aha, "Darpa's explainable artificial intelligence (xai) program," *AI magazine*, vol. 40, no. 2, pp. 44–58, 2019.

[12] A. B. Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. García, S. Gil-López, D. Molina, R. Benjamins *et al.*, "Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai," *Information fusion*, vol. 58, pp. 82–115, 2020.

[13] B. H. Van der Velden, H. J. Kuijf, K. G. Gilhuijs, and M. A. Viergever, "Explainable artificial intelligence (xai) in deep learning-based medical image analysis," *Medical Image Analysis*, vol. 79, p. 102470, 2022.

[14] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, "Universal adversarial perturbations," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1765–1773.

[15] D. Hendrycks and T. Dietterich, "Benchmarking neural network robustness to common corruptions and perturbations," *arXiv preprint arXiv:1903.12261*, 2019.

[16] J. Zhang and C. Li, "Adversarial examples: Opportunities and challenges," *IEEE transactions on neural networks and learning systems*, vol. 31, no. 7, pp. 2578–2593, 2019.

[17] D. Justus, J. Brennan, S. Bonner, and A. S. McGough, "Predicting the computational cost of deep learning models," in *2018 IEEE international conference on big data (Big Data)*. IEEE, 2018, pp. 3873–3882.

[18] J. Huang, Z. Wang, D. Li, and Y. Liu, "The analysis and development of an xai process on feature contribution explanation," in *2022 IEEE International Conference on Big Data (Big Data)*, 2022, pp. 5039–5048.

[19] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 618–626.

[20] A. Chattopadhay, A. Sarkar, P. Howlader, and V. N. Balasubramanian, "Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks," in *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2018, pp. 839–847.

[21] R. L. Draelos and L. Carin, "Hirescam: Faithful location representation in visual attention for explainable 3d medical image classification," *arXiv preprint arXiv:2011.08891*, 2020.

[22] R. Fu, Q. Hu, X. Dong, Y. Guo, Y. Gao, and B. Li, "Axiom-based grad-cam: Towards accurate visualization and explanation of cnns," *arXiv preprint arXiv:2008.02312*, 2020.

[23] P.-T. Jiang, C.-B. Zhang, Q. Hou, M.-M. Cheng, and Y. Wei, "Layercam: Exploring hierarchical class activation maps for localization," *IEEE Transactions on Image Processing*, vol. 30, pp. 5875–5888, 2021.

[24] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. Curran Associates, Inc., 2017.

[25] K. Abhishek and D. Kamath, "Attribution-based xai methods in computer vision: A review," *arXiv preprint arXiv:2211.14736*, 2022.

[26] D. Li, Y. Liu, J. Huang, and Z. Wang, "A trustworthy view on explainable artificial intelligence method evaluation," *Computer*, vol. 56, no. 4, pp. 50–60, 2023.

[27] B. S. and R. Nagapadma, "RT-IoT2022 ," UCI Machine Learning Repository, 2024, DOI: https://doi.org/10.24432/C5P338.

[28] V. Arya, R. K. Bellamy, P.-Y. Chen, A. Dhurandhar, M. Hind, S. C. Hoffman, S. Houde, Q. V. Liao, R. Luss, A. Mojsilović *et al.*, "One explanation does not fit all: A toolkit and taxonomy of ai explainability techniques," *arXiv preprint arXiv:1909.03012*, 2019.

[29] H. Nori, S. Jenkins, P. Koch, and R. Caruana, "Interpretml: A unified framework for machine learning interpretability," *arXiv preprint arXiv:1909.09223*, 2019.

[30] W. Yang, H. Le, S. Savarese, and S. C. Hoi, "Omnixai: A library for explainable ai," *arXiv preprint arXiv:2206.01612*, 2022.

[31] S. Palacio, A. Lucieri, M. Munir, S. Ahmed, J. Hees, and A. Dengel, "Xai handbook: towards a unified framework for explainable ai," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 3766–3775.

[32] X.-H. Li, Y. Shi, H. Li, W. Bai, Y. Song, C. C. Cao, and L. Chen, "Quantitative evaluations on saliency methods: An experimental study," *arXiv preprint arXiv:2012.15616*, 2020.

[33] Y. Zhang, F. Xu, J. Zou, O. L. Petrosian, and K. V. Krinkin, "Xai evaluation: Evaluating black-box model explanations for prediction," in *2021 II International Conference on Neural Networks and Neurotechnologies (NeuroNT)*, 2021, pp. 13–16.

[34] C. Xiao, B. Li, J.-Y. Zhu, W. He, M. Liu, and D. Song, "Generating adversarial examples with adversarial networks," *arXiv preprint arXiv:1801.02610*, 2018.

[35] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.

[36] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *2017 ieee symposium on security and privacy (sp)*. Ieee, 2017, pp. 39–57.

[37] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "The limitations of deep learning in adversarial settings," in *2016 IEEE European symposium on security and privacy (EuroS&P)*. IEEE, 2016, pp. 372–387.

[38] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "Deepfool: A simple and accurate method to fool deep neural networks," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 2574–2582.

[39] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, "Universal adversarial perturbations," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1765–1773.

[40] M. Zhou, J. Wu, Y. Liu, S. Liu, and C. Zhu, "Dast: Data-free substitute training for adversarial attacks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 234–243.

[41] M. Cisse, Y. Adi, N. Neverova, and J. Keshet, "Houdini: Fooling deep structured prediction models," *arXiv preprint arXiv:1707.05373*, 2017.

[42] P.-Y. Chen, H. Zhang, Y. Sharma, J. Yi, and C.-J. Hsieh, "Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models," in *Proceedings of the 10th ACM workshop on artificial intelligence and security*, 2017, pp. 15–26.

[43] J. Su, D. V. Vargas, and K. Sakurai, "One pixel attack for fooling deep neural networks," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 5, pp. 828–841, 2019.

[44] D. Hendrycks and T. Dietterich, "Benchmarking neural network robustness to common corruptions and perturbations," *arXiv preprint arXiv:1903.12261*, 2019.

[45] D. Slack, S. Hilgard, E. Jia, S. Singh, and H. Lakkaraju, "Fooling lime and shap: Adversarial attacks on post hoc explanation methods," in

Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society, 2020, pp. 180–186.

[46] M. T. Ribeiro, S. Singh, and C. 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*, 2016, pp. 1135–1144.

[47] M. Felderer and R. Ramler, "Quality assurance for ai-based systems: overview and challenges," *arXiv preprint arXiv:2102.05351*, 2021.

[48] Z. Kurtanović and W. Maalej, "Automatically classifying functional and non-functional requirements using supervised machine learning," in *2017 IEEE 25th International Requirements Engineering Conference (RE)*. Ieee, 2017, pp. 490–495.

[49] K. Naik and P. Tripathy, *Software testing and quality assurance: theory and practice*. John Wiley & Sons, 2011.

[50] S. Amershi, A. Begel, C. Bird, R. DeLine, H. Gall, E. Kamar, N. Nagappan, B. Nushi, and T. Zimmermann, "Software engineering for machine learning: A case study," in *2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*, 2019, pp. 291–300.

[51] L. Budach, M. Feuerpfeil, N. Ihde, A. Nathansen, N. Noack, H. Patzlaff, F. Naumann, and H. Harmouch, "The effects of data quality on machine learning performance," *arXiv preprint arXiv:2207.14529*, 2022.

[52] Z. Wang, Y. Liu, A. Arumugam Thiruselvi, and A. Hamou-Lhadj, "Xaiport: A service framework for the early adoption of xai in ai model development," in *Proceedings of the 46th International Conference on Software Engineering (ICSE 2024), New Ideas and Emerging Results*. Lisbon, Portugal: ACM, April 2024.

[53] A. Lacoste, A. Luccioni, V. Schmidt, and T. Dandres, "Quantifying the carbon emissions of machine learning," *arXiv preprint arXiv:1910.09700*, 2019.

[54] P. Cremonesi, Y. Koren, and R. Turrin, "Performance of recommender algorithms on top-n recommendation tasks," in *Proceedings of the fourth ACM conference on Recommender systems*, 2010, pp. 39–46.

[55] C. D. Manning, *An introduction to information retrieval*. Cambridge university press, 2009.

[56] J. N. Mandrekar, "Receiver operating characteristic curve in diagnostic test assessment," *Journal of Thoracic Oncology*, vol. 5, no. 9, pp. 1315–1316, 2010.

[57] J. H. Drew, A. G. Glen, and L. M. Leemis, "Computing the cumulative distribution function of the kolmogorov–smirnov statistic," *Computational statistics & data analysis*, vol. 34, no. 1, pp. 1–15, 2000.

[58] N. Sobahi, O. Atila, E. Deniz, A. Sengur, and U. R. Acharya, "Explainable covid-19 detection using fractal dimension and vision transformer with grad-cam on cough sounds," *Biocybernetics and Biomedical Engineering*, vol. 42, no. 3, pp. 1066–1080, 2022.

[59] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255.

[60] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," *arXiv preprint arXiv:1312.6199*, 2013.

[61] J. Larson, S. Mattu, L. Kirchner, and J. Angwin, "How we analyzed the compas recidivism algorithm," *ProPublica (5 2016)*, vol. 9, no. 1, pp. 3–3, 2016.

[62] L. Akritidis, "Product Classification and Clustering," UCI Machine Learning Repository, 2023, DOI: https://doi.org/10.24432/C5M91Z.

[63] X. Huang, A. Khetan, M. Cvitkovic, and Z. Karnin, "Tabtransformer: Tabular data modeling using contextual embeddings," *arXiv preprint arXiv:2012.06678*, 2020.

[64] S. Ö. Arik and T. Pfister, "Tabnet: Attentive interpretable tabular learning," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 35, no. 8, 2021, pp. 6679–6687.

[65] Y. Gorishniy, I. Rubachev, V. Khrulkov, and A. Babenko, "Revisiting deep learning models for tabular data," *Advances in Neural Information Processing Systems*, vol. 34, pp. 18 932–18 943, 2021.

[66] N. Cliff, *Ordinal methods for behavioral data analysis*. Psychology Press, 2014.