Demystifying AI: The Need for Explainable AI and Its Applications

Aaryan Chokshi
CSE Department,
Nirma University,
Ahmedabad, India
20bce002@nirmauni.ac.in

Devansh Shah
CSE Department,
Nirma University,
Ahmedabad, India
20bce055@nirmauni.ac.in

Devasy Patel
CSE Department,
Nirma University,
Ahmedabad, India
20bce057@nirmauni.ac.in

***Abstract*—With the advent of Large Language Models like ChatGPT and with the growing progress and fascination in Machine Learning and Deep Learning Models, it has become imperative for all to understand how to efficiently use Artificial Intelligence Techniques in various domains. Explainable AI provides a pathway for the upcoming engineers and domain experts to understand the mysterious black box, i.e deep learning models in order to better utilize them.**

***Index Terms*—XAI, SHAP, LIME, InterpretML, Deep Learning, Machine learning**

I. Introduction

A. *Definitions*

Artificial Intelligence is the science and engineering of making intelligent machines, especially building intelligent computer programs and systems that are able to learn with limited human intervention [1]. The inspiration that led to the genesis of artificial intelligence is the working of the human brain. Ever since; the field of artificial intelligence has specialized into many fields; most notable being Machine Learning and Deep Learning.

Machine learning is the design and analysis of algorithms designed to learn important information regarding the task at hand without human intervention. These models use statistical algorithms to learn and identify patterns in data.

Deep learning is the inception of a new class of data-learning models that are made by attempting to mimic the human brain. The models consist of artificial 'neurons'; computational units that perform a relatively small computation over the input data, which when aggregated over a large number of neurons; are joined by varying amounts of 'weights', thereby simulating the synaptic connections of the human brain; could perform increasingly complex tasks which could not be achieved by machine learning algorithms.

B. *Need of Explainable AI*

A large number of architectures have been built in the deep learning domain over the past 20 years, with increasingly complex and deep networks coming up in recent years. With the advent of complex natural-language processing AI such as ChatGPT; which can respond to input queries with astonishingly humane responses and StableDiffusion; which can generate almost real, human-like artwork, it becomes imperative to possess tools that can explain the learning and 'thought process' behind the decisions taken by such AI; which can help researchers with training deeper AI networks, during training and validation. Precious time and resources can be saved if the researchers knew how the learning of AI progressed over time and if it was approaching the desired state or not.

Deep neural networks are extremely difficult to debug and explain how they arrived at a particular output decision; as the information about their learning is encoded within the interconnected weights and biases of the network. The single neuron and its subsequent connections could be easily explained and a group of neurons with relative difficulty; but the whole is beyond [3]. This provided researchers essentially with a black box model; where the only parameter they can effectively evaluate the model's effectiveness was the output obtained. This being the only parameter until now greatly diminished the possibilities of probing the existing models further in order to obtain a more comprehensive understanding of the model and subsequently making further optimizations more effective and less time-consuming.

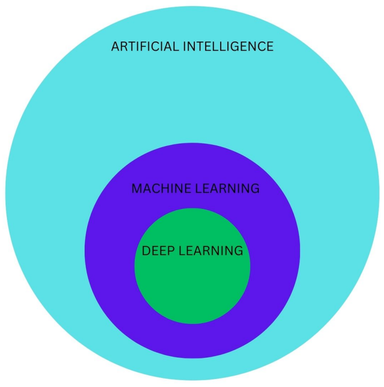

Fig. 1. AI - ML Hirerarchy

In order to address the aforementioned problems in the AI research community, we propose a few XAI tools and methodologies that can assist in generating specific parameters that measure the efficiency of learning of the model; as well as give insights as to how the deep networks operate the way they do while they present an output.

II. Techniques for XAI

The techniques of XAI are mainly concerned with designing and implementing algorithms that enable the generation of interpretable models. These algorithms can be broadly classified into two categories:

A. Model-specific techniques

Model-specific techniques are XAI techniques designed to provide insights into specific models, such as decision trees, rule-based systems, or linear models. These techniques are tailored to the specific model architecture and the way it makes decisions. Model-specific techniques are those that rely on a certain model structure to generate explanations for its predictions. They are more tailored and precise than model-agnostic techniques, but may not be applicable to other models.

1) Saliency Maps: A technique that visualizes the gradient of the output with respect to the input pixels, highlighting the regions that are most relevant for the prediction [36]. This technique is useful for image classification models, such as convolutional neural networks (CNNs), as it shows which parts of the image contribute most to the class label. However, it may not work well for other types of models or inputs, such as text or tabular data.

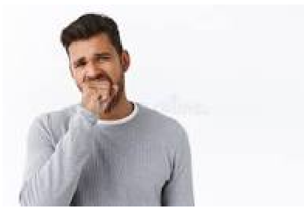
Fig. 2. Original Image

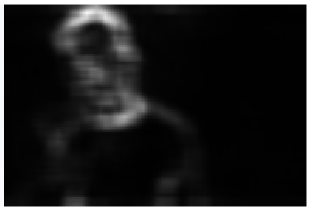
Fig. 3. With Saliency Maps

As we see in above image, the pixels that represent the yawning person are being highlighted!

2) LRP: LRP stands for layer-wise relevance propagation, which is a technique that explains the predictions of neural networks by propagating relevance scores from the output layer to the input layer. LRP can help understand how neural networks process information and what features are important for their decisions. LRP can be applied to different types of neural networks, such as convolutional neural networks (CNNs) or recurrent neural networks (RNNs).

i) LRP is based on the principle of conservation of relevance, which states that the total relevance at the output layer should be equal to the total relevance at the input layer. Relevance is defined as the contribution of an input feature to an output prediction. LRP assigns relevance scores to each neuron in each layer based on its activation value and its connection weights with other neurons. LRP uses different rules to distribute relevance scores from one layer to another, depending on the type and function of each layer [17]. ii) LRP can produce saliency maps that visualize the relevance scores of each input feature for a given output prediction. Saliency maps can highlight which parts of an image or text are most relevant for a classification or regression task. LRP can also produce heatmaps that compare the relevance scores of different output classes for a given input. Heatmaps can show which features are discriminative or common for different classes [17].

3) Attention Mechanisms: A technique that learns to assign weights to different parts of the input, such as words in a sentence or regions in an image, indicating how much attention the model pays to them [**?**]. This technique is useful for natural language processing (NLP) models, such as recurrent neural networks (RNNs) or transformers, as it shows which words or phrases are most important for the task. However, it may not be applicable to other types of models or inputs, such as numerical or categorical data.

4) Tree Interpreter: A technique that decomposes the prediction of a tree-based model into the contributions of each node along the path from the root to the leaf2. This technique is useful for regression or classification models, such as decision trees or random forests, as it shows how each feature affects the prediction. However, it may not work well for other types of models or inputs, such as neural networks or images.

5) Rule Extraction: Rule extraction is a technique that aims to extract comprehensible and interpretable rules from complex machine learning models, such as neural networks or support vector machines. Rule extraction can help users understand how the model makes predictions, verify its correctness, and improve its transparency and trustworthiness. Rule extraction can be classified into three types: decompositional, pedagogical, and eclectic [19].

i) Decompositional rule extraction is a technique that extracts rules by analyzing the internal structure and parameters of the model. For example, one can extract rules from a neural network by examining the weights and activation functions of each neuron. Decompositional rule extraction is model-specific and requires detailed knowledge of the model architecture and implementation.

ii) Pedagogical rule extraction is a technique that extracts rules by treating the model as a black box and observing its input-output behavior. For example, one can extract rules from a support vector machine by sampling its decision boundary and generating rules that cover the regions of different classes. Pedagogical rule extraction is model-agnostic and does not require access to the model internals.

iii) Eclectic rule extraction is a technique that combines both decompositional and pedagogical approaches to extract rules. For example, one can extract rules from a neural network by first decomposing it into subnetworks and then applying pedagogical methods to each subnetwork. Eclectic rule extraction is hybrid and can leverage both model-specific and model-agnostic information.

6) Prototype-based Extraction: Prototype-based extraction is a technique that extracts prototypes from complex machine learning models, such as clustering or classification models.

Prototypes are representative examples that capture the main characteristics of a class or a cluster. Prototype-based extraction can help users understand the similarities and differences among different groups of data, as well as provide intuitive explanations for predictions. Prototype-based extraction can be classified into two types: centroid-based and medoid-based

7) Rule-based: Rule-based systems are systems that use rules to represent knowledge and perform reasoning or inference. Rules are statements that express logical relationships among facts or conditions. Rule-based systems can help users solve problems, make decisions, or perform actions based on predefined rules. Rule-based systems can be classified into two types: deductive and inductive.

8) ANOVA: ANOVA stands for analysis of variance, which is a statistical test that compares the means of three or more groups of data. ANOVA can help determine whether the differences among the group means are significant or due to chance. ANOVA can be classified into different types depending on the number and nature of the independent variables. The most common types are one-way ANOVA and two-way ANOVA.

i) One-way ANOVA is a type of ANOVA that uses one independent variable with at least three levels (groups or categories). For example, one can use a one-way ANOVA to test the effect of three types of fertilizer on crop yield. The null hypothesis of one-way ANOVA is that there is no difference among the group means, and the alternative hypothesis is that at least one group mean differs from the others. One-way ANOVA uses the F test to compare the variance between groups with the variance within groups. If the F value is large enough, it indicates that the group means are not equal and the null hypothesis is rejected [18]. ii) Two-way ANOVA is a type of ANOVA that uses two independent variables with at least two levels each. For example, one can use a two-way ANOVA to test the effect of fertilizer type and planting density on crop yield. The null hypothesis of two-way ANOVA is that there is no interaction effect between the two independent variables, and no main effect of either independent variable. The alternative hypothesis is that there is an interaction effect, a main effect, or both. Two-way ANOVA uses the F test to compare the variance due to interaction, main effects, and error. If any of the F values are large enough, it indicates that there is a significant effect and the null hypothesis is rejected [18].

B. Model-agnostic techniques

Model-agnostic methods work for any kind of ML models, while model-specific methods rely on a certain model structure. Model-agnostic methods are more general and flexible, but may not capture all the nuances of a particular model.

1) PDP: Partial Dependence Plot (PDP) is an XAI technique used to explain how a model's output changes with variation in one or more features of the input. PDP is a model-agnostic technique and works by plotting the marginal effect of a feature on the model's output. The marginal effect is the average change in the model's output as the feature of interest varies, while all other features remain constant. The PDP plot shows how the model's output varies with changes in the value of the feature of interest, while keeping other features at their average value. PDPs can be used to identify which features are most important to the model's prediction, and to detect interactions between features. One advantage of PDP is that it is simple to interpret and can help identify important features that can be used for feature selection. However, one drawback of PDP is that it can hide interactions between features, and may not reveal the full complexity of the model. Another limitation of PDP is that it assumes independence between features, which may not be the case in some models. The formula for calculating the Partial Dependence Plot for a single feature i is given by:

$$f_{x_{-i}}(x_i) = \frac{1}{N-1} \sum_{j=1, j\neq i}^{N} f(x_{-i}, x_{ij})$$

where $\widehat{f}x{-}i(x_i)$ is the estimated function of the feature i, x_{-i} is a vector of all the features except i, x_{ij} is the jth observation of the feature i, and N is the number of observations. This formula computes the average output of the model when the feature of interest is set to x_i while keeping all other features constant at their observed values. PDPs are useful for understanding the impact of a single feature on the model output

2) ICE: Individual Conditional Expectation (ICE) is an explainable machine learning technique that visualizes how a specific feature affects the prediction of a model for a given input. ICE generates a separate line plot for each instance in the dataset, where the x-axis represents the values of a chosen feature, and the y-axis represents the predicted outcome of the model. This method provides a more detailed understanding of the relationship between a feature and the model's prediction, especially when there is a non-linear relationship between the two. ICE is a model-agnostic method, which means it can be used for any type of model, and it is relatively fast to compute. However, ICE does not consider the interaction between features, which can be important in some cases, and it may not provide a global understanding of the model's behavior.

The formula for Individual Conditional Expectation is as follows:

For a dataset D, with input x, and feature vector x_i:
$ICE(x_i, D) = f(x_1, x_i), f(x_2, x_i), ..., f(x_n, x_i)$

Where f is the machine learning model that takes input x and returns the predicted output.

To represent the ICE plot, the above formula is used to compute the predicted outcomes for each instance in the dataset. Then, a line plot is generated for each instance, where the x-axis represents the values of the chosen feature, and the y-axis represents the predicted outcome.

3) LIME: LIME (Local Interpretable Model-Agnostic Explanations) is a technique for explaining the predictions of any machine learning model in a localized setting. It was proposed in 2016 by Marco Tulio Ribeiro and his colleagues. The goal of LIME is to provide an explanation for the decision made

TABLE I
COMPARISON OF MODEL-SPECIFIC XAI TECHNIQUES

Technique	Category	Description
Decision Tree Induction	Pros	Easy to interpret and explain. Provides decision rules and paths that can be visualized to understand the factors that the model considers important.
	Cons	Limited to models that use decision trees. May not work well with complex models such as deep neural networks.
Rule Extraction	Pros	Provides a set of human-readable rules that can be used to explain how the model works and why it makes certain decisions. Can work with complex models such as deep neural networks.
	Cons	Can be computationally expensive. May not work well with models that are highly nonlinear or have a large number of parameters.
Feature Importance Analysis	Pros	Provides a quantitative measure of the contribution of each feature to the model's output. Can be used to identify the most important features for a model's predictions.
	Cons	May not capture interactions between features. Can be sensitive to feature scaling and correlation.
Prototype-based Explanations	Pros	Provides an intuitive explanation of the model's decision by identifying similar examples from the training dataset. Can work with any model architecture.
	Cons	May not capture the full complexity of the model's decision process. Requires a large and diverse training dataset.
Rule-based Systems	Pros	Easy to Interpret and explain. Provides a set of if-then rules that can be used to explain how the model works and why it makes certain decisions. Can work with any model architecture.
	Cons	May not capture the full complexity of the model's decision process. Rules may be difficult to construct and maintain for complex models.

by the model [2], which is specific to the prediction it makes for a particular instance.

LIME generates these explanations by constructing a local explanation model that approximates the target ML model's behaviour in a specific locality [2]. This explanation model can take inputs in different formats, such as text, tabular, or image data, and generate explanations for any ML model's prediction behind it.

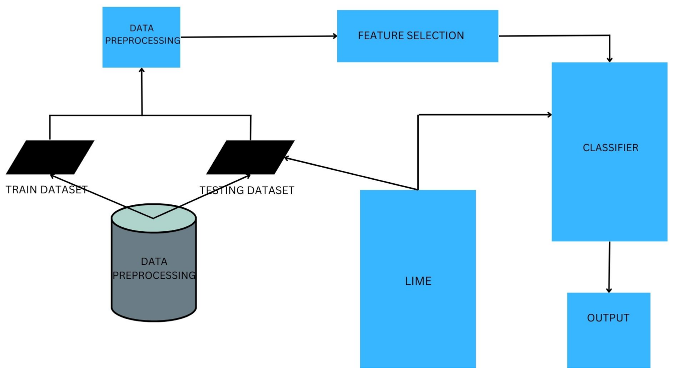

Fig. 4. Lime process diagram

The main algorithm behind LIME is a minimization problem, which aims to minimize the loss function $L(f, g, (pi) * x)$ [2]. The authors in [2] describe LIME as a local explanation model denoted by g, which provides justification for the final output. Hence, by addressing the minimization problem, it provides a very accurate local explanation model g thus giving a more accurate local explanation model for the prediction of the target ML model f on the input x [2].

LIME optimises the loss function by modification during the training of this local explanation model. The technique creates additional instances x' in the region $(pi) * x$ surrounding the first input point x. The local explanation model g is then updated with this fresh dataset of perturbed examples, allowing LIME to further enhance g's local domain accuracy.

LIME's explanations are optimized to be as simple as possible. To ensure that the explainable models g are as simple as possible, [2]the $W(g)$ term measures the general complexity behind the explanation function and is included in the minimization problem. Keeping these explanations simple increases usability and interpretability for end-users.

Despite the best efforts of LIME's developers to simplify the explanations generated by this algorithm, [2] the interpretability of the final results is still mediocre for non-expert users. LIME only remains faithful in its predictions for an ML model on a localized level, and it doesn't convey its limitations regarding locality to end-users. LIME also has consistency issues, often changing its explanations completely despite minimal changes being made to the input and output of the ML model. Additionally, LIME is vulnerable to adversarial attacks that attempt to exploit its weaknesses as mentioned in [2].

4) SHAPley values: SHapley Additive explanations (SHAP) is an explainable artificial intelligence (XAI) technique proposed by Scott M. Lundberg in 2017. It relies on the calculation of Shapley values, which are a mathematical concept that emerged from game theory, to explain the importance of each feature within a machine learning (ML) model's prediction [2]. The calculation of Shapley values involves iterating through all possible subsets of the original feature space and calculating the difference in the model's output with and without the inclusion of a specific feature. [2] SHAP uses these values to explain the influence of each feature on the model's prediction.

SHAP has several advantages, such as its mathematically enforced properties [2], including consistency and accuracy, and its ability to evaluate feature weights in a way that fits human intuition better than other XAI techniques, such as LIME [2]. However, SHAP's computational time is relatively slow, even with the use of the approximation implementation called Kernel SHAP. Additionally, Kernel SHAP does not consider the interaction between features when calculating weights, and SHAP has been shown to be inconsistent and vulnerable to adversarial attacks despite the mathematical properties of Shapley Values.

In conclusion, SHAP is a model-agnostic XAI technique that relies on the calculation of Shapley values [2] to explain the influence of each feature on a machine learning model's prediction. Although it has several advantages, such as its mathematically enforced properties and ability to fit human intuition better than LIME, its computational time is slow, and it has limitations, such as not considering feature interactions and being vulnerable to adversarial attacks.

III. Tools

1) Flowcast: Flowcast is a patented white-box machine learning model that generates high-performing machine learning prediction algorithms that better leverages risk, measures the company dilution and the risk of delinquency with higher accuracy. The model has speed, accuracy and thin-data predictions; i.e. making reasonably accurate predictions even with a small number of data for firms. [20]

Flowcast predominantly works on the following spheres of business finance :

- Credit decisions : A significant problem of today's business market is that firms are not able to assess the risk for small businesses which do not have a lot of data to back their creditworthiness with. Hence, predicting risk using traditional deep networks may not necessarily provide a clear picture of the business' finances and credit value. [22]
  Flowcast provides accurate predictive solutions that merge data from various available and emerging sources that provide a more comprehensive outlook compared to traditional credit-scoring methods.
- Collections : The current lender mindset is reactive; rather than proactive. This leads to higher cost of provisions.
  To mitigate this issue, Flowcast employs pre-incident handling mechanisms due to the fact that there are more re-negotiation options; which are useful in extreme cases pertaining to delinquency and liquidation.

2) TensorBoard: It is a visualization toolkit for TensorFlow that enables the visualization of model training and validation metrics, graph structures, and other statistics.

It provides a comprehensive understanding of the model in question by providing statistical measures like histograms related to each weight and bias changes over time, visualizing the model graph and reducing the dimensions of word embeddings; among other functionalities. [21]

It is directly integratable with TensorFlow and Keras libraries and can be accessed via a separate window that can display all the relevant visualizations.

3) IBM's AI Explainability 360: It is an open-source toolkit that provides a suite of algorithms and tools for interpreting and explaining machine learning models.

This tool is a collection of state-of-the-art tools that support the interpretability and explainability of the model. This allows users and model architects to gain a better understanding of the model by gaining insights to the model's decision-making processes. People with differing expertise in their fields of interest would affect their decision-taking ability as well as their viewpoint on a given issue. IBM's AI Explainability 360 precisely takes care of the same through case-based reasoning; directly interpretable rules that can cater the varying requirements of explanation required to the end-user. [24]

Moreover, the open-source nature of the software enables researchers, policy-makers, data scientists and the general public to contribute to AI explainability which sends out a powerful message in the AI community. As a result, it is likely to attract more people into the explainable AI community and likely; further the knowledge on the same.

4) Captum: It is a PyTorch-based library that provides a set of model interpretability algorithms, including Saliency Maps, Integrated Gradients, and DeepLift.

In PyTorch 1.3, Captum was added as a tool to help with this problem. Captum examines ML models using cutting-edge interpretability techniques, enabling developers to build AI systems that are easier to comprehend and, as a result, more dependable and predictable. Engineers and developers can use Fiddler to get meaningful insights by analysing the decision-making behaviour of AI algorithms. They can prevent and deal with difficulties involving errors in judgement, bias, and unfairness thanks to these insights. We provide an effective set of tools for studying and using AI explainability by combining Fiddler with Captum, PyTorch's open-source model interpretability library. [23]

5) InterpretML: It is a Python library that provides a suite of model-agnostic techniques for interpretability, including feature importance analysis, partial dependence plots, and individual conditional expectation plots.

6) wandb.ai: Weights and Biases (Wandb) is a tool that helps machine learning practitioners track, visualize, and manage their machine learning experiments. It provides a suite of tools for tracking metrics, visualizing results, and collaborating with team members. It is a key part in Explainable AI because it provides real time tracking and visualising of the models like the accuracy, loss, learning rate and it also tracks their change over time.With this, we gain insights into how our machine learning model is performing, identify potential issues, and debug problems more quickly. [25]

Moreover, wandb provides tools for visualizing the decision boundaries of models, allowing you to see how the model is making its predictions.

TABLE II
COMPARISON OF EXPLAIN-ABILITY TOOL KITS

Category	Tensorboard	Captum	Flowcast	IBM 360
Definition	A visualization toolkit for machine learning experimentation	A model interpretability library for PyTorch	A platform for building and deploying explainable AI applications	An open-source toolkit for explainable AI
Supported languages	Python, C++, Java, Go	Python	Python, R, SQL	Python, R
Supported frameworks	TensorFlow, PyTorch	PyTorch	TensorFlow, PyTorch, scikit-learn, XGBoost	TensorFlow, PyTorch, scikit-learn, XGBoost, Keras
Supported models	Any TensorFlow or PyTorch model	Any PyTorch model	Any TensorFlow, PyTorch, scikit-learn or XGBoost model	Any TensorFlow, PyTorch, scikit-learn, XGBoost or Keras model
Visualization features	Tracking and visualizing metrics, model graph, histograms of weights and biases, embeddings projection, images, text and audio data display, profiling	Attribution methods (feature importance), feature ablation (feature removal), visualization of input/output gradients and activations	Interactive dashboards, decision trees, partial dependence plots, Shapley values, counterfactuals	Interactive dashboards, decision trees, partial dependence plots, Shapley values, counterfactuals, contrastive explanations
Interactivity features	Metadata editor, distance metric/space selection, neighborhood function selection, t-SNE perturbation and semi-supervision	None (only provides APIs)	Data exploration and manipulation tools	Data exploration and manipulation tools
Scalability features	Distributed training support	None (only supports single GPU)	Distributed training support	Distributed training support
Documentation and community support	Comprehensive documentation and tutorials, large and active community	Comprehensive documentation and tutorials, moderate community	Limited documentation, small community	Comprehensive documentation and tutorials, moderate community

IV. CHARACTERISTICS

A. Explainability

The model works in a way that the decision-maker / expert can comprehend. It communicates both the solution and the process by which it arrived at the solution by elaborating the features.

B. Understanding

To comprehend the structure and design of the model, the traits and behavioural elements of each individual component are examined.By describing what the model does inside, we can better improve model's performance.

C. Fidelity

It assesses suitability and explains the implementation processes. Each system component's operational process is described.

D. Transparency

Because every model has a different degree of understandability and legibility on the decisions that are made as part of producing a better outcome, a paradigm is said to be transparent if it is understood on its own.

E. Adaptability

The ability of the environment in which the model is deployed to operate to learn is the key to the model's adaptability. The model must have the ability to quickly adjust to new information and understand its importance and accordingly, modify its predictions.

V. APPLICATIONS

The advent of Machine Learning and Deep Learning Techniques have completely changed our life. Although Deep Learning Models are giving higher accuracy and performing better than traditional techniques like Neural Networks, Clustering and Regression, there is a growing need for transparency regarding how the model is performing because eventually, the model is a huge blackbox full of mysteries. Hence, Explainable AI is critical not just for regulatory compliance, but also for developing confidence with end users.

It assists in establishing confidence and increases the usability of AI systems by offering clear and simple explanations for the decisions made by AI models. There are diverse applications of Explainable AI ranging from healthcare and finance to autonomous systems and beyond.

1) Healthcare: The authors in paper [4] discussed the evolving Explainable AI Techniques for Healthcare across the previous decade and how it is shaping the healthcare industry.

According to the authors in [8], they discuss the Clever Hans phenomena which proves that sometimes, deep learning model learns data and not features and metadata. A real life example was when researchers in Mount Sinai Hospital developed a model to distinguish high-risk patients but the model did not work well outside the hospital. Hence, the need of how did we arrive at the conclusion rose. [7]

i) Saliency Maps was a huge breakthrough in the field of medical imaging. Now, the Chest X-rays and CT Scans can be better visualized by Saliency Mapping which finds the most prominent regions in the image and thus, the doctors can easily identify any malicious cancerous cells. [?]

ii) XAI has a big use case in Drug Discovery and Testing. The authors in [5] gave a comprehensive review on how we can best use the available technology to advance in Molecular Sciences. Methods based on graph convolution offer a potent tool in drug discovery. The ability to highlight atoms that are important for a specific prediction can enhance a model's justification as well as its ability to provide information about the underlying biological and chemical processes. [6]

2) Finance: For example, in the stock market, a single decision can make or break millions and billions of rupees. Hence, it is highly essential that we understand why the model is telling us to buy, hold or sell. The explainability of the model along with an experienced human together will lead to great profits. Some applications in finance sector include:

i) Default Prediction: The goal of default prediction is to estimate the likelihood that lenders will go into default using data from their loan histories or profile. It is necessary to regularly and continuously assess the default risk. AI techniques assist in these efforts, However, the motivations behind these recommendations are sometimes opaque or incomprehensible, which breeds resistance to their execution. By adding justifications to AI-based ideas to make sure they are impartial and well-founded, Explainable AI allays these concerns and makes AI approaches practical for use in business. Moreover, it removes any room for human error. [9] [10]

ii) Fraud Detection: Because of the widespread use of online transactions, fraudulent transactions have become widely prevalent. Effective fraud detection is hampered by unbalanced datasets, which include a disproportionately large proportion of negative samples relative to positive instances. Moreover, the fraudsters keep on changing techniques hence, it is complicated even for modern ML and DL algorithms to keep on correctly identifying. The authors in [11] had 2 unique approaches. To address dataset imbalances, they suggested using a special nonlinear embedded clustering. Moreover, they advocated using a Deep Belief Network [12] for recognising fraudulent transactions which also provides Explainability by using SHAP which calculates the influence of each and every characteristic on the final result. Authors in [13] also discuss using Explainable AI to detect Money Laundering.

iii) Real Estate: It is important to know which features contributed maximum to a prediction to optimize our purchase. For example, a real-estate trader might want to know the percentage contribution of each feature to accordingly design the house.

3) Language Processing: : Language Processing has widespread uses right from Machine Translation to Information Retrieval Systems. Explainable AI can help identify the key features that the model is using to make its classifications. This information can be used to improve the model's accuracy or to identify areas where the model may be making incorrect classifications.

4) Software Engineering: Software Engineering is the process of planning, creating, testing, and maintaining software applications. The authors of [14] elaborate in detail how Explainable AI can be used in various Software Engineering Processes to improve efficiency!

With the growing number of applications, and increasing updates, the codes are becoming more and more complex and difficult to maintain. Software flaws are common and extremely expensive, but they are also exceedingly difficult to identify, anticipate, and fix.

i) Localizing Faults - Initially, random forest classification was used for file-level defect prediction models. Using LIME's Model Agnostics, they could determine which tokens and lines contributed to the prediction of each file using which they accurately pinpointed which lines of code were the most dangerous. [15]

ii) Identifying why defective: Using LIME's Model Agnostics, and considering 3 main features: Class and Method Declaration Lines, Contribution and Ownership, we can accurately identify the Software Defect.

A future research can be based on giving actionable inputs like based on the application type, recommend the developer regarding the ideal conditions (do's) and (dont's).

VI. Implementation

To demonstrate our concepts of Explainable AI, we implemented 3 projects:

1) Training a Waste Detection Model using WANDB AI: This focusses on the tools and frameworks part to implement Explainable AI. We trained a Waste Detection Model from a dataset on kaggle [26].This dataset contains 2,452 images in the training dataset and 522 images in the validation dataset. The images therein are full-HD with 1920 x 1080 pixels in width and height, respectively. Each image has class-related information present in it along with bounding box information. The goal is to accurately identify and detect each type of waste. While training, we used the WANDB AI tool to log various accuracy measures like Precision, Recall etc and also give a good visualization, the figures of which are shown below.
   This helps us analyze and understand how our training procedure is working real-time and save time. Another Use-case involves reproducing any model checkpoints!

2) Implementing LIME and PDP techniques: We implemented LIME technique on the IRIS dataset. The IRIS dataset is a commonly used dataset in machine learning and pattern recognition that consists of 150 samples of flowers. Each sample has four features: sepal length, sepal width, petal length, and petal width. The goal of the dataset is to classify the species of iris based on the

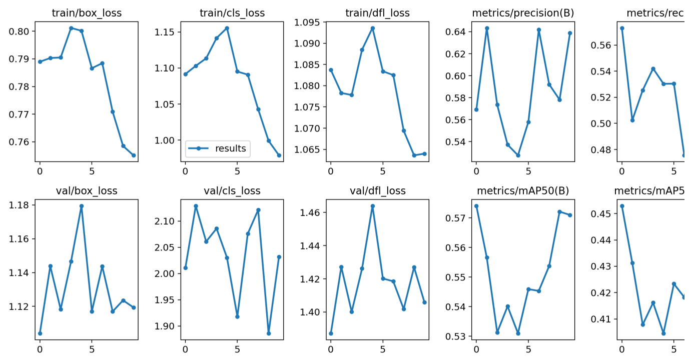

Fig. 5. Results on training YOLOV8 to detect waste

four features. We trained a Random Forest Algorithm and then to understand why a particular data belongs to a particular category, we used LIME technique.

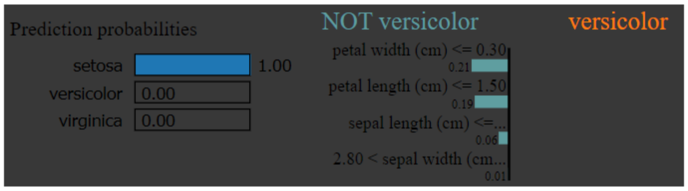

Fig. 6. Implementing LIME on IRIS Dataset to evaluate predictions

3. Implementing LIME on CIFAR-10 and MNIST Dataset MNIST dataset is a widely used benchmark dataset in the field of computer vision. It consists of 70,000 grayscale images of size 28 x 28, representing handwritten digits from 0 to 9. The dataset is split into 60,000 training images and 10,000 testing images. MNIST has been used extensively to develop and test various machine learning algorithms for image classification. In this project, LIME technique was implemented on the MNIST dataset to explain the predictions of a neural network model. LIME generates local explanations for a given prediction by approximating the decision boundary around that prediction using a set of interpretable model explanations. The resulting explanations can help understand the model's decision-making process and identify potential biases or inaccuracies. Figure 7 shows using LIME how the model trained on MNIST Dataset understands the number.

CIFAR10 dataset is another benchmark dataset in computer vision, consisting of 60,000 color images of size 32 x 32, belonging to 10 classes such as airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck. The dataset is split into 50,000 training images and 10,000 testing images. CIFAR10 has been used to evaluate the performance of various machine learning algorithms for image classification. In this project, LIME technique was also implemented on the CIFAR10 dataset to explain the predictions of a neural network model. Figure 8 shows using LIME how the model trained on CIFAR10 Dataset understands and classifies the image

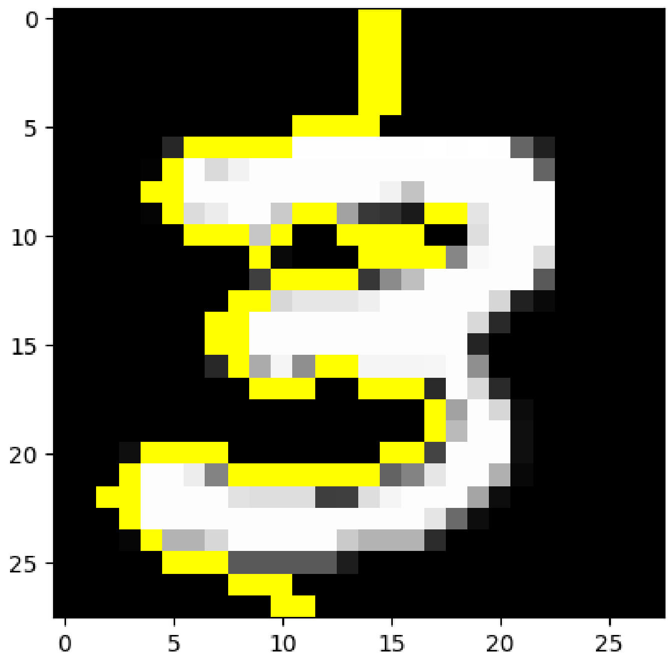

Fig. 7. Implementing LIME on MNIST Dataset

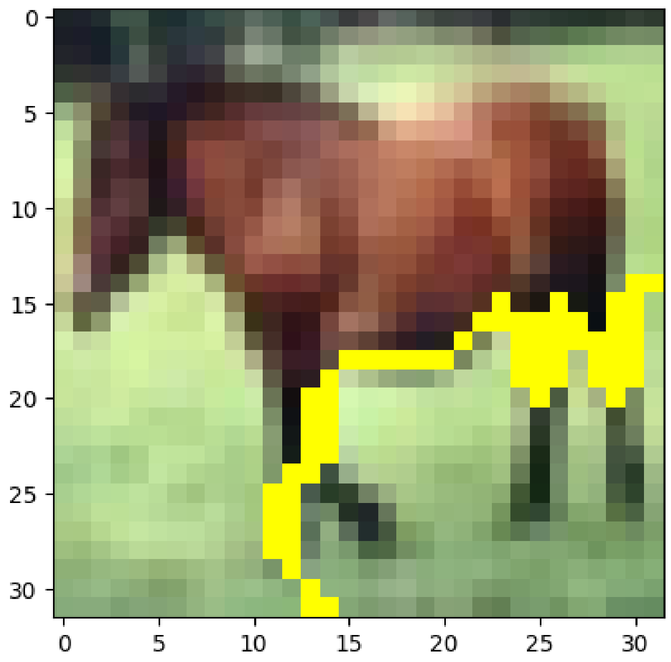

Fig. 8. Implementing LIME on CIFAR-10 Dataset

by highlighting the important pixels. To implement LIME, first, a neural network model was trained on the respective datasets using the Keras library. Then, the LIME library was used to generate local explanations for a selected image. The LIME explainer was instantiated with the hyperparameters such as kernel width and number of features. The explain instance() method was called on the explainer object with the input image, the trained neural network model, and the number of artificial data points (num samples) as arguments. The method returned a set of interpretable model explanations in the form of superpixels, which were overlaid on top of the input image to highlight the regions that contributed the most to the model's prediction. The resulting explanations were helpful in understanding the model's decision-making process and identifying potential biases or inaccuracies.

VII. Conclusion

This paper highlights the need for using Explainable AI and how it helps researchers to further optimize and understand the deep networks better in order to develop better techniques and debug them better. In order to take measures in order to measure the effectiveness and the learning of AI, a few tools have been discussed at length in this paper. These tools help in providing a better and clearer understanding of deep neural networks. We have also discussed some applications for the researchers to get a broad picture for further research scope!

References

[1] ”IBM Artificial Intelligence : What is Artificial Intelligence?”

[2] ”Model-Agnostic XAI Models: Benefits, Limitations and Research Directions — TU Delft Repositories.” https://repository.tudelft.nl/islandora/object/uuid:34785364-3a1a-4ac0-be8b-668e4fd01721, Accessed 13 April. 2023.

[3] ”CGP Grey : How do intelligent machines work?”

[4] Hui Wen Loh, Chui Ping Ooi, Silvia Seoni, Prabal Datta Barua, Filippo Molinari, U Rajendra Acharya, Application of explainable artificial intelligence for healthcare: A systematic review of the last decade (2011–2022), Computer Methods and Programs in Biomedicine, Volume 226, 2022, 107161, ISSN 0169-2607, https://doi.org/10.1016/j.cmpb.2022.107161

[5] Jiménez-Luna, J., Grisoni, F, Schneider, G. Drug discovery with explainable artificial intelligence. Nat Mach Intell 2, 573–584 (2020). https://doi.org/10.1038/s42256-020-00236-4.

[6] Ying, Z., Bourgeois, D., You, J., Zitnik, M. Leskovec, J. GNNExplainer: generating explanations for graph neural networks. Adv. Neural Inf. Process. Syst. 32, 9240–9251 (2019).

[7] Zech JR, Badgeley MA, Liu M, Costa AB, Titano JJ, Oermann EK. Variable generalization performance of a deep learning model to detect pneumonia in chest radiographs: a cross-sectional study. PLOS Med. 2018;15:e1002683.

[8] Lapuschkin S, Wäldchen S, Binder A, Montavon G, Samek W, Müller K-R. Unmasking Clever Hans predictors and assessing what machines really learn. Nat Commun. 2019;10:1096.

[9] Park MS, Son H, Hyun C, Hwang HJ (2021) Explainability of machine learning models for bankruptcy prediction. IEEE Access 9:124887–124899. https://doi.org/10.1109/ACCESS.2021.3110270

[10] Weber, P., Carl, K.V. and Hinz, O. Applications of Explainable Artificial Intelligence in Finance—a systematic review of Finance, Information Systems, and Computer Science literature. Manag Rev Q (2023). https://doi.org/10.1007/s11301-023-00320-0

[11] A. Bhowmik, M. Sannigrahi, D. Chowdhury, A. D. Dwivedi and R. Rao Mukkamala, ”DBNex: Deep Belief Network and Explainable AI based Financial Fraud Detection,” 2022 IEEE International Conference on Big Data (Big Data), Osaka, Japan, 2022, pp. 3033-3042, doi: 10.1109/BigData55660.2022.10020494.

[12] A.-R. Mohamed, G. Dahl, G. Hinton et al., ”Deep belief networks for phone recognition”, Nips workshop on deep learning for speech recognition and related applications, vol. 1, no. 9, pp. 39, 2009.

[13] D. V. Kute, B. Pradhan, N. Shukla and A. Alamri, ”Deep Learning and Explainable Artificial Intelligence Techniques Applied for Detecting Money Laundering–A Critical Review,” in IEEE Access, vol. 9, pp. 82300-82317, 2021, doi: 10.1109/ACCESS.2021.3086230.

[14] C. K. Tantithamthavorn and J. Jiarpakdee, ”Explainable AI for Software Engineering,” 2021 36th IEEE/ACM International Conference on Automated Software Engineering (ASE), Melbourne, Australia, 2021, pp. 1-2, doi: 10.1109/ASE51524.2021.9678580.

[15] J. Jiarpakdee, C. Tantithamthavorn, H. K. Dam and J. Grundy, ”An Empirical Study of Model-Agnostic Techniques for Defect Prediction Models”, IEEE Transactions on Software Engineering (TSE), 2020.

[16] J. Jiarpakdee, C. Tantithamthavorn and A. E. Hassan, ”The Impact of Correlated Metrics on Defect Models”, IEEE Transactions on Software Engineering (TSE), 2019.

[17] ”InDepth: Layer-Wise Relevance Propagation — by Eugen Lindwurm — Towards Data Science.” https://towardsdatascience.com/indepth-layer-wise-relevance-propagation-340f95deb1ea, Accessed 14 Apr. 2023

[18] van der Waa, Jasper, et al. ”Evaluating XAI: A Comparison of Rule-Based and Example-Based Explanations.” Artificial Intelligence, vol. 291, 2021, p. 103404, https://doi.org/10.1016/j.artint.2020.103404. Accessed 18 Apr. 2023.

[19] Macha, Dawid, et al. ”RuleXAI—A Package for Rule-Based Explanations of Machine Learning Model.” SoftwareX, vol. 20, 2022, p. 101209, https://doi.org/10.1016/j.softx.2022.101209. Accessed 19 Apr. 2023.

[20] Flowcast : Revolutionizing debt collection using Machine Learning, a Whitepaper.

[21] ”TensorBoard — TensorFlow.” https://www.tensorflow.org/tensorboard, Accessed 11 Apr. 2023

[22] ”Flowcast — Powering Smarter Credit.” https://www.flowcast.ai/, Accessed 12 April. 2023

[23] ”Captum · Model Interpretability for PyTorch.” https://captum.ai/, Accessed 11 April. 2023

[24] ”Introducing AI Explainability 360 — IBM Research Blog.” https://www.ibm.com/blogs/research/2019/08/ai-explainability-360/, Accessed 13 April. 2023

[25] ”WANDB AI” https://wandb.ai/site, Accessed 30 April. 2023

[26] “WaRP - Waste Recycling Plant Dataset.” WaRP - Waste Recycling Plant Dataset — Kaggle, /datasets/parohod/warp-waste-recycling-plant-dataset.

[27] Giudici, Paolo, and Raffinetti, Emanuela. ”Shapley-Lorenz EXplainable Artificial Intelligence.” Expert Systems with Applications, vol. 167, 2021, p. 114104, https://doi.org/10.1016/j.eswa.2020.114104. Accessed 4 May 2023.

[28] D. Fryer, I. Strümke and H. Nguyen, ”Shapley Values for Feature Selection: The Good, the Bad, and the Axioms,” in IEEE Access, vol. 9, pp. 144352-144360, 2021, doi: 10.1109/ACCESS.2021.3119110.

[29] Gramegna, Alex, and Giudici, Paolo. ”SHAP and LIME: An Evaluation of Discriminative Power in Credit Risk.” Frontiers in Artificial Intelligence, vol. 4, 2021, https://doi.org/10.3389/frai.2021.752558. Accessed 4 May 2023.

[30] J. Duell, X. Fan, B. Burnett, G. Aarts and S. -M. Zhou, ”A Comparison of Explanations Given by Explainable Artificial Intelligence Methods on Analysing Electronic Health Records,” 2021 IEEE EMBS International Conference on Biomedical and Health Informatics (BHI), Athens, Greece, 2021, pp. 1-4, doi: 10.1109/BHI50953.2021.9508618.

[31] Dieber, Jürgen Kirrane, Sabrina. (2020). Why model why? Assessing the strengths and limitations of LIME.

[32] ”Explainable AI(XAI) Using LIME - GeeksforGeeks.” https://www.geeksforgeeks.org/introduction-to-explainable-aixai-using-lime/, Accessed 1 May. 2023.

[33] Kohlbrenner, Maximilian, et al. ”Towards Best Practice in Explaining Neural Network Decisions with LRP.” ArXiv, 2019, /abs/1910.09840. Accessed 4 May 2023.

[34] ”Explainable AI (XAI): A survey of recents methods, applications and frameworks — AI Summer.” https://theaisummer.com/xai/, Accessed 1 Jan. 1970.

[35] Mayor Torres, Juan and Medina-DeVilliers, Sara and Clarkson, Tessa and Lerner, Matthew and Riccardi, Giuseppe. (2023). Evaluation of Interpretability for Deep Learning algorithms in EEG Emotion Recognition: A case study in Autism. Artificial Intelligence in Medicine.

[36] Simonyan, Karen, Andrea Vedaldi, and Andrew Zisserman. ”Deep inside convolutional networks: Visualising image classification models and saliency maps.” arXiv preprint arXiv:1312.6034 (2013).