Automated Testing of AI Models

Swagatam Haldar, Deepak Vijaykeerthy, Diptikalyan Saha
swagatam.haldar@ibm.com, deepakvij@in.ibm.com, diptsaha@in.ibm.com
IBM Research AI
India

ABSTRACT
The last decade has seen tremendous progress in AI technology and applications. With such widespread adoption, ensuring the reliability of the AI models is crucial. In past, we took the first step of creating a testing framework called AITEST for metamorphic properties such as fairness, robustness properties for tabular, time-series, and text classification models. In this paper, we extend the capability of the AITEST tool to include the testing techniques for Image and Speech-to-text models along with interpretability testing for tabular models. These novel extensions make AITEST a comprehensive framework for testing AI models.

1 INTRODUCTION
AI has been used in many diverse applications where the decision taken by the model directly impacts human life. It is therefore of utmost importance to make AI models as reliable as possible. Unfortunately, data scientists predominantly look for improving the accuracy or other generalization properties such as precision, recall, etc., often ignoring critical properties such as fairness, robustness while training the model. As a result, we have seen many instances of unfairness/bias and robustness issues in scenarios such as recidivism [27], job applications[1], etc.

Even though there are many techniques present for testing [32], there has been a dearth of comprehensive ML model testing tools which can work across modalities, different types of models, and go beyond generalizability. There exist few toolkits like AIF360 [2], CHECKLIST [22] which either concentrates on a single property such as fairness or single modality such as text, but a comprehensive set of testing algorithms under a common framework is scarce.

In [4], we presented a tool called AITEST, a framework for testing black-box models to solve the above problems. It has covered tabular, text, and time-series modalities and focused on generalizability, fairness, and robustness properties. Yet, our coverage was incomplete when it comes to testing a variety of models used by our clients [3].

In this paper, we include a variety of testing algorithms in our tool in three modalities viz. tabular, image, speech-to-text. Extensible design of AITEST helped us include such analysis without much change in the framework.

One of the major problems of AI models is interpretability. While the accuracy of AI models has increased over the years, the models have become more complex and un-interpretable. Interpretability testing deals with whether a given black-box model can be effectively simulated by a given interpretable model. This is essentially related to global explainability [7] and essential related to model auditing, especially in the finance industry who predominantly uses models built on tabular data. We include tree-simulatability testing to check whether a given black-box model can be simulated by a decision-tree model having interpretable characteristics specified by the user. We further add Image testing capabilities - essentially adding various kinds of transformations under which the model should be robust i.e. not change its decision. Specifically, we present eleven generic image transformations (inverse, saturation, etc.) and a Bayesian optimization-based algorithm for the adversarial attack on the black-block model.

Industrial conversation systems often use a speech-to-Text (STT) engine to convert human speech to text before the text is fed into the dialog system. There are many instances in which the failure of the dialog system is attributed to the STT. We, therefore, introduce the testing capability of fairness and robustness properties for STT models. Our contributions are summarized below:

- We present a tool called AITEST with functionalities related to a) interpretability testing of tabular AI models, b) comprehensive sets of image transformations for testing black-box image classifiers, c) a comprehensive set of audio transformations for testing fairness and robustness properties for speech-to-text models.
- We present a short evaluation to demonstrate the effectiveness of the above techniques.

Our tool is part of IBM’s Ignite quality platform [3] and is used in testing multiple industrial AI models.

The rest of the paper is organized as follows. The next section presents the flow of the tool. Section 3 describes our testing algorithms. Section 4 presents the experimental results. Section 5 presents the related work. We conclude in Section 6.

2 FLOW
AITEST uses a uniform flow across testing of models across different modalities (see left). The user (typically AI tester, data scientist or model risk manager) will start with registering a model into AITEST by providing model API details (endpoint, input template, output template, authentication headers), type of models (tabular-data classifier, time-series prediction, text-classifier, conversation-classifier, image-classifier, speech-to-text models), and seed data (training or test data which can be used in creating test samples).

Users can select the newly registered model or an existing registered model and proceed to select the model type-specific properties and input necessary configurations required for each property (the threshold for fairness, transformation control parameters, etc.) to schedule a test run. Users can view the status of the test run which shows the number of test cases generated and their status.
(passed/failed) specific to each selected property. Users can then view the metrics corresponding to the run for each completed property along with the failure samples and textual explanation. Users can compare the result of multiple runs.

3 TESTING ALGORITHMS

3.1 Interpretability Testing of Tabular Classifiers

Understanding the decision making process of most machine learning models are hard for human beings as typically they are optimized over performance metrics and not designed to be interpretable. The global explainability problem refers to creating an interpretable model which can mimic the reasoning of the target model for a given set of samples, usually the entire training data. Rule based models like decision trees or rule sets [5, 15, 30] and sparse linear models [21] are popular choices for such surrogates.

In our framework, we assess the interpretability of a given target blackbox classification model $M$ by first constructing a decision tree [5] surrogate $T$ over the entire training data but with the predictions from the model instead of the training data labels. We then evaluate the following three decision tree interpretability characteristics that the user provides. Note that huge decision trees are not really interpretable. While building $T$, we leverage data generation [7] to overcome sample scarcity to decide attribute splits at nodes far from the root.

**Average Path Length (APL).** It is defined [31] as the expected no. of nodes along any root to leaf decision path or equivalently, the expected length of a path in $T$ which simulates the target model $M$. The expectation is computed over all such paths in the surrogate.

$$\text{APL} = \frac{\sum_{\text{path} \in T} \text{No. of decision nodes in path}}{\text{Total no. of paths in } T} \tag{1}$$

Intuitively, this tries to measure the expected number of decisions to be made to arrive at an outcome for a sample according to the model $M$. Each decision here is essentially a boolean predicate evaluation for some attributes of the sample.

**Maximum Path Length (MPL).** It is the maximum number of nodes present in any decision path in the surrogate tree $T$.

$$\text{MPL} = \max_{\text{path} \in T} \text{No. of decision nodes in path} \tag{2}$$

This metric captures, in the worst case, how many decisions are to be made to predict the outcome of a sample. Interpretability favours small values of APL and MPL.

**Fidelity.** It is computed as the percentage of test samples (from a reserved suite $\mathcal{D}$) for which predictions for labels from target model $M$ and the surrogate tree $T$ match i.e., are same.

$$\text{Fidelity} = \frac{\sum_{x \in \mathcal{D}} 1_{M(x) = T(x)}}{\text{Total no. of samples in } \mathcal{D}} \tag{3}$$

where $x$ is a sample from $\mathcal{D}$.

Fidelity is a measure of how accurately the surrogate tree $T$ resembles the target model $M$ in terms of output and decision boundary.

The successful test of tree-simulatability testing generates a decision tree model which satisfies all the above characteristics.

### Table 1: Image transforms

| Name               | Type     | Description                                                                 |
|--------------------|----------|----------------------------------------------------------------------------|
| Inverse            | Linear   | Inverts the image                                                          |
| Scale              | Affine   | Scales the image by a factor                                              |
| Rotate             | Affine   | Rotates the image by an angle                                              |
| Shear              | Affine   | Slants an image by an angle                                                |
| Saturation         | Linear   | Perturbs the saturation levels of the images                               |
| Brightness         | Linear   | Changes the brightness of the image                                        |
| Contrast           | Linear   | Changes the contrast of the image                                          |
| Fog                | Convolution | Simulates realistic fog on the image                                       |
| Gaussian Blur      | Convolution | Blurs the image through a random Gaussian Kernel                           |
| Zoom & Blur        | Convolution | Zooms to part of the image & blurs it                                      |
| Gaussian Noise     | Linear   | Adds a random noise sampled from a unit Gaussian distribution to the image |
| Bayesian Optimisation | Adversarial | Generates adversarial examples via Bayesian Optimisation                  |

3.2 Image Classifier Testing

Deep Neural Networks (DNNs) have found widespread usage in several Computer Vision (CV) tasks. Even though they match (or exceed) the performance of humans on these tasks, in practice, DNNs are vulnerable to malicious inputs [14, 29]. Consequently, there is an increasing need to develop techniques to test these models before deployment. In our framework, we propose a rich set of transforms (refer to table 1) to test DNNs for CV tasks, each of which checks whether the prediction of the model changes on the transformed images.

The transforms used by our framework broadly fall into two categories. Static transforms such as translation, scaling, rotation, blurring, etc. [26], and dynamic ones that generates black-box adversarial examples via Bayesian Optimisation [25]. Static transforms can be further classified into the following three categories: linear, affine, and convolution.

**Static Transforms.** In the case of linear transforms, we add, subtract or multiply either a constant or random noise to each pixel of an image. For instance, to change the brightness of a given image, we add (or subtract) a constant value, and in the case of random Gaussian noise transform, we add random noise sampled from a unit Gaussian distribution to the image. Affine transforms modify the geometric structure of the image while preserving proportions of lines, but not necessarily the lengths and angles. Typically, affine transforms can be represented by a 2D matrix, and hence are amenable to compositions. In practice, these transforms (or their compositions) are used to simulate several image deformations which a model would encounter during a real-world deployment.

Convolutions are general purpose filters that work by multiplying a pixel’s and its neighboring pixel’s value by a matrix (aka a kernel matrix). Intuitively, the value of each transformed pixel is computed by adding the products of each surrounding pixel value with the corresponding kernel value. Transforms such as Blur, Fog, etc., are a few examples of convolution transforms.

**Dynamic Transforms.** Adversarial examples are malicious inputs crafted by adversaries to fool DNNs. In our framework, we generate
We solve the above problem via Bayesian Optimisation which of-
f typically, Bayesian Optimisers has 2 key ingredients, a surrogate value to the parts of the input space where the value of the target
sampled and denote the original clip by $x$ and its corresponding perturbed clip by $x'$. We solve the above problem via Bayesian Optimisation which offers an efficient approach to solve global optimisation problems. Typically, Bayesian Optimisers has 2 key ingredients, a surrogate model like a Gaussian process (GP) or a Bayesian Neural Network (BNN), and an acquisition function $a(.)$ which provides us the next location to query the target function. Intuitively, an acquisition function balances the exploration and exploitation by giving higher value to the parts of the input space where the value of the target function is typically high and the surrogate model is very uncertain. The approach of using Bayesian Optimisation with a GP surrogate to attack the target model is described in Algorithm 1.

### Algorithm 1: Bayesian Optimisation Attack

**INPUT**: A black box function $f(x)$, seed data $D_0$, no. of
iterations $T$

**OUTPUT**: Adversarial noise $\delta_T$

for $m = 1, 2, \ldots, T$ do
  Select $\delta_m = \arg \max_y \{ f(x + \delta) \neq y \}$
  $y_m = f(x + \delta_m)$ and $D_m \leftarrow D_{m-1} \cup \{ \delta_m, y_m \}$
  Update the surrogate model with $D_m$

3.3 Speech-to-Text Model Testing

Automatic speech recognition engines are being widely used in many automated tasks (voice assistants to a myriad of downstream activities) as such systems make it very convenient for humans to clarify their intents. Examples include Speech-to-Text services from Google, Microsoft, Apple, Amazon, IBM etc. and also offline engines like Mozilla’s DeepSpeech [9]. The performance of the downstream tasks often depends on the precision of text transcription of the audio since it is fed to language understanding models next.

It is important to realize that humans are not always in a conducive surrounding for giving clear voice commands and regularly noise artefacts get blended in the input speech. In our framework we test the robustness of the blackbox audio transcription models under white background noise and various interfering environmental perturbations to its input clips. We also test for simple fairness properties such as changing the gender or accent of the input speech having identical textual content.

We measure the Word Error Rates (defined later) of text transcriptions to measure the effectiveness of our testing.

For the following paragraphs, we consider an audio clip as a one-dimensional signal of appropriately sampled and normalized values and denote the original clip by $x$ and its corresponding perturbed clip by $x'$. We test for simple robustness to its input clips. We also test for simple fairness properties such as changing the gender or accent of the input speech having identical textual content. Also note that in case of standard normal distribution, since it has zero mean, its variance is the $RMS_{\text{noise}}$ value squared.

$$\sigma^2 = RMS_{\text{noise}}^2$$

Using the above equations we can compute the perturbed signals for any desired $SNR_{\text{db}}$ specified during testing.

**Interference Transform.** In this transformation, we overlay environmental noises that simulate different practical scenarios on top of the original clip. Example scenarios include restaurant, rainfall, festival, water dripping, windy noise etc. We control the signal strength by using the parameter $\theta$ while generating the linear combination of signal and noise that produces the interference effect.

$$x' = \theta x + (1 - \theta) \eta$$

**Fairness Transforms.** Speech-to-Text transcription models are expected to have similar performance irrespective of the sensitive or protected attributes of the speaker. In this pretest, we test the models by synthesizing the perturbed input $x'$ with the same spoken words or tokens in the same language but flipping its gender (male to female and vice versa) or by changing the speaker’s accent such as Indian, American, French etc.

We synthesize the perturbed inputs using open-source speech generation engines or pretrained Text-to-Speech models (such as Tacotron2 [24] and Waveglow [20]) that synthesize audio from natural language transcripts directly. By training these models on audio from different speakers, we can leverage the learnt transformations. These transforms try to capture the fairness of the models to different genders and varying ethnicity of the end-users who may be consuming the product or using in a critical decision making scenario.

4 EVALUATION

In this section we present some basic evaluation of our techniques of the key underlying novel testing techniques. This demonstrates that AITEST can be effectively used.

**Interpretability Testing for Tabular Data.** In Table 2 we provide the interpretability metrics for some well known datasets: Adult Census Income [12], Bank Marketing [19] and US Execution [18] all obtained from the UCI repository [8]. We have taken blackbox tabular models from IBM Watson WML framework. Here the user may set appropriate thresholds for the three metrics to decide whether the test fails or not.
| Dataset       | APL | MPL | Fidelity(%) |
|--------------|-----|-----|-------------|
| Adult Income | 11.8| 18  | 91.12       |
| Bank Marketing| 11.29| 18  | 94.25       |
| Execution    | 8.07| 12  | 97.92       |

Table 2: Interpretability results

Figure 2: Failure Image Transformations

| Transform | Dataset | MNIST | CIFAR10 |
|-----------|---------|-------|---------|
| None      | 99.8    | 92.1  |
| Inverse   | 97.1    | 86.1  |
| Scale     | 96.1    | 88.2  |
| Rotate    | 93.7    | 91    |
| Shear     | 97.8    | 85.1  |
| Saturation| 97.1    | 90.1  |
| Brightness| 97.3    | 90.3  |
| Contrast  | 96.5    | 90.5  |
| Fog       | 91.7    | 86.4  |
| Gaussian Blur | 94 | 85.6 |
| Zoom & Blur| 91     | 82.1  |
| Gaussian Noise | 92 | 89.4 |
| Bayesian Optimisation | 10 | 9 |

Table 3: Accuracy (%) of various models on transformed example

Image. To illustrate the effectiveness of our framework, we tested a couple of standard models on popular data sets LeNet5 [16] on MNIST [17] and ResNet32 [10] on CIFAR10 [13]. From Table 3, we can see that test cases generated by various transforms in our framework is quite effective in fooling the models. User can view such cases in the tool as shown in Figure 2.

Speech-to-Text. For evaluating the perturbations, we use Word Error Rate (WER) as a metric which is defined as the fraction of words omitted, substituted or newly inserted in the text transcription of the perturbed clip, when compared with the original text. If we have multiple audio samples, we simply the average the metric over all samples.

We have tested it in a client-scenario which requires a voice assistant to process simple commands or queries. We provide some example failures detected by our method in Table 4. User can play the original and transformed audio for the failure cases as shown in Figure 3. We direct interested readers to [23] and our supplement video that contain more examples and details.

A short video demonstration of our tool with a subset of properties include in this paper is available at https://youtu.be/sc0YEHx4m0c.

5 RELATED WORK

In the recent past, there has been a lot of interest in developing techniques for testing DNNs for Computer Vision tasks that vary from simple transforms such a translation, rotation, etc. [29], to richer formulations based on GANs [33], Concolic Testing [28] & adversarial examples generation [14, 25], we leverage some of these techniques in our framework. Similar techniques have also gained traction for speech recognition models. [6] uses fully white-box gradient-based methods to generate targeted and untargeted attacks for models such as [9]. Although the notion of simulatability and tree-simulatability are not new [31] and testing a model for tree-simulatability or interpretability seems to be novel. Overall, AITEST remains the comprehensive tool for testing these many modalities.

6 CONCLUSION AND FUTURE WORK

The last decade has seen various types of models and their applications. Only in recent time, efforts has been made to define a concrete Data and AI lifecycle [11] and testing plays an important role in the lifecycle to ensure the reliability of AI models. We present a tool AITEST which encompasses variety of testing techniques across multiple modalities. In this paper, we include some novel capabilities such as interpretability testing and fairness testing of Speech-to-text models along with the implementation of some known properties. However, implementation of all the properties under one framework which provides easy usability is very useful for serving variety of client models.

In future, we plan to add support for testing videos, multi-modal inputs, model compositions. The current implementation only supports black-box testing, which when configured, can be applied to a large number of other similar models of the same type without much change. We plan to add support for white-box testing algorithms.

REFERENCES

[1] Amazon scraps secret AI recruiting tool that showed bias against women, Last accessed 18th August, 2021. https://www.reuters.com/article/us-amazon-com-jobs-automation-insight-idUSKCN1MK08G.
[2] IBM AIF360, Last accessed 18th August, 2021. https://github.com/IBM/AIF360.
[3] IBM IGNITE, Last accessed 18th August, 2021. https://www.ibm.com/in-en/services/applications/testing.
[4] Aniya Aggarwal, Samuulla Shaikh, Sandeep Hans, Swastik Haldar, Rema Ananthanarayanan, and Diptikalyan Saha. Testing framework for black-box ai models. In 2021 IEEE/ACM 43rd International Conference on Software Engineering: Companion Proceedings (ICSE-Companion), pages 81–84. IEEE, 2021.
