DOI: 10.1111/cgf.14733

COMPUTER GRAPHICS *forum*
Volume 42 (2023), number 1 pp. 319–355

State of the Art of Visual Analytics for eXplainable Deep Learning

B. La Rosa,[1] G. Blasilli,[1] R. Bourqui,[2] D. Auber,[2] G. Santucci,[1] R. Capobianco,[1,3] E. Bertini,[4] R. Giot[2] and M. Angelini[1]

[1]Sapienza Università di Roma, Rome, Italy
[2]CNRS, Bordeaux INP, LaBRI, UMR5800, University of Bordeaux, Talence, France
[3]Sony AI, Zurich, Switzerland
[4]Northeastern University, Boston, Massachusetts, USA

Abstract

The use and creation of machine-learning-based solutions to solve problems or reduce their computational costs are becoming increasingly widespread in many domains. Deep Learning plays a large part in this growth. However, it has drawbacks such as a lack of explainability and behaving as a black-box model. During the last few years, Visual Analytics has provided several proposals to cope with these drawbacks, supporting the emerging eXplainable Deep Learning field. This survey aims to (i) systematically report the contributions of Visual Analytics for eXplainable Deep Learning; (ii) spot gaps and challenges; (iii) serve as an anthology of visual analytical solutions ready to be exploited and put into operation by the Deep Learning community (architects, trainers and end users) and (iv) prove the degree of maturity, ease of integration and results for specific domains. The survey concludes by identifying future research challenges and bridging activities that are helpful to strengthen the role of Visual Analytics as effective support for eXplainable Deep Learning and to foster the adoption of Visual Analytics solutions in the eXplainable Deep Learning community. An interactive explorable version of this survey is available online at https://aware-diag-sapienza.github.io/VA4XDL.

Keywords: deep learning, explainable artificial intelligence, interpretability, neural networks, visual analytics, visualization

CCS Concepts: • General and reference → Surveys and overviews; • Human-centred computing → Visual analytics; • Computing methodologies → Neural networks; Artificial intelligence

1. Introduction

Ranging from health care [APA*16] and cybersecurity [XKL*18] to self-autonomous vehicles [HY22] and natural language processing [Gol17], Machine Learning (ML) approaches for automatically solving tasks and domain problems are becoming increasingly widespread. Among them, Deep Learning (DL) [LBH15] techniques correspond to a family of state-of-the-art ML methods that handle large amounts of data thanks to Neural Networks (NNs) composed of several stacked layers of computation and thousands of neurons.

Despite its great success, DL suffers from a significant issue: the complexity of these networks makes it difficult to understand how they make decisions and why they fail. In the last few years, this problem has led to the rise of eXplainable Deep Learning (XDL) [RXGD22], a sub-field of eXplainable Artificial Intelligence (XAI) [GSC*19] which aims at ML as a whole. At the same time, Visual Analytics (VA) [HKPC19] solutions designed to support explainability and interpretability for DL have been developed to meet the needs of various stakeholders. These contributions help identify visualization and VA as well-suited disciplines to support researchers, developers and users of DL solutions.

However, the current level of maturity of the integration of the proposed VA solutions and the XDL approaches is unclear. More specifically, we do not yet know when VA is a desirable solution for a given application domain, which solutions are pre-dominant and why, and which XDL solutions the literature adopts. We propose this survey on VA for XDL to answer these questions.

Objectives of the survey. This paper presents a timely survey and an analysis of the existing works that advance the capability of VA solutions to improve the understanding of DL models. Our goal is threefold: (i) collect and organize the design choices, explanations and solutions proposed by these VA systems; (ii) analyse them,

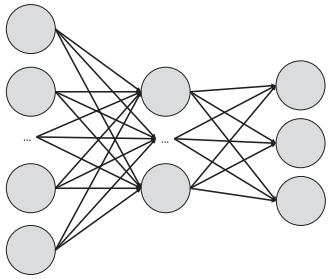
extracting common characteristics, foundations and limits and (iii) bridge the communities to which this article is directed. For the last point, we aim to make them aware of the best integration practices, identify promising areas for collaboration and present limitations in the current state of the art. The manuscript targets researchers and practitioners working in DL, XAI and VA. We aim to make DL and XAI practitioners aware of the benefits and opportunities of current state-of-the-art VA solutions. Specifically, they can gain knowledge of solutions that can improve the understanding of their models and potentiate the benefits of explanations returned by explanation methods. At the same time, practitioners and researchers working in VA can find a concise summary of the solutions adopted in the literature when dealing with DL models that are ready to be exploited by practitioners. Finally, identifying research gaps could stimulate the investigation of novel research directions by researchers working in both the VA and XDL fields to support different functionalities and explanations.

Comparison with existing work. While other works in the literature partially describe the state of the art of VA for XDL, their scope is different from that in our proposal. For example, Alicioglu and Sun [AS22] focused on the whole area of XAI, while our proposal targets only XDL. Hohman *et al.* [HKPC19] proposed a survey of DL visualization; however, unlike our proposal, their work did not focus on the explainability problem. Finally, Choo and Liu [CL18] provided an interesting initial overview of VA for XDL, but it is not a survey and contains just a brief overview of 18 research works. In contrast, we review the whole field, having analysed more than 60 papers.

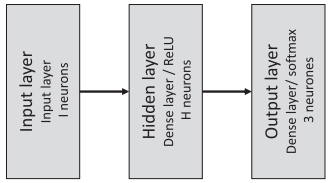
Contributions and findings. Our paper contributes to the literature as follows. We analysed the literature on the topic and proposed a five-way categorization (Section 4) to organize, compare and place solutions with similar goals in the literature. We identified, extracted and analysed 38 dimensions useful to classify and analyse the papers on the topic (Section 3). From the analysis (Section 5), we observe a rising interest in the subject and a progressive increase in complexity in the adopted solutions, especially for the most recent DL techniques. The advantages of these systems are undeniable: they help experts in designing, understanding and correcting the failures of DL models. However, we also identify some areas where future research could bring additional benefits. Namely, we argue for more research on VA systems supporting end users and systems supporting more confirmatory and what-if analysis in addition to exploratory analysis. Those analyses should work at both the model level (e.g. by changing the DL model internals) and the input level (i.e. by changing input features). Additionally, we invite researchers of the three communities to a tighter collaboration (Section 7) to fix some issues and challenges identified in the literature, such as the usage of a limited set of explanation methods, the trustworthiness of these systems and the lack of a standard interface between their frameworks (Section 6).

2. Background

To provide basic knowledge to the reader unfamiliar with some of them, this section introduces the core concepts and terminology used in the research areas of DL (Section 2.1), XDL (Section 2.2) and VA (Section 2.3).



(a) *Computational graph.*



(b) *Layered abstraction graph.*

Figure 1: *Graph-based representations of a DNN. The computational graph (a) encodes all the operations, while the layered abstraction graph (b) provides an overview of the architecture.*

2.1. Deep Learning

This section presents the basics of DL [DWA21]. We invite interested readers to look at recent surveys [PSY*19, DWA21] and popular books [GBC16] on the subject. A DL system is characterized by the *data* it handles, its *architecture*, the *training* and *inference* procedures, and its *evaluation*.

Data. We consider two types of data: input and output data. The *input data*, often called *samples*, are composed of several *features* and correspond to the information consumed by the Deep Neural Network (DNN) during the *inference* phase. Input data can be associated with *labels*, often called *ground truth*, which correspond to the expected results of the *inference* on the samples. Their shape depends on the *task* to solve (e.g. classification, regression, segmentation), and, together with their modality (e.g. image, video, text, time series), they directly influence the *architecture* of the DNN. The *output data* correspond to the result of the inference by the DNN of the input sample. For classification tasks, the ground truth and output are usually represented by a probability vector whose dimension corresponds to the number of classes.

Architecture. The smallest component of a DNN is the *neuron*, which computes an *activation* value by applying a non-linear transformation on the weighted sum of its input, where the weights are learned during the *training* process. A complete DNN is built from a computational graph (or *network*) (Figure 1a) where the nodes are neurons, and the edges are their dependency (weights). We name *model* a network with its learned weights. Neurons are usually arranged in layers: neurons of the same layer share the same input and output neurons. This computational graph can be abstracted by an *architecture* graph (Figure 1b) where each node is a layer, and the edges represent a quantity (output) sent by a layer to the next one. The layers are classified based on their location: *input*, *hidden* and *output layers*. The *input layer* is the first layer of the network; each

neuron i provides the features x_i of the input sample $\mathbf{x}$ to the next layer. Neurons of *hidden layers* take as input a set of activations of the previous layers and send their activations to the next layer. Finally, the *output layer* neurons take as input the activation of the last hidden layer, and their activations are precisely the network's output. Complex networks can have multiple input or output layers (e.g. taking an image and a caption as inputs or generating them). Layers can rely on the *attention* mechanism, which aims at computing, at inference time, a type of sample-dependent weight that allows the layer to selectively focus on some parts of the input while ignoring other irrelevant information [XBK*15].

The type and the operations performed by layers vary depending on the complexity of their connections. For example, recurrent layers maintain a memory of their state and reuse it for following operations, while convolutional layers convolve kernels on local patches of the activations of the previous layer. Thus, we describe the main families of architectures analysed by VA papers.

Convolutional Neural Networks (CNNs) [KSZQ20] leverage on convolutions, pooling and fully connected layers [LSL*17] to analyse matrix-based data while exploiting spatial information [LBH15]. The convolutional layer's output is called a *feature map* or *activation map*.

Recurrent Neural Networks (RNNs) [YSHZ19] exploit the temporal information encoded in the data. Neurons process one step at a time, adding the results of the operations performed on the previous features as additional input to the current one. The Long Short-Term Memory Network (LSTM) [HS97] is an example of these networks, widely used to deal with time series and textual data, a type of data that typically exhibits meaningful temporal patterns ready to be exploited by these models.

Transformers [VSP*17] are composed of several layers employing attention mechanisms across neurons' activation. Each layer contains several heads, specialized in capturing different aspects of the input through learned attention weights. While they were initially designed for machine translation, their application spreads to several domains, including computer vision and text classification.

Autoencoders such as the one used in *Generative Networks* and *Language Models* project input data in a latent space and then transform them into artificial data. They are trained to approximate the training data distribution and produce new samples similar to them. Inputs of these networks can be a description of what to generate, a starting input to be transformed, or a generic instruction. Variational Auto-Encoders [KW13] and Generative Adversarial Networks [GPM*14] are examples of this class.

Graph Neural Networks (GNNs) [WPC*21] are DNNs designed to deal with graph data. Based on the layers' definition, the research community explored several variants of GNNs, among which the most popular are: Graph Convolutional Networks [KW17], which use convolutional layers similar to CNNs; Graph Attention Networks [VCC*18], which use attention weights; Graph Isomorphism Network [XHLJ19], which use non-linear layers.

Deep Reinforcement Learning Models leverage previously presented models to solve tasks where agents interact with the environments [MKS*13]. Usually, they take as input the environment state

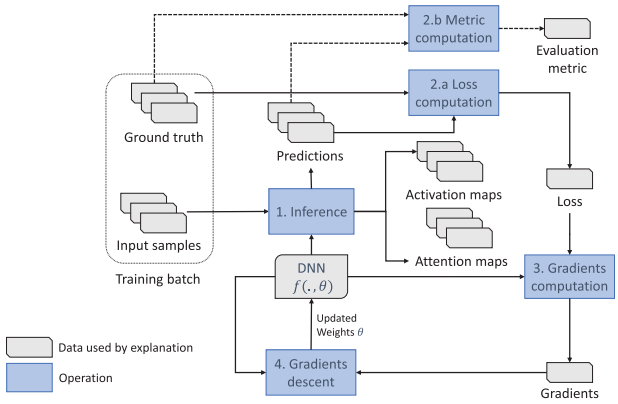


Figure 2: *Illustration of inference and gradient descent over a batch for a DNN. The process generates several data: predictions, activation maps, loss, metrics and gradients. Each of them can be used to compute explanations.*

observed by the agent and suggest which actions to take to maximize a reward. For example, the Deep-Q Networks [MKS*13] is a CNN, while the Advantage Actor-Critic (A3C) [MPV*16] Network adds an LSTM stacked on top of a CNN.

Similarly, self-explainable *Prototypes-based Models* leverage previously presented models. Additionally, they use specific representations of input training samples or artificially generated ones as *prototypes* to represent a family of samples [SSZ17, CLT*19]. The idea is to use the similarity between the computed prototypes and the current input to aid the model during the inference process.

Inference. During inference (Figure 2), the *model* is executed on unlabelled input data to generate its *prediction*: $\mathbf{o} = f(\mathbf{x}, \theta)$. This output, as well as the result of the inner computations (*latent vectors* and *attention*), can serve for XDL methods. The *latent vector* $\mathbf{h}_l$ of $\mathbf{x}$ for layer l corresponds to the activations obtained at layer l when applying $\mathbf{x}$ to $f(., \theta)$. The model until layer l is a data projector in another manifold, the *latent space*. The latent space of the last hidden layer p is of interest in many applications: it contains the latent vector $\mathbf{h}_p$ generated by the model, directly used by the output layer to compute the final prediction $\mathbf{o}$. Knowing what information is encoded in this space could allow users to understand the reasons behind the predictions of these systems.

Training. The training process (Figure 2) aims to adjust the model's parameters θ (i.e. the weights). It is an iterative process where N-sized batches of samples are fed to the model at each step j. An optimization algorithm (the *optimizer*) [LNC*11] computes the error of the approximation between the predicted output and the ground truth using a *loss function*. Then it adjusts the weights accordingly to the *gradients* of the model (e.g. the partial derivatives with respect to each parameter for each input sample/neuron). An *epoch* has been executed when all the training samples have been covered in the previous batches; a training process usually performs several epochs to train a network.

The public availability of models *pre-trained* on large corpora is one of the key elements that boosted the spread of DL. *Transfer*

learning and *fine-tuning* techniques [TSK*18] use them as starting points for the training process on a different dataset by using the weights of the pre-trained models as the initial weights of the new model. While transfer learning allows the model to adjust the pre-learned weights during the new training process, fine-tuning keeps them frozen but the ones of the last layer. The idea is that the pre-training on the large corpora makes the network capable of capturing the essential latent characteristics common to several tasks, then exploited to speed up the learning process of the current task.

Evaluation. Once the training ends (Figure 2), the model's performance is evaluated on an unseen dataset, called the *testing dataset*, to assess its generalization power (i.e. its ability to make correct predictions on unseen samples). This process uses *loss* functions and *evaluation metrics* that are task-dependent (e.g. precision, accuracy, RMSE, IOU).

2.2. eXplainable deep learning

XDL field aims at developing methods to improve the explainability of systems that use DL models. In literature, there is no consensus about the difference between the terms explainability and interpretability, and their definitions [CPC19, AB18]. In this regard, we do not take any side and use the term explainability as a general term, including all the methods that 'enable human users to understand, appropriately trust and effectively manage the emerging generation of AI systems' [GA19].

There are several ways to classify XDL methods. A first coarse distinction separates *post hoc* approaches and *self-explainable DNNs* [ZTLT21, ADS*20]:

- *post hoc* approaches use external means, like input perturbations or gradients, to explain the behaviour of a model that is not explainable by design [ADS*20];
- *self-explainable DNNs* include components embedded in the architecture to ease the explanation of the results, but without explaining the whole inference process yet.

This last category has recently emerged as a novel category of DNNs. Examples of this category are attentive models [BCB15], models based on prototypes [CLT*19] or models that generate an explanation along with the prediction (e.g. neural language models [LYW19]).

A further distinction [AB18] separates between *local* and *global* methods:

- *local* methods explain the decision of a model for a specific input sample;
- *global* methods describe the model behaviour on a wider range of inputs, often a whole dataset, capturing and extracting common patterns.

These methods can rely on various components to generate an explanation, such as:

1. *Gradients*. Their magnitude describes how a function $f(\cdot,\cdot)$ (i.e. the DNN) changes around the values of a variable (i.e. the feature x_i). It is often used to compute the contribution of each feature towards the current prediction by propagating back the gradient information from the output to the input. Intuitively, a high gradient towards a feature means a more significant impact since changes in its value produce big changes in the prediction; GradCAM [SCD*19] and Integrated Gradients [STY17] are two widely used methods of this category. Gradients can also be combined with activations to guide the *generation of synthetic inputs* containing a 'summary' of features recognized or learned by a neuron [OMS17, MV15].
2. *Perturbations*. The idea of perturbation-based methods is to modify the current input to probe the model's behaviour to test different scenarios and extract insights about its decision process. For example, by erasing or editing parts of the current input features and observing the effect on the output, it is possible to estimate the importance of each input feature [GKDF18, FV17, PDS18]. Another set of methods uses a dataset of perturbations of the current input to train *surrogate models*, like LIME [RSG16] or SHAP [LL17], that approximate the model behaviour on the neighbourhood of a given input. Finally, perturbations can also be used to extract contrasting explanations or counterfactuals, samples similar to the current input but associated with a different prediction.
3. *Activations*. They are the core elements of DNNs, and several methods propose to analyse them to extract insights about their behaviour. These methods usually start from specific (local) settings and analyse the changes in activation strengths since carrying a global analysis is often prohibitive due to a large number of neurons. For example, their changes can be used by optimization processes to discover what type of patterns in input features produce the highest activations [OMS17, MV15] or to find the most influential concepts of the input [KWG*18].
4. *Search algorithms*. Explanation methods can use search algorithms to extract samples from data and use them as explanations. For example, when they are used to find dataset samples similar to the current input, they can be used to reinforce or invalidate the current prediction. Other methods use them for discovering the most influential examples for the training process [KL17], or they can be combined with neuron activations to select dataset samples that *maximally activate* them [BZK*17].
5. *Attention weights*. They are one of the latest tools used for explainability, and they can be combined with the structure of the models, often self-explainable DNNs (e.g. prototypes-based models), to extract insights and explanations about their decision process [SGPR18, LCN20, AZ20].

While there exist several other taxonomies in the XDL literature that further distinguish between types of methods [ADS*20, GMR*19, AB18, ABZ21], they appear too broad and detailed in the context of this survey. For this reason, we propose a smaller categorization described in Section 4.1 alongside the main characteristics of explanation methods.

2.3. Visual analytics

VA is the science of analytical reasoning supported by interactive visual interfaces [TC06]. Keim *et al.* [KAF*08] provided a more formal definition: '*Visual Analytics combines automated analysis techniques with interactive visualizations for an effective understanding, reasoning and decision-making based on large and*

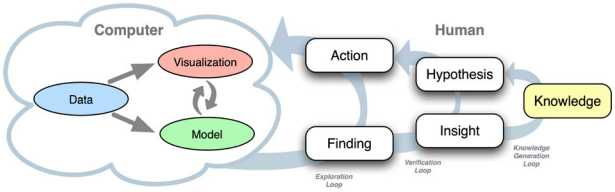


Figure 3: *VA process model [SSS*14] composed of four stages:* data*,* computational model*,* visualization *and* knowledge*. They are involved in a reasoning process based on three loops: exploration loop, verification loop and knowledge generation loop.*

complex data sets'. According to Kohlhammer *et al.* [KKP*11], a VA system should be able to synthesize and derive insights from massive, dynamic and uncertain data. The goal is to detect the expected, discover the unexpected and communicate these assessments effectively to the human user for further actions.

VA joins the computer-driven and the human-driven components by exploiting what computers and humans are good at [AAF*20]. The VA process model was proposed by Keim *et al.* [KAF*08] and extended by Sacha *et al.* [SSS*14] into the *Knowledge Generation Model*, emphasizing the human-centred part, as shown in Figure 3. The VA process model combines automatic and visual analysis methods and comprises four stages: *data*, *computational model*, *visualization* and *knowledge*. The first three stages represent the computer-driven component, while *knowledge* represents the human-driven component. There is no clear separation between the two parts since both are required in the most general workflow. While *data* is the starting point of all VA systems, computational models (e.g. DL models) work on them and transform them by using different techniques such as descriptive statistics, data mining and ML algorithms. *Visualization* is often the primary interface between analysts and VA systems, allowing the analyst to detect relationships and insights. *Knowledge* consists in finding evidence for existing assumptions or learning new knowledge about the problem domain. This stage is part of a broad reasoning process based on three loops: *exploration loop*, *verification loop* and *knowledge generation loop*.

Visual design guidelines often follow the Shneiderman's *Visual Information Seeking Mantra* [Shn96]: *overview first, zoom and filter, then details-on-demand*. The overview allows users to observe global patterns and general properties of the data. The details, typically in separate views, enable the users to comprehend the characteristics of the data at the low-level grain of analysis. In the context of VA, the mantra has been extended by Keim et al. in 'Analyse first, show the important, zoom/filter, analyse further, details-on-demand' [KMSZ06]. It indicates that more than retrieving and displaying the data using a visual metaphor is required. In fact, it is necessary to analyse the data, show the most relevant aspects and support the users by providing interaction models to get details.

Interaction. Human interaction is a key component in the VA workflow [TS20]. By interacting with a VA system, the user can steer it to generate new visualizations or computational models [EFN12, HASS22], analyse data from different perspectives, visually explore parameterization spaces for model and visualization [SHB*14, SW22]. Interaction allows both practical purposes like bringing the users closer to the desired goal and helping them to create a better mental model of the investigated problem. Yi *et al.* [YKSJ07] proposed seven high-level classes of interaction intents. The user can: *Select* something interesting by marking it for further investigation; *Explore* data to have a comprehensive understanding; *Reconfigure* data to obtain different insights; visually *Encode* data to adapt to her needs; *Abstract/Elaborate* to see the big picture or switch between different levels of detail; *Filter* to restrict the space of analysis and *Connect/Compare* to evaluate data similarities or relations.

Visualization methods and techniques. Visualization techniques represent data by exploiting combinations of *visual marks* (e.g. points, lines, areas) and *visual variables* (e.g. position, length, area, shape, angle, colour) [Ber67]. Visualization techniques depend on the kind of represented data, such as *text*, *numbers*, *multi-dimensional* or *hierarchical* data, or networks.

Text visualization techniques (e.g. [KK15]) range from classic matrix and area charts to word clouds [AWS05, HLLE14], and to the more complex word sequences. The latter exploit sequence visualization techniques [HDH*13] using node-link diagrams, glyph-based small multiples or directly annotating a text.

Number visualization techniques are various [AAF*20, KKEM10, TS20] and the common ones are *bar charts*, *pie charts*, *line charts* and *area charts*, and *heatmaps*. Data distributions, instead, are commonly represented with *histograms*, *box plots* and *violin plots* [HN98].

Multi-dimensional Data visualization techniques are various. The *scatterplot* [SG18] represents data as 2D or 3D points using Cartesian or polar coordinates. For high-dimensional data, often dimensionality reduction techniques (e.g. PCA, MDS, t-SNE, ISOMAP and UMAP) [EMK*21] are used with *scatterplots*. They allow projecting the data in a low-dimensional space while trying to keep intact characteristics of the data existing in the original high-dimensional space. *Parallel coordinate plots (PCPs)* [ID90] show multi-dimensional data as polylines that cross vertical parallel axes (dimensions). Several approaches have been proposed to enhance Parallel Coordinate Plot (PCP) visual quality, like clustering or sampling the polylines or axes sorting [HW12, JLJC05, BZP*20]. *RadViz* [HGM*97] and *star coordinates* [Kan00] are two of the most popular projection methods that represent high-dimensional data as 2D points on a radial layout while preserving at the same time their relationship with the original dimensions [RLSR21]. Several parameterizations have been proposed to enhance their visual quality (e.g. [DFF10, dCT19, ABL*22, RLSR21, RS14, ABL*19]).

Hierarchical data visualizations usually correspond to *Treemaps* [Shn92, SW01], a space-filling method of visualizing hierarchical datasets showing the hierarchy using nested rectangles. Their area encodes a numerical attribute, while colour [STLD20] or even more complex glyphs inside them [ABC*19] encode additional information.

Network visualization techniques are in general, *adjacency matrix* or *node-link diagrams*. The first is a basic technique, and the discovery of patterns highly depends on the rows/columns order [BBR*16]. The second represents graph nodes as circles

and edges as lines that connect them. The visibility of interesting sub-structures is highly dependent on the used layout algorithm [BBDW16].

3. Procedure

This section describes the procedure for collecting and analysing this survey's final set of papers. We adopted a three-fold strategy:

- Keyword search on the main indexing platforms (e.g. Elsevier Scopus, IEEE Xplore) where the main searched keywords are: *visual analytics*, *visualization*, *deep learning*, *neural network*, *deep model*, *explainable*, *interpretable* and *understanding*;
- The collection of seed papers coming from existing surveys on similar but broader topics; we considered the works by Alicioglu and Sun [AS22] reviewing XAI literature, Hohman *et al.* [HKPC19] on DL visualization and Choo and Liu [CL18] brief overview on VA for XDL;
- The systematic collection of papers from the last 10 years of the main journals, conferences and workshops related to the topic of this survey, namely: ML (e.g. NeurIPS, ACL, ICML), Human–computer interaction conferences (e.g. ACM SIGCHI), visualization and VA (e.g. IEEE TVCG, IEEE CGA, CGF, IEEE VIS, EUROVIS), as well as some works available through preprint repositories (arXiv) or web-journals (Distill).

We constrained each collected paper to several key requirements for selecting them for this survey:

- It must contain a VA solution where the user can actively interact with the data/model;
- The proposed solution must involve a DNN or must be tested on it (e.g. cases of model-agnostic solutions);
- The proposal aims to explain the behaviour of a DL model in terms of either what it has learned, why/how it produces a given prediction or identifying what elements of the input most influence the current prediction.

We do not include works that focus on monitoring and improving the performance of a DNN, as they do not pertain to our primary goal of describing solutions that help explain and understand a given network's behaviour.

According to these criteria, our survey does not cover the following types of proposals:

- model-agnostic systems that are not tested on DL architectures (including the ones only tested on shallow architectures) [KDS*17, KCC*20, CYO*20];
- systems that focus on embeddings [STN*16, LNH*18] since they often include embeddings computed with algorithms not based on DL;
- educational systems that focus on simplified simulations rather than real-world applications [KTC*19, NQ17, Har15, SCS*17, WTS*20];
- static visualizations [WLC18, HGBA20];
- systems that focus only on assessing and presenting the performance of the system [CPCS20, LCJ*19, RAL*17, AHH*14, ACD*15];

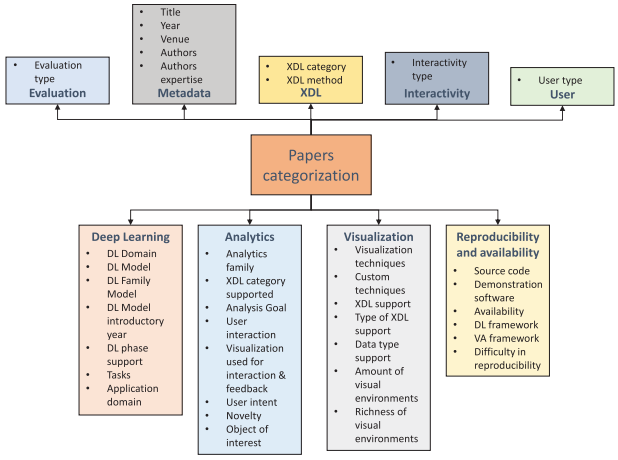


Figure 4: *Thirty-eight dimensions were used to characterize each paper, aggregated within nine groups. They allowed us to properly extract information of interest for each paper and organize the paper.*

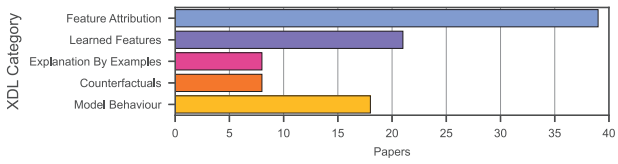


Figure 5: *Distribution of the VA solutions considered in this survey grouped by the methods employed by the XDL field.*

- analytic or visual solutions that aim at improving the explainability of DNN, but that are not implemented in a VA system [RFFT17, CEP20];
- systems that identify bugs and problems on DNN without connecting them to the knowledge learned by the systems [CEH*19, ZWM*19] (e.g. classifying a model as biased using metrics, without inspecting feature attributions or features learned by the neurons).

This 5-month process allowed us to initially collect 196 papers, then filtered to 67, thanks to our key requirements. Each paper was read by at least two team members, categorized in one of the five categories (Section 4.1), and then analysed according to 38 dimensions (some of them with multiple sub-categorization, in particular for visualization usage). The dimensions are organized into nine groups (papers metadata, DL, XDL category, visualization, analytics, interactivity, users, evaluation, reproducibility and availability) capturing paper-related (e.g. year of publication, venue, information about the code), DL-related (e.g. model, year of publication of the model's paper, phase, data type) and VA-related (e.g. target users, application domain, interactivity type, degree of evaluation, visualization techniques, analytics, dimensionality reduction techniques) aspects. The full categorization is visible in Figure 4. Based on the forward and backward analysis of citations and references, a second refinement phase brought the final number of papers to 67, whose distribution is visible in Figure 5. This final set of papers is listed in Table 1 and discussed in the following sections.

Table 1: *List of the 67 Visual Analytics systems considered in this state-of-the-art report. The systems belong to five XDL categories: feature attribution (**FA**), Learned features (**LF**), explanation by examples (**EE**), counterfactuals examples (**CE**) and model behaviour (**MB**). Target users can be: architects (**A**), trainers (**T**) and end users (**E**). Interactivity can be: passive (**P**), interactive input observations (**I**), and interactive model observations (**M**). Phase can be: training (**TR**) and testing (**TE**). Evaluation can be: quantitative user study (**Q-USt**), user study with feedback (**F-USt**), case study with feedback (**F-CSt**), case study (**CSt**) and usage scenarios (**Usc**). Furthermore, the table reports whether the authors have provided the source code of a system. Table 2 shows additional aspects of the considered systems.*

#	System Name	Reference	Venue	Year	Category					Application Domain	DL Model	Users			Interactivity			Phase		Evaluation					Code
					FA	LF	EE	CE	MB			A	T	E	P	I	M	TR	TE	Q-USt	F-USt	F-CSt	CSt	USc	
01	DG-Viz	[LYY*20]	JMIR	2020	■					Medical	RNN			■		■			■				✓		✓
02	RetainVis	[KCK*19]	TVCG	2018	■					Medical	RNN			■		■	■		■				✓		✓
03	V-Awake	[CWGvW19]	CGF	2019	■					Medical	Other			■	■				■			✓			✓
04	ViSFA	[WWM20]	ArXiv	2020	■					Domain agnostic	RNN			■	■			■						✓	✓
05	VisLRPDesigner	[HJZ*21]	CGF	2021	■					XDL	CNN		■			■	■		■	✓	✓			✓	✓
06	VisQA	[JKV*22]	TVCG	2021	■					Domain agnostic	Transformers	■				■	■		■	✓		✓			
07	DeepVID	[WGZ*19]	TVCG	2019	■					Domain agnostic	Other	■					■	■				✓			
08	GLANCE	[vdBCR*20]	IOVS	2020	■					Medical	CNN			■	■				■	✓	✓	✓			
09	IVDAS	[HSL*21]	JVIS	2021	■					BioInformatics	CNN		■			■			■				✓		
10	GANViz	[WGYS18]	TVCG	2018	■					Domain agnostic	Generative	■			■			■			✓		✓		✓
11	explAIner	[SSSE19]	TVCG	2019	■					Domain agnostic	Variable		■		■			■	■		✓				✓
12	AttributionHeatmaps	[WONM18]	IEEE Big Data	2018	■					Domain agnostic	RNN		■		■			■					✓		✓
13	NNVA	[HLW*19]	TVCG	2020	■					BioInformatics	Variable			■		■	■		■			✓			
14	NJM-Vis	[JCM20]	IUI	2020	■					Domain agnostic	Variable		■		■				■		✓		✓		
15	BERTViz	[Vig19]	ACL-SD	2019	■					Domain agnostic	Transformers		■			■			■					✓	✓
16	–	[DWSZ20]	PacificVIS	2020	■					Domain agnostic	RNN			■		■			■	✓				✓	✓
17	Dodrio	[WTC21]	ACL-SD	2021	■					Domain agnostic	Transformers		■		■				■					✓	✓
18	Attention Flows	[DWB21]	TVCG	2020	■					Domain agnostic	Transformers	■				■			■		✓		✓		✓
19	–	[CHS20]	Vis.Inform.	2020	■					Linguistics	CNN		■			■			■				✓		
20	–	[HCC*20]	Distill	2020	■					Game agent	DRL	■			■				■					✓	✓
21	TSViz	[SMM*19]	IEEE Access	2019	■					Domain agnostic	CNN	■			■				■					✓	✓
22	–	[CGR*17]	VADL	2017	■					Domain agnostic	RNN	■			■			■						✓	
23	AttViz	[ŠSE*21]	EACL	2021	■					Domain agnostic	Transformers		■		■				■					✓	
24	–	[MFH*21]	TVCG	2020		■				Domain agnostic	CNN		■		■				■		✓		✓		✓
25	DeepEyes	[PHG*18]	TVCG	2017		■				Domain agnostic	CNN	■			■			■						✓	
26	ProtoViewer	[ZDXR20]	VIS	2020		■				Domain agnostic	CNN	■			■				■					✓	
27	Deep View	[ZXZ*17]	ICML-Viz	2017		■				Domain agnostic	CNN	■			■			■						✓	
28	CNNComparator	[ZHP*17]	VADL	2017		■				Domain agnostic	CNN		■		■				■					✓	
29	DeepVis	[YCN*15]	ICML	2015		■				Domain agnostic	CNN	■			■				■					✓	
30	ShapeShop	[HHC17]	CHI	2017		■				Domain agnostic	Variable		■			■			■					✓	
31	Bluff	[DPW*20]	VIS	2020		■				Domain agnostic	CNN	■			■				■					✓	✓
32	NeuroCartography	[PDD*22]	TVCG	2021		■				Domain agnostic	CNN		■		■				■	✓				✓	✓
33	Summit	[HPRC20]	TVCG	2019		■				Domain agnostic	CNN	■			■				■					✓	
34	TopoAct	[RCPW21]	CGF	2021		■				Domain agnostic	Variable		■		■				■				✓		
35	CNNVis	[LSL*17]	TVCG	2016		■				Domain agnostic	CNN	■			■				■				✓		
36	RNNVis	[MCZ*17]	VIS	2017		■				Domain agnostic	RNN		■			■			■		✓		✓		
37	DeepCompare	[MMD*19]	CG&A	2019		■				Domain agnostic	Variable		■		■				■				✓	✓	
38	What-If Tool	[WPB*19]	TVCG	2019				■		Domain agnostic	Variable		■			■			■				✓		✓
39	DECE	[CMQ21]	TVCG	2020				■		Domain agnostic	Variable		■			■		■	■		✓			✓	✓
40	–	[BDME20]	VizSec	2020					■	Cybersecurity	Variable	■			■				■					✓	
41	GNNLens	[JWW*22]	TVCG	2020					■	Domain agnostic	GNN	■			■				■		✓		✓		✓
42	ActiVis	[KAKC18]	TVCG	2017					■	Domain agnostic	Variable		■			■			■				✓	✓	
43	DRLIVE	[WZY*22]	TVCG	2021					■	Game agent	DRL		■			■			■			✓			✓
44	SCANViz	[WZY20]	PacificVIS	2020					■	Domain agnostic	Generative	■				■	■	■				✓			✓
45	–	[SW17]	VADL	2017	■	■				Domain agnostic	CNN		■		■				■				✓		
46	SANVis	[PCN*19]	VIS	2019	■	■				Domain agnostic	Transformers	■				■			■				✓		
47	Seq2Seq-Vis	[SGB*19]	TVCG	2018	■		■			Domain agnostic	Other		■			■	■		■				✓		
48	exBERT	[HSG20]	ACL-SD	2020	■		■			Domain agnostic	Transformers		■			■			■				✓		✓
49	ProtoSteer	[MXC*20]	TVCG	2019	■		■			Domain agnostic	Other			■			■		■			✓			✓
50	PolicyExplainer	[MSHB22]	PacificVIS	2022	■			■		Domain agnostic	DRL			■		■			■		✓		✓		✓
51	Recast	[WSP*21]	ACM-HCI	2021	■			■		Linguistics	Transformers			■		■			■		✓				
52	MultiRNNExplorer	[SWJ*20]	PacificVIS	2020	■				■	Environment	RNN		■		■				■					✓	
53	DQNViz	[WGSY19]	TVCG	2018	■				■	Game agent	DRL	■			■			■				✓			
54	DRLViz	[JVW20]	CGF	2020	■				■	Game agent	DRL	■			■				■		✓				
55	–	[ZZM16]	ICML	2016	■				■	Game agent	DRL	■			■				■				✓		
56	VATUN	[PYN*21]	EuroVis	2021	■				■	Domain agnostic	CNN		■			■			■					✓	✓
57	M2Lens	[WHJ*22]	TVCG	2021	■				■	Linguistics	Variable		■		■				■			✓			✓
58	Blocks	[BJY*18]	TVCG	2017		■			■	Domain agnostic	CNN	■			■			■					✓		
59	AEVis	[LLS*18]	TVCG	2020		■			■	Domain agnostic	CNN	■			■				■				✓		
60	TNNVis	[NHP*18]	PacificVIS	2018		■			■	Domain agnostic	CNN	■				■		■	■				✓		
61	CNN2DT	[JLL*19]	JVIS	2020		■			■	Domain agnostic	CNN		■		■				■		✓		✓		✓
62	DynamicsExplorer	[HLvB*20]	PacificVIS	2020			■		■	Robotics	DRL	■			■				■					✓	
63	LSTMVis	[SGPR18]	TVCG	2017			■	■	■	Domain agnostic	RNN		■			■	■		■		✓			✓	✓
64	USEVis	[JTH*21]	Vis.Inform.	2021	■	■	■			Domain agnostic	Transformers		■		■				■			✓			
65	Melody	[CBN*20]	ArXiv	2020	■		■	■		Domain agnostic	Variable	■				■			■					✓	
66	HypperSteer	[WM20]	ArXiv	2020	■			■	■	Domain agnostic	RNN		■			■			■					✓	
67	NLIZE	[LLL*19]	TVCG	2018	■		■	■		Linguistics	Other	■				■	■		■		✓		✓		✓

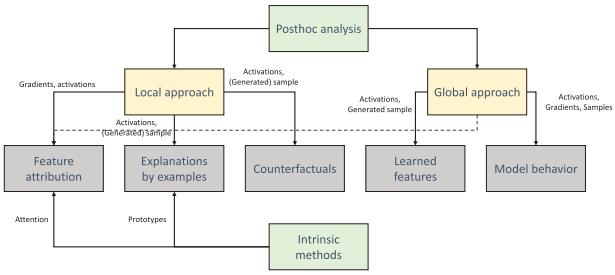


Figure 6: *Categorization of explanation methods. We can distinguish between methods that use external means (post hoc) and methods that use the model's internals (intrinsic), and between methods that provide explanations only for the current input (local) and the ones for the general model's behaviour (global).*

The complete categorization is provided in the supplemental material and through an interactive literature explorer[1] based on SurVis [BKW16], which allows to explore and analyse the final set of papers.

4. Papers Categorization

This section provides a general overview of the analysed VA solutions by describing them in terms of the explanation category they support. We introduce our categorization, describing its rationale and some key XDL methods in each category (Section 4.1). Then, we describe how VA solutions support and use them for explainability (Section 4.2).

4.1. Explanations categories

While several taxonomies have been proposed in the literature to distinguish between types of methods [ADS*20, GMR*19, AB18], they usually focus on multi-level and fine-grained categorizations. However, considering the current literature on VA for XDL, these taxonomies could lead to many categories that include only one or few papers, thus missing the objective of providing proper discrimination of the works. Moreover, often the analysed papers use different wording to refer to the same explanation, thus lacking consistency across the literature.

Starting from a cross-analysis between common categories used in XAI surveys [ADS*20, GMR*19, AB18], the ones currently supported by VA systems, and a process of abstraction, we propose to separate the methods using the following categories (Figure 6): *feature attribution* [BSH*10, STY17], *learned features* [EBCV09], *explanation by example* [Lip18, KK19], *counterfactuals* [WMR17] and *model behaviour*. The first four are well-known concepts in the XDL literature, but they are often further divided into more fine-grained sub-categories. Conversely, the *model behaviour* category is specific to VA systems and identified during our analysis.

[1] https://aware-diag-sapienza.github.io/VA4XDL

In particular, we follow the rationale of getting a dense categorization where each category includes a single group of methods that share the same goals. This rationale leads us to merge some too fine-grained categories and split others. For example, in XDL taxonomies, *feature attribution* techniques are often split based on the model [ADS*20], the data on which they are applied [GMR*19], or the method [AB18], and *counterfactuals* are set as a sub-category of *explanation by example* [AB18] since both categories elect examples similar to the input. Conversely, in our categorization, we merge all the types of attribution methods in a single category to keep together methods with the same goal and, simultaneously, to avoid producing a sparse categorization. At the same time, we split *explanations by examples* and *counterfactuals* since they have different goals and require different methods.

Below, we describe each category, the questions they address, how they can be computed, and some popular methods.

1. **Feature attribution**: These methods assign a score to each input feature based on its impact in determining the predicted outcome from the model [STY17]. They answer the question 'Where is the model focusing on for computing the prediction?' and give clues about the question 'Why does the model return this specific output?'. They can be computed at the global and local levels, using either post hoc methods or self-explainable DNNs. This category is the most studied in the literature; thus, we observe the highest heterogeneity in the proposed techniques. They include gradients-based methods, like Grad-CAM [SCD*19] and Integrated Gradients [STY17], methods based on *hand-crafted decision rules* that back-propagate information from the last layers back to the inputs, like Layer Relevance Propagation (LRP) [BBM*15] and Deconvolution Network [ZKTF10, ZF14], *perturbation-based* methods, like LIME [RSG16] or SHAP [LL17], and *intrinsic methods*, which combine properties of the model and attention weights [AZ20].
2. **Learned features**: These methods associate sets of concepts to neurons, groups of neurons or layers in terms of features they can recognize. They address the question of 'What has it learned during the training process?'. Most of them exploit a combination of activations, gradients, search algorithms and supporting models (e.g. generative models). They can be combined to select dataset samples that *maximally activate* the neurons, to guide the *generation of synthetic inputs* or to extract rules [AK12] and decision trees [CS95] that approximate the system's behaviour. When these methods consider different activation ranges, the extracted learned features are referred to as 'multi-facet learned features', where each facet captures one behaviour of a given range. Since the goal is to interpret part of the network, they can be considered global and post hoc.
3. **Explanations by examples**: These methods extract and use training samples as explanations. They address the question of 'Which samples are considered similar by the model' by showing samples on which the model acts similarly. The idea is to expose such samples and let the user extracts the features that lead the model to create the association between the current input and the prediction. Methods are usually search-based and differ in the way in which they define the similarity (e.g. at the input, latent space or feature attributions level) and are

usually based on enhanced versions of the K-nearest neighbour algorithm (KNN) [Lip18] or on self-explainable DNN [CLT*19, RCN22]. These methods are usually local and post hoc, but they can also be computed at the global level for some self-explainable DNNs.

4. **Counterfactuals**: these methods aim at finding the minimum number and magnitude of edits needed on the current input sample to obtain a different prediction [LLM*19]. In other words, counterfactuals are samples as similar as possible to the current input but associated with a different prediction. They answer the question 'What do I have to change to obtain a different outcome' and they are beneficial for recourse (i.e. the actions required for reversing unfavourable decisions by algorithms [VA20]). Counterfactuals can be generated by perturbations or algorithms [PSS*20], satisfying some constraints about the edits, or extracted from a dataset [WMR17], using proper distance functions and search algorithms. Methods of this category are local and post hoc methods.
5. **Model Behaviour**: Methods of this category aim at extracting common patterns of the model behaviour. They address the question 'How does the model react in a given situation?', where the situation is a set of similar inputs or the whole dataset. They combine pattern mining on activations, human-in-the-loop (i.e. interactions with the user) techniques, and often methods of the previous categories. They can be categorized as global post hoc methods. Examples of this category are methods that use activation patterns of the last hidden layer to understand and steer the output of generative models [BLW*20], methods that combine activation patterns and learned features to explain misclassifications on adversarial examples [LLS*18], or methods that combine patterns in inputs and outputs of the model with feature attribution to extract the policies followed by an agent trained using reinforcement learning [WGSY19].

4.2. Papers overview

In this section, we describe how VA solutions support and use the categories presented in the previous section for explainability purposes. In particular, while the categories represent the *WHAT*, and thus the explanation objects used to provide explanations, here we describe *WHY* VA systems use them in terms of addressed analytical tasks and goals, and *HOW* they support them, in terms of combinations of visualizations, analytics and user interactions. The VA solutions are listed in Tables 1 and 2, while Figure 5 shows their distribution according to the categorization.

4.2.1. Feature attribution

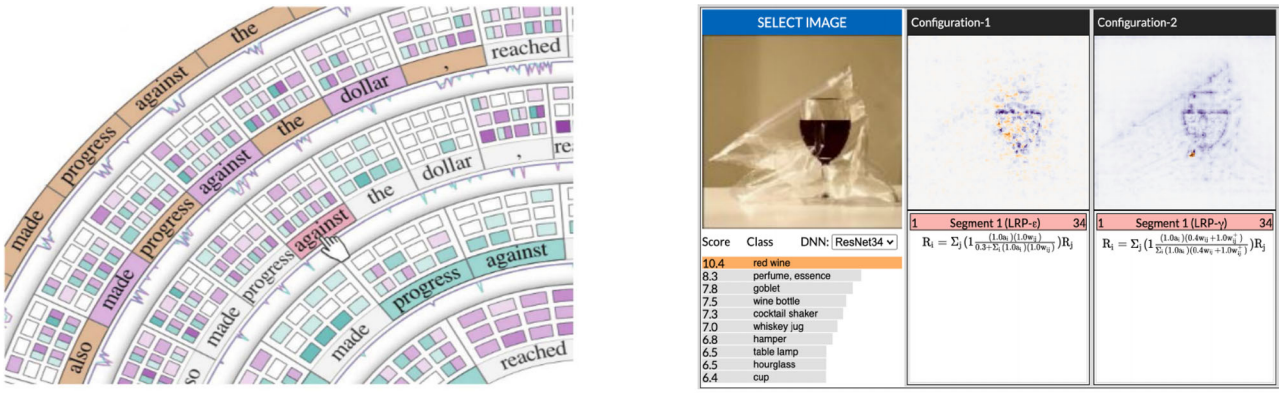
As described in the previous section, feature attribution methods highlight the input features the model is focusing on at the inference stage. VA systems use them for providing explainability and supporting several analytical tasks, such as understanding the motivations behind the model's predictions. They can achieve this goal by identifying the key factors (i.e. features) affecting the prediction results [SMM*19, CHS20, vdBCR*20, CWGvW19, KCK*19, ŠSE*21, HSL*21, WGYS18], without going into detail about the mathematical operations behind the model. By visualizing and analysing these information, users can assess the reliability [vdBCR*20] or the robustness [PYN*21] of a prediction, detect when a decision is biased [Vig19, JKV*22], dissect failures [CHS20, HCC*20, WGZ*19], discover new relations among factors, especially in the medical domain [vdBCR*20], compare different models' behaviour [HSL*21, DWB21], or improve the design of the models themselves [HSL*21].

We observe heterogeneous visual encodings adopted to support feature attributions. The most used ones are heatmaps for local post hoc feature attributions on images (Figure 7b) [vdBCR*20, HSL*21, HJZ*21, ZZM16, HCC*20, WGZ*19, SW17, CBN*20, JVW20] and text [CHS20, CGR*17, ŠSE*21, JTH*21]; matrices [WONM18, DWB21, JKV*22, PCN*19, LLL*19], node-link diagrams [JTH*21, Vig19, JCM20, LLL*19] and custom Sankey diagrams [DWSZ20, PCN*19, HSG20, MSHB22] for self-explainable attentive models; bar charts[WWM20, PCN*19] or averaged inputs [WGYS18, WGSY19] for global feature attribution; and enhanced line [SMM*19, CWGvW19, LYY*20, SWJ*20, ŠSE*21], area chart [KCK*19] or bar chart [MXC*20, KCK*19, SWJ*20, WWM20] for post hoc approaches to sequential data. Among them, systems that support the analysis of attentive models, and in particular of Transformers, employ the most complex and novel visualization techniques (Figure 7a) such as radial layouts [WTC21, DWB21] or grid ones [WTC21, DWB21, ŠSE*21]: these systems must show the flow of attention weights across multiple layers simultaneously to help the user understand the most important features.

These visualizations are usually enriched by additional elements and linked to other system views. Combined with interactions, they allow users to investigate and analyse the models. Examples of added information are the attribution scores' magnitude, which is encoded using colours, size [JCM20], opacity [WSP*21] or just its value, or bounding boxes [HSL*21, JKV*22, CBN*20], which highlight the most important region over images. While sorting [CHS20, MXC*20, KCK*19, Vig19, PCN*19] and filtering [WONM18, DWSZ20, JKV*22, JTH*21] by attribution scores capabilities are quite common to ease the data exploration and reduce the visual clutters, some works provide additional tools for a deeper understanding. For example, feature removal interactions guided by local attributions [HSL*21, HJZ*21, KCK*19] (e.g. via brushing over an image) can be exploited to perform a *What-If Analysis*. A similar analysis is also supported by the VATUN system [PYN*21] where, by applying several transformations, the user can alter the current input to let the system show the difference between post hoc feature attributions of the original and the transformed image as heatmaps. Interactive comparison between attributions of several data instances [vdBCR*20, KCK*19, SWJ*20, JTH*21] can guide the user to discover crucial features across whole categories. In the case of self-explainable DNNs, some systems also allow users to dynamically change the value of the elements used for providing intrinsic feature attributions (e.g. attention weights) for a given input to see how the model changes its prediction accordingly [KCK*19, JKV*22, SGB*19, LLL*19] or even steer the model forcing it to update its parameters to align the attribution scores to the expected ones [MXC*20, KCK*19]. Additionally, in the global feature attribution case, by using classic interactions, like lasso selection and filtering [HLW*19], users can select a sub-set of the dataset,

Table 2: *List of the 67 Visual Analytics systems considered in this state-of-the-art report by looking at the VA aspects: visualization and analytics. Concerning the analytics proposed to the users, the table reports the high-level family, and the subject (i.e. data handled). The analytics are then divided into novel analytics (●), custom analytics (○) based on slight modifications of existing solutions, and classic analytics that are existing solutions used as is (blank space). The IDs are consistent with Table 1.*

#	Category					Visualizations																Analytics Family											Analytics Subject	Analytics Novelty
	FA	LF	EE	CE	MB	Heatmap	Node-Link	Line/Area	Bar	Image	Scatter	Words	Table	Matrix	Sankey	Pie	PCP	Radial	Treemap	Boxplot	Tree	Similarity Analysis	Feature Influence	Aggregation/Summarization	Clustering	What-if Analysis	Model/Framework	Search/Mining	Saliency Analysis	Activation Maximization	Statistical Analysis	Semantic Analysis	Pattern Analysis		
01	■						■	■	■		■					■						■				■	■							Input	●
02	■							■	■		■		■									■					■							Input/Output	●
03	■					■		■			■		■	■						■		■												Temporal stages	
04	■							■	■		■			■									■											Features	●
05	■					■			■	■	■		■	■											■									Image	●
06	■					■	■			■				■											■									Attention Heads	○
07	■					■				■	■						■																	Image	●
08	■					■		■		■																		■						Image	
09	■					■	■	■	■	■	■		■															■						Image	
10	■					■	■	■		■	■												■											Image	○
11	■					■	■	■	■	■																■								All	○
12	■					■	■	■					■	■														■						Input	
13	■					■	■	■	■				■					■					■											Input	○
14	■						■		■			■	■										■											Words	○
15	■					■	■					■																				■		Attention Heads	
16	■					■	■					■																			■			Sentences/Words	○
17	■						■					■	■					■				■									■			Words	●
18	■											■				■		■					■	■										Words	●
19	■								■	■		■											■											Words	●
20	■					■		■	■	■																								Image	
21	■							■	■														■											Input/CNN Filters	●
22	■							■	■			■					■								■										Sentences/Words	
23	■							■			■	■													■										Words	○
24		■				■		■	■	■	■					■				■							■								Network	●
25		■				■		■	■		■								■													■			Filters	●
26		■						■	■	■			■	■									■												Prototypes	●
27		■				■		■		■	■												■		■										Neurons	●
28		■				■			■	■													■								■				Input/Model	
29		■				■				■				■														■						Images	○
30		■				■				■																			■					Activations	
31		■					■	■		■					■														■					Input	○
32		■				■	■			■	■				■							■												Neurons	●
33		■					■		■	■	■				■								■											Neurons	●
34		■					■			■													■											Activations	○
35		■				■	■								■								■	■										Neurons	○
36		■				■	■	■	■			■			■					■				■										Hidden states	○
37		■				■			■			■	■						■											■				Activations	
38				■		■			■	■		■											■					■						Input	○
39				■			■	■	■				■															■						Input	●
40					■				■		■										■	■		■						■					Activations	○
41					■		■		■		■		■				■		■			■												Internals	○
42					■		■				■		■	■								■												Activations	
43					■	■	■	■	■	■	■					■						■			■									Internals/Input	○
44					■			■					■			■	■													■				Internals	
45	■	■				■				■																			■					Input/Activations	○
46	■	■				■	■		■			■		■	■										■							■		Attention Heads	○
47	■		■				■				■	■			■							■				■								Input/Attention Heads	○
48	■		■			■	■		■			■															■						Words	○
49	■		■				■		■			■															■						Sentences	○
50	■			■			■	■								■		■					■											Internals	●
51	■			■								■															■							Words	○
52	■				■			■	■		■			■						■			■		■									Features/Hidden units	●
53	■				■	■		■	■	■						■						■										■		Internals/Input	
54	■				■	■		■		■	■		■									■								■				Internals	
55	■				■	■				■	■											■							■					Image	
56	■				■	■		■		■	■												■											Input	
57	■				■	■	■	■		■	■	■			■			■				■	■											Features	●
58		■			■	■				■				■							■		■					■						Neurons	●
59		■			■	■	■			■				■									■											Internals	●
60		■			■	■	■																											nan	
61		■			■	■	■			■	■		■								■					■								Model	○
62			■		■	■											■					■												Hidden States	
63			■	■	■	■		■				■										■												Hidden States	
64	■	■	■			■	■					■					■					■		■										Attention Heads	○
65	■		■	■		■	■		■	■							■									■								Internals	●
66	■			■	■			■	■		■											■												Internals	
67	■		■	■		■	■		■		■							■	■							■								Words	○



(a) *Attention Flow [DWB21]* **(b)** *VisLRPDesigner [HJZ*21]*

Figure 7: *Examples of how VA systems support feature attribution (a) attention flow [DWB21] employs a complex visualization for text data and Transformers models. It supports comparing and analysing attention weights across layers and heads between two different models by adopting a radial layout. Each ring corresponds to a given layer, while small rectangles adjacent to the word encode the attention heads. The cell colour indicates whether the token is attended equally (orange) or one model attends it stronger than the other (purple or turquoise). Users can select tokens or heads, and the system will highlight paths of attention relative to the selection. (b) A colour-encoded heatmap for feature attribution on images in the VisLRPDesigner [HJZ*21] system, where orange and blue colours encode input pixels that contributed to the prediction in a positive or negative way, respectively. The user can brush over the heatmap or the input image to change the contribution's 'sign' or remove some pixels and study whether the relevance scores are faithful to the CNN behaviour.*

usually depicted in a scatter plot, on which to aggregate the attribution scores based on the sum [CHS20, JTH*21], the mean [ZDXR20] or *Clustering* algorithms [PCN*19] of the individual contributions. The computed scores are then visualized using aggregated saliency maps, bar plots, graphs or more complex visualizations for attention heads.

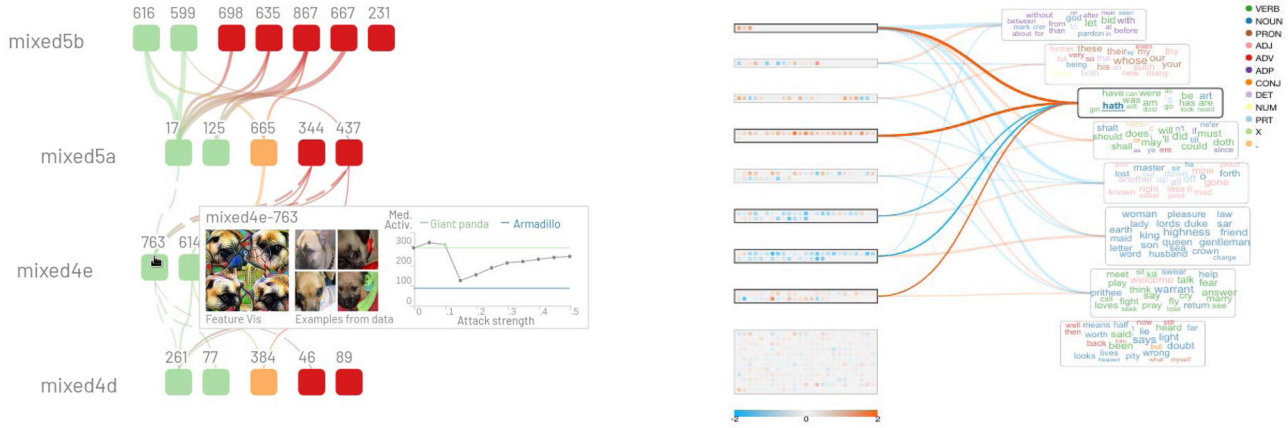
Illustrative example. Attention flow (Figure 7a) deeply relies on the *feature attribution* method. One of its evaluations proves that the application is useful for tasks of type question answer verification: when the user has selected the glyphs that correspond to the words of an answer, the system has highlighted dependent tokens' glyphs that correspond to the question. This shows the model has properly learned the concepts of interest.

Summary. Overall, most systems choose an a priori static feature attribution method and its configuration to compute the scores and then use its outputs. Few of them [HJZ*21, WGZ*19, SSSE19] allow users to modify the method's configuration or choose an alternative method. Only two systems address the problem of feature attribution for multi-modal models. They can highlight the impact of each modality by using side-by-side visualization [JKV*22] or swarm plots [WHJ*22] both at the local and global levels. They highlight the most important modality for the current prediction in the first case. In the second case, they measure and aggregate the influence of each modality across the whole dataset. Despite that, we observe a high heterogeneity in the adopted solutions, covering a wide range of models and data. Therefore, feature attributions appear as (i) the most supported category, (ii) the category that supports the highest number of analytical tasks and (iii) the most mature one in terms of visualizations, interactions and analytics. This result is not surprising since feature attribution is the most popular category for XDL methods and probably the easiest to relate to when approaching the XDL research field.

4.2.2. Learned features

Learned features methods aim to discover which features a neuron, group of neurons, or layer has learned to recognize during the training. They differ from global feature attributions because there is no relation with predictions, but they only focus on properties recognized by the component. VA systems employ them mainly to help users discover the semantics captured by neurons [BJY*18, HPRC20, JLL*19, ZDXR20, MCZ*17, MMD*19, RCPW21, PCN*19] and understand how the network works, by checking how low-level features are aggregated into high-level features [LSL*17, HPRC20, DPW*20, PDD*22, JTH*21, LLS*18]. However, learned features can also be used to diagnose the training process [ZXZ*17, PHG*18], for example, by visualizing the evolution of neurons over different epochs [ZXZ*17], to explore the role of layers [SW17], to decipher adversarial attacks [DPW*20], to check that the learned knowledge is reasonable or aligned to the expected one [LSL*17, JTH*21] or to compare the learned knowledge of different models [MFH*21].

The core visual elements representing learned features are modality-dependent but stay quite homogeneous. Most VA systems focusing on learned features are related to vision or text modalities. Vision modality leverages on image patches [YCN*15, ZHP*17, HPRC20, DPW*20, JLL*19, ZDXR20, PDD*22, RCPW21, LLS*18] or generated synthetic images [YCN*15, HPRC20, MFH*21, DPW*20, HHC17, RCPW21, SW17], while text modality mainly relies on word clouds [MCZ*17, PCN*19, JTH*21] (Figure 8b). While these are usually accessed through separated views or popups [YCN*15, ZHP*17, ZDXR20, PDD*22, MCZ*17, NHP*18] triggered by users during the exploration and inspection of the network, some VA systems use them as the core element of their interface, building more complex environments[LSL*17, HPRC20, DPW*20, MCZ*17].



(a) *BLUFF [DPW*20]* **(b)** *RNNVis [MCZ*17]*

Figure 8: *Examples of how VA systems support learned features. (a) An example of abstractions that helps users to understand how the model misclassifies a giant panda (green) as an armadillo (blue) when attacked. The BLUFF system [DPW*20] abstracts the network structure by highlighting only the neurons and their connections that change their behaviour significantly after the attack. Users can hover over a neuron to visualize the learned feature associated with it, in terms of dataset samples and generated images that maximally activate it. (b) RNNVis [MCZ*17] helps users to interpret what and where information is captured by hidden states in RNNs for text data by using a co-cluster layout. It clusters both hidden states of the given layer and the recognized words and then represents them as memory chips and word clouds, respectively. When the user clicks on a memory chip, the edges of its highly correlated word clusters are highlighted, showing what information is captured. When the user clicks on a word, the system visualizes the model's expected response as a heat map in the memory chips.*

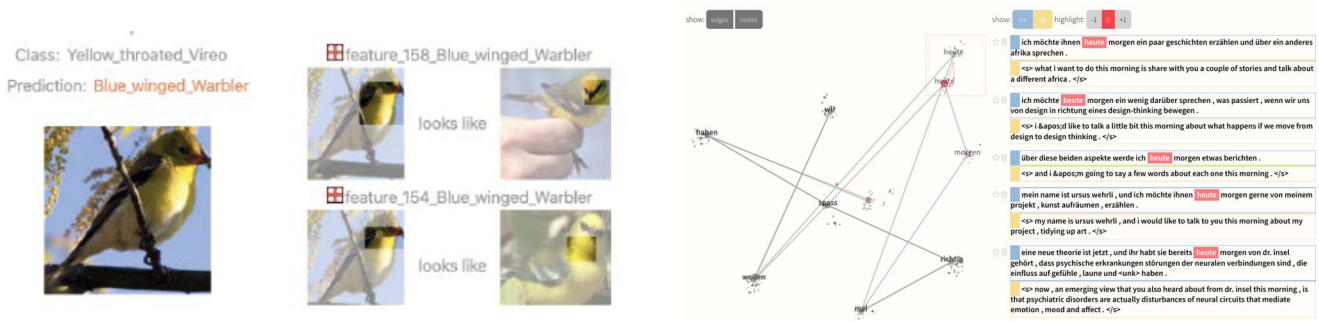
Systems that support users in the analysis of how low-level features (e.g. individual pixels of an image) are aggregated into high-level features (e.g. the pixels that form the 'cat' concepts) can be placed into the latter category since they use learned features methods as the main focus of the system. In this context, they have to deal with several challenges, such as the fact that deep networks have numerous layers, and each layer contains thousands of neurons; thus, the visualization and analysis of learned features for all of them simultaneously is a challenging task to accomplish.

A common solution is to provide an abstraction as an overview that summarizes the concepts learned by groups of neurons or layers and then let the user access more details-on-demand. This solution aims to facilitate user exploration by lightening the cognitive burden needed to explore the full spectrum of the network. The crucial step, in this case, is to provide an abstraction useful for the user task. The literature proposes several summarization techniques by using aggregation or average of the activations [HPRC20, JTH*21], Clustering [ZXZ*17, PDD*22], a combination of rectangle packing algorithms and hierarchical clustering [LSL*17], and pathways extracted based on neurons activations or importance [DPW*20]. These techniques are usually based on the similarity between learned features or activations. The results can be then compactly represented in enhanced Sankey diagrams [HPRC20, DPW*20], segmented DAG visualizations [LLS*18], graphs [RCPW21] or scatter plots using dimensionality reduction techniques, such as t-SNE[vdMH08] and its variants [PHL*16], where neurons that recognize similar concepts are embedded closely.

Usually, users can explore these visualizations in depth through zoom [HPRC20, PDD*22], details-on-demand [HPRC20, DPW*20, LLS*18] and filtering [HPRC20, PDD*22]. When the abstraction is provided through average or aggregation, other useful capabilities are letting the user change the clusters [LSL*17, ZXZ*17], analysing them [LSL*17], and switching between facets [LSL*17, HPRC20]. When the visualization is a graph, highlighting edges that flow in and out of a selected neuron makes it easier to understand how features are aggregated layer by layer [HPRC20, PDD*22].

Illustrative example. One evaluation scenario of RNNVIs (Figure 8b) considers sentiment analysis with a single-layered GRU with 50 cell states. The expert involved in the evaluation process used its co-clustering visualization to detect that two word clouds of two hidden unit clusters correspond to different sentiments. This shows that different parts of the network focus on different kind of words whose semantics is clear and understandable by an expert.

Summary. Overall, learned feature methods tend to be used as a secondary tool for inspecting and validating the knowledge learned by the networks. Thus, they rely on well-established techniques for visualization and analysis. The systems that exhibit the most novel designs are the ones that extract the activation pathways along with the network, where several Clustering algorithms and representations have been proposed [LSL*17, HPRC20, DPW*20]. Almost all the analysed systems assume a trained network as input, while only a couple of them [BJY*18, ZXZ*17] allow the user to inspect how the learned knowledge unfolds during the training process. This gap represents a promising direction for further development.



(a) *Melody [CBN*20]* **(b)** *Seq2Seq-Vis [SGB*19]*

Figure 9: *Examples of how VA systems support explanations by examples. (a) The Melody system [CBN*20] visualizes explanations by examples for the current image by highlighting regions similar to the most important one of the input. In the case shown, the model predicted a blue-winged warbler since its peck and neck are similar to the features of other blue-winged warblers. (b) Explanations by examples for text data. The task is to help users to understand machine translations of Seq2Seq models. The system visualizes the hidden states of the input sequence as a trajectory, where dots in the plane represent similar states. The user can select vertices on the graph, and then the system will show a list of sentences that produces similar states to the selected ones as explanations by example. The colours (blue and yellow) indicate whether the similar states come from the encoder or decoder, while the word that produced the similar state is highlighted in red.*

4.2.3. Explanations by examples

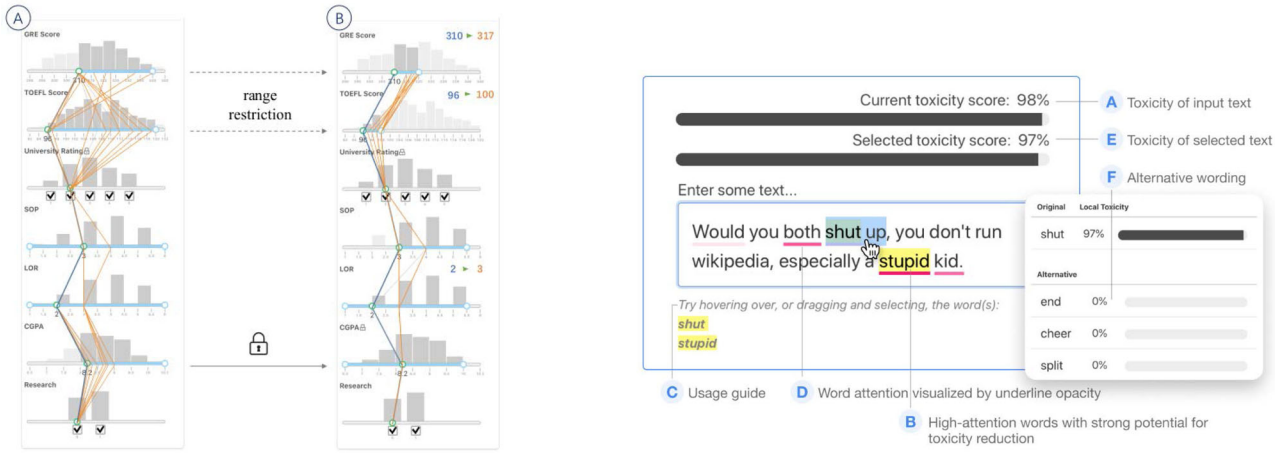
Explanations by examples methods extract and use training samples as explanations. The idea is to expose such samples and let the user extract the features that lead the model to create the association between the current input and the prediction.

The idea of many VA systems concerned with this aspect is to use samples from training data as a proxy to understand better the decision made on the input [SGB*19] or use them to estimate the meaning of latent vectors by looking at samples that produce similar ones [SGB*19, HSG20, SGPR18]. The latter facilitates error identification and training adjustments [SGB*19] and allows users to better assess the representativeness of a prototype representation [MXC*20] in self-explainable DNNs based on prototypes, thus easing the task of improving the model design [MXC*20]. Additionally, by providing explanations by examples for multiple layers simultaneously, users can analyse the sequence of explanations layer by layer and extract insights about the behaviour of the system [HSG20, SGB*19]. For example, users can compare two similar inputs predicted in different classes and check when (i.e. in which layer) and how (i.e. what is the difference) their representations diverged.

Usually, systems that employ post hoc approaches exploit information retrieval (IR) techniques to extract explanations by examples and achieve the previous goals. These techniques can be based on the latent representation [SGB*19, HSG20, HLvB*20], attention patterns [JTH*21] or feature patterns [HLvB*20]. Most algorithms include a hard-coded threshold to limit the number of elements to visualize, thus reducing the cognitive load and the visual clutter when depicted. Thresholds are usually based on the distance measured using popular metrics, such as L_2 norm or cosine distance [JTH*21]. A particular case concerns systems that support data sequence, where the models generate several latent vectors (one for each step of the sequence): the IR algorithms must search for patterns of latent vectors instead of a single latent vector. This change is not trivial since there are cases where the representation of two different inputs can be misaligned, for example, when data include sequences of various lengths. In these cases, a solution is to use Dynamic Time Warping (DTW) algorithms [SC07, SC78] to align sub-representations and then use a standard distance across them [HLvB*20].

The visualization of explanations by examples, usually represented as a list of inputs, can be enriched by attaching additional information or other explanations [CBN*20] (e.g. feature attributions). Examples of information that enrich the visualization are the summary of the common features between the neighbours and the current input [HSG20, CBN*20, SGPR18, CBN*20] (Figure 9a), the predictions [CBN*20], the similarity score or other metadata [HLvB*20, SGPR18]. Valuable functionalities connected to this category highlight the common features between the input and the neighbour selected by the user [SGPR18] and sorting mechanisms. These functionalities can be used together with explanations by examples to conduct What-if analysis. Indeed, some systems for natural language processing [SGB*19] use the similarity between the current word latent representations and the latent representation of the explanation by examples to suggest edits to the user. In the case of sequential data, the sequences can be compactly visualized either as a node-link diagram, using layouts based on dimensionality reduction techniques [SGB*19] (Figure 9b), or by linking the sequence to PCPs and visualizing the neighbours as lists. Moreover, providing more details-on-demand is crucial for this type of system, especially through comparison modes that allow users to analyse commonalities and differences between selected samples and an explanation by examples [HLvB*20].

Illustrative example. One use case of Melody (Figure 9a) is to understand a DL model for image classification. When the expert involved in the experiment has removed clusters too small or with too low explanation values, he has discovered three broad groups of birds with similar prediction logics but different visual explanatory features. By selecting one instance cluster, he has understood how the network classifies some birds by first looking at



(a) *DECE [CMQ21]* **(b)** *Recast [WSP*21]*

Figure 10: *Examples of how VA systems support Counterfactuals. (a) DECE [CMQ21] employs an enhanced parallel coordinate view to let the user analyse counterfactuals (orange) for tabular data inputs (blue). The histograms show the distribution of the feature in the dataset. The user can interact and customize counterfactuals by changing the value of the input features, setting constraints that counterfactuals should satisfy, or setting their number. (b) Recast [WSP*21] exploits counterfactuals to help user to lower the toxicity score of their text. When the user hovers over a word or a part of the sentence, the systems show counterfactuals in terms of words that the user should replace to obtain a different score. A tooltip shows the list of possible replacements, while the bars give the user hints about how the prediction changes. The user can click on any suggestion to update the input accordingly.*

coarse-level features (colour) and then more detailed ones (head or belly). He has also identified wrongly classified birds and has observed that it is because they share similar features to other classes. This shows how the application can help the expert to discover some parts of the reasoning process of the network.

Summary. Despite its ease of implementation and use, this category of explanations is not widely adopted yet and is mainly used as a complementary tool for other types of explanations. The only exceptions are VA systems that support self-explainable DNNs based on prototypes, where explanations by examples are used to guide the user to specify alternative prototypes or modify them when they cannot satisfactorily represent their nearby samples [MXC*20]. While in almost all the other cases, explanations by examples are valid only for the current input (i.e. they are local), in these cases, they can be used to get insights into the global behaviour captured by the model. They can also be exploited to improve the design of the model themselves by letting the user to specify the nearby desired explanation by examples. Then the system generates a new prototype that includes the desired samples as neighbours [MXC*20]. We do not observe noticeable complex novel visual solutions for this category, and most of the effort is directed towards the post hoc algorithms needed to extract them. The customization is very limited, often only to the sorting mechanism and rarely to the selection of the number of explanations to visualize [JTH*21].

4.2.4. Counterfactual examples

Counterfactual examples [WMR17, WPB*19] correspond to samples different but close to the current input and associated with a different prediction. VA systems use them to support users in verifying and refining the hypotheses about the decision process on selected data instances [CMQ21, SGPR18], in applying edits to input to obtain a different decision [WPB*19, WSP*21, LLL*19, WM20] or in exploring alternative scenarios on reinforcement learning agents [MSHB22]. These explanations can be employed when the DL model is deployed and static, and the user wants to understand the change one has to make to the input sample to obtain a modified output. Application scenario examples are loan applications and toxicity detection on text [WSP*21] (Figure 10b). Given the few works about self-explainable DNNs that support counterfactuals and their scope, all the analysed VA systems use post hoc local approaches.

Counterfactuals can be selected from the dataset [WPB*19, CBN*20, WM20, SGPR18], based on the distance between latent representations [SGPR18], input features [WPB*19] or feature attributions [CBN*20]. They can also be generated [CMQ21, MSHB22, WSP*21] by exploiting XDL algorithms using perturbations [CMQ21, WSP*21], synonyms [WSP*21] or generative models [WSP*21]. In most cases, the configurations about counterfactuals are fixed by design [MSHB22, WSP*21, LLL*19, CBN*20]. However, some systems that select counterfactuals from the dataset allow the user to choose between preselected distances [WPB*19]. Systems that generate them can allow the user to specify the number of counterfactuals, the number of features that are allowed to change, or which features can be changed [CMQ21].

The most common visual solutions to summarize the number of counterfactuals found [CMQ21, WM20] are enhanced bar charts [CMQ21], enhanced tables (e.g. Table Lens[RC94]) or Sankey diagrams [CBN*20]. These can be used to show how changing the input features affects the distribution of predictions and

counterfactuals, especially for systems that support users in verifying and refining the user hypotheses. Conversely, when the focus is on individual instances, it is important to highlight both the differences between the input and the counterfactuals (Figure 10a), and the difference in terms of predictions [WSP*21, SGPR18], for example, by using tables [WPB*19, CMQ21], enhanced representation of the input [CBN*20] (e.g. heatmaps for images) or colour coding. [WPB*19]. Finally, VA systems can sort and filter the explanations by applying hard-coded thresholds [WM20, SGPR18] or leave the task of adjusting the visualization to the user through sorting options or lasso selections [CMQ21, LLL*19].

Illustrative example. One evaluation scenario of DECE (Figure 10a) considers graduate admissions where a student wants to know how to improve her chance of being admitted to a school when a classifier predicted his rejection. Thanks to the interaction (e.g. lock of ratings that cannot be changed) with DECE, he has detected that a better 'GRE' or 'TOEFL' score would increase his chance. He has also identified some minimum boundaries for some scores. The acquired knowledge would help him to focus on which lesson to improve his grades.

Summary. Overall, only one system uses counterfactuals as the primary tool for explainability [CMQ21], while the others use them as a complementary approach, thus being supported similarly to the explanations by examples category (Section 4.2.3). It is also worth noting that half of the analysed papers supporting this category also support explanation by examples; it suggests that these two explanations can be complementary even if they aim at different goals. Finally, their adoption seems quite limited, and we do not observe noticeable novel visual solutions for this category since most of the effort is directed towards the algorithms needed to extract them rather than their representation.

4.2.5. Model behaviour

Systems that support the *model behaviour* category employ techniques to extract patterns from the model's inputs, outputs or internals, and link them to specific behaviours of the model through VA. These techniques aim to extract global explanations about the model and make its behaviour more predictable. They combine pattern mining on activations, human-in-the-loop (i.e. interactions with the user) techniques, and often methods of the previous categories.

The resulting explanations can be used to explore the role of different layers [BDME20], visualize the logical process of a model [ZZM16, PYN*21, WGSY19, JVW20, WHJ*22], extract decision rules for neurons or layers [JLL*19], analyse the cause of error patterns [JWW*22], or formulate and refine hypotheses about the semantics associated with the latent spaces [SWJ*20, WZY*22, WZY20, SGPR18].

When the goal is to approximate the logical process followed by layers or the entire model, VA systems employ algorithms [HPYM04] that record and aggregate activation or feature attribution across data and layers [WSP*21, WM20] into *patterns*, and then provide them directly to the user. These can be represented as sequences of lists of neurons and their associated semantics [PYN*21], as tables [WHJ*22], as sets of partial dependency plots [WM20], or summarized into decision trees, which can be represented as icicle plots [BDME20] or novel visualizations, like TreeFlow [JLL*19] (Figure 11a). Views including these representations are usually linked to scatter plots that show the extent to which a decision rule holds and open the door to deeper analysis [WZY*22]. Since these summary decision trees can be very deep or wide, several VA systems employ summarization techniques, such as automatic tree cutting [JLL*19], and allow users to expand or shrink the visualizations to hide or get more details [JLL*19].

On the other side, when the goal is to explore the latent space and identify its associated semantics, the systems provide more tools for the user to discover the patterns. This can be done through *Interactive Input Observations* and *Interactive Model Observations* analytics (Section 5.3.3). A popular solution, in this case, is to link enhanced PCPs, representing the latent space of a set of input, to other views that show input data [WZY20, SGPR18], and then make it possible for the user to modify either the range of activations or the input data [WZY20, SGPR18]. In this way, the user can inspect and extract patterns associated with categories of inputs or concepts associated with latent dimensions. Some systems [SWJ*20, BJY*18, KAKC18] can be placed in between those solutions (i.e. the ones that approximate the logical process and the ones that explore the latent space) since they ease the detection of patterns in the model behaviour through sub-sets' analysis, sub-sets definition and sorting algorithms but without directly extracting the patterns.

It is worth mentioning the case when the model is a reinforcement learning agent. Here, systems use patterns to reconstruct the policy the agent is following [ZZM16, WZY*22, WGSY19]. This task is not achievable using only one of the categories mentioned above (e.g. feature attribution) since the policy aggregates information from previous experience and often makes decisions based on future outcomes. Instead, since reinforcement learning involves sequences of actions, VA systems combine segment clustering, pattern mining and algorithms to align the sequences (e.g. DTW), to extract common patterns [WGSY19] and achieve the goal. Then, they visualize the extracted information by relying upon linked views that combine feature attributions, line charts [WGSY19] depicting sequences of actions, dendrograms representing the segments clusters [WGSY19] and plots summarizing statistics or activations [ZZM16]. In this way, they simultaneously provide different points of view to the user about the agent's behaviour in that situation (Figure 11b). In these cases, semantic zoom [WZY*22], lasso selection [WZY*22] and other similar functionalities that allow users to filter and highlight information are extremely useful. Linked views that combine dimensionality reduction techniques depicting different points of view of the same data and functionalities, such as lasso selection or semantic zoom [WZY*22, JWW*22, JVW20], are also popular among the systems that aim at exposing the cause of error patterns [JWW*22]. Moreover, like in the case of PCPs, some systems allow specifying patterns in terms of actions, rewards or input features, to ease the analysis [WGSY19], and they are often associated with feature attribution methods that highlight the most important features for each step [WGSY19, JVW20].

Illustrative example. One use case of CNN2DT (Figure 11a) considers the interpretation of a surrogate decision tree that represents VGG16 [SZ14] trained on CIFAR10 [KH*09]. The analysis of the Semantic map projection view has shown that the visual semantics of different car parts is extracted by different neurons. The

(a) *CNN2DT [JLL*19]* **(b)** *DRLive [WZY*22]*

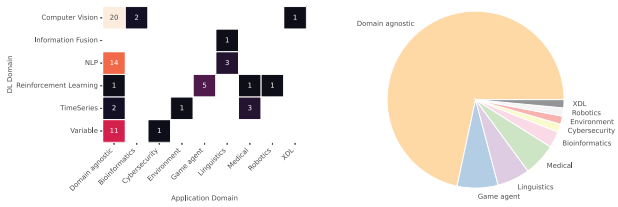
Figure 11: *Examples of how VA systems support model behaviour. (a) CNN2DT [JLL*19] extracts a surrogate decision tree to approximate the model behaviour of a CNN and combines it with learned features. The user can inspect the decision tree through the TreeFlow representation. The widths of brands correspond to the proportions of samples of each class, colours represent classes, and the label represents the decision rule. Users can navigate the tree by collapsing and expanding leave, and they can access the complexity of neuron semantics, computed by a learned feature method, by clicking on a tree node. (b) DRLive [WZY*22] helps the user to interpret the behaviour of a reinforcement learning agent by presenting data in five synchronized t-SNE scatter plots depicting the internals of the model, actions and inputs. Each point represents one game step and is coloured based on the action of that step. The user can click on the bars to highlight and analyse consecutive game steps or select a cluster of points to get more details in the form of game replays that show the average screen for that steps.*

analysis of TreeFlow confirms that most nodes of the decision tree use coherent semantic information to classify the samples. More interaction and filtering have allowed the user to understand the reason for true positives and false positive decisions by investigating the patterns searched by some neurons.

Summary. Overall, we observe a high heterogeneity in the adopted solutions both from a visualization perspective and supported analytics. Moreover, we note that the model behaviour category is often used as a support in the most challenging cases when other post hoc explanation methods alone are not applicable or not mature enough. This is the case, for example, of reinforcement learning agents or latent space interpretation for DNNs dealing with data sequences. Finally, as previously mentioned, this is the category where the VA solutions expose their full power to assist the user with tasks much harder to address without a system of this type.

5. Papers Analysis

This section analyses the general concepts behind the solutions described in Section 4, using the taxonomical scheme presented in Figure 4 and adopting a VA-oriented focus. It analyses supported *application and DL domains* (Section 5.1), *target users* (Section 5.2), *common patterns* in VA implementation (Section 5.3), *reproducibility and availability* (Section 5.4), *evaluation* of solutions (Section 5.5) and *temporal trends* (Section 5.6). We present a general description and refer to a few papers, selected as illustrative examples. The findings of this section set the stage for identifying open challenges (Section 6) and future actionable activities (Section 7). We compactly summarize the provided categorization and the characteristics of the papers in Tables 1 and 2.



(a) **(b)**

Figure 12: *(a) Number of papers at the intersection between the DL domain and the application domain. Each paper targets one or more DL domains and one application domain. Domain agnostic refers to research papers targeting general applications. (b) Distribution of the number of papers targeting a specific application domain.*

5.1. Application and deep learning domains

This section analyses the domains for which VA systems have been designed and provides a hint about their adoption. Each system is designed both for a *DL-domain* and an *application domain*. The term *DL-domain* indicates the DL area of research involved in the system (e.g. vision domain and reinforcement learning). In contrast, the *application domain* refers to real-world applications (e.g. medical applications and games). We use the category *application domain agnostic* to group systems tested only on research applications with no real-world case and where the user is typically a DL expert. Figure 12a suggests that most works are related to the computer vision DL-domain and are application domain agnostic.

Among the 48 *application domain agnostic* papers, some are generic enough to potentially target a large set of application

domains [KAKC18]. Others are limited to specific data types and DL methods [LSL*17]. A large number of papers suggests that these VA systems mainly target the DL community to help researchers understand their models (Figure 12b). For those targeting specific tasks of specific application domains, the three most popular application domains are *games*, *linguistics* and *medical domains*. The *games* domain includes five papers. They propose interfaces to visualize specific player actions [ZZM16, WGSY19] and extract their learned strategies. While several components are general enough to be applied to different games, some encoding and visual components remain tightly linked to their specific game of interest [JVW20]. The popularity of this category can be explained by the recent rise of reinforcement learning techniques for games. The *linguistics* domain contains four papers using visual elements similar to the tools for linguistics. For example, the visualization techniques employed by Recast [WSP*21] (Figure 10b) are similar to spellcheckers, thus making the usage by the application domain experts easier. In the *medical* domain (four papers), VA solutions help clinicians to benefit from the high performance of the DL models and, at the same time, verify that their behaviour is correct, a crucial task given the impact of their decisions. For example, given the tasks' peculiar characteristics, these systems often employ views specific to the task and are hardly generalizable to other tasks of the same application domain.

Finally, a few other VA systems are directed towards *XDL*, *Environment*, *Cybersecurity* and *Bioinformatics* domains. Among them, it is worth mentioning: VisLRPDesigner [HJZ*21] (Figure 7b), the work of Shen *et al.* [SWJ*20], and the works that focus on *Bioinformatics*. The first allows XDL experts to configure their methods [BBM*15]. The second proposes glyphs that mix information from neurons and sensors locations. The works focusing on Bioinformatics employ views that are hard to reuse in other contexts since they are specific to their problems.

Considering the categorization introduced in Section 4, we found that 16 of the 19 systems targeting a specific application domain rely on *feature attribution*, eight on *model behaviour*, two on *counterfactual*, two on *explanation by example* and none on *learned features* methods. This analysis suggests that the learned features are a type of explanation directed mainly to DL research and are less effective when the user is a domain expert with limited knowledge of ML. It is worth noting that these systems represent a fraction of the surveyed systems and the results need future confirmation and investigation in adapting them to different application domains. We argue for more applicative research efforts to test the efficacy of domain-agnostic solutions when applied to specific domains.

5.2. Users

In this section, we analyse the surveyed papers for different kinds of users. This paper adopts the taxonomy defined by Strobelt *et al.*[SGPR18] to classify the target users of a VA system, grouping them into *architects*, *trainers* and *end users*.

- *Architects* are DL experts that develop new DL components or architectures and modify the existing ones for application in new domains.
- *Trainers* have a background in DL; their task is not to develop novel architectures but to apply the existing ones to new application domains. They apply well-known recipes for various tasks of their application domain, limiting the modifications to hyperparameters and data. They are also named *practitioners* in the literature.
- *End users* have limited or no DL knowledge and use pretrained models in their specific application domain. Examples of this category are clinicians, domain experts and the public.

The categorization is not mutually exclusive: a system designed for end users is understandable for trainers and architects. However, the opposite is false, and a usable system for architects is hardly understandable for trainers and end users. Consequently, each paper is assigned to the less expert category of users that can use the system and fully understand it. When the paper claims the intended target users, the lowest ones are associated with it. Otherwise, the category is chosen based on the type of users involved in the evaluation or the one closest to the general system description.

Architects are the target users of 27 papers, which mainly rely on *feature attribution* (13), *learned features* (11) and *model behaviour* (10).

Trainers are the target users of 30 systems; again, they mainly rely on *feature attribution* (17), *learned features* (10) and *model behaviour* (8).

End users are the smallest group of target users, targeted by only 10 papers; they mainly rely on *feature attribution* (10), with two also coping with *counterfactuals*. No paper is related to *model behaviour* and *learned features*, stressing the insight that these types of explanations are specific to DL experts. How to bring these types of explanations to the end users is an open research topic.

Explanation category coverage. Relating these data to the proposed categorization (Figure 13), we note the following insights.

- Among the 13 collected contributions for *feature attribution* that have images as the data type, only one targets end users [vdBCR*20]. It supports them with simple visualizations (e.g. heatmaps, images, area and line charts). The other papers are split into *architects* (7) and *trainers* (4). For the text data type, instead, while *trainers* are the most targeted users (7), the solutions for *architects* (4) present the highest number of visual environments and richness (e.g. many visualizations per environment are present). Interestingly, three solutions target *end users*, with ProtoSteer [MXC*20] as an example of a medium-complex VA system that allows end users to steer the DL model, which is an unusual task for end users and more often provided to *architects*. Even more interestingly, when the data type is time series, only one solution targets *architects* [SMM*19]; *end users* are the most prominent class (5) with a strong presence in the health care domain (4).
- The *learned features* category does not present any work targeting *end users*. ShapeShop [HHC17] is the closest solution for supporting them, using a plain environment composed of just classic heatmaps and node-link diagrams. *Trainers* (10) and *architects* (11) are the most prominent targets.

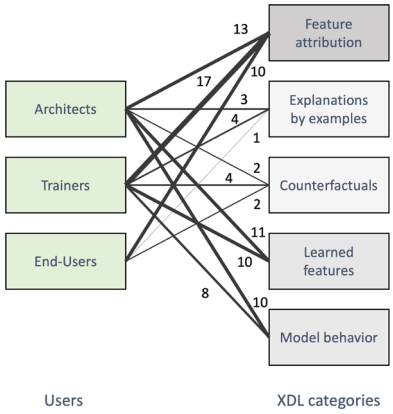


Figure 13: *Visual summary of user support, where the edge width is proportional to the number of users, and the opacity of XDL categories is proportional to the total number of contributions. Feature attribution presents the most varied support, but it strongly depends on the data type. Learned features and model behaviour do not present any support to end users, while evenly supporting the other ones. The remaining categories present a slight skew towards trainers and architects but for a much lower number of total contributions.*

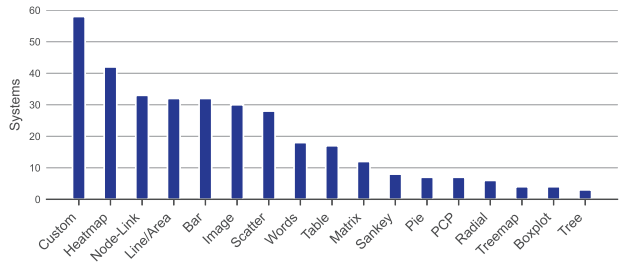


Figure 14: *Distribution of general families of visualization techniques adopted by the systems. While most works use custom visualization solutions, heatmaps and node-link diagrams appear to be the second most commonly used techniques. Few systems use treemaps, boxplots and tree visualizations, such as icicle plots.*

■ The same result holds for the *explanations-by-example* category, where *end users* are slightly targeted (1) and the solutions are split among *trainers* and *architects*. Only exBERT [HSG20] and LSTMVis [SGPR18] provide more advanced environments, focusing on custom word sequence visualizations and targeting the *trainers*.

■ Contributions in the *counterfactuals* category present two solutions for *end users*, four for *trainers* and two for *architects*. Among them, only DECE [CMQ21] and PolicyExplainer [MSHB22] present custom visual solutions for *trainers* and *end users*, respectively. Among the two works targeting *architects*, NLIZE [LLL*19] proposes a rich visual environment composed of novel visualization techniques.

■ As expected for *model behaviour*, no solution exists that targets end users. More surprisingly, the collected 18 solutions are split almost evenly among *architects* (10) and *trainers* (8). Focusing on the former, they present a high level of custom visual solutions, while the latter use more classic visualization techniques for supporting trainers.

In summary, most VA systems target trainers or architects (85% of the papers) since they need to understand what happens with their system and update them accordingly to the identified errors. Works targeting end users are less numerous (15%). The explanation can be twofold: they target either simple systems with an educational focus or specific systems that require strong application domain knowledge to be designed (e.g. medical domain). In the latter case, the domain knowledge is acquired by exchanges and the participation of application domain experts, which is costly. Thus, this impacts the speed and the number of proposals.

5.3. Visual analytics implementation

This section illustrates how the surveyed solutions implemented the VA principles. The first part focuses on the usage of *visualization techniques* (Section 5.3.1), the second part focuses on the degree of *analytical support* provided by the VA solutions (Section 5.3.2) and the final part illustrates the interactive workflow capabilities they provide (Section 5.3.3), completing the VA cycle.

5.3.1. *Visualization*

Visualization techniques usage. The VA systems considered in this survey use different visualization techniques, which are reported in Table 2 and summarized in Figure 14. Looking at the global usage of visual encodings in the surveyed solutions (Figure 14), *charts for numeric data* can handle input data, activations, output data, network weights and evaluation metrics with different encodings. We report *line charts* as the most prominent examples for performance analysis over epochs [CGR*17], *stacked area charts* for action distribution [WGSY19], *histograms* for parameters [ZHP*17], *pie charts* for the number of actions [WGSY19], *bar plots* for feature visualization [KCK*19] and *heatmaps* [HJZ*21]) for feature attributions. *Charts for multi-dimensional data* mostly concern input data and activation with *scatter plots* (e.g. for sample visualization [WGYS18]), *PCP* (e.g. for logits visualization [CGR*17]), *tables* (e.g. for experimental parameters [HLW*19]) or *images* for input data and generated features [HPRC20]). *Charts for hierarchical or relational data* handle data such as the model itself, the sample hierarchy or dependencies with different visual techniques: *node link diagrams* (e.g. for depicting proximity in clusters of samples [RCPW21]), *icicle plots* (e.g. to depict a decision tree [BDME20]), *treemaps* (e.g. for error distributions [MMD*19]), *chord diagrams* (*e.g.* to link words [WTC21]) or *Sankey diagrams* (e.g. for feature distribution over samples [JWW*22]). Finally, *charts for textual data* visualize textual information using *Word clouds* [MCZ*17] and *Word sequence* [HSG20]), possibly augmented with a *heatmap* [SGPR18].

Most papers complement these standard visualization techniques with custom or novel techniques, usually central in the relative VA system and related to the explanations. We report using custom/enhanced form visualizations for all the reported visualization

techniques. Unlike the trend of their classic usage, the novelty tends to be focused on visualizations for network data (Sankey diagrams [MCZ*17, PDD*22, DPW*20, HPRC20, LSL*17, PCN*19, SGB*19], node-link graphs [SWJ*20, WGYS18, WONM18, JKV*22, NHP*18, LLS*18], matrices [ZDXR20], hierarchical data (trees [DWB21, JLL*19, PHG*18] and word sequences [WTC21, CHS20, HSG20]) and map data, particularly for images (heatmaps and saliency maps [ZHP*17, JVW20, LLL*19, HCC*20, Vig19]). Some contributions also propose novel glyphs for summarizing data, such as *MultiRNNExplorer* [SWJ*20] for input data and activations, *Dodrio* [WTC21] to encode the behaviour of attention heads at specific layers, and *Attention Flows* [DWB21] (Figure 7a), which proposes a specific view that contains attention information computed at different words of input sentences.

Explanation category coverage. After looking at the general trend, we analyse the visualization usage for each category of explanation (Section 4.1). The full results are shown in Table 2. In conducting this activity, we modelled the *complexity* of a VA system as a combination of two properties: the number of visual environments (e.g. pages of a web-based system) of which the system is composed (quantitative factor) and the visual richness of each visual environment comprising visualizations, i.e. the visual sophistication of each environment (qualitative factor). The richness can take three possible values, low, medium or high, based on human assessment from two VA experts and one XDL expert. The complexity scale is qualitatively evaluated with five discrete values (low, medium/low, medium, medium/high and high). We assessed the system itself if it was available, and in its absence, we referred to supplemental video and figures of a paper representing the VA solution.

Feature attribution. The 40 solutions in this category use different visualization techniques depending on the supported data type: text, images, time series and multi-modal data.

The 15 solutions supporting the text data type exhibit environments of variable complexity, with most of them composed of a single environment, even if with variable richness (equal presence of low, medium and high). The majority use standard node-link diagrams as a common approach to represent attention-based information. Five solutions introduce custom designed visualizations: enhanced word sequences [MXC*20, WTC21, CHS20] and enhanced Sankey diagrams [SGB*19, PCN*19]. The use of dimensionality reduction visualizations is not common.

The 12 solutions supporting the image data type exhibit environments of medium average complexity, most composed of three or more environments with medium average richness. While all of them use heatmaps, node-link diagrams are used by a few [HSL*21, CBN*20, WGYS18]. None of them propose custom visualizations to support feature attribution, except for an enhanced bar plot [CBN*20] and a custom video augmentation [HCC*20]. Only five works use dimensionality reduction visualizations, all relying on t-SNE.

The nine solutions supporting time series exhibit environments of average complexity, most composed of three or more environments with medium to high richness. Line charts are widely used, with two custom approaches proposed [CWGvW19, MSHB22]. The use of dimensionality reduction visualizations is common, and most rely on t-SNE.

The four solutions supporting multi-modal data exhibit environments of high average complexity, half composed of a single environment [SSSE19, JKV*22] and the remaining composed of at least four environments [WHJ*22, HLW*19]. No custom visualization techniques emerged from the analysis.

In summary, the visual support for the feature attribution category seems highly dependent on the data type used, which influences most of the visualization techniques used and the number and richness of visual environments. All the proposed systems present a medium-to-high visual richness, even if they rely on classic techniques with limited novelty.

Learned features. The 22 solutions in this category exhibit environments of medium/low average complexity; one-third support the image data type. Among them, 14 solutions propose visualizations specifically designed as entry points for the analysis of learned features, and most of them propose novel or custom visualization designs in the form of enhanced Sankey diagrams [DPW*20, HPRC20, LSL*17, PDD*22, PCN*19], enhanced node-link diagrams [LLS*18, NHP*18] and an enhanced tree representation called TreeFlow [JLL*19]. Only systems that exclusively support the *learned features* category use dimensionality reduction visualizations, relying on different techniques (MDS, PCA, t-SNE and UMAP).

In summary, unlike the previous category, the VA systems for learned features show a high degree of novel visual solutions paired with lower average complexity (fewer visual environments and fewer visualizations per environment), customized for this specific explanation. We report a frequent usage of dimensionality reduction techniques.

Explanations by examples. The seven solutions in this category exhibit environments of medium/high average complexity using standard visualization techniques, such as heatmaps and node-link diagrams, while none uses dimensionality reduction visualizations. Among them, four solutions propose custom visualizations specifically designed for *explanations by examples*, such as enhanced word sequence views (e.g. [SGPR18, HSG20]).

Counterfactuals. The eight solutions in this category show a medium average complexity, most composed of a single environment, with the visualization of the input/output data as a common goal. Only NLIZE [LLL*19], DECE [CMQ21] and LSTMVis [SGPR18] propose custom visualization techniques explicitly designed for counterfactual analysis. Hypperster [WM20] is the only solution that uses a t-SNE visualization as the entry point of the analysis workflow.

Model behaviour. The 13 solutions in this category visualize the model internals as a common goal and show a medium average complexity. They can be divided into two groups: (a) four solutions that exclusively belong to the *model behaviour* category and (b) five solutions that also belong to the *feature attribution* category. Solutions in (a) rely on standard visualizations such as node-link diagrams and dimensionality reduction visualizations (t-SNE). The only exception is GNNLens [JWW*22], which proposes an enhanced visualization for the model internals, still based on t-SNE. Conversely, solutions in (b) exhibit environments of high complexity. Most of them rely on custom visualizations, such as

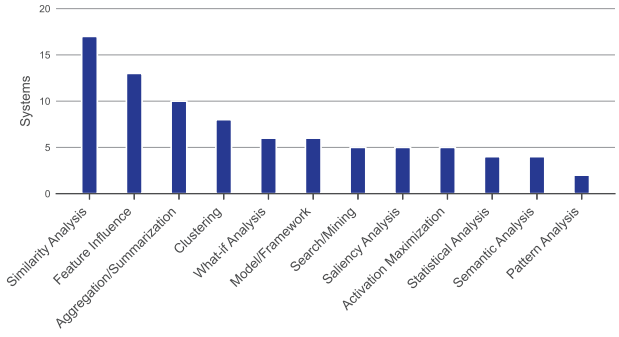


Figure 15: *Distribution of general families of complex analytics techniques used by the systems. Analytics belonging to the Similarity Analysis family are the most prominent ones, followed by Feature Influence, Aggregation/Summarization and Clustering families. The remaining families show a similar adoption, except for Pattern Analysis.*

enhanced Sankey diagrams [WHJ*22], enhanced node-link diagrams [LLS*18, NHP*18] or techniques aimed at addressing scalability problems, such as pixel-based or unitary visualizations (e.g. [JVW20, WHJ*22]). Most of the solutions use node-link diagrams or scatter plots as entry points for the analysis workflow, except for three [SGPR18, HLvB*20, WZY20], which use PCP.

Overall, the reported analysis shows the effect of the targeted XDL category on the design of the VA systems supporting it, predominantly in terms of using classic visual solutions instead of novel or custom ones, and it shows the average complexity of the visual sub-system and usage of dimensionality reduction techniques. Specifically, on this last topic, we note that only two solutions propose custom dimensionality reduction visualizations [WHJ*22, JWW*22], enriching the visualization with glyphs. Given the frequent usage of dimensionality reduction techniques for learned features and model behaviour and the presence of some specific techniques to adapt them to DL [RFFT17, CEP20, CP20], it would be interesting for future VA research to focus more on adapting them to XDL. We proceed by looking at what kinds of analytics those systems support.

5.3.2. *Analytics*

Analytics usage. The VA systems considered in this survey use different analytics techniques, reported in Table 2 and summarized in Figure 15. While almost every system implements basic analytics supporting data exploration and filtering, the use of more complex analytics depends on the supported tasks and the XDL category. We collected 89 instances of complex analytics, grouped into 12 families: *Similarity Analysis*, *Feature Influence*, *Aggregation/Summarization*, *Clustering*, *What-if Analysis*, *Model/Framework* (declared by authors and meaning a full stack composed of multiple analytics), *Search/Mining*, *Saliency Analysis*, *Activation Maximization*, *Statistical Analysis*, *Semantic Analysis* and *Pattern Analysis*. Complex analytics are divided into three categories, namely, *novel* analytics, *custom* analytics based on slight modifications of existing solutions and *classic* analytics, which are existing solutions used as is. While basic analytics provide no novelty by definition, for complex analytics, we based our analysis on the description of the analytics support in each paper and on the claims that each contribution stated for each analytics (usually resulting in a deep explanation of its behaviour and implementation).

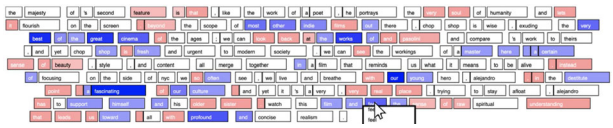


Figure 16: *The* Token-wise Sentiment Decomposition *[CHS20] analytics computes the contributions of each token in the input sentence towards the final classification class and represents them using an enhanced word sequence visualization.*

Explanation category support. After looking at the general trend, we analysed the analytics support for each category of explanation (Section 4.1). *Feature attribution*. The 40 solutions in this category use different analytics techniques depending on the supported data type: text, images, time series and multi-modal data. The 15 solutions supporting the text data type show a variable usage of analytics. Analytics belonging to the *Feature Influence* family are the most used, and among them, three novel solutions exist [CHS20, DWB21, WHJ*22], aiming at identifying the most influencing tokens for the predictions. For example, Chawla *et al.* [CHS20] propose the *Token-wise Sentiment Decomposition* analytics to compute the contributions of each token in the input sentence towards the final classification class and represented them using an enhanced word sequence visualization (Figure 16). At the same time, AttentionFlows [DWB21] presents a novel radial layout to represent feature attributions for text data in Transformers models. Five papers use analytics in the *Clustering* or *Aggregation/Summarization* family but without proposing novel approaches. Two systems support *What-if Analysis*, allowing users to change attention weights [SGB*19] or prune the attention heads [JKV*22].

The 12 solutions supporting the image data type mostly use common *Saliency Analysis* or *Activation Maximization* analytics. Notable are two solutions [WGYS18, CBN*20] that propose two novel frameworks oriented towards ML architects and based on node-link diagrams to interpret the model.

The nine solutions supporting time series data show a heterogeneous usage of the analytic families. Novel approaches are oriented towards *Feature Influence* analytics [SMM*19, SWJ*20, WWM20, MSHB22], paired in most cases with line charts or their visually enhanced versions (e.g. [MSHB22]).

The four solutions that support multi-modal data include *Feature Influence* and *What-if Analysis*. For example, M2Lens [WHJ*22] proposes a novel analytics that aims to identify the influences of features on model predictions.

In summary, similar to what we observed for visualizations, we note that the feature attribution category tends to use classic analytics to support the interactive analysis, where *Aggregation/Summarization* and *Clustering* are the analytics families most commonly used to support exploration. In contrast, *Feature*

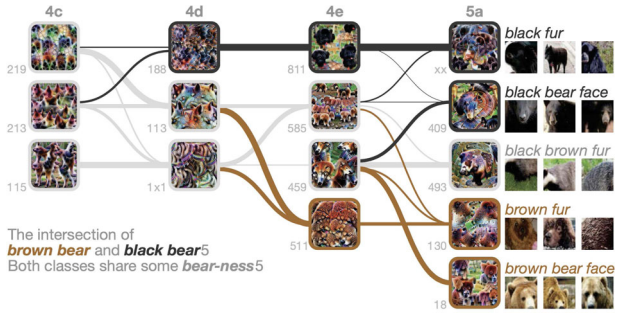


Figure 17: *SUMMIT [HPRC20] presents the* Attribution Graph View*, a VA approach that can reveal and summarize crucial neuron associations and sub-structures that contribute to the outcome of a model. The system presents an enhanced Sankey diagram based on the aggregation of activations and intralayer influences.*

Influence and *Activation Maximization* are mostly used to support the explanation. In addition, on the analytics side, the data type has much less influence on the choice of the analytics compared to its influence on the choice of visualization technique.

■ *Learned features*. The 22 solutions in this category make a heterogeneous use of complex analytics, covering 10 out of 12 analytics families. The prominent families are *Aggregation/Summarization* (6), *Clustering* (3), *Activation Maximization* (3), *Model/Framework* (3) and *Similarity Analysis* (3). *Pattern Analysis* and *What-if Analysis* are not supported. Although most of the solutions propose custom visualizations and use complex analytics, only SUMMIT [HPRC20] presents a novel visualization design coupled with a novel complex analytics. Among the solutions supporting *Aggregation/Summarization*, three analytics approaches are novel [BJY*18, HPRC20, ZDXR20]. SUMMIT [HPRC20] (Figure 17) presents both an enhanced Sankey diagram and a novel *Aggregation Analytics* approach to visualize highly activated neurons. The remaining novel solutions belong to *Model/Framework* [MFH*21, LYY*20] and *Similarity Analysis* [PHG*18, PDD*22]. An example *Activation Maximization* analytics is the one used by BLUFF [DPW*20] (Figure 8a), which extracts the learned features and presents synthetic images that maximally activate the neurons over an enhanced Sankey diagram.

In summary, the learned features category is confirmed to present a higher degree of novelty and solution coverage, even from an analytics aspect. On the other hand, note that the novelty in visualizations and analytics is rarely coupled, meaning that novel analytics results tend to be visually represented in a traditional way. In contrast, basic analytics tend to exploit novel visual solutions. The rationale could lay in the designers of these systems using visualization and analytics separately to support explainability. A second rationale could be to avoid overloading the user to understand novel analytics and novel visual encodings simultaneously. ■ *Explanations by examples*. Solutions in this category use only three families of complex analytics: *Similarity Analysis* (2), *Search/Mining* (1) and *What-if Analysis* (1). No completely novel analytics have been proposed, while all are customized forms of classic ones. The exBERT [HSG20] system is remarkable, which uses an enhanced word sequence visualization and a custom *Search/Mining* analytics to support the nearest neighbour search of tokens and attention heads for *explanations by examples*.

■ *Counterfactuals*. The solutions belonging to this category focus only on the *Search/Mining* (3) and *Feature Influence* (1) families. Among them, only DECE [CMQ21] presents a novel analytic approach. As shown in Figure 10a, the system presents a counterfactual generation method coupled with an enhanced parallel coordinate view to let the user analyse counterfactuals.

■ *Model behaviour*. Most of the 13 solutions in this category (8) rely on *Similarity Analysis*, where all systems use classic dimensionality reduction algorithms, with t-SNE as the most common method. For example, DRLive [WZY*22] (Figure 11b) helps the user interpret the behaviour of a reinforcement learning agent by presenting data in five synchronized t-SNE scatter plots depicting the internals of the model, actions and inputs. The second most common analytics approach is *Statistical Analysis* (2), which is oriented towards the model internals [WZY20] and the model memory [JVW20].

Overall, the analysis of analytics use in the surveyed system confirms a variety of approaches for feature attribution and learned features categories, with the first proposing more conservative approaches (but more present in the literature), while the second shows novel efforts. Explanations by example and counterfactuals categories present a more balanced usage of classic and novel analytics (both slight modifications or completely novel) but are flawed by the presence of fewer contributions. Finally, model behaviour surprisingly has a more conservative approach on the analytics side than expected. More research could be conducted to expand the capabilities of VA systems, given the natural tendency of this category to include human-in-the-loop approaches.

As a final remark, we note the scarce support for more than exploration capabilities, with only six solutions [HJZ*21, LYY*20, JKV*22, SGB*19, LLL*19, WZY*22] exploiting the knowledge generation model to support not only exploration tasks but also verification tasks. More on this will follow in the next section.

5.3.3. *Interactivity*

This section discusses the degree of visual interactivity the surveyed solutions offer to the targeted users (Figure 18). We obtain inspiration from the categorization of Gehrmann *et al.* [GSK*19], differentiating between three types of interactions: *Passive Observations*, *Interactive Input Observations* and *Interactive Model Observations*.

- *Passive Observations* include interactions that do not allow the editing of input data or models loaded into the system. Examples are the navigation across data, layers, explanations or the selection of dataset samples.
- *Interactive Input Observations* allow users to modify models' input data or create new ones on demand (e.g. through forms). An example is brushing over an image to delete some pixels and check if the model changes its predictions.
- *Interactive Model Observations* allow users to interact with the model by modifying, for example, activations or attention weights and checking how its behaviour changes (e.g. select neurons that must be shut down during the next iteration).

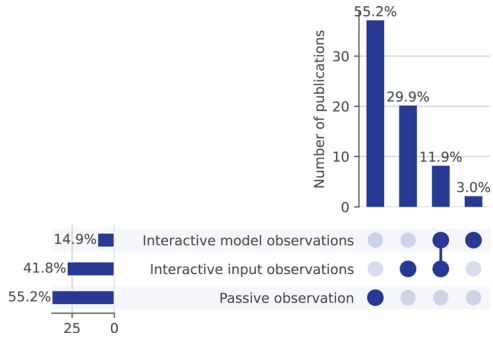


Figure 18: *Distribution of the type of interactivity. Systems can allow users to modify the input (Interactive Input Observation), the model's internals (Interactive Model Observation) or make them unmodifiable (Passive Observations).*

The 37 papers using *Passive Observations* rely on *feature attribution* (19) and *learned features* (17). No paper is related to *counterfactuals*, while two papers rely on *explanations by examples* and 11 rely on *model behaviour*. An example of a supported task is given by *Blocks* [BJY*18], which allows the analysis of prediction errors. It only needs the classification results and the sample data to make its representation, and it exploits the interactivity to mainly support exploration, summarization and filtering capabilities.

The 20 papers using *Interactive Input Observations* rely on *feature attribution* (13), *counterfactual* (6) and *model behaviour* (5). This type of interaction is a core interaction for counterfactuals and model behaviour. In the first case, the interaction allows users to perform counterfactual reasoning [WPB*19], investigate decision boundaries and analyse the incidence of general changes to samples. In the second case, it is an entry point used to verify the system behaviour in multiple cases and extract explanations.

Interactive Model Observations are supported by 10 papers overall, mainly for *feature attribution* (eight papers). The remaining papers cover the remaining XDL categories with only one or two samples per category (usually with more than one category covered per paper), apart from *learned features*. As in the previous case, this is a key component for systems that support the extraction of the model behaviour since it allows the user to verify the model behaviour for different configurations and steer it. For example, *ProtoSteer* [MXC*20] allows the user to update the prototypes during the training stage of a self-explainable deep sequence model. Surprisingly, this steering ability and the possibility of observing the training behaviour [PHG*18] in real time is barely present in the literature (column **TR** of Table 1) and is a direction for future research (e.g. [FCdMP21]).

In summary, the totality of the surveyed contributions implements *Passive Observations* by definition, resulting in exploration capabilities for precomputed data for all the five XDL categories except counterfactuals. Twenty works add VA capabilities for confirmatory analysis and hypothesis testing by acting on the input observations. Feature attribution, counterfactuals and model behaviour are the XDL categories benefiting the most from this approach, while future research contributions could target learned features and explanations by examples. Finally, *Interactive Model Observations* are mainly dedicated to just one XDL category (*feature attribution*) and present the highest degree of user control in the analysis workflow, including steering capability and what-if analysis. Supporting this level of control and allowing model steerability for the remaining XDL categories are promising research directions.

5.4. Reproducibility and availability

This section addresses the reproducibility and availability of a solution by analysing its capability to be exploited by researchers.

These aspects directly impact the ease of use of a solution, depending on different degrees of availability of materials related to a contribution. The lowest degree is represented by just the paper itself, followed by the availability of a demonstration environment, which helps experience the solution. Moving towards better availability, we find the usage of well-established implementation frameworks, source code availability and difficulty in using the provided code. This last consideration applies only to contributions that make source code available. We classified them into one of the three subcategories: easy to use (meaning that the material is easy to use as is), easy to reproduce results and easy to extend (meaning that the material can be customized to user needs).

Three members of our team investigated the ease of using, reproducing and extending the surveyed VA solutions by acting as a researcher needed to conduct these activities. Each grade was binary (i.e. easy/not easy). To limit the subjectivity of the evaluation, we formed each judgement by a majority vote. The evaluation concerned both the experimental setup and the analysis of documentation.

Forty-two out of 67 contributions provide only the textual paper without any supporting material. While two papers provide only a demonstration environment, 23 papers provide public source code, where eight only provide source code [KCK*19, WWM20, SGPR18, MCZ*17, MFH*21, LLL*19, YCN*15, SMM*19], and 15 only provide source code and demonstration environment [JVW20, SGB*19, JWW*22, HSG20, LYY*20, HPRC20, Vig19, SSSE19, ŠSE*21, WPB*19, JKV*22, RCPW21, WTC21, DPW*20, PDD*22]. In general, only approximately 36% of contributions provide something to support the interested researcher in testing the proposal, and only 33% provide source code.

For works providing source code, it is possible to examine the most commonly used implementation frameworks in the back-end and front-end. Eighteen out of 23 works use a back-end, where *PyTorch* (11) and *TensorFlow* (4) represent the predominant choices (others: 3). *Flask* is the most commonly used web server. On the other hand, 16 of 23 works use a dedicated front-end, while the remaining delegate the visualization management to the back-end part. Among the former, *D3.js* is the most commonly used framework (10), followed by other JavaScript-based environments (e.g. *React.js* and *vue.js*) with six occurrences. Interestingly, only one contribution uses computer graphics technology [SMM*19] (*Unity Game Engine*). Overall, where available, the solutions use standard and well-accepted technology for their implementations. Most use web-based technologies, whereas none seem to rely on native applications (e.g. *C++*, *OpenGL*) that could be preferred due to their

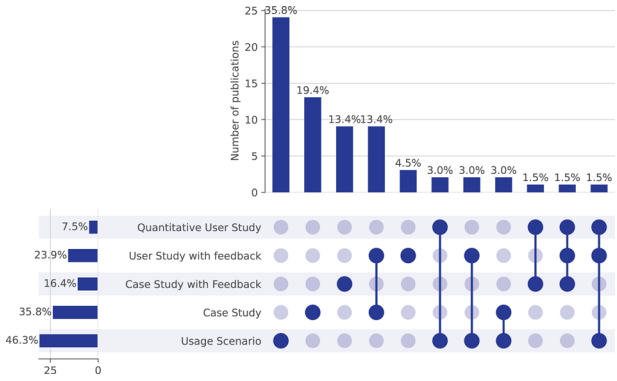


Figure 19: *Distribution of the evaluation procedure. The usage scenario is the most common approach used to evaluate VA systems, whereas few have conducted quantitative user studies.*

better performance and scalability. In the future, these technologies could be used more due to the increasing size of DNNs.

Looking at the ease of a researcher would encounter in exploiting those solutions, our tests show that four contributions are already considered difficult to use as intended by the authors due to the limited documentation [KCK*19, WWM20, SMM*19, MFH*21]. Seven are easy to use, and their results are easy to reproduce. The remaining 10 are easy to use, their results are easy to reproduce and their functionalities can be extended. Finally, two are considered easy to use, their results are easy to reproduce and they are very easy to extend due to their plugin nature [WPB*19, SSSE19].

Looking at the results for papers providing source code, approximately 80% allow easy reproducibility and use. This result is good since it means that researchers working in XDL or VA can easily exploit the capabilities of the solutions. At the same time, we must consider that no form of code or demonstration environment is provided for the remaining part of the papers (45).

5.5. Evaluation

Although some XDL methods in Section 4 have been individually evaluated (e.g. *features attribution* [AGM*18]), VA systems for XDL have to be evaluated as a whole to assert their efficacy and capability to be trusted by users. A global evaluation is essential because incorrect explanations influence humans to make bad decisions when teamed with an AI system [BWZ*21]. Since the field is new, there is not yet any well-accepted and evaluation pipeline in use. However, some authors [LGM20] propose mitigation in this direction by providing a list of questions that could be used as a checklist for evaluating a system.

Figure 19 shows that the VA solutions for XDL follow common trends in VA fields. Specifically, they mainly evaluate their systems using *user studies* with quantitative information or feedback, *case studies* with or without feedback and *usage scenarios*. Other than the modality chosen by the authors to evaluate a proposed solution, papers are systematically peer-reviewed and validated by expert reviewers.

Q-USt: A *Quantitative User Study* (five papers) involves participants recruited to interact with the system and answer a questionnaire [vdBCR*20, HJZ*21, DWSZ20, JKV*22, PDD*22]. The results are provided in a quantifiable way (e.g. the amount of time a system performs better than another).

F-USt: A *User Study With Feedback* (16 papers) is similar to a *Quantitative User Study* but only provides descriptive feedback in a qualitative form. Usually, the first part of the process consists of an interview to acquire the experience and expectations of users. Then the system is presented to them with the task they must solve. The focus is on the user, and the study can use a mix of realistic and synthetic data, with the constraint for the data to be fit for the task. Users must think aloud when solving their tasks, and another interview is conducted afterwards.

F-CSt: A *Case Study With Feedback* (11 papers) corresponds to a case study run by participants that use the tool in a controlled way for a specific task and where participants also provide feedback [vdBCR*20, WGZ*19, MXC*20, CWGvW19, JKV*22, HLW*19, JTH*21, WZY*22, WZY20, WGSY19, WHJ*22]. The focus is on the case under analysis, and the case study is normally run in real conditions using real data, with the goal of generalizing the results over cases in similar conditions.

CSt: A *Case Study* (24 papers) without feedback is also available in the literature. In this case, papers describe case studies with experts. The discoveries and workflows of the experts are reported in the results. Sometimes feedback can still be indirectly collected during case studies.

USc: *Usage Scenario* (31 papers) corresponds to the execution of a scenario by the authors without participants and/or the experts involved.

Looking at the distribution of evaluation types, almost 36% of papers propose only a *Usage Scenario* (24 papers), with additional five papers complementing it with an additional evaluation activities. Twenty-four works provide case studies, whereas only $\approx 8\%$ (five papers) include a *Quantitative User Study*, exclusively covering *feature attribution* and *learned features* categories. Overall the evaluation activities align with what is expected by VA best practices. At the same time, increasing the user's involvement in testing activities is recommended, particularly looking at DL experts and end users (e.g. application domain experts).

Although user involvement is essential, it is not the only resource to consider, and the correct evaluation methodology choice depends on the kind of problem or research question at hand [GB08]. Note that also alternative evaluation methods can also be used, even looking at practices in the XDL domain (Section 6.1).

5.6. Temporal trends

Temporal trends allow one to understand how methods are adopted and abandoned over time. Figure 20 shows when the 67 papers considered in this survey were published. While the years 2015 and 2016 show one paper each, since 2017, the frequency of publication has considerably increased, reaching a peak of 19 papers in 2020.

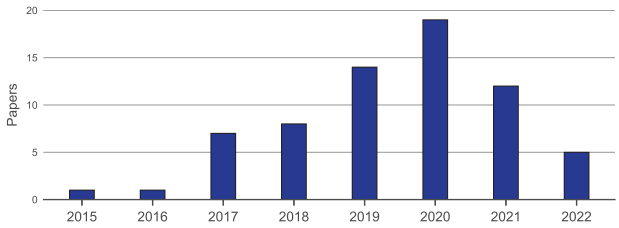


Figure 20: *Publication year for the 67 papers in this survey.*

The fact that 2015 and 2016 presented only two papers means that those years can be considered the starting epoch of VA for XDL.

Further analyses of temporal trends are summarized in Figure 21. Figure 21a shows how the VA systems employed methods in the XDL field during the years, according to the categorization presented in Section 4. *Feature attribution* is the most employed: approximately 50% of papers use them every year.

Concerning the coverage of DL models, as shown in Figure 21b, the most commonly used models belonged to the CNN class until 2018, after which their adoption started to decline. Conversely, transformers usage emerged during the last 4 years, appearing in 42% of the papers in 2021. This trend aligns with the AI research trends, where transformers are gradually replacing CNNs in computer vision and RNNs in natural language processing [KNH*22].

An interesting aspect to consider is the time elapsed between the first publication of a DL model and the publication of VA systems targeting it and proposing XDL approaches. The solution showing the shortest gap is BertViz [Vig19]: the authors proposed explainable approaches for transformers in 2019, the same year the relative DL models (BERT and GPT-2) were proposed. Conversely, in 2020, MultiRNNExplorer [SWJ*20] and HypperSteer [WM20] proposed explainable approaches for RNN models that were introduced 23 years before.

Figure 21d shows the distribution of the gaps in terms of years grouped by classes. While the median of the gaps is 4 years, with a lower quartile of 2 years and an upper quartile of 6 years, Figure 21c shows that models belonging to the RNN class are affected by the highest gaps. They show a median and lower quartile in line with the trend, 4 years and 2 years, respectively, but an upper quartile of 20 years. On the other hand, transformers show the lowest gaps, with a median of 2 years, suggesting increasing interest in the explainability of those models and, more generally, in the newly proposed models. As stated before, VA for XDL already emerged when transformers emerged in the DL community [KNH*22].

6. Research Challenges

This section lists some important challenges to address in future VA works for XDL.

6.1. Make the XDL systems more trustworthy thanks to VA

The adoption of any system is closely linked to the trust users have in it. We identified several directions systems designers should follow to improve the trustworthiness of their systems. First, we consider it necessary to improve systems evaluation procedures by considering their explanation performance.

In Section 5.5, we analysed the main trends in evaluating VA systems for XDL. However, these evaluations are rather generic for any VA system, and they do not consider the specificities of XDL that also have proper evaluation methods from the XDL community. Therefore, we think research is needed to properly define evaluation procedures dedicated to such systems by taking inspiration from both the VA and XDL communities. For example, Meske *et al.* [MBSG20] and Mohseni *et al.* [MZR21] described several quality criteria to quantify the effectiveness of explanations. They can be extended by considering VA aspects, such as the evolution over time and the interaction with the user. Additionally, building benchmarks that include well-defined data and tasks ready to be solved by VA systems can help create more easily comparable systems thanks to their standard evaluation procedure.

Building systems is not sufficient; it is necessary to enforce their trust. Since few works focus on this aspect, a large gap exists. One way to increase trust is to incorporate semantics within the explanations or use ad hoc methods [CMJ*20]. The preliminary work of Panigutti *et al.* [PPP20] uses ontologies that label the input data, but we expect future works to use input datasets with no ontology annotations. Poli *et al.* [POP21] presented a sentence generation system for image segmentation. Such an approach can be adapted to similar contexts relying on DL.

Most explainable systems generate visualizations to help understand the DL model. However, it is still up to the user to infer knowledge from this generated information. For example, it is common for

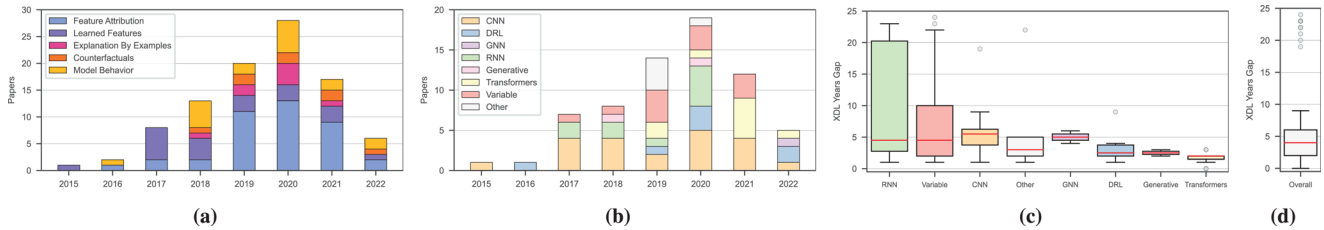


Figure 21: *Temporal trend analyses. (a) XDL method usage over the years. (b) Models usage over the years, grouped by class. (c) Distributions of elapsed years between the first publication of a DL model and the publication of VA systems proposing XDL approaches on that model. (d) Overall trend grouping all the models of the previous point.*

feature-attribution-like methods to generate a heatmap of features of interest and overlay it on the input image data (Section 4.2.1): the user can easily see which pixels of the image strongly take part in the final decision. However, there is no more information, and the user should infer the reasons for their importance without knowing why the network focused on this specific part. VA solutions can provide more guidance [CGM*17] and better support the user in insight generation and verification.

6.2. Make the explanations more versatile

Most VA systems target standard CNNs used in a classification context with few classes without necessarily using state-of-the-art XDL methods and without exploring the most challenging and recent problems, as also noted by [PvSvdE*22]. We think systems should be more versatile by focusing on a deeper variety of DL model families and by using more recent XDL methods.

Although there is a considerable diversity of DL architecture families studied in the VA and XDL literature, there is a substantial balancing issue among them. Indeed, while there are many systems about CNNs and RNNs, others have been barely studied, such as generative networks. It would be interesting, in the future, to focus on a broader family of models and input data, possibly by proposing more VA systems that are model-agnostic or data-agnostic.

Up to 2021, many works directed towards the VA community rarely include recent models and explanation methods, and works directed towards the XDL community often provide basic visualizations. For example, only five systems include recent and popular baselines, such as SHAP [LL17], Grad-CAM [SCD*19] and LIME [RSG16]. In all the other cases, systems prefer to use older methods (e.g. Deconvolution [ZF14], Vanilla Saliency Maps [SZ14]). The same observation holds for other categories of explanation methods (Section 2.2) and models used. While these choices have no impact on the quality of VA systems, they can limit their usefulness and spread, especially when directed towards DL experts.

6.3. Increased adoption of XDL through VA

End users need to be concerned about DL, as it can profoundly impact them. We think that XDL is the key to helping understand the outcomes of DL and that VA can make XDL entertaining, understandable and usable (Figure 22).

We observe that most VA systems target DL or application domain specialists, whereas explanations should also be understandable by non-experts since they are an important target [GTFA19]. In fact, if the public cannot understand the benefits or drawbacks of DL systems, they cannot trust them or make appropriate decisions based on the use of such systems. The main difficulty relies on finding a trade-off between the high complexity of DL models and what users can understand without a background in ML. In this regard, the combination of explanations and VA systems appears crucial for future research. Verbalization [SBE*18] and the generation of explanations with a balance between cognitive load and explanation accuracy [AvdWKL20] (adaptation) could be promising ways to broaden the audience of such tools.

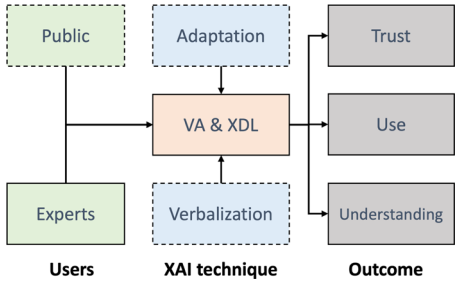


Figure 22: *VA systems for XDL should help the public better handle DL applications. The dotted rectangles represent unhandled aspects: public users, the adaptation of the explanation to the user and verbalization of the explanation.*

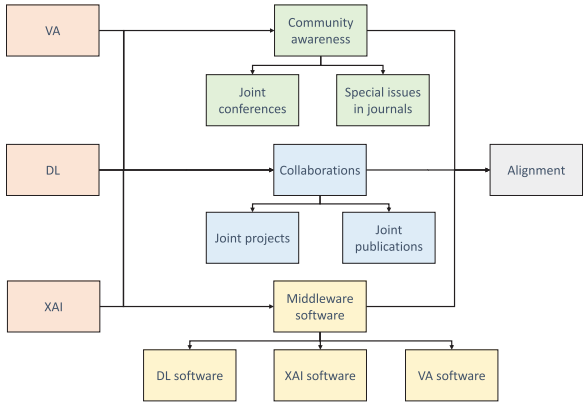


Figure 23: *Steps needed to bridge the gaps among XDL, VA and DL. The steps are related to the communities, their collaborations and the software.*

7. Bridging XAI, VA and DL

Section 6 has discussed research challenges at the intersection between VA, DL and XAI. Here, we identify a series of temporal action items that could help fill those gaps (Figure 23).

1. **Community awareness**. The first step is to increase awareness among the communities. To build complete and adequate systems, developers must be aware of the strengths and weaknesses of each involved field. In this direction, some VA and AI conferences have already hosted workshops that discuss the topic. Examples include, VISxAI [PBH*] and VADL 2017 [CYPL] for VA conferences, and XAI4Debugging [CFG*] and EDL-AI for XAI [BQ] for AI conferences. While these workshops are not all specific to the VA topic for explaining DL, they discuss it to some extent. The *distill.pub* journal was another initiative in this direction (i.e. currently in a hiatus). It promoted interactive peer-reviewed articles on ML, where users can visually interact with the models and findings of the papers. Many articles deal with XDL, thus making it clear that visual interaction is a key element to understand them better. Similar initiatives, especially when they involve experts from all the communities involved, are crucial to increase awareness and the exchange between the communities, helping them grow.

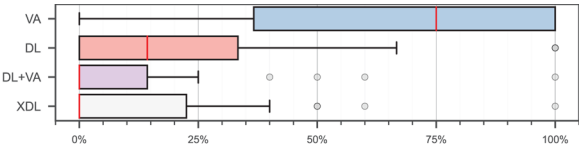


Figure 24: *Distribution of the areas of expertise the authors in VA, DL and XAI. VA expertise is pre-dominant, partly due to the provenance (e.g. venue) of most of the surveyed contributions.*

2. **Collaborations**. A more significant number of initiatives and deeper awareness would increase collaborations between the communities. We rated the authors' VA, XAI and DL expertise in the surveyed papers on a qualitative scale (two values: expert or not showing expertise for each area) by analysing their publication histories. The primary source was Google Scholar and we looked at the number of papers published in each area by an author. Two-thirds of the authors (170 over 234) of the analysed papers came from the VA community, while one-third are from the DL community. Among them, only a small fraction (23 authors) has expertise on XAI (i.e. has published at least two papers on the topic). This result is also confirmed by the distribution of authors for papers (Figure 24
24), where VA is the dominant area, less than half of papers usually involve at least one author from DL, and very few involve XAI experts. While most papers come from VA venues, the majority of the authors are expected to come from VA; the result concerning the involved XAI experts is still surprising, given the topic of the analysed papers. This phase aims to balance the distribution and promote the inclusion of experts from all fields in each design phase.
3. **Alignment between areas**. Starting from the considerations expressed in Sections 5.6 and 6, the actions described in the previous paragraphs can help communities close the temporal gap between the solutions adopted in DL, XAI and VA. Greater awareness and collaboration between the communities, and the availability of more integrated tools, would make implementing novel architectures and XDL methods on VA systems more accessible and faster. It would be possible to reduce the temporal gap observed between the availability of novel AI solutions to the public and their support from VA systems. Moreover, VA systems that support state-of-the-art models and explanation methods could further speed up the innovations in XDL, the main area of target users of these systems, making these efforts profitable for all three areas.
4. **Middleware software**. Finally, in a more mature field where both areas contribute significantly, we foresee a critical further step: building standard interfaces between VA and DL in terms of libraries and tools. Currently, a user must follow instructions, often tailored to the specific VA system, to upload a custom model or dataset into a VA system. The same difficulties arise when VA researchers have to adapt their systems to different models and workflows. Hence, there is a need for a set of tools and APIs that, starting from the already available frameworks (e.g. PyTorch [PGM*19], TensorFlow [ABC*16] and *OpenAI Gym* [BCP*16]) can be used as an interface between the DL libraries and the VA libraries (e.g. *D3.js* [BOH11]). Ideally, they should abstract the access to the DL frameworks, making support for a wide range of modifications easier and speeding up the spread of such systems. While the plugins for DL frameworks (Section 5.4) represent an initial attempt, they cannot modify the visual components yet. Apart from technical considerations, the study and research of more integrated data analysis pipelines that include humans in the loop are needed. These efforts could mitigate the contrast between DL (data-centric-controlled-loop) and VA (human-centric-controlled-loop) and help in developing more effective middleware. The availability of these tools could also boost the number of works that publish their code contextually in the paper.

8. Conclusions

This paper presented a report on the state of the art of VA for XDL. We hope this work provides researchers from VA, DL or XAI with the correct overview to begin novel research activities at the intersection of those fields. We provide them with the main background concepts, the existing works that fit those fields, and the analysis of trends, commonalities and specificities of 67 VA solutions in coping with XDL. Specifically, we hope that researchers in DL, or XAI have gained a solid understanding of the functionalities provided by VA in terms of support for explaining DNN. At the same time, practitioners working in VA can use this manuscript as a guide for the state-of-the-art solution available in specific contexts. Finally, we encourage VA researchers to start from this analysis to fill research gaps and improve the support for XDL in terms of available explanation methods and supporting analyses.

As additional future directions to explore the integration of these communities, we foresee an analysis of the trade-off between added advantages and the carbon footprint [MBB*22] of these systems compared to the usage of XDL methods alone and the employment of an alternative medium to improve the ease of use for end users, such as paper-based interfaces [BvOR21] and augmented reality.

Acknowledgements

We want to thank Nicholas Journet of the University of Bordeaux for his multiple comments that helped to improve the work before submission as well as the anonymous reviewers who suggested modifications that improved the quality of the paper.

Open Access Funding provided by Universita degli Studi di Roma La Sapienza within the CRUI-CARE Agreement.

References

[AAF*20] Andrienko N., Andrienko G., Fuchs G., Slingsby A., Turkay C., Wrobel S.: *Visual Analytics for Data Scientists*. Springer International Publishing, Cham, 2020. https://doi.org/10.1007/978-3-030-56146-8

[AB18] Adadi A., Berrada M.: Peeking inside the black-box: A survey on explainable artificial intelligence (XAI). *IEEE Access 6* (2018), 52138–52160. https://doi.org/10.1109/access.2018.2870052

[ABC*16] Abadi M., Barham P., Chen J., Chen Z., Davis A., Dean J., Devin M., Ghemawat S., Irving G., Isard M., Kudlur M., Levenberg J., Monga R., Moore S., Murray D. G., Steiner B., Tucker P., Vasudevan V., Warden P., Wicke M., Yu Y., Zheng X.: TensorFlow: A system for large-scale machine learning. In *Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)* (Savannah, GA, Nov. 2016), USENIX Association, pp. 265–283. https://www.usenix.org/conference/osdi16/technical-sessions/presentation/abadi

[ABC*19] Angelini M., Blasilli G., Catarci T., Lenti S., Santucci G.: Vulnus: Visual vulnerability analysis for network security. *IEEE Transactions on Visualization and Computer Graphics 25*, 1 (1 2019), 183–192. https://doi.org/10.1109/tvcg.2018.2865028

[ABL*19] Angelini M., Blasilli G., Lenti S., Palleschi A., Santucci G.: Towards enhancing RadViz analysis and interpretation. In Proceedings of the 2019 IEEE Visualization Conference (VIS) (Oct. 2019), IEEE, pp. 226–230. https://doi.org/10.1109/visual.2019.8933775

[ABL*22] Angelini M., Blasilli G., Lenti S., Palleschi A., Santucci G.: Effectiveness error: Measuring and improving RadViz visual effectiveness. *IEEE Transactions on Visualization and Computer Graphics 28*, 12 (12 2022), 4770–4786. https://doi.org/10.1109/tvcg.2021.3104879

[ABZ21] Ayyar M. P., Benois-Pineau J., Zemmari A.: Review of white box methods for explanations of convolutional neural networks in image classification tasks. Journal of Electronic Imaging 30, *5* (Sep. 2021), 050901. https://doi.org/10.1117/1.jei.30.5.050901

[ACD*15] Amershi S., Chickering M., Drucker S. M., Lee B., Simard P., Suh J.: ModelTracker: Redesigning performance analysis tools for machine learning. In Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (Apr. 2015), ACM, pp. 337–346. https://doi.org/10.1145/2702123.2702509

[ADS*20] Arrieta A. B., Díaz-Rodríguez N., Ser J. D., Bennetot A., Tabik S., Barbado A., Garcia S., Gil-Lopez S., Molina D., Benjamins R., Chatila R., Herrera F.: Explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion 58* (June 2020), 82–115. https://doi.org/10.1016/j.inffus.2019.12.012

[AGM*18] Adebayo J., Gilmer J., Muelly M., Goodfellow I., Hardt M., Kim B.: Sanity checks for saliency maps. In NIPS'18: Proceedings of the 32nd International Conference on Neural Information Processing Systems (Red Hook, NY, USA, 2018), Curran Associates Inc., pp. 9525–9536.

[AHH*14] Alsallakh B., Hanbury A., Hauser H., Miksch S., Rauber A.: Visual methods for analyzing probabilistic classification data. *IEEE Transactions on Visualization and Computer Graphics 20*, 12 (Dec. 2014), 1703–1712. https://doi.org/10.1109/tvcg.2014.2346660

[AK12] Augasta M. G., Kathirvalavakumar T.: Rule extraction from neural networks—a comparative study. In Proceedings of the International Conference on Pattern Recognition, Informatics and Medical Engineering (PRIME-2012) (Mar. 2012), IEEE. https://doi.org/10.1109/icprime.2012.6208380

[APA*16] Aliper A., Plis S., Artemov A., Ulloa A., Mamoshina P., Zhavoronkov A.: Deep learning applications for predicting pharmacological properties of drugs and drug repurposing using transcriptomic data. *Molecular Pharmaceutics 13*, 7 (June 2016), 2524–2530. https://doi.org/10.1021/acs.molpharmaceut.6b00248

[AS22] Alicioglu G., Sun B.: A survey of visual analytics for explainable artificial intelligence methods. *Computers & Graphics 102* (Feb. 2022), 502–520. https://doi.org/10.1016/j.cag.2021.09.002

[AvdWKL20] Abdul A., von der Weth C., Kankanhalli M., Lim B. Y.: COGAM: Measuring and moderating cognitive load in machine learning model explanations. In Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems (Apr. 2020), ACM, pp. 1–14. https://doi.org/10.1145/3313831.3376615

[AWS05] Albrecht-Buehler C., Watson B., Shamma D.: Visualizing live text streams using motion and temporal pooling. *IEEE Computer Graphics and Applications 25*, 3 (May 2005), 52–59. https://doi.org/10.1109/mcg.2005.70

[AZ20] Abnar S., Zuidema W.: Quantifying attention flow in transformers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (2020), Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.acl-main.385

[BBDW16] Beck F., Burch M., Diehl S., Weiskopf D.: A taxonomy and survey of dynamic graph visualization. *Computer Graphics Forum 36*, 1 (Jan. 2016), 133–159. https://doi.org/10.1111/cgf.12791

[BBM*15] Bach S., Binder A., Montavon G., Klauschen F., Müller K.-R., Samek W.: On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLOS ONE 10*, 7 (July 2015), e0130140. https://doi.org/10.1371/journal.pone.0130140

[BBR*16] Behrisch M., Bach B., Riche N. H., Schreck T., Fekete J.-D.: Matrix reordering methods for table and network visualization. *Computer Graphics Forum 35*, 3 (June 2016), 693–716. https://doi.org/10.1111/cgf.12935

[BCB15] Bahdanau D., Cho K., Bengio Y.: Neural machine translation by jointly learning to align and translate. In *Proceedings of the 3rd International Conference on Learning Representations, ICLR 2015, May 7–9, 2015, Conference Track*

Proceedings (San Diego, CA, USA, 2015), Y. Bengio and Y. LeCun (Eds.).

[BCP*16] Brockman G., Cheung V., Pettersson L., Schneider J., Schulman J., Tang J., Zaremba W.: *Openai gym. arXiv preprint arXiv:1606.01540* (June 2016). http://arxiv.org/abs/1606.01540

[BDME20] Becker F., Drichel A., Muller C., Ertl T.: Interpretable visualizations of deep neural networks for domain generation algorithm detection. In Proceedings of the 2020 IEEE Symposium on Visualization for Cyber Security (VizSec) (Oct. 2020), IEEE. https://doi.org/10.1109/vizsec51108.2020.00010

[Ber67] Bertin J.: *Sémiologie Graphique*. Gauthier-Villars, Paris, France, 1967.

[BJY*18] Bilal A., Jourabloo A., Ye M., Liu X., Ren L.: Do convolutional neural networks learn class hierarchy? *IEEE Transactions on Visualization and Computer Graphics 24*, 1 (Jan. 2018), 152–162. https://doi.org/10.1109/tvcg.2017.2744683

[BKW16] Beck F., Koch S., Weiskopf D.: Visual analysis and dissemination of scientific literature collections with SurVis. *IEEE Transactions on Visualization and Computer Graphics 22*, 1 (2016), 180–189. https://doi.org/10.1109/TVCG.2015.2467757

[BLW*20] Bau D., Liu S., Wang T., Zhu J.-Y., Torralba A.: Rewriting a deep generative model. In *Computer Vision – ECCV 2020*. Springer International Publishing, Cham, Switzerland (2020), pp. 351–369. https://doi.org/10.1007/978-3-030-58452-8_21

[BOH11] Bostock M., Ogievetsky V., Heer J.: D$\hat{3}$ data-driven documents. *IEEE Transactions on Visualization and Computer Graphics 17*, 12 (Dec. 2011), 2301–2309. https://doi.org/10.1109/tvcg.2011.185

[BQ] Benois-Pineau J., Quenot G.: ICPR2020 workshop explainable deep learning-AI (2020). https://edl-ai-icpr.labri.fr/. (Accessed 27 January 2022).

[BSH*10] Baehrens D., Schroeter T., Harmeling S., Kawanabe M., Hansen K., Müller K.-R.: How to explain individual classification decisions. *Journal of Machine Learning Research 11*, 61 (2010), 1803–1831. http://jmlr.org/papers/v11/baehrens10a.html

[BvOR21] Bauerle A., van Onzenoodt C., Ropinski T.: Net2vis—a visual grammar for automatically generating publication-tailored CNN architecture visualizations. *IEEE Transactions on Visualization and Computer Graphics 27*, 6 (June 2021), 2980–2991. https://doi.org/10.1109/tvcg.2021.3057483

[BWZ*21] Bansal G., Wu T., Zhou J., Fok R., Nushi B., Kamar E., Ribeiro M. T., Weld D.: Does the whole exceed its parts? The effect of AI explanations on complementary team performance. In Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems (May 2021), ACM, pp. 1–16. https://doi.org/10.1145/3411764.3445717

[BZK*17] Bau D., Zhou B., Khosla A., Oliva A., Torralba A.: Network dissection: Quantifying interpretability of deep visual representations. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (July 2017), IEEE. https://doi.org/10.1109/cvpr.2017.354

[BZP*20] Blumenschein M., Zhang X., Pomerenke D., Keim D. A., Fuchs J.: Evaluating reordering strategies for cluster identification in parallel coordinates. *Computer Graphics Forum 39*, 3 (June 2020), 537–549. https://doi.org/10.1111/cgf.14000

[CBN*20] Chan G. Y.-Y., Bertini E., Nonato L. G., Barr B., Silva C. T.: Melody: Generating and visualizing machine learning model summary to understand data and classifiers together. arXiv preprint arXiv:2007.10614 (July 2020). http://arxiv.org/abs/2007.10614

[CEH*19] Cabrera Á. A., Epperson W., Hohman F., Kahng M., Morgenstern J., Chau D. H.: FAIRVIS: Visual analytics for discovering intersectional bias in machine learning. In Proceedings of the 2019 IEEE Conference on Visual Analytics Science and Technology (VAST) (Oct. 2019), IEEE, pp. 46–56. https://doi.org/10.1109/vast47406.2019.8986948

[CEP20] Cantareira G. D., Etemad E., Paulovich F. V.: Exploring neural network hidden layer activity using vector fields. *Information 11*, 9 (Aug. 2020), 426. https://doi.org/10.3390/info11090426

[CFG*] Capobianco R., Feldman A., Gilpin L. H., Rosa B. L., Sun W., Xiang A.: Explainable AI approaches for debugging and diagnosis. Workshop @ NeurIPS2021 (2021). https://xai4debugging.github.io/. (Accessed 27 January 2022).

[CGM*17] Ceneda D., Gschwandtner T., May T., Miksch S., Schulz H.-J., Streit M., Tominski C.: Characterizing guidance in visual analytics. *IEEE Transactions on Visualization and Computer Graphics 23*, 1 (Jan. 2017), 111–120. https://doi.org/10.1109/tvcg.2016.2598468

[CGR*17] Chae J., Gao S., Ramanathan A., Steed C. A., Tourassi G.: *Visualization for Classification in Deep Neural Networks. Tech. Rep., Oak Ridge National Laboratory (ORNL)*, Oak Ridge, TN, United States, Oct. 2017. https://www.osti.gov/biblio/1407764

[CHS20] Chawla P., Hazarika S., Shen H.-W.: Token-wise sentiment decomposition for ConvNet: Visualizing a sentiment classifier. *Visual Informatics 4*, 2 (June 2020), 132–141. https://doi.org/10.1016/j.visinf.2020.04.006

[CL18] Choo J., Liu S.: Visual analytics for explainable deep learning. *IEEE Computer Graphics and Applications 38*, 4 (July 2018), 84–92. https://doi.org/10.1109/mcg.2018.042731661

[CLT*19] Chen C., Li O., Tao C., Barnett A. J., Su J., Rudin C.: This looks like that: Deep learning for interpretable image recognition. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems* (Red Hook, NY, USA, 2019), Curran Associates Inc.

[CMJ*20] Chatzimparmpas A., Martins R. M., Jusufi I., Kucher K., Rossi F., Kerren A.: The State of the Art in Enhancing Trust in Machine Learning Models with the Use of Visualizations. *Computer Graphics Forum 39*, 3 (June 2020), 713–756. https://doi.org/10.1111/cgf.14034

[CMQ21] Cheng F., Ming Y., Qu H.: DECE: Decision explorer with counterfactual explanations for machine learning models. *IEEE Transactions on Visualization and Computer Graphics 27*, 2 (Feb. 2021), 1438–1447. https://doi.org/10.1109/tvcg.2020.3030342

[CP20] Cantareira G. D., Paulovich F. V.: A generic model for projection alignment applied to neural network visualization. In *Proceedings of the EuroVis Workshop on Visual Analytics (EuroVA)* (2020), C. Turkay and K. Vrotsou (Eds.), The Eurographics Association. https://doi.org/10.2312/EUROVA.20201089

[CPC19] Carvalho D. V., Pereira E. M., Cardoso J. S.: Machine learning interpretability: A survey on methods and metrics. *Electronics 8*, 8 (July 2019), 832. https://doi.org/10.3390/electronics8080832

[CPCS20] Cashman D., Perer A., Chang R., Strobelt H.: Ablate, variate, and contemplate: Visual analytics for discovering neural architectures. *IEEE Transactions on Visualization and Computer Graphics 26*, 1 (Jan. 2020), 863–873. https://doi.org/10.1109/tvcg.2019.2934261

[CS95] Craven M. W., Shavlik J. W.: Extracting tree-structured representations of trained networks. In NIPS'95: Proceedings of the 8th International Conference on Neural Information Processing Systems (Cambridge, MA, USA, 1995), MIT Press, pp. 24–30.

[CWGvW19] Caballero H. S. G., Westenberg M. A., Gebre B., van Wijk J. J.: V-awake: A visual analytics approach for correcting sleep predictions from deep learning models. *Computer Graphics Forum 38*, 3 (June 2019), 1–12. https://doi.org/10.1111/cgf.13667

[CYO*20] Chan G. Y.-Y., Yuan J., Overton K., Barr B., Rees K., Nonato L. G., Bertini E., Silva C. T.: SUBPLEX: A visual analytics approach to understand local model explanations at the subpopulation level. IEEE Computer Graphics and Applications 42.6 (Nov. 2022), 24–36. https://doi.org/10.1109/mcg.2022. 3199727 6. https://vadl2017.github.io/. (Accessed 27 January 2022).

[CYPL] Choo J., Yosinski J., Park D., Liu S.: VADL 2017: Workshop on visual analytics for deep learning (2017). https://vadl2017.github.io/. (Accessed 27 January 2022).

[dCT19] de Carvalho Pagliosa L., Telea A. C.: RadViz: Improvements on radial-based visualizations. *Informatics 6*, 2 (Apr. 2019), 16. https://doi.org/10.3390/informatics6020016

[DFF10] Di Caro L., Frias-Martinez V., Frias-Martinez E.: Analyzing the role of dimension arrangement for data visualization in Radviz. In *Advances in Knowledge Discovery and Data Mining*. Springer, Berlin Heidelberg, 2010, pp. 125–132. https://doi.org/10.1007/978-3-642-13672-6_13

[DPW*20] Das N., Park H., Wang Z. J., Hohman F., Firstman R., Rogers E., Chau D. H. P.: Bluff: Interactively deciphering adversarial attacks on deep neural networks. In Proceedings of the 2020 IEEE Visualization Conference (VIS) (Oct. 2020), IEEE. https://doi.org/10.1109/vis47514.2020.00061

[DWA21] Dong S., Wang P., Abbas K.: A survey on deep learning and its applications. *Computer Science Review 40* (May 2021), 100379. https://doi.org/10.1016/j.cosrev.2021.100379

[DWB21] DeRose J. F., Wang J., Berger M.: Attention flows: Analyzing and comparing attention mechanisms in language models. *IEEE Transactions on Visualization and Computer Graphics 27*, 2 (Feb. 2021), 1160–1170. https://doi.org/10.1109/tvcg.2020.3028976

[DWSZ20] Dong Z., Wu T., Song S., Zhang M.: Interactive attention model explorer for natural language processing tasks with unbalanced data sizes. In Proceedings of the 2020 IEEE Pacific Visualization Symposium (PacificVis) (June 2020), IEEE. https://doi.org/10.1109/pacificvis48177.2020.1031

[EBCV09] Erhan D., Bengio Y., Courville A., Vincent P.: Visualizing Higher-layer Features of A Deep Network. 1341.3, University of Montreal, 2009, 1.

[EFN12] Endert A., Fiaux P., North C.: Semantic interaction for sensemaking: Inferring analytical reasoning for model steering. *IEEE Transactions on Visualization and Computer Graphics 18*, 12 (2012), 2879–2888. https://doi.org/10.1109/TVCG.2012.260

[EMK*21] Espadoto M., Martins R. M., Kerren A., Hirata N. S. T., Telea A. C.: Toward a quantitative survey of dimension reduction techniques. *IEEE Transactions on Visualization and Computer Graphics 27*, 3 (Mar. 2021), 2153–2173. https://doi.org/10.1109/TVCG.2019.2944182

[FCdMP21] Ferreira M. D., Cantareira G. D., de Mello R. F., Paulovich F. V.: Neural network training fingerprint: Visual analytics of the training process in classification neural networks. *Journal of Visualization 25*, 3 (Nov. 2021), 593–612. https://doi.org/10.1007/s12650-021-00809-4

[FV17] Fong R. C., Vedaldi A.: Interpretable explanations of black boxes by meaningful perturbation. In Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV) (Oct. 2017), IEEE. https://doi.org/10.1109/iccv.2017.371

[GA19] Gunning D., Aha D.: DARPA's explainable artificial intelligence (XAI) program. *AI Magazine 40*, 2 (June 2019), 44–58. https://doi.org/10.1609/aimag.v40i2.2850

[GB08] Greenberg S., Buxton B.: Usability evaluation considered harmful (some of the time). In *CHI"08: Proceeding of the Twenty-sixth Annual CHI Conference on Human Factors in*

Computing Systems (2008), ACM Press, pp. 111–120. https://doi.org/10.1145/1357054.1357074

[GBC16] Goodfellow I., Bengio Y., Courville A.: *Deep Learning*. MIT Press, Cambridge, Massachusetts, United States, 2016. http://www.deeplearningbook.org

[GKDF18] Greydanus S., Koul A., Dodge J., Fern A.: Visualizing and understanding Atari agents. In Proceedings of the 35th International Conference on Machine Learning (July 2018), J. Dy and A. Krause (Eds.), Proceedings of Machine Learning Research, PMLR, vol. *80*, pp. 1792–1801. URL: https://proceedings.mlr.press/v80/greydanus18a.html.

[GMR*19] Guidotti R., Monreale A., Ruggieri S., Turini F., Giannotti F., Pedreschi D.: A survey of methods for explaining black box models. *ACM Computing Surveys 51*, 5 (Sep. 2019), 1–42. https://doi.org/10.1145/3236009

[Gol17] Goldberg Y.: *Neural Network Methods for Natural Language Processing*. Springer International Publishing, Cham, 2017. https://doi.org/10.1007/978-3-031-02165-7

[GPM*14] Goodfellow I., Pouget-Abadie J., Mirza M., Xu B., Warde-Farley D., Ozair S., Courville A., Bengio Y.: Generative adversarial nets. In *Advances in Neural Information Processing Systems*. Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence and K. Q. Weinberger (Eds.). Curran Associates, Inc, New York, New York, United States (2014), vol. *27*.

[GSC*19] Gunning D., Stefik M., Choi J., Miller T., Stumpf S., Yang G.-Z.: XAI—explainable artificial intelligence. *Science Robotics 4*, 37 (Dec. 2019). https://doi.org/10.1126/scirobotics.aay7120

[GSK*19] Gehrmann S., Strobelt H., Kruger R., Pfister H., Rush A. M.: Visual interaction with deep learning models through collaborative semantic inference. *IEEE Transactions on Visualization and Computer Graphics* (2019), 1. https://doi.org/10.1109/tvcg.2019.2934595

[GTFA19] Gilpin L. H., Testart C., Fruchter N., Adebayo J.: Explaining explanations to society. In Workshop on Ethical, Social and Governance Issues in AI (NeurIPS 2018), (Dec. 2019). http://arxiv.org/abs/1901.06560

[Har15] Harley A. W.: An interactive node-link visualization of convolutional neural networks. In *Advances in Visual Computing*. Springer International Publishing, Cham, Switzerland (2015), pp. 867–877. https://doi.org/10.1007/978-3-319-27857-5_77

[HASS22] Hogräfer M., Angelini M., Santucci G., Schulz H.-J.: Steering-by-example for progressive visual analytics. *ACM Transactions on Intelligent Systems and Technology 13*, 6 (Sep. 2022). https://doi.org/10.1145/3531229

[HCC*20] Hilton J., Cammarata N., Carter S., Goh G., Olah C.: Understanding RL vision. *Distill 5*, 11 (Nov. 2020). https://doi.org/10.23915/distill.00029

[HDH*13] Hullman J., Drucker S., Henry Riche N., Lee B., Fisher D., Adar E.: A deeper understanding of sequence in narrative visualization. *IEEE Transactions on Visualization and Computer Graphics 19*, 12 (Dec. 2013), 2406–2415. https://doi.org/10.1109/tvcg.2013.119

[HGBA20] Halnaut A., Giot R., Bourqui R., Auber D.: Deep dive into deep neural networks with flows. In *Proceedings of the 15th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications* (2020), SCITEPRESS - Science and Technology Publications. https://doi.org/10.5220/0008989702310239

[HGM*97] Hoffman P., Grinstein G., Marx K., Grosse I., Stanley E.: DNA visual and analytic data mining. In *Proceedings of the Visualization'97 (Cat. No. 97CB36155)* (1997), IEEE, pp. 437–441. https://doi.org/10.1109/visual.1997.663916

[HHC17] Hohman F., Hodas N., Chau D. H.: ShapeShop. In Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems (May 2017), ACM. https://doi.org/10.1145/3027063.3053103

[HJZ*21] Huang X., Jamonnak S., Zhao Y., Wu T. H., Xu W.: A visual designer of layer-wise relevance propagation models. *Computer Graphics Forum 40*, 3 (June 2021), 227–238. https://doi.org/10.1111/cgf.14302

[HKPC19] Hohman F., Kahng M., Pienta R., Chau D. H.: Visual analytics in deep learning: An interrogative survey for the next frontiers. *IEEE Transactions on Visualization and Computer Graphics 25*, 8 (Aug. 2019), 2674–2693. https://doi.org/10.1109/tvcg.2018.2843369

[HLLE14] Heimerl F., Lohmann S., Lange S., Ertl T.: Word cloud explorer: Text analytics based on word clouds. In Proceedings of the 2014 47th Hawaii International Conference on System Sciences (Jan. 2014), IEEE, pp. 1833–1842. https://doi.org/10.1109/HICSS.2014.231

[HLvB*20] He W., Lee T.-Y., van Baar J., Wittenburg K., Shen H.-W.: DynamicsExplorer: Visual analytics for robot control tasks involving dynamics and LSTM-based control policies. In Proceedings of the 2020 IEEE Pacific Visualization Symposium (PacificVis) (June 2020), IEEE. https://doi.org/10.1109/pacificvis48177.2020.7127

[HLW*19] Hazarika S., Li H., Wang K.-C., Shen H.-W., Chou C.-S.: NNVA: Neural network assisted visual analysis of yeast cell polarization simulation. *IEEE Transactions on Visualization and Computer Graphics* (2019), 34–44. https://doi.org/10.1109/tvcg.2019.2934591

[HN98] Hintze J. L., Nelson R. D.: Violin plots: A box plot-density trace synergism. *The American Statistician 52*, 2 (May 1998), 181–184. https://doi.org/10.1080/00031305.1998.10480559

[HPRC20] Hohman F., Park H., Robinson C., Chau D. H. P.: Summit: Scaling deep learning interpretability by visualizing

activation and attribution summarizations. *IEEE Transactions on Visualization and Computer Graphics 26*, 1 (Jan. 2020), 1096–1106. https://doi.org/10.1109/tvcg.2019.2934659

[HPYM04] Han J., Pei J., Yin Y., Mao R.: Mining frequent patterns without candidate generation: A frequent-pattern tree approach. *Data Mining and Knowledge Discovery 8*, 1 (Jan. 2004), 53–87. https://doi.org/10.1023/b:dami.0000005258.31418.83

[HS97] Hochreiter S., Schmidhuber J.: Long short-term memory. *Neural Computation 9*, 8 (Nov. 1997), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

[HSG20] Hoover B., Strobelt H., Gehrmann S.: exBERT: A visual analysis tool to explore learned representations in transformer models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations* (2020), Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.acl-demos.22

[HSL*21] Han X., Shen H.-W., Li G., Lu X., Shan G., Wang Y.: IVDAS: An interactive visual design and analysis system for image data symmetry detection of CNN models. *Journal of Visualization 24*, 3 (Jan. 2021), 615–629. https://doi.org/10.1007/s12650-020-00721-3

[HW12] Heinrich J., Weiskopf D.: State of the Art of Parallel Coordinates. In *Eurographics 2013 - State of the Art Reports* (2012), M. Sbert and L. Szirmay-Kalos (Eds.), The Eurographics Association. https://doi.org/10.2312/CONF/EG2013/STARS/095-116

[HY22] Haydari A., Yilmaz Y.: Deep reinforcement learning for intelligent transportation systems: A survey. *IEEE Transactions on Intelligent Transportation Systems 23*, 1 (Jan. 2022), 11–32. https://doi.org/10.1109/TITS.2020.3008612

[ID90] Inselberg A., Dimsdale B.: Parallel coordinates: A tool for visualizing multi-dimensional geometry. In *Proceedings of the First IEEE Conference on Visualization: Visualization '90* (1990), IEEE Computer Society Press, pp. 361–378. https://doi.org/10.1109/visual.1990.146402

[JCM20] Johnson D., Carenini G., Murray G.: NJM-Vis: interpreting neural joint models in NLP. In Proceedings of the 25th International Conference on Intelligent User Interfaces (Mar. 2020), ACM. https://doi.org/10.1145/3377325.3377513

[JKV*22] Jaunet T., Kervadec C., Vuillemot R., Antipov G., Baccouche M., Wolf C.: VisQA: X-raying vision and language reasoning in transformers. *IEEE Transactions on Visualization and Computer Graphics 28*, 1 (Jan. 2022), 976–986. https://doi.org/10.1109/tvcg.2021.3114683

[JLJC05] Johansson J., Ljung P., Jern M., Cooper M.: Revealing structure within clustered parallel coordinates displays. In *IEEE Symposium on Information Visualization, 2005. INFOVIS 2005.* (2005), IEEE, pp. 125–132. https://doi.org/10.1109/INFVIS.2005.1532138

[JLL*19] Jia S., Lin P., Li Z., Zhang J., Liu S.: Visualizing surrogate decision trees of convolutional neural networks. *Journal of Visualization 23*, 1 (Nov. 2019), 141–156. https://doi.org/10.1007/s12650-019-00607-z

[JTH*21] Ji X., Tu Y., He W., Wang J., Shen H.-W., Yen P.-Y.: USEVis: Visual analytics of attention-based neural embedding in information retrieval. *Visual Informatics 5*, 2 (June 2021), 1–12. https://doi.org/10.1016/j.visinf.2021.03.003

[JVW20] Jaunet T., Vuillemot R., Wolf C.: DRLViz: Understanding decisions and memory in deep reinforcement learning. *Computer Graphics Forum 39*, 3 (June 2020), 49–61. https://doi.org/10.1111/cgf.13962

[JWW*22] Jin Z., Wang Y., Wang Q., Ming Y., Ma T., Qu H.: GNNLens: A visual analytics approach for prediction error diagnosis of graph neural networks. *IEEE Transactions on Visualization and Computer Graphics* (2022), 1. https://doi.org/10.1109/TVCG.2022.3148107

[KAF*08] Keim D., Andrienko G., Fekete J.-D., Görg C., Kohlhammer J., Melançon G.: *Visual Analytics: Definition, Process, and Challenges*. Springer, Berlin, Germany, 2008, pp. 154–175.

[KAKC18] Kahng M., Andrews P. Y., Kalro A., Chau D. H.: ActiVis: Visual exploration of industry-scale deep neural network models. *IEEE Transactions on Visualization and Computer Graphics 24*, 1 (Jan. 2018), 88–97. https://doi.org/10.1109/tvcg.2017.2744718

[Kan00] Kandogan E.: Star coordinates: A multi-dimensional visualization technique with uniform treatment of dimensions. In Proceedings of the IEEE Information Visualization Symposium, Late Breaking Hot Topics (2000), pp. 9–12.

[KCC*20] Kwon B. C., Chakraborty P., Codella J., Dhurandhar A., Sow D., Ng K.: Visually exploring contrastive explanation for diagnostic risk prediction on electronic health records. In Proceedings of the ICML 2020 Workshop on Human Interpretability in Machine Learning (WHI) (2020).

[KCK*19] Kwon B. C., Choi M., Kim J. T., Choi E., Kim Y. B., Kwon S., Sun J., Choo J.: RetainVis: Visual analytics with interpretable and interactive recurrent neural networks on electronic medical records. *IEEE Transactions on Visualization and Computer Graphics 25*, 1 (Jan. 2019), 299–309. https://doi.org/10.1109/tvcg.2018.2865027

[KDS*17] Krause J., Dasgupta A., Swartz J., Aphinyanaphongs Y., Bertini E.: A workflow for visual diagnostics of binary classifiers using instance-level explanations. In Proceedings of the 2017 IEEE Conference on Visual Analytics Science and Technology (VAST) (Oct. 2017), IEEE. https://doi.org/10.1109/vast.2017.8585720

[KH*09] Krizhevsky A.: Learning multiple layers of features from tiny images.

[KK15] Kucher K., Kerren A.: Text visualization techniques: Taxonomy, visual survey, and community insights. In Proceedings of the 2015 IEEE Pacific Visualization Symposium (PacificVis) (Apr. 2015), IEEE, pp. 117–121. https://doi.org/10.1109/pacificvis.2015.7156366

[KK19] Keane M. T., Kenny E. M.: How case-based reasoning explains neural networks: A theoretical analysis of XAI using post-hoc explanation-by-example from a survey of ANN-CBR twin-systems. In *Case-Based Reasoning Research and Development*. Springer International Publishing, Cham, Switzerland, 2019, pp. 155–171. https://doi.org/10.1007/978-3-030-29249-2_11

[KKEM10] Keim D., Kohlhammer J., Ellis G., Mansmann F.: *Mastering the Information Age: Solving Problems with Visual Analytics*. Eurographics Association, Goslar, Germany, 2010.

[KKP*11] Kohlhammer J., Keim D., Pohl M., Santucci G., Andrienko G.: Solving problems with visual analytics. *Procedia Computer Science 7* (2011), 117–120. Proceedings of the 2nd European Future Technologies Conference and Exhibition 2011 (FET 11). https://doi.org/10.1016/j.procs.2011.12.035

[KL17] Koh P. W., Liang P.: Understanding black-box predictions via influence functions. In Proceedings of the 34th International Conference on Machine Learning (Aug. 2017), Precup D., Teh Y. W., (Eds.), Proceedings of the Machine Learning Research, PMLR, vol. *70*, pp. 1885–1894. https://proceedings.mlr.press/v70/koh17a.html

[KMSZ06] Keim D., Mansmann F., Schneidewind J., Ziegler H.: Challenges in visual data analysis. In *IV'06: Proceedings of the Tenth International Conference on Information Visualisation* (2006), IEEE, pp. 9–16. https://doi.org/10.1109/iv.2006.31

[KNH*22] Khan S., Naseer M., Hayat M., Zamir S. W., Khan F. S., Shah M.: Transformers in vision: A survey. *ACM Computing Surveys 54*, 10s (Jan. 2022), 1–41. https://doi.org/10.1145/3505244

[KSZQ20] Khan A., Sohail A., Zahoora U., Qureshi A. S.: A survey of the recent architectures of deep convolutional neural networks. *Artificial Intelligence Review 53*, 8 (Apr. 2020), 5455–5516. https://doi.org/10.1007/s10462-020-09825-6

[KTC*19] Kahng M., Thorat N., Chau D. H. P., Viegas F. B., Wattenberg M.: GAN lab: Understanding complex deep generative models using interactive visual experimentation. *IEEE Transactions on Visualization and Computer Graphics 25*, 1 (Jan. 2019), 310–320. https://doi.org/10.1109/tvcg.2018.2864500

[KW13] Kingma D. P., Welling M.: Auto-encoding variational Bayes. In 2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14–16 (2013). http://arxiv.org/abs/1312.6114

[KW17] Kipf T. N., Welling M.: Semi-supervised classification with graph convolutional networks. In Proceedings of the 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24–26, Conference Track Proceedings (2017).

[KWG*18] Kim B., Wattenberg M., Gilmer J., Cai C., Wexler J., Viegas F., sayres R.: Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (TCAV). In Proceedings of the 35th International Conference on Machine Learning (July 2018), J. Dy and A. Krause (Eds.), *Proceedings of Machine Learning Research*, PMLR, vol. *80*, pp. 2668–2677. https://proceedings.mlr.press/v80/kim18d.html

[LBH15] LeCun Y., Bengio Y., Hinton G.: Deep learning. *Nature 521*, 7553 (May 2015), 436–444. https://doi.org/10.1038/nature14539

[LCJ*19] Liu D., Cui W., Jin K., Guo Y., Qu H.: DeepTracker: Visualizing the training process of convolutional neural networks. *ACM Transactions on Intelligent Systems and Technology 10*, 1 (Jan. 2019), 1–25. https://doi.org/10.1145/3200489

[LCN20] La Rosa B., Capobianco R., Nardi D.: Explainable inference on sequential data via memory-tracking. In Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (July 2020), International Joint Conferences on Artificial Intelligence Organization. https://doi.org/10.24963/ijcai.2020/278

[LGM20] Liao Q. V., Gruen D., Miller S.: Questioning the AI: Informing design practices for explainable AI user experiences. In Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems (Apr. 2020), ACM, pp. 1–15. https://doi.org/10.1145/3313831.3376590

[Lip18] Lipton Z. C.: The mythos of model interpretability. *Queue 16*, 3 (June 2018), 31–57. https://doi.org/10.1145/3236386.3241340

[LL17] Lundberg S. M., Lee S.-I.: A unified approach to interpreting model predictions. In NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing Systems (Red Hook, NY, USA, 2017), Curran Associates Inc., pp. 4768–4777.

[LLL*19] Liu S., Li Z., Li T., Srikumar V., Pascucci V., Bremer P.-T.: NLIZE: A perturbation-driven visual interrogation tool for analyzing and interpreting natural language inference models. *IEEE Transactions on Visualization and Computer Graphics 25*, 1 (Jan. 2019), 651–660. https://doi.org/10.1109/tvcg.2018.2865230

[LLM*19] Laugel T., Lesot M.-J., Marsala C., Renard X., Detyniecki M.: The dangers of post-hoc interpretability: Unjustified counterfactual explanations. In Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (Aug. 2019), International Joint Conferences on Artificial Intelligence Organization. https://doi.org/10.24963/ijcai.2019/388

[LLS*18] Liu M., Liu S., Su H., Cao K., Zhu J.: Analyzing the noise robustness of deep neural networks. In Proceedings of the 2018 IEEE Conference on Visual Analytics Science and

Technology (VAST) (Oct. 2018), IEEE. https://doi.org/10.1109/vast.2018.8802509

[LNC*11] Le Q. V., Ngiam J., Coates A., Lahiri A., Prochnow B., Ng A. Y.: On optimization methods for deep learning. In ICML'11: Proceedings of the 28th International Conference on International Conference on Machine Learning (Madison, WI, USA, 2011), Omnipress, pp. 265–272.

[LNH*18] Li Q., Njotoprawiro K. S., Haleem H., Chen Q., Yi C., Ma X.: EmbeddingVis: A visual analytics approach to comparative network embedding inspection. In Proceedings of the 2018 IEEE Conference on Visual Analytics Science and Technology (VAST) (Oct. 2018), IEEE. https://doi.org/10.1109/vast.2018.8802454

[LSL*17] Liu M., Shi J., Li Z., Li C., Zhu J., Liu S.: Towards better analysis of deep convolutional neural networks. *IEEE Transactions on Visualization and Computer Graphics 23*, 1 (Jan. 2017), 91–100. https://doi.org/10.1109/tvcg.2016.2598831

[LYW19] Liu H., Yin Q., Wang W. Y.: Towards explainable NLP: A generative explanation framework for text classification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics* (2019), Association for Computational Linguistics, pp. 5570–5581. https://doi.org/10.18653/v1/p19-1560

[LYY*20] Li R., Yin C., Yang S., Qian B., Zhang P.: Marrying medical domain knowledge with deep learning on electronic health records: A deep visual analytics approach. *Journal of Medical Internet Research 22*, 9 (Sep. 2020), e20645. https://doi.org/10.2196/20645

[MBB*22] Mariette J., Blanchard O., Berné O., Aumont O., Carrey J., Ligozat A., Lellouch E., Roche P.-E., Guennebaud G., Thanwerdas J., Bardou P., Salin G., Maigne E., Servan S., Ben-Ari T.: An open-source tool to assess the carbon footprint of research. *Environmental Research: Infrastructure and Sustainability 2*, 3 (Sep. 2022), 035008. https://doi.org/10.1088/2634-4505/ac84a4

[MBSG20] Meske C., Bunde E., Schneider J., Gersch M.: Explainable artificial intelligence: Objectives, stakeholders, and future research opportunities. *Information Systems Management 39*, 1 (Dec. 2020), 53–63. https://doi.org/10.1080/10580530.2020.1849465

[MCZ*17] Ming Y., Cao S., Zhang R., Li Z., Chen Y., Song Y., Qu H.: Understanding hidden memories of recurrent neural networks. In Proceedings of the 2017 IEEE Conference on Visual Analytics Science and Technology (VAST) (Oct. 2017), IEEE. https://doi.org/10.1109/vast.2017.8585721

[MFH*21] Ma Y., Fan A., He J., Nelakurthi A. R., Maciejewski R.: A visual analytics framework for explaining and diagnosing transfer learning processes. *IEEE Transactions on Visualization and Computer Graphics 27*, 2 (Feb. 2021), 1385–1395. https://doi.org/10.1109/tvcg.2020.3028888

[MKS*13] Mnih V., Kavukcuoglu K., Silver D., Graves A., Antonoglou I., Wierstra D., Riedmiller M.: Playing atari with deep reinforcement learning. In Deep Learning Workshop (NIPS 2013), (Dec. 2013). http://arxiv.org/abs/1312.5602

[MMD*19] Murugesan S., Malik S., Du F., Koh E., Lai T. M.: DeepCompare: Visual and interactive comparison of deep learning model performance. *IEEE Computer Graphics and Applications 39*, 5 (Sep. 2019), 47–59. https://doi.org/10.1109/mcg.2019.2919033

[MPV*16] Mirowski P., Pascanu R., Viola F., Soyer H., Ballard A. J., Banino A., Denil M., Goroshin R., Sifre L., Kavukcuoglu K., Kumaran D., Hadsell R.: Learning to navigate in complex environments. In Deep Learning for Action and Interaction (NIPS 2016) (Dec. 2016). http://arxiv.org/abs/1611.03673

[MSHB22] Mishra A., Soni U., Huang J., Bryan C.: Why? why not? when? visual explanations of agent behaviour in reinforcement learning. In *Proceedings of the 2022 IEEE 15th Pacific Visualization Symposium (PacificVis)* (Los Alamitos, CA, USA, Apr. 2022), IEEE, pp. 111–120. https://doi.org/10.1109/pacificvis53943.2022.00020

[MV15] Mahendran A., Vedaldi A.: Understanding deep image representations by inverting them. In Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2015), IEEE. https://doi.org/10.1109/cvpr.2015.7299155

[MXC*20] Ming Y., Xu P., Cheng F., Qu H., Ren L.: ProtoSteer: Steering deep sequence model with prototypes. *IEEE Transactions on Visualization and Computer Graphics 26*, 1 (Jan. 2020), 238–248. https://doi.org/10.1109/tvcg.2019.2934267

[MZR21] Mohseni S., Zarei N., Ragan E. D.: A multidisciplinary survey and framework for design and evaluation of explainable AI systems. *ACM Transactions on Interactive Intelligent Systems 11*, 3-4 (Dec. 2021), 1–45. https://doi.org/10.1145/3387166

[NHP*18] Nie S., Healey C., Padia K., Leeman-Munk S., Benson J., Caira D., Sethi S., Devarajan R.: Visualizing deep neural networks for text analytics. In Proceedings of the 2018 IEEE Pacific Visualization Symposium (PacificVis) (Apr. 2018), IEEE. https://doi.org/10.1109/pacificvis.2018.00031

[NQ17] Norton A. P., Qi Y.: Adversarial-playground: A visualization suite showing how adversarial examples fool deep learning. In Proceedings of the 2017 IEEE Symposium on Visualization for Cyber Security (VizSec) (Oct. 2017), IEEE. https://doi.org/10.1109/vizsec.2017.8062202

[OMS17] Olah C., Mordvintsev A., Schubert L.: Feature visualization. *Distill 2*, 11 (Nov. 2017). https://doi.org/10.23915/distill.00007

[PBH*] Perer A., Boggust A., Hohman F., Strobelt H., El-Assady M., Wang Z. J.: 5th VISxAI workshop at IEEE VIS 2022 (2022). https://visxai.io/. (Accessed 27 January 2022).

[PCN*19] Park C., Choo J., Na I., Jo Y., Shin S., Yoo J., Kwon B. C., Zhao J., Noh H., Lee Y.: SANVis: Visual analytics for understanding self-attention networks. In Proceedings of the 2019 IEEE Visualization Conference (VIS) (Oct. 2019), IEEE. https://doi.org/10.1109/visual.2019.8933677

[PDD*22] Park H., Das N., Duggal R., Wright A. P., Shaikh O., Hohman F., Chau D. H. P.: NeuroCartography: Scalable automatic visual summarization of concepts in deep neural networks. *IEEE Transactions on Visualization and Computer Graphics 28*, 1 (Jan. 2022), 813–823. https://doi.org/10.1109/tvcg.2021.3114858

[PDS18] Petsiuk V., Das A., Saenko K.: Rise: Randomized input sampling for explanation of black-box models. In *Proceedings of the British Machine Vision Conference (BMVC)* (2018). http://bmvc2018.org/contents/papers/1064.pdf

[PGM*19] Paszke A., Gross S., Massa F., Lerer A., Bradbury J., Chanan G., Killeen T., Lin Z., Gimelshein N., Antiga L., Desmaison A., Köpf A., Yang E., De Vito Z., Raison M., Tejani A., Chilamkurthy S., Steiner B., Fang L., Bai J., Chintala S.: Pytorch: An imperative style, high-performance deep learning library. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems* (Red Hook, NY, USA, 2019), Curran Associates Inc.

[PHG*18] Pezzotti N., Hollt T., Gemert J. V., Lelieveldt B. P., Eisemann E., Vilanova A.: DeepEyes: Progressive visual analytics for designing deep neural networks. *IEEE Transactions on Visualization and Computer Graphics 24*, 1 (Jan. 2018), 98–108. https://doi.org/10.1109/tvcg.2017.2744358

[PHL*16] Pezzotti N., Höllt T., Lelieveldt B., Eisemann E., Vilanova A.: Hierarchical stochastic neighbor embedding. *Computer Graphics Forum 35*, 3 (June 2016), 21–30. https://doi.org/10.1111/cgf.12878

[POP21] Poli J.-P., Ouerdane W., Pierrard R.: Generation of textual explanations in XAI: The case of semantic annotation. In Proceedings of the 2021 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE) (July 2021), IEEE, pp. 1–6. https://doi.org/10.1109/fuzz45933.2021.9494589

[PPP20] Panigutti C., Perotti A., Pedreschi D.: Doctor XAI: An ontology-based approach to black-box sequential data classification explanations. In Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency (Jan. 2020), ACM. https://doi.org/10.1145/3351095.3372855

[PSS*20] Poyiadzi R., Sokol K., Santos-Rodriguez R., Bie T. D., Flach P.: FACE. In Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society (Feb. 2020), ACM. https://doi.org/10.1145/3375627.3375850

[PSY*19] Pouyanfar S., Sadiq S., Yan Y., Tian H., Tao Y., Reyes M. P., Shyu M.-L., Chen S.-C., Iyengar S. S.: A survey on deep learning. *ACM Computing Surveys 51*, 5 (Sep. 2019), 1–36. https://doi.org/10.1145/3234150

[PvSvdE*22] Prasad V., van Sloun R. J. G., van den Elzen S., Vilanova A., Pezzotti N.: The transform-and-perform framework: Explainable deep learning beyond classification. *IEEE Transactions on Visualization and Computer Graphics* (2022), 1–14. https://doi.org/10.1109/tvcg.2022.3219248

[PYN*21] Park C., Yang S., Na I., Chung S., Shin S., Kwon B. C., Park D., Choo J.: Vatun: Visual analytics for testing and understanding convolutional neural networks. *EuroVis 2021 - Short Papers* (2021). https://doi.org/10.2312/EVS.20211047

[RAL*17] Ren D., Amershi S., Lee B., Suh J., Williams J. D.: Squares: Supporting interactive performance analysis for multiclass classifiers. *IEEE Transactions on Visualization and Computer Graphics 23*, 1 (Jan. 2017), 61–70. https://doi.org/10.1109/tvcg.2016.2598828

[RC94] Rao R., Card S. K.: The table lens: Merging graphical and symbolic representations in an interactive focus + context visualization for tabular information. In *CHI'94: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems—CHI '94* (New York, NY, USA, 1994), Association for Computing Machinery, pp. 318–322. https://doi.org/10.1145/191666.191776

[RCN22] Rosa B. L., Capobianco R., Nardi D.: A self-interpretable module for deep image classification on small data. Applied Intelligence (Aug. 2022). https://doi.org/10.1007/s10489-022-03886-6

[RCPW21] Rathore A., Chalapathi N., Palande S., Wang B.: TopoAct: Visually exploring the shape of activations in deep learning. *Computer Graphics Forum 40*, 1 (Jan. 2021), 382–397. https://doi.org/10.1111/cgf.14195

[RFFT17] Rauber P. E., Fadel S. G., Falcao A. X., Telea A. C.: Visualizing the hidden activity of artificial neural networks. *IEEE Transactions on Visualization and Computer Graphics 23*, 1 (Jan. 2017), 101–110. https://doi.org/10.1109/tvcg.2016.2598838

[RLSR21] Rubio-Sánchez M., Lehmann D. J., Sanchez A., Rojo-Álvarez J. L.: Optimal axes for data value estimation in star coordinates and radial axes plots. *Computer Graphics Forum 40*, 3 (June 2021), 483–494. https://doi.org/10.1111/cgf.14323

[RS14] Rubio-Sánchez M., Sanchez A.: Axis calibration for improving data attribute estimation in star coordinates plots. *IEEE Transactions on Visualization and Computer Graphics 20*, 12 (Dec. 2014), 2013–2022. https://doi.org/10.1109/tvcg.2014.2346258

[RSG16] Ribeiro M. T., Singh S., Guestrin C.: "Why should i trust you?". In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (Aug. 2016), ACM. https://doi.org/10.1145/2939672.2939778

[RXGD22] Ras G., Xie N., Gerven M. V., Doran D.: Explainable deep learning: A field guide for the uninitiated. *Journal of Artificial Intelligence Research 73* (Jan. 2022), 329–397. https://doi.org/10.1613/jair.1.13200

[SBE*18] Sevastjanova R., Beck F., Ell B., Turkay C., Henkin R., Butt M., Keim D., El-Assady M.: Going beyond visualization: Verbalization as complementary medium to explain machine learning models. In Details: Workshop on Visualization for AI Explainability at IEEE VIS. Berlin (Oct. 2018).

[SC78] Sakoe H., Chiba S.: Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing 26*, 1 (Feb. 1978), 43–49. https://doi.org/10.1109/tassp.1978.1163055

[SC07] Salvador S., Chan P.: Toward accurate dynamic time warping in linear time and space. *Intelligent Data Analysis 11*, 5 (Oct. 2007), 561–580. https://doi.org/10.3233/ida-2007-11508

[SCD*19] Selvaraju R. R., Cogswell M., Das A., Vedantam R., Parikh D., Batra D.: Grad-CAM: Visual explanations from deep networks via gradient-based localization. *International Journal of Computer Vision 128*, 2 (Oct. 2019), 336–359. https://doi.org/10.1007/s11263-019-01228-7

[SCS*17] Smilkov D., Carter S., Sculley D., Viégas F. B., Wattenberg M.: Direct-manipulation visualization of deep networks. In ICML Visualization Workshop, (Jun. 2017). http://arxiv.org/abs/1708.03788

[SG18] Sarikaya A., Gleicher M.: Scatterplots: Tasks, data, and designs. *IEEE Transactions on Visualization and Computer Graphics 24*, 1 (Jan. 2018), 402–412. https://doi.org/10.1109/tvcg.2017.2744184

[SGB*19] Strobelt H., Gehrmann S., Behrisch M., Perer A., Pfister H., Rush A. M.: Seq2seq-vis: A visual debugging tool for sequence-to-sequence models. *IEEE Transactions on Visualization and Computer Graphics 25*, 1 (Jan. 2019), 353–363. https://doi.org/10.1109/tvcg.2018.2865044

[SGPR18] Strobelt H., Gehrmann S., Pfister H., Rush A. M.: LSTMVis: A tool for visual analysis of hidden state dynamics in recurrent neural networks. *IEEE Transactions on Visualization and Computer Graphics 24*, 1 (Jan. 2018), 667–676. https://doi.org/10.1109/tvcg.2017.2744158

[SHB*14] Sedlmair M., Heinzl C., Bruckner S., Piringer H., Möller T.: Visual parameter space analysis: A conceptual framework. *IEEE Transactions on Visualization and Computer Graphics 20*, 12 (2014), 2161–2170. https://doi.org/10.1109/TVCG.2014.2346321

[Shn92] Shneiderman B.: Tree visualization with tree-maps: 2-D space-filling approach. *ACM Transactions on Graphics 11*, 1 (Jan. 1992), 92–99. https://doi.org/10.1145/102377.115768

[Shn96] Shneiderman B.: The eyes have it: A task by data type taxonomy for information visualizations. In *Proceedings of the 1996 IEEE Symposium on Visual Languages* (1996), IEEE Computer Society Press, pp. 336–343. https://doi.org/10.1109/vl.1996.545307

[SMM*19] Siddiqui S. A., Mercier D., Munir M., Dengel A., Ahmed S.: TSViz: Demystification of deep learning models for time-series analysis. *IEEE Access 7* (2019), 67027–67040. https://doi.org/10.1109/access.2019.2912823

[ŠSE*21] Škrlj B., Sheehan S., Eržen N., Robnik-Šikonja M., Luz S., Pollak S.: Exploring neural language models via analysis of local and global self-attention spaces. In *Proceedings of the EACL Hackashop on News Media Content Analysis and Automated Report Generation* (Online, Apr. 2021), Association for Computational Linguistics, pp. 76–83. https://aclanthology.org/2021.hackashop-1.11

[SSS*14] Sacha D., Stoffel A., Stoffel F., Kwon B. C., Ellis G., Keim D. A.: Knowledge generation model for visual analytics. *IEEE Transactions on Visualization and Computer Graphics 20*, 12 (Dec. 2014), 1604–1613. https://doi.org/10.1109/tvcg.2014.2346481

[SSSE19] Spinner T., Schlegel U., Schafer H., El-Assady M.: explAIner: A visual analytics framework for interactive and explainable machine learning. *IEEE Transactions on Visualization and Computer Graphics* (2019), 1. https://doi.org/10.1109/tvcg.2019.2934629

[SSZ17] Snell J., Swersky K., Zemel R.: Prototypical networks for few-shot learning. In NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing Systems (Red Hook, NY, USA, 2017), Curran Associates Inc., pp. 4080–4090.

[STLD20] Scheibel W., Trapp M., Limberger D., Döllner J.: A taxonomy of treemap visualization techniques. In Proceedings of the VISIGRAPP (2020).

[STN*16] Smilkov D., Thorat N., Nicholson C., Reif E., Viégas F. B., Wattenberg M.: Embedding projector: Interactive visualization and interpretation of embeddings. In Interpretable Machine Learning for Complex Systems Workshop (NIPS 2016), (Dec. 2016). Barcelona, Spain. arXiv preprint arXiv:1611.05469. http://arxiv.org/abs/1611.05469

[STY17] Sundararajan M., Taly A., Yan Q.: Axiomatic attribution for deep networks. In ICML'17: Proceedings of the 34th International Conference on Machine Learning (2017), JMLR.org, Vol. *70*, pp. 3319–3328.

[SW01] Shneiderman B., Wattenberg M.: Ordered treemap layouts. In *Proceedings of the IEEE Symposium on Information Visualization, 2001. INFOVIS 2001* (2001), IEEE, pp. 73–78. https://doi.org/10.1109/infvis.2001.963283

[SW17] Strezoski G., Worring M.: Plug-and-play interactive deep network visualization. In *Proceedings of the VADL: Visual Analytics for Deep Learning* (2017), 0100–0106.

[SW22] Sun C., Wang K.-C.: DLA-VPS: Deep-learning-assisted visual parameter space analysis of cosmological simulations. *IEEE Computer Graphics and Applications 42*, 3 (2022), 41–52. https://doi.org/10.1109/MCG.2022.3169554

[SWJ*20] Shen Q., Wu Y., Jiang Y., Zeng W., LAU A. K. H., Vianova A., Qu H.: Visual interpretation of recurrent neural network on multi-dimensional time-series forecast. In Proceedings of the 2020 IEEE Pacific Visualization Symposium (PacificVis) (June 2020), IEEE. https://doi.org/10.1109/pacificvis48177.2020.2785

[SZ14] Simonyan K., Zisserman A.: Very deep convolutional networks for large-scale image recognition. In 3rd International Conference on Learning Representations (ICLR), (May 2015), 1–14. Computational and Biological Learning Society.

[TC06] Thomas J., Cook K.: A visual analytics agenda. *IEEE Computer Graphics and Applications 26*, 1 (Jan. 2006), 10–13. https://doi.org/10.1109/mcg.2006.5

[TS20] Tominski C., Schumann H.: *Interactive Visual Data Analysis*. AK Peters Visualization Series. CRC Press, Natick, Massachusetts, United States, Apr. 2020. https://doi.org/10.1201/9781315152707

[TSK*18] Tan C., Sun F., Kong T., Zhang W., Yang C., Liu C.: A survey on deep transfer learning. In *Artificial Neural Networks and Machine Learning—ICANN 2018*. Springer International Publishing, Cham, Switzerland, 2018, pp. 270–279. https://doi.org/10.1007/978-3-030-01424-7_27

[VA20] Venkatasubramanian S., Alfano M.: The philosophical basis of algorithmic recourse. In Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency (Jan. 2020), ACM. https://doi.org/10.1145/3351095.3372876

[VCC*18] Velic̆ković P., Cucurull G., Casanova A., Romero A., Liò P., Bengio Y.: Graph attention networks. In *Proceedings of the International Conference on Learning Representations* (2018). https://openreview.net/forum?id=rJXMpikCZ

[vdBCR*20] van den Brandt A., Christopher M., Rezapour J., Welsbie D. S., Camp A. S., Baxter S. L., Do J., Moghimi S., Belghith A., Bowd C., Bowd C., Goldbaum M. H., Weinreb R. N., Westenberg M. A., Snijders C. C. P., Zangwill L. M.: Glance: A visual analytics approach for opening the black box to explain deep learning predictions of glaucomatous visual field damage from optical coherence tomography scans. *Investigative Ophthalmology & Visual Science 61*, 7 (2020), 4527–4527.

[vdMH08] van der Maaten L., Hinton G.: Visualizing data using t-SNE. *Journal of Machine Learning Research 9*, 86 (2008), 2579–2605. http://jmlr.org/papers/v9/vandermaaten08a.html

[Vig19] Vig J.: A multiscale visualization of attention in the transformer model. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations* (2019), Association for Computational Linguistics, pp. 37–42. https://doi.org/10.18653/v1/p19-3007

[VSP*17] Vaswani A., Shazeer N., Parmar N., Uszkoreit J., Jones L., Gomez A. N., Kaiser L., Polosukhin I.: Attention is all you need. In NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing Systems (Red Hook, NY, USA, 2017), Curran Associates Inc., pp. 6000–6010.

[WGSY19] Wang J., Gou L., Shen H.-W., Yang H.: DQNViz: A visual analytics approach to understand deep q-networks. *IEEE Transactions on Visualization and Computer Graphics 25*, 1 (Jan. 2019), 288–298. https://doi.org/10.1109/tvcg.2018.2864504

[WGYS18] Wang J., Gou L., Yang H., Shen H.-W.: GANViz: A visual analytics approach to understand the adversarial game. *IEEE Transactions on Visualization and Computer Graphics 24*, 6 (June 2018), 1905–1917. https://doi.org/10.1109/tvcg.2018.2816223

[WGZ*19] Wang J., Gou L., Zhang W., Yang H., Shen H.-W.: DeepVID: Deep visual interpretation and diagnosis for image classifiers via knowledge distillation. *IEEE Transactions on Visualization and Computer Graphics 25*, 6 (June 2019), 2168–2180. https://doi.org/10.1109/tvcg.2019.2903943

[WHJ*22] Wang X., He J., Jin Z., Yang M., Wang Y., Qu H.: M2lens: Visualizing and explaining multimodal models for sentiment analysis. *IEEE Transactions on Visualization and Computer Graphics 28*, 1 (Jan. 2022), 802–812. https://doi.org/10.1109/tvcg.2021.3114794

[WLC18] Wang F., Liu H., Cheng J.: Visualizing deep neural network by alternately image blurring and deblurring. *Neural Networks 97* (Jan. 2018), 162–172. https://doi.org/10.1016/j.neunet.2017.09.007

[WM20] Wang C., Ma K.-L.: HypperSteer: Hypothetical steering and data perturbation in sequence prediction with deep learning. arXiv preprint arXiv:2011.02149 (Nov. 2020). http://arxiv.org/abs/2011.02149

[WMR17] Wachter S., Mittelstadt B., Russell C.: Counterfactual explanations without opening the black box: Automated decisions and the GDPR. *SSRN Electronic Journal* (2017). https://doi.org/10.2139/ssrn.3063289

[WONM18] Wang C., Onishi T., Nemoto K., Ma K.-L.: Visual reasoning of feature attribution with deep recurrent neural networks. In Proceedings of the 2018 IEEE International Conference on Big Data (Big Data) (Dec. 2018), IEEE, pp. 1661–1668. https://doi.org/10.1109/bigdata.2018.8622502

[WPB*19] Wexler J., Pushkarna M., Bolukbasi T., Wattenberg M., Viegas F., Wilson J.: The what-if tool: Interactive probing of machine learning models. *IEEE Transactions on Visualization and Computer Graphics* (2019), 1. https://doi.org/10.1109/tvcg.2019.2934619

[WPC*21] Wu Z., Pan S., Chen F., Long G., Zhang C., Philip S. Y.: A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems 32*, 1 (Jan. 2021), 4–24. https://doi.org/10.1109/tnnls.2020.2978386

[WSP*21] Wright A. P., Shaikh O., Park H., Epperson W., Ahmed M., Pinel S., Chau D. H. P., Yang D.: RECAST: Enabling user recourse and interpretability of toxicity detection models with interactive visualization. *Proceedings of the ACM on Human-Computer Interaction 5*, CSCW1 (Apr. 2021), 1–26. https://doi.org/10.1145/3449280

[WTC21] Wang Z. J., Turko R., Chau D. H.: Dodrio: Exploring transformer models with interactive visualization. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations* (Online, Aug. 2021), Association for Computational Linguistics, pp. 132–141. https://doi.org/10.18653/v1/2021.acl-demo.16

[WTS*20] Wang Z. J., Turko R., Shaikh O., Park H., Das N., Hohman F., Kahng M., Chau D. H.: CNN 101: Interactive visual learning for convolutional neural networks. In Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems (Apr. 2020), ACM. https://doi.org/10.1145/3334480.3382899

[WWM20] Wang C., Wang X., Ma K.-L.: Interactive visualization for explaining value-level feature attribution in event prediction with RNNs. arXiv preprint arXiv:2001.08379 (Jan. 2020). http://arxiv.org/abs/2001.08379

[WZY20] Wang J., Zhang W., Yang H.: SCANViz: Interpreting the symbol-concept association captured by deep neural networks through visual analytics. In Proceedings of the 2020 IEEE Pacific Visualization Symposium (PacificVis) (June 2020), IEEE. https://doi.org/10.1109/pacificvis48177.2020.3542

[WZY*22] Wang J., Zhang W., Yang H., Yeh C.-C. M., Wang L.: Visual analytics for RNN-based deep reinforcement learning. *IEEE Transactions on Visualization and Computer Graphics 28*, 12 (Dec. 2022), 4141–4155. https://doi.org/10.1109/tvcg.2021.3076749

[XBK*15] Xu K., Ba J. L., Kiros R., Cho K., Courville A., Salakhutdinov R., Zemel R. S., Bengio Y.: Show, attend and tell: Neural image caption generation with visual attention. In ICML'15: Proceedings of the 32nd International Conference on International Conference on Machine Learning (2015), JMLR.org, vol. *37*, pp. 2048–2057.

[XHLJ19] Xu K., Hu W., Leskovec J., Jegelka S.: How powerful are graph neural networks? In Proceedings of the International Conference on Learning Representations (2019).

[XKL*18] Xin Y., Kong L., Liu Z., Chen Y., Li Y., Zhu H., Gao M., Hou H., Wang C.: Machine learning and deep learning methods for cybersecurity. *IEEE Access 6* (2018), 35365–35381. https://doi.org/10.1109/access.2018.2836950

[YCN*15] Yosinski J., Clune J., Nguyen A., Fuchs T., Lipson H.: Understanding neural networks through deep visualization. In Deep Learning Workshop, 31st International Conference on Machine Learning, Lille, France, (Jul. 2015). http://arxiv.org/abs/1506.06579

[YKSJ07] Yi J. S., Kang Y. a., Stasko J., Jacko J.: Toward a deeper understanding of the role of interaction in information visualization. *IEEE Transactions on Visualization and Computer Graphics 13*, 6 (2007), 1224–1231. https://doi.org/10.1109/TVCG.2007.70515

[YSHZ19] Yu Y., Si X., Hu C., Zhang J.: A review of recurrent neural networks: LSTM cells and network architectures. *Neural Computation 31*, 7 (July 2019), 1235–1270. https://doi.org/10.1162/neco_a_01199

[ZDXR20] Zhao J., Dai Z., Xu P., Ren L.: ProtoViewer: Visual interpretation and diagnostics of deep neural networks with factorized prototypes. In Proceedings of the 2020 IEEE Visualization Conference (VIS) (Oct. 2020), IEEE. https://doi.org/10.1109/vis47514.2020.00064

[ZF14] Zeiler M. D., Fergus R.: Visualizing and understanding convolutional networks. In *Computer Vision—ECCV 2014*. Springer International Publishing, 2014, pp. 818–833. https://doi.org/10.1007/978-3-319-10590-1_53

[ZHP*17] Zeng H., Haleem H., Plantaz X., Cao N., Qu H.: Cnncomparator: Comparative analytics of convolutional neural networks. In Workshop on Visual Analytics for Deep Learning (VADL 2017) at IEEE VIS 2017: Phoenix, Arizona, United States, (Oct. 2017). http://arxiv.org/abs/1710.05285

[ZKTF10] Zeiler M. D., Krishnan D., Taylor G. W., Fergus R.: Deconvolutional networks. In Proceedings of the 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (June 2010), IEEE. https://doi.org/10.1109/cvpr.2010.5539957

[ZTLT21] Zhang Y., Tino P., Leonardis A., Tang K.: A survey on neural network interpretability. *IEEE Transactions on Emerging Topics in Computational Intelligence 5*, 5 (Oct. 2021), 726–742. https://doi.org/10.1109/tetci.2021.3100641

[ZWM*19] Zhang J., Wang Y., Molino P., Li L., Ebert D. S.: Manifold: A model-agnostic framework for interpretation and diagnosis of machine learning models. *IEEE Transactions on Visualization and Computer Graphics 25*, 1 (Jan. 2019), 364–373. https://doi.org/10.1109/tvcg.2018.2864499

[ZXZ*17] Zhong W., Xie C., Zhong Y., Wang Y., Xu W., Cheng S., Mueller K.: Evolutionary visual analysis of deep neural networks. In Proceedings of the ICML Workshop on Visualization for Deep Learning (2017), pp. 9.

[ZZM16] Zahavy T., Zrihem N. B., Mannor S.: Graying the black box: Understanding DQNs. In ICML'16: Proceedings of the 33rd International Conference on International Conference on Machine Learning (2016), JMLR.org, vol. *48*, pp. 1899–1908.