Model-Agnostic Scoring Methods for Artificial Intelligence Assurance

Md Nazmul Kabir Sikder
Department of Electrical and Computer Engineering (ECE)
Virginia Tech
Arlington, VA
nazmulkabir@vt.edu

Feras A. Batarseh (IEEE Senior Member)
Department of Biological Systems Engineering
Virginia Tech
Arlington, VA
batarseh@vt.edu

Pei Wang
Department of Computer Science (CS)
Virginia Tech
Arlington, VA
pwang1@vt.edu

Nitish Gorentala
Department of CS
Virginia Tech
Blacksburg, VA
nitishg@vt.edu

***Abstract*—State of the art Artificial Intelligence Assurance (AIA) methods validate AI systems based on predefined goals and standards, are applied within a given domain, and are designed for a specific AI algorithm. Existing works do not provide information on assuring subjective AI goals such as fairness and trustworthiness. Other assurance goals are frequently required in an intelligent deployment, including explainability, safety, and security. Accordingly, issues such as value loading, generalization, context, and scalability arise; however, achieving multiple assurance goals without major trade-offs is generally deemed an unattainable task. In this manuscript, we present two AIA pipelines that are model-agnostic, independent of the domain (such as: healthcare, energy, banking), and provide scores for AIA goals including explainability, safety, and security. The two pipelines: Adversarial Logging Scoring Pipeline (ALSP) and Requirements Feedback Scoring Pipeline (RFSP) are scalable and tested with multiple use cases, such as a water distribution network and a telecommunications network, to illustrate their benefits. ALSP optimizes models using a game theory approach and it also logs and scores the actions of an AI model to detect adversarial inputs, and assures the datasets used for training. RFSP identifies the best hyper-parameters using a Bayesian approach and provides assurance scores for subjective goals such as ethical AI using user inputs and statistical assurance measures. Each pipeline has three algorithms that enforce the final assurance scores and other outcomes. Unlike ALSP (which is a parallel process), RFSP is user-driven and its actions are sequential. Data are collected for experimentation; the results of both pipelines are presented and contrasted.**

***Index Terms*—AI Assurance, Deep Learning, Equilibrium, Optimization, Scoring Methods**

I. Introduction

In a recent review paper on AIA [1], assurance is defined as: *"a process that is applied at all stages of the AI engineering lifecycle ensuring that any intelligent system is producing outcomes that are valid, verified, data-driven, trustworthy, and explainable to a layman, ethical in the context of its deployment, unbiased in its learning, and fair to its users"*. AIA aims to define empirical ways of evaluating subjective measures that are commonly domain-dependent such as fairness and explainability. For instance, an AI model that is used for recruiting might contain some bias towards certain candidates, but that bias is very difficult to quantify. Being able to measure fairness would help assure that the model is fit for use within organizational and legal constraints. Another example where AIA would be critical is for example, in hospitals. Trusting an AI model in making subtle decisions that might lead to health-related consequences is a process that requires high trust in the model. Through AIA models presented in this paper, one can measure the trustworthiness of an AI model. However, representing such subjective measures in a quantified manner is unquestionably complicated. AIA goals can be achieved by either a model-specific or model-agnostic approach. A model-specific approach manages a domain-specific AI algorithm such as assurance of fairness-aware outlier detection [2], whereas a model-agnostic approach is a generic and universal approach that facilitates verifying AI algorithms irrespective of the domain of study. In the algorithms presented in this manuscript, we can be able to provide unique quantifiable scores for six AIA goals [1] including Explainable AI (XAI), Trustworthy AI (TAI), Fair AI (FAI), Ethical AI (EAI), Secure AI (CAI), and Safe AI (SAI). The six goals are, however, quasi-mutually exclusive, trade-offs are often enforced when choosing amongst them. In literature, it has been highly arguable whether to make an AI model highly explainable and less safe for instance.

The trade-off between goals depends on the requirements of each specific application. It is a subjective decision whether to compromise a model's safety to achieve other assurance goals (such as safety). Regardless of the challenges, considering few trade-offs, we provide tools to AI engineers to manage the six goals of assurance in this manuscript. Accordingly, we present Model Agnostic Assurance (MAA) for domain-independent applications through two pipelines: ALSP and RFSP (Fig. 1 and Fig. 2). ALPS includes three algorithms: Weight Assessment, Reverse Learning, and Secret Inversion. RFSP is a user-driven framework where users provide their expected AIA weights as inputs to dictate outcome of the system. The final outcome of the RFSP pipeline includes quantifiable AIA scores; RFSP involves three working algorithms: Economic Equilibrium, Extreme Data Segmentation, and Model Optimization. Unlike Weight Assessment, Reverse Learning, and Secret Inversion, these algorithms are executed sequentially yielding scores. The major difference between these two pipelines is that one is user-driven (RFSP) i.e. requires user input, whereas the other is not.

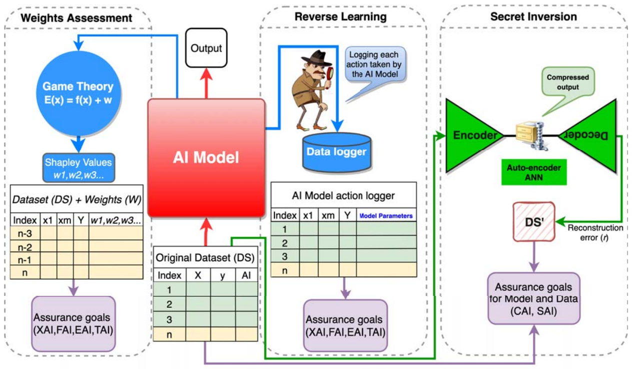

Weights Assessment
Game Theory E(x) = f(x) + w
Shapley Values w1,w2,w3...
Dataset (DS) + Weights (W)
Index x1 xm Y w1,w2,w3...
n-3
n-2
n-1
n
Assurance goals (XAI,FAI,EAI,TAI)
Output
AI Model
Original Dataset (DS)
Index X y AI
1
2
3
n
Reverse Learning
Logging each action taken by the AI Model
Data logger
AI Model action logger
Index x1 xm Y Model Parameters
Assurance goals (XAI,FAI,EAI,TAI)
Secret Inversion
Compressed output
Encoder
Decoder
Auto-encoder ANN
DS'
Reconstruction error (r)
Assurance goals for Model and Data (CAI, SAI)

Fig. 1. ALSP

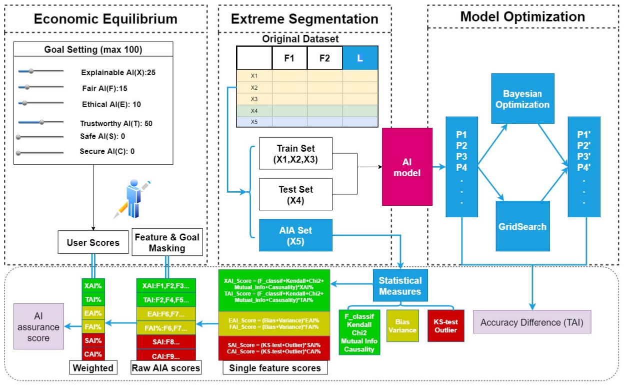


Fig. 2. RFSP

The rest of the paper is structured as follows: The next section presented related studies in AIA. Afterwards, we present both pipelines, along with the design process and reasoning for each algorithm. Experiments are presented in section 4, section 5 presents the results as well as example use cases. Lastly, section 6 presents conclusions and potential future works related to the MAA pipelines.

II. Related Work

A model-agnostic approach explains a model in a post-hoc fashion (hindsight) by accepting that the model is as a black-box; with the inner workings of the model hidden from sight, and then attempting to approximate its behavior [3]. One such example of a model-agnostic approach is Local Interpretable Model-agnostic Explanations (LIME) [4]. This approach attempts to provide local explanations in the form of linear approximations of the model, accurate in small regions of the space. It is also practical when explaining, for example, why a particular individual has been denied a mortgage application. Other post-hoc model-agnostic algorithms provide explanations in ranking features, even when an underlying model is not linear. This includes InterpretML [5], SHapley Additive exPlanations (SHAP) [6], and Partial Dependence Plots (PDP) [7]. Each of these algorithms takes a distinct approach to the process of determining the important contributors to an ML model. Albeit MAA is rare, there is no scarcity of model-specific assurance, we present a brief review by AIA goal, with the most related study found in the literature for each.

EAI: In a study conducted by [8] on EAI, the researchers employ a combination of symbolic and logic-based approaches, a data-driven approach, as well as consideration of a rule-based and data-driven hybrid approach. This last approach is one they propose with the question of how we determine when preference breeches ethics as a breach in ethics can make the model not viable to use. We employ a system that weighs each assurance goal in each model to attempt to combat this issue.

FAI: Another work by [9] investigates how biased and unbiased data can both result in AI models that treat certain inputs unfairly when compared to others. They constructed a framework that helps to prevent discrimination between inputs in AI models. Our research looks not just at how we can assure outputs, but also inputs (the dataset) to aid in preventing such bias.

SAI and CAI: In work presented in [10], the researchers investigated SAI, measuring it using states and the reachability of what is defined to be a secure state. They use this algorithm to obtain a quantitative score for security. While these researchers used the reachability of the output of the AI model, we use various techniques such as an auto-encoder to detect outliers in the data and obtain reconstruction error to use to measure a security score.

TAI: In a study conducted by [11], the researchers constructed SUNNY, "a new algorithm for trust inference in social networks using probabilistic confidence models". This algorithm outperformed another trustworthiness measure relevant at the time in their testing. Our models differ in that they employ the measurement of feature contribution with game theory and Shapley [12] and causal values to obtain a score for trustworthiness.

XAI: In a study conducted by [13], the researchers employ various formulas involving cognitive chunks to determine a quantitative score for XAI. They mention that "explainable decisions from commercial AI systems are going to be a standard imposed by regulators to eliminate bias and discrimination, and ensure trust". XAI is undoubtedly the most studied [1] out of AIA goals. The next section presents both pipelines and how they address the six goals.

III. AI Assurance Pipelines

A. Pipeline 1: ALSP

ALSP is a pipeline that validates an AI system for generating quantifiable AIA scores using a combination of both data-driven and AI model-driven approaches. More specifically, ALSP optimizes models using a game theory approach and it also logs and scores the actions of an AI model to detect adversarial inputs, and assures the datasets used for training. It is quite difficult to assure all six goals for an AI system using a single algorithm, therefore we propose three separate algorithms in ALSP pipeline including Weight Assessment, Reverse Learning, and Secret Inversion that, are capable of achieving all six goals combined for different contextual applications.

1) Weight Assessment: The Weight Assessment is an algorithm that applies Game Theory to the AI model of particular interest for calculating the Shapley values [12] at every epoch of learning; this algorithm aims to achieve assurance goals including XAI, FAI, and TAI as a form of quantifiable AIA scores by assigning scores per data point and not as an aggregate score. It is important to note that, we assume an AI model for scoring the given dataset. For Weight Assessment study, we apply XGBDT as our baseline model. Shapley values are outcomes of a *game* that assumes cooperation among players and achieves overall gain from alliances. These values represent each player's bargaining power and the payoff that is reasonable to expect in a given context. The alliance in the game can be represented by a characteristic function. This characteristic function (v) can be mapped as $v : 2^N \to R$ for a set of N; therefore, each player i gets a *fair* distribution, assuming the game is cooperative and can be represented mathematically as (1). Considering a fair cooperation game, a player can expect $v(S \cup \{i\}) - v(S)$ by averaging the set of possible different permutations in which the alliance was formed.

$$\varphi_i(v) = \frac{1}{n} \sum_{S \subseteq N \setminus \{i\}} \left(\begin{array}{c} n-1 \\ |S| \end{array} \right)^{-1} (v(S \cup \{i\}) - v(S)) \quad (1)$$

A unique set of values that indicates the importance of each feature in a given dataset is assigned. Along with the game theory-driven weights, these values also represent a heuristic expectation from an assurance perspective that is provided by domain experts. Similar to how labels are used as independent variables in a training-testing learning approach, here, we add assurance labels, namely: AIA Columns (AIAC). However, AIAC values, as assigned by the domain expert, are not directly fed to the AI model as inputs. For instance, some features in a dataset can be relevant to specific assurance goals such as fairness (consider data on gender or on race), accordingly, these features are labeled as FAI features for instance. The use cases provided in this manuscript provide further information on the usability of these labels.

```
Algorithm 1: Weight Assessment
  Inputs   : Dataset ($X$) and AI Model ($f(x)$: XGBDT)
  Execute  : AIA scores (XAI, FAI, TAI, EAI)
1 Train and test the AI model ($f(x)$:XGBDT) with the given dataset ($X$)
2 Apply Game Theory to evaluate the expected values for each feature $\varphi_i(v)$ from the trained AI model using (1)
3 Measure the AIA scores = $\varphi_i(v) * (AIAC_i)$
```

Algorithm 1 represents steps of the Weight Assessment technique. To generate AIA scores, we matrix-multiply Shapley value weights $\varphi_i(v)$ and AIACs $(AIAC_i)$. This matrix multiplication generates the AIA score for each row/column. Experimental work provides further information on that.

2) Reverse Learning: Reverse Learning is a log-based algorithm (Algorithm 2) that can trace back assurance issues using a table of recorded learning actions (i.e. reverse engineering). Reverse Learning accomplishes AIA goals such as XAI. XAI can be enforced if learning details and evolution is available through a log of actions. The *log* records the learning process and indicates to points in time during learning where the algorithm's learning accuracy -for instance- has decreased or seized to improve. That is illustrated in the experimental section. While designing the AI model, we use the Gradient Boosting Decision Tree (GBDT) model as an example. Though we select GBDT as our AI model of choice, for experimentation purposes, other AI algorithms can be applied as well. While developing the AI algorithm, the primary focus was to log the actions of each epoch of learning. The outcome of this algorithm is two-fold: the optimized number of epochs to minimizing the loss function and a logged action of each epoch. For GBDT, values including pseudo-residuals r_{im}, gamma γ_{im}, log of odds for the labels $l_m(x)$, probability p_{mi} are saved during each epoch. Equation 2 presents a prediction of GBDT model after each epoch. Equation 3 presents the logarithmic loss function of GBDT model.

$$f_{mi} = \Big\{0 \to p_{mi} < 0.5, 1 \to p_{mi} >= 0.5 \quad (2)$$

$$L = -\sum_{i=1}^{N} \Big(y \log(odds) - \log\Big(1 + e^{\log(odds)}\Big)\Big) \quad (3)$$

Here, f_{mi} is the prediction of the GBDT model, L is the loss function in terms of logs of each odd epoch of the GBDT model.

Unlike Weight Assessment, Reverse Learning doesn't provide AIA scores, however it serves as a tool to manually verify and optimize the AI algorithm.

Algorithm 2: Reverse Learning

```
Inputs   : Dataset ($X$) and AI Model ($f(x)$: GBDT)
Execute  : Log Model action, Optimize loss function
1 Train and test the AI model ($f(x)$: GBDT) with the given
    dataset ($X$)
2 Log actions (For GBDT: pseudo-residuals, gamma, log of
    odds for the labels, probability) of each epoch in a
    collection of data frame
3 Minimize loss function, for GBDT
    $L = -\sum_{i=1}^{N}\left(y\log(odds) - \log\left(1+e^{\log(odds)}\right)\right)$
```

Algorithm 3: Secret Inversion

```
Inputs   : Dataset ($X$), number of epochs ($e$), batch size
           ($b$), learning rate ($l$), compression ratio ($cf$)
Execute  : CAI, SAI scores
1 Load $X$, $e$, $b$, $l$, $c$
2 Train the AE with a few sample of the dataset ($X$)
3 Loop: Apply SGD for updating connection weight and define
    $\phi$ (4) and $\psi$ (5) to minimize reconstruction error
4 Select a proper threshold $\theta$. By selecting threshold ($\theta$), SAI
    and CAI scores = Reconstruction errors ($r$)
```

3) Secret Inversion: The Secret Inversion algorithm (Algorithm 3) performs exhaustive comparisons amongst features by reconstructing them using an *Autoencoder*. We assume that, given the reconstruction errors (r) work using an encoder-decoder mechanism, the AIA scores that are relevant in this case are goals such SAI and CAI. An Autoencoder learns a meaningful pattern of the data model by reducing its feature dimensions. It consists of a feed-forward neural network and works as a self-supervised model where the encoder-decoder can be characterized as an hourglass shape compressor and decompressor. The encoder compresses feature space and translates it into codes that are decompressed by the decoder. The decoder reconstructs the feature space from the codes. However, the reconstructed signal is not always the same as the input (if its not the same it indicates an alteration in the data - which could be a SAI/CAI issue). This difference is represented by the reconstruction errors (r). Let, ϕ and ψ as the encoder and decoder respectively, mathematically an AE can be represented as,

$$\phi : \mathcal{X} \rightarrow \mathcal{F} \tag{4}$$

$$\psi : \mathcal{F} \rightarrow \mathcal{X} \tag{5}$$

Here, $X \in \mathcal{X}$ is input for the encoder and $F \in \mathcal{F}$ represents the code as the input of the decoder. By defining ϕ and ψ, the AE minimizes the reconstruction errors and measure connection weights across the training phase for all elements of input X. It can be defined mathematically as,

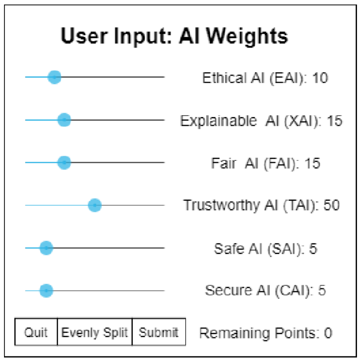
$$\|X - (\phi \circ \psi)X\|^2 \tag{6}$$

Where $\circ$ is a composition operator. The reconstruction error can be minimized using either Adam Optimizer or Stochastic Gradient Descent (SGD) algorithm. Both optimizer is capable of quickly updating the connection weights after a few learning cycles.

B. Pipeline 2: RFSP

RFSP is a framework that optimizes and generates scores for AI systems by using user input as basis. It identifies the best hyper-parameters for a model using a Bayesian approach and provides scores for AIA goals based on user inputs through a user interface. RFSP creates statistical assurance measures that can contrast with what the AI user (i.e. layman or domain expert) requires and what the system actually provides. RFSP has three algorithms, presented next.

1) Economic Equilibrium: Economic Equilibrium is an algorithm that provides the optimal *trade-off* between all assurance goals (like equilibriums in economics) using supply/demand and price elasticity foundations. This algorithm provides a graphical user interface for adjusting expected assurance weights up-to 100 points. However, it does not allow the sum of all goals to exceed a total of 100 points, since we assume that 100 is the maximum amount of weights combined that can be assigned for all six AIA goals. For instance, an AI system that is within the context of a bank may has less need for safety than one flying a commercial plane. All points must be allocated and the weights of all six AI goals are transferred to the pipeline for generating quantifiable scores. Fig. 3 shows the graphical user interface used for providing user expectations. After selecting the expected AIA weights, they can be submitted using the “Submit” button. If the assignment of weights is a total of 100 then the values are passed to the next algorithm for other statistical measures.


User Input: AI Weights

Ethical AI (EAI): 10
Explainable AI (XAI): 15
Fair AI (FAI): 15
Trustworthy AI (TAI): 50
Safe AI (SAI): 5
Secure AI (CAI): 5

Remaining Points: 0

Fig. 3. User interface for providing inputs for AIA’s equilibrium

2) Extreme Data Segmentation: Extreme Data Segmentation is an algorithm that splits the whole dataset into three parts: the training set, the testing set, and the AIA set (in lieu of the traditional testing-training split). In RFSP, the training set and testing set are used for AI model development, while the AIA set is used for generating assurance scores. For the AIA set, a series of statistical measures are calculated of each feature (Table I), also the relations between statistical measures and AIA goals are presented. Chi-Squared, Kendall rank, Mutual Information, and ANOVA are measures used to analyze the significance of the type response (correctness) against the labels [14]. Additionally, the Kolmogorov–Smirnov test is used, it is a non-parametric test of the equality of continuous one-dimensional probability distributions [15]. Lastly, we measure bias and variance as additional measures to indicate to issues such as FAI in the outcomes. Finally, this algorithm generates a weighted average assurance score utilizing the input weights from Economic Equilibrium's interface selections. The final scores help to compare different states such as bias and variance of an AI system. In the experiment section, the design procedure of this algorithm is provided.

TABLE I
STATISTICAL MEASURES AND RELEVANCE TO AIA GOALS

Statistical Measure	**Implication**	**Limitation**	**Related AIA goals**
ANOVA-F	Linear Dependency	Num to Cate	XAI, TAI
Kendall	Nonlinear Dependency	Num to Cate	XAI, TAI
Mutual Info	Dependency	Agnostic to data	XAI, TAI
Chi2	Dependency	Cate to Cate	XAI, TAI
Ks-test	Distribution (Distance)	Num cols	SAI, CAI
Outlier	Percentage out of 3 std	Num cols	SAI, CAI
Bias	Prediction accuracy	All cols	FAI, EAI
Variance	Prediction consistency	All cols	FAI, EAI

3) Model Optimization: Model Optimization applies two different optimization techniques to allocate the best hyper-parameter values combination in a given deployment: Grid Search and Bayesian Optimization [16]. For instance, in a classification algorithm, Model Optimization compares the F1-scores reached by the model compared with F1-scores from the open-source *default* set of hyper-parameters, using (7).

$$x^\star = \arg\min_{x \in \mathcal{X}} f(x) \quad (7)$$

Where $f(x)$ is the objective function to minimize, $x^\star$ is the optimized hyper-parameter set that yields the best results of the objective function. The goal is to investigate an optimized hyper-parameter set that generates the best scores using a validation metric set.

IV. EXPERIMENTAL WORK

In this study, we design the experiments with both synthetic and real-world datasets. For ALSP, we use three datasets: Water distribution network, Pima Indian Diabetic, and Bank Loans. The water distribution network dataset is synthetic data [17] that are generated using an emulator; it represents a sensor network within a hydraulic system. It also represents the Supervisory Control and Data Acquisition (SCADA) monitoring system (a commonplace dataset for simulations). We select this data to emphasize our study for assuring AI models in critical contexts. The remaining datasets are collected from Kaggle and University of California Irvine (UCI) Machine Learning Repository; Pima Indian Diabetic Dataset [1] and Bank Loan[2]. The Pima Indians Diabetes dataset come from the National Institute of Diabetes and Digestive and Kidney Diseases. The dataset include data from Pima Indian heritage female patients who are at least 21 years old and classifies if a patient is diabetic or not based on a specific diagnostic condition. We also collect Cellular Carrier data from the public website: Kaggle[3], where the data are provided by China Unicom (a mobile operator), the data contain 25 features and more than 1 Million instances.

[1]http://archive.ics.uci.edu/ml

[2]https://www.kaggle.com/zaurbegiev/my-dataset

[3]https://www.kaggle.com/pwang001/user-package-information-of- mobile-operators

A. Testing ALSP

In this experiment, we test if the algorithm provides accurate AIA scores for each sample of a given dataset using Weight Assessment. For Reverse Learning, we test if the actions of an AI model (GBDT) are logged for each epoch and illustrate how they provide XAI outcomes. Finally, for Secret Inversion, we test if the algorithm can detect adversarial inputs from the SCADA dataset.

1) Weight Assessment- Scoring AI System: For this experiment, we calculate Shapley values that help generate AIA scores for each sample of the dataset, namely: weights (W) of each feature contributing towards the final outcome of our AI model. These weights and the AIA in the diabetic dataset are multiplied to generate scores for each observation. We select the Pima Indian Diabetic dataset (number of samples, N= 768 and number of features m = 8) for this study, where the label indicates if a patient is diabetic or not. Since there is a total of eight features in this dataset, accordingly, a total of new eight AIACs are added that represent feature expectations. Feature expectations dictate the assurance goals for the AI system, therefore they must be designed based on the application requirements, we label binary values for each AIAC. Additionally, for this experiment we select Extreme Gradient Boosting Decision Tree (XGBDT) for the AI model with the hyper-parameters set as follows: learning rate = 0.1, number of estimators = 1000, maximum depth = 5, minimum child weight = 1, gamma = 0, subsample = 0.8, colsample by tree = 0.8, objective = "binary:logistic", number of thread = 4, scale position weight = 1, and seed = 27.

The deployed AIACs represent TAI expectations. to test that, we inject bias and compared the scores with the unbiased dataset (Fig. 4). For injecting bias, we apply Gaussian noise (mean = 0.3 and standard deviation = 0.1) and present the difference between the biased and unbiased datasets using Gaussian distribution. Fig. 5 represents Gaussian score distribution difference for biased and unbiased datasets. It is evident from the Fig. 5 that intentional bias injection generates different AIA scores that helps to explain relevant changes in the AI system.

The scores are the expected outcome of the AI system; however, they do not necessarily mean anything until we deploy the AIACs properly in the dataset. We present that

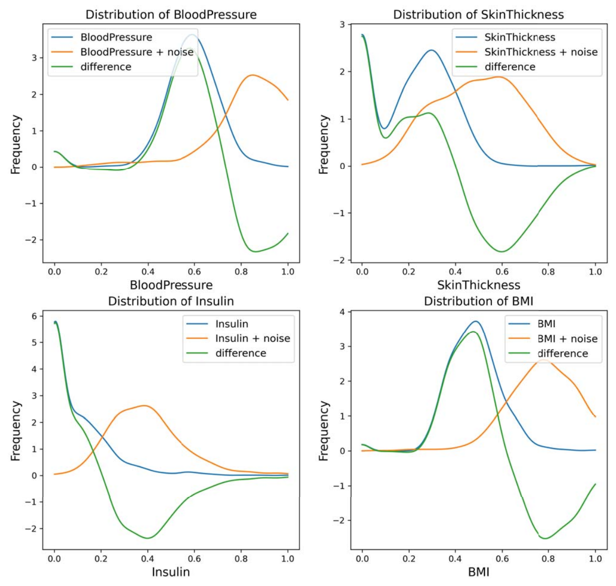


Fig. 4. Distribution - normal and biased datasets

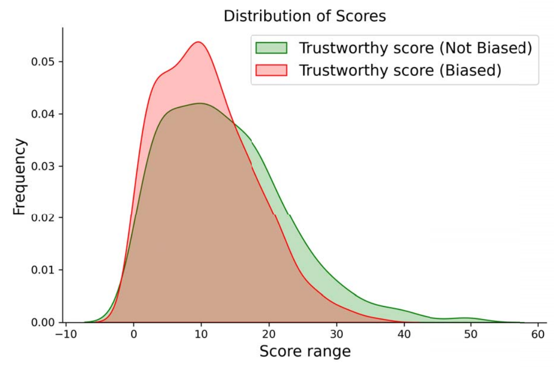


Fig. 5. Distribution of TAI scores - normal and biased datasets

Weight Assessment generates AIA scores for every observation of the dataset. Additionally, after altering the dataset, we get a meaningful score representation of that alteration. It is evident from Fig. 5 that the score distribution changed when we inject *minimal* bias into the dataset, which was captured by the pipeline. Accordingly, this helps in indicating whether the outcomes are deemed more trustworthy or not.

2) Reverse Learning- Log and Optimize: In this experiment, we incorporate GBDT as our main AI algorithm. For GBDT, the primary target is to log each action while minimizing the loss function, therefore we design the AI algorithm as a white box (didnt use any off-the-shelf library). Reverse Learning doesn't provide any AIA score but dictates AIA goals such as FAI and EAI by visualizing and using exhaustive explanations (Explanations mean checking all the calculations and comparing them with the logs). This algorithm returns statistics on each epoch.

We select the Bank Loan dataset (number of samples, N= 614 and number of features m = 12) where the label indicates if a customer is likely to be accepted or not for a loan application. The features of this dataset are Gender, Married Dependents, Education, Self Employed, Applicant Income, Co-applicant Income, Loan Amount, Loan Amount Term, Credit History, and Property Area. The GBDT model predicts the status of an applicant. Since our focus is minimizing loss function, we use default hyper-parameters including learning rate l = 0.1, max depth of the decision tree d = 4, and max leaf nodes nl = 7. For the training stage, we perform 50 epochs for training the GBDT model. From Fig. 6, it's evident that the loss minimizes during the *13th epoch* (Fig. 7); the idea here is that all epochs post 13 are obsolete, because the accuracy decreases afterwards - something that wouldn't be traceable or explainable otherwise except by using an overly simple line chart. Table II presents the first 15 observations within the 13th epoch where the rest of the observations (as well as all code and data used in this study) can be found in the MAA GitHub repository [4].

TABLE II
LOGS OF GBDT ALGORITHM LEARNING CYCLE (REVERSE LEARNING)

Row	$p(t)$	$l(t)$	r	γ	$l(t+1)$	$p(t+1)$
1	0.74	1.06	-0.74	0.36	1.09	0.75
2	0.79	1.32	0.21	0.31	1.35	0.79
3	0.74	1.06	0.25	-1.58	0.90	0.71
4	0.71	0.91	0.28	0.63	0.97	0.72
5	0.74	1.04	0.26	0.63	1.10	0.75
6	0.74	1.06	0.25	-1.58	0.90	0.71
7	0.26	-1.02	-0.26	-0.84	-1.11	0.24
8	0.74	1.04	0.26	-1.58	0.88	0.70
9	0.64	0.59	-0.64	0.63	0.66	0.65
10	0.78	1.32	0.21	-1.58	1.16	0.76
11	0.74	1.04	0.26	-1.58	0.88	0.70
12	0.65	0.65	-0.65	-1.58	0.49	0.62
13	0.78	1.30	0.21	0.01	1.30	0.78
14	0.71	0.91	0.28	0.63	0.97	0.72
15	0.26	-1.02	-0.26	-0.84	-1.11	0.24

3) Secret Inversion- Detection of Adversarial Data Points: For the Secret Inversion algorithm, we design the experiment to present the successful detection of adversarial inputs in an AI system. Final outcomes are scores for AIA goals including SAI and CAI (Table III). For AE, detection performance varies with compression factor (cf) and the number of hidden layers (nl). Since we use labeled data, we select the best hyper-parameter (approximately we build 20 AE models) set for maximum accuracy. The outcome is the status of the network, binary classification, which represents if the overall system is under attack or not. Additionally, instead of *sigmoidal* function we select rectified linear units due to their higher training performance in deep neural networks [18]. Here we use SCADA dataset to test secret inversion algorithm. Two-thirds of the training dataset is used for model development and the other one-third is used for validation purposes. During validation, we are able to perform early stopping for preventing

[4]https://github.com/AI-VTRC/MAA

model overfitting scenarios. We use Adam algorithm [19] for our AE model where we find the best trade-off between execution speed and convergence (number of training epochs e = 200 and mini-batch size b = 300 samples). The mini-batch size is important because it updates the connection weights by propagating into the AE's network and computing the gradient. The average reconstruction errors, which are the mean squared errors between input and reconstructed patterns, are optimized using the SDG technique. Fig. 8 represents two different test datasets where scaled reconstruction errors are plotted on the 1st test dataset. By properly selecting threshold (θ), normal and adversarial samples for all 43 features have separated. For this experiment, we select the range between 99% and 100% percentile of the error distribution as adversarial inputs where the rest of the ranges are considered normal samples. The detection accuracy of test datasets 1 and 2 are 94.66% and 91.28% respectively. Table III presents the performance evaluation of the AE model.

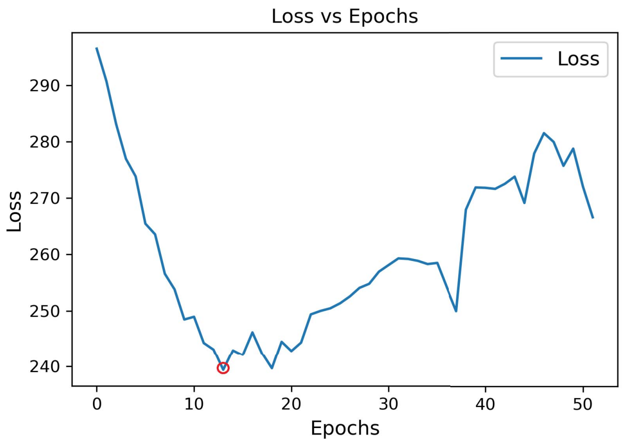


Fig. 6. Loss function vs learning epochs

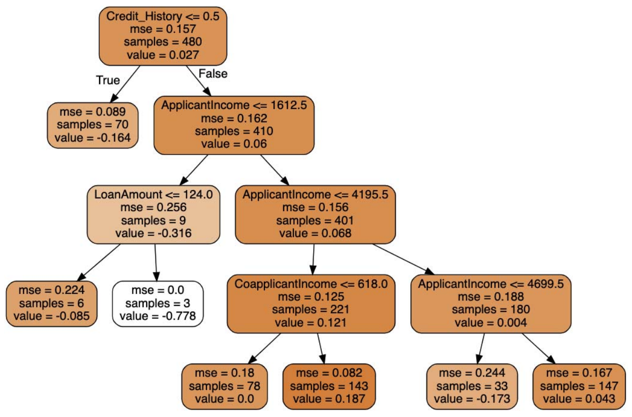


Fig. 7. Decision tree 13 - minimum loss epoch

B. *Testing RFSP*

In the experiment for RFSP, we test if the pipeline generates lower AIA scores due to bias insertion, and we optimize the

TABLE III
INTRUSION INPUT DETECTION PERFORMANCE USING AE

Test Dataset	Accuracy	F1_Score	Precision	Recall
01	0.9466	0.7194	0.9438	0.5813
02	0.9128	0.7182	0.9707	0.5700

TABLE IV
HYPER-PARAMETERS OF LIGHTGBM ON TELCO DATASET

Hyper-parameters	Default	Bayesian Optimization	Grid Search
Learning rate	0.1	0.05	0.07
Max depth	-1(unlimited)	14	9
Bagging fraction	1	0.8	1
Number of trees	100	396	100
Number of leaves	31	25	12

TABLE V
TAI F1 SCORES USING DIFFERENT MODELS

AI-Model	Telco	SCADA
Default	0.767	0.802
Grid Search	0.746	0.794
Bayesian Optimization	0.772	0.842

TABLE VI
TELCO FEATURES - AIA GOALS MAPPING

Features	XAI	TAI	SAI	CAI	FAI	EAI
service_type	1	1	0	0	0	0
is_mix_service	1	1	0	0	0	0
month_traffic	1	1	0	1	0	0
many_over_bill	1	1	0	0	0	0
contract_type	1	1	0	0	0	0
contract_time	1	1	0	0	0	0
pay_num	1	1	0	1	0	0
last_month_traffic	1	1	0	0	0	0
service1_caller_time	1	1	0	0	0	0
service2_caller_time	1	1	0	0	0	0
gender	0	0	0	1	1	1
age	0	0	0	1	1	1
complaint_level	0	0	0	1	1	0
former_complaint_num	0	0	0	1	1	0
former_complaint_fee	0	0	0	0	1	0

TABLE VII
CAI SCORES FOR DIFFERENT INTENTIONAL BIASES

Bias Injection (SCADA)	CAI Score (SCADA)	Bias Injection (Telco)	CAI Score (Telco)
3.33%	54.4	0.45%	16.7
2.67%	52.7	0.36%	16.8
2%	53.4	0.27%	17.0
1.34%	57.3	0.18%	17.2
0.67%	63.6	0.09%	17.7
0%	71.5	0%	17.8

TABLE VIII
TELCO FEATURES - STATISTICAL MEASURES AND AIA SCORES

Features	Causality	ANOVA-F	Mutual_info	Chi2	Kendall	Outlier	Bias	Variance	Distribution	XAI	TAI	SAI	CAI	FAI	EAI
service_type	-2.31	0	1.12	0.84	0	0	0.79	-0.2	0	0.44	0.44	0	0	0	0
complaint_level	0.66	0	-1.15	-0.62	0	0	1.7	-0.2	0	0	0	0	0.51	0.54	0
contract_type	0.05	0	0.43	2.33	0	0	0.48	-0.2	0	0.8	0.8	0	0	0	0
gender	-0.36	0	-1.08	-0.68	0	0	1.01	-0.2	0	0	0	0	0.51	0.67	0.67
net_service	-0.08	0	-1.1	-0.68	0	0	1.88	-0.2	0	0.27	0.27	0	0	0	0
age	0.23	-0.69	-0.57	0	-1.42	-1.26	-0.64	-0.2	-1.79	0	0	0	1	0.98	0.98
contract_time	0.32	1.92	0.56	0	-2.27	-2.02	-0.08	-0.2	0.81	0.54	0.54	0	0	0	0
complaint_fee	0.29	-1.05	-1.17	0	-0.5	-2.03	-0.59	4.9	1.94	0	0	0	0	0	0
complaint_num	0.1	-1	-1.16	0	0	1.12	0.67	-0.2	2.07	0	0	0	0	0.73	0
last_month_traffic	0.29	-0.32	0.62	0	-1.17	1.33	-0.71	-0.2	0.61	0.42	0.42	0	0	0	0
local_caller_time	0.29	-0.72	-0.75	0	0.43	0.38	-0.72	-0.2	0.2	0.4	0.4	0	0	0	0
month_traffic	0.29	-0.59	0.57	0	-0.57	-0.13	-0.72	-0.2	0.42	0.45	0.45	0	0.47	0	0
online_time	0.3	0.84	0.55	0	-1.11	0.79	-0.68	-0.2	-0.3	0.55	0.55	0	0.43	0	0
service1_caller_time	0.28	-1.03	-0.66	0	-0.18	0.7	-0.71	-0.2	0.44	0.3	0.3	0	0	0	0
service2_caller_time	0.3	0.16	0.68	0	0.63	-0.03	-0.72	-0.2	-0.13	0.69	0.69	0	0	0	0

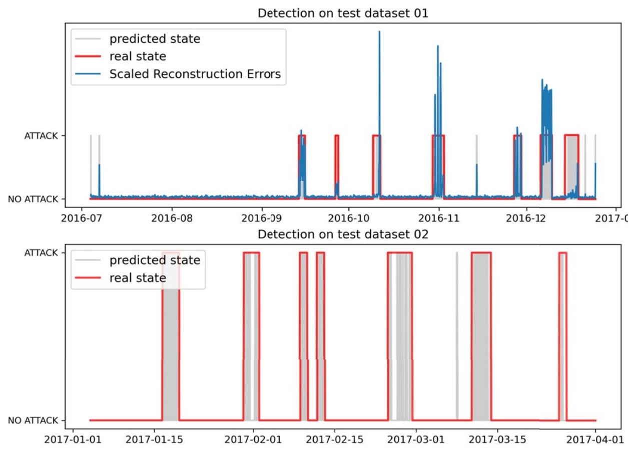


Fig. 8. Intrusion detection using r on test datasets 1 and 2

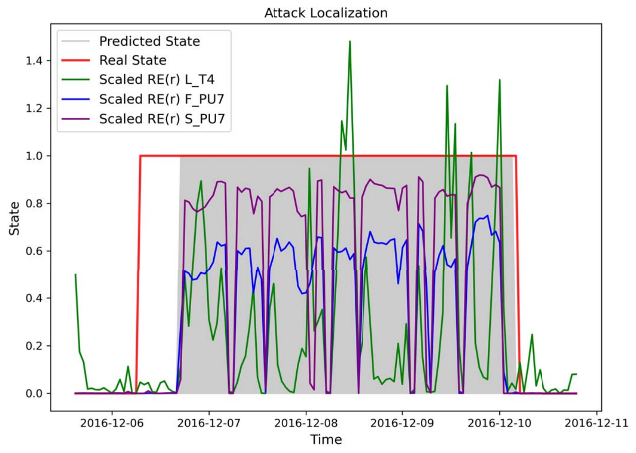


Fig. 9. Attack localization (XAI) using SAI scores

hyper-parameters to create higher assurance scores.

1) Bayesian Hyper-parameters Optimization: We apply Grid Search and Bayesian Optimization (Tree Parzen Estimator) for fine-tuning hyper-parameters of a LightGBM model using both the SCADA and the Telco datasets. The hyper-parameters are listed in Table IV. Afterwards, we compare the performance (F1-score) of two models to the default set of hyper-parameters:

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{8}$$

It is evident from the Table V, Bayesian Optimization accomplishes a higher performance level than the default set of hyper-parameters; and Grid Search arrived at a poorer performance level. Accordingly, in this case, Bayesian Optimization has higher efficiency in finding the best hyper-parameters using a Gaussian process other than an exhaustive Grid Search, which is expected. This higher efficiency allows Bayesian Optimization's support of a broader search-space of hyper-parameters, and avoids a local maximum, a notion that is more possible with Grid Search. Accordingly, we generate the TAI scores based on the difference of F1-scores reached by default hyper-parameters and the "Bayesian" hyper-parameters. In this case, the TAI score for Telco is $(-0.005+1 = 0.995)$, the TAI score for SCADA is $(-0.04 + 1 = 0.96)$. Both numbers are indicators of the level of trust in the outcomes of the systems, a notion that could be compared with the initial user inputs and their preferences.

In this experiment, we inject different levels of outlier data in order to change the distribution of the original SCADA and Telco datasets, followed by running RFSP to observe whether the pipeline could reflect the injection of outlier data. For the dataset with varying levels of outlier injections, we execute following workflow:

1) For Economic Equilibrium, we set the user score of 6 AIA goals evenly specifically *16.6* for each AIA goal.
2) We assign features to different AIA goals as shown in Table VI. The rest of the features (along with all project data and code) can be found in the GitHub repository [5].

[5]https://github.com/AI-VTRC/MAA

3) From Table VII it is evident that, with the level of outlier injection increasing, CAI score is decreased. However, after reaching a threshold, increasing the outlier injection, CAI might increase as well. This is because addition of more outliers leads to normalizing the data distribution.
4) Nine statistical measures are calculated of each feature as shown in Table VIII, the result is normalized using z-scores.
5) The AIA scores of each feature are calculated using the results of step 2 and step 3 along with the developed mechanism for converting statistical measures to the scores of 6 AIA goals; value=0 means the feature has no relation with definite AIA goals. Results are shown in Table VIII.
6) The average score is calculated for the 6 AIA goals, as follows: XAI:0.43, TAI:0.43, SAI:0, CAI:0.17, FAI:0.12, and EAI:0.05 based on different assigned features and their independent AIA goals scores. A weighted average of overall AIA goals could also be calculated using the results of step 1 (Weights) and step 4 (Statistical Measures), which is 0.195 (on a scale of 0 to 1).

V. Experimental Use Cases

Earlier, we discuss how ALSP and RFSP can provide pointers to achieving quantifiable assurance goals. However, without actual use cases, the pipelines don't illustrate their real-world purposes. In this section, we provide one use case for ALSP using SCADA and one for RFSP using gender bias in Telco plans selection. SCADA data represent a critical infrastructure, for that we explore AIA goals XAI, CAI, and SAI. Blue lines in Fig. 8 represent reconstruction errors (r) and SAI scores for the water distribution system. We use SGD to minimize r, where the errors (r) also provide localized information of adversarial inputs in case the system is under attack. The attack localization explains how and why an attack happens in the water distribution network. Fig. 9 illustrates that the attack appeared during $6^{th} - 11^{th}$ December, 2016 on tank 4 (L_{T4}) due to intentional pressure change on valve 7 (F_{PU7}) where the sensor reading (S_{PU7}) deviated as well. The proposed pipeline is able to detect such adversarial inputs during that time which is explained by r leading to SAI pointers. For the second use case, we use RFSP and compared FAI scores between the dataset *with* and *without* gender [20]. The pipeline generates a higher FAI score of 0.12 for the dataset *without* gender data (shown in Table VII).

VI. Summary and Conclusion

In this manuscript, we provide two MAA pipelines for achieving quantifiable assurance goals including XAI, FAI, TAI, EAI, CAI, and SAI. Although the algorithms are model-agnostic in nature, the use cases are model-specific (SCADA and Telco). ALSP is a model-driven approach that generates quantifiable assurance scores. It leverages game theory, AE, and logging to provide AIA goals; RFSP is a user-driven approach, where user input their expected AIA weights as a form of an equilibrium, the desired optimum set points dictate the final outcomes of assurance. Many works in AI systems management exist [21], but due to the unavailability of benchmark assurance standards, we are unable to compare the results with existing algorithms; nonetheless, in this manuscript, we present multiple empirical outcomes that are deemed successful for that goal. The two use cases presented are for: (1) a critical infrastructure: SCADA system, where we explain attack localization as a form of explainability using reconstruction errors from the AE and showed that Secret Inversion algorithm is capable of detecting adversarial inputs; and for a (2) Telco dataset, we test it by injecting intentional bias and testing if the pipeline detects it and reflects that in AIA scores. The algorithms had different success rates, albeit they all improved on the assurance of the AI systems at hand. Furthermore, potential areas of application include but are not limited to water distribution system, smart grids, and telecommunication systems. As part of future work, we plan to test other AI models using our framework and aim to create benchmarks for water treatment plants usage of AI, with the long term goal of securing complex and critical water distribution networks across the country.

References

[1] F. A. Batarseh, L. Freeman, and C.-H. Huang, "A survey on artificial intelligence assurance," *Journal of Big Data*, vol. 8, no. 1, p. 60, 2021. [Online]. Available: https://doi.org/10.1186/s40537-021-00445-7
[2] B. Mathew, P. Saha, S. M. Yimam, C. Biemann, P. Goyal, and A. Mukherjee, "Hatexplain: A benchmark dataset for explainable hate speech detection," in *AAAI*, 2021.
[3] S. Wachter, B. D. Mittelstadt, and C. Russell, "Counterfactual explanations without opening the black box: Automated decisions and the GDPR," *CoRR*, vol. abs/1711.00399, 2017.
[4] M. T. Ribeiro, S. Singh, and C. Guestrin, ""why should I trust you?": Explaining the predictions of any classifier," *CoRR*, vol. abs/1602.04938, 2016.
[5] H. Nori, S. Jenkins, P. Koch, and R. Caruana, "Interpretml: A unified framework for machine learning interpretability," *CoRR*, vol. abs/1909.09223, 2019.
[6] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," *ArXiv*, vol. abs/1705.07874, 2017.
[7] J. H. Friedman, "Greedy function approximation: A gradient boosting machine." *The Annals of Statistics*, vol. 29, no. 5, pp. 1189–1232, 2001. [Online]. Available: https://doi.org/10.1214/aos/1013203451
[8] F. Rossi and N. Mattei, "Building ethically bounded ai," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 9785–9789.
[9] S. Aghaei, M. J. Azizi, and P. Vayanos, "Learning optimal and fair decision trees for non-discriminative decision-making," *33rd AAAI Conference on Artificial Intelligence, AAAI 2019, 31st Innovative Applications of Artificial Intelligence Conference, IAAI 2019 and the 9th AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019*, pp. 1418–1426, 2019.
[10] T. Loeser and Y. Iwasaki, "Safety Verification Proofs for Physical Systems."
[11] U. Kuter and J. Golbeck, "SUNNY: A new algorithm for trust inference in social networks using probabilistic confidence models," *Proceedings of the National Conference on Artificial Intelligence*, vol. 2, pp. 1377–1382, 2007.
[12] L. S. Shapley, *A Value for N-Person Games*. Santa Monica, CA: Princeton University Press, 1953.
[13] S. R. Islam, W. Eberle, and S. K. Ghafoor, "Towards quantification of explainability in explainable artificial intelligence methods," *CoRR*, vol. abs/1911.10104, 2019.

[14] S. Rosenthal, M. Veloso, and A. Dey, “Asking Questions and Developing Trust.” 2009, pp. 133–140.

[15] J. Vrbik, “Small-sample Correction to Kolmogorov-Smirnov Test Statistics,” vol. 15, pp. 15–23, 2018.

[16] J. Bergstra, D. Yamins, and D. Cox, “Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures,” in *Proceedings of the 30th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, S. Dasgupta and D. McAllester, Eds., vol. 28, no. 1. Atlanta, Georgia, USA: PMLR, 17–19 Jun 2013, pp. 115–123. [Online]. Available: https://proceedings.mlr.press/v28/bergstra13.html

[17] R. Taormina and S. Galelli, “Real-time detection of cyber-physical attacks on water distribution systems using deep learning,” in *World Environmental and Water Resources Congress 2017*, 2017, pp. 469–479.

[18] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, available at:http://www.deeplearningbook.org.

[19] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *CoRR*, vol. abs/1412.6980, 2015.

[20] S. Leavy, G. Meaney, K. Wade, and D. Greene, “Mitigating gender bias in machine learning data sets,” *ArXiv*, vol. abs/2005.06898, 2020.

[21] A. Kulkarni, D. Chong, and F. A. Batarseh, “Foundations of data imbalance and solutions for a data democracy,” in *data democracy*. Elsevier, 2020, pp. 83–106.