An Overview of Deep Neural Network Explainability

Anna Tzatzopoulou, 2021

***Abstract*—Aritificial intelligence, and especially deep neural networks, have evolved substancially in the recent years, infiltrating numerous domains of applications, often greatly impactful to society's well-being. As a result, the need to decode how these models operate in depth has become more vital than ever. Tending to this demand, the following paper aims to provide a thorough overview of deep neural network explainability approaches, while categorizing them in a compehensive way that paints a clear picture of the overall field.**

***Index Terms*—explainability, explainable, artificial intelligence, machine learning, deep neural networks, XAI, xDNN.**

I. Introduction

AS a result of the increasing processing power of computers, as well as the abundance of labelled data, artificial intelligence (AI) systems have reached, or in many cases surpassed, human performance on a variety of complex tasks.

Deep learning has evolved into a significant part of this recent progress in artificial intelligence, as deep neural networks (DNNs) have outperformed traditional machine learning models, such as decision trees and support vector machines, in numerous prediction tasks. Various architectures of deep neural networks have been developed, each exhibiting different properties. For instance, deep convolution networks have shown great performance in processing images and videos, whereas recurrent ones have shown great success for sequential data. However, despite their assets, deep neural networks have their own limitations and drawbacks. The most significant one is their black-box[1] nature [1], meaning that these models' internal logic is nontransparent to their users, with little support on the rationale behind their predictions.

Blindly trusting a machine learning system with no access to its decision reasoning is undesirable, or even unacceptable in some cases. For instance, in the medical domain, the reasons behind any artificially produced diagnosis or prediction should be examined by a human expert for approval [2]. Also, in self-driving cars, the features that direct each decision of the model should be accessible for confirmation, to protect safety and avoid costly incorrect actions [2]. Consequently, explainability arises as an imperative need, even necessity, for high-impact applications to protect human well-being. Explainability can also assist machine learning system developers and researchers to verify the model, improve it by locating its failure points responsible for incorrect decisions and consequently increase the system's performance and reliability. Moreover, regulators have recently started to introduce explainability-related laws, such as European Union's General Data Protection Regulation (GDPR) [3] aka "right to explanation" [4], US government's "Algorithmic Accountability Act of 2019" [5], or U.S. Department of Defense's Ethical Principles for Artificial Intelligence [6], to tackle primarily fairness, accountability and privacy risks concerning automated decision-making systems. This need for explainable decisions is recently becoming a major interest in the research community leading to the growth of a subfield of deep learning, named xDNN, that focuses on developing methods that could demystify the DNNs' modus operandi, as well as the root cause behind their outcomes.

[1]In some contexts the term black-box is used to indicate cases where the model's topology and parameters are unknown/inaccessible. However, in this paper the term black-box refers to a model being uninterpretable.

II. Contribution

This paper aims to contribute to that direction by holistically examining the progress on the research field of deep neural networks explainability.

The main contributions of this work are:

- a systematic and comprehensive overview of the up-to-date xDNN approaches.
- a novel categorization of xDNN methods according to whether they explain the model's prediction or the knowledge stored inside the model, as well as how they do so.

The rest of the paper is arranged as follows. Sections III-V discuss the discrimination of xDNN methods based on their application stage, model dependency and scope of explanation respectively. Next, Section VI presents the xDNN methods that explain the predictions of the models. Section VII examines the methods that explain the knowledge distribution inside DNNs. Finally, Section VIII summarizes the crucial points that emerge from the total of the aforementioned analysis and concludes the paper.

III. Application stage

Explainability methods can also be categorized depending on when these methods are introduced, or, in other words, based on their application stage. Methods that are applicable to the data of the model's domain, and thus irrelevant to its architecture, are called pre-model. Pre-model explainability methods are applied before building, or even selecting, the model, and they consist of exploratory data analysis techniques. Methods that are integrated into the model are called in-model (elsewhere referred to as intrinsic). In this case, explanations are constructed as a part of the decision process, thus the models that contain this kind of functionality can be characterized as self-explaining. Lastly, explainability methods that are applied as external post-processing algorithms, after

Fig. 1: Paper outline

building the model, are termed as post-model (elsewhere referred to as post-hoc). [7] [8]

IV. Model dependency

Another important characteristic to consider when classifying explainability methods is the extent of their model dependency. Methods that are tied to some specific model class by utilizing its internal components and structure are considered model-specific, whereas methods that do not assume access to the model's inner architecture, and can thus be applied to many model types, are termed as model-agnostic. This classification criterion often coincides with the distinction between intrinsic and post-hoc explainability methods, as intrinsic methods are by definition model-specific and most of the post-hoc methods are model-agnostic. [9] [7]

V. Scope of prediction explanation

According to the scope of explainability methods, two subclasses can be distinguished: global explainability and local explainability. Local explainability, aims to provide understanding of the reasons behind an individual decision of the model, i.e. the outcome for a single input instance [9]. Global explainability, on the contrary, aims to collectively explain the outcome for all possible inputs, e.g. via constructing a representative collection of local explanations or by constructing a surrogate model that approximates the to-be-explained model's behavior in an interpretable way [10].

VI. Prediction explanation

Prediction explanation refers to the explanation of the reasoning that led the model to produce a specific decision (aforementioned as local) or a set of decisions (global). By examining the literature of the state of the art works, two subclasses can be identified: attribution-based and evidence-based.

A. Attribution-based

Methods that explain deep neural networks by assigning importance or relevance values to input components, such as pixels or words, are termed as attribution-based. The goal of an attribution-based method is to reflect the contribution of these components to the output of the model. [8]

1) Attention:

A common scheme in attribution-based explainability methods is employing attention mechanisms to indicate which input parts the neural network is focusing on. Even though this technique conceptually resembles human attention, it is hard to elaborate a formal description of its generic processing. [11] defines that "An attention function can be described as mapping a query and a set of key-value pairs to an output, where the query, keys, values, and output are all vectors. The output is computed as a weighted sum of the values, where the weight assigned to each value is computed by a compatibility function of the query with the corresponding key.". Moreover, according to [12], "An attention mechanism . . . learns conditional distribution over given input units, composing a weighted contextual vector for downstream processing. ". As outlined in the example methods below, this notion can be realised into practice in various ways.

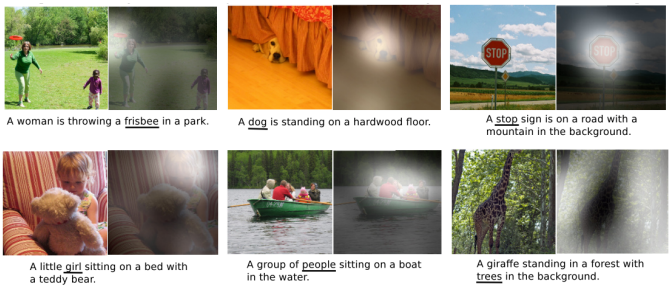


Fig. 2: Caption generation with visual attention [13].

[16] proposed an LSTM model for system log anomally detection that can be explained via an inherent attention mechanism. Given a value matrix $V_{(t)}$ that contains the LSTM hiddent states, a set of keys $K_{(t)}$ is computed for each log token: $K_{(t)} = tanh(V_{(t)}W^a)$, where W^a are learned parameters. Then, the hidden states' attention values are calculated as: $a_{(t)} = softmax(q_{(t)}K_{(t)}^T)V_{(t)}$, where $q_{(t)}$ is the query vector. Based on this, the probability distribution over the token at step t is: $p_{(t)} = softmax([h_{(t-1)} \quad a_{(t-1)}]W + b)$. [17] developed an attention-based explainable framework that detects fake news by examining news sentences and user comments. Given the feature matrix of sentences $S = [s^1, ..., s^N]$ and that of user comments $C = [c^1, ...c^T]$, sentence and comment attention maps are calculated as: $H^s = tanh(W_sS + (W_cC)F)$ and $H^c = tanh(W_cC + (W_cS)F^T)$, where $F = tanh(C^TW_lS)$ and W_l, W_s, W_c are weight parameters learned by the network. Based on these, the corresponding attention probabilities of each sentence and comment feature in S and C are computed as: $a^s = softmax(w_{hs}^T H^s)$, $a^c = softmax(w_{hc}^T H^c)$, where w_{hs}, w_{hc} are weight parameters. The attended vectors $\hat{s} = \sum_{i=1}^{N} a_i^s s^i$ and $\hat{c} = \sum_{j=1}^{N} a_j^c c^j$ are then used for the model's prediction $\hat{y} = softmax([\hat{s}, \hat{c}]W_f + b_f)$, where W_f

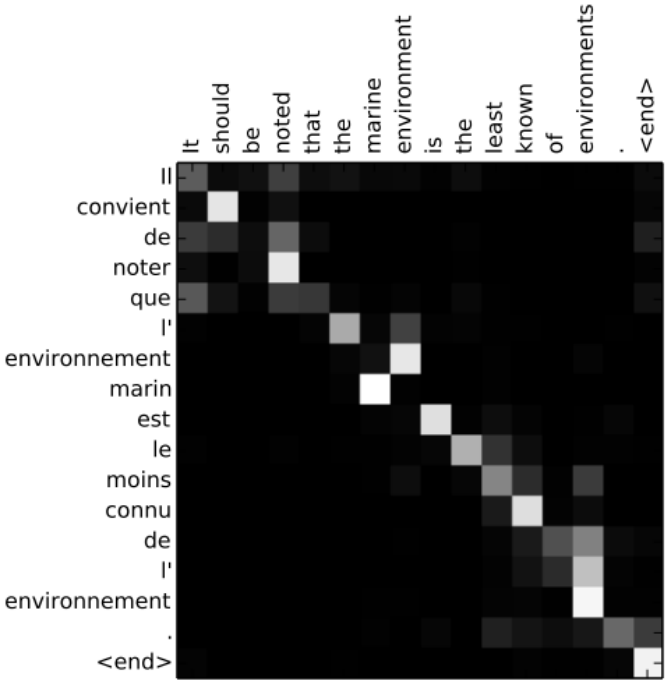


Fig. 3: Alignment between source sentence (English) and generated sentence (French) [14].

Fig. 4: Positive and negative attention-based explanations generated by SCOUTER [15].

and b_f are weight and bias terms. [18] proposed GCAN, a framework that uses the same dual co-attention mechanism to explain fake news detection. The main difference with the previous method is that, instead of utilizing user comments, GCAN examines the time propagation of user tweets that are triggered by the news source. [19] developed NAIRS, an attention-based explainable recommendation system. The attention value of each historical item j is calculated as: $a_{uj} = \frac{exp(e(p_j))}{|\sum_{k \in R_u^+} exp(e(p_k))|^\beta}$, where $e(p_j) = V^T g(W \cdot p_j + b)$ expresses the contribution of j to the user's profile, R_u^+ is the user-item interaction matrix, β is a smoothing hyper parameter, V and W are learned weights and $g(\cdot)$ is the activation function. [20] proposed an attention mechanism to interpret a deep learned drug-target interaction. Given P and D, the context matrices of a protein and a drug respectively, a soft alignment matrix $A \in \mathbb{R}^{L_p \times L_d}$ is computed: $A = tanh(P^T U D)$, where U is a learned parameter. Based on this, the attention weights that express the importance of the input units, are calculated as: $[a_p]_i = max_{1 \geq j \geq L_d} A_{i,j}$ and $[a_d]_j = max_{1 \geq i \geq L_p} A_{i,j}$. The final context vectors are then obtained as: $r_p = P \cdot softmax(a_p)$ and $r_d = D \cdot softmax(a_d)$, where $[softmax(v)]_i = \frac{e^{v_i}}{\sum_j e^{v_i}}$. [15] introduced SCOUTER (Slot-based COnfigUrable and Transparent classifiER), a model that can replace a generic FC layer to produce explainable classifications based on multiple attention modules. Given a set of image features extracted by any network, SCOUTER calculates a feature subset that supports the decision towards or against a specific category. [21] developed an attentive neural network that predicts clinical outcomes based on the patient's medical data of past hospital visits. The model consists of two levels of bidirectional GRUs, each of which incorporates an attention mechanism to express the relative contribution of a visit to the model's prediction, as well as the importance of a given visit's record. [22] introduced an attention-based Encoder-Decoder Transformer architecture for explainable VQA. The proposed model contains attention layers that generate relevancy maps indicative of the interactions between image and text inputs. [23] developed MDNet, an interpretable medical diagnosis network that utilizes an attention-augmented LSTM to learn a mapping between image features and word tokens of medical reports. [24] explains a CNN's predictions via applying attention maps on the input images. The attention values are obtained by calculating compatibility scores defined as the linear mapping from the set of feature vectors extracted at an intermediate layer to the output feature vector. Reverse Time AttentIoN (RETAIN) [25] explains a prediction based on sequences of multifactor events, by training two RNNs in a reversed order to compute the appropriate attention components. [13] implements a CNN to encode an input image to a context vector and an attentive LSTM to generate descriptive captions, where words reflect relevant parts of the image. In this way, the produced visualization indicates what the model is looking at when generating a word. In [14], an attention module is added to a neural machine translation model to allow the decoder to generate the ouput by iteratively focusing on different components of the input sentence. This technique allows the method to visualize language translation based on how respective words depend on each other.

2) Class activation mapping:
Class activation mapping is another attribution-based explanation technique, which has been developed to explain convolutional neural networks' predictions. The central characteristic of this family of methods is the computation of pixel importance heatmaps based on the CNNs' feature maps.

The original implementation of Class Activation Map (CAM) [28] computes the linear combination of A^k, the final feature maps of the convolutional network, weighted by w_k^c, the weight coefficients corresponding to unit k for class k, to generate a saliency map that reflects the areas in the image that are strongly associated to the predicted classification class . The resulting map is thus given by: $L_{CAM}^c = \sum_k w_k^c A^k$. An important limitation of [28] is that it requires the model to employ a global average pooling (GAP) layer. To remove this constraint, Gradient-weighted

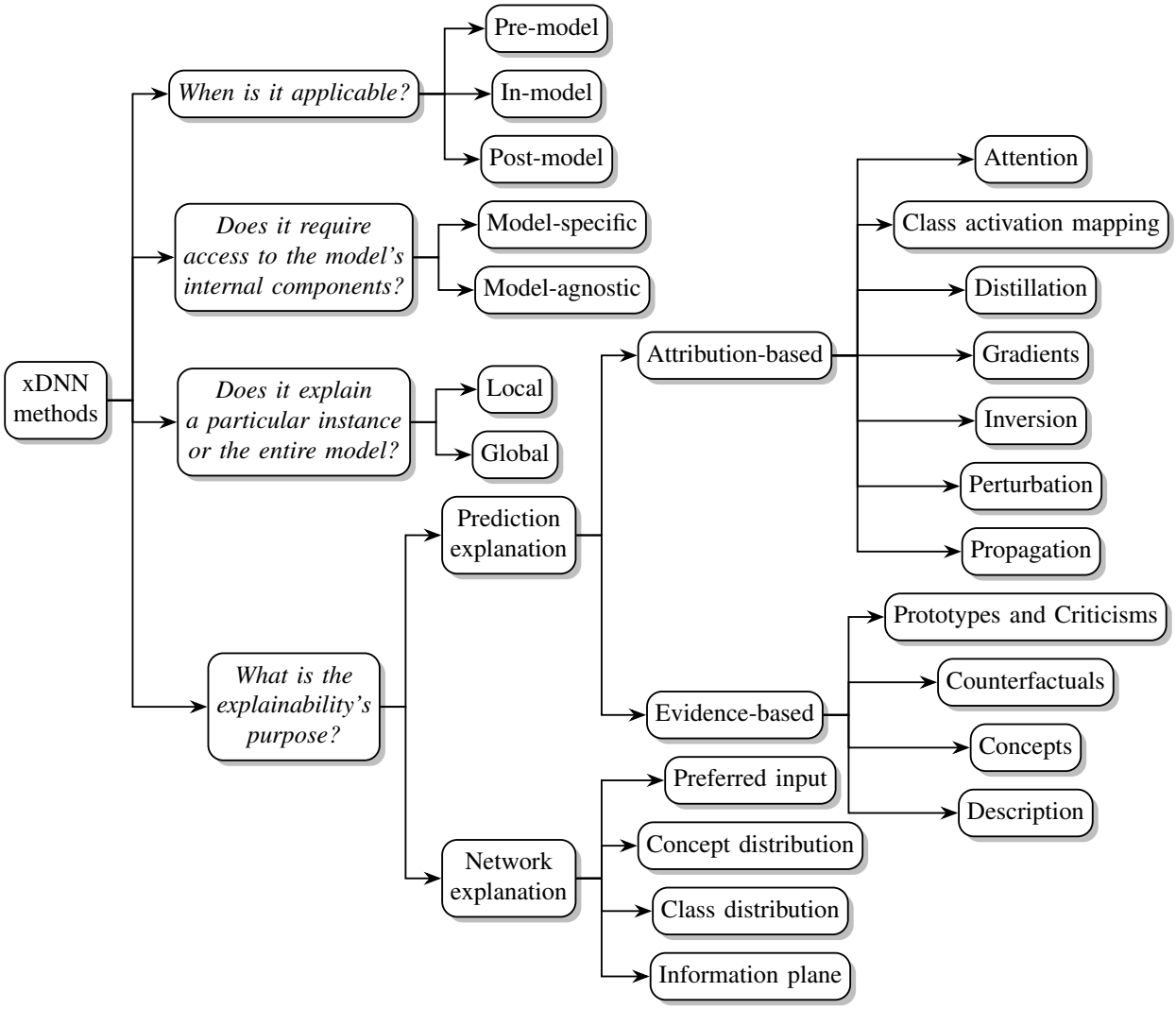
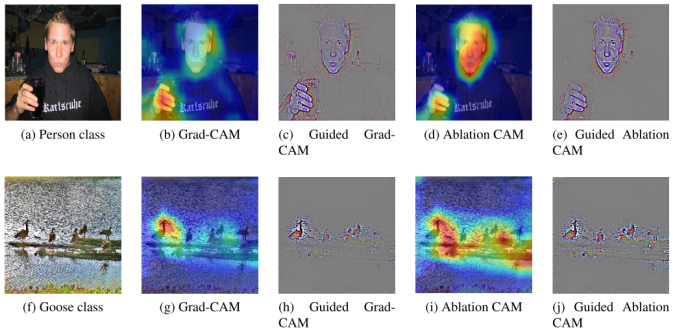


Fig. 5: Classification of xDNN methods



(a) Person class (b) Grad-CAM (c) Guided Grad-CAM (d) Ablation CAM (e) Guided Ablation CAM

(f) Goose class (g) Grad-CAM (h) Guided Grad-CAM (i) Ablation CAM (j) Guided Ablation CAM

Fig. 6: Ablation-CAM and Grad-CAM visualizations, along with their Guided form [26].

Fig. 7: Explanation generated by PRM [27].

Class Activation Map (Grad-CAM) [29] was developed as a generalization of the CAM method. Instead of employing the weights between the GAP layer and the fully connected output, Grad-CAM uses gradients as the weights to combine the different feature maps. For each feature map A_k in the final convolutional layer of the network, a gradient of the logit y_c of class c with respect to every node in A^k is computed and averaged to get an importance score a_k^c for feature map A^k: $a_k^c = \frac{1}{m \cdot n} \sum_i \sum_j \frac{\partial y^c}{\partial A_{ij}^k}$ where A_{ij}^k is a neuron positioned at (i, j) in the m×n feature map A^k. Next, Grad-CAM linearly combines the importance scores of each feature map and passes them through a ReLU to obtain a $m \times n$-dimensional saliency map: $L_{Grad-CAM}^c = ReLU\left(\sum_k a_k^c A^k\right)$. In [29], Guided Grad-CAM is also introduced as a combination of Grad-CAM with Guided Backpropagation through an element-

wise product, in order to enhance the granularity of the produced explanation by obtaining pixel-scale representations. A limitation of the Grad-CAM method is that if there are multiple occurrences of an object or parts of an object that excite different feature maps, the final saliency map fails to highlight the presence of the object in all feature maps with equal importance. Grad-CAM++ [30] addresses this problem by using a weighted average of the pixel-wise gradients. Specifically, the weights w_k^c of the Grad-CAM methodology are reformulated as: $\sum_i \sum_j a_{ij}^{kc} ReLU\left(\frac{\partial y^c}{\partial A_{ij}^k}\right)$, where $a_{ij}^{kc} = \begin{cases} \frac{1}{\sum_{l,m} \frac{\partial y^c}{\partial A_{lm}^k}}, & \text{if } \frac{\partial y^c}{\partial A_{ij}^k} = 1 \\ 0, & \text{otherwise} \end{cases}$. [31] presents Smooth Grad-CAM++, a modified version of Grad-CAM++ that produces visually enhanced saliency maps by applying a smoothening technique in the gradients calculation. The first step is to generate n noised sample images by adding Gaussian noise to the original input. Followingly, the weight coefficients a^{kc} are computed as: $a_{i,j}^{kc} = \frac{\frac{1}{n}\sum_1^n D_1^k}{2\frac{1}{n}\sum_1^n D_2^k + \sum_a \sum_b A_{a,b}^k \frac{1}{n} D_3^k}$, where D_1^k,D_2^k and D_3^k respectively denote the matrices of the 1st, 2nd and 3rd order partial derivatives for feature map k of all the noised inputs. The Grad-CAM++ weights w_k^c then become: $w_k^c = \sum_i \sum_j a_{i,j}^{kc} ReLU\left(\frac{1}{n}\sum_1^n D_1^k\right)$. There also exists a gradient-free variation of the Grad-CAM method, called Ablation-CAM, which was developed in [26]. Firstly, a component y_k^c is obtained by setting all the individual cell values of feature map A_k to zero and repeating the forward pass of the input. Then, similarly to Grad-CAM, Ablation-CAM is computed as: $L_{Ablation-CAM} = ReLU\left(\sum_k w_k^c A_k\right)$, where $w_k^c = \frac{y^c - y_k^c}{y^c}$ expresses the slope of the activation score of class c when feature map A_k is removed. [32] Score-CAM also removes the gradient dependence of the Grad-CAM method by calculating the weight of each feature map via its forward passing score on the outcome class. The final result is once more obtained by a linear combination of weights and feature maps. [33] presents ICAM, a framework that improves class activation mapping via including a variance analysis module. Specifically, a VAE-GAN is employed to disentangle class relevance from background features that are shared between classes. [34] proposes Respond-weighted Class Activation Mapping (Respond-CAM) as an extension of the original CAM method for 3D images. Firstly, the importance coefficient of each feature map A^l is computed as the weighted-average of all the gradients in the feature map using equation: $a_l^c = \frac{\sum_{ijk} A_{ijk}^l \frac{\partial y^c}{\partial A_{ijk}^l}}{\sum_{ijk} A_{ijk}^l + \epsilon}$, where y^c is the classification score for class c, A_{ijk}^l stands for the position (i,j,k) of the l-th feature map and ϵ is a sufficiently small positive number for numerical stability that can be usually ignored. Then the saliency map is computed by taking a linear combination of A^l and a_l^c: $L_{Respond-CAM}^c = \sum_l a_l^c A^l$. [35] introduces a way to enhance the class activation mapping's localization (measured by the mean Intersection over Union value). Specifically, it proposes to generate the CAM as an average of the class activation maps of a small number of highly dissimilar classes. In the first step, the task's classes

(a) Original Image (b) Explaining *Electric guitar* (c) Explaining *Acoustic guitar* (d) Explaining *Labrador*

Figure 4: Explaining an image classification prediction made by Google's Inception neural network. The top 3 classes predicted are "Electric Guitar" ($p = 0.32$), "Acoustic guitar" ($p = 0.24$) and "Labrador" ($p = 0.21$)

Fig. 8: Image classification explanation by LIME [38].

are divided into several clusters and representative classes from different clusters are selected. In the second step, given a class c_i and its representative classes $s = \{s_1, ..., s_N\}$, the activation maps by each binary classification task (c_i, s_k) are calculated and then combined to generate the CAM. Inspired by CAM, [36] proposed Regression Activation Mapping (RAM), an attribution method for CNN applied on time series data instead of temporal. The explanatory map is calculated as: $r_t = \sum_i^d z_{i,t} w_{z,i}$, where $\{z_{1,t}, ..., z_{d,t}\}$ are the input time seires to the GAP layer and $w_{z,i}$ are the weights of the ouput linear combination. Furthermore, [27] developed Peak Response Map (PRM), a method that does not in essence belong to the CAM family but will be mentioned here in absense of a more fitting attribution-based technique group. The proposed algorithm proceeds as following. Given M, the class responce maps of the top convolutional layer, the location of the maps' peaks is computed. Next, for each class the corresponding peaks are backpropagated to the input image to obtain the class's peak response map and a respective segmentation mask is selected from a proposal gallery. This selection is based on the calculation of a resemblance score that takes into account the mask and both the class and peak responce map of the class. By assembling these visualizations for all detected classes, the objects that are present in the input image can be segmented and concurrently be attributed a specific class label.

3) Distillation:

Another common approach to attribution-based explainability methods is building surrogate models, also termed as model distillation. A surrogate model is an inherently interpretable model which is trained on the decisions of a complex model to mimic its input/output behavior [9]. The distilled model approximately imitates the qualities of the original model and is thus able to provide explanations fit to the statistical properties of the latter [1]. Examples of model types that can be used as surrogates are linear models, which can be interpreted based on their parameters, and rule-based, tree-based or graph-based models, whose structure can be utilized to trace their decision back to their input source [37].

Following this approach, [38] proposed an explanation method called LIME (Local Interpretable Model-agnostic Explanations). LIME generates a new dataset by randomly perturbing sample data points in the neighborhood of the instance to be explained. Then it feeds this dataset to the target black-box model and uses its predictions, weighted by the proximity of the perturbed instances to the original ones, to train an inherently interpretable surrogate model, e.g. a decision tree

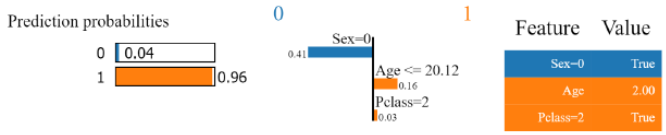


Fig. 9: LIME explanation for observation from Titanic data set. Blue color indicates the reasons for passenger's death, orange indicates reasons for survival [39].

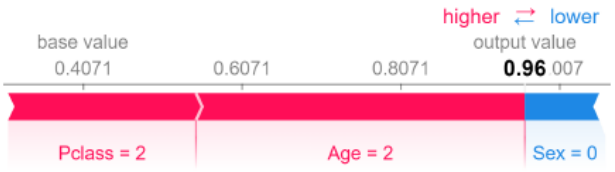


Fig. 10: SHAP explanations for observation from Titanic data set. Blue color indicates the features, which decrease probability of survival, red indicates features increasing this probability [39].

or a linear regression model. [38] also introduced SPLIME, a LIME extension that provides global explanations of the model under study, using a submodular pick algorithm that is briefly described below. Given a set of instances X, SPLIME creates a matrix W of their corresponding explanations produced by LIME. Then, it computes the feature importance for each component of W and uses greedy optimization to choose a subset of X in a way that will include the most representative features detected by the original model. The size of this subset is selected as the number of explanations that the user is willing to observe to understand the model. Followingly, the authors of [38] developed Anchor Local Interpretable Model-Agnostic Explanations (aLIME) [41], a variation of LIME that provides model-agnostic interpretations of individual decisions in the form of if-then rules. These rules are produced in such a way that alterations in the rest of the input's parameters do not affect the model's decision with high probability. Based on [38], various methods have been developed to extend LIME in other tasks or to improve issues of the original method. [42] develops Sound-LIME (SLIME), an extension of the LIME algorithm to music content analysis by temporal segmentation and frequency and

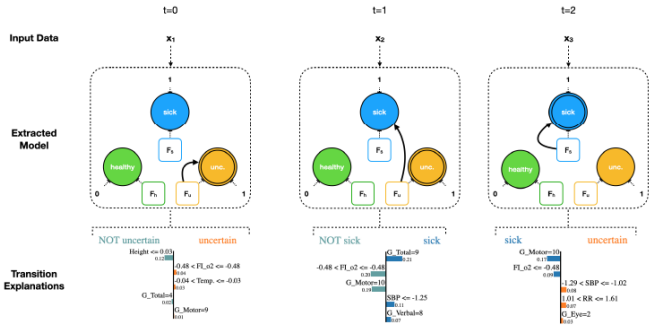


Fig. 11: Extracted sequence interpretation generated by [40].

time-frequency segmentation of an input spectrogram. In [43] the author proposes KL-LIME, a Kullback-Leibler divergence - based version of LIME that explains Bayesian predictive models. Similarly to LIME, local interpretable explanations are generated by projecting information from the predictive distribution of the original model to a simpler interpretable probabilistic explanation model, with respect to minimizing the KL-divergence between their decisions. Quadratic- LIME (QLIME) [44] expands the linear relations of LIME to non-linear relationships, by considering the linear approximations as tangential steps within a complex function. [45] suggests an approach that enhances the stability of LIME algorithm. First, agglomerative Hierarchical Clustering (HC) is deployed to create groups of training data and the K-Nearest Neighbor (KNN) algorithm is used to find the closest neighboring cluster to the instance to be explained. Then, this selected cluster is passed as input to the perturbation step of the LIME algorithm and the computations continue as in the original method. GraphLime [46] is an extended version of the LIME algorithm for deep graph models, that studies the importance of different node features for node classification tasks. Given a target node in the input graph, GraphLime considers the set of N-hop neighboring nodes and their predictions as its local dataset. Then a nonlinear surrogate model is created to fit the local dataset, using Hilbert-Schmidt Independence Criterion (HSIC) Lasso, a kernel-based feature selection algorithm. Finally, based on the weights of different features in HSIC Lasso, the most important features are selected and regarded as the explanations of the original GNN prediction. In [47] the authors introduced Modified Perturbed Sampling operation for LIME (MPS-LIME), a variation of the LIME algorithm that considers the correlation between features, instead of generating perturbed samples from a uniform distribution. The authors formulate the picking procedure of the data points as a clique set construction operation, by converting the original image's superpixels to an undirected graph. This formulation makes the method more time-efficient and enhances the understandability of the produced explanations. To reduce the time complexity, [48] introduces STREAK, which is similar to LIME [38] but approaches the new dataset generation differently. Instead of randomly generating instances to train an interpretable linear model, STREAK directly selects critical input components by greedily solving a combinatorial maximization problem. In an image classification task for example, the input image which is predicted as a certain class by the original black-box model, is first segmented into superpixels via an image segmentation algorithm. Then, a superpixel set is iteratively formed to contain the parts of the image that maximize the probability of the opaque model to predict the given class and thus are the most important for the black-box model's decision. This dataset is followingly used to create an interpretable surrogate model in a way similar to the LIME explanation method [38].

Another explanation method that follows the distillation approach is SHAP (SHapley Additive exPlanations) [49], based on the Shapley Value [50] developed for coalitional game theory to implement fair reward-sharing among cooperative participants. SHAP approximates the original model via a

mechanism of additive feature attribution methods. Specifically, it builds a linear function of input features to obtain their relative contribution in the model's decision. [49] also proposes several variations of the baseline SHAP method: KernelSHAP reduces the multitude of evaluations required for large inputs, LinearSHAP estimates SHAP values from a linear model's weight coefficients by assuming input feature independence, Low-Order SHAP is efficient for small maximum coalition size, Max SHAP reduces the complexity for the Shapley values computation of a max function, DeepSHAP employs DeepLIFT for the approximation of the SHAP values to improve computational performance by tapping into the compositional nature of deep neural networks. [51] proposed an extension of KernelSHAP that explains the model's decision by combining individual Shapley values for groups of dependent features. [52] developed Neuron Shapley, a framework that utilizes SHAP to compute the contribution of neurons to the model's individual decisions, as well as general performance. [53] introduced causal Shapley values that calculate the contribution of input features to the model's prediction splitting their effect to direct and indirect. [54] proposed Asymmetric Shapley values (ASVs), a variation of Shapley values that attends to consolidate any causal structure that may exist in the input data, by restricting the features' permutations during the SHAP calculation to the given causal ordering. [55] proposed L-Shapley and C-Shapley, two Shapley value variation approaches that achieve linear complexity when the data is graph-structured. L(ocal)-Shapley value is calculated by restraining the Shapley value to a neghbourhood of the target variable, whereas C(onnected)-Shapley value further limits the computation to regard only connected subgraphs of the local neighbourhood. [56] introduced Shapley Additive Global importancE (SAGE), a framework that combines the SHAP methodology with the notion of input features' predictive power, i.e. their exclusion's degrading effect on the model's prediction. In essence, SHAP is applied to the difference between the model's prediction loss when all features are considered and when a certain feature or subset of features is ignored.

[57] also proposes a distillation-based explanation method, the Variational Information Bottleneck for Interpretation (VIBI). VIBI adopts an information theoretic principle, the information bottleneck (analyzed later on in this survey), as a criterion for finding brief but comprehensive explanations. VIBI is composed of an explainer and an approximator, both modeled by a deep neural network. The explainer returns a vector of probabilities that indicate whether a chunk of features such as a word, phrase, sentence or a group of pixels will be selected as an explanation or not for each instance, and the approximator mimics the behaviour of the to-be-explained black-box model using the selected keys as the input. Using the information bottleneck principle, the explainer is trained to favor brief explanations by selecting key features that are maximally compressed about the input (briefness), while concurrently the explanations are enforced to suffice for accurate approximations to the black-box model by being informative about the output decision (comprehensive).

Distilling a neural network to provide explanations of its decisions has further been implemented through the construction of graphs. In [58], the method XGNN is proposed to explain GNNs by generating graph patterns that can maximize a prediction of the original graph model. This task is employed as a reinforcement learning problem. For each step the graph generator predicts how to add an edge to the current graph and then gets trained by feeding the generated graphs into the original GNN and receiving gradient policy feedback. The produced graphs are regarded as explanations for the target prediction and are expected to contain discriminative graph patterns. The validity and human-comprehensibility of these explanations is additionally enhanced by the incorporation of several graph rules. PGM-Explainer [59] builds a probabilistic graphical model as a surrogate to provide explanations for GNNs. Given an input graph, PGM-Explainer randomly perturbs the node features of several random nodes within the computational graph and detects the influence of the perturbation on the GNN predictions. By repeating this procedure multiple times, a local dataset is obtained. Then, the dataset is reduced by selecting the top dependent variables using the Grow-Shrink (GS) algorithm. Finally, the local dataset is fitted by an interpretable bayesian network that is used to explain the predictions of the original GNN model. In [60] and the following [61], the authors propose to explain the knowledge hierarchy behind a convolutional network's prediction, by building a semantic graph based on the CNN's feature maps. The explanatory graph has a multilayer structure, much like CNNs. Each layer corresponds to a convolutional layer of the CNN and consists of many nodes, as every layer of the CNN consists of many filters. Each node represents a part pattern of the detected object, while the edges linking different nodes encode the latent spatial relationship between layers. It is also worth mentioning that every layer expresses different granularity, meaning that the larger object parts tend to be reflected in higher layers while smaller parts appear in lower layers. The proposed method can also be utilized as a network explanation method, as, along with explaining a prediction by localizing the most relevant parts of the image, it can also illuminate the overall relationship between the convolutional filters of a CNN and the parts of the objects it identifies. [40] introduced MEME (Model Explanation via Model Extraction), a model extraction approach that distills RNNs to intepretable models resembling FSAs (Finite State Automata). Such a model is constructed via three steps: 1) the hidden space of a given RNN is approximated with a set of concepts that are extracted by summarizing clusters of the training data points' hidden states, 2) the transitions between these concepts are then approximated by a set of classifier functions, and 3) the concepts are mapped to the task's classes. At each timestep of the input sequence the produced MEME model calculates the next state from the current state and the next input token and outputs a class label based on this next state.

Another effort to gain insight into deep neural networks via surrogate models is the distillation of a DNN into a set of rules describing the logic behind its decisions. Works following this approach propose to extract rules that approximate the decision-making process of a trained network by utilizing its

inputs and outputs. [62] includes an overview of mechanisms designed to extract rules from trained neural networks. It is worth noticing that the presented approaches are not generalizable as they strongly depend on the black box and on the specific type of decision rules. ANCHORS [63] uses sample inputs and outputs of a target model to generate a rule list that can be used to highlight the most influential input areas, so-called anchors. This can be achieved by starting from an empty set of rules and iteratively adding a rule for each feature predicate. Alternatively, the computation can start from a set containing all the possible candidate rules and then select the best ones in terms of precision. MUSE [64], which is an evolution of BETA [65] by the same authors, also produces explanations by formulating decision sets. This framework allows end-users to customize the model explanations by inputting the features that interest them. Then, given the predictions of the target black-box model, MUSE generates a decision set based on an objective function that optimizes the explanations with respect to fidelity, interpretability, and unambiguity. LORE (Local Rule-Based Explanations) [66] distills the target model by learning a local inherently interpretable predictor on a neighborhood produced by a genetic algorithm. Then, based on this predictor, a set of rules is generated to explain the original model's decision as well as suggest the changes in the input's components that would result in a different output. [67] proposed a novel rule extraction approach that only takes into consideration the input neurons that were important for the classification. Similarly, Genetic Rule EXtraction (G-REX) [68] produces IF-THEN explanation rules by deploying genetic algorithms to create surrogate models, such as decision trees, kNN-rules, fuzzy-rules or regression models. MAGIX (Model Agnostic Globally Interpretable eXplanations) [69] also generates if-then rules to interpret the decision of a classifier by first extracting conditions that were important to the prediction and then running a genetic algorithm to build the explanatory set fo rules. [70] introduces Bayesian Rule Lists (BRL) a method that learns a decision list from a distribution using Bayesian techniques. Firstly, it creates a distribution of the most frequent rule patterns. Then, picking a sample rule list from this priori distribution, BRL optimizes the list by iteratively adding and editing the rules, in such a way that the new rule distribution follows the target posteriori distribution. [71] approximates the performance of an LSTM model by building a rule-based classifier. After calculating an importance score for each word component of the output, the method selects the phrases that consist of the words with the higher joint importance. These extracted phrase patterns are then used in the rule-based classifier to approximate the output of the LSTM in an interpretable way.

Similarly to the rule extraction technique described above, surrogate models able to explain black-box systems, such as neural networks, can be implemented by the construction of decision trees. For example, Model Extraction [72] generates decision trees via the Classification And Regression Trees algorithm (CART) and trains them based on a mixture of Gaussian distributions fit to the input data. [73] also used CART to distill a DNN's decision process into a decision tree. [74] proposed GBTmimic, a mimc learning approach that generates gradient boosting trees as interpretable surrogate models. PALM [75] distills a black-box model into a two-part surrogate model: a meta-model, constrained to be a decision tree, that partitions the training data, and a set of sub-models fitting the patterns within each partition. TreeView [76] implements hierarchical partitioning of the feature space to construct a decision tree that represents the mechanics of a trained deep neural network. [77] proposes to represent a black-box model's behavior by generating decision trees based on the combination of logical conjunctions across paths from inputs to predictions and logical disjunctions across all the paths related to an output class. In [78], the authors train a decision tree that explains the percentage of contribution each input component has to the output decision. The building blocks of the decision tree are extracted via mining of semantic patters and "decision modes" from the target deep neural model. Nodes close to the top of the tree correspond to common modes shared by multiple instances, while nodes at the bottom represent fine-grained modes with respect to specific instances. [79] introduced a hybrid model that contains an LSTM that is used for the main classification task, as well as an XGBoost surrogate tree classifier that interprets the contribution of the input features to the LSTM's decision. [80] proposed to enhance the approximation achieved by a surrogate decision tree by actively sampling new training points during the extraction. Instead of training a surrogate decision tree directly on the DNN's data, [81] proposed to use learned filters in the tree's training to obtain more informative targets. [82] developed Global Interpretation via Recursive Partitioning (GIRP), an algorithm that, given a set of local instance-wise explanations, generates a surrogate tree that interprets a black-box model globally. To train the binary tree, input variables are averaged from the these explanations between divident spaces and their importance score difference is maximized through recursive partitions.

4) Gradients:

Attribution-based explanations can also be derived from some computation of the gradients of the model's output with respect to the input. The determined gradient-based values are regarded as approximations of the contribution of input features, elicited from the idea that larger gradient volume represents a more substantial sensitivity, and thus relevance, of a feature to the output decision. [1]

Sensitivity analysis (SA) [84] [85] [86] [87] [88] [89] [90] [91] directly employs the squared values of gradients. Mathematically, it quantifies the importance of each input variable i as $R_i = \left\| \frac{\partial f(x)}{\partial x_i} \right\|$. This value indicates how much a change in each input factor would a change the prediction f in a small neighborhood around the input. It is assumed that the most relevant input features are those to which the output is most sensitive. The Global Sensitivity Analysis (GSA) method [92] extends the applicability of SA methods, allowing them for example to deal with discrete variables and distinct scanning functions. Guided Backpropagation (GBP) [88] [93] follows the same idea as Sensitivity Analysis but modifies the gradient backpropagation by restricting negative gradients from flowing back through ReLU units. This generally results

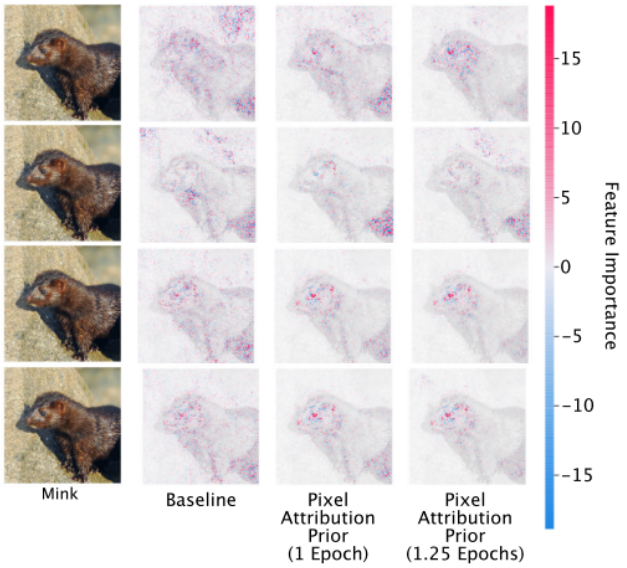


Fig. 12: Attribution maps generated by Gradients, Gradients*Input, Integrated Gradients and Expected Gradients (top to bottom) before and after fine-tuning using an attribution prior [83].

in sharper explanations. Gradient*Input (GI) [94] is a method that was initially introduced to enhance the display resolution of attribution maps. Explanations are produced by computing the partial derivatives of the output with respect to the input and element-wise multiplying them with the input itself: $R_i = \frac{\partial f(x)}{\partial x_i} \cdot x_i$. The input in this product acts as a smoothing filter that reduces the noise of the attribution. [95] shows that Gradient*Input is equivalent to DeepLift and ε-LRP (methods analyzed in the following paragraph) for a network that contains only ReLUs and no additive biases. Similarly to Gradient * Input, Integrated Gradients (IG) [96] also utilizes the partial derivatives of the output with respect to each input feature. However, instead of computing a single derivative evaluated at the provided input x, Integrated Gradients integrates the gradient of the model function at all points along the straight-line path from a baseline input x' (usually a zero matrix or vector) to x: $IG_i(x) = (x_i - x'_i) \int_0^1 \frac{\partial f(x' + a(x - x'))}{\partial x_i} da$, where i describes the dimension along which the gradient is calculated and a is associated with the path from x to x' and is smoothly distributed in range $[0, 1]$. Integrated Gradients is in essence a smooth version of Gradient * Input and reduces to the latter when the function is linear on the integration domain. Built on Integrated Gradients, [97] introduced XRAI, a method that improves attribution in cases when the image is not well fixed towards the object of interest. To achieve this, XRAI segments the input image to many overlapping parts of varrying contours and then fuses smaller regions to larger segments based on their IG-based attribution importance. Furthermore, as a substitute for Integrated Gradients that removes the influence of the selected baseline, [83] introduced Expected Gradients (EG): $EG(x) = \int_{x'} \left((x_j - x'_j) \int_0^1 \frac{\partial f(x' + a(x - x'))}{\partial x_j} da \right) \cdot p_D(x') dx'$, where D is the distribution of the underlying data domain. [98] proposed another variation of Integrated Gradients, called Blur Integrated Gradients (BlurIG), that obtains the integration path by successively blurring the input via a 2D Gaussian blur filter. Based on this idea, the integration is computed as: $BlurIG(x) = \int_{a=\infty}^{0} \frac{\partial f(L(x,a))}{\partial L(x,a)} \frac{\partial L(x,a)}{\partial a} da$, where $L(x, a)$ is the convolution of the input with the Gaussian kernel. This is claimed to produce sharper heatmaps by producing localization "in both scale/frequency and space". [99] developed Integrated Hessians, an extension of Integrated Gradients that takes into account the interactions between the features within the neural model. Specifically, this method computes the importance of input feature i with respect to the feature j as: $\Gamma_{i,j}(x) = (x_i - x'_i)(x_j - x'_j) \int_{\beta=0}^{1} \int_{\alpha=0}^{1} \alpha\beta \frac{\partial^2 f(x' + \alpha\beta(x - x'))}{\partial x_i \partial x_j} d\alpha d\beta$. To achieve more precise confinement of the produced heatmap around the localized object, [100] developed another gradient-based method, named Full-Gradient. Given a ReLU neural network $f(x, b) = \nabla_x f(x, b)^T x + \nabla_b f(x, b)^T b$, where b are its biases, $\nabla_x f(x, b)$ is defined as the input-gradient and $\nabla_b f(x, b) \odot b$ as a bias-gradient. Based on these, FullGrad aggregates the input-gradient with bias-gradients of intermediate layers to produce the final FullGrad heatmap. As an improvement for the above type of methods, SmoothGrad [101] was developed, a technique that can utilize any gradient-based method and produce a sharper, more accurate attribution map. Specifically, SmoothGrad alleviates noise by averaging out the explanations over a batch of noisy copies of the input. Given an explanation E, SmoothGrad is defined as $E_{sg}(x) = \frac{1}{N} \sum_{i=1}^{N} E(x + g_i)$, where noise vectors $g_i \sim N(0, \sigma^2)$ are drawn from a normal distribution. There is also a variance analog of SmoothGrad, VarGrad introduced in [102], that aggregates the noisy estimates by computing the variance of the noisy set rather than the mean: $E_{vg}(x) = Var(E(x + g_i))$.

5) Inversion:

Inversion is another attribution-based technique, which has been developed to explain deep convolutional networks. Its key idea is to approximately reconstruct the original image from intermediate layers' feature maps and project the inverse representation on the input image as a heatmap. The motivation of this approach is to reveal what image information is preserved in the inner layers, as it is assumed that the convolutional layers mainly consider the features that are relevant for the network's decision. [103]

[106] [107] propose to reconstruct the image from each layer by using gradient descent and a regularization term. It uses the same architecture and parameters as the original CNN before the visualization target layer and adjusts each pixel of the to be reconstructed image to minimize the objective loss function error between the target reconstruction feature map and the feature map of the original input image. [108] [109] reconstruct the image by training a dedicated Up-convolutional Neural Network (UpconvNet). The UpconvNet computes an inverse path for the feature maps back to the image dimension. The parameters of the UpconvNet are adjusted to minimize the objective loss function error between the reconstructed image and the original input image. Guided Feature Inversion [110] generates an inverted image representation consisting of the weighted sum of the original image and another noisy

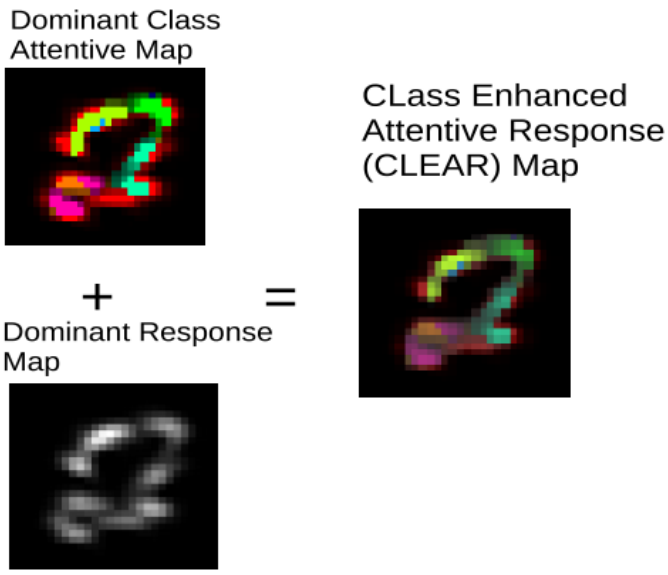


Fig. 13: Visual explanation maps generated by CLEAR [104].

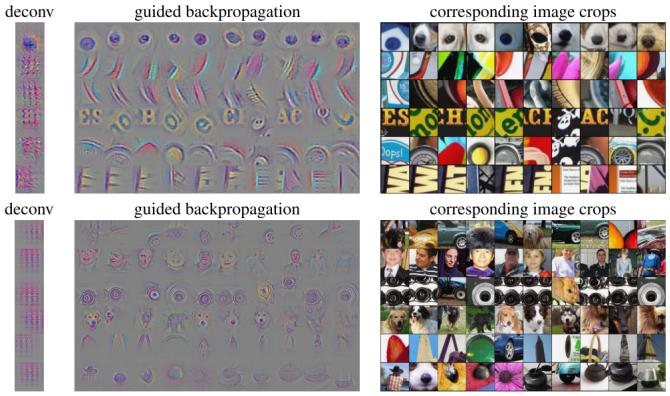


Fig. 14: Visualization of patterns learned by the layer conv6 (top) and layer conv9 (bottom) of a network. Each row corresponds to one filter [93].

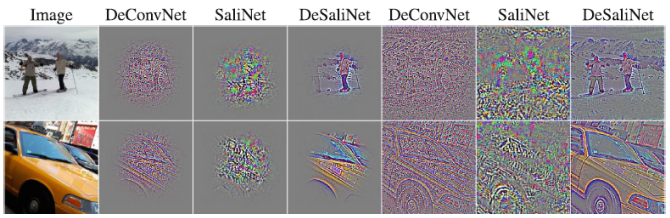


Fig. 15: Comparison of DeconvNet with DeSaliNet [105].

background image, i.e. a Gaussian white noise image. The weights are calculated to highlight the smallest area that contains the most relevant features and to blur out everything else.

Another form of inversion, that can illuminate the image information that a CNN detects, is deconvolution. In [111], the authors proposed a deconvolutional neural network structure, referred to as DeconvNet, aiming to capture certain general features for reconstructing the original image by projecting a highly diverse set of low-dimension feature maps to high dimension. DeconvNet implements the inverse operations of a CNN, consisting of reversed convolutional layers (namely the deconvolutional layer), reversed rectification layers, and reversed max-pooling layers (namely unpooling layer) in the DeconvNet structure. Then, in [112] they utilized the DeconvNet structure to decompose an image hierarchically from low-level edges to high-level object parts. However, the method gained popularity when [113] proposed its application on visualizing higher layer features in the input space, in other words mapping hidden features on pixels, to shed light into the learned activation of convolutional layers. Central to the [113] method are discriminative patterns encoded in the max pooling layers of the network, that contain the information about which pixels in the input space are of the highest importance for the resulting feature activations. [93] surpassed this limitation, generalizing the DeconvNet to also work with networks that do not have max pooling layers. Proceeding in this direction, [105] introduced DeSaliNet via combining the DeconvNet architecture with the sensitivity analysis of [84] to enhance the resulting heatmap's localization. Utilizing the above deconvolutional approaches, Relevant Features Selection was developed in [114]. This method uses a two-step algorithm to identify the internal features of a neural network mostly relevant for the prediction. First, a set of relevant layer/filter pairs are identified for every class of interest by finding those pairs that reduce at the minimum the differences between the predicted and the actual labels. These relevance values are elementwise multiplied with the internal filter-wise response vectors to obtain the contribution scores of the layer/filter pairs that are most important for the prediction. Then, this information is fed to the Deconvnet-based visualization method described in [115] to generate an explanation heatmap. One more method that interprets convolutional neural networks based on the work of [111] is CLass-Enhanced Attentive Response (CLEAR) [104]. By performing deconvolution on the network's last layer, CLEAR produces attribution maps that not only facilitate the visualization of the attentive regions mainly responsible for the prediction, but also the class that has the dominant influence on each pixel of these regions.

6) Perturbation:

Among the attribution-based explanation methods, there is also a line of work built on the notion that measuring how the outcoming prediction is affected by alterations of a feature can reflect this feature's contribution [1]. This technique is referred to as perturbation and it practically suggests to determine the attribution of an input feature or set of features by removing, masking or modifying them, computing the output on this new input and measuring the difference with the original output

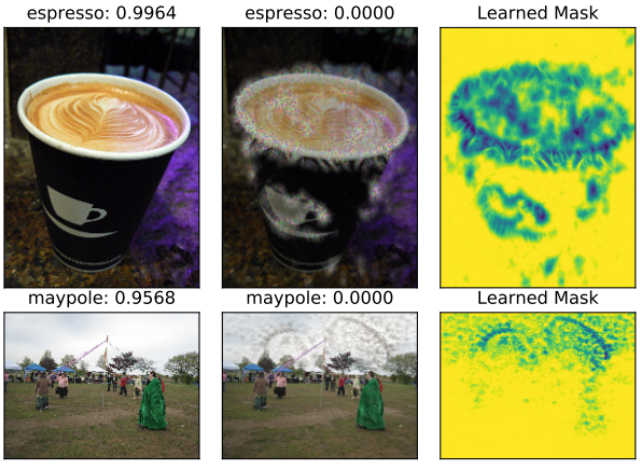


Fig. 16: An image correctly classified with large confidence; a perturbed image that is not recognized correctly anymore; the deletion mask [117].

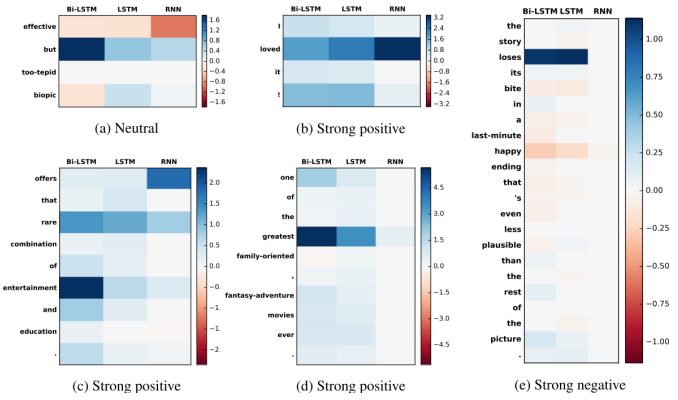


Fig. 17: Heatmap of word importance in sentiment analysis generated by Representation Erasure [118].

[116]. Previously discussed methods PGM-Explainer, LIME and SHAP use this technique to construct their surrogates.

Explain [119] was developed to calculate the contribution of a particular input variable by hiding its actual value and measuring the difference between the output probability scores predicted with the altered and the original input, $p(c|x_{-i})$ and $p(c|x)$ respectively. The marginal probability $p(c|x_{-i})$ is calculated as $p(c|x_{-i}) = \sum_{x_i} p(x_i|x_{-i})p(c|x_{-i}, x_i)$, where x denotes all input features, x_{-i} denotes all features except x_i, and the sum iterates over all possible values of x_i. Then, the prediction difference is calculated by $Diff_i(c|x) = log_2(odds(c|x)) - log_2(odds(c|x_{-i}))$, where $odds(c|x) = \frac{p(c|x)}{1-p(c|x)}$. However, the one-variable-at-a-time approach of Explain often results in unintuitive explanations, as it ignores the redundancies between input features. Attending to this weakness, [120] proposed Ime, a variation of Explain that observes all subsets of input variables. [121] also proposed a variation of Explain that replaces $p(x_i|x_{-i})$ with $p(x_i|\hat{x}_{-i})$, where $\hat{x}_{-i}$ contains only the pixels surrounding x_i instead of including all pixels except x_i. Followingly, Prediction Difference Analysis (PDA) [122] was developed, also based on Explain. Instead of assessing the contribution of a feature, in this case pixel, at a time, PDA considers patches of pixels to atone the high dependence that each pixel has on the surrounding pixels. The patches are overlapping so that, ultimately, an individual pixel's relevance is calculated as the average relevance of the different patches it was in. [123] proposed a variation of PDA that, instead of using rectangular patches, performs segmentation of the input image at multiple scales. The importance of each of these parts is then calculated in a way similar to PDA. Occlusion Sensitivity is another method following the perturbation approach, which was introduced in [113] discussed in the previous paragraph. It works by sweeping a grey patch that occludes pixels over the image and sees how the model prediction and/or the hidden layers' activation varies as the patch is covering different positions. When the patch covers a critical area the prediction performance drops significantly. Based on this notion, the method produces a visualization that depicts the most sensitive areas of an image with respect to its classification label and/or one or more intermediate layers. Representation Erasure [118] is an analogous explanation method that was developed for the natural language processing (NLP) domain. It measures the importance of each input word or each dimension of intermediate hidden activations by deleting the word or zeroing the dimension correspondingly and observing the influence on the model prediction. The influence of multiple words or phrases combined is also evaluated with the use of reinforcement by finding the minimum changes in the text that cause a flipping of the network's decision. Similarly, [124] proposed Sampling and Occlusion (SOC), an algorithm that estimates the importance of input components for an NLP classification decision, by observing the prediction difference: $\phi(p, x) = s(x) - s(x_{-p}; 0_p$, caused by replacing words or phrases p with padding tokens 0_p. Meaningful Perturbation [117] produces explanations in the forms of saliency maps by omitting information from different areas of the image and checking the resulting change in the output. The authors define three kinds of perturbations to delete information: i) constant, replacing a region with a constant value ii) noise, adding noise to the region, and iii) blur, blurring the region area. To this approach, [125] introduced an algorithm that, having found the mask that maximizes the model's ouput, optimizes its bound to obtain the smallest mask that causes at least this output level. In this way, the visualization confines the object of interest more accurately. [126] is a CNN explanation method that generates masks similarly to Meaningful Perturbation but considering SDR (Smallest Destroying Region, i.e. the minimal image region that when ommited leads the model to a false classification) along with SSR (Smallest Sufficient Region, i.e. the minimal region that is sufficient for the model to result in the correct classification) in the objective that is being optimized. The main difference however is that instead of generating the masks iteratively which has high time requirements, a novel masking model is trained to create the masks in a single forward pass. Permutation Feature Importance [127] also measures the importance of each specific feature of a deep model to the overall performance by calculating how the prediction accuracy deviates

after permuting the values of that feature. Randomized Input Sampling for Explanation (RISE) [128] perturbs an input image by multiplying it with randomized masks. The masked images are then fed to the black-box model and the scores of the target class corresponding to the masked inputs are used as coefficients to compute a weighted average of the respective masks. This sum is returned as a heatmap that indicates the important image areas for the specific prediction. [129] introduced Semantic Input Sampling for Explanation (SISE), a modified version of RISE, that uses non-random sampling to avoid instability issues. SISE is implemented in three main steps: 1) A set of feature maps is extracted from a layer of the convolutional model. 2) From this set, the most important maps are selected based on the gradient of the model's output with respect to each of them. 3) These feature maps are processed to generate targeted perturbation masks that are used in place of the random ones in the RISE methodology. Steps 1-3 are applied to multiple layers and the resulting saliency maps are aggregated to obtain the final explanation map. [130] provides explanations in the form of partitioned dependency graphs, produced in three main steps. The first step involves sampling perturbed versions of the data using a VAE. The second step uses the perturbed input-output pairs to determine dependencies between the original input and output tokens. The last step selects the explanation sets. [131] introduces Quantitative Input Influence (QII), a set of perturbation-based measures that indicate the influence of input features on the output of a black-box model. [132] proposed a method to explain Deep Echo State Networks via measuring the "pixel absence effect", in other words by observing the output difference caused by zeroing an input data point, which might be either a pixel or a point of the input sequence for image/video and time series classification tasks respectively. GNNExplainer [133] uses masks for edges and node features to identify a compact subgraph structure and a small dataset of features that are mostly responsible for a GNN's prediction. The masks are randomly initialized and their optimization is implemented by maximizing the mutual information between the predictions of the original graph and the predictions of the generated masked graph. PGExplainer [134] explains a GNN's predictions via edge masks that are produced by training a parameterized mask predictor. Given an input graph, the predictor first obtains the embeddings for each edge by concatenating node embeddings. Then the predictor uses the edge embeddings to compute the probability of each edge to be selected, which is considered as the importance score of each edge. Next, the approximated masks are sampled and applied to the graph. Finally, the mutual information between the new and the original predictions is maximized to train the predictor. GraphMask [135] explains edge importance in graph networks, by generating an edge mask for each GNN layer. It trains a classifier to predict whether an edge can be dropped without affecting the original predictions. To avoid changing the graph structure, the dropped edges are replaced by learnable baseline connections, which are vectors with the same dimensions as the node embeddings. The classifier is trained using the whole dataset by minimizing a divergence term, which measures the difference between network predictions. Contrastive Explanations Method (CEM) [136] utilizes a convolutional autoencoder (CAE) to compute the minimal amount of features that need to be present for the model to result in the same prediction ("pertinent positives"), as well as the minimal amount of features that should be absent to mantain the classification result ("pertinent negatives"). Thanks to the use of the CAE, the inflicted perturbations are closer to the data manifold, hence resulting in more human comprehensible explanations. SImilarity Difference and Uniqueness Method (SIDU) [137] is another visual explanation approach that utilizes the notion of masking the model's input image to estimate pixel saliency. Each feature activation map is converted into a binary mask M_i^c that gets bi-linearly interpolated to match the size of the input image I. The produced mask is element-wise multiplied with I resulting in a set of perturbed images A_i^c. Each A^c is then fed to the CNN model to calculate the corresponding probability prediction scores P_i^c. Based on these and the probability prediction score for the original image, P_{org}^c, similarity difference and uniqueness values are computed for each feature: $SD_i^c = exp(\frac{-1}{2\sigma^2}\|P_{org}^c - P_i^c\|)$ and $U_i^c = \sum_{j=1}^{N}\|P_i^c - P_j^c\|$ respectively. The final explanatory heatmap of the CNN's prediction is then obtained as: $S_c = \frac{1}{N}\sum_i^N W_i^c \cdot A_i^c$, where $W_i^c = SD_i^c \cdot U_i^c$. [130] developed a framework that returns causally related input and output tokens. Inputs are perturbed by a variational autoencoder and the black-box model's prediction for each of the new instances is used to create an input-ouput graph. A partitioning problem can then be solved to highlight the most important input tokens. [138] observes the model's prediction difference when input features are replaced with "uninformative" counterfactuals. [139] proposed an enhancement for perturbation-based methods that calculate the importance of an image part feature via heuristic occlusion, e.g. via blurring it or replacing it with noise. Instead, the paper proposes to use a generative model, conditioned, on the rest of the image, that removes the part of interest in a way that respects its relationship with neighbouring image regions. In this way, the perturbed instances are more realistic, thus resulting in more compact and comprehensive explanations. [140] developed another generative "in-filler" for the same cause. [141] proposed a masking perturbation approach that produces two types of explanations: "Explanation by Preservation", which indicates the minimal region of the original image that is needed for the model to result in the same prediction, and "Explanation by Deletion", that highlights the minimal region that must be deleted to alter the model's output. [142] introduced a maks generator that learns to locate image regions that are discriminative for a deep classifier's prediction. The generator produces a probability map from which a discrete mask can be sampled. The masked input image is then fed to the deep model and the cross entropy between the new and the original output is calculated to train the generator. [143] proposed CXPlain, a method that trains a seperate explanation model to interpret a predictive model by estimating the contribution of either a single input feature or a group of features to the model's accuracy. For each training data instance, the model's prediction loss when considering all available input features is calculated, as well as the loss

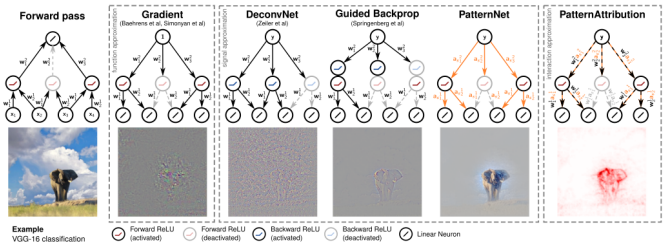


Fig. 18: Heatmaps of pixel-wise contributions [146].

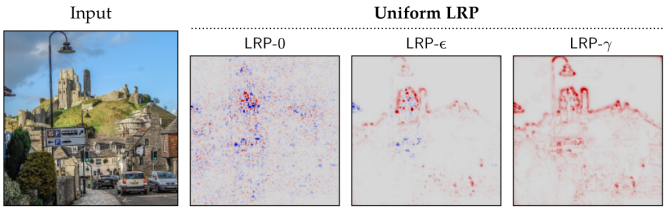


Fig. 19: Pixel-wise explanations of the output neuron 'castle' obtained with various LRP procedures. [147].

when the target input feature's information is excluded. The normalised difference between these two losses is regarded as the importance score of the feature and the explanatory model is trained to approximate this target importance distribution. [144] proposed a perturbation-based method for NLP tasks, that estimates the information contained in a word in the hidden states of a deep model's intermediate layers. This is achieved by calculating the change in the hidden state's confidence when replacing the target word with other random words.

7) Propagation:

Another mechanism that is commonly employed in attribution-based methods is decomposition with propagation rules. This approach is constituted by progressively redistributing the output backwards layer by layer. Considering the model's prediction as the initial target score, this score is decomposed and distributed to the neurons in the previous layer following the propagation rules. By repeating such procedures until the input layer, importance scores for each node of the neural network can be obtained. [145]

Deep Learning Important FeaTures (DeepLIFT) [151] computes the contribution scores of features based on the difference between the original activation of each neuron and a 'reference activation' value computed by propagating a

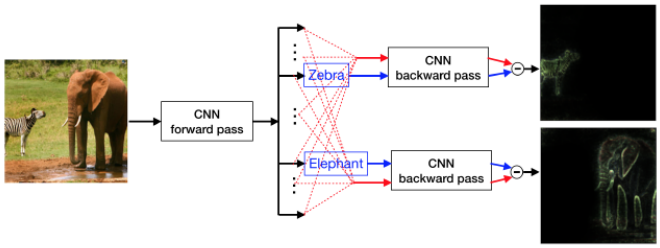


Fig. 20: Explanation generated by CLRP [148].

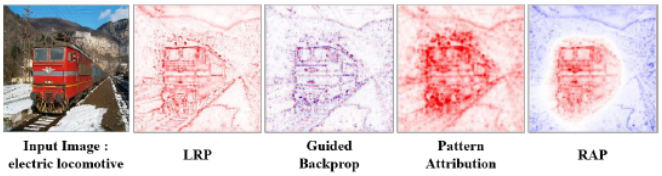


Fig. 21: Comparison of conventional propagation explanations to RSP [149].

Fig. 22: Explanations generated by RSP [150].

'reference input' (e.g. a white image) through the network. In more detail, DeepLIFT assigns a relevance score $R_{\Delta x,\Delta t}$ for input feature x_i such that: $\Delta t = \sum_{i=1}^{N} R_{\Delta x_i,\Delta t}$, where N is the number of input neurons that are necessary to compute t, $\Delta t = f(x) - f(x')$ is the difference-from-reference of an interested neuron output of the network between x and reference x', and $\Delta x = x - x'$ is the difference between x and x'. In this formulation, $R_{\Delta x_i,\Delta t}$ can be thought of as a weight denoting how much influence Δx_i had on Δt. This relevance score can be calculated via the Linear rule, Rescale rule, or RevealCancel rule, described in the paper. A multiplier $m_{\Delta x,\Delta t}$ is defined as $m_{\Delta x,\Delta t} = \frac{R_{\Delta x,\Delta t}}{\Delta x}$ indicating the relevance of Δx with respect to Δt, averaged by Δx. Given a hidden layer l of nodes $a^l = (a_1^l, a_2^l, \ldots, a_K^l)$, whose upstream connections are the input nodes $x = (x_1, x_2, \ldots, x_N)$, and a downstream target node is t, the DeepLIFT paper proves the effectiveness of the chain rule: $m_{\Delta x,\Delta t} = \sum_{j=1}^{K} m_{\Delta x_i \Delta a_j^l} m_{\Delta a_j^l \Delta t}$. This chain rule allows for layer-by-layer computation of the relevance scores of each hidden layer node via backpropagation. The DeepLIFT paper and appendix specify particular rules for computing $m_{\Delta x_i \Delta a_j^l}$ based on the architecture of the hidden layer a^l.

Layer-wise Relevance Propagation (LRP) [152] [153] [147] was proposed as a method for pixel-wise decomposition of relevance to a decision. Mathematically, it redistributes the prediction $f(x)$ backwards using local redistribution rules until it assigns a relevance score R_i to each input variable. The Relevance conservation is the key property of the redistribution process and can be summarized as $\sum_i R_i = \ldots = \sum_j R_j = \sum_k R_k = \ldots = f(x)$. This property ensures that no relevance is artificially added or removed during redistribution. The relevance score R_i of each input variable determines the variable's contribution to the prediction. A general propagation rule is given by: $R_j = \sum_k \frac{a_j \cdot (w_{jk} + \gamma w_{jk}^+)}{\epsilon + \sum_{0,j} a_j \cdot (w_{jk} + \gamma w_{jk}^+)} R_k$, where a_j is the neuron activations at layer l, R_j and R_k are the relevance scores associated to the neurons at layer l and $l+1$ respectively, w_{jk} is the weight connecting neuron j to neuron k and γ, ϵ are hyperparameters to be set. The rule reduces to LRP-γ when choosing $\gamma > 0$ and $\epsilon = 0$, it reduces to LRP-ϵ

when choosing $\gamma = 0$ and $\epsilon > 0$, and finally, setting $\gamma, \epsilon = 0$ gives LRP-0. The "alpha-beta" rule is an alternative redistribution rule: $R_j = \sum_k \left(a \cdot \frac{(x_j w_{jk})^+}{\sum_j (x_j w_{jk})^+} - \beta \cdot \frac{(x_j w_{jk})^-}{\sum_j (x_j w_{jk})^-} \right) R_k$, where $()^+$ and $()^-$ denote the positive and negative parts respectively. The conservation of relevance is enforced by the constraint $a - \beta = 1$. In [154] the original authors proposed an approach to extend LRP to neural networks with local renormalization layers. In [155] [156] additional propagation rules were proposed for LSTM blocks. [148] introduced Contrastive Layer-wise Relevance Propagation (CLRP) as an extention of LRP to generate class-discriminative explanations. The Contrastive explanation is defined as: $R_{CLRP} = max(0, (R - R_{dual}))$, where the function $max(0, X)$ means zeroing the negative elements of X. $R = f_{LRP}(X, W, S_{y_j})$ is the LRP explanation for the score S_{y_j} of the target class, where $W = \{W^1, W^2, ...W^{L-1}, W^L_j\}$ are the weights of the layers between the j-th class-specific neuron y_j and the input variables, and $R_{dual} = f_{LRP}(X, \bar{W}, S_{y_j})$ is the explanation of LRP for the "non"-classification, where $\bar{W} = \{W^1, W^2, ...W^{L-1}, W^L_{\{-j\}}\}$ are the weights connected to the output layer excluding the j-th neuron. Alternatively, $\bar{W}$ can be obtained as $\bar{W} = \{W^1, W^2, ...W^{L-1}, -W^L_j\}$. Deep Taylor Decomposition (DTD) [157] [158] [159] can be seen as a generic type of LRP. This method is based on the fact that f is differentiable and hence can be approximated by a Taylor expansion of f at some root $\hat{x}$ for which $f(\hat{x}) = 0$ and $f(x) = f(\hat{x}) + \nabla_{\hat{x}} f \cdot (x - \hat{x}) + \epsilon = \sum_i^N \frac{\partial f}{\partial x_i}(\hat{x_i}) \cdot (x_i - \hat{x_i}) + \epsilon$, where ϵ encapsulates all second order and higher terms in the Taylor expansion. A good root point is one that is as minimally different from x and that causes the function $f(x)$ to output a different prediction. The relevance score of the nodes can then be seen as the terms inside of the summation: $r_i = \frac{\partial f}{\partial x_i}(\hat{x_i}) \cdot (x_i - \hat{x_i})$. To extend this idea to a deep network, the deep Taylor decomposition algorithm considers a conservative decomposition of relevance scores across layers of the network: $r^l_i = \sum_j^M r^l_{i,j}$, where r^l_i is the relevance score of a node i at layer l and $r^l_{i,j}$ are the relevance scores of all nodes in layer $l + 1$ that node i in layer l connects to. The relevance score with respect to the input space can thus be calculated by decomposing the relevance score of the later layers to the relevance scores of former layers. The DTD propagation rule between two hidden layers is given by $R_j = \sum_k \frac{a_j w^+_{jk}}{\sum_j a_j w^+_{jk}} R_k$, where $w^+_{jk} = max(0, w_{jk})$ and a_j are positive activations. Another DTD rule specific to the input layer receiving as input pixel intensities $x_i \in [l_i, h_i]$ is given by $R_i = \sum_j \frac{x_i w_{ij} - l_i w^+_{ij} - h_i w^-_{ij}}{\sum_i x_i w_{ij} - l_i w^+_{ij} - h_i w^-_{ij}} R_k$ with $w^+_{ij} = max(o, w_{jk})$ and $w^-_{ij} = min(0, w_{ij})$. Utilizing LRP, [160] proposed a method to explain the predictions of hierarchical neural networks based on contribution propagation. Specifically, the contribution of a node X^l_i to the classification is defined as: $C(X^l_i) = \sum_{X^{l+1}_j \in parents(X^l_i)} C(X^l_i | X^{l+1}_j) C(X^{l+1}_j)$, where $C(X^{l+1}_j)$ is the contribution of node X^{l+1}_j to the classification, $C(X^l_i | X^{l+1}_j)$ is the partial contribution of node X^l_i to its parent node X^{l+1}_j. The calculations start from the contribution of the network's output nodes, which for an SVM, with a set V of support vectors, is defined as $C_{SVM}(X^L_i) = X^L_i(\sum_{v \in V} a_v V_i)$, where X^L_i is a classified feature vector. The calculation of $C(X^l_i | X^{l+1}_j)$ is further described in the paper both for radial basis and for maximum functions. Spectral Relevance Analysis (SpRAy) [161] was also built on top of the LRP method. The authors described a spectral clustering algorithm on a set of local explanations provided by LRP to understand the decision-making process of the model globally. By analyzing the occurrence frequency of attributions in LRP explanations, SpRAy identifies typical and atypical decision behaviors of the underlying model. Excitation Backpropagation (EB) [[162] [163] also uses propagation rules, built on the law of total probability. Based on the notion that the probability of a neuron in the current layer is equal to the total probabilities it outputs to all connected neurons in the next layer, EB defines the score propagation rule as a decomposition of the target probability into several conditional probability terms. PatternNet and PatternAttribution [146] emphasizes the distinction between the 'signal' dimension, which is the part of the input that contains information about the output class, and the 'distractor', which is the rest of the input, i.e. the image background, and aims at measuring the contribution of the 'signal' to the prediction as well as how good the network is at filtering out the 'distractor'. PatterNet implements the layer-wise back-projection of the estimated signal to the input space, whereas PatternAttribution calculates the neuron-wise contribution of the estimated signal to the output. It is worth noticing that PatternAttribution can be seen as a DTD extension that searches for rootpoints in the signal direction of each neuron. Similarly to the LRP notion, [149] proposed Relative Attributing Propagation (RAP), a method that assigns each neuron with a bi-polar contribution score that reflects its relevance and irrelevance to the output prediction of the DNN. In this way, the produced explanatory visualization is more attentive due to the seperated attributions. Followingly, [150] developed Relative Sectional Propagation (RSP) as a variation of RAP that is able to capture class-discriminative contributions. This method introduced the "hostile factor" which reflects the input components that oppose to the target classification's attribution. Based on this, instead of discriminating between relevance and irrelevance, RSP assigns the neurons with a bi-polar score of its target and hostile contributions.

B. Evidence-based

Methods that explain deep neural networks by providing evidence beyond the input instance are termed as evidence-based [9]. These evidence might be particular instances of the dataset that are observed as similar or counterfactual to the input, related concepts or some linguistic description, the derivation of which will be described in a following paragraph.

1) Prototypes and Criticisms:

The decision of a black-box model can be rationalized via the presentation of instances, usually from the training and testing datasets, that are either most similar to the input instance, called prototypes, or conversely less similar, called criticisms. This sort of explanation has precedent in how

humans sometimes justify actions by analogy or disanalogy. [164]

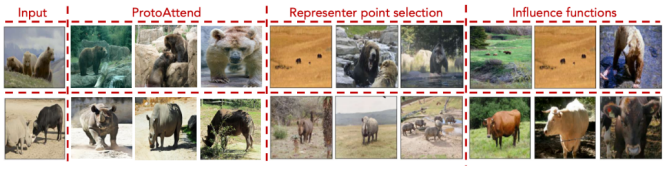


Fig. 23: Samples found by ProtoAttend vs. representer point selection and influence function [165].

[167] proposes the use of influence functions to explain a model's decision by tracing which instances from the training set were the more influential with respect to the input instance. Similarly, [168] develops an explanation method that decomposes the prediction into a linear combination of activations of training points, termed as Representer Points. This method detects both positive and negative representative values corresponding to excitatory and inhibitory training points respectively. ProtoAttend [165] also employs this linear decomposition notion but the importance weights for the training instances are learned via an attention mechanism. [164] provides the user with the entries of the training dataset that are the most similar to the to-be-explained instance, by appplying Euclidean metrics to all the input features. In an analogous way, [169] proposed to explain the recommendations of an autoencoder deep neural network based on the ratings of neighbouring users, i.e. users with similar preferences. [170] formulates Prototype Selection (PS). This approach provides a global explanation in which every instance must have a prototype corresponding to its label in its neighborhood, no instances should have a prototype with a different label in its neighborhood, and there should be as few prototypes as possible. [171] developed an algorithm called MMD-critic that analyzes a dataset to form prototype and critic groups corresponding to different instance classes. [172] generalizes [171] by building a framework that not only selects prototypes and criticisms but also associates importance weights to the instances composing each group. [173] introduced a novel architecture that bases its classification on prototypes which can

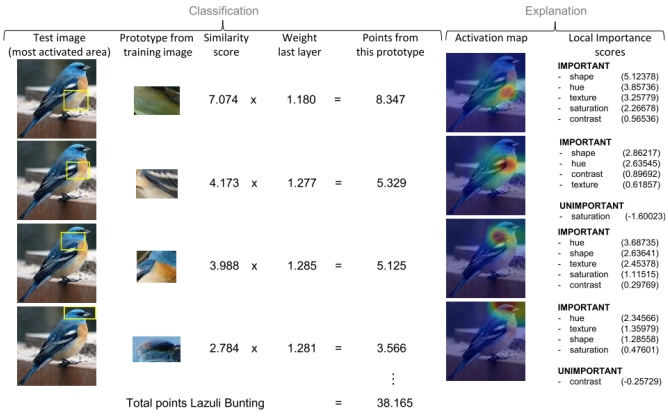


Fig. 24: ProtoPNet reasoning with a subset of all prototypes of the Lazuli Bunting class [166].

concurrently be returned as explanations. The architecture consists of a multilayer convolutional autoencoder and a prototype classifier network. Firstly, the original image is fed into the encoder and the produced feature maps are flattened into a vector. Then, the distances betwen this and a set of prototype vectors (selected so to be representative of the whole training dataset) are computed and regarded as weights for a fully connected layer followed by a softmax layer that makes the classification. Additionally, the prototype vectors can be visualized via the decoder of the network to be returned as explanations of the resulted decision. ProtoPNet (Prototypical Part Network) [174] also followed this prototype layer approach. For all instances of the training dataset, the patches of the convolutional output that are the most relevant for identifying their class (i.e. the patches that cause the strongest activations) are selected. In this way, a set of prototype parts that represent each class is collected. Each input image is first passed through a number of convolutional layers. The resulting feature maps are fed to the prototype layer where the distances between all patches of these maps and each prototype part are computed. Based on these scores a fully connected layer makes the classification and the parts with the highest similarity can be returned as its interpretation. [166] introduces an explainability method that can enhance any prototype similarity-based technique by providing extra information about visual characteristics that the model perceives as similar. Specifically, the similarity is justified based on the importance of color hue, shape, texture, contrast and saturation. To achieve this, the input image is perturbed with respect to each of these characteristics and the similarity measures between the prototype and the modified images are obtained. The importance of each visual trait can then be computed based on the difference between the prototype-image similarity of the original and the respectively perturbed images. [175] developed another prototype-based explainable deep neural network architecture. The training of the introduced framework is carried out seperately for each class, as follows. Firstly, the feature layer (that can be formed by the fully connected layer of a pre-trained convolutional neural network) extracts feature vectors for each training image of the class. Based on these features, the density layer computes the proximity between the processed images. Then, these values are fed to the typicality layer that estimates a probability distribution function (pdf) of the images. The prototypes are extracted in the prototype layer as the peaks of this pdf. Finally, the neighbouring prototypes that belong to the same class are merged in the MegaClouds layer. Based on the prototype sets that are obtained during this training, the framework classifies a new unlabeled image in the following way. A feature layer, same as before, extracts feature vectors and the similarity between the input image and the nearest prototypes of each class is computed in the prototypes layer. Then, the prototype with the maximum similarity is selected per class in the next layer, the local decision-making layer, and, by obtaining the maximum of these similarities, the dominate class is determined in the global decision-making layer and the corresponding label is assigned to the input.

2) Counterfactuals:

The counterfactual approach indicates the minimum required

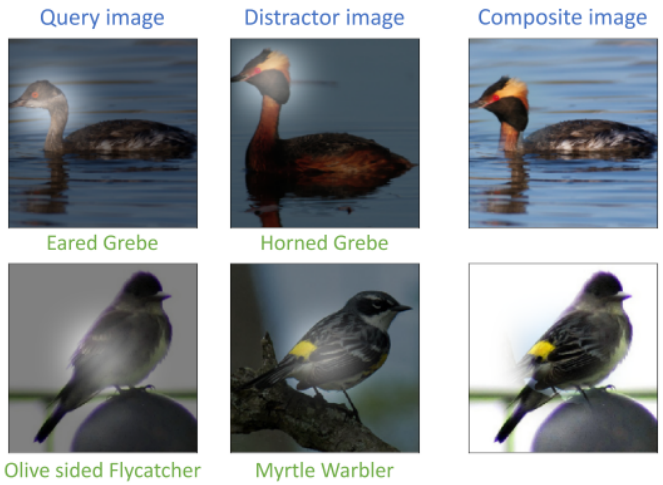


Fig. 25: Counterfactual explanations [178].

Model : NER Location Tagger
Source: N/A, Target: Perturb Location-Tag

Input Sentence: *My friend lives in beautiful London.*
Counterfactual Text Samples :
[1] My friend lives in majestic downtown Chicago.
[2] My friend lives in gorgeous London.
[3] My friend lives in the city of New Orleans.

Model : Sentiment Classifier
Source: Negative Class Label, Target: Positive Class Label

Input Sentence: *I am very disappointed with the service.*
Counterfactual Text Samples :
[1] I am very pleased with the service.
[2] I am very happy with this service
[3] I am very pleased to get a good service.

Model : Topic Classifier
Source Topic: World, Target Topic: Sci-Fi

Input Sentence: *The country is at war with terrorism.*
Counterfactual Text Samples :
[1] The country is at war with piracy at international waters.
[2] The country is at war with its own beurocracy.
[3] The country is at war with piracy offenses.

Fig. 26: Counterfactual samples generated by GYC [179].

alterations of the input features that would have led the network to an alternative decision [176]. The motivation behind this technique is that presenting users with a set of diversified examples can help shed light on how the system works, and can concurrently ease the adoption of these changes in case a different result is desirable [177].

[180] proposed DiCE (Diverse Counterfactual Explanations), a framework for providing counterfactual explanations based on determinantal point processes. The generated counterfactuals are optimized with respect to feasibility and diversity. CERTIFAI (Counterfactual Explanations for Robustness, Transparency, Interpretability, and Fairness of Artificial Intelligence) [181] uses a custom genetic algorithm to generate counterfactuals that can explain any black-box model and any type of model data. [182] introduces LIME-Counterfactual (LIME-C) and SHAP-Counterfactual (SHAP-C) by aligning the previously discussed methods LIME and SHAP with the notion of counterfactuals. Specifically, perturbations are utilized to find a set of features of the input instance (e.g. pixels or words) such that zeroing these features causes the predicted class to change. GYC (Generate Your Counterfactuals), developed in [179], is an explanation framework that generates a set of counterfactual text samples for natural language processing (NLP) models. The generation of these counterfactuals can be directed towards a specified 'condition', such as the sentiment label, the topic or the location tag. [178] implements counterfactual generation to explain image classification problems. Given the classification of an input image, this method selects another image that the model classifies differently and identifies special regions in the chosen image that replacing the corresponding regions in the input image would push its predicted class to be the same as the selected one. Also attending to the domain of image classification, CounteRGAN [183] utilizes a Residual Generative Adversarial Network (RGAN) to generate realistic counterfactual explanations. [184] developed a counterfactual generation approach for time series classification tasks. More specifically, this method locates the nearest instance that belongs to a different class and uses it as a guide for altering the input time series to produce the desired counterfactual. [185] similarly suggests a technique to utilize class representative prototypes to assist the counterfactual search procedure. [186] introduces VAEX, a Hierarchical VAE designed to to explain classifier networks by producing realistic counterfactuals, that reflect targeted modifications on selected semantic features of the original image whilst avoiding pixel loss. [187] proposes a methodology that leverages a set of 'good' counterfactual explanations, which are assumed to be already available or feasible, for the generation of other high-quality counterfactuals that can interpret new instances. [188] developed a novel recourse generating method, CRUDS (Counterfactual Recourse Using Disentangled Subspaces), that uses a Conditional Subspace Variational Autoencoder (CSVAE) to produce multiple paths that can alter a model's decision directed both by the data structure as well as the end-users. CoCoX [189] explains a CNN's decision by determining the semantic features, e.g. stripes or ears of a portrayed animal, that need to be introduced or ablated from an image to shift its prediction to another specified class. [190] introduces DACE (Distribution-Aware Counterfactual Explanation), a method that generates counterfactual explanations taking into consideration the empirical data distribution. [191] developed FACE (Feasible and Actionable Counterfactual Explanations), an algorithm that considers density-weighted metrics to locate the shortest path distances and uses these optimal "feasible paths" of change to generate counterfactual explanations. [192] proposed DECE (Decision Explorer with Counterfactual Explanations), an interactive visual analytics tool that compute and present counterfactual explanations both

on individual instances and data subsets. User interactions are also enabled to customize the generation process according to their individual needs. ViCE (Visual Counterfactual Explanations) [193] is another interactive visualization interface that explains a model's decision by providing counterfactuals. The developed algorithm starts with a set of features whose alteration is plausible. Then, the values of each feature are altered to different directions and the directions that have the greatest effect on the output decision are selected. This procedure is repeated iteratively as long as the constraints allow it or until the model's decision changes. [194] proposed to construct counterfactual explanations based on the combination of a generative model that learns to represent the underlying data distribution and a causality-based method that selects the generated latent factors that are highly influencial to the classifier's decision. [195] proposed a method that addresses the issue of counterfactual feasibility. The authors introduce a causal proximity regularizer based on constraints deducted by a structural causal model. For cases when constraints cannot be easily obtained, a VAE-based generative model is proposed that can learn to produce feasible counterfactuals based on user feedback. [196] that generates counterfactual explanations for video classification tasks. Given spatio-temporal region and corresponding linguistic attributes, the proposed explanation model is trained to calculate [attribute, region] combinations that support the output class prediction against another class of interest. Actionable Recourse Summaries (AReS) [197] generates a summary of counterfactual recourses for the entire training dataset of a model to globally explain it. Specifically, the framework learns rule sets that express the plausible recourses for subpopulations of the data and then aggregates them to obtain the desired summary. [190] proposes a framework that calculates feasible counterfactual actions by evaluating their executability based on the empirical data distribution. [198] introduces Ordered Counterfactual Explanation (OrdCE), a framework that provides ordered counterfactual actions, by taking into consideration the order of the proposed feature perturbations. The optimal combination of actions and order are calculated based on feature interactions via a mixed-integer linear optimization objective. [199] propose to gradually exaggerates features of the input instance to shift the posterior probability from the originally predicted class to its negation. Features that are irrelevant to the classification are preserved unaltered, so that the produced counterfactuals span across the data manifold in a way that clearly indicates the decision boundary of interest. [200] proposes an explanation method that gradually generates a counterfactual interpretation of the model's decision by iterating over masking and composition steps. The masking step locates the input features that are the most important to the output, and the composition step optimizes these features to set their output close to the logit space of the training data that belong to another target class. Hence, this method provides a combination of attribution and evidence based explanations.

3) Concepts:

High-level concepts, such as curls or strips, can also be presented as evidence to explain the prediction of a deep model.

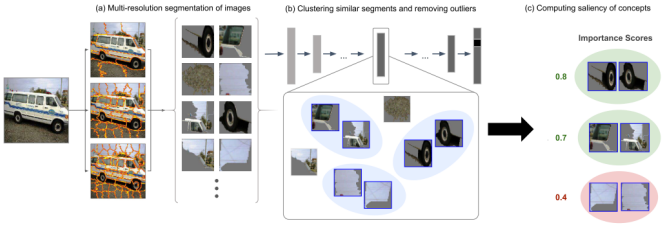


Fig. 27: ACE explanation procedure [201].

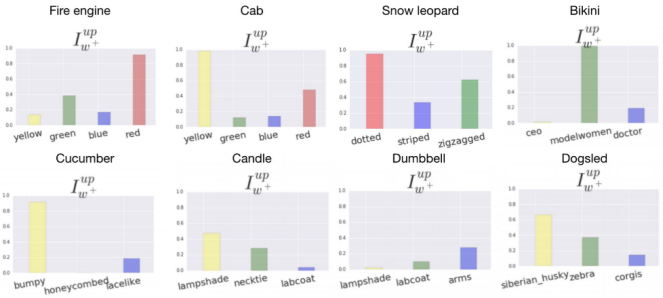


Fig. 28: Simplistic example of concept-based explanation generated by TCAV [202].

[202] [203] introduces Concept Activation Vectors (CAVs) and uses them in a novel method called Testing with CAVs (TCAV) that quantifies the degree to which a user-defined concept contributes to a classification prediction. Given input $x \in R^n$ and a feedforward layer l having m neurons, the activation at that layer is given by $f_l : R^n \rightarrow R^m$. Each class can be deduced to a set of positive concepts P_C, i.e. concepts that are humanly conceived as representative for this class (e.g. if a detected class is the animal zebra, then a positive concept would be stripes). A negative set of concepts N can be gathered, for example a set of random photos, to contrast the concepts of the class. These two sets are used to train a binary classifier $v_C^l \in R^m$ that partitions $f_l(x) : x \in P_C$ and $f_l(x) : x \in N$. CAV is thus defined as the normal vector to the hyperplane that separates the positive examples from the negative ones. Then, directional derivatives are used to evaluate the sensitivity of class predictions of f to the changes in given inputs towards the direction of a concept C for a specific layer l. If $h(l,k)$ is the logit of layer l for class k for a particular input, the conceptual sensitivity of k to C can be computed as the directional derivative $S_{C,k,l}(x) = \nabla h_{l,k}(f_l(x)) \cdot v_C^l$ for a concept vector $v_c^l \in R^m$. A TCAV score can then be calculated to find the influence of inputs towards C: $TCAV_{Q_{C,k,l}} = \frac{|x \in X_k : S_{C,k,l}(x) > 0|}{|X_k|}$, where X_k denotes all inputs with label k. In [204] the authors extended TCAV, which originally considers the presence or absence of concepts, to detect continuous concepts. Instead of seeking a discriminator between two concepts, or one concept and random inputs, their method seeks the direction of greatest increase of the measures for a single continuous concept. Firstly, Regression Concept Vectors (RCVs) are computed via least squares linear regression of the concept measures for a set of inputs. Then, the relevance of the concept is measured with bidirectional relevance scores that assume positive or

negative values based on how increasing values of the concept measures affect the classification. A further improvement over the above two methods was introduced in [205] that used a new metric called Uniform unit Ball surface Sampling (UBS) to provide layer-agnostic explanations for continuous value concepts with improved scaling to high dimensional spaces. [201] developed a method called Automatic Concept-based Explanations (ACE) that improves on TCAV by being able to explain a trained classifier without human supervision. For a set of images from the same class, each image is segmented with multiple resolutions, e.g. texture, object parts and objects. All segments are resized to the standard input size of the model and outliers are removed. Then, the segments are clustered, as examples of the same concept, using the activation space of a specific chosen bottleneck layer as similarity space. Finally, TCAV scores are computed for each concept and the most important concepts are returned as an explanation for the particular classification. Causal Concept Effect (CaCE) [206] also improved the TCAV method by examining the causal effect of the retention or ablation of high level concepts towards a deep model's prediction. In this way, the calculated concept measures are protected from the influence of biases or correlations in the dataset, which can fool methods like TCAV. CaCE can be computed exactly by using controlled environments that allow direct intervention on parts of the image generation process (GT-CaCE). Alternatively, a variational autoencoder (VAE) can be used to implement the interventions needed for the computation of CaCE (VAE-CaCE). [207] introduced ConceptSHAP, a method that employs the previously discussed SHAP method in presenting concepts that explain a model's prediction. Similar to ACE, ConceptSHAP uses segmentation and clustering to discover concepts from a dataset. However, ConceptSHAP computes the score of each individual concept by utilizing Shapley values for importance attribution. [208] explains deep networks by providing 'visual summaries' of the detected classes. First, masks that highlight the most important regions of training images are computed, in a way similar to the perturbation-based attribution methods discussed previously. Then, the component regions of each class's masked instances are clustered together. In this way, every class can be represented by a set of visual concepts that interpret the parts that mainly led to the decision of the model.

4) Description:

Lastly, another form of explanatory evidence that can be provided to the end-user of a model are linguistic descriptions of the decision-making rationale. This is generally achieved by training an auxiliary model to produce these explanations based on captions or some other form of human generated interpretations.

[211] introduced a method to justify image classification by providing natural language explanations. This is achieved by employing a caption method with a fine-grained recognition pipeline to produce strong features and then feeding the extracted captions to an LSTM that generates an explanatory sequence of words. A novel loss function, based on sampling and reinforcement learning, is suggested to optimize the sentence generation for class-specificity. [212] proposed a method that aims to find a list of visual attributes that

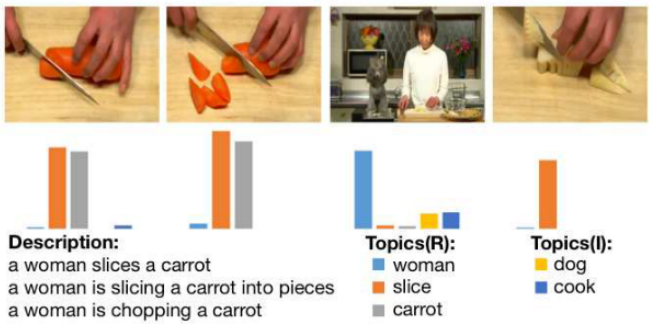


Fig. 29: Activations of one neuron according to relevant (R) and irrelevant (I) topics through time in video action recognition [209].

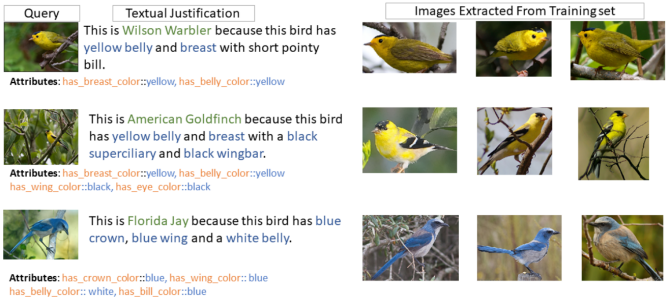


Fig. 30: Visual explanation generated by EVCA [210].

are the most important for a decision of the network. Firstly, image-level visual attributes for the training inputs are either obtained by decomposing image caption or through ground truth annotation. Then, the final convolutional layer filter is associated with these attributes, by calculating the filter attribute probability density function as a posterior probability, learned by a bayesian inference algorithm, with the input images as latent variables. Finally, based on the filter attribute probability density function (p.d.f.), the attributes are ranked by re-weighting each filter attribute p.d.f. with class specific weights. The top attributes are returned in a template sentence as evidence that explains the model's classification. [213] [214] established an AI rationalization technique, which treats the generation of explanations as a problem of translation between ad-hoc representations of states and actions in an autonomous system's environment and natural language. To do this, first, a corpus of natural language utterances was collected from people that were asked to 'think aloud' while completing the agent task's in a virtual environment. Then, this corpus was used along with state information to train an encoder-decoder neural network that translates between state-actions and natural language. This translation acts as a rationalization of the agent's behavior that can potentially be more satisfying than other types of explanations. EVCA (for Explaining Visual Classification using Attributes), developed in [210], uses captioning methods to generate descriptive attributes for each input image, a CNN for the classification task and then an LSTM to produce a natural language explanation that contains the classification class along with a list of attributes associated with the image. Additionally, it retrieves similar images from

the training set, using pairwise distance and checking that they also contain the attributes. This means that EVCA provides a combination of evidence to justify the model's decision, both description and prototypes (discussed in a previous paragraph). [215] introduced an explanation framework called TED (Teaching Explanations for Decisions). In addition to the usual two components that traditional ML models are trained on, a set of inputs X and a corresponding set of labels/decisions/classifications Y, TED also requires a third component, an explanation E for each decision. Based on this augmented training set, TED produces a classifier that predicts both labels and respective explanations. This is accomplished via a cartesian product approach that encodes Y and E together into a new classification YE, which, along with X, is provided to an ML algorithm to produce a classifier that predicts YE's. In a similar manner, GEF (Generative Explanation Framework) [216], a framework designed for NLP (Natural Language Processing) tasks, learns to produce explanations for its predictions by being fed fine-grained reasoning along with the training input and output pairs. [217] proposed FLEX (Faithful Linguistic Explanations), a method that explains a CNN's decision by returning both linguistic description and attribution maps. GradCAM (which was described in the attribution-based methods section) is utilized to obtain feature maps that are important for the model's prediction. Then, given caption word attributes of the images in the training dataset, FLEX introduces a novel way to associate words with features. For each noun/adjective w in the caption dictionary, the method computes its co-occurence score with each feature u as: $score(w,u) = \frac{2 \times occur(w,u)}{count(w)+count(u)}$, where $count(w)$ is the number of occurences of w in the ground truth descriptions, $count(u)$ is the number of occurences of u in the training dataset and $occur(w,u)$ is the number of times u and w occur together. The word w is associated with the feature u that has the highest score. Finally, based on the feature maps and the captions of the image dataset, as well as the word to feature assotiation, FLEX trains a model composed of two stacked LSTMs to generate linguistic justifications of the CNN's decisions. [218] developed ELV (Explanations as Latent Variables), a framework that treats natural language explanations as latent variables that reflect the internal reasoning of a neural model. As opposed to previous akin methods, ELV requires only a small set of human annotated explanations for training. In the first step, called E-step, the generator $p(e|x,y)$ is trained to generate explanations given labeled data. For labeled data with annotated explanations the likelihood of the ground truth explanations is maximized. For labeled data without explanations the Kullback-Leibler (KL) divergence between the variational distribution $q_\theta(e|x,y)$ and the ground truth posterior $p(e|x,y)$, which is calculated with the help of the prediction model, is minimized. During the second step, called M-step, the explanations generated in E-step are used to train the predictor $p(y|x,e)$ with maximum likelihood estimation (MLE). [219] develops an interpretable CADx (Computer Aided Diagnosis) framework (ICADx). During its training, the framework is fed with images of masses along with their corresponding standardized medical descriptions. Utilizing these inputs, a generative network is implemented to map the detected malignancies to radiologists' interpretations. A diagnostic CNN is trained jointly with this generator network to be able to provide explanatory descriptions along with its decisions. [220] introduces an HSCNN (Hierarchical Semantic Convolutional Neural Network) that is structured in a way that provides two levels of ouput; low-level semantic features and a high-level prediction. The low-level module is composed of branches, each representing a distinct human comprehensive feature. These resulting features are utilized along with the input to compute the final high-level prediction, while at the same time operating as interpretation of this decision. Given a set of descriptive attributes corresponding to each class as input, [221] proposes to learn a function that maps training images close to their respective class properties. Then, to interpret a prediction, the input instance is perturbed and attributions are predicted for both the clean and the adversial image via the computed mapping function. Based on the similarity between these two sets of attributes, the properties that are mostly important for the prediction are selected and provided as explanations. Directed perturbations are also applied to classify the image into a different class and observe the effect on its attributions. In this way, the method can additionally provide 'counter-attributes' and 'counter-examples' as explanations, resembling the previously discussed counterfactual technique. InterpNET [222] also utilizes a dataset that contains classified images along with respective class-discriminative descriptions. A large set of features is extracted for each image via a bilinear compact pooling network and fed to a fully connected (FC) network that classifies the image. The internal activations of the FC network are then provided as inputs to an RNN trained on the aforementioned dataset to generate a linguistic explanation of the classification. [223] introduces a framework that equips a CNN predictor with two interpretation modules; the explainer and the selector. The explainer generates a vector containing attribute values that describe the input image, based on explanations that were included in the training dataset. The selector uses the input and the produced linguistic explanation to select a few example instances that are most similar to the prediction of interest.

VII. Network explanation

Network explanation concerns the explanation of the distribution of knowledge that is formed inside the model, for example which layers have learned to represent each class, or how the learning evolves during the training epochs. This class of methods tends to focus on several different objectives that are discriminated followingly.

1) Preferred input:

A common objective is visualizing the preferred inputs of neurons in each layer to indicate what features they have learned.

The fundamental algorithm, termed as Activation Maximization (AM), was introduced in [226] [227] that proposed to visualize important features in any layer of a deep architecture by synthesizing an image x^* which maximizes the activation

Fig. 31: Images synthesized from scratch to highly activate certain output neurons [224].

Fig. 32: Multifaceted visualization of example neuron feature detectors from all eight layers of a deep convolutional neural network [225].

a of the chosen unit i in a layer l: $x^* = argmax_x a_{i,l}(\theta, x)$, where θ denotes the fixed network parameter sets (weight and bias). Firstly, an image $x = x_0$ with random pixel values is set to be the input to the activation computation. Next, the gradients with respect to the noise image $\frac{\partial a_{i,l}}{\partial x}$ are computed using backpropagation. Then, each pixel of the noise image is changed iteratively to maximize the activation of the neuron, by applying the update: $x \leftarrow x + \eta \cdot \frac{\partial a_{i,l}(\theta,x)}{\partial x}$ where η denotes the gradient ascent step size. This process terminates on convergence, returning a specific pattern x^* that is considered the preferred input for the target neuron. [224] built on the original activation maximization technique by utilizing an image generator network for the pixel tuning. This results in a more realistic synthesized image, which greatly improves the interpretability of the visualized patterns. This method is called Deep Generative Network Activation Maximization (DGN-AM). The DGN-AM implementation can be viewed as: $x^* = argmax_x(a_{i,l}(\theta, G(x)) - \lambda(x))$, where G indicates the generative network that takes the initial noise image as input. The above activation maximization techniques generate images without taking into consideration that each neuron can fire in response to different types of related features (e.g. the bell pepper class detects bell peppers of different colors, alone or in groups, cut open or whole, etc.), thus creating inappropriate mixes of colors, parts of objects, scales, orientations, etc. To tackle this, [225] proposed MFV (Multifaceted Feature Visualization), a method that separately synthesizes the multiple facets of each neuron by introducing an alternative initialization procedure for the activation maximization algorithm. To obtain an initialization per facet, the training images that maximally activate a neuron are projected into a low dimensional space and then clustered via k-means, and finally the n closest images to each cluster centroid are averaged to produce the initial image. The authors also introduced a novel center-biased regularization technique which reduces AM's tendency to produce repeated object fragments and instead tends to produce one central object. This is achieved by allowing more optimization iterations for center pixels than edge pixels. [228] improves the diversity and quality of the samples produced via DGN-AM [224] by introducing an additional prior that directs optimization to realistic-looking images. This is done by developing a probabilistic framework for activation maximization based on a novel type of energy-based models, called Plug and Play Generative Networks (PPGN), where the energy function is a sum of multiple constraint terms: (a) priors, that can for example bias images to look realistic, and (b) conditions, that are typically a category of a separately trained classification model. An approximate Metropolis-adjusted Langevin algorithm is used to sample iteratively from these models. In essence, PPGNs are composed of a generator network capable of drawing a wide range of image types and a replaceable "condition" network that directs what the generator produces. Hence, one is free to design an energy function, and "plug and play" with different priors and conditions to form a new generative model. Moreover, instead of conditioning on a class output neuron, PPGNs can condition on a hidden neuron, revealing many facets that a neuron has learned to detect. Thus, this method can also be seen to improve the aforementioned MFV [225] which generates the set of synthetic inputs that activate a neuron to gain a better understanding of its stimulus preferencies.

2) Concept distribution:

Another form of network explanation seeks to shed light on the way various semantic concepts (i.e. humanly conceived entities or properties, such as man, chair, stripes, color) are represented throughout a trained network.

Network Dissection [231] [232] [229] associates each convolutional neuron with semantic concepts of six kinds; scene, object, part, material, texture, and color. The authors unified several densely labeled image datasets to create the Broden dataset (Broadly and Densely Labeled Dataset) that provides a ground truth set of exemplars for a set of visual concepts. In addition, masks were applied to cover the image content not related to the assigned semantics. The correlation of a convolutional neuron, or multiple neurons, with each concept

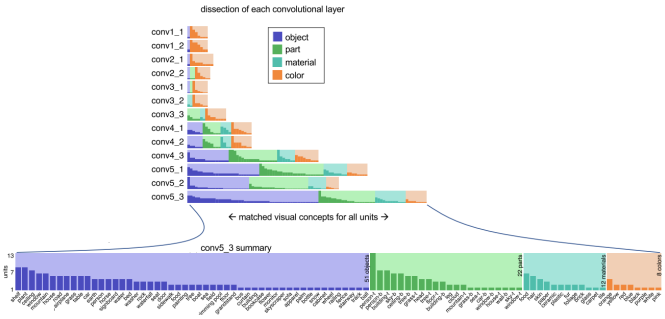


Fig. 33: Dissection of a CNN object detector [229].

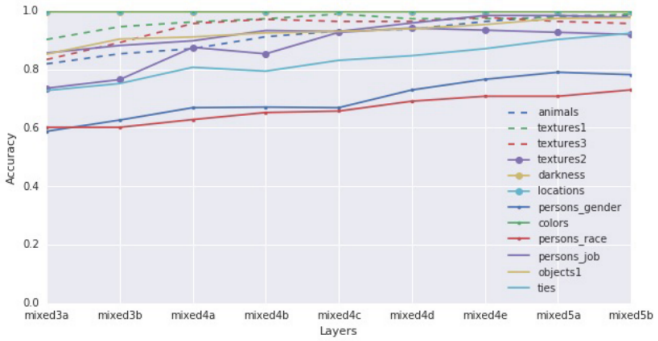


Fig. 34: Where each of the CAVs are learned [202].

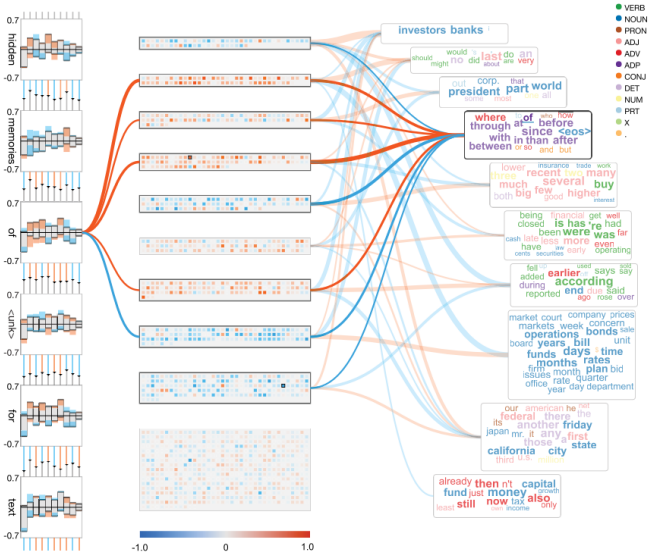


Fig. 35: Glyph-based sentence visualization, memory chips visualization for hidden state clusters, and word clouds visualization for word clusters (left to right) [230].

was evaluated by locating the neurons that strongly respond to image content with the specific semantic concept. This analysis allows the examination of what concepts and concept categories (color, object, part, material) are represented in each layer, which can shed light into how the network conceives information. Net2Vec [233] likewise maps semantic concepts to corresponding individual DNN filter responses, in a way very similar to Network Dissection. GAN Dissection [234] was designed to understand the inferential process of GANs (Generative Adversarial Networks) at different levels of abstraction (unit-, object- and scene-level). Firstly, images generated by the network are segmented and the correlation of individual units' feature maps to image regions is calculated. The units with the highest correlations are ablated by zeroing their activations and the average causal effect of this ablation is examined. In this way, the method uncovers semantic concepts that units (or groups of units) are related to. [230] introduced a method to explain the concept distribution inside recurrent neural networks (RNNs), as well as two of their variants; long short-term memory (LSTM) networks and gated recurrent units (GRUs). More specifically, the authors proposed a technique to measure the response of hidden state units to input words. Based on these responses hidden units are clusterred to 'memory chips' and words to 'word clouds' (which can be considered as concepts), and a bipartite graph that associates the two is constructed. Part Of Speech (POS) tags are utilized as well to further illuminate the hidden units' functionality. Furthermore, the authors described a glyph-based technique to extend the method to sentence-level associations by aggregate information. [209] proposes a method that computes the association between neurons and topics on the video captioning domain. Using the human descriptions that compose the training dataset, a semantically wide set of 'meaningful topics' are extracted. Then, the desired neuron-concept correspondence is estimated by computing the effect of the ablation of each individual neuron with respect to each topic for all the videos in the traning set. The previously discussed method TCAV [202] [203] can also be utilized to explain where each concept is learned in the network by examining the accuracy of the linear classifiers at every layer for each type of CAV.

3) Class distribution:

Several methods have also been developed to clarify how knowledge about the target classes is distributed across the neurons or layers of a deep neural network.

SVCCA (singular vector canonical correlation analysis) [236] is a network explainability method that combines CCA (Canonical Correlation Analysis) and SVD (Singular Value Decomposition). SVCCA finds out the relation between two layers of a network $l_k = (z_1^{l_k}, \ldots, z_{m_k}^{l_k})$ for $k = 1, 2$, where $z_i^l = (z_i^l(x_1), \ldots, z_i^l(x_m))$ is the activation of neuron i at layer l given an input data set $X = \{x_1, \ldots, x_m\}$. SVCCA uses SVD to extract the most informative components l'_k and uses CCA to transform l'_1 and l'_2 so that $\bar{l}_1 = W_X l'_1$ and $\bar{l}_2 = W_Y l'_2$ have the maximum correlations $\rho = \{\rho_1, \ldots, \rho_{min(m'_1, m'_2)}\}$. The algorithm returns the aligned directions $(\bar{x}_i, \bar{y}_i)$ and how well they align ρ_i, along with the singular values and directions $\lambda_X^{(i)}$, $x'^{(i)}$, $\lambda_Y^{(j)}$, $y'^{(j)}$. In this way, SVCCA can be used

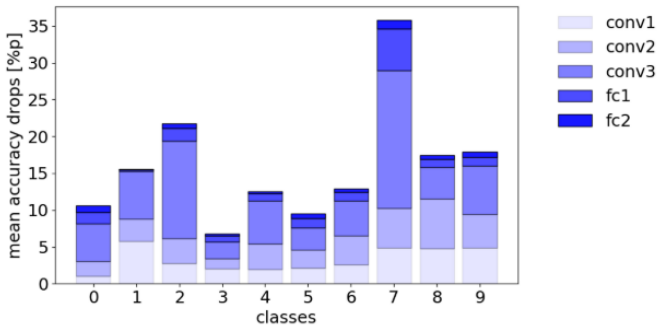


Fig. 36: Stacked bar plot of mean accuracy changes as a result of network ablations [235].

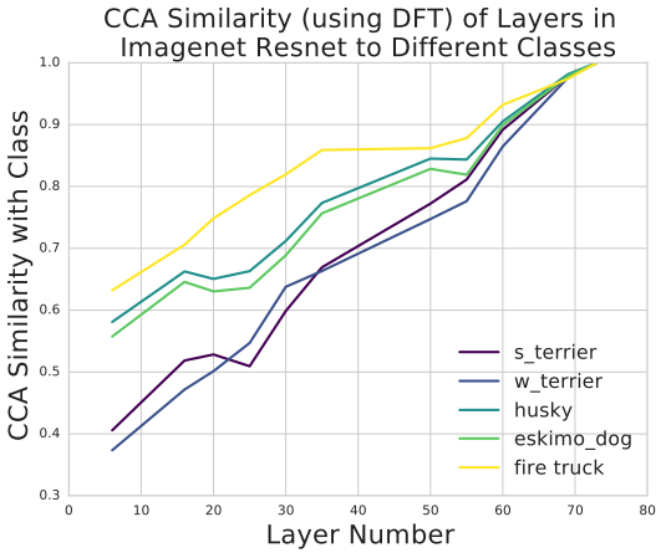


Fig. 37: CCA similarity using the Discrete Fourier Transform between the logits of five classes and layers [236].

to measure the class distribution throughout the network by comparing the correlation of representations in each layer with the logits of each class. SVCCA can also illuminate how the representation at each layer evolves during training, by comparing the final instance of the given DNN to other instances of the network at different points in the training process. [235] iteratively ablates half of the units within each layer and measures the drop in accuracy for all classes. Based on these ablation effects, the capacity that every layer devotes to the representation of each class can be computed and visualized to reveal the respective distributions. To promote class-filter interpretability, [237] proposes a strategy for training CNNs in a way that encourages class-specific filter formulation. Specifically, a novel structure called Class-Specific Gate (CSG) is introduced in addition to the standard (STD) path of forward propagation. CSG allows a filter's activation to pass only when the input samples come from a specific assigned class. In this way, the CSG path acts as regularization for filter differentiation training.

4) Information plane:

Lastly, there is a family of methods that promotes the understanding of a DNN's inner workings by analyzing its information plane; the plane of the mutual information between the input and output variables.

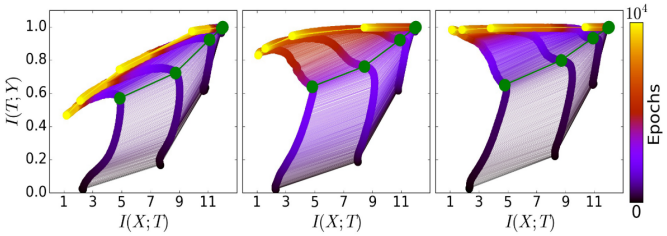


Fig. 38: The evolution of the layers with the training epochs in the information plane, for different training samples (5%, 45% and 85% of the data, left to right) [238].

This notion was seminally proposed in [239] as the Information Bottleneck (IB) method, which describes the problem of representing an observation X in a lossy manner, such that its representation T is informative of a relevance variable Y. Mathematically, the IB problem aims to find a lossy compression scheme described by a conditional distribution $P_{T|X}$ that is a minimizer of the following functional: $min_{P_{T|X}}(I(X;T) - \beta I(Y;T))$. In [240], it was proposed to analyze deep neural networks via the IB framework prinicple, by showing that any DNN can be quantified by the mutual information between the layers and the input and output variables. This representation promotes the calculation of the optimal information theoretic limits of the DNN, leading to the attainment of finite sample generalization bounds and the simplification of the network. It was also suggested that the goal of the network is to optimize the tradeoff between compression and prediction, successively for each layer. [238] continued to build on this idea by demonstrating several points of understanding that the Information-Plane visualization offers about DNNs. Specifically, it was shown that there are two distinct phases in the course of deep learning training; in the first the layers increase the information on the labels (fitting) and in the second the layers reduce the information on the input (compression). The second phase is proven to be much longer and it amounts to a stochastic relaxation that maximizes the conditional entropy of the layers subject to the empirical error constraint. Furthermore, it was demonstrated that the maps from the input to each layer (encoder) and from the layer to the output (decoder) satisfy the IB self-consistent optimality conditions. Lastly, it was established that the main advantage of hidden layers is training acceleration as they reduce the stochastic relaxation times. This slowing down of the relaxation process near phase transitions on the IB curve causes the hidden layers to converge to such critical points. [241] proposed Deep Variational Information Bottleneck (Deep VIB), a variational approximation of [239] that allows the compression scheme $P_{T|X}$ and the conditional distribution $P_{Y|T}$ to be parameterized by NNs, as well as the use of the reparameterization trick to enhance training efficiency. [242] parameterizes the information bottleneck model via neural networks trained to minimize an upper bound on $(I(X;T))^2 - \beta I(Y;T)$ using a combination of variational

and non-parametric bounding approaches. [243] formulates the principle of minimum necessary information and derives from it the conditional entropy bottleneck functional, which uses the chain rule of mutual information to replace $I(X;T)$ by $I(X;T|Y)$. [244] employed the IB theory to understand the inner behavior of convolutional neural networks. Interestingly, it was demonstrated that the compression phase is not observed in some cases, suggesting, in contrast to the conclusions of [238], that compression is not a universal mechanism in deep learning.

VIII. Conclusion

Explainable deep neural networks is an open and active field of research, with various approaches emerging every year. This paper presents a clear categorization and comprehensive overview of existing explainability techniques, aiming to provide a good insight into the current progress on xDNNs and a deeper understanding of their application. Considerable research has been focused on post-model explainability algorithms due to the ease of their application. Furthermore, the great majority of the developed methods aim to explain the models' predictions. Therefore, it seems that more focus is needed towards approaches that shed light into the inner behavior of the networks, in order to promote a more holistic advancement of the domain.

References

[1] M. Du, N. Liu, and X. Hu, "Techniques for interpretable machine learning," *arXiv*, 2018, ISSN: 23318422. arXiv: 1808.00033.

[2] W. Samek, T. Wiegand, and K. R. Müller, "Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models," *arXiv*, no. 1, pp. 1–10, 2017, ISSN: 23318422. arXiv: 1708.08296.

[3] G. Ras, M. Van Gerven, and P. Haselager, "Explanation methods in deep learning: Users, values, concerns and challenges," *arXiv*, pp. 1–15, 2018, ISSN: 23318422. DOI: 10.1007/978-3-319-98131-4_2. arXiv: 1803.07517.

[4] B. Goodman and S. Flaxman, "European union regulations on algorithmic decision making and a "right to explanation"," *AI Magazine*, vol. 38, no. 3, pp. 50–57, 2017, ISSN: 07384602. DOI: 10.1609/aimag.v38i3.2741. arXiv: 1606.08813.

[5] R. Wyden, *S.1108 - Algorithmic Accountability Act of 2019*, 2019.

[6] M. T. Esper, *AI ethical principles*, 2020. [Online]. Available: https://www.defense.gov/Newsroom/Releases/Release/Article/2091996/?dod-adoptsethical-principles-for-artificial-intelligence=,February2020.

[7] D. V. Carvalho, E. M. Pereira, and J. S. Cardoso, "Machine learning interpretability: A survey on methods and metrics," *Electronics (Switzerland)*, vol. 8, no. 8, pp. 1–34, 2019, ISSN: 20799292. DOI: 10.3390/electronics8080832.

[8] A. Singh, S. Sengupta, and V. Lakshminarayanan, "Explainable deep learning models in medical image analysis," *Journal of Imaging*, vol. 6, no. 6, pp. 1–19, 2020, ISSN: 2313433X. DOI: 10.3390/JIMAGING6060052.

[9] A. Adadi and M. Berrada, "Peeking Inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI)," *IEEE Access*, vol. 6, pp. 52 138–52 160, 2018, ISSN: 21693536. DOI: 10.1109/ACCESS.2018.2870052.

[10] H. Lakkaraju, J. Adebayo, and S. Singh, "Explaining Machine Learning Predictions: State-of-the-art, Challenges, Opportunities," in *Neural Information Processing Systems*, 2020.

[11] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, "Attention is all you need," *Advances in Neural Information Processing Systems*, vol. 2017-Decem, no. Nips, pp. 5999–6009, 2017, ISSN: 10495258. arXiv: 1706.03762.

[12] N. Xie, G. Ras, M. van Gerven, and D. Doran, "Explainable deep learning: A field guide for the uninitiated," *arXiv*, 2020, ISSN: 23318422. arXiv: 2004.14545.

[13] K. Xu, J. Lei Ba, R. Kiros, K. H. Cho, and A. Courville, "Show Attend and Tell-Neural Image Caption Generation with Visual Attention," in *Proceedings of the 32nd International Conference on Machine Learning*, vol. 37, 2015.

[14] D. Bahdanau, K. H. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, pp. 1–15, 2015. arXiv: 1409.0473.

[15] L. Li, B. Wang, M. Verma, Y. Nakashima, R. Kawasaki, and H. Nagahara, "SCOUTER: Slot Attention-based Classifier for Explainable Image Recognition," 2020. arXiv: 2009.06138. [Online]. Available: http://arxiv.org/abs/2009.06138.

[16] A. Brown, B. Hutchinson, A. Tuor, and N. Nichols, "Recurrent neural network attention mechanisms for interpretable system log anomaly detection," *Proceedings of the 1st Workshop on Machine Learning for Computing Systems, MLCS 2018 - In conjunction with HPDC*, no. June, 2018. DOI: 10.1145/3217871.3217872. arXiv: 1803.04967.

[17] K. Shu, L. Cui, S. Wang, D. Lee, and H. Liu, "Defend: Explainable fake news detection," *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 395–405, 2019. DOI: 10.1145/3292500.3330935.

[18] Y.-J. Lu and C.-T. Li, "GCAN: Graph-aware Co-Attention Networks for Explainable Fake News Detection on Social Media," pp. 505–514, 2020. DOI: 10.18653/v1/2020.acl-main.48. arXiv: 2004.11648.

[19] S. Yu, Y. Wang, M. Yang, B. Li, Q. Qu, and J. Shen, "NAIRS: A neural attentive interpretable recommendation system," *WSDM 2019 - Proceedings of the 12th ACM International Conference on Web Search and Data Mining*, pp. 786–789, 2019. DOI: 10.1145/3289600.3290609. arXiv: 1902.07494.

[20] K. Yingkai Gao, A. Fokoue, H. Luo, A. Iyengar, S. Dey, and P. Zhang, "Interpretable drug target prediction using deep neural representation," *IJCAI International Joint Conference on Artificial Intelligence*, vol. 2018-July, pp. 3371–3377, 2018, ISSN: 10450823. DOI: 10.24963/ijcai.2018/468.

[21] Y. Sha and M. D. Wang, "Interpretable predictions of clinical outcomes with an attention-based recurrent neural network," *ACM-BCB 2017 - Proceedings of the 8th ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics*, pp. 233–240, 2017. DOI: 10.1145/3107411.3107445.

[22] H. Chefer, S. Gur, and L. Wolf, "Generic Attention-model Explainability for Interpreting Bi-Modal and Encoder-Decoder Transformers," 2021. arXiv: 2103.15679. [Online]. Available: http://arxiv.org/abs/2103.15679.

[23] Z. Zhang, Y. Xie, F. Xing, M. Mcgough, and L. Yang, "MDNet: A Semantically and Visually Interpretable Medical Image Diagnosis Network," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[24] F. Li, H. Zhou, Z. Wang, and X. Wu, "ADDCNN: An Attention-Based Deep Dilated Convolutional Neural Network for Seismic Facies Analysis with Interpretable Spatial-Spectral Maps," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 59, no. 2, pp. 1733–1744, 2020, ISSN: 15580644. DOI: 10.1109/TGRS.2020.2999365.

[25] E. Choi, M. T. Bahadori, J. A. Kulas, A. Schuetz, W. F. Stewart, and J. Sun, "RETAIN: An interpretable predictive model for healthcare using reverse time attention mechanism," *Advances in Neural Information Processing Systems*, no. Nips, pp. 3512–3520, 2016, ISSN: 10495258. arXiv: 1608.05745.

[26] S. Desai and H. G. Ramaswamy, "Ablation-CAM: Visual explanations for deep convolutional network via gradient-free localization," *Proceedings - 2020 IEEE Winter Conference on Applications of Computer Vision, WACV 2020*, pp. 972–980, 2020. DOI: 10.1109/WACV45572.2020.9093360.

[27] Y. Zhou, Y. Zhu, Q. Ye, Q. Qiu, and J. Jiao, "Weakly Supervised Instance Segmentation Using Class Peak Response," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 3791–3800, 2018, ISSN: 10636919. DOI: 10.1109/CVPR.2018.00399. arXiv: 1804.00880.

[28] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning Deep Features for Discriminative Localization," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2016-Decem, pp. 2921–2929, 2016, ISSN: 10636919. DOI: 10.1109/CVPR.2016.319. arXiv: 1512.04150.

[29] R. R. Selvaraju, A. Das, R. Vedantam, M. Cogswell, D. Parikh, and D. Batra, "Grad-CAM: Why did you say that?," pp. 1–4, 2016. arXiv: 1611.07450. [Online]. Available: http://arxiv.org/abs/1611.07450.

[30] A. Chattopadhay, A. Sarkar, P. Howlader, and V. N. Balasubramanian, "Grad-CAM++: Improved visual explanations for deep convolutional networks," *Proceedings - 2018 IEEE Winter Conference on Applications of Computer Vision, WACV 2018*, vol. 2018-Janua, pp. 839–847, 2018. DOI: 10.1109/WACV.2018.00097. arXiv: arXiv:1710.11063v3.

[31] D. Omeiza, S. Speakman, C. Cintas, and K. Weldemariam, "Smooth Grad-CAM++: An enhanced inference level visualization technique for deep convolutional neural network models," *arXiv*, pp. 1–10, 2019, ISSN: 23318422. arXiv: 1908.01224.

[32] H. Wang, Z. Wang, M. Du, *et al.*, "Score-CAM: Score-weighted visual explanations for convolutional neural networks," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, vol. 2020-June, pp. 111–119, 2020, ISSN: 21607516. DOI: 10.1109/CVPRW50498.2020.00020. arXiv: 1910.01279.

[33] C. Bass, P. D. Tudosiu, M. da Silva, S. M. Smith, C. Sudre, and E. C. Robinson, "ICAM: Interpretable Classification via Disentangled Representations and Feature Attribution Mapping," *arXiv*, 2020, ISSN: 23318422. arXiv: 2006.08287.

[34] G. Zhao, B. Zhou, K. Wang, R. Jiang, and M. Xu, "Respond-CAM: Analyzing deep models for 3D imaging data by visualizations," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11070 LNCS, pp. 485–492, 2018, ISSN: 16113349. DOI: 10.1007/978-3-030-00928-1_55. arXiv: 1806.00102.

[35] F. Meng, K. Huang, H. Li, and Q. Wu, "Class activation map generation by representative class selection and multi-layer feature fusion," *arXiv*, 2019, ISSN: 23318422. arXiv: 1901.07683.

[36] A. Wolanin, G. Camps-Valls, and L. Gomez-Chova, "Estimating and understanding crop yields with explainable deep learning in the Indian Wheat Belt," *Environmental Research Letters*, 2020, ISSN: 01681656. DOI: 10.1016/j.enzmictec.2006.09.022.

[37] A. Das and P. Rad, "Opportunities and Challenges in Explainable Artificial Intelligence (XAI): A Survey," *arXiv*, pp. 1–24, 2020, ISSN: 23318422. arXiv: 2006.11371.

[38] M. T. Ribeiro, S. Singh, and C. Guestrin, ""Why should i trust you?" Explaining the predictions of any classifier," *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, vol. 13-17-Augu, pp. 1135–1144, 2016. DOI: 10.1145/2939672.2939778. arXiv: 1602.04938.

[39] A. Gosiewska and P. Biecek, "iBreakDown: Uncertainty of Model Explanations for Non-additive Predictive Models," no. April, 2019. arXiv: 1903.11420. [Online]. Available: http://arxiv.org/abs/1903.11420.

[40] D. Kazhdan, B. Dimanov, M. Jamnik, and P. Liò, "MEME: Generating RNN Model Explanations via Model Extraction," pp. 1–17, 2020. arXiv: 2012.

06954. [Online]. Available: http://arxiv.org/abs/2012.06954.

[41] M. T. Ribeiro, S. Singh, and C. Guestrin, "Nothing Else Matters: Model-Agnostic Explanations By Identifying Prediction Invariance," no. Nips, 2016. arXiv: 1611.05817. [Online]. Available: http://arxiv.org/abs/1611.05817.

[42] S. Mishra, B. L. Sturm, and S. Dixon, "Local interpretable model-agnostic explanations for music content analysis," *Proceedings of the 18th International Society for Music Information Retrieval Conference, ISMIR 2017*, pp. 537–543, 2017.

[43] T. Peltola, "Local Interpretable Model-agnostic Explanations of Bayesian Predictive Models via Kullback–Leibler Projections," *arXiv*, 2018, ISSN: 23318422. arXiv: 1810.02678.

[44] S. Bramhall, H. Horn, M. Tieu, and N. Lohia, "QLIME—A quadratic local interpretable model-agnostic explanation approach," *SMU Data Science Review*, vol. 3, no. 1, pp. 73–88, 2020.

[45] M. R. Zafar and N. M. Khan, "DLIME: A Deterministic Local Interpretable Model-Agnostic Explanations approach for Computer-Aided Diagnosis Systems," *arXiv*, 2019, ISSN: 23318422. arXiv: 1906.10263.

[46] Q. Huang, M. Yamada, Y. Tian, D. Singh, D. Yin, and Y. Chang, "Graphlime: Local interpretable model explanations for graph neural networks," *arXiv*, pp. 1–11, 2020, ISSN: 23318422. arXiv: 2001.06216.

[47] S. Shi, X. Zhang, and W. Fan, "A modified perturbed sampling method for local interpretable model-agnostic explanation," *arXiv*, pp. 1–5, 2020, ISSN: 23318422. arXiv: 2002.07434.

[48] E. R. Elenberg, A. G. Dimakis, M. Feldman, and A. Karbasi, "Streaming weak submodularity: Interpreting neural networks on the fly," *Advances in Neural Information Processing Systems*, vol. 2017-Decem, pp. 4045–4055, 2017, ISSN: 10495258. arXiv: 1703.02647.

[49] S. M. Lundberg and S. I. Lee, "A unified approach to interpreting model predictions," *Advances in Neural Information Processing Systems*, vol. 2017-Decem, no. Section 2, pp. 4766–4775, 2017, ISSN: 10495258. arXiv: 1705.07874.

[50] A. E. Roth, *A value for n-person games*, 1988.

[51] K. Aas, M. Jullum, and A. Løland, "Explaining individual predictions when features are dependent: More accurate approximations to Shapley values," *Artificial Intelligence*, vol. 298, pp. 1–28, 2021, ISSN: 00043702. DOI: 10.1016/j.artint.2021.103502. arXiv: 1903.10464.

[52] A. Ghorbani and J. Zou, "Neuron Shapley: Discovering the Responsible Neurons," no. NeurIPS, 2020, ISSN: 10495258. arXiv: 2002.09815. [Online]. Available: http://arxiv.org/abs/2002.09815.

[53] T. Heskes, E. Sijben, I. G. Bucur, and T. Claassen, "Causal Shapley Values: Exploiting Causal Knowledge to Explain Individual Predictions of Complex Models," no. NeurIPS, 2020, ISSN: 10495258. arXiv: 2011.01625. [Online]. Available: http://arxiv.org/abs/2011.01625.

[54] C. Frye, C. Rowat, and I. Feige, "Asymmetric Shapley values: incorporating causal knowledge into model-agnostic explainability," no. NeurIPS, 2019, ISSN: 10495258. arXiv: 1910.06358. [Online]. Available: http://arxiv.org/abs/1910.06358.

[55] J. Chen, L. Song, M. J. Wainwright, and M. I. Jordan, "L-Shapley and C-Shapley: Efficient model interpretation for structured data," *7th International Conference on Learning Representations, ICLR 2019*, pp. 1–17, 2019. arXiv: 1808.02610.

[56] I. Covert, S. Lundberg, and S.-I. Lee, "Understanding Global Feature Contributions With Additive Importance Measures," no. NeurIPS, 2020, ISSN: 10495258. arXiv: 2004.00668. [Online]. Available: http://arxiv.org/abs/2004.00668.

[57] S. Bang, P. Xie, H. Lee, W. Wu, and E. Xing, "Explaining a black-box using Deep Variational Information Bottleneck Approach," no. Mi, 2019. arXiv: 1902.06918. [Online]. Available: http://arxiv.org/abs/1902.06918.

[58] H. Yuan, J. Tang, X. Hu, and S. Ji, "XGNN: Towards Model-Level Explanations of Graph Neural Networks," *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 430–438, 2020. DOI: 10.1145/3394486.3403085. arXiv: 2006.02587.

[59] M. N. Vu and M. T. Thai, "PGM-explainer: Probabilistic graphical model explanations for graph neural networks," *arXiv*, no. NeurIPS, 2020, ISSN: 23318422. arXiv: 2010.05788.

[60] Q. Zhang, R. Cao, Y. N. Wu, and S. C. Zhu, "Growing Interpretable Part Graphs on ConvNets via Multi-Shot Learning," *Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17) Growing*, pp. 2898–2906, 2017.

[61] Q. Zhang, R. Cao, F. Shi, Y. N. Wu, and S. C. Zhu, "Interpreting CNN knowledge via an explanatory graph," *arXiv*, pp. 4454–4463, 2017, ISSN: 23318422.

[62] R. Andrews, J. Diederich, and A. B. Tickle, "Survey and critique of techniques for extracting rules from trained artificial neural networks," *Knowledge-Based Systems*, vol. 8, no. 6, pp. 373–389, 1995, ISSN: 09507051. DOI: 10.1016/0950-7051(96)81920-4.

[63] M. T. Ribeiro, S. Singh, and C. Guestrin, "Anchors: High-precision model-agnostic explanations," *32nd AAAI Conference on Artificial Intelligence, AAAI 2018*, pp. 1527–1535, 2018.

[64] H. Lakkaraju, R. Caruana, E. Kamar, and J. Leskovec, "Faithful and customizable explanations of black box models," *AIES 2019 - Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, pp. 131–138, 2019. DOI: 10.1145/3306618.3314229.

[65] H. Lakkaraju, E. Kamar, R. Caruana, and J. Leskovec, "Interpretable Explorable Approximations of Black Box Models," 2017. arXiv: 1707.01154. [Online]. Available: http://arxiv.org/abs/1707.01154.

[66] R. Guidotti, A. Monreale, S. Ruggieri, D. Pedreschi, F. Turini, and F. Giannotti, "Local rule-based explanations of black box decision systems," *arXiv*, no. May, 2018, ISSN: 23318422. arXiv: 1805.10820.

[67] M. Gethsiyal Augasta and T. Kathirvalavakumar, "Reverse engineering the neural networks for rule extraction in classification problems," *Neural Processing Letters*, vol. 35, no. 2, pp. 131–150, 2012, ISSN: 13704621. DOI: 10.1007/s11063-011-9207-8.

[68] R. Konig, U. Johansson, and L. Niklasson, "G-REX: A versatile framework for evolutionary data mining," *Proceedings - IEEE International Conference on Data Mining Workshops, ICDM Workshops 2008*, pp. 971–974, 2008. DOI: 10.1109/ICDMW.2008.117.

[69] N. Puri, P. Gupta, P. Agarwal, S. Verma, and B. Krishnamurthy, "MAGIX: Model Agnostic Globally Interpretable Explanations," 2017. arXiv: 1706.07160. [Online]. Available: http://arxiv.org/abs/1706.07160.

[70] B. Letham, C. Rudin, T. H. McCormick, and D. Madigan, "Interpretable classifiers using rules and bayesian analysis: Building a better stroke prediction model," *Annals of Applied Statistics*, vol. 9, no. 3, pp. 1350–1371, 2015, ISSN: 19417330. DOI: 10.1214/15-AOAS848. arXiv: 1511.01644.

[71] W. J. Murdoch and A. Szlam, "Automatic rule extraction from long short term memory networks," *arXiv*, no. 2016, pp. 1–12, 2017, ISSN: 23318422. arXiv: 1702.02540.

[72] H. Bride, J. Dong, J. S. Dong, and Z. Hóu, "Towards dependable and explainable machine learning using automated reasoning," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11232 LNCS, pp. 412–416, 2018, ISSN: 16113349. DOI: 10.1007/978-3-030-02450-5_25.

[73] X. Liu, X. Wang, and S. Matwin, "Improving the interpretability of deep neural networks with knowledge distillation," *IEEE International Conference on Data Mining Workshops, ICDMW*, vol. 2018-Novem, pp. 905–912, 2019, ISSN: 23759259. DOI: 10.1109/ICDMW.2018.00132. arXiv: 1812.10924.

[74] Z. Che, S. Purushotham, R. Khemani, and Y. Liu, "Interpretable Deep Models for ICU Outcome Prediction," *AMIA ... Annual Symposium proceedings. AMIA Symposium*, vol. 2016, pp. 371–380, 2016, ISSN: 1942597X.

[75] S. Krishnan and E. Wu, "PALM: Machine Learning Explanations for Iterative Debugging," *Notes and Queries*, vol. s3-I, no. 12, pp. 230–231, 1862, ISSN: 00293970. DOI: 10.1093/nq/s3-I.12.230-g.

[76] J. J. Thiagarajan, B. Kailkhura, P. Sattigeri, and K. N. Ramamurthy, "TreeView: Peeking into Deep Neural Networks Via Feature-Space Partitioning," no. Nips, 2016. arXiv: 1611.07429. [Online]. Available: http://arxiv.org/abs/1611.07429.

[77] O. Bastani, C. Kim, and H. Bastani, "Interpretability via model extraction," *arXiv*, 2017, ISSN: 23318422. arXiv: 1706.09773.

[78] Q. Zhang, Y. Yang, H. Ma, and Y. N. Wu, "Interpreting cnns via decision trees," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2019-June, pp. 6254–6263, 2019, ISSN: 10636919. DOI: 10.1109/CVPR.2019.00642. arXiv: 1802.00121.

[79] J. Chatterjee and N. Dethlefs, "Deep learning with knowledge transfer for explainable anomaly prediction in wind turbines," *Wind Energy*, vol. 23, no. 8, pp. 1693–1710, 2020, ISSN: 10991824. DOI: 10.1002/we.2510.

[80] O. Bastani, C. Kim, and H. Bastani, "Interpreting Blackbox Models via Model Extraction," 2017. arXiv: 1705.08504. [Online]. Available: http://arxiv.org/abs/1705.08504.

[81] N. Frosst and G. rey Hinton, "Distilling a neural network into a soft decision tree," *CEUR Workshop Proceedings*, vol. 2071, 2018, ISSN: 16130073. arXiv: 1711.09784.

[82] C. Yang, A. Rangarajan, and S. Ranka, "Global Model Interpretation Via Recursive Partitioning," *Proceedings - 20th International Conference on High Performance Computing and Communications, 16th International Conference on Smart City and 4th International Conference on Data Science and Systems, HPCC/SmartCity/DSS 2018*, pp. 1563–1570, 2019. DOI: 10.1109/HPCC/SmartCity/DSS.2018.00256. arXiv: 1802.04253.

[83] G. Erion, J. D. Janizek, P. Sturmfels, S. Lundberg, and S.-I. Lee, "Learning Explainable Models Using Attribution Priors," 2019. arXiv: 1906.10670. [Online]. Available: http://arxiv.org/abs/1906.10670.

[84] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," *2nd International Conference on Learning Representations, ICLR 2014 - Workshop Track Proceedings*, pp. 1–8, 2014. arXiv: 1312.6034.

[85] D. Baehrens, T. Schroeter, S. Harmeling, M. Kawanabe, K. Hansen, and K. R. Müller, "How to explain individual classification decisions," *Journal of Machine Learning Research*, vol. 11, pp. 1803–1831, 2010, ISSN: 15324435. arXiv: 0912.1128.

[86] J. Li, X. Chen, E. Hovy, and D. Jurafsky, "Visualizing and understanding neural models in NLP," *2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL HLT 2016 - Proceedings of the Conference*, pp. 681–691, 2016. DOI: 10.18653/v1/n16-1082. arXiv: 1506.01066.

[87] H. Yuan, Y. Chen, X. Hu, and S. Ji, "Interpreting deep models for text analysis via optimization and regularization methods," *33rd AAAI Conference on Artificial Intelligence, AAAI 2019, 31st Innovative Applications of Artificial Intelligence Conference, IAAI 2019 and the 9th AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019*, pp. 5717–5724,

2019, ISSN: 2159-5399. DOI: 10.1609/aaai.v33i01.33015717.

[88] F. Baldassarre and H. Azizpour, "Explainability Techniques for Graph Convolutional Networks," *arXiv*, 2019, ISSN: 23318422. arXiv: 1905.13686.

[89] A. Vilamala, K. H. Madsen, and L. K. Hansen, "Deep convolutional neural networks for interpretable analysis of EEG sleep stage scoring," *IEEE International Workshop on Machine Learning for Signal Processing, MLSP*, vol. 2017-Septe, no. October, pp. 1–6, 2017, ISSN: 21610371. DOI: 10.1109/MLSP.2017.8168133. arXiv: 1710.00633.

[90] K. Leino, S. Sen, A. Datta, M. Fredrikson, and L. Li, "Influence-Directed Explanations for Deep Convolutional Networks," *Proceedings - International Test Conference*, vol. 2018-Octob, no. 1, pp. 1–8, 2019, ISSN: 10893539. DOI: 10.1109/TEST.2018.8624792. arXiv: 1802.03788.

[91] P. Cortez and M. J. Embrechts, "Using sensitivity analysis and visualization techniques to open black box data mining models," *Information Sciences*, vol. 225, pp. 1–17, 2013, ISSN: 00200255. DOI: 10.1016/j.ins.2012.10.039. [Online]. Available: http://dx.doi.org/10.1016/j.ins.2012.10.039.

[92] P. Cortez and M. J. Embrechts, "Opening black box Data Mining models using Sensitivity Analysis," *IEEE Symposium on Computational Intelligence and Data Mining*, pp. 341–348, 2011. DOI: 10.1109/CIDM.2011.5949423.

[93] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, "Striving for simplicity: The all convolutional net," *3rd International Conference on Learning Representations, ICLR 2015 - Workshop Track Proceedings*, pp. 1–14, 2015. arXiv: 1412.6806.

[94] A. Shrikumar, P. Greenside, and A. Kundaje, "Not just a black box: Learning important features through propagating activation differences," *34th International Conference on Machine Learning, ICML 2017*, vol. 7, pp. 4844–4866, 2017. arXiv: 1704.02685.

[95] M. Ancona, E. Ceolini, C. Öztireli, and M. Gross, "Towards better understanding of gradient-based attribution methods for deep neural networks," *arXiv*, pp. 1–16, 2017, ISSN: 23318422. arXiv: 1711.06104.

[96] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic attribution for deep networks," *34th International Conference on Machine Learning, ICML 2017*, vol. 7, pp. 5109–5118, 2017. arXiv: 1703.01365.

[97] A. Kapishnikov, T. Bolukbasi, F. Viegas, and M. Terry, "XRAI: Better attributions through regions," *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2019-Octob, pp. 4947–4956, 2019, ISSN: 15505499. DOI: 10.1109/ICCV.2019.00505. arXiv: 1906.02825.

[98] S. Xu, S. Venugopalan, and M. Sundararajan, "Attribution in scale and space," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 9677–9686, 2020, ISSN: 10636919. DOI: 10.1109/CVPR42600.2020.00970. arXiv: 2004.03383.

[99] J. D. Janizek, P. Sturmfels, and S. I. Lee, "Explaining explanations: Axiomatic feature interactions for deep networks," *Journal of Machine Learning Research*, vol. 22, 2021, ISSN: 15337928. arXiv: 2002.04138.

[100] S. Srinivas and F. Fleuret, "Full-gradient representation for neural network visualization," *Advances in Neural Information Processing Systems*, vol. 32, no. NeurIPS, pp. 1–10, 2019, ISSN: 10495258. arXiv: 1905.00780.

[101] D. Smilkov, N. Thorat, B. Kim, F. Viégas, and M. Wattenberg, "SmoothGrad: Removing noise by adding noise," *arXiv*, 2017, ISSN: 23318422. arXiv: 1706.03825.

[102] M. Sundararajan and A. Taly, "Local explanation methods for deep neural networks lack sensitivity to parameter values," *arXiv*, pp. 1–10, 2018, ISSN: 23318422. arXiv: 1806.04205.

[103] G. Vilone and L. Longo, "Explainable Artificial Intelligence: a Systematic Review," *arXiv*, no. Dl, 2020, ISSN: 23318422. arXiv: 2006.00093.

[104] D. Kumar, A. Wong, and G. W. Taylor, "Explaining the Unexplained: A CLass-Enhanced Attentive Response (CLEAR) Approach to Understanding Deep Neural Networks," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, vol. 2017-July, pp. 1686–1694, 2017, ISSN: 21607516. DOI: 10.1109/CVPRW.2017.215. arXiv: 1704.04133.

[105] A. Mahendran and A. Vedaldi, "Salient Deconvolutional Networks," in *European Conference on Computer Vision*, 2016, ISBN: 9783319464749. DOI: 10.1007/978-3-319-46466-4.

[106] A. Mahendran and A. Vedaldi, "Understanding deep image representations by inverting them," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 07-12-June, pp. 5188–5196, 2015, ISSN: 10636919. DOI: 10.1109/CVPR.2015.7299155. arXiv: 1412.0035.

[107] A. Mahendran and A. Vedaldi, "Visualizing Deep Convolutional Neural Networks Using Natural Preimages," *International Journal of Computer Vision*, vol. 120, no. 3, pp. 233–255, 2016, ISSN: 15731405. DOI: 10.1007/s11263-016-0911-8. arXiv: 1512.02017.

[108] A. Dosovitskiy and T. Brox, "Inverting visual representations with convolutional networks," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2016-Decem, pp. 4829–4837, 2016, ISSN: 10636919. DOI: 10.1109/CVPR.2016.522. arXiv: 1506.02753.

[109] A. Dosovitskiy, J. T. Springenberg, M. Tatarchenko, and T. Brox, "Learning to Generate Chairs, Tables and Cars with Convolutional Networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 4, pp. 692–705, 2017, ISSN: 01628828. DOI: 10.1109/TPAMI.2016.2567384. arXiv: 1411.5928.

[110] M. Du, N. Liu, Q. Song, and X. Hu, "Towards explanation of DNN-based prediction with guided feature

inversion," *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1358–1367, 2018. DOI: 10.1145/3219819.3220099. arXiv: 1804.00506.

[111] M. D. Zeiler, D. Krishnan, G. W. Taylor, and R. Fergus, "Deconvolutional Networks," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2010. DOI: 10.1016/j.ins.2020.01.028.

[112] M. D. Zeiler, G. W. Taylor, and R. Fergus, "Adaptive deconvolutional networks for mid and high level feature learning," *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2018–2025, 2011. DOI: 10.1109/ICCV.2011.6126474.

[113] M. D. Zeiler and F. Rob, "Visualizing and Understanding Convolutional Networks," in *European Conference on Computer Vision*, 2014, pp. 818–833. DOI: 10.1016/j.ancr.2017.02.001.

[114] M. José Oramas, K. Wang, and T. Tuytelaars, "Visual explanation by interpretation: Improving visual feedback capabilities of deep neural networks," *arXiv*, no. 2015, 2017, ISSN: 23318422. arXiv: 1712.06302.

[115] F. Grün, C. Rupprecht, N. Navab, and F. Tombari, "A Taxonomy and Library for Visualizing Learned Features in Convolutional Neural Networks," vol. 48, 2016. arXiv: 1606.07757. [Online]. Available: http://arxiv.org/abs/1606.07757.

[116] M. Ancona, E. Ceolini, C. Öztireli, and M. Gross, "A unified view of gradient-based attribution methods for Deep Neural Networks," *arXiv*, no. Section 3, 2017, ISSN: 23318422. arXiv: 1711.06104.

[117] R. C. Fong and A. Vedaldi, "Interpretable Explanations of Black Boxes by Meaningful Perturbation," *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2017-Octob, pp. 3449–3457, 2017, ISSN: 15505499. DOI: 10.1109/ICCV.2017.371. arXiv: 1704.03296.

[118] J. Li, W. Monroe, and D. Jurafsky, "Understanding Neural Networks through Representation Erasure," 2016. arXiv: 1612.08220. [Online]. Available: http://arxiv.org/abs/1612.08220.

[119] M. Robnik-Šikonja and I. Kononenko, "Explaining classifications for individual instances," *IEEE Transactions on Knowledge and Data Engineering*, vol. 20, no. 5, pp. 589–600, 2008, ISSN: 10414347. DOI: 10.1109/TKDE.2007.190734.

[120] E. Štrumbelj, I. Kononenko, and M. Robnik Šikonja, "Explaining instance classifications with interactions of subsets of feature values," *Data and Knowledge Engineering*, vol. 68, no. 10, pp. 886–904, 2009, ISSN: 0169023X. DOI: 10.1016/j.datak.2009.01.004. [Online]. Available: http://dx.doi.org/10.1016/j.datak.2009.01.004.

[121] L. M. Zintgraf, T. S. Cohen, and M. Welling, "A New Method to Visualize Deep Neural Networks," 2016. arXiv: 1603.02518. [Online]. Available: http://arxiv.org/abs/1603.02518.

[122] L. M. Zintgraf, T. S. Cohen, T. Adel, and M. Welling, "Visualizing deep neural network decisions: Prediction difference analysis," *arXiv*, pp. 1–12, 2017, ISSN: 23318422. arXiv: 1702.04595.

[123] D. Seo, K. Oh, and I. S. Oh, "Regional multi-scale approach for visually pleasing explanations of deep neural networks," *IEEE Access*, vol. 8, no. Nips, pp. 8572–8582, 2020, ISSN: 21693536. DOI: 10.1109/ACCESS.2019.2963055. arXiv: 1807.11720.

[124] X. Jin, Z. Wei, J. Du, X. Xue, and X. Ren, "Towards Hierarchical Importance Attribution: Explaining Compositional Semantics for Neural Sequence Models," no. Cd, pp. 1–15, 2019. arXiv: 1911.06194. [Online]. Available: http://arxiv.org/abs/1911.06194.

[125] R. Fong, M. Patrick, and A. Vedaldi, "Understanding deep networks via extremal perturbations and smooth masks," *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2019-Octob, pp. 2950–2958, 2019, ISSN: 15505499. DOI: 10.1109/ICCV.2019.00304. arXiv: 1910.08485.

[126] P. Dabkowski and Y. Gal, "Real time image saliency for black box classifiers," *Advances in Neural Information Processing Systems*, vol. 2017-Decem, no. Nips, pp. 6968–6977, 2017, ISSN: 10495258. arXiv: 1705.07857.

[127] A. Altmann, L. Toloşi, O. Sander, and T. Lengauer, "Permutation importance: A corrected feature importance measure," *Bioinformatics*, vol. 26, no. 10, pp. 1340–1347, 2010, ISSN: 13674803. DOI: 10.1093/bioinformatics/btq134.

[128] V. Petsiuk, A. Das, and K. Saenko, "RISE: Randomized input sampling for explanation of black-box models," *29th British Machine Vision Conference, BMVC 2018*, 2018, ISSN: 23318422. arXiv: 1806.07421.

[129] S. Sattarzadeh, M. Sudhakar, A. Lem, *et al.*, "Explaining convolutional neural networks through attribution-based input sampling and block-wise feature aggregation," *arXiv*, 2020, ISSN: 23318422. arXiv: 2010.00672.

[130] D. Alvarez-Melis and T. S. Jaakkola, "A causal framework for explaining the predictions of black-box sequence-to-sequence models," *EMNLP 2017 - Conference on Empirical Methods in Natural Language Processing, Proceedings*, pp. 412–421, 2017. DOI: 10.18653/v1/d17-1042. arXiv: 1707.01943.

[131] A. Datta, S. Sen, and Y. Zick, "Algorithmic Transparency via Quantitative Input Influence," pp. 71–94, 2017. DOI: 10.1007/978-3-319-54024-5_4.

[132] A. B. Arrieta, S. Gil-Lopez, I. Laña, M. N. Bilbao, and J. Del Ser, "On the Post-hoc Explainability of Deep Echo State Networks for Time Series Forecasting, Image and Video Classification," 2021. arXiv: 2102.08634. [Online]. Available: http://arxiv.org/abs/2102.08634.

[133] R. Ying, D. Bourgeois, J. You, M. Zitnik, and J. Leskovec, "GNNExplainer: Generating explanations for graph neural networks," *arXiv*, no. iii, 2019, ISSN: 23318422. arXiv: 1903.03894.

[134] D. Luo, W. Cheng, D. Xu, *et al.*, "Parameterized Explainer for Graph Neural Network," no. NeurIPS,

pp. 1–17, 2020. arXiv: 2011.04573. [Online]. Available: http://arxiv.org/abs/2011.04573.

[135] M. S. Schlichtkrull, N. de Cao, and I. Titov, "Interpreting graph neural networks for NLP with differentiable edge masking," *arXiv*, no. January, 2020, ISSN: 23318422. arXiv: 2010.00577.

[136] A. Dhurandhar, P. Y. Chen, R. Luss, *et al.*, "Explanations based on the Missing: Towards contrastive explanations with pertinent negatives," *Advances in Neural Information Processing Systems*, vol. 2018-Decem, pp. 592–603, 2018, ISSN: 10495258. arXiv: 1802.07623.

[137] S. M. Muddamsetty, N. S. Mohammad, and T. B. Moeslund, "SIDU: Similarity Difference and Uniqueness Method for Explainable AI," *Proceedings - International Conference on Image Processing, ICIP*, vol. 2020-Octob, pp. 3269–3273, 2020, ISSN: 15224880. DOI: 10.1109/ICIP40778.2020.9190952. arXiv: 2006.03122.

[138] C. Burns, J. Thomason, and W. Tansey, "Interpreting Black Box Models via Hypothesis Testing," *FODS 2020 - Proceedings of the 2020 ACM-IMS Foundations of Data Science Conference*, pp. 47–57, 2020. DOI: 10.1145/3412815.3416889. arXiv: 1904.00045.

[139] C. H. Chang, E. Creager, A. Goldenberg, and D. Duvenaud, "Explaining image classifiers by counterfactual generation," *7th International Conference on Learning Representations, ICLR 2019*, no. 2018, pp. 1–19, 2019. arXiv: 1807.08024.

[140] C. Agarwal and A. Nguyen, "Explaining Image Classifiers by Removing Input Features Using Generative Models," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 12627 LNCS, no. 1, pp. 101–118, 2021, ISSN: 16113349. DOI: 10.1007/978-3-030-69544-6_7. arXiv: 1910.04256.

[141] J. Wagner, J. M. Kohler, and T. Gindele, "Interpretable and Fine-Grained Visual Explanations for Convolutional Neural Networks," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, ISBN: 9789811336478. DOI: 10.1007/978-981-13-3648-5_33.

[142] H. Yuan, L. Cai, X. Hu, J. Wang, and S. Ji, "Interpreting Image Classifiers by Generating Discrete Masks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8828, no. c, pp. 1–1, 2020, ISSN: 0162-8828. DOI: 10.1109/tpami.2020.3028783.

[143] P. Schwab and W. Karlen, "CXPlain: Causal explanations for model interpretation under uncertainty," *Advances in Neural Information Processing Systems*, vol. 32, no. NeurIPS, 2019, ISSN: 10495258. arXiv: 1910.12336.

[144] C. Guan, X. Wang, Q. Zhang, R. Chen, D. He, and X. Xie, "Towards a deep and unified understanding of deep neural models in NLP," *36th International Conference on Machine Learning, ICML 2019*, vol. 2019-June, pp. 4366–4375, 2019.

[145] H. Yuan, H. Yu, S. Gui, and S. Ji, "Explainability in Graph Neural Networks: A Taxonomic Survey," pp. 1–14, 2020. arXiv: 2012.15445. [Online]. Available: http://arxiv.org/abs/2012.15445.

[146] P. J. Kindermans, K. T. Schütt, M. Alber, *et al.*, "Learning how to explain neural networks: Patternnet and Patternattribution," *6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings*, pp. 1–12, 2018. arXiv: 1705.05598.

[147] G. Montavon, A. Binder, S. Lapuschkin, W. Samek, and K. R. Müller, "Layer-Wise Relevance Propagation: An Overview," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11700 LNCS, pp. 193–209, 2019, ISSN: 16113349. DOI: 10.1007/978-3-030-28954-6_10.

[148] J. Gu, Y. Yang, and V. Tresp, "Understanding Individual Decisions of CNNs via Contrastive Backpropagation," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11363 LNCS, pp. 119–134, 2019, ISSN: 16113349. DOI: 10.1007/978-3-030-20893-6_8. arXiv: 1812.02100.

[149] W. J. Nam, S. Gur, J. Choi, L. Wolf, and S. W. Lee, "Relative attributing propagation: Interpreting the comparative contributions of individual units in deep neural networks," *AAAI 2020 - 34th AAAI Conference on Artificial Intelligence*, pp. 2501–2508, 2020, ISSN: 2159-5399. DOI: 10.1609/aaai.v34i03.5632. arXiv: 1904.00605.

[150] W.-J. Nam, J. Choi, and S.-W. Lee, "Interpreting Deep Neural Networks with Relative Sectional Propagation by Analyzing Comparative Gradients and Hostile Activations," 2020. arXiv: 2012.03434. [Online]. Available: http://arxiv.org/abs/2012.03434.

[151] A. Shrikumar, P. Greenside, and A. Kundaje, "Learning important features through propagating activation differences," *34th International Conference on Machine Learning, ICML 2017*, vol. 7, pp. 4844–4866, 2017. arXiv: 1704.02685.

[152] S. Bach, A. Binder, G. Montavon, F. Klauschen, K. R. Müller, and W. Samek, "On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation," *PLoS ONE*, vol. 10, no. 7, pp. 1–46, 2015, ISSN: 19326203. DOI: 10.1371/journal.pone.0130140.

[153] G. Montavon, W. Samek, and K. R. Müller, "Methods for interpreting and understanding deep neural networks," *Digital Signal Processing: A Review Journal*, vol. 73, pp. 1–15, 2018, ISSN: 10512004. DOI: 10.1016/j.dsp.2017.10.011. arXiv: 1706.07979. [Online]. Available: https://doi.org/10.1016/j.dsp.2017.10.011.

[154] A. Binder, G. Montavon, S. Lapuschkin, K. R. Müller, and W. Samek, "Layer-wise relevance propagation for neural networks with local renormalization layers," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and*

Lecture Notes in Bioinformatics), vol. 9887 LNCS, pp. 63–71, 2016, ISSN: 16113349. DOI: 10.1007/978-3-319-44781-0_8. arXiv: 1604.00825.

[155] L. Arras, G. Montavon, K.-R. Müller, and W. Samek, "Explaining Recurrent Neural Network Predictions in Sentiment Analysis," pp. 159–168, 2018. DOI: 10.18653/v1/w17-5221. arXiv: 1706.07206.

[156] L. Arras, J. Arjona-Medina, M. Widrich, *et al.*, "Explaining and Interpreting LSTMs," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11700 LNCS, no. 2019, pp. 211–238, 2019, ISSN: 16113349. DOI: 10.1007/978-3-030-28954-6_11. arXiv: 1909.12114.

[157] 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, ISSN: 00313203. DOI: 10.1016/j.patcog.2016.11.008. arXiv: 1512.02479. [Online]. Available: http://dx.doi.org/10.1016/j.patcog.2016.11.008.

[158] J. Kauffmann, K. R. Müller, and G. Montavon, "Towards explaining anomalies: A deep Taylor decomposition of one-class models," *Pattern Recognition*, vol. 101, 2020, ISSN: 00313203. DOI: 10.1016/j.patcog.2020.107198. arXiv: 1805.06230.

[159] G. Montavon, S. Bach, A. Binder, W. Samek, and K.-R. Müller, "Deep Taylor Decomposition of Neural Networks," *ICML'16 Workshop on Visualization for Deep Learning*, pp. 1–3, 2016.

[160] W. Landecker, M. D. Thomure, L. M. Bettencourt, M. Mitchell, G. T. Kenyon, and S. P. Brumby, "Interpreting individual classifications of hierarchical networks," *Proceedings of the 2013 IEEE Symposium on Computational Intelligence and Data Mining, CIDM 2013 - 2013 IEEE Symposium Series on Computational Intelligence, SSCI 2013*, pp. 32–38, 2013. DOI: 10.1109/CIDM.2013.6597214.

[161] S. Lapuschkin, S. Wäldchen, A. Binder, G. Montavon, W. Samek, and K. R. Müller, "Unmasking Clever Hans predictors and assessing what machines really learn," *Nature Communications*, vol. 10, no. 1, 2019, ISSN: 20411723. DOI: 10.1038/s41467-019-08987-4. arXiv: 1902.10178.

[162] J. Zhang, S. A. Bargal, Z. Lin, J. Brandt, X. Shen, and S. Sclaroff, "Top-Down Neural Attention by Excitation Backprop," *International Journal of Computer Vision*, vol. 126, no. 10, pp. 1084–1102, 2018, ISSN: 15731405. DOI: 10.1007/s11263-017-1059-x. arXiv: 1608.00507.

[163] P. E. Pope, S. Kolouri, M. Rostami, C. E. Martin, and H. Hoffmann, "Explainability methods for graph convolutional neural networks," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2019-June, pp. 10 764–10 773, 2019, ISSN: 10636919. DOI: 10.1109/CVPR.2019.01103.

[164] R. Caruana, H. Kangarloo, J. D. Dionisio, U. Sinha, and D. Johnson, "Case-based explanation of non-case-based learning methods.," *Proceedings / AMIA ... Annual Symposium. AMIA Symposium*, pp. 212–215, 1999, ISSN: 1531605X.

[165] S. O. Arik and T. Pfister, "Protoattend: Attention-based prototypical learning," *Journal of Machine Learning Research*, vol. 21, no. 2, pp. 1–23, 2020, ISSN: 15337928. arXiv: 1902.06292.

[166] M. Nauta, A. Jutte, J. Provoost, and C. Seifert, *This Looks Like That, Because ... Explaining Prototypes for Interpretable Image Recognition*. Association for Computing Machinery, 2020, vol. 1. arXiv: 2011.02863. [Online]. Available: http://arxiv.org/abs/2011.02863.

[167] P. Wei Koh and P. Liang, "Understanding Black-box Predictions via Influence Functions," in *Proceedings of the 34th International Conference on Machine Learning*, vol. 70, 2017, pp. 1885–1894.

[168] C. K. Yeh, J. S. Kim, I. E. Yen, and P. Ravikumar, "Representer point selection for explaining deep neural networks," *Advances in Neural Information Processing Systems*, vol. 2018-Decem, no. Nips, pp. 9291–9301, 2018, ISSN: 10495258. arXiv: 1811.09720.

[169] P. S. Haghighi, O. Seton, and O. Nasraoui, "An Explainable Autoencoder For Collaborative Filtering Recommendation," 2019. arXiv: 2001.04344. [Online]. Available: http://arxiv.org/abs/2001.04344.

[170] J. Bien and R. Tibshirani, "PROTOTYPE SELECTION FOR INTERPRETABLE CLASSIFICATION," *The annuals of applied statistics*, vol. 5, no. 4, pp. 2403–2424, 2011. DOI: 10.1214/11.

[171] B. Kim, R. Khanna, and O. Koyejo, "Examples are not enough, learn to criticize! Criticism for interpretability," *Advances in Neural Information Processing Systems*, no. Nips, pp. 2288–2296, 2016, ISSN: 10495258.

[172] K. S. Gurumoorthy, A. Dhurandhar, G. Cecchi, and C. Aggarwal, "Efficient data representation by selecting prototypes with importance weights," *Proceedings - IEEE International Conference on Data Mining, ICDM*, vol. 2019-Novem, pp. 260–269, 2019, ISSN: 15504786. DOI: 10.1109/ICDM.2019.00036. arXiv: 1707.01212.

[173] O. Li, H. Liu, C. Chen, and C. Rudin, "Deep learning for case-based reasoning through prototypes: A neural network that explains its predictions," *32nd AAAI Conference on Artificial Intelligence, AAAI 2018*, pp. 3530–3537, 2018. arXiv: 1710.04806.

[174] C. Chen, O. Li, C. Tao, A. J. Barnett, J. Su, and C. Rudin, "This looks like that: Deep learning for interpretable image recognition," *Advances in Neural Information Processing Systems*, vol. 32, no. NeurIPS, pp. 1–12, 2019, ISSN: 10495258. arXiv: 1806.10574.

[175] P. Angelov and E. Soares, "Towards explainable deep neural networks (xDNN)," *Neural Networks*, vol. 130, pp. 185–194, 2020, ISSN: 18792782. DOI: 10.1016/j.neunet.2020.07.010. arXiv: 1912.02523. [Online].

Available: https://doi.org/10.1016/j.neunet.2020.07.010.

[176] S. Wachter, B. Mittelstadt, and C. Russell, "Counterfactual Explanations Without Opening the Black Box: Automated Decisions and the GDPR," *SSRN Electronic Journal*, pp. 1–52, 2017, ISSN: 1556-5068. DOI: 10.2139/ssrn.3063289. arXiv: 1711.00399.

[177] R. Confalonieri, L. Coba, B. Wagner, and T. R. Besold, "A historical perspective of explainable Artificial Intelligence," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 11, no. 1, pp. 1–21, 2021, ISSN: 19424795. DOI: 10.1002/widm.1391.

[178] Y. Goyal, Z. Wu, J. Ernst, D. Batra, D. Parikh, and S. Lee, "Counterfactual visual explanations," *36th International Conference on Machine Learning, ICML 2019*, vol. 2019-June, pp. 4254–4262, 2019. arXiv: 1904.07451.

[179] N. Madaan, I. Padhi, N. Panwar, and D. Saha, "Generate Your Counterfactuals: Towards Controlled Counterfactual Generation for Text," 2020. arXiv: 2012.04698. [Online]. Available: http://arxiv.org/abs/2012.04698.

[180] R. K. Mothilal, A. Sharma, and C. Tan, "Explaining machine learning classifiers through diverse counterfactual explanations," *FAT* 2020 - Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, pp. 607–617, 2020. DOI: 10.1145/3351095.3372850. arXiv: 1905.07697.

[181] S. Sharma, J. Henderson, and J. Ghosh, "CERTIFAI: Counterfactual Explanations for Robustness, Transparency, Interpretability, and Fairness of Artificial Intelligence models," 2019. DOI: 10.1145/3375627.3375812. arXiv: 1905.07857. [Online]. Available: http://arxiv.org/abs/1905.07857%0Ahttp://dx.doi.org/10.1145/3375627.3375812.

[182] Y. Ramon, D. Martens, F. Provost, and T. Evgeniou, "Counterfactual Explanation Algorithms for Behavioral and Textual Data," 2019. arXiv: 1912.01819. [Online]. Available: http://arxiv.org/abs/1912.01819.

[183] D. Nemirovsky, N. Thiebaut, Y. Xu, and A. Gupta, "CounteRGAN: Generating Realistic Counterfactuals with Residual Generative Adversarial Nets," 2020. arXiv: 2009.05199. [Online]. Available: http://arxiv.org/abs/2009.05199.

[184] E. Delaney, D. Greene, and M. T. Keane, "Instance-Based Counterfactual Explanations for Time Series Classification," 2020. arXiv: 2009.13211. [Online]. Available: http://arxiv.org/abs/2009.13211.

[185] A. Van Looveren and J. Klaise, "Interpretable Counterfactual Explanations Guided by Prototypes," 2019. arXiv: 1907.02584. [Online]. Available: http://arxiv.org/abs/1907.02584.

[186] N. Vercheval and A. Pizurica, "Hierarchical Variational Autoencoder for Visual Counterfactuals," pp. 1–5, 2021. arXiv: 2102.00854. [Online]. Available: http://arxiv.org/abs/2102.00854.

[187] M. T. Keane and B. Smyth, "Good Counterfactuals and Where to Find Them: A Case-Based Technique for Generating Counterfactuals for Explainable AI (XAI)," *28th International Conference on Case Based Reasoning (ICCBR2020)*, 2020. arXiv: 2007.05684. [Online]. Available: http://arxiv.org/abs/2007.05684.

[188] M. Downs, J. L. Chu, Y. Yacoby, F. Doshi-Velez, and W. Pan, "CRUDS: Counterfactual Recourse Using Disentangled Subspaces," no. Whi, 2020.

[189] A. Akula, S. Wang, and S.-C. Zhu, "CoCoX: Generating Conceptual and Counterfactual Explanations via Fault-Lines," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 03, pp. 2594–2601, 2020, ISSN: 2159-5399. DOI: 10.1609/aaai.v34i03.5643.

[190] K. Kanamori, T. Takagi, K. Kobayashi, and H. Arimura, "DACE: Distribution-aware counterfactual explanation by mixed-integer linear optimization," *IJCAI International Joint Conference on Artificial Intelligence*, vol. 2021-Janua, pp. 2855–2862, 2020, ISSN: 10450823. DOI: 10.24963/ijcai.2020/395.

[191] R. Poyiadzi, K. Sokol, R. Santos-Rodriguez, T. De Bie, and P. Flach, "FACE: Feasible and actionable counterfactual explanations," *AIES 2020 - Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, pp. 344–350, 2020. DOI: 10.1145/3375627.3375850. arXiv: 1909.09369.

[192] F. Cheng, Y. Ming, and H. Qu, "DECE: Decision Explorer with Counterfactual Explanations for Machine Learning Models," *IEEE Transactions on Visualization and Computer Graphics*, vol. 27, no. 2, pp. 1438–1447, 2021, ISSN: 19410506. DOI: 10.1109/TVCG.2020.3030342. arXiv: 2008.08353.

[193] O. Gomez, S. Holter, J. Yuan, and E. Bertini, "ViCE: Visual Counterfactual Explanations for Machine Learning Models," 2020. arXiv: 2003.02428. [Online]. Available: http://arxiv.org/abs/2003.02428.

[194] M. O'Shaughnessy, G. Canal, M. Connor, M. Davenport, and C. Rozell, "Generative causal explanations of black-box classifiers," vol. 1, no. NeurIPS, pp. 1–34, 2020. arXiv: 2006.13913. [Online]. Available: http://arxiv.org/abs/2006.13913.

[195] D. Mahajan, C. Tan, and A. Sharma, "Preserving Causal Constraints in Counterfactual Explanations for Machine Learning Classifiers," no. NeurIPS, 2019. arXiv: 1912.03277. [Online]. Available: http://arxiv.org/abs/1912.03277.

[196] A. Kanehira, K. Takemoto, S. Inayoshi, and T. Harada, "Multimodal explanations by predicting counterfactuality in videos," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2019-June, pp. 8586–8594, 2019, ISSN: 10636919. DOI: 10.1109/CVPR.2019.00879. arXiv: 1812.01263.

[197] K. Rawal and H. Lakkaraju, "Beyond Individualized Recourse: Interpretable and Interactive Summaries of Actionable Recourses," no. NeurIPS, 2020. arXiv: 2009.07165. [Online]. Available: http://arxiv.org/abs/2009.07165.

[198] K. Kanamori, T. Takagi, and K. Kobayashi, "Ordered Counterfactual Explanation by Mixed-Integer Linear Optimization," in *35th AAAI Conference on Artificial Intelligence (AAAI 2021)*, 2021. arXiv: arXiv:2012.11782v1.

[199] B. Pollack and J. Chen, "EXPLANATION BY PROGRESSIVE EXAGGERATION," in *ICLR International Conference on Learning Representations*, 2020, pp. 1–20.

[200] S.-H. Kang, H.-G. Jung, D.-O. Won, and S.-W. Lee, "Counterfactual Explanation Based on Gradual Construction for Deep Networks," pp. 1–26, 2020. arXiv: 2008.01897. [Online]. Available: http://arxiv.org/abs/2008.01897.

[201] A. Ghorbani, J. Wexler, J. Zou, and B. Kim, "Towards automatic concept-based explanations," *Advances in Neural Information Processing Systems*, vol. 32, 2019, ISSN: 10495258. arXiv: 1902.03129.

[202] B. Kim, J. Gilmer, F. Viegas, U. Erlingsson, and M. Wattenberg, "TCAV RELATIVE CONCEPT IMPORTANCE TESTING WITH LINEAR CONCEPT ACTIVATION VECTORS," 2017. arXiv: 1711.11279.

[203] B. Kim, M. Wattenberg, J. Gilmer, *et al.*, "Interpretability beyond feature attribution: Quantitative Testing with Concept Activation Vectors (TCAV)," *35th International Conference on Machine Learning, ICML 2018*, vol. 6, pp. 4186–4195, 2018. arXiv: 1711.11279.

[204] M. Graziani, V. Andrearczyk, and H. Müller, "Regression concept vectors for bidirectional explanations in histopathology," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11038 LNCS, pp. 124–132, 2018, ISSN: 16113349. DOI: 10.1007/978-3-030-02628-8_14. arXiv: 1904.04520.

[205] H. Yeche, J. Harrison, and T. Berthier, "UBS: A dimension-agnostic metric for concept vector interpretability applied to radiomics," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11797 LNCS, pp. 12–20, 2019, ISSN: 16113349. DOI: 10.1007/978-3-030-33850-3_2.

[206] Y. Goyal, A. Feder, U. Shalit, and B. Kim, "Explaining Classifiers with Causal Concept Effect (CaCE)," 2019. arXiv: 1907.07165. [Online]. Available: http://arxiv.org/abs/1907.07165.

[207] C.-K. Yeh, B. Kim, S. O. Arik, C.-L. Li, T. Pfister, and P. Ravikumar, "On Concept-Based Explanations in Deep Neural Networks," 2019. arXiv: 1910.07969. [Online]. Available: http://arxiv.org/abs/1910.07969.

[208] M. Godi, M. Carletti, M. Aghaei, F. Giuliari, and M. Cristani, "Understanding deep architectures by visual summaries," *British Machine Vision Conference 2018, BMVC 2018*, pp. 1–12, 2019. arXiv: 1801.09103.

[209] Y. Dong, H. Su, J. Zhu, and B. Zhang, "Improving interpretability of deep neural networks with semantic information," *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, vol. 2017-Janua, pp. 975–983, 2017. DOI: 10.1109/CVPR.2017.110. arXiv: 1703.04096.

[210] M. Ul Hassan, P. Mulhem, D. Pellerin, and G. Quenot, "Explaining Visual Classification using Attributes," *Proceedings - International Workshop on Content-Based Multimedia Indexing*, vol. 2019-Septe, pp. 1–6, 2019, ISSN: 19493991. DOI: 10.1109/CBMI.2019.8877393.

[211] L. A. Hendricks, Z. Akata, M. Rohrbach, J. Donahue, B. Schiele, and T. Darrell, "Generating visual explanations," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 9908 LNCS, pp. 3–19, 2016, ISSN: 16113349. DOI: 10.1007/978-3-319-46493-0_1. arXiv: 1603.08507.

[212] P. Guo, C. Anderson, K. Pearson, and R. Farrell, "Neural Network Interpretation via Fine Grained Textual Summarization," 2018. arXiv: 1805.08969. [Online]. Available: http://arxiv.org/abs/1805.08969.

[213] U. Ehsan, B. Harrison, L. Chan, and M. O. Riedl, "Rationalization: A Neural Machine Translation Approach to Generating Natural Language Explanations," *AIES 2018 - Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, pp. 81–87, 2018. DOI: 10.1145/3278721.3278736. arXiv: 1702.07826.

[214] U. Ehsan, P. Tambwekar, L. Chan, B. Harrison, and M. O. Riedl, "Automated rationale generation: A technique for explainable AI and its effects on human perceptions," *International Conference on Intelligent User Interfaces, Proceedings IUI*, vol. Part F1476, pp. 263–274, 2019. DOI: 10.1145/3301275.3302316. arXiv: 1901.03729.

[215] M. Hind, D. Wei, M. Campbell, *et al.*, "TED: Teaching AI to explain its decisions," *AIES 2019 - Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, pp. 123–129, 2019. DOI: 10.1145/3306618.3314273. arXiv: 1811.04896.

[216] H. Liu, Q. Yin, and W. Y. Wang, "Towards explainable NLP: A generative explanation framework for text classification," *ACL 2019 - 57th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, pp. 5570–5581, 2020. DOI: 10.18653/v1/p19-1560. arXiv: 1811.00196.

[217] S. Wickramanayake, W. Hsu, and M. L. Lee, "FLEX: Faithful linguistic explanations for neural net based model decisions," *33rd AAAI Conference on Artificial Intelligence, AAAI 2019, 31st Innovative Applications of Artificial Intelligence Conference, IAAI 2019 and the 9th AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019*, vol. 3, pp. 2539–2546, 2019, ISSN: 2159-5399. DOI: 10.1609/aaai.v33i01.33012539.

[218] W. Zhou, J. Hu, H. Zhang, *et al.*, "Towards Interpretable Natural Language Understanding with Explanations as Latent Variables," no. NeurIPS, pp. 1–16, 2020. arXiv: 2011.05268. [Online]. Available: http://arxiv.org/abs/2011.05268.

[219] S. T. Kim, H. Lee, H. G. Kim, and Y. M. Ro, "ICADx: interpretable computer aided diagnosis of breast masses," p. 73, 2018, ISSN: 16057422. DOI: 10.1117/12.2293570. arXiv: 1805.08960.

[220] S. Shen, S. X. Han, D. R. Aberle, A. A. Bui, and W. Hsu, "An interpretable deep hierarchical semantic convolutional neural network for lung nodule malignancy classification," *Expert Systems with Applications*, vol. 128, pp. 84–95, 2019, ISSN: 09574174. DOI: 10.1016/j.eswa.2019.01.048. arXiv: 1806.00712. [Online]. Available: https://doi.org/10.1016/j.eswa.2019.01.048.

[221] S. Gulshad and A. Smeulders, "Explaining with counter visual attributes and examples," *ICMR 2020 - Proceedings of the 2020 International Conference on Multimedia Retrieval*, pp. 35–43, 2020. DOI: 10.1145/3372278.3390672. arXiv: 2001.09671.

[222] S. Barratt, "InterpNET: Neural Introspection for Interpretable Deep Learning," no. Nips, 2017. arXiv: 1710.09511. [Online]. Available: http://arxiv.org/abs/1710.09511.

[223] A. Kanehira and T. Harada, "Learning to explain with complemental examples," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2019-June, pp. 8595–8603, 2019, ISSN: 10636919. DOI: 10.1109/CVPR.2019.00880. arXiv: 1812.01280.

[224] A. Nguyen, A. Dosovitskiy, J. Yosinski, T. Brox, and J. Clune, "Synthesizing the preferred inputs for neurons in neural networks via deep generator networks," *Advances in Neural Information Processing Systems*, no. Nips, pp. 3395–3403, 2016, ISSN: 10495258. arXiv: 1605.09304.

[225] A. Nguyen, J. Yosinski, and J. Clune, "Multifaceted Feature Visualization: Uncovering the Different Types of Features Learned By Each Neuron in Deep Neural Networks," 2016. arXiv: 1602.03616. [Online]. Available: http://arxiv.org/abs/1602.03616.

[226] D. Erhan, Y. Bengio, A. Courville, and P. Vincent, "Visualizing higher-layer features of a deep network," *Bernoulli*, no. 1341, pp. 1–13, 2009. [Online]. Available: http://igva2012.wikispaces.asu.edu/file/view/Erhan+2009+Visualizing+higher+layer+features+of+a+deep+network.pdf.

[227] D. Erhan, A. Courville, and Y. Bengio, "Understanding Representations Learned in Deep Architectures," *Network*, no. October 2014, pp. 1–25, 2010, ISSN: 15324435. arXiv: arXiv:1206.5538v1.

[228] A. Nguyen, J. Yosinski, Y. Bengio, A. Dosovitskiy, and J. Clune, "Plug Play Generative Networks: Conditional Iterative Generation of Images in Latent Space," *Iccv*, no. 1, pp. 4467–4477, 2017. arXiv: 1738978.

[229] D. Bau, J. Y. Zhu, H. Strobelt, A. Lapedriza, B. Zhou, and A. Torralba, "Understanding the role of individual units in a deep neural network," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 117, no. 48, pp. 30 071–30 078, 2020, ISSN: 10916490. DOI: 10.1073/pnas.1907375117. arXiv: 2009.05041.

[230] Y. Ming, S. Cao, R. Zhang, *et al.*, "Understanding Hidden Memories of Recurrent Neural Networks," *2017 IEEE Conference on Visual Analytics Science and Technology, VAST 2017 - Proceedings*, pp. 13–24, 2018. DOI: 10.1109/VAST.2017.8585721. arXiv: 1710.10777.

[231] D. Bau, B. Zhou, A. Khosla, A. Oliva, and A. Torralba, "Network dissection: Quantifying interpretability of deep visual representations," *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, vol. 2017-Janua, pp. 3319–3327, 2017. DOI: 10.1109/CVPR.2017.354. arXiv: 1704.05796.

[232] B. Zhou, D. Bau, A. Oliva, and A. Torralba, "Interpreting Deep Visual Representations via Network Dissection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 9, pp. 2131–2145, 2019, ISSN: 19393539. DOI: 10.1109/TPAMI.2018.2858759. arXiv: 1711.05611.

[233] R. Fong and A. Vedaldi, "Net2Vec: Quantifying and Explaining How Concepts are Encoded by Filters in Deep Neural Networks," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 8730–8738, 2018, ISSN: 10636919. DOI: 10.1109/CVPR.2018.00910. arXiv: 1801.03454.

[234] D. Bau, J. Y. Zhu, H. Strobelt, *et al.*, "Gan Dissection: Visualizing and Understanding Generative Adversarial Networks," *arXiv*, 2018, ISSN: 23318422. arXiv: 1811.10597.

[235] R. Meyes, C. W. de Puiseau, A. Posada-Moreno, and T. Meisen, "Under the Hood of Neural Networks: Characterizing Learned Representations by Functional Neuron Populations and Network Ablations," 2020. arXiv: 2004.01254. [Online]. Available: http://arxiv.org/abs/2004.01254.

[236] M. Raghu, J. Gilmer, J. Yosinski, and J. Sohl-Dickstein, "SVCCA: Singular vector canonical correlation analysis for deep learning dynamics and interpretability," *Advances in Neural Information Processing Systems*, vol. 2017-Decem, no. Nips, pp. 6077–6086, 2017, ISSN: 10495258. arXiv: 1706.05806.

[237] H. Liang, Z. Ouyang, Y. Zeng, *et al.*, "Training Interpretable Convolutional Neural Networks by Differentiating Class-Specific Filters," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 12347 LNCS, pp. 622–638, 2020, ISSN: 16113349. DOI: 10.1007/978-3-030-58536-5_37. arXiv: 2007.08194.

[238] R. Shwartz-Ziv and N. Tishby, "Opening the Black Box of Deep Neural Networks via Information," pp. 1–19, 2017. arXiv: 1703.00810. [Online]. Available: http://arxiv.org/abs/1703.00810.

[239] N. Tishby, F. C. Pereira, and W. Bialek, "The information bottleneck method," pp. 1–16, 2000. arXiv:

0004057. [Online]. Available: http://arxiv.org/abs/physics/0004057.

[240] N. Tishby and N. Zaslavsky, "Deep learning and the information bottleneck principle," *2015 IEEE Information Theory Workshop, ITW 2015*, 2015. DOI: 10.1109/ITW.2015.7133169. arXiv: 1503.02406.

[241] A. A. Alemi, I. Fischer, J. V. Dillon, and K. Murphy, "Deep variational information bottleneck," *5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings*, pp. 1–19, 2017. arXiv: 1612.00410.

[242] A. Kolchinsky, B. D. Tracey, and D. H. Wolpert, "Nonlinear information bottleneck," *Entropy*, vol. 21, no. 12, pp. 1–15, 2019, ISSN: 10994300. DOI: 10.3390/e21121181. arXiv: 1705.02436.

[243] I. Fischer, "The conditional entropy bottleneck," *Entropy*, vol. 22, no. 9, 2020, ISSN: 10994300. DOI: 10.3390/e22090999. arXiv: 2002.05379.

[244] J. Li and D. Liu, "Information Bottleneck Theory on Convolutional Neural Networks," *Neural Processing Letters*, vol. 53, no. 2, pp. 1385–1400, 2021, ISSN: 1573773X. DOI: 10.1007/s11063-021-10445-6. arXiv: 1911.03722.