An Empirical Study on Low- and High-Level Explanations of Deep Learning Misbehaviours

Tahereh Zohdinasab
Università della Svizzera italiana
Lugano, Switzerland
Email: tahereh.zohdinasab@usi.ch

Vincenzo Riccio
Università della Svizzera italiana
Lugano, Switzerland
Email: vincenzo.riccio@usi.ch

Paolo Tonella
Università della Svizzera italiana
Lugano, Switzerland
Email: paolo.tonella@usi.ch

***Abstract*—*Background*: Most quality assessment approaches for Deep Learning (DL) focus on finding misbehaviour-inducing inputs. However, it is difficult to clearly understand the causes of misbehaviours, due to the DL software opaqueness. Recent research proposed different techniques to explain DL misbehaviours, producing input explanations either at a "low level" (raw input elements) or at a "high level" (input features).**
***Aims*: We aim to compare the similarity between different explanations and assess to what extent they are understandable.**
***Method*: We have conducted an empirical study involving 3 state-of-the-art techniques for DL explanation in 13 configurations, applied to 2 different DL tasks. We have also collected answers from 48 questionnaires submitted to SE experts.**
***Results*: Low- and high-level techniques provide dissimilar explanations for the same inputs. However, experts deemed none of the explanations as useful in 28% of the cases.**
***Conclusion*: Despite the complementarity of existing explanations, further research is needed to produce better explanations.**
***Index Terms*—deep learning, software testing, explainable artificial intelligence**

I. Introduction

Deep Learning (DL) algorithms are pervasively adopted in modern software systems, as they proved their ability to solve complex tasks, including safety-critical ones. A large part of the literature focuses on evaluating the quality of DL systems by triggering misbehaviours, i.e., deviations from their expected behaviour [1], [2]. Quality improvement requires understanding the causes of such misbehaviours, which is still an open problem [3], [4]. In fact, the increasing complexity of DL models, trained on large datasets thanks to the availability of scalable and high-performance infrastructures, introduces the risk to make (mis-)predictions whose motivations are hard to understand for humans. Recent DL algorithms, such as Deep Neural Networks (DNNs), involve a massive amount of computations and, thus, are "black boxes" to SE practitioners, since it is hard for them to understand how DL software arrives at a final decision [4], [5].

The lack of explainability of DL models may impact the trust in their predictions and consequently hinder the adoption of such systems. A common issue of DL systems is that they may take wrong decisions based on spurious correlations or biases in the training data, e.g., classifying wolves images as huskies due to the presence of snow on the background rather than leveraging the characteristics of the animal [6].

Multiple techniques address the aforementioned issues and explain DL systems' (mis-)behaviours. Existing explanatory techniques characterise the misbehaviour-inducing inputs either at a low level, by identifying the raw input elements (e.g., image pixels) that are most relevant for the prediction, or at a high level, in terms of abstract features that characterise the input as a whole. We will refer to these techniques as follows:

Low Level Explanations identify a portion of the input as relevant to a specific DL prediction. Such identification is performed directly in the input space of the DL model. Low-level techniques can provide explanations with different aggregations, reporting, e.g., atomic input elements (e.g., pixels/words for an image/text) or contiguous regions of input elements (e.g., sets of contiguous pixels/words). We considered two techniques with different aggregations, representing the state of the art in the eXplainable AI (XAI) community: LIME [6] and Integrated Gradients (IG) [7].

High Level Explanations identify relevant features that characterise the whole input. Such identification is performed in the feature space, a manually-defined abstraction of the input space. As high-level technique, we considered Feature Maps (FM) [8] that characterise inputs through human-interpretable features defined by domain experts. As an example, for the hand-written digit recognition task, feature maps characterise a misbehaviour by indicating how much the hand stroke is bold or discontinous. FM proved to be highly discriminative of misbehaviours [8], so they indeed provide an explanation for misbehaviours.

In this work, we conduct an empirical study to investigate the similarity between explanations provided by low- and high-level techniques for the same sets of inputs. In this way, we can understand whether and to what extent different techniques are overlapping or complementary in explaining DL misbehaviours. Explanatory techniques provide an input characterisation that practitioners (e.g., software testers) can use to understand why an input triggered a misbehaviour [9]. For this reason, we involved human experts in our study

This work was partially supported by the H2020 project PRECRIME, funded under the ERC Advanced Grant 2017 Program (Agreement n. 787703).

978-1-6654-5223-6/23/$31.00 ©2023 IEEE

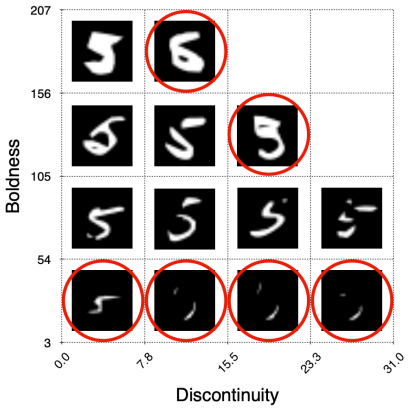


Fig. 1: FM for a handwritten digit classifier. Axes quantify boldness and discontinuity of the digits. Cells report the most challenging input for the corresponding feature values, and the cells highlighted with red circles are mis-classified.

to assess the understandability of the produced explanations, i.e., whether human assessors consider the explanations as effectively pointing to a plausible cause of the misbehaviour.

By leveraging input clustering and a novel ad-hoc similarity metric, our results show that high- and low-level techniques provide different explanations, i.e., they partition the same inputs in different ways. For most of the inputs, human experts chose either a low- or a high-level technique as effective in explaining misbehaviours. These results show that high- and low-level explanations are highly complementary. Results also show that both explanations often do not match human judgement, especially when explaining misbehaviours for image classifiers, hence demanding for novel XAI techniques that can cover the mis-explained cases. In summary, our paper makes the following contributions:

- We introduce an empirical framework for comparing explanatory techniques with different granularity levels;
- We evaluate explanatory techniques on DNNs within two different domains, i.e., image and text classification;
- We perform a human study for comparing the understandability of explanations for DNN misbehaviours;

II. Background

A. Feature Map (FM)

A FM represents the space of the features that are relevant for characterising the test inputs as a multi-dimensional grid, in which each axis corresponds to a considered feature. A feature can either be *structural* (i.e., characterising the input itself) or *behavioural* (i.e., characterising the system's output when exercised by the given input). In particular, we refer to high-level, human-interpretable input features defined by experts [8], such as digit boldness and discontinuity for the handwritten digit classifier shown in Figure 1. Inputs are assigned a map cell, computed by measuring the metric that quantifies each feature. Therefore, each cell contains the inputs characterized by features that fall in the corresponding value

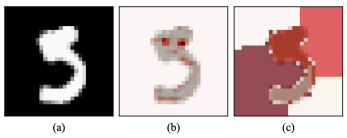


Fig. 2: (a) A sample mis-classified handwritten digit; (b) Explanation by IG; (c) Explanation by LIME. (darker red highlights higher contribution to the DNN output)

interval. The map granularity (i.e., the number of map cells in each dimension) is a configurable parameter of a FM that can be changed depending on the desired level of discrimination.

B. Integrated Gradients (IG)

Associating the prediction of a DNN to the input elements that caused such prediction (i.e., *input attribution*) is a way of explaining the model's behaviour. For instance, in an image classification model, an attribution method could reveal which pixels of the image were responsible for a certain prediction.

The gradient of the output with respect to the input elements (e.g., pixels) quantifies the impact of each input element on the prediction of a DNN, therefore the gradient computation can be considered as the most basic attribution method.

Sundararajan et al. [7] took an axiomatic approach for attributing prediction of DNN to its input features, called IG. They introduced two axioms to be preserved, *Sensitivity* and *Implementation Invariance*, and proposed IG by considering the straight-line path in DNN from the baseline (for example, black image in image classification task) to the input and cumulating the gradients at all points along the path for a given number of steps. The *number of steps* is a hyperparameter of this technique, representing the steps needed by IG in the gradient approximation for each input image. The output of this technique is a heatmap highlighting the important elements in the input (e.g. pixels of an image or words in a text). IG can be implemented to generate explanation for a batch of inputs with predefined size. Figure 2 (b) shows a heatmap generated by IG for an image classification task.

C. Local Interpretable Model-agnostic Explanations (LIME)

LIME provides a low-level explanation for the predictions of any classification model by identifying the portions of the input that mostly affect the model's output. It produces an explanation for each input instance by mimicking the behaviour of the model in the neighbourhood of the input. LIME perturbs the input to train a linear model around the input itself and generates a predefined number of samples, while the model under test is treated as a black box. The number of samples can be customized, as it can be set as a hyperparameter of the technique. Such perturbed input regions are called *super-pixels* and their impact on the linear model's decision determines their importance in the final explanation. LIME is applicable to both images and texts and provides a visual representation of the generated explanations. It has been used in different studies to understand the reasons behind

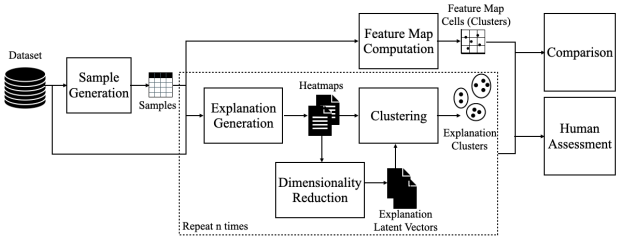


Fig. 3: An overview of our evaluation pipeline

DNN decisions [10]–[12]. Figure 2 (c) shows an example of explanation generated by LIME for a hand written digit 5.

III. Comparing High- and Low-level explanations

To compare high-level and low-level explanations, we designed the evaluation pipeline shown in Figure 3. Initially, we gather the inputs to be explained. Such inputs can be either obtained directly from an existing dataset or can be automatically generated by test generators (*Sample Generation* step). For each input, we obtain high-level explanations in the *Feature Map Computation* step, i.e., we compute the FM cells it belongs to. In this way, each FM cell represents a cluster of inputs sharing similar high-level features. To obtain low-level explanations, we perform the *Explanation Generation* step, in which we apply low-level techniques that generate heatmaps. Moreover, we perform a *Dimensionality Reduction* step on low-level explanations to generate lower-dimensional latent vectors. A *latent vector* is the projection of an input onto the latent space. Using latent vectors allows us to avoid high variability and sparseness in the high-dimensional heatmap explanations. Indeed, a sparse, highly variable representation of the inputs would prevent the construction of meaningful clusters of the inputs, as in the higher-dimensional space all distances/similarities tend to degenerate to the extreme values, giving raise to degenerate clusters. Since both low-level explanatory techniques and dimensionality reduction approaches are non-deterministic, we repeat the low-level explanation generation multiple times. To make the explanations provided by low-level techniques comparable to the high-level ones, we cluster low-level explanations by using a *Clustering* technique. In the *Clustering* step, we group together similar heatmap explanations, or the corresponding latent vectors (to address the sparseness problem). In the *Comparison* step, we assess how similar the clusters generated by high-level and low-level techiques are, based on our custom definition of a *Gini similarity* metric, which is close to zero when the elements grouped together by one technique are scattered across many clusters produced by the other technique and is equal to one when the elements grouped together by one technique are all in the same cluster produced by the other technique. Finally, we submit a questionnaire to experts in order to evaluate the understandability of explanations in the *Human Assessment* step. In the following, we provide a detailed explanation of each step of our evaluation pipeline.

Sample Generation: Since we are interested in investigating DL misbehaviours, we resort to failure-inducing inputs artificially crafted by test generators, besides the ones from the original dataset. In this way, we can collect enough misbehaviour-inducing inputs even for robust DL models, i.e., for which there are only a few misbehaviours in the original test set. By considering both types of inputs, we perform our study on a sufficient number of misbehaviours, covering a diversified variety of samples. We do not use the train set because we mimic the testing phase, when developers evaluate a DL model against new, unseen inputs

FM Computation: To provide high-level explanations, we used FM. Feature Maps can be generated with different feature combinations and numbers of dimensions. For the sake of completeness, we considered all possible numbers of dimensions and feature combinations when computing the FM. In this way, we can discuss the similarity between high- and low-level explanations at increasing FM dimensionality (e.g., when moving from one isolated feature to a combination of multiple features). For each input obtained in the Sample Generation step, we compute its corresponding feature values, so we can assign each input to the corresponding FM cell, i.e., the map cell whose value intervals contain the measured input feature values. We use the non-empty FM cells, i.e., those containing at least one misbehaviour-inducing input, as high-level explanation clusters.

Explanation Generation: In this step, we consider two popular XAI techniques as our low-level explanatory techniques: IG for fine-grained explanations and *LIME* for coarse-grained explanations. We apply the two considered XAI techniques separately on the generated inputs and extract heatmaps as vectors representing the relevance of input elements/regions.

Dimensionality Reduction: Beside the heatmap explanations in the original input space, we consider explanations projected onto a latent space, i.e., latent vectors. In fact, by projecting heatmaps to latent space, we keep a lower number of dimensions, which are more representative of the meaningful directions of variability of the inputs, possibly avoiding the construction of degenerate clusters. For generating latent vectors, we choose the *t-SNE* algorithm [13], [14], since it projects similar inputs to neighbouring points and dissimilar inputs to distant points with high probability.

We generate latent vectors in two modes: (1) *global latent*: the projections of explanations in the latent space are computed considering inputs from all output classes; (2) *local latent*: the projections of explanations in the latent space are computed considering only inputs from a specific class. While the local latent space can achieve good separation of inputs from a specific class, the global latent space aims at separating both different classes of inputs and inputs within each class. The former is better if the generated inputs stay confined within one class, while the latter might be beneficial when the generated inputs tend to cross the borders between classes.

Clustering: The explanations generated by high-level techniques are FM cells based on high-level feature values of the inputs, whereas low-level explanations are vectors of

contributions of low-level input elements/regions. Therefore, there is no way to directly measure the similarity between explanations at these two different abstraction levels. To make the comparison feasible, we propose to compare the *clusters of explanations* instead of the raw explanations (which are uncomparable). The underlying idea is the following: if two explanatory techniques group the inputs in a similar way, then they can be deemed similar; otherwise they are different.

For high-level explanations we use map cells as clusters. For low-level explanations, we need to group the explanations by using clustering approaches in such a way that objects in the same group, i.e., a cluster, are more similar to each other than to those in other groups. There exist multiple clustering techniques in the literature [15]–[18]. For our study, we rely on the *Affinity Propagation* clustering technique [18] which recursively exchanges messages between data points; such messages encode the affinity of one point when choosing another point as its neighbor. The recursive exchange of messages continues until a set of highly-affine groups emerges. The main advantage of this clustering technique is that, unlike other techniques such as K-Means [15], (1) all the points are considered as possible centroids of the clusters, which avoids biasing the clusters to some randomly chosen points, used as the initial centroids; and (2) there is no need to provide the number of clusters to the algorithm in advance. In this step, we cluster the explanations in the three considered input spaces: *original*, consisting of raw explanation vectors; *global latent*, consisting of globally projected explanations; and *local latent*, consisting of locally projected explanations. Of course, each considered space may lead to different clusters, based on the distribution of the inputs in the corresponding space. The output of this step consists of the clusters generated for the fine-grained and the coarse-grained explanations, in each of the considered input spaces.

Comparison: In the previous steps, we generated high-level and low-level explanations for the considered inputs and processed them to generate clusters. The main evaluation step of our study is the comparison of these explanation clusters. Existing similarity or distance metrics to compare two sets of clusters, such as the MoJo metric [19], are based on the transformation of one cluster into the other. The computational complexity involved in the computation of these metrics is typically high (exponential in the worst case), but what is even more concerning in our usage scenario is that such metrics are highly sensitive to the number of clusters, and when there is a disparity in such number the distance tends to grow (similarity tends to decrease), because more transformation steps are needed to change one clustering into the other. However, a high disparity in the number of clusters is not necessarily an indicator of distance in our case. In fact, if all clusters in one set are *pure*, because their elements come from the same clusters in the other set, we deem the two clusterings very similar between each other, regardless of any disparity in the number of clusters. To capture such a notion of similarity between two sets of clusters, we define a novel metric based on *Gini Impurity (GI)* [20]. GI reflects the impurity level of a group of entities by indicating the probability that two samples from the given group have different labels, i.e., belong to different classes. So, to compute GI we need to define what are the *groups* and what are the *labels*. In our setting, when comparing two sets of clusters (low- vs high-level explanations), groups are the clusters identified in one set (source clustering), while labels are the cluster identifiers from the other set (target clustering). Hence, the impurity of a group of data D (i.e., a cluster from the source clustering) against the target clustering A can be measured as follows:

$$GI(D,\ A) = 1 - \sum_{i=1}^{|A|} p_{Ai}^2 \tag{1}$$

where $|A|$ is equal to the number of clusters in A, and p_{Ai} is the probability that cluster id i of clustering A occurs in dataset D. GI ranges between 0 and 1, being 0 when D contains no impurity (i.e., all its elements belong to the same cluster from A) and being minimum when D is uniformly impure (i.e., all its elements belong to a different cluster from A).

Let us consider cluster $CB1$ in Figure 4 (d). GI of $CB1$ against clustering A (Figure 4 (a)) is computed as follows: $GI(CB1, A) = 1 - \sum_{i=1}^{3} p_{Ai}^2 = 1 - (1 + 0 + 0) = 0$. ($p_{A1} = 1$ is the probability of cluster $CA1$ to be found in cluster $CB1$; $p_{A2} = p_{A3} = 0$ is the probability to find $CA2$, $CA3$).

We can now aggregate the computation of GI across all clusters that belong to the source clustering B, using the clusters in A as labels, by taking the average cluster impurity. The complement of such average impurity gives a *Gini Similarity* (GS) metric between clusterings, ranging between 0 and 1, where 1 is achieved when all clusters in B are pure (i.e., have GI = 0):

$$GS_{(B,A)} = 1 - \frac{1}{|B|} \sum_{i=1}^{|B|} GI(CB_i, A) \tag{2}$$

where $|B|$ is equal to the number of clusters in clustering B, and C_i is the i^{th} cluster in clustering B.

Figure 4 (a) and (b) present two examples of clusterings A and B. To compute the similarity between A and B, we consider the clustering B as source and then we color (and label) it elements based on the clusters in clustering A (see Figure 4 (d)). $GS_{(B,A)} = 1 - \frac{1}{4}(GI(CB1, A) + GI(CB2, A) + GI(CB3, A) + GI(CB4, A)) = 1 - \frac{1}{4}(0 + \frac{1}{2} + \frac{4}{9} + 0) = 0.76$.

GS is not symmetric by definition, as the labels assigned to the elements of one clustering depend on the other, and such labeling changes when we swap the two clusterings in the computation. For the example in Figure 4, let us now compute the similarity between A and B. We now consider the clustering A as source and we color (label) its elements based on the clusters they belong to in clustering B (see Figure 4 (c)). $GS_{(A,B)} = 1 - \frac{1}{3}(GI(CA1, B) + GI(CA2, B) + GI(CA3, B)) = 1 - \frac{1}{3}(\frac{4}{9} + \frac{1}{2} + \frac{4}{9}) = 0.53$.

To deal with the asymmetric nature of our metric, we report the maximum of the two *GS* values obtained when considering source and target clusterings in both orders:

$$Similarity(A, B) = Max(GS_{(A,B)}, GS_{(B,A)}) \tag{3}$$

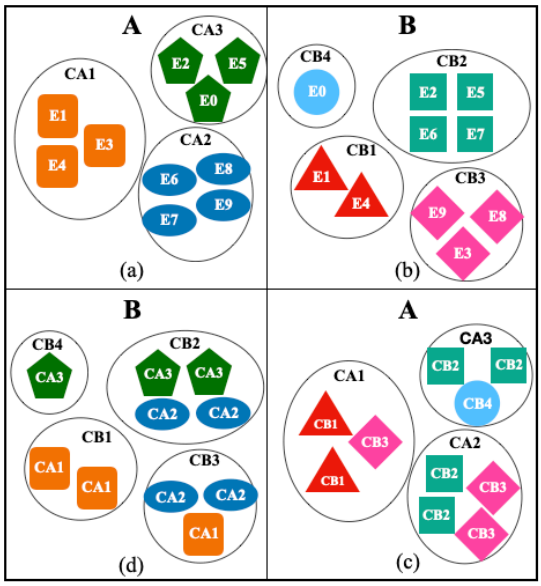


Fig. 4: Gini Impurity (GI) computation with source B and target A (col. (d), (a)), and with source A and target B (col. (c), (b)); colors and text in (d), (c) are used to indicate the labels obtained from the target clusters (resp. (a), (b))

Considering clusters A and B in Figure 4, $Similarity(A, B) = Max(0.53, 0.76) = 0.76$

Human Assessment: In this step, we conduct a human study to investigate the understandability of explanations. We design a survey and provide human assessors with low- and high-level explanations of misbehaviour-inducing inputs. Then, we ask the assessors whether the shown explanations effectively indicate the cause of the misbehaviour.

IV. Experimental Evaluation

A. Research Questions

RQ1 (Similarity): *How similar are high-level and low-level explanations of DL misbehaviours?*

High- and low-level approaches provide explanations about failure-inducing inputs from different perspectives. We aim to measure the similarity of these different explanations to assess to what extent they are comparable or complementary.

Metrics: To assess how similar the explanations generated by high-level and low-level techniques are, we compare the way inputs are grouped by similar explanations. To this aim, we measure (1) the number of clusters generated by each technique and (2) the similarity between these clusters according to our custom Gini Similarity (GS) metric. If two techniques produce explanations that group the inputs in a similar way (i.e., they produce nearly the same number of clusters that are pairwise very similar to each other), then they can be deemed similar; otherwise they differ, as they partition the input vectors in a different way.

RQ2 (Understandability): *How understandable are high-level and low-level explanations of DL misbehaviours?*

TABLE I: Hyperparameters and configuration details

	MNIST	IMDB
Number of cells per feature	5	5
Number of runs	10	10
t-SNE components	2	2
t-SNE perplexity	1	1
Similarity metric	Euclidean	Euclidean
Input size	28×28	2000
Vocabulary size	-	10000
Number of inputs	250	250
Target class label	"5"	"positive"
Number of steps for IG	50	50
Batch size for IG	64	100
Number of samples for LIME	100	5000
Explanation library	*Xplique*	*Alibi*, *Lime*

Since high-level techniques quantify high-level input features, while low-level approaches highlight low-level elements of the inputs, it is important to investigate whether these two types of explanations are equally understandable to humans.

Metrics: Effectively assessing the understandability of an explanation requires humans in the loop. Therefore, we designed a human study to assess if explanations are understandable and if they match the human expectations. We count the number of cases in which the explanation Matches with Human (MH), i.e., the number of times human assessors select a given explanation as possibly pointing to the cause of the misbehaviour.

B. Subject Systems

In our study, we consider two DL systems belonging to different domains, i.e., handwritten digit recognition and sentiment analysis, which have been widely used in the literature to generate explanations for DL systems [6], [7].

The MNIST system is a classifier which recognises handwritten digits within image inputs from the MNIST dataset [21]. This DL system accepts greyscale images as inputs and predicts the corresponding digit classes (from 0 to 9). We considered the convolutional DNN architecture provided by Keras [22] and trained it on the the MNIST training set using Tensorflow v2. More specifically, we used its default configuration, i.e. 12 epochs, batches of size 128, a learning rate equal to 1, and the *Adam* optimizer, which achieved 99.8% test accuracy.

The IMDB system classifies textual inputs based on their sentiments. It accepts the text of a movie review as input and predicts if it is positive or negative. We adopted a convolutional DNN with an embbeding layer provided by Keras [23], and we used tokenised (with vocabulary size equals to 10000) and padded text with length limited to 2000 words. We trained the model on the IMDB training set in Tensorflow v2. We used 10 epochs, batches of size 32 with early stopping, and the *Adam* optimizer, achieving 88.0% test accuracy.

C. Experimental Procedure

To obtain the explanations, we adopted the pipeline described in section III (configuration reported in Table I).

As misbehaviour-inducing inputs, for MNIST, we used a set of 250 inputs belonging to the same ground-truth class (i.e., digit "5"), either from the test set or generated by test input generators. We used the DeepJanus [24] and Sinvad [25] test generators since they (1) belong to different families of approaches (i.e., model-based and DL generative, respectively), and (2) have been demonstrated to produce the highest ratio of valid input that preserve their ground-truth label [26]. In particular, we could obtain only 11 misclassified inputs belonging to the class digit "5" from the MNIST test set, due to high accuracy of the considered model. Therefore, we also included 35 inputs generated by DeepJanus and 204 inputs generated by Sinvad, to have enough and diverse misbehaviour-inducing inputs for our study.

For IMDB, we also used a set of 250 inputs either from the test set or generated by test input generators, all belonging to the same class ("positive" sentiment). We obtained 1924 misclassified inputs belonging to the class "positive" from the test set. Then, to have more diverse inputs we also included 663 inputs generated by the considered input generator. In particular, we used the DeepHyperion [8] tool since it supports the generation of textual inputs. Finally, we randomly selected 250 inputs from the overall set of 2587 candidate inputs.

To obtain low-level explanations, we used three open source libraries for images and textual inputs, i.e., *Xqlique* [27] for image, and *Alibi* [28] and *Lime* [29] for text.

To obtain high-level explanations, we needed high-level features for each considered domain. For MNIST inputs, we used the three high-level features defined by Zohdinasab et al. [8] in their study involving human experts, i.e., (1) *Luminosity (Lum)*: number of light pixels in the image, obtained by counting the pixels whose value is above 127; (2) *Orientation (Or)*: vertical orientation of the digit, obtained by computing the angular coefficient of the linear regression of the non-black pixels; (3) *Moves* (Mov): sum of the Euclidean distances between pairs of consecutive segments of the digit, the distance being zero when consecutive segments are connected and greater than zero when there are discontinuities. For IMDB reviews, we defined three features: (1) *Positive word count (Pos)*: number of words in the text with positive polarity, obtained by counting the words tagged as positive in the English Opinion Lexicon [30]; (2) *Negative word count (Neg)*: number of words in the text with negative polarity, obtained by counting the words tagged as negative in the English Opinion Lexicon; (3) *Verb count (Verb)*: number of verbs in the text, a proxy for the text complexity, computed by counting the words with a *verb* tag, according to the part-of-speech (POS) tagging produced by the NLTK library.

Our experimental procedure consists of the following two steps: (1) high- and low-level explanations comparison, using clustering; and (2) human assessment of the explanations, by means of two surveys: one for MNIST and one for IMDB.

1) High- and Low-Level Explanations Comparison: We generate FEATURE MAPS with 1, 2, and 3 dimensions re-scaled to size 5 per dimension. We selected 5 as number of cells since the number of clusters (i.e., filled cells) with this configuration is comparable to the number of clusters obtained through low-level techniques, which allowed us to interpret each cell as a cluster without any need for an additional clustering step. We ran FM generation only once since the clusters (i.e., the cells) are deterministic.

As regards low-level explanations, we ran IG and LIME approaches 10 times each, since the corresponding clusterings are obtained trough t-SNE, which is non-deterministic. To obtain the best configuration for the *t-SNE* hyperparameters (i.e., number of components and perplexity), we performed preliminary runs with different configurations and selected the one producing the highest *silhouette score*. The best configurations of the hyperparameters of IG and LIME are reported in Table I.

For clustering the low-level explanations, we used the *Euclidean distance* to compute the similarity matrices, where distances are computed in three different vector spaces: *original*, *global latent*, and *local latent* (see Figure III). In the original space, we consider heatmaps as vectors to be clustered. For MNIST, a heatmap is a 28×28 matrix, where each vector component e_{ij} is the contribution value of the pixel occupying the i-th row and the j-th column in the image. The matrix is flattened into a vector before applying clustering. For IMDB, a heatmap is a vector of size 10000, the size of the vocabulary used by the tokenizer (which maps each word to the corresponding one-hot encoding) and each vector component e_i is the contribution value of the i-th word in the text. In both global latent and local latent spaces, we use vectors of size 2, the same number of dimensions that was found to be optimal for the t-SNE algorithm.

2) Human Assessment of the Explanations: To determine whether an explanation is human-understandable, we asked humans to determine which of the explanations provided them meaningful information about the reason why the model misbehaves for specific inputs. We published two surveys (one for each case study) by using Qualtrics, a survey platform commonly used also for software engineering research [31], [32]. To ensure the assessment quality, we selected software engineering researchers and allowed each of them to devote up to 1 week to answer the questions and take at most 1 questionnaire for each case study.

For MNIST, we created a survey with 10 questions to be answered by human assessors. To this aim, we randomly selected 10 "5" digit images from the MNIST data set which are misclassified by the considered model and we computed high-level and low-level explanations for each of them. More specifically, for the high-level explanations we reported the feature values provided by the three-dimensional FM and for the low-level explanations we visualized IG and LIME heatmaps, overlayed on the original digit images. We used 3D FEATURE MAPS for the human study to include all the available features for the high-level explanations. For each MNIST image, we showed the human assessors the possible explanations (i.e. FM, IG, and LIME) and asked them to answer the following question: *"Below is a digit "5" that was incorrectly believed to be another digit by the computer.*

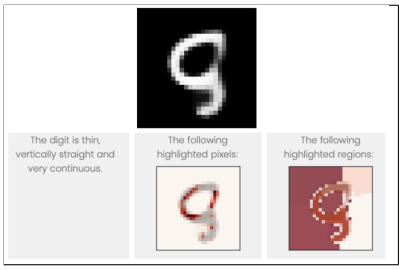


Fig. 5: MNIST input from the human study for which the majority selected IG.

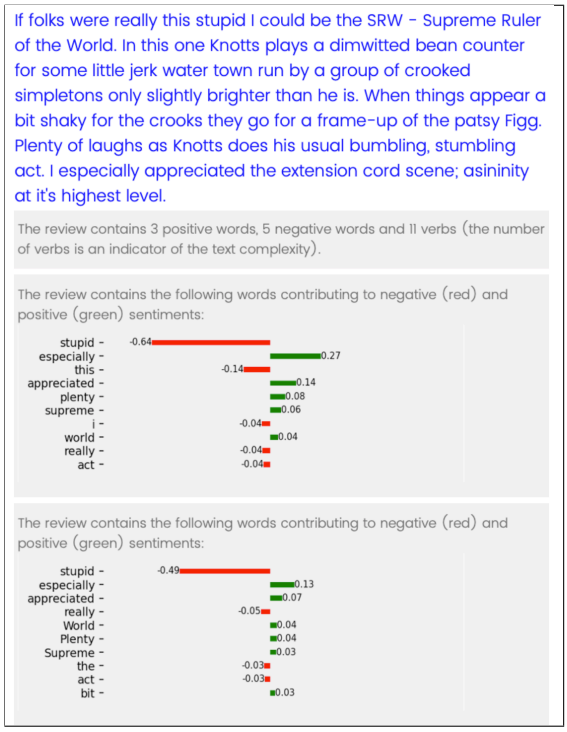


Fig. 6: IMDB input from the human study for which the majority selected both FM and IG explanations.

Please, select a possible explanation for such a mistake (you are allowed to select more than one explanation if they make sense, or none of them if they do not make sense)". Figure 5 shows a misclassified digit "5" with the three different explanations.The assessors could select the explanations that they considered as a plausible reason for the misbehaviour. In particular, the assessors were allowed to choose zero or more explanations. We collected 29 answers from the human assessors in the MNIST survey.

For IMDB, we created a survey with 10 questions by randomly selecting 10 positive reviews from the IMDB dataset, which are misclassified by the model. For the high-level explanations, we reported the feature values provided by the three-dimensional FM and the explanations generated by LIME and IG and we visualised the words with highest contributions to negative and positive sentiments in a bar chart (i.e., we reported up to 10 most contributing words with non-zero contribution). For each text, we asked humans to answer the following question: *"Below is a positive review that was incorrectly believed to be a negative review by the computer. Please, select a possible explanation for such a mistake (you are allowed to select more than one explanations if they make sense, or none of them if they do not make sense)"*. Also in this case, we showed explanations provided by the FM, IG, and LIME (as shown in Figure 6). In the IMDB survey we collected 19 answers from the human assessors.

TABLE II: RQ1 - Number of clusters (NC) for high-level techniques for MNIST and IMDB.

HL Technique	MNIST Features	MNIST NC	IMDB Features	IMDB NC
Feature map 3D	Mov-Lum-Or	31	Pos-Neg-Verb	27
Feature map 2D	Mov-Lum	17	Pos-Neg	15
	Lum-Or	15	Neg-Verb	14
	Or-Mov	14	Pos-Verb	12
Feature map 1D	Mov	5	Pos	5
	Lum	5	Neg	5
	Or	5	Verb	5

TABLE III: RQ1 - Number of clusters (NC) for low-level techniques for MNIST and IMDB.

LL Technique	Input space	MNIST NC [min, max]	IMDB NC [min, max]
IG	Original	44.0 [44, 44]	35.1 [35, 36]
	Global Latent	18.7 [17, 20]	14.8 [12, 18]
	Local Latent	20.5 [19, 21]	18.2 [16, 22]
LIME	Original	39 [37, 44]	38.5 [37, 39]
	Global Latent	17.7 [16, 19]	17.3 [16, 19]
	Local Latent	19.0 [17, 21]	18.3 [16, 21]

V. Results

A. RQ1 (Similarity)

Tables II and III report the number of clusters generated by each explanatory technique. For each low-level approach, we report the minimum and maximum number of clusters (NC) in order to show their variability due to non-determinism.

For MNIST, the number of clusters generated by FM 3D is 31, larger than the number of clusters generated by FM with any other dimensionality (i.e., 1D and 2D). Not surprisingly, the lower the dimensionality, the lower the number of clusters. In fact, all the FM dimensions were rescaled to size 5, leading to a lower number of cells for lower dimensionality maps (e.g., in 3D maps there are at most 125 cells to fill, while for 1D maps there are only 5 cells). The highest difference between high- and low-level techniques is between FM 1D and IG in the Original input space (39 clusters). Instead, the NC value for both FM 2D and 3D is comparable to the number of clusters generated by low-level techniques (resp. global/local latent space and original space).

As regards IMDB, the highest difference between high- and low-level techniques is the one between FM 1D and LIME in the Original input space, i.e., 33.5. Also for IMDB, the closest

TABLE IV: RQ1 - Comparing high-level and low-level explanations' Similarity (Sim); boldface indicate statistical significance when comparing original with latent space similarities.

HL technique	LL technique	Input space	MNIST Features	MNIST Sim	IMDB Features	IMDB Sim
Feature map 3D	IG	Original	Mov-Lum-Or	**0.70**	Pos-Neg-Verb	**0.74**
		Global Latent		0.55		0.68
		Local Latent		0.55		0.66
	LIME	Original		0.55		**0.81**
		Global Latent		0.53		0.66
		Local Latent		0.53		0.68
Feature map 2D	IG	Original	Mov-Lum	**0.71**	Pos-Neg	**0.74**
		Global Latent		0.40		0.52
		Local Latent		0.41		0.53
	LIME	Original		**0.56**		**0.77**
		Global Latent		0.39		0.55
		Local Latent		0.40		0.56
Feature map 2D	IG	Original	Lum-or	**0.76**	Neg-Verb	**0.78**
		Global Latent		0.52		0.60
		Local Latent		0.52		0.60
	LIME	Original		**0.65**		**0.82**
		Global Latent		0.49		0.61
		Local Latent		0.49		0.61
Feature map 2D	IG	Original	Mov-Or	**0.80**	Pos-Verb	**0.77**
		Global Latent		0.54		0.57
		Local Latent		0.54		0.59
	LIME	Original		**0.64**		**0.81**
		Global Latent		0.52		0.61
		Local Latent		0.51		0.63
Feature map 1D	IG	Original	Mov	**0.83**	Pos	**0.80**
		Global Latent		0.49		0.64
		Local Latent		0.52		0.65
	LIME	Original		**0.68**		**0.83**
		Global Latent		0.50		0.66
		Local Latent		0.54		0.69
Feature map 1D	IG	Original	Lum	**0.78**	Neg	**0.82**
		Global Latent		0.47		0.63
		Local Latent		0.53		0.65
	LIME	Original		**0.69**		**0.84**
		Global Latent		0.45		0.67
		Local Latent		0.50		0.67
Feature map 1D	IG	Original	Or	**0.94**	Verb	**0.87**
		Global Latent		0.82		0.71
		Local Latent		0.83		0.73
	LIME	Original		**0.89**		**0.89**
		Global Latent		0.82		0.72
		Local Latent		0.83		0.74

NC values between the two classes of techniques is produced by FM 2D and low-level techniques in the global/local latent space. The number of clusters for FM 3D is comparable to low-level techniques in the original space.

For both case studies, IG produces a more stable number of clusters in the original space due to its deterministic nature (i.e. it is based on a mathematical equation which computes linear interpolation from a baseline), whereas there are some variations for global and local latent spaces due to the non-deterministic nature of the *t-SNE* algorithm. Instead, LIME always generates slightly different explanations across multiple runs, because of the randomness introduced by input sampling and surrogate model training during the explanation computation, and thus produces a variable number of clusters.

Table IV reports the similarity between clusterings. To assess the statistical significance of the comparisons between different configurations, we performed the Mann-Whitney U-test and measured the effect size by means of the Vargha-Delaney's A12 statistic [33]. For both MNIST and IMDB (fifth and seventh columns), the similarity values between high- and low-level techniques are almost always significantly higher (p-value < 0.05, large effect size) when considering the original input space, rather than global and local latent spaces within the same configuration (i.e., up to 0.34 more in the comparison between FM 1D and IG for MNIST). This result may be due to the fact that the heatmaps in the original space retain more feature-related information than their projections into the latent space. For MNIST, the highest similarity is achieved between FM 1D (Orientation) and IG in the original space (0.94). For IMDB, the highest clustering similarity is between FM that considers the Verb feature only and LIME explanations in the original space (0.89). The most different partitions between low- and high-level techniques (i.e., lowest similarity) are obtained when considering FM 2D and low-level explanations in the latent space (either local or global). In fact, for MNIST we have the lowest similarity when considering Mov-Lum maps and LIME in the global latent space (0.39), whereas for IMDB the lowest similarity is between Pos-Neg maps and IG in the global latent space.

On average, the similarity between FM and IG, across input spaces and subjects, is 0.65, while it is 0.64 with LIME. These are relatively low values. As a reference, such similarity values are obtained when a cluster with 100 elements contains 23 impure elements, i.e., 23 elements that are assigned a different cluster by the other technique (see Equation 2).

> **RQ1:** High-level and low-level techniques partition inputs in different ways. Despite FM 2D and low-level techniques in the global latent space produce nearly the same number of clusters, those show low similarity. Likewise, FM 1D produces clusterings similar to the ones obtained by low-level techniques in the original space, but with very different number of clusters.

B. RQ2 (Understandability)

Table V shows the results extracted from questionnaires for MNIST and IMDB. Each column reports the number of assessors who chose the explanation provided by each considered technique (choices were not mutually exclusive). The last column highlights the cases for which no explanation was selected by the assessor. Since each assessor can select 0 to 3 explanations in each question, the sum of the values in each row does not correspond to the number of assessors.

Table V (left) reports the answers we collected for MNIST. IG was selected more than other explanations 4 times, e.g., for Q2 IG has been chosen 18 times more than the others. The explanations by FM are selected more than the other explanations 4 times. LIME was selected more than the others

TABLE V: RQ2 - Number of Matches with Human Explanations (MH); 'None' indicates the number of cases when no match was found.

	MH							
	MNIST				IMDB			
Q#	FM 3D	IG	LIME	None	FM 3D	IG	LIME	None
Q1	12	2	10	8	2	13	3	3
Q2	5	23	5	1	5	3	3	9
Q3	4	7	9	12	11	17	8	0
Q4	6	7	5	14	6	15	3	2
Q5	3	15	2	11	10	14	7	1
Q6	7	6	4	14	12	15	8	0
Q7	7	11	7	10	0	14	5	3
Q8	9	5	8	13	11	14	12	1
Q9	5	1	9	17	6	8	4	4
Q10	13	10	5	8	11	12	7	2
Sum	71	87	64	108	74	125	60	25

in only 2 questions. We can conclude that there is no technique clearly more understandable than the others for explaining digit misclassications, according to human assessors. Overall, the explanations generated by IG match more frequently with the human expectations, as they have been selected 30% of the times by the assessors. FM explanations and LIME heatmaps have been selected 24% and 22% of the times. However, in more than one third of the answers the assessors chose none of the explanations. This result suggests that there is large room for improvement in explaining image misclassifications. Table V (right) reports the answers for IMDB. IG was selected more than the other techniques 9 times, while FM was selected more than the others in the remaining case (i.e., Q2). Instead, LIME explanations were never selected more often than the other techniques. IG was selected the highest number of times by human assessors, i.e., 66% , followed by FM with 39% and LIME with 31%. Explanations were more understandable for textual inputs, than for handwritten digit images. In fact, only 13% of the times the assessors did not choose any explanation for a question on IMDB. For MNIST, only for 2 questions we have a majority, i.e., more than half of the assessors selected the same answer. Instead, for IMDB in 8 questions we have answers selected by the majority of assessors. This indicates that the explanations provided for IMDB find more consensus among humans than the explanations provided for MNIST.

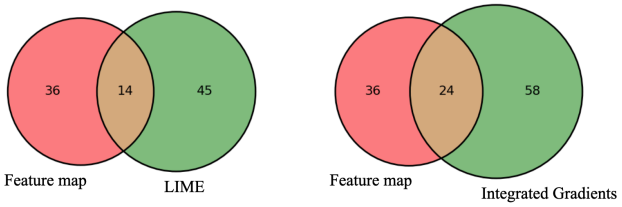
RQ2: Both high- and low-level techniques produce human-understandable explanations of misbehaviours of text classifiers. In particular, IG explanations were selected 125 times out of 190 answers for IMDB. On the other hand, the explanations for misbehaviours of image classifiers poorly matched with the human judgement. In fact, in more than one third of the answers the assessors chose none of the proposed explanations for MNIST. On the remaining two thirds, high- and low-level explanations are chosen approximately the same number of times.

C. Discussion

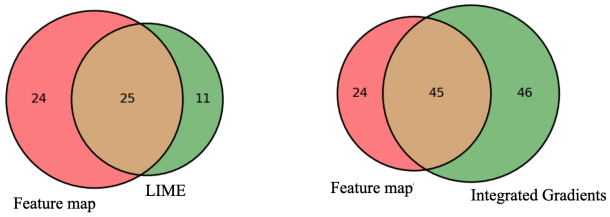
The answers provided by the human assessors offer several insights that we discuss qualitatively in the following.

- For MNIST, the two most understandable explanations are IG and FM. Not only are these explanations at different levels (i.e., low- and high-level, respectively), but they show also some degree of complementarity. In fact, for Q1, Q8 and Q9, FM were selected more times than IG with a MH difference higher than 4. On the other hand, for Q2, Q5 and Q7, IG was selected more times than FM. Remarkably, for question Q2 (reported in Figure 5), IG was selected 18 times more than FM. In this case, the assessors did not find the FM explanation useful, while IG highlights the pixels that make the upper part of the five look like a nine, i.e., the leftmost pixels, making the upper part round, and the rightmost pixels, close the circle.
- For IMDB, FM and IG explanations are the most chosen by human assessors. In particular, for half of the questions, IG and FM were chosen by at least 10 assessors and the difference in MH was lower than or equal to 4. For instance, Figure 6 reports Q8, where the assessors selected FM 11 times and IG 14 times. In this case, the assessors acknowledged that the review actually contained more negative than positive words as reported by the FM explanation. At the same time, the assessors probably agreed with the IG heatmap that the word "stupid" contained in the text can negatively affect the prediction.
- For some questions, all the explanations were judged as poorly understandable by humans, who did not choose any of the provided explanations as possible causes of the misbehaviour. For MNIST, Q9 can be considered the most challenging question, since only 15 times out of 87 times an explanation was selected, while the majority of the assessors selected none of the explanations. Similarly, in the IMDB survey, Q2 is the question with the lowest number of selected explanations, since the assessors chose an explanation only 11 times out of 57, while the majority chose none.
- As shown in Figure 7 (a), for MNIST, assessors selected only FM 36 times and only LIME 45 times while they selected both explanations 14 times. Similarly, they selected only FM 36 times, only IG 58 times and both FM and IG 24 times. This suggests that high-level and low-level explanations for MNIST digit misclassifications are highly complementary. Figure 7 (b) shows that for IMDB, most of the times both high- and low-level explanations have been selected together. However, the number of times assessors selected one of the two and not the other remains quite high, which indicates substantial complementarity.

In summary, in several instances either high- or low-level explanations look understandable for humans and provide hints on what are the properties of the inputs that make the DNN misbehave. In most cases, either high- or low-level explanations, but not both, were chosen as useful explanations, indicating that such explanations are highly complementary.



(a) MNIST



(b) IMDB

Fig. 7: Comparison of the number of matches with human between high-level and low-level techniques.

On the other hand, there is still a large fraction of cases in which none of the two are deemed useful by humans, which points to the need for better DNN explanations.

D. Threats to Validity

Construct Validity: FM depends on the features selected to generate the map dimensions. For feature selection, we relied on previous work [8], while for the combination of features, we exhaustively considered all available dimensions and all their combinations. Moreover, the selection of different hyper-parameters for our study can affect the final results. To reduce this risk, we developed our framework in order to be configurable and we performed some preliminary runs with different hyper-parameters to select the best configuration.

External Validity: The selection of subjects could be a threat to external validity. Therefore, we considered two different domains, i.e., image and text. To represent different types of explanations, we considered one state-of-the-art, openly available technique for each explanation type.

Conclusion Validity: The stochastic nature of some of the considered low-level techniques, the dimensionality reduction approach and the clustering algorithm might affect the final outcome. To mitigate this threat, we ran these techniques multiple times and reported average value across runs.

Reproducibility: To make our results reproducible, We make our implementation and the experimental data openly accessible online: https://github.com/testingautomated-usi/unboxer.

VI. Related Work

A. Comparison of Explainable AI techniques

During the last decade, a variety of techniques have been proposed to provide explanations for DL [6], [7], [34]–[38]. Although these techniques share a common goal, they differ in various aspects. Therefore, there is a growing need for comparisons of such explanatory techniques.

Tjoa and Guan [39] proposed a survey on explanatory techniques and categorised them based on their interpretability. Their analysis is based solely on the mathematical properties of each technique and on experimental results available from the literature, while we explicitly ran the explanatory techniques and compared them both quantitatively and qualitatively. Samek et al. [40] provided a survey on explanatory techniques applicable to DNNs. Unlike our work, they did not evaluate the similarity among explanations produced by different techniques and they analysed human interpretability of explanations without conducting any human assessment involving humans. In fact, they measured the size of the explanation file generated by each technique as a measure of human interpretability. Ledel and Berthold [10] examined the quality of explanations provided by LIME and SHAP for a different task, i.e., bug prediction, through an empirical study involving human assessors. Unlike them, we compared different techniques also quantitatively.

To the best of our knowledge, existing empirical comparisons only considered low-level explanations, without evaluating high-level explanations.

B. Explainable AI for DL Testing

In the literature, both low- and high-level explanations have been leveraged to interpret the output of DL testing. As for low-level explanations, HUDD [41] identifies root causes of DNN failures by using heatmap explanations generated by the LRP technique and then clustering inputs with similar heatmaps. Stocco el. al [42] proposed ThirdEye, a monitor for autonomous driving systems which relies on heatmaps produced by the SmoothGrad [37] technique to predict unsafe conditions in advance. High-level explanations provided by FM have been used for different testing tasks. DeepHyperion [8], [43], [44] is a test generator that leverages FM for extensively exploring the feature space and finding failure-inducing inputs with different characteristics. Nguyen et al. [45] used FM for test selection. FM has also been used to assess test suite adequacy, measured as the number of cells covered by the test inputs [46].

Existing works either considered low- or high-level explanations. Instead, we consider both explanation levels and compare them to assess their usefulness and understandability in explaining DL misbehaviours.

VII. Conclusions and Future Work

In this paper, we provide an in-depth comparison of explanatory techniques for DL misbehaviours. Our empirical results show that high- and low-level explanations are both understandable for humans, but provide different and complementary insights. Our human study suggests that current explanations are not always satisfactory, as may not provide human-interpretable causes of misbehaviours.

As future work, we will perform a larger evaluation, involving additional explanatory techniques, clustering solutions, DL tasks, and experts from the industry.

REFERENCES

[1] V. Riccio, G. Jahangirova, A. Stocco, N. Humbatova, M. Weiss, and P. Tonella, "Testing machine learning based systems: a systematic mapping," *Empir. Softw. Eng.*, vol. 25, no. 6, pp. 5193–5254, 2020.

[2] J. M. Zhang, M. Harman, L. Ma, and Y. Liu, "Machine learning testing: Survey, landscapes and horizons," *IEEE Transactions on Software Engineering*, 2020.

[3] N. Humbatova, G. Jahangirova, G. Bavota, V. Riccio, A. Stocco, and P. Tonella, "Taxonomy of real faults in deep learning systems," in *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*, ICSE '20, p. 1110–1121, ACM, 2020.

[4] C. Tantithamthavorn and J. Jiarpakdee, *Explainable AI for Software Engineering*. Monash University, 2021. Retrieved 2021-05-17.

[5] R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, F. Giannotti, and D. Pedreschi, "A survey of methods for explaining black box models," *ACM computing surveys (CSUR)*, vol. 51, no. 5, pp. 1–42, 2018.

[6] M. T. Ribeiro, S. Singh, and C. Guestrin, ""why should i trust you?" explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1135–1144, 2016.

[7] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic attribution for deep networks," in *International conference on machine learning*, pp. 3319–3328, PMLR, 2017.

[8] T. Zohdinasab, V. Riccio, A. Gambi, and P. Tonella, "Deephyperion: exploring the feature space of deep learning-based systems through illumination search," in *Proceedings of the 30th ACM SIGSOFT International Symposium on Software Testing and Analysis*, 2021.

[9] T. Miller, "Explanation in artificial intelligence: Insights from the social sciences," *Artificial intelligence*, vol. 267, pp. 1–38, 2019.

[10] B. Ledel and S. Herbold, "Studying the explanations for the automated prediction of bug and non-bug issues using lime and shap," *arXiv preprint arXiv:2209.07623*, 2022.

[11] P. R. Magesh, R. D. Myloth, and R. J. Tom, "An explainable machine learning model for early detection of parkinson's disease using lime on datscan imagery," *Computers in Biology and Medicine*, vol. 126, p. 104041, 2020.

[12] Y. Goyal, T. Khot, D. Summers-Stay, D. Batra, and D. Parikh, "Making the v in vqa matter: Elevating the role of image understanding in visual question answering," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6904–6913, 2017.

[13] G. E. Hinton and S. Roweis, "Stochastic neighbor embedding," *Advances in neural information processing systems*, vol. 15, 2002.

[14] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne.," *Journal of machine learning research*, vol. 9, no. 11, 2008.

[15] J. MacQueen, "Classification and analysis of multivariate observations," in *5th Berkeley Symp. Math. Statist. Probability*, pp. 281–297, 1967.

[16] R. S. King, *Cluster analysis and data mining: An introduction*. Mercury Learning and Information, 2015.

[17] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96, p. 226–231, AAAI Press, 1996.

[18] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *science*, vol. 315, no. 5814, pp. 972–976, 2007.

[19] V. Tzerpos and R. C. Holt, "Mojo: A distance metric for software clusterings," in *Sixth Working Conference on Reverse Engineering, WCRE '99, Atlanta, Georgia, USA, October 6-8, 1999*, p. 187, 1999.

[20] L. Rokach and O. Maimon, "Top-down induction of decision trees classifiers-a survey," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 35, no. 4, 2005.

[21] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[22] F. Chollet, "Simple mnist convnet." https://github.com/keras-team/keras-io/blob/master/examples/vision/mnist_convnet.py, 2020.

[23] M. Omernick and F. Chollet, "Text classification from scratch." https://github.com/keras-team/keras-io/blob/master/examples/nlp/text_classification_from_scratch.py, 2020.

[24] V. Riccio and P. Tonella, "Model-based exploration of the frontier of behaviours for deep learning system testing," in *Proceedings of the ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, ESEC/FSE '20, ACM, 2020.

[25] S. Kang, R. Feldt, and S. Yoo, "Sinvad: Search-based image space navigation for dnn image classifier test input generation," in *Proceedings of the IEEE/ACM 42nd International Conference on Software Engineering Workshops*, pp. 521–528, 2020.

[26] V. Riccio and P. Tonella, "When and why test generators for deep learning produce invalid inputs: an empirical study," in *Proceedings of the IEEE/ACM International Conference on Software Engineering*, ICSE '23, IEEE/ACM, 2023.

[27] T. Fel, L. Hervier, D. Vigouroux, A. Poche, J. Plakoo, R. Cadene, M. Chalvidal, J. Colin, T. Boissin, L. Bethune, A. Picard, C. Nicodeme, L. Gardes, G. Flandin, and T. Serre, "Xplique: A deep learning explainability toolbox," *Workshop on Explainable Artificial Intelligence for Computer Vision (CVPR)*, 2022.

[28] J. Klaise, A. V. Looveren, G. Vacanti, and A. Coca, "Alibi explain: Algorithms for explaining machine learning models," *Journal of Machine Learning Research*, vol. 22, no. 181, pp. 1–7, 2021.

[29] M. T. Ribeiro, S. Singh, and C. Guestrin, "Lime." https://github.com/marcotcr/lime, 2017.

[30] M. Hu and B. Liu, "Opinion lexicon." https://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html, 2004.

[31] M. Vidoni, "Evaluating unit testing practices in r packages," in *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE)*, pp. 1523–1534, IEEE, 2021.

[32] M. Linares-Vásquez, B. Li, C. Vendome, and D. Poshyvanyk, "Documenting database usages and schema constraints in database-centric applications," in *Proceedings of the 25th International Symposium on Software Testing and Analysis*, pp. 270–281, 2016.

[33] A. Arcuri and L. Briand, "A hitchhiker's guide to statistical tests for assessing randomized algorithms in software engineering," *Software Testing, Verification and Reliability*, vol. 24, no. 3, pp. 219–250, 2014.

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

[35] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," *Advances in neural information processing systems*, 2017.

[36] G. Montavon, S. Lapuschkin, A. Binder, W. Samek, and K.-R. Müller, "Explaining nonlinear classification decisions with deep taylor decomposition," *Pattern recognition*, vol. 65, pp. 211–222, 2017.

[37] D. Smilkov, N. Thorat, B. Kim, F. Viégas, and M. Wattenberg, "Smoothgrad: removing noise by adding noise," *arXiv preprint arXiv:1706.03825*, 2017.

[38] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, "Striving for simplicity: The all convolutional net," *arXiv preprint arXiv:1412.6806*, 2014.

[39] E. Tjoa and C. Guan, "A survey on explainable artificial intelligence (xai): Toward medical xai," *IEEE transactions on neural networks and learning systems*, vol. 32, no. 11, pp. 4793–4813, 2020.

[40] W. Samek, G. Montavon, S. Lapuschkin, C. J. Anders, and K.-R. Müller, "Explaining deep neural networks and beyond: A review of methods and applications," *Proceedings of the IEEE*, 2021.

[41] H. Fahmy, F. Pastore, M. Bagherzadeh, and L. Briand, "Supporting deep neural network safety analysis and retraining through heatmap-based unsupervised learning," *IEEE Transactions on Reliability*, vol. 70, no. 4, pp. 1641–1657, 2021.

[42] A. Stocco, P. J. Nunes, M. d'Amorim, and P. Tonella, "Thirdeye: Attention maps for safe autonomous driving systems," in *Proceedings of 37th IEEE/ACM International Conference on Automated Software Engineering, ASE*, vol. 22, 2022.

[43] T. Zohdinasab, V. Riccio, A. Gambi, and P. Tonella, "Efficient and effective feature space exploration for testing deep learning systems," *ACM Transactions on Software Engineering and Methodology*, 2023.

[44] T. Zohdinasab, V. Riccio, and P. Tonella, "Deepatash: Focused test generation for deep learning systems," in *Proceedings of the 32nd ACM SIGSOFT International Symposium on Software Testing and Analysis*, 2023.

[45] V. Nguyen, S. Huber, and A. Gambi, "Salvo: Automated generation of diversified tests for self-driving cars from existing maps," in *2021 IEEE International Conference on Artificial Intelligence Testing (AITest)*, pp. 128–135, IEEE, 2021.

[46] M. Biagiola, S. Klikovits, J. Peltomaki, and V. Riccio, "SBFT tool competition 2023 - cyber-physical systems track," in *16th IEEE/ACM International Workshop on Search-Based And Fuzz Testing, SBFT 2023, Melbourne, Australia, May 14, 2023*, 2023.