International Journal of Multimedia Data Engineering and Management
Volume 14 • Issue 1

Automation of Explainability Auditing for Image Recognition

Duleep Rathgamage Don, Kennesaw State University, USA*
Jonathan Boardman, Kennesaw State University, USA
Sudhashree Sayenju, Kennesaw State University, USA
Ramazan Aygun, Kennesaw State University, USA
https://orcid.org/0000-0001-7244-7475
Yifan Zhang, Kennesaw State University, USA
Bill Franks, Kennesaw State University, USA
Sereres Johnston, The Travelers Companies, Inc., USA
George Lee, The Travelers Companies, Inc., USA
Dan Sullivan, The Travelers Companies, Inc., USA
Girish Modgil, The Travelers Companies, Inc., USA

ABSTRACT

XAI requires artificial intelligence systems to provide explanations for their decisions and actions for review. Nevertheless, for big data systems where decisions are made frequently, it is technically impossible to have an expert monitor every decision. To solve this problem, the authors propose an explainability auditing method for image recognition whether the explanations are relevant for the decision made by a black box model, and involve an expert as needed when explanations are doubtful. The explainability auditing system classifies explanations as weak or satisfactory using a local explainability model by analyzing the image segments that impacted the decision. This version of the proposed method uses LIME to generate the local explanations as superpixels. Then a bag of image patches is extracted from the superpixels to determine their texture and evaluate the local explanations. Using a rooftop image dataset, the authors show that 95.7% of the cases to be audited can be detected by the proposed method.

KEYWORDS

Classification, Computer Vision, Deep Learning, Explainability, Explainable Artificial Intelligence, Image Recognition

INTRODUCTION

During the last decade, artificial intelligence has claimed many achievements matching or surpassing human-level performance in some application domains such as object recognition. The performance of deep learning algorithms has been boosted with the introduction of additional layers or residuals

DOI: 10.4018/IJMDEM.332882 *Corresponding Author

from earlier layers continued to improve the performance (He et al., 2016). However, as the complexity of models has increased, the model interpretability has decreased, and as a result such black box models have become problematic in high-stakes decision-making domains, where safe and reliable performance are critical due to the high cost associated with errors (Guidotti et al., 2018). This is exacerbated by the realization that the patterns learned by discriminative deep architectures are less robust than what previously thought and vulnerability to adversarial attacks is the rule rather than the exception. In some cases, changing a single pixel is enough to fool a trained model (Su et al., 2019). Attacks can even be carried out in the real world by, for example, attaching a piece of black tape to a stop sign (Eykholt et al., 2018).

There are many ways we may wish to employ Explainable Artificial Intelligence (XAI) methods, and the choice of method and nature of the explanation should be informed by the problem context. Many different approaches to interpretability have emerged to meet this demand, and they can be categorized along several dimensions such as global vs. local, model-specific vs. model-agnostic, and intrinsic vs. post-hoc (Molnar et al., 2020; Rai, 2020). For deep neural networks, intrinsic interpretability may not be attainable. It has been noted that model interpretability and model flexibility or accuracy tend to be inversely related (Freitas, 2014). As the complexity of classification models increases, high accuracies in predictions can be achieved, but interpretability suffers. For example, Slack et al. (2019) investigate and conclude that decision trees and logistic regression are locally interpretable models while neural networks are not.

In contrast to global explainability techniques, which seek to explain the entire model (either by designing the model to be intrinsically interpretable or through an interpretable surrogate model), local explainability techniques provide explanations for individual predictions. Ribeiro et al., (2016) introduce local Interpretable Model-Agnostic Explanations (LIME) as a simple local explainability technique that generates simulated data points using random perturbations in the neighborhood of an instance to be predicted by the black-box model and fits a weighted linear regression on the simulated data to create explanations for the prediction. One of the main advantages of LIME is being model agnostic and hence, it may diminish the need for interpretable models. Usually, local explainability techniques provide interpretations of how an individual sample is analyzed, and the analysis may convince an expert to determine whether the model focuses on the right components or segments of data to make the decision. For example, Ribeiro et al. (2016), show that husky vs wolf image classification was done based on the signal of the background rather than focusing on the features of the animal. In other words, the learned model recognizes a domestic environment (e.g., home) compared to a wild environment (e.g., forest). This helps the expert determine whether the learned model is reliable or not, and this makes it a spectacular tool for individual analysis of samples. However, if the goal is to uncover systematic issues with the model, an expert must check the explanation of every sample.

Deep learning models may be trained on huge datasets of which the size may range from terabytes to petabytes. Monitoring explanations of these models by hand during the training process is out of the question. Even whenever it is possible, what matters is how those trained machine learning models behave in the wild for previously unseen data since critical decisions may rely on these models. Regardless of the possibility of manual checking, such a costly approach voids one of the main benefits of using machine learning–scalability.

This paper presents an automated explainability audit framework known as *ExplainabilityAudit* (DR Don et al., 2022) to investigate local interpretability in image recognition. As shown in Figure 1, the proposed method analyzes the reliability of classification by processing the explanations. After analyzing the explanations, it returns *satisfactory* if the explanations are good or *weak* if the explanations are poor. This technique requires training another model based on explanations. If this audit model determines that an explanation of a decision by the main model is weak (not reliable), this would require the involvement of a human expert to analyze the prediction and explanation. A human expert would only be required to step in for relatively few cases instead of potentially thousands or millions.

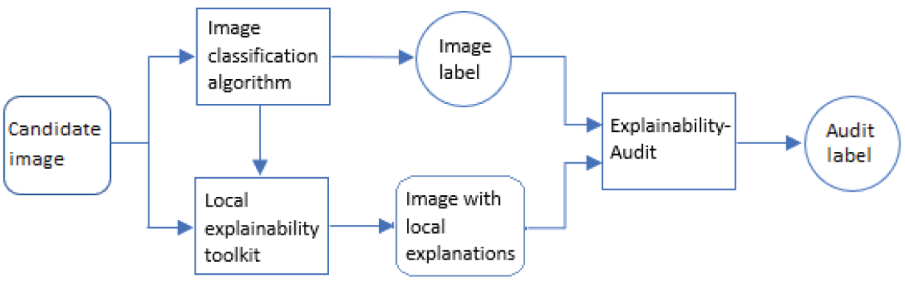
We introduce a version of *ExplainabilityAudit*, that uses the original LIME toolkit to generate local explanations for rooftop images that are classified by a deep convolutional neural network. The goal of the rooftop classification is to distinguish flat roofs among various other types of roofs in a nadir rooftop image dataset where the footprint of the rooftops is often surrounded by various neighboring objects such as ground, trees, vehicles, driveways, etc. In this case, the regions other than the rooftop are considered to be the background. The proposed method uses LIME in the following manner. First, it splits a candidate image into many superpixels and creates a synthetic dataset using random perturbations of the candidate image. Then a locally weighted interpretable linear model is trained on the new image dataset. The superpixels that correspond to the highest estimated coefficients are chosen to be the top local explanations. Then our method analyzes the texture features of the top local explanations to determine if they belong to the rooftops or background. Then the audit label *satisfactory* is produced when most of the local explanations represent the rooftops or similar objects. Our experiment is limited to extracting the largest segment of the local explanations for each validation image. In determining the local explanations, we demonstrate that a patch-based auditing approach to analyze texture features is more efficient than applying Convolutional Neural Network (CNN) algorithms on the local explanations as a whole.

RELATED WORK

The high accuracy of deep learning models is not necessarily an indication of extracting and learning proper features. Deep learning models depending on unreliable and ungeneralizable features may yield critical Type I and Type II errors, which could lead to adverse effects, especially in medical applications (Burkart & Huber, 2021; Holzinger et al., 2019). Explainability is the extent to which the internal mechanics of a machine or deep learning model can be explained in terms more understandable to humans (Rosenfeld & Richardson, 2019). Especially for artificial neural networks, interpreting how the model behaves with respect to input data is not simple. Although fully interpreting neural networks directly in terms of their features is not as straightforward as in regression models, explainability tools give a better picture of a model's patterns. Thus, explainability tools enable us to convert black box models more into grey boxes.

Explainability generally falls into two main types of tasks: model understanding (global explainability) and decision understanding (local explainability). Model understanding or global explainability involves finding out how the model behaves for general data. Particularly, this means the task of recognizing the patterns in its predictive features or model parameters on classification.

Figure 1. The role of explainability audit in image recognition pipeline



Note: First, the image classification algorithm predicts the label of a candidate image. Then the local interpretability toolkit generates local explanations for each prediction. Finally, the *ExplainabilityAudit* algorithm works on both the image label and respective local explanations to decide if the prediction is reliable.

On the other hand, decision understanding or local explainability is concerned with the task of the model behavior only on a particular data instance. Here, the aim is to find how the input features affect a single data point's classification. Most of the research in explainability has focused on decision understanding. Simple tools like *what-if* (Wexler et al., 2019) offer dashboards called data point instance editor and feature statistics which help deduce explainability indirectly. Another popular method to generate explanations is using Shapley values (Lundberg & Lee, 2017). The concept of Shapley values derives from game theory and is based on probability theory. Shapley values are the average marginal contribution of a feature across all possible coalitions. AWS SageMaker Clarify (Hardt et al., 2021) uses Shapley values to explain a black box model.

Deep Learning Important Features (DeepLIFT) by Shrikumar et al., (2017) is a method that is used on a fully trained Keras model. In a single backward pass, importance scores are calculated for all input features. Thus, it is computationally efficient. The authors gave an example of a piece of text representing a genome sequence and a neural network for the classification of the sequence. Every alphabet (input feature) in the sequence gets a score based on the backward pass calculation. The scores indicate how proportionally or inversely each alphabet affected the classification label. DeepLIFT does not require any manual intervention. Contrarily, TCAV (Testing with Concept Activation Vectors) by Kim et al., (2017) uses pre-defined human concepts to train linear classifiers that separate those concepts from random inputs. If the concepts were learned, it indicated that correct explanations were learned by the model too. Kim et al. (2017) demonstrated the tool with a computer vision task. For example, the manually created concept of stripes includes a group of images of various stripe patterns. The group of random input includes images that are not stripes. TCAV learns a linear classifier that separates stripes from random input. Another tool called Path-Integrated Gradients (Sundararajan et al., 2017) uses a pre-defined human baseline input to calculate attribution scores for each feature. The baseline input is necessary to characterize situations when the absence of a feature can be informative. Alternatively, counterfactual methods generate adversarial scenarios or instances to explain the prediction with more stability than most feature importance-based XAI methods (Vermeire, et al., 2022, Singla, et al., 2023).

The main reason why deep learning models are considered black boxes is that their behavior is not linear, and hence it is hard to come up with a global but simple interpretation. For decision understanding of a single data instance, it is not necessary to understand the complete non-linear behavior of the model. Thus, Ribeiro et al., (2016) provided an explanation of the model around the local region of the data instances being scrutinized. These explanations have locally linear fidelity: linear behavior of the model in the vicinity of the prediction instance. LIME learns the locally linear classifier by minimizing a loss function that minimizes the error between the actual model in the region and the explainable model. More comprehensive explainability tools like LIT (Tenney et al., 2020), and ELI5 (Korobov, 2017) include LIME to enhance model explainability.

However, LIME has some potential pitfalls. Therefore, several important modifications have been introduced in the past few years to address those issues. DLIME (Zafar & Khan, 2019) is a deterministic version of LIME in which the random perturbation is replaced with agglomerative hierarchical clustering to create clusters within the training dataset and then applying K-Nearest Neighbor (KNN) to find the relevant cluster for the new observation. Then more stable explanations are generated by training a linear model over the selected cluster. ALIME (Shankaranarayana & Runje, 2019) applies an alternative approach to reduce instability in generated explanations while maintaining local fidelity. In this method, many synthetic data points are sampled from a Gaussian distribution and weighted for the locality by a denoising autoencoder. Lee et al. (2019) stated that the mean and the standard deviation of weighted superpixels of a test image produced by LIME demonstrate that the generated explanations are relatively stable. MPS-Lime (Shi et al., 2020) is a modified perturbated sampling for LIME that avoids correlation between the superpixels. In this method, the superpixels are represented by an undirected graph, and the perturbed sampling is formalized as a clique-set construction problem. Also, BayLIME (Zhao et al., 2020) is a Bayesian

extension to the LIME framework that applies prior knowledge and Bayesian reasoning to enhance the stability of the explanations.

METHODOLOGY

In this section, we discuss the proposed method by presenting its architecture and providing explanations for the design of each algorithm used.

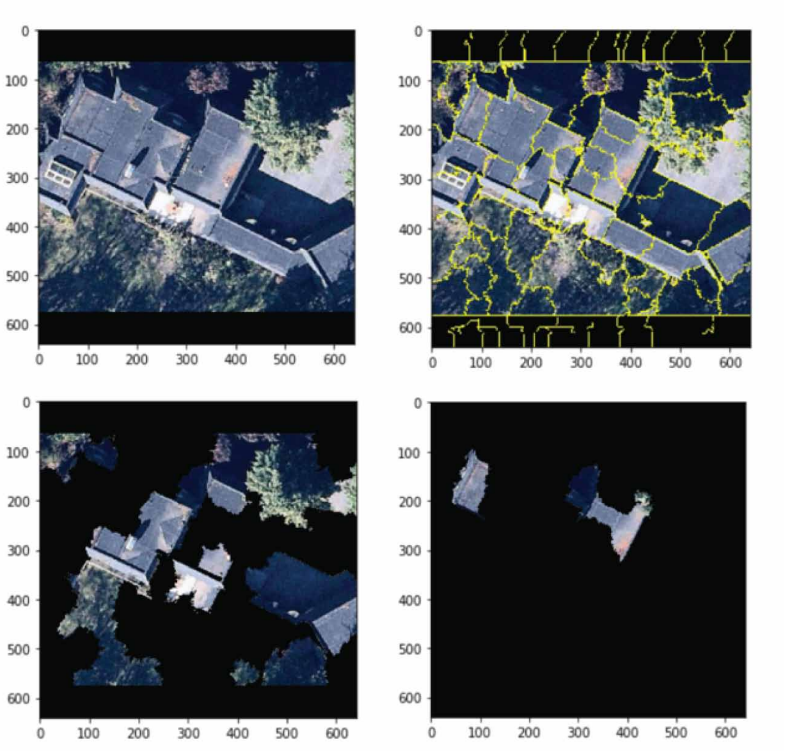
Local Explainability Tool Kit

Although the proposed method may be integrated with any upgraded version of LIME or other explainability toolkits, we selected the original LIME framework as the local explainability toolkit in the proposed method.

Local Interpretable Model-Agnostic Explanations (LIME)

We assumed that the image recognition algorithm was a deep neural network represented by a real function f such that $f : X \to Y,\ X \subseteq \mathbb{R}^d$ and $Y \subseteq \mathbb{R}$, where d is the number of RGB pixels of the image predicted. For candidate image $x \in X,$ let $f\left(x\right)$ to be the probability that x belongs to a certain class. In our method, LIME would split x into d' number of superpixels such that each superpixel was a contiguous patch of similar pixels as shown in Figure 2. An interpretable

Figure 2. LIME framework for generating local explanations



Note: Top left: A candidate image x. Top right: The distribution of superpixels. Bottom left: The random perturbation z. Bottom right: The image with local explanations ℓ. (Aerial images courtesy of Nearmap US, Inc.)

representation of x given by a binary vector x' was obtained such that $x' = \{0,1\}^{d'}$, where 0 and 1 represent the absence and presence of superpixels respectively. A random perturbation z was generated by blacking out some superpixels in x. Its interpretable representation z' is also a binary vector as previously mentioned. Then a set of N number of random perturbations Z was generated by sampling uniformly at random around x' in order to train a locally weighted linear regression model $g \in G$, where f is the class of potential interpretable models. Also, the perturbations in the original presentation were recovered and predicted by using the image recognition algorithm f to obtain a target set $f(x)$ for the prediction of x. For a weight function π_x, an exponential kernel defined on some distance function D, was introduced as a proximity measure $\pi_x(z) = \exp\left(-D(x,z)^2 / \sigma^2\right)$, where $D(x,z)$ is the distance between x and z, while σ is the kernel width (Ribeiro et al., 2016). To measure how unfaithful the local linear model g to the image recognition algorithm f, the following loss function η was used:

$$\eta\left(f, g, \pi_x(z)\right) = \sum_{z, z' \in Z} \pi_x(z)\left(f(z) - g(z')\right)^2 \tag{1}$$

Note that every linear model g is not simple enough to be interpretable. Therefore, another loss function Ω known as the measure of complexity was introduced to determine the complexity of the linear model g. $\Omega(g)$ is the number of nonzero weights for a linear model. It was added to the loss in equation (1.1) to obtain the total loss. Finally, a linear model ξ was trained on the dataset consisting of z' and $f(z)$. The local explanations were obtained from the linear model that minimizes the total loss:

$$\xi(x) = \arg\min_{g \in G}\left(\eta\left(f, g, \pi_x\right) + \Omega(g)\right) \tag{2}$$

Images With Local Explanations

Setting ξ to be a linear regression in equation (1.2), the top local explanations were extracted as superpixels that represent the highest k estimated regression coefficients. The set of local explanations generated for the image x by the image recognition algorithm f can be visualized as a single image denoted by ℓ. An audit model was built to classify the image ℓ by admitting the class of x to evaluate the local explanations generated. The purpose of the audit model is to determine whether f has focused on the proper regions in the image or not.

Although LIME shows some instability in the generated explanations, a typical image ℓ can be considered as a sparse representation of the corresponding image x, since a very high percentage of pixels in x is blacked out in ℓ. The presence of an overwhelming black background in ℓ could aggravate the task of feature extraction. Thus, we attempted two different approaches presented in the following subsection.

Explainability Audit Method

Superpixel-Based Auditing vs. Patch-Based Auditing

We studied two possible techniques to analyze the local explanations produced by the local explainability toolkit: Superpixel-based auditing and patch-based auditing. In superpixel-based auditing, we have considered a superpixel of local explanations as a whole can be satisfactory for auditing. Even though this is rather an intuitive method, the selected superpixels need to be

preprocessed before feeding into CNN. Thus, a fixed-size bounding box should be used to crop the superpixels and generate a set of images. A major problem of this method arises due to the wide range of sizes of the superpixels. A possible solution to address this issue is to upsample or downsample the selected superpixels until they barely fit the bounding box and then apply padding pixels to fill the background. However, the upsampling of images by a great percentage could distort the features of the original class of the low-resolution image as the upsampled images possess high-variance information (Menon et al., 2020). This could eventually result in poor performance of the CNN classifier. Therefore, our experiments only focused on the patch-based auditing method. This approach focused on some superpixels of the local explanations that are large enough to contain the texture information necessary to identify them correctly. In fact, the patches in our use case are two small to be classified using CNN. Therefore, we applied the following preprocessing of the local explanations to enable *ExplainabilityAudit* for the image ℓ. as shown in Figure 3.

Image Segmentation

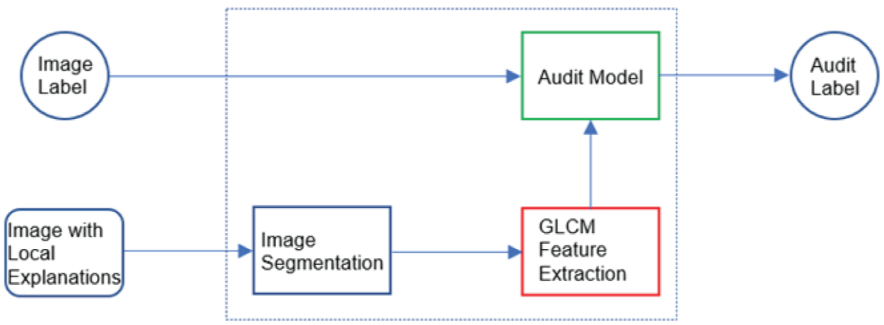
First, the image ℓ was converted to an 8-bit greyscale image in which each pixel is represented by an integer 0 – 255. Then appropriate masking was applied to the greyscale image to determine the extreme outer contour of each segment of the local explanations. Also, it is possible that these image segments contain several neighboring superpixels. For our experiments, only the image segment with the largest contour was selected and then a rectangular bounding box was applied to extract the image segment. This preprocessing step can technically be applied to each image segment that contains at least a single $p \times p$ image patch, where p is the patch size in pixels. It outputs grayscale image segments.

GLCM Feature Extraction

In this preprocessing step, the grayscale image segments were used to produce image patches. From each grayscale image segment, the maximum number of qualified image patches were produced. Then the Grey Level Co-occurrence Matrix (GLCM) features were extracted from each image patch to construct the GLCM texture feature dataset as shown in Figure 4. The techniques used in this step are outlined below:

- **Grid Search for Maximizing Number of Patches:** The local explanations present in the grayscale image segments have rough boundaries separating the explanations from the black background. First, we split the grayscale images into a grid containing $p \times p$ cells. These cells were then used

Figure 3. The architecture of the explainability audit



Note: The main components of *ExplainabilityAudit* method. The local explanations are segmented to produce GLCM features. The audit model evaluates the GLCM features with respect to the image label to predict the audit label.

to obtain image patches. An image patch can often contain pixels from the black background. Thus, we introduced a threshold θ for the highest percentage of black pixels to be allowed in a patch. To maximize the number of patches that can be obtained from a given grayscale image using the grid, we calculated all possible solutions performing a grid search. Initially, the point origin $\left(0,0\right)$ of a virtual grid of $p \times p$ cells was aligned with the top left corner of a given grayscale image segment. Assuming the coordinates of the top left corner as $\left(m,n\right)$, where m,n are integers. We changed the position of the grayscale image segment relative to the grid such that $0 \le m \le p,$ and $0 \le n \le p.$ At each relative location, the number of valid image patches were computed and the maximizer $\left(m,n\right)$ was determined. For this study we used $p = 10,$ and produced the image patches choosing the maximizer for the origin of the grid.

- **Creating GLCM Texture Feature Dataset:** We realized that 10×10 image patches with 8-bit are not suitable to extract the widely used GLCM features (Hall-Beyer, M., 2000) available in scikit-image Python library and presented in Table 1. The GLCM of such an image patch would be a sparse matrix and some Haralick features might decrease in amplitude and generate a poor representation of the texture of the image patch (Rosenfeld & Richardson, 2019). Therefore, we transformed the selected greyscale image patches to 4-bit to produce its GLCM as shown in Figure 4.

For a greyscale image patch of $k \times k$ pixels, the GLCM is a square matrix that needs to store the frequency at which pairs of pixels with certain values in a given spatial orientation.

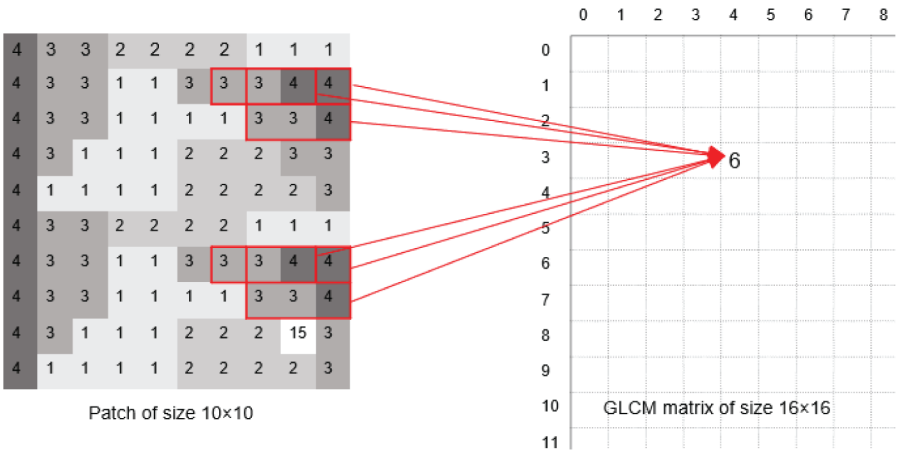
Thus, the size of the GLCM simply becomes the bit depth of the greyscale image patch. We used two parameters to construct the GLCM: distance and angle. The distance measured the magnitude of the displacement between two pixels (from 2 to $k-1$) and the angle measured

Table 1. The selected GLCM features

GLCM Feature	Formula
Dissimilarity	$\sum_{i,j=0}^{n-1} p_{ij} \left\lvert i-j \right\rvert$
Correlation	$\sum_{i,j=0}^{n-1} p_{ij} \left[\left(i-\mu_i\right)\left(j-\eta_j\right) / \sqrt{\sigma_i^2 \sigma_j^2} \right]$
Contrast	$\sum_{i,j=0}^{n-1} p_{ij} \left(i-j\right)^2$
Homogeneity	$\sum_{i,j=0}^{n-1} \left(p_{ij} / \left(1+\left(i-j\right)^2\right)\right)$
Angular Second Moment (ASM)	$\sum_{i,j=0}^{n-1} p_{ij}^2$
Energy	$\sqrt{ASM}$

Note: In formulae, p_{ij} is the probability of values i and j occurring in adjacent pixels in the original image within the window defining the neighborhood. n is the order of the GLCM.

Figure 4. The construction of GLCM



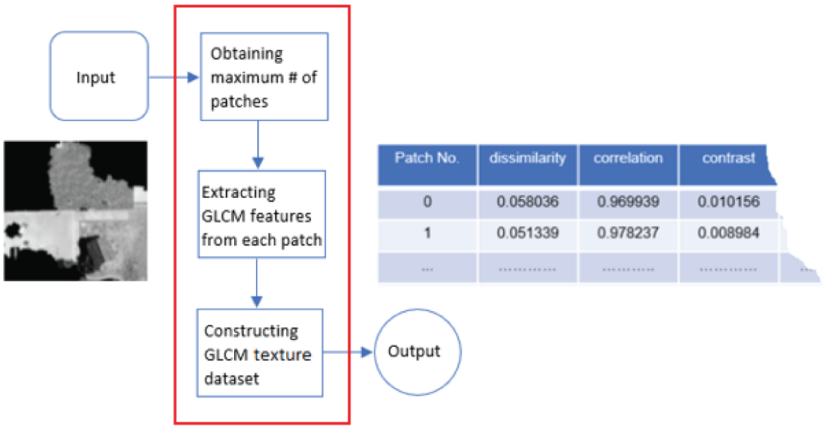
Note: A 10×10 pixel image patch is used to construct a GLCM of size 16×16 with distance 2 and angle 0°. There are 6 readings for the pixel pair (3, 4) at a displacement of 2 along horizontal direction.

the displacement's direction ($0°, 45°, 90°, 135°$) made with the axis. The extracted GLCM features can be used to construct a GLCM texture feature dataset for each candidate image as shown in Figure 5.

Audit Model

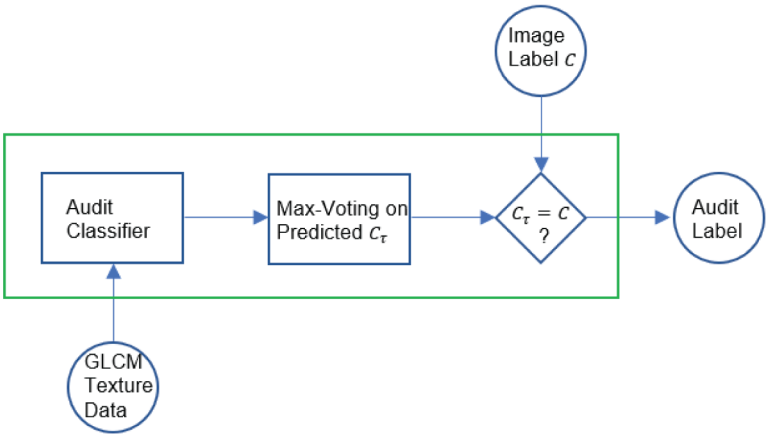
In the first stage of the audit model shown in Figure 6, a supervised machine learning algorithm called *audit classifier* was used to classify each patch represented by the GLCM texture feature dataset into the union of the set of image labels and the background. In the second stage, the entire explanation

Figure 5. The GLCM feature extraction



Note: The input grayscale image is transformed into a bag of image patches. Then some GLCM features are extracted from each image patch to construct the GLCM dataset. (Aerial images courtesy of Nearmap US, Inc.)

Figure 6. The audit model



Note: The proposed audit model: The audit model utilizes a multimodal architecture in which an audit classifier applies the max-voting strategy on prediction of patches and the result C_τ is compared with the predicted image label C. If both are equal, then the audit label is satisfactory and weak otherwise.

was classified based on max-voting of the predicted patches. The next step was to compare this result with the image label. If the result matched the image label, then the explanation was considered satisfactory or weak otherwise. In the output, the satisfactory class indicates that the local explanation supports the prediction of x by f. Whereas the weak class denotes that it may not be possible to make a correct decision or decision could be unreliable.

Figure 7 summarizes the proposed method by providing the algorithm of the Explainability Audit method.

EXPERIMENTS AND EVALUATION

In this section, we explain the rooftop dataset used in the experiments, the tuning of LIME algorithm, and the classifiers used for auditing explanations. Then we provide the results of our experiments before the discussion. To conduct experiments, we used AWS/Amazon SageMaker instance p3.2xlarge. The machine learning models were trained using TensorFlow and Keras 2 on Python 3 with CUDA 9.0 and MKL-DNN.

Experimental Setup

Datasets

The original dataset used in this study was exclusively maintained in-house by The Travelers Indemnity Company with the courtesy of Nearmap US Inc., Therefore, either this dataset or its citation is not publicly available. The original dataset has a nadir rooftop imagery consisting of 3715 RGB images split into 2956 training images and 759 validation images. The images have a fixed size of 640 × 640 pixels. The label sets have two nearly balanced classes: flat and non-flat. Each rooftop image was preprocessed to produce a polygonal bounding box consisting of the footprint of the rooftop and possible background objects. This dataset was used to train our image recognition (rooftop detection) algorithm. To extract patches, another dataset was created by randomly selecting and processing some 200 training images. In this case, the image patches were extracted from rectangular regions of either

Figure 7. The algorithm of explainability audit method

Algorithm 1: The patch-based ExplainabilityAudit method

Data: image ℓ: explanation of image x; c: the predicted label of image x; θ: threshold for the fraction of non-black pixels; $p \times p$: patch size, k: the number of top local explanations
Result: prediction of image x is reliable or prediction of image x is not reliable
$S \leftarrow \{\}$;
$patch \leftarrow \{\theta, p\}$;
$\ell \leftarrow grayscale(\ell)$; /* Apply a grayscale filter */
$E \leftarrow local\_explanations(\ell)$; /* Get the set of $t \leq k$ explanations (segments) */
for e_i *in* E **do**
  $S \leftarrow e_i$ such that $e_i \geq p$; /* Use valid segments of local explanations */
end
for $s_i \in S$ **do**
  $B_i \leftarrow \{\}$;
  for i *in* $1,\ldots,t$ **do**
    maximize n_i subject to $n_i \times p \leq s_i$;
    $B_i \leftarrow patches(s_i, n_i)$;
  end
  $\phi \leftarrow \{\}$; /* GLCM feature dataset */
  for $patch \in B_i$ **do**
    convert *patch* into 4-bit;
    $\phi \leftarrow$ extract_GLCM_features(*patch*) ; /* GLCM feature extraction */
  end
end
for $r \in \phi$ **do**
  h: $r \mapsto C_j$ $(C_j \in \mathbb{N})$; /* Classify patches using the audit model h */
end
$\tau = \underset{j \in 1,\ldots,y}{\operatorname{argmax}} \left|\{C_j\}\right|$; /* $c \leq y$ */
if $C_\tau = c$ **then**
  prediction of image x is *satisfactory*; /* For binary classification, c indicates satisfactory */
  ;
else
  if $C_\tau \neq c$ **then**
    prediction of image x is *weak*;
  end
end

rooftops or background but not both. Also, a randomly selected sample of 88 images belonging to the original validation dataset was drawn to construct the evaluation dataset.

When extracting image patches, the threshold of accepting black pixels $\theta = 5\%$ was used. The training GLCM texture dataset consisted of 7162 observations, each representing a training patch belonging to one of the 200 training images and the GLCM features given in Table 1 were extracted with all possible combinations of distance 2 and 3 and recommended angles were used. The most effective combination of distance 2 and angle 0° was used train the audit classifier. A random sampling was used to create a training GLCM texture feature dataset with 0.7 observations, and the rest was used for validation of the audit classifier. Note that our experiments were limited to the largest segment of the local explanations in each image ℓ.

Tuning LIME Algorithm

The following parameter settings were used for the LIME algorithm. The number of superpixels d', generated for each image x was in the range (80, 120). The number of random perturbations N was set to be 1000. The weighted linear g was linear regression. In the weight function π_x, the distance function D and the kernel width σ were cosine and 0.25 respectively. Setting $k = 8$ led to the extraction of the best local explanations.

Rooftop Recognition

The rooftop recognition algorithm was a deep neural network created by modifying a pretrained ResNet50 architecture using transfer learning. In this process, the top layer of the ResNet50 was replaced by three fully connected layers including a new top layer for binary classification. Then only the classification head was trained with the rooftop training dataset. After training for 15 epochs, the performance of this image recognition algorithm was validated using the complete test dataset, and the following performance measures were observed as follows. Accuracy: 0.8326, Precision: 0.7904, Recall: 0.8225, F-1: 0.8109, and ROC: 0.9159.

Audit Classifier

For the audit classifier, three binary classification algorithms known as Support Vector Machine (SVM), Artificial Neural Network (ANN) with one hidden layer and 10 neurons per layer, and K-Nearest Neighbor (KNN) were used. These algorithms were trained on the same training GLCM dataset using the following hyperparameters. For SVM with RBF kernel, gamma and cost were chosen to be 1 and 1000 respectively. For ANN built using Scikit Learn MLPC module, the default batch size, optimization, and learning rates were applied. The number of epochs is set to be 1000. For KNN, K is selected to be 15.

Results

Since the difference between image labels was ignored in this experiment, the local explanation was considered satisfactory if most patches belong to a rooftop. Also, the local explanation was considered weak if most patches belonged to the background. We tested the audit model with three different machine learning methods, using as the audit classifier: SVM, KNN, and ANN, applying the same GLCM training and GLCM validation datasets. The results for classifying individual patches are shown in Table 2.

For each method in Table 2, the experiments were conducted under different hyper-parameter settings, and for each method, only the best performance was presented. The SVM equipped with RBF kernel exhibited the best performance in predicting image patches of rooftops and backgrounds. The corresponding cost and gamma were noted as 1000 and 1 respectively. Table 3 shows the performance of the audit model on the validation images. In this case, the audit model used the two-stage prediction presented in Figure 6. The SVM-based audit classifier outperformed the other variants by a significant margin in accuracy, recall, and F1 score. Therefore, we decided to analyze SVM further as the most effective audit classifier and conducted 5 fold cross-validation. The resulting mean values of accuracy, precision, recall, and F1 score were computed as 86.6, 88.3, 95.7, and 91.8 respectively.

Table 2. Performance of the different audit classifiers on the validation GLCM texture feature dataset

Audit Classifier	Accuracy	Precision	Recall	F1
SVM $\left(c = 10^3, \gamma = 1\right)$	87.5	88.2	97.1	92.4
KNN $\left(k = 15\right)$	68.2	95.6	62.3	75.4
ANN $\left(iter = 10^3\right)$	79.5	91.8	81.2	86.2

Table 3. Performance of the audit model using different audit classifiers on the evaluation dataset

Method	Accuracy	Precision	Recall	F1
SVM $\left(c = 10^3, \gamma = 1\right)$	80.8	79.1	86.0	82.4
KNN $\left(k = 15\right)$	73.2	74.2	75.0	74.6
ANN $\left(iter = 10^3\right)$	79.1	78.5	82.8	80.6

DISCUSSION

Audit of explainability is a type of sanity check for the original classifier. The major purpose of this auditing is to detect cases where the original classifier is likely to misclassify. However, this may lead to cases where validation by a human expert may be deemed unnecessary although validation could be beneficial, or vice versa. In this section, we cover four cases with respect to the quality of audit and explanation for validating results by a human expert as given in Table 4. In this discussion, rather than focusing on successful auditing, we provide one example per case. Figure 8 and Figure 9 show validation images, selected segments of the local explanation, with corresponding image patches. Any image patch predicted as the rooftop is marked with a green pixel, and any image patch predicted as the background is marked with a red pixel.

Case 1 - Validation is recommended: The top row of Figure 8 illustrates a local explanation indicating a view of a rooftop obstructed by branches or shadows of trees. In this case, the audit classifier technically classifies the segment of local explanation as weak since most image patches are like the ones that come from the background. Regardless of the original model classification, such an image requires additional review to avoid errors. Hence, validation is recommended.

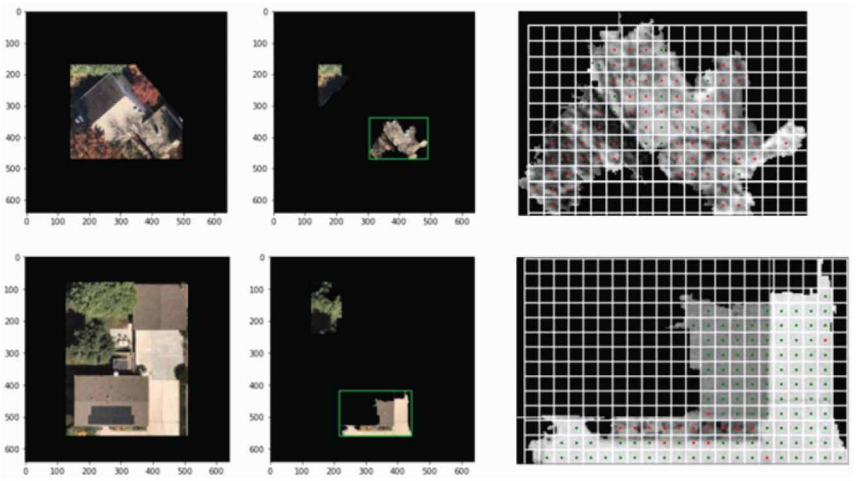
Case 2 - Validation is unnecessary: In the bottom row of Figure 8, the audit classifier predicts this explanation as satisfactory. In this case, the image has rooftop-like regions including the rooftop and the driveway. Since the driveways have similar textures as flat roofs this explanation is considered satisfactory. We should note that this auditing does not check the ability of the original classifier to distinguish a driveway from a flat roof. This is rather an indication that the classifier analyzes proper regions from the image. Since the classifier analyzes proper regions for determining the rooftop type, validation is unnecessary.

Case 3 - Validation is missed: The top row of Figure 9 illustrates a segment of the local explanation revealing the background but classified incorrectly as satisfactory. Normally, it would be beneficial to validate this case by an expert regardless of the original model's classification.

Table 4. Cases of validation

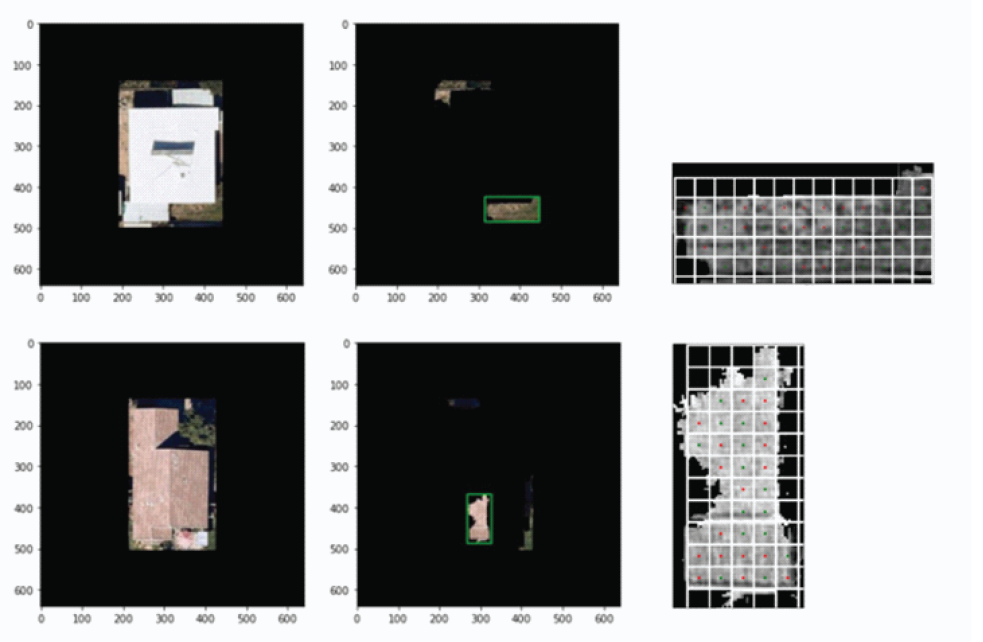
Validation Case	Audit Result	Ground Truth
Recommended	Weak	Weak
Unnecessary	Satisfactory	Satisfactory
Missed	Satisfactory	Weak
Extraneous	Weak	Satisfactory

Figure 8. Results of the explainability audit: Special Cases 1 and 2



Note: Special cases of detection and no review. From left to right: image x, image ℓ, and audit result of the local explanations. Top row: shadows or branches of trees obstruct the rooftop in the image ℓ. Bottom row: background very similar to some flatroofs confuse the classifiers. (Original aerial images courtesy of Nearmap US, Inc.)

Figure 9. Results of the explainability audit: Special Cases 3 and 4



Note: Special cases of possible miss and unnecessary review. From left to right: image x, image ℓ, and audit result of the local explanations. Top row: smooth texture and unsupportive color variance lead to a possible miss. Bottom row: rough texture and unsupportive color variance leads to unnecessary review. (Original aerial images courtesy of Nearmap US, Inc.)

Case 4 - Validation is extraneous: The bottom row of Figure 9 illustrates a segment of the local explanation revealing the rooftop but classified as weak. This leads to an unnecessary review by an expert.

CONCLUSION AND FUTURE WORK

In this paper, we propose a framework for the automation of auditing local explainability in image recognition. As the volume of image data increases, it is impractical for a human expert to check the local explanation for each prediction made by the image recognition system. Random or arbitrary checks are insufficient to guarantee an overall local explanation. The proposed method analyzes whether the right segments or components of images are processed by the image recognition models to make the prediction. Our experimental results confirm that the current version of the proposed method is capable of predicting the reliability of the image recognition algorithm with a satisfactory recall of 95.7%. In the future, the possibility of integrating different explainability toolkits is prominent. Further, the experiments need to be conducted in multiclass settings, identifying the weaknesses of the image recognition algorithm with respect to different class labels.

AUTHOR NOTE

The work was supported primarily by The Travelers Indemnity Company. The opinions, findings and conclusions or recommendations expressed in this material only reflect those of the authors in their individual capacities. Data collection and image annotation were conducted by an in-house team. A part of this research was presented at the 2022 IEEE 23rd International Conference on Information Reuse and Integration. We have no conflicts of interest to disclose.

Correspondence concerning this article should be addressed to Duleep Rathgamage Don, Kennesaw State University, 3391 Town Point Drive, Suite 2400, Kennesaw, GA 30144, United States. Email: drathgam@students.kennesaw.edu.

REFERENCES

Burkart, N., & Huber, M. F. (2021). A survey on the explainability of supervised machine learning. *Journal of Artificial Intelligence Research*, *70*, 245–317. doi:10.1613/jair.1.12228

Eykholt, K., Evtimov, I., Fernandes, E., Li, B., Rahmati, A., Xiao, C., Prakash, A., Kohno, T., & Song, D. (2018). Robust Physical-World attacks on deep learning visual classification. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 1625–1634. doi:10.1109/CVPR.2018.00175

Freitas, A. A. (2014). Comprehensible classification models. *SIGKDD Explorations*, *15*(1), 1–10. doi:10.1145/2594473.2594475

Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Giannotti, F., & Pedreschi, D. (2018). A survey of methods for explaining black box models. *ACM Computing Surveys*, *51*(5), 1–42. doi:10.1145/3236009

Hall-Beyer, M. (2000). GLCM texture: A tutorial. *National Council on Geographic Information and Analysis Remote Sensing Core Curriculum, 3*(1), 75. https://prism.ucalgary.ca/items/8833a1fc-5efb-4b9b-93a6-ac4ff268091c

Hardt, M., Chen, X., Cheng, X., Donini, M., Gelman, J., Gollaprolu, S., He, J., Larroy, P., Liu, X., McCarthy, N., Rathi, A., Rees, S., Siva, A., Tsai, E., Vasist, K., Yilmaz, P., Zafar, M. B., Das, S., Haas, K., & Kenthapadi, K. et al. (2021). Amazon SageMaker Clarify: Machine Learning bias detection and explainability in the cloud. *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. doi:10.1145/3447548.3467177

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving deep into rectifiers: Surpassing Human-Level performance on ImageNet classification. *2015 IEEE International Conference on Computer Vision (ICCV)*, 1026–1034. doi:10.1109/ICCV.2015.123

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778. doi:10.1109/CVPR.2016.90

Holzinger, A., Langs, G., Denk, H., Zatloukal, K., & Müller, H. (2019). Causability and explainability of artificial intelligence in medicine. *Wiley Interdisciplinary Reviews. Data Mining and Knowledge Discovery*, *9*(4), e1312. Advance online publication. doi:10.1002/widm.1312 PMID:32089788

Kim, B., Wattenberg, M., Gilmer, J., Cai, C., Wexler, J., Viegas, F., & Sayres, R. (2017). *Interpretability beyond feature attribution: Quantitative testing with concept activation vectors*. TCAV., doi:10.48550/arXiv.1711.11279

Korobov, M. (2017). Explaining behavior of machine learning models with eli5 library. *EuroPython*. 10.5446/33771

Lee, E., Braines, D., Stiffler, M., Hudler, A., & Harborne, D. (2019). Developing the sensitivity of LIME for better machine learning explanation. *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications*, *11006*, 349–356. doi:10.1117/12.2520149

Lipton, Z. C. (2018). The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery. *ACM Queue; Tomorrow's Computing Today*, *16*(3), 31–57. doi:10.1145/3236386.3241340

Lundberg, S. M., & Lee, S. I. (2017). A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems* (Vol. 30). https://proceedings. neurips.cc/paper/2017/file/8a20a8621978632d76c43dfd28b67767-Paper.pdf

Menon, S., Damian, A., Hu, S., Ravi, N., & Rudin, C. (2020). *PULSE: Self-Supervised photo upsampling via latent space exploration of generative models*. 10.1109/CVPR42600.2020.00251

Molnar, C., Casalicchio, G., & Bischl, B. (2020). Interpretable machine learning – a brief history, State-of-the-Art and challenges. In *ECML PKDD 2020 Workshops* (pp. 417–431). Springer. doi:10.1007/978-3-030-65965-3_28

Rai, A. (2020). Explainable AI: From black box to glass box. *Journal of the Academy of Marketing Science*, *48*(1), 137–141. doi:10.1007/s11747-019-00710-5

Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). "Why should I trust you?": Explaining the Predictions of Any Classifier. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1135–1144. doi:10.1145/2939672.2939778

Rosenfeld, A., & Richardson, A. (2019). Explainability in human–agent systems. *Autonomous Agents and Multi-Agent Systems*, *33*(6), 673–705. doi:10.1007/s10458-019-09408-y

Shankaranarayana, S. M., & Runje, D. (2019). *ALIME: Autoencoder based approach for local interpretability*. 10.1007/978-3-030-33607-3_49

Shi, S., Zhang, X., & Fan, W. (2020). *A modified perturbed sampling method for local interpretable model-agnostic explanation*. 10.48550/arXiv.2002.07434

Shrikumar, A., Greenside, P., & Kundaje, A. (2017). *Learning important features through propagating activation differences*. https://doi.org/10.48550/arXiv.1704.02685

Singla, S., Eslami, M., Pollack, B., Wallace, S., & Batmanghelich, K. (2023). Explaining the black-box smoothly—A counterfactual approach. *Medical Image Analysis*, *84*, 102721. doi:10.1016/j.media.2022.102721 PMID:36571975

Slack, D., Friedler, S. A., Scheidegger, C., & Roy, C. D. (2019). *Assessing the local interpretability of machine learning models*. https://doi.org/10.48550/arXiv.1902.03501

Su, J., Vargas, D. V., & Sakurai, K. (2019). One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation*, *23*(5), 828–841. doi:10.1109/TEVC.2019.2890858

Sundararajan, M., Taly, A., & Yan, Q. (2017). *Axiomatic attribution for deep networks*. https://doi.org/10.48550/arXiv.1703.01365

Tenney, I., Wexler, J., Bastings, J., Bolukbasi, T., Coenen, A., Gehrmann, S., Jiang, E., Pushkarna, M., Radebaugh, C., Reif, E., & Yuan, A. (2020). *The language interpretability tool: Extensible, interactive visualizations and analysis for NLP models*. /arXiv.2008.0512210.18653/v1/2020.emnlp-demos.15

Vermeire, T., Brughmans, D., Goethals, S., de Oliveira, R. M. B., & Martens, D. (2022). Explainable image classification with evidence counterfactual. *Pattern Analysis & Applications*, *25*(2), 315–335. doi:10.1007/s10044-021-01055-y

Wexler, J., Pushkarna, M., Bolukbasi, T., Wattenberg, M., Viegas, F., & Wilson, J. (2020). The What-If tool: Interactive probing of machine learning models. *IEEE Transactions on Visualization and Computer Graphics*, *26*(1), 56–65. doi:10.1109/TVCG.2019.2934619 PMID:31442996

Zafar, M. R., & Khan, N. M. (2019). *DLIME: A deterministic local interpretable Model-Agnostic explanations approach for Computer-Aided diagnosis systems*. https://doi.org/10.48550/arXiv.1906.10263

Zhao, X., Huang, W., Huang, X., Robu, V., & Flynn, D. (2020). *BayLIME: Bayesian local interpretable Model-Agnostic explanations*. https://doi.org/10.48550 /arXiv.2012.03058

Ramazan Aygun is the Director for Center for Research Computing, Associate Professor of Computer science with joint appointment at School of Data Science and Analytics. He joined Kennesaw State University in 2020.