# Visualizing surrogate decision trees of convolutional neural networks

**Article** *in* Journal of Visualization · November 2019

DOI: 10.1007/s12650-019-00607-z



**5 authors**, including:

Shichao Jia
Tianjin University
**5** PUBLICATIONS **5** CITATIONS
SEE PROFILE

Zeyu Li
Tianjin University
**4** PUBLICATIONS **4** CITATIONS
SEE PROFILE

Jiawan Zhang
Tianjin University
**141** PUBLICATIONS **885** CITATIONS
SEE PROFILE

J Vis
https://doi.org/10.1007/s12650-019-00607-z

REGULAR PAPER

**Shichao Jia** · **Peiwen Lin** · **Zeyu Li** · **Jiawan Zhang** · **Shixia Liu**

# Visualizing surrogate decision trees of convolutional neural networks

Received: 6 July 2019 / Accepted: 3 September 2019


**Abstract** Interpreting the decision-making of black boxes in machine learning becomes urgent nowadays due to their lack of transparency. One effective way to interpret these models is to transform them into interpretable surrogate models such as decision trees and rule lists. Compared with other methods that open the black boxes, rule extraction is a universal method which can theoretically extend to any black boxes. However, in practice, it is not appropriate for deep learning models such as convolutional neural networks (CNNs), since the extracted rules or decision trees are too large to interpret and the rules are not at the semantic level. These two drawbacks limit the usability of rule extraction for deep learning models. In this paper, we adopt a new strategy to solve the problem. We first decompose a CNN into a feature extractor and a classifier. Then extract the decision tree only from the classifier. Then, we leverage lots of segmented labeled images to learn the concepts of each feature. This method can extract human-readable decision trees from CNNs. Finally, we build CNN2DT, a visual analysis system to enable users to explore the surrogate decision trees. Use cases show that CNN2DT provides global and local interpretations of the CNN decision process. Besides, users can easily find the misclassification reasons for single images and the discriminating capacity of different models. A user study has demonstrated the effectiveness of CNN2DT on AlexNet and VGG16 for image classification.

**Keywords** Rule extraction · Surrogate decision tree · Convolutional neural networks · Deep learning · Model interpretation

**Electronic supplementary material** The online version of this article (https://doi.org/10.1007/s12650-019-00607-z) contains supplementary material, which is available to authorized users.

S. Jia (✉) · P. Lin · Z. Li · J. Zhang
College of Intelligence and Computing, Tianjin University, Tianjin, China
E-mail: jsc_se@tju.edu.cn

P. Lin
E-mail: peiwenlin@tju.edu.cn

Z. Li
E-mail: lzytianda@tju.edu.cn

J. Zhang
E-mail: jwzhang@tju.edu.cn

S. Liu
School of Software, Tsinghua University, Beijing, China
E-mail: shixia@tsinghua.edu.cn

Published online: 01 November 2019

## 1 Introduction

Interpreting black boxes in machine learning (ML) especially deep learning (DL) is becoming urgent today (Hohman et al. 2018), due to their lack of transparency. End users who make critical decisions using ML need to understand how a model comes to its decisions; otherwise, they will not necessarily trust it (Ribeiro et al. 2016). Researchers also need insights into how black boxes derive knowledge and make decisions. To open the black boxes of DL, handful machine-centric (Zhang and Zhu 2018) and human-centric (Hohman et al. 2018) techniques have been proposed recently, thanks to the joint work of ML, computer vision, and Vis communities.

Among existing methods, training surrogate models such as rule lists, explanatory graphs or decision trees from ML models has been proven to be one effective way for interpretation (Yao Ming et al. 2019). The basic idea is that, given a black box, find one interpretable model to approximate its behavior as closely as possible. Since this method does not consider the structure of black boxes, it is well suitable for non-experts of ML (Hohman et al. 2018). Therefore, several rule extraction methods (Craven and Shavlik 1994, 1996) have emerged to interpret neural networks for the previous two decades. Until now, researchers have developed different customized methods for specific DL models including the distillation-based method (Wu et al. 2017) for recurrent neural network and gradient-based method (Zhang et al. 2018) for the convolutional neural network (CNN). In this paper, we focus on tree-based rule extraction method (Sato and Tsukimoto 2001) since it is a universal and relative mature technique, and surrogate decision trees are more close to the decision-making of humans compared with other formats.

Rule extraction is a universal method theoretically. However, because of the innate complex and non-linear structure of DL models, directly applying rule extraction on DL models is unpractical. On the one hand, surrogate decision trees extracted from CNNs end-to-end will be too large to interpret. Even for the customized gradient-based method for CNN, the derived decision tree has over 100 layers (Zhang et al. 2018). On the other hand, rule extraction is more suitable for models whose inputs are interpretable features, because the input describes the rules. For example, we can train a surrogate decision tree from CNN used for image classification. However, the rules of each node will be based on the color of each pixel of given images, which is unlikely for humans to interpret. These two drawbacks limit the usability of rule extraction for DL models.

In this study, we propose a new strategy to solve the problems. We first decompose a CNN into a feature extractor and a classifier. The feature extractor consists of several convolution layers and pooling layers to

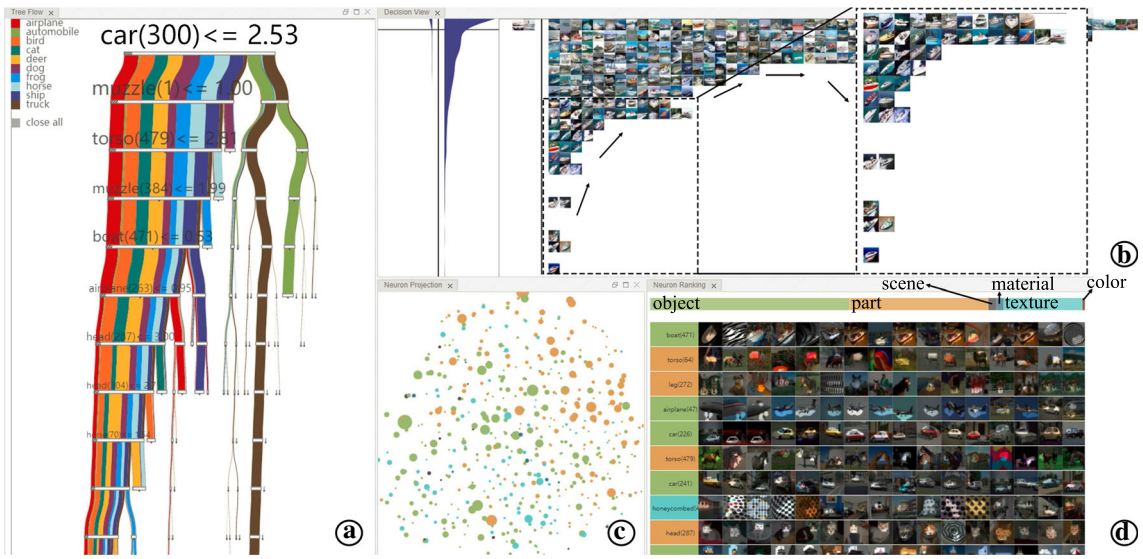


**Fig. 1** CNN2DT: A multi-view visualization system used to interpret the decision process of convolutional neural network (CNN): **a** TreeFlow is a collapsible tree visualization that shows the data flow through the surrogate decision tree of CNN. **b** Decision view interprets rules of each node of TreeFlow. It contains a thumbnail of the data distribution (left) and the corresponding image histogram (right). In this example, arrows stand for the main directions of ships. **c** Neuron projection projects neurons based on the semantics similarity into the 2D plane. **d** Neuron ranking provides the semantic labels and salience maps for each neuron

extract features from input images. These features will be used as input of the classifier, containing several fully connected layers. Instead of extracting rules from a CNN end-to-end, we extract rules only from the classifier, which significantly reduces the sizes of rules. Then, we match each feature with human-readable semantic labels using network dissection technique (Bau et al. 2017). This strategy alleviates the drawbacks described above, and the surrogate decision trees will be human-readable. Based on that, we propose CNN2DT (Convolutional Neural Network to Decision Tree), a visual analysis system (Fig. 1) to interpret the decision process of CNNs using extracted surrogate decision trees and maintain the interpretability in a manner that is similar to human decision-making. Although CNN2DT is designed for ML practitioners who are not DL experts, we believe it can also be helpful for other practitioners with different level of ML knowledge. In conclusion, our contributions include twofolds:

- A new framework that transforms CNNs into semantic-level decision trees
- CNN2DT, a visual analysis system that enables users to explore the surrogate decision trees

## 2 Background

Our work does not require users to have the fundamentals of DL and structures of CNNs. However, understanding the implementations of CNN2DT involves some knowledge of CNNs. We include the least essentials of CNNs in the following.

*CNNs* are deep feed-forward artificial neural networks, which have been successfully applied to computer vision. A CNN can be viewed as a combination of *feature extractor* and *classifier* (Fig. 2a). The feature extractor consists of multiple layers of *convolution* and *pooling*, whereas the classifier is composed of several fully connected layers. The feature extractor is a mapping from input space $X$ to feature space $Y$. Then, the classifier transforms features to output space $Z$. As we go deeper in the convolution layers, features will be transformed from low level (such as colors, shapes, and textures) to high-level (such as objects, parts, and scenes) semantics (Zeiler and Fergus 2014).

*Rule extraction* aims to provide the entire decision process of models in a rule-based manner. Given a black box $g$ and the data on which it was trained, rule extraction finds another more interpretable model $g'$ to approximate the behavior of $g$ (Huysmans et al. 2006), that is $g' = \arg\min_{g'}\{D(g,g') + \alpha C(g')\}$, where $D(g,g')$ evaluates the difference between $g$ and $g'$, $C(g')$ evaluates the complexity of function $g'$ and parameter $\alpha$ balances $D(g,g')$ with $C(g')$. Different algorithms may have different strategies to balance the two.

Existing methods convert the trained model into four kinds of rules: *if-then rules*, *M-of-N rules*, *decision trees*, and *decision diagrams*. Selecting which representations is not the focus of our paper since there are several debates on which one is better (Lakkaraju et al. 2016). Therefore, we limit our output representation as decision trees, because they are intuitively easier to understand and closer to the decision-making of humans.

## 3 Related work

*Visualizing and Understanding DL.* We divide existing visual analysis techniques for DL based on design space as follows. For a more comprehensive survey, we recommend users to read these works (Hohman et al. 2018; Choo and Liu 2018; Sacha et al. 2019).

Network-based visualizations reveal the structures inside DL models to open the black boxes. Node-link diagram (Tzeng and Ma 2005) is first used to show the neural network in which nodes stand for neurons and links represent weights. Liu et al. (2017c) improved the DAG layout to address serious visual clutter for CNNs. *ActiVis* (Kahng et al. 2017) presents computing graphs to help users understand the structure of DL models and find interesting layers to explore. *Graph Visualizer* (Wongsuphasawat et al. 2017) and *DGMTracker* (Liu et al. 2017) both show the data flows of models. These works aim at opening the black boxes of neural networks by visualizing their structures. Instead, we adopt a different strategy by converting neural networks into surrogate decision trees.

Mask-based visualizations add masks (heatmaps or salience maps Dabkowski and Gal 2017) onto images to highlight features that neurons or kernels have learned. These visualizations give insights into what neurons have learned, but understanding hundreds of neurons can be time-consuming. Therefore, *network*

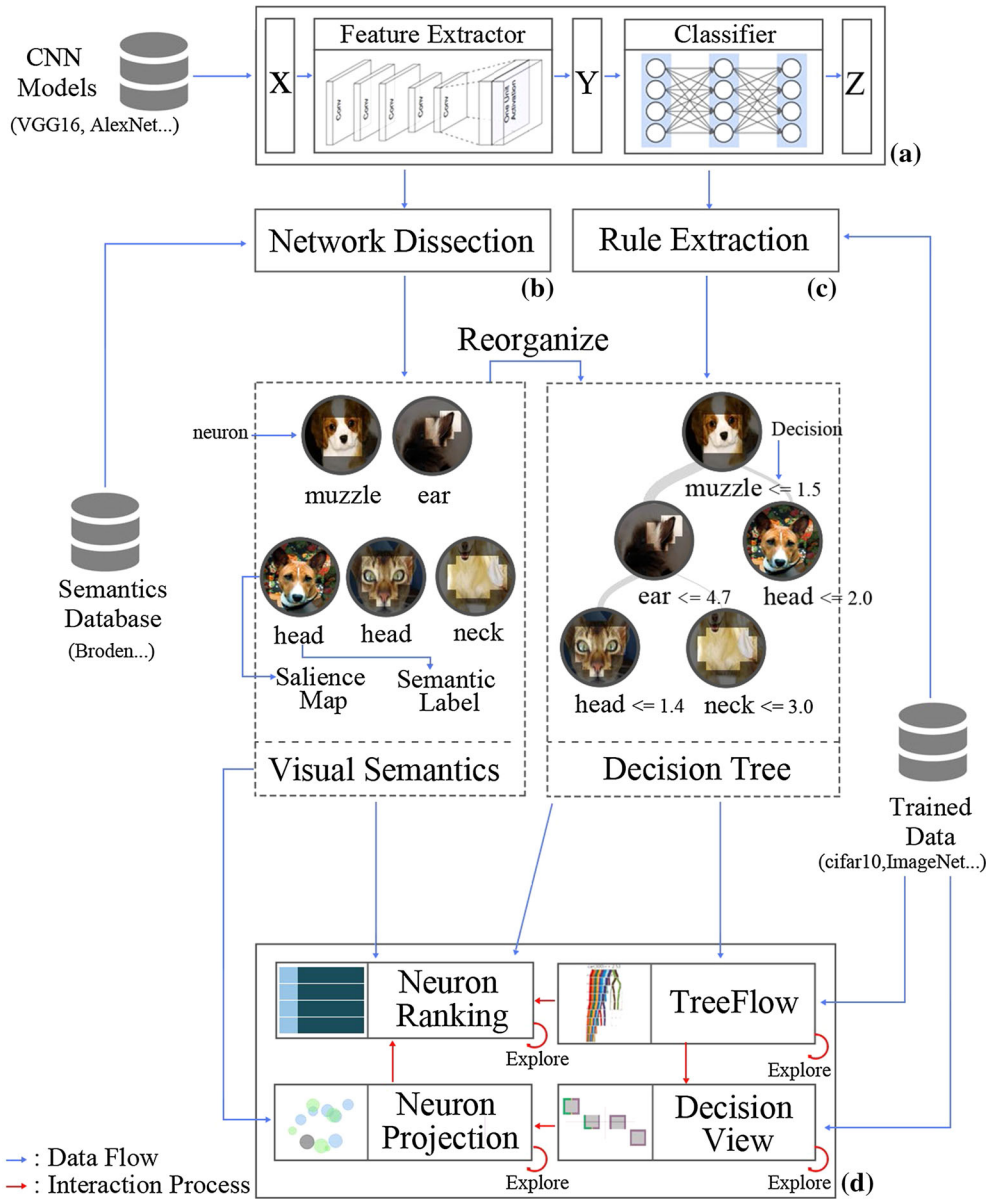


**Fig. 2** Pipeline to visualize surrogate decision tree of CNN: **a** Generalized CNN model includes feature extractor, classifier, and three corresponding spaces *X*, *Y*, and *Z*. **b** Network dissection (Bau et al. 2017) distills visual semantics of each neuron given a specific layer using a semantics database. **c** Rule extraction learns a surrogate decision tree from the classifier. **d** Multi-view visualization is designed to improve interpretation of the global and local decision processes of CNN

*dissection* Bau et al. (2017) proposed to match representations of CNNs with *semantic labels* automatically. Given any CNN model, it can label visual semantics onto hidden units at each intermediate convolutional layer. We adopt this technique to label neurons in the last convolution layer so that the surrogate decision tree will be human-readable.

Projection-based visualizations use dimensionality reduction techniques to project high-dimensional data points or neurons onto 2D or 3D spaces so that the relationships among them can be captured. Rauber et al. (2017) used *t-SNE* (Lvd and Hinton 2008) to visualize the relationships between learned representations of observations and the relationships between neurons. *ActiVis* (Kahng et al. 2017) and *DeepEyes* (Pezzotti et al. 2017) adopt the same technique to show instance activations. In this work, we project neurons based on their semantic similarities onto a 2D plane for hypothesis testing.

Instead of opening the black boxes, other techniques close the black boxes and visualizing the surrogate models to approximate the behaviors of the original models either globally or locally. Yao Ming et al. (2019) propose *RuleMatrix* as a visual interface to understand neural networks and SVMs using extracted rule lists. However, their method cannot be directly used in DL models such as CNNs. Wang et al. propose DeepVID 2019 to inspect the image classifiers using distillation-based methods locally. In this work, we propose a new strategy to extract global human-readable decision trees from CNNs.

*Decision tree visualization* Several main visualizations can be considered in representing decision trees, including node-link diagram, TreeMap, icicle plot, and sunburst. Node-link diagram (Han and Cercone 2001) can perfectly show the logic of a decision tree, in which each node stands for one decision rule, and each link connects parent and child nodes. TreeMap (Asahi et al. 1995) recursively divides a rectangle based on the proportion of child node quantity in the parent node. TreeMap is efficient in conveying node size but makes it difficult to compare nodes in the same level. Icicle plot (Liu and Salvendy 2007) divides nodes level by level based on the quantity proportions of nodes. It can also be displayed in radial layout, namely, *sunburst* (Mansmann et al. 2012). Icicle plot and sunburst can convey the clear hierarchical relationship of nodes, but the decision boundaries are not explicit.

Other methods usually combine the node-link diagram and icicle plot to show both hierarchical logic and data flow. For example, *BaobabView*(Van Den Elzen et al. 2011) enables domain experts to construct and analyze decision trees interactively. Each link consists of different colored brands according to the class, and the width of brands corresponds to the quantity proportions. *TreePOD*(Muhlbacher et al. 2017) and *BOOSTVis*(Liu et al. 2017b) adopt this method, in which the nodes show the decision rules. In this work, we propose TreeFlow to visualize surrogate decision trees using similar design as BaobabView. The key difference is that TreeFlow leverages a different layout algorithm to satisfy aesthetic rules.

## 4 CNN2DT

### 4.1 Overview

The main purpose of CNN2DT is to obtain insights from the surrogate decision tree. We aim at the following analysis tasks for *model interpretation*. These tasks are generated gradually after discussions with DL experts. Specific tasks with this objective include the following:

*(T1) Interpretation of surrogate decision tree* can be analyzed in the quick overview and details, so that users can gain the whole knowledge of the decision process.

*(T2) Interpretation of the visual semantics* learned by CNNs should indicate the similarity among the semantics so that users can understand relationships of neurons.

*(T3) Interpretation of the decision rules* should be conducted in a data-driven way so that users can evidence reasonable or irrational decisions.

*(T4) Interpretation of the reasons behind the misjudgment* so that users can learn the imperfection of models and the orientation for improvement.

The analysis process needs data preprocessing for visual semantics extraction (Fig. 2b) and decision tree extraction (Fig. 2c). Analysts should first upload a pre-trained model such as VGG16 including predefined architecture and trained weights files.

Visual semantics are extracted from the last convolution layer of CNN as the neurons in the highest layer have learned the most comprehensible visual semantics. In this study, we refer to visual semantics as textural semantic labels and feature visualizations (such as heat maps or salience maps) of neurons. Each neuron is annotated with one most likely semantic label during forward-propagation and evidence sequence images together with corresponding salience masks. Salience masks highlight the regions where the neuron focuses on. Meanwhile, the rules using the neurons for classification can be extracted using any existing rule extraction algorithm. We choose the tree-based rule extraction because the decision trees are closer to human decision-making. In this way, the surrogate decision tree uses semantic labels to divide data, in which each node stands for one decision rule.

It is a challenging task to find patterns from the extracted surrogate decision tree. We show one example visualized by Graphviz in Fig. 3. First, the number of extracted rules is enormous, and simple visualization such as the graph layout of these rules will result in a mess. A more effective method is required to provide a quick overview and drill-down explorations. In this condition, we implement compacted visualization *TreeFlow*, to show the data flow through decision tree (T1). Furthermore, TreeFlow can be collapsed,


**Fig. 3** Surrogate decision tree with more than 2k nodes extracted from last three fully connected layers of VGG16, visualized by Graphviz (Ellson et al. 2004): **a** One decision node of the tree: when in this zoom of A4 paper, the whole plot will be 11.5m $\times$ 0.9m. Understanding such a huge decision tree is indeed a great challenge if without the help of effective visualization techniques

zoomed, selected by data class, and linked with other views. To give a close inspection of each rule of the surrogate decision tree, we design *decision view* to provide a summarized data distribution using a smoothed histogram and detailed image distribution (T3). *Neuron projection* shows the semantics similarity of different neurons. This view gives the context of what the selected neuron means inside the semantic map (T2). When users click an image in the decision view, neuron projection will switch to its activations of all neurons (T4). *Neuron ranking* visualizes what each neuron look at so that users can quickly understand the concept of each neuron.

### 4.2 Match semantic labels to neurons

We adopt network dissection (Bau et al. 2017) to probe visual concepts in the last convolutional layer. The basic idea of this technique is to calculate the degree of intersections between the activated image patches of each neurons with human labeled image patches. Each neuron is labeled using the same semantic labels of top activated images.

Network dissection uses Broden dataset to align the visual concepts. Broden is a broadly and densely labeled dataset including a broad range of concept groups including objects, scenes, parts, textures, and materials. Each group includes lots of corresponding semantic labels. For example, the object concept group contains the head, muzzle, eye labels, etc. Every image pixel in the dataset is annotated with several semantic labels.

Then, we apply all the images in Broden to CNNs using the only forward pass for every neuron in the last convolutional layer without any training or backpropagation. This step results in one activation map $A_k(x)$ for the input image $x$ given one neuron $k$. In order to compare with the labeled mask $L_c(x)$ for semantic concept c of image $x$, network dissection scales up the $A_k(x)$ using bilinear interpolation to $M_k(x)$, so that $M_k(x)$ has the same resolution as $L_c(x)$. For every $(k, c)$ pair, we calculate their dataset-wide intersection over union score

$$IoU = \frac{|M_k(x) \cap L_c(x)|}{|M_k(x) \cup L_c(x)|}$$

where |.| is the cardinality of a set. The higher IoU value, the more possible the neuron looks at the same concept. We then label each neuron using the semantic label that has the highest IoU value.

### 4.3 Extract surrogate decision tree from CNN

Rule extraction can learn surrogate decision tree from any black box using the training data labeled by the model. The basic idea is to use a greedy algorithm to choose one attribute that makes the data purest each time. Purity is usually defined using gain ratio or Gini index.

Gini index for training set $S$ containing data points from $C$ different classes is defined as $\text{Gini}(S) = \sum_{i=1}^{C}\sum_{i\neq j} p_i p_j = 1 - \sum_{i=1}^{C} p_i^2$ where $p_i$ and $p_j$ are the proportions of examples of classes $i$ and $j$ in the node. $\text{Gini}(S)$ stands for the probability that two samples randomly selected from $S$ have different labels. The lower the $\text{Gini}(S)$ is, the purer the$S$ is. Then, Gini index of attribute $a$ to split $S$ into two subsets $S_1$ and $S_2$ is $\text{Gini\_}index(S, a) = \frac{|S_1|}{|S|}\text{Gini}(S_1) + \frac{|S_2|}{|S|}\text{Gini}(S_2)$. The earlier chosen attribute or neuron has more discriminative power than the others and frequently used neurons are more useful. We define $D(n) = \sum \frac{1}{d}$ as the discriminative power of neuron $n$, where $d(d >= 1)$ is the depth of $n$ in the tree. $D(n)$ is useful when interpreting the semantics of neurons to see which neuron is more dominant during the decision process.

## 4.4 TreeFlow

### *4.4.1 Visual design and interactions*

TreeFlow provides a compact visualization of the surrogate decision tree. Weighted and colored brands run from top to bottom, showing data flow along decision tree. The widths of brands correspond to the proportions of samples of each class, and colors represent classes.

TreeFlow is collapsible during exploration, which means that the subtree expands or shrinks to one node. The collapsed bar (Fig. 4d) triggers a toggle when users click on it and shows the false positive (FP) and true positive (TP) portions in this node. The label above the collapsed bar combines the semantic label with the decision boundary to describe the decision rule. The font size of the labels shrinks from top to bottom to save space and prevent overlapping. Users can click on the detail button below the bar (Fig. 4e) to update the decision view for more details and roll up the corresponding neuron item in neuron ranking. The structure of TreeFlow reorganizes when users toggle switches given that users may only be interested in some specific classes. These interactions can provide users the freedom to explore the rules based on their interests and focus. Automatic tree cutting is used during interaction to further boost the visualization. In this way, TreeFlow becomes neater and clearer by filtering and closing the subtree in which the selected class has smaller samples than a predefined threshold.

### *4.4.2 Visualization alternatives and problems*

We choose the classic R-T algorithm to layout the tree skeleton (red lines centered at the bands, see Fig. 5). It is stated that the following four aesthetics are required to preserve a tidy drawing of trees using conventional R-T algorithm in the original paper (Reingold and Tilford 1981).

*Aesthetic 1.* Nodes at the same level should lie along a straight line, and the lines should be parallel.

*Aesthetic 2.* A left child should be positioned to the left of its parent and a right child to the right.

*Aesthetic 3.* A parent should be centered over its children.

*Aesthetic 4.* A tree and its mirror image should be symmetric. Moreover, a subtree should be drawn in the same way regardless of where it occurs in the tree.

Aesthetic 4 maintains the symmetry of trees in local and global ways, which is a stronger requirement compared with previous methods. Moreover, although it is based on the aesthetic consideration, Aesthetic 2 is necessary to visualize a decision tree. The growth directions of the left child and right child of a parent have semantics during interpretation, that is, true or false of the decision rules. Therefore, breaking Aesthetic 2 will lead to cognitive misleading.

Before the final design of TreeFlow, we have considered other visualization techniques. The first alternative (Fig. 5a) is the combination of node-link diagram and icicle diagram as it is widely adopted in

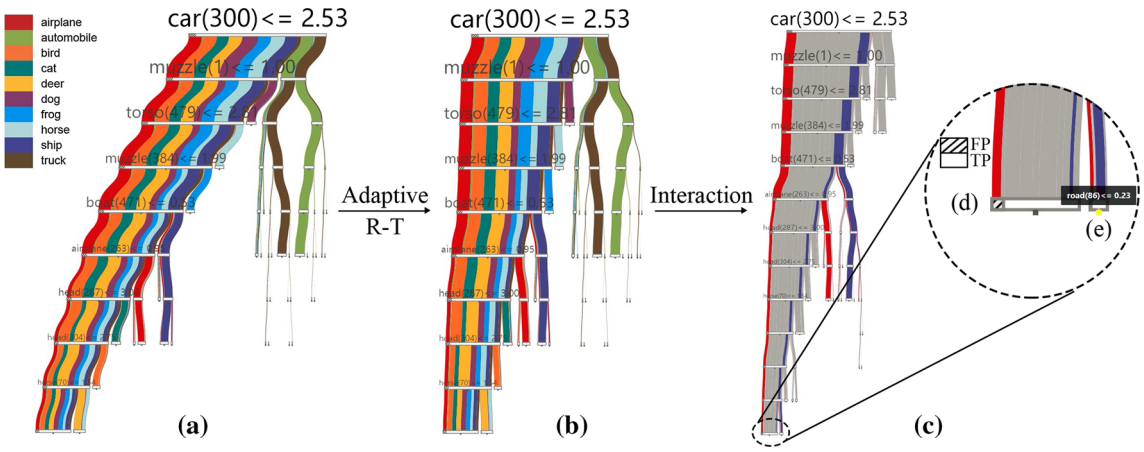


**Fig. 4** Tree Flow visualizes the rules in Fig. 3**a** Layout using conventional R-T algorithm violates Aesthetics 2 and 4, and causes cognitive misleading during interpretation. **b** Layout using adaptive R-T algorithm maintains Aesthetic 1–4 and provides an even tidier drawing. **c** By shutting off or turning on the classes, the structure of the decision tree reorganizes automatically and selected streams are highlighted. Automatic tree cutting is used to filter irrelevant or thin branches. **d** Collapse bar of one node and **e** detail button to link decision view

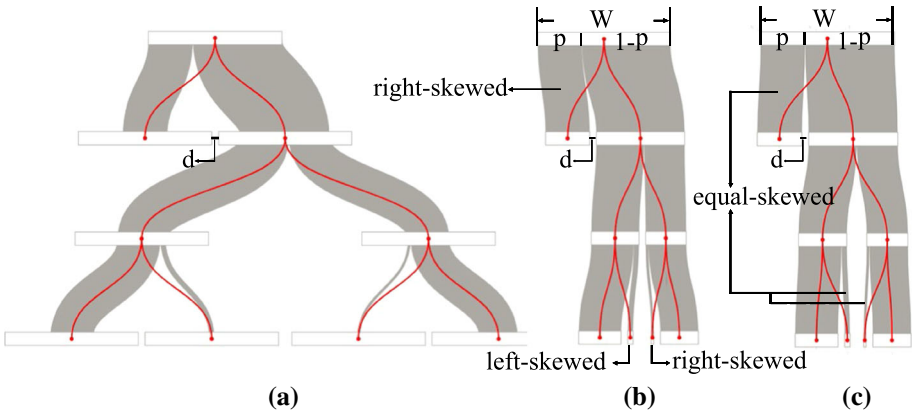


**Fig. 5** Visualization Alternatives and Final Design: **a** Visualization combining node-link diagram and icicle diagram wastes a large amount of space. **b** Visualization combining node-link diagram and Sankey diagram violates Aesthetics 2 and 4. **c** TreeFlow maintains Aesthetics 1–4 using adaptive R-T algorithm to produce the layout

the literature (Van Den Elzen et al. 2011; Muhlbacher et al. 2017; Liu et al. 2017b). However, after trying this technique, we find that it is not sufficient for large-scale decision trees. Because as the tree goes deeper, the occupied space increases exponentially with the fixed width of nodes. Therefore, we decide to change the widths of nodes to make them consistent with the link widths. That is, the second alternative is the combination of the node-link diagram and Sankey diagram. However, this alternative violates Aesthetics 2 and 4 (Fig. 5b) although the skeleton (red lines centered at the bands, see Fig. 5) does not. Therefore, we propose the adaptive R-T algorithm to distort the skeleton of trees (Fig. 5c), so that the flow (gray bands in Fig. 5) keeps Aesthetics 1-4. We leave the details of this algorithm in Appendix.

### 4.5 Neuron projection and neuron ranking

Neuron projection (Fig. 1c) helps users understand the complexity of neuron semantics and provides evidence of TP and FP classifications. It consists of a neuron semantic map (Fig. 1c) and an activation map (Fig. 6e), which can be switched based on the analysis tasks. An overview of the relationships among them is essential (T2) before diving into the visual details each neuron has learned.

We adopt the same technique (Rauber et al. 2017) to project the neurons into a 2D plane based on their semantic similarity. Each dot stands for one neuron; its color is the label category (object, part, scene, material, texture, and color), and its size represents $D(n)$. We use bubbleset (Collins et al. 2009) to highlight the regions of neurons with the same label, otherwise small-size neurons may be neglected (Fig. 7).

An activation map can be transferred from a semantic map when users click an image in the decision view. This map shows the activations of all neurons, in which the position of the dot is consistent with a semantic map, and the lightness corresponds to activation intensity. The activation map is useful when linked with neuron ranking to interpret misclassification (T4).

Neuron ranking (Fig. 1d) provides semantic summarizations and details for neurons, and it is designed for hypothesis testing during exploration. The top bar shows the proportion of neuron categories, and the bar below shows the semantic list. Each item in the list indicates the category (color), label, and evidence image sequence with salience maps for details. The list is ordered by $D(n)$. The corresponding neuron item rolls up for attention when a node in TreeFlow or one dot in neuron projection is selected.

### 4.6 Decision view

Decision view (Fig. 8) interprets each decision rule of TreeFlow (T3) when users select the detail button as shown in Fig. 4e. The left part of the decision view is the *decision flow* showing the distribution of selected classes from top to bottom with horizontal decision boundary. The left side of the decision flow shows the distribution of FP data, whereas its right side shows TP data.

Corresponding to the decision flow as a thumbnail, an *image histogram* at the right part provides the details of each data point. Image histogram separates images by the activation values of the selected neuron from top to bottom. The activation of images at the same line increases from the middle to the left for FP

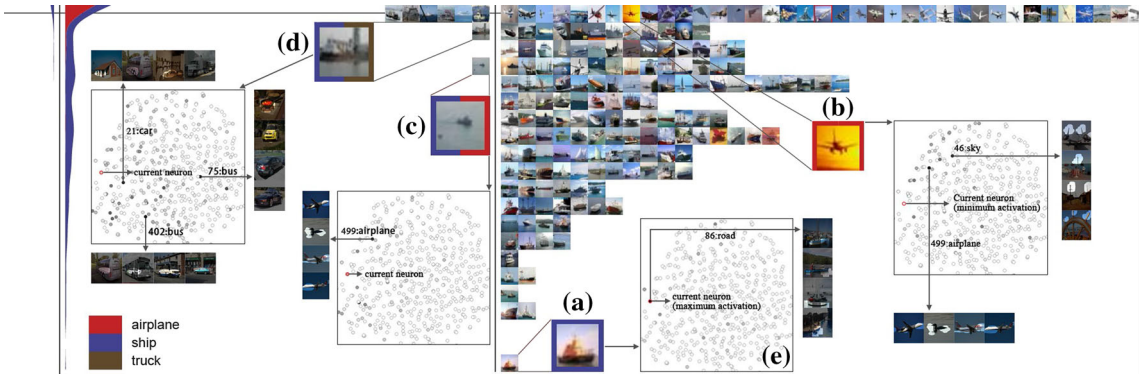


**Fig. 6** Interpreting decision rule with neuron 86 labeled as the road using decision view, neuron projection and neuron ranking: decision-making for four instances **a**–**d** to interpret the TP data below the decision boundary (**a**), above the decision boundary (**b**) and FP data (**c**, **d**); **e** is the activation map of (**a**)

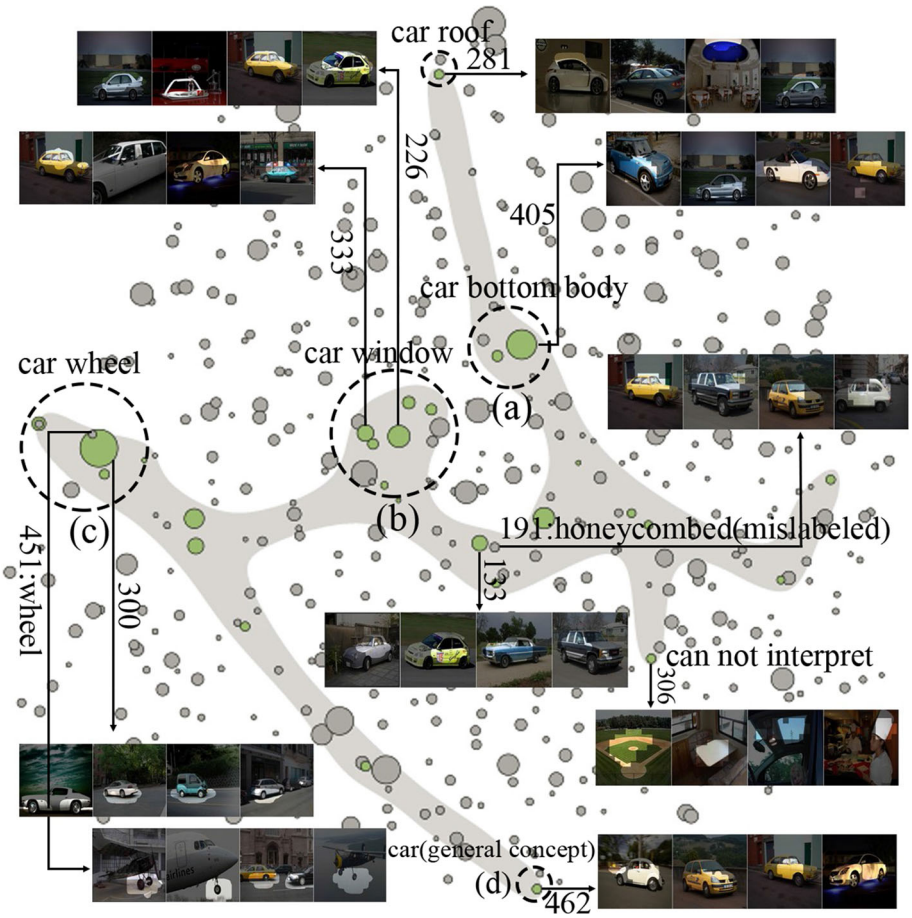


**Fig. 7** Semantic map of neuron projection: neurons labeled as the car have different visual semantics and clusters in subgroups

data or to the right for TP data. We design a border glyph to discriminate the true label (left border) and predicted label (right border) when the mouse moves over each image. Moreover, for images overlapping the decision boundary, stands for the upper one, whereas stands for the lower one.

A particular procedure when interpreting the decision rules (T3) and the misclassification reasons (T4) is as follows. The user first explores TreeFlow for rule overview and finds one decision node s/he is interested in (T1). S/He clicks the detail button to update the decision view (Fig. 2b) and rolls up the corresponding neuron in the neuron ranking automatically (Fig. 2a). Based on the label and neuron features in neuron

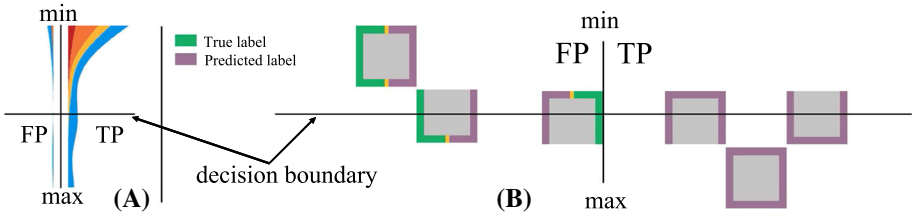


**Fig. 8** Decision view illustration: **a** Decision flow overviews the distribution of activations of selected FP (false positive) and TP (true positive) data (different colors stand for different classes). **b** Border glyph when the image is on, above, or below the decision boundary. The color of the left border (green) stands for the true label, and the color of the right border (purple) stands for the predicted label. Images distribute along the vertical axis based on their activations of selected neuron from top to bottom. Images on the left of the vertical axis stand for FP data, while those on the right stand for TP data

ranking, s/he finds a possible explanation given that most of the images below the decision boundary have features described in neuron ranking, while those above do not (T3). For the data that cannot be interpreted well, the user can obtain their activation maps by clicking the image (Fig. 2c). By checking and understanding the semantics of dominant neurons through the link between neuron projection and neuron ranking (Fig. 2d), s/he obtains more insights and the direction for next exploration.

## 5 Case study

### 5.1 Overview and implementation

We choose VGG16 (Simonyan et al. 2013) and AlexNet (Krizhevsky et al. 2012) as benchmarks for experiments because they are the classic models to complete image classification. We choose Cifar10 (Krizhevsky and Hinton 2009) as input and split it into a training set with 50k images and a test set with 10k images. Section 5.2 illustrates the interpreting process of VGG16 comprehensively based on analysis tasks (T1–T4). Section 5.3 extends the interpreting process to AlexNet and compares the two surrogate decision trees. We reduce the number of filters for both models because it is sufficient for ten classes. We train and test the two models using a deep learning framework, Caffe (Karpathy et al. 2014). VGG16 gained 89.5% accuracy on the test set, whereas AlexNet obtained 85%.

Second, we use network dissection (Bau et al. 2017) to label the neurons of the last convolution layer. Network dissection uses Broden to probe convolutional layers by evaluating the performance of each neuron on various vision tasks. Broden contains a broad range of objects, scenes, parts, textures, materials, and colors in a variety of contexts (Bau et al. 2017). Each of the six categories consists of different types of fine-grained classes. Finally, each neuron is labeled with most likely visual semantics.

A decision tree is then extracted using CRAT (Breiman et al. 1984) from the classification result of models. We select CRAT because it has always been used as a benchmark and can gain approximately 75% fidelity which is sufficient for exemplification. Researchers can switch to any tree-based rule extraction algorithm for experiments. CART constructs binary trees using the feature and threshold that yields the largest Gini index at each node, and we choose the optimized version of it from scikit-learn (Pedregosa 2011). Based on the surrogate decision tree and visual labels on each tree node, we visualize the results using the design discussed in Sect. 4.

### 5.2 Use case 1: interpreting surrogate decision tree

*Neurons with the same labels have rich semantics (T2).* We look at what neurons have learned first before diving into the rules extracted from CNN. The semantics have no sensible clustering in Fig. 1c, and neurons with higher $D(n)$ seem to have a random distribution. We find the reasons after examining the salience maps with the same semantic label. Semantic labels are coarse-grained and therefore cannot describe the features precisely. Even with the same label, neurons have different fine-grained semantics that self-cluster in subgroups. For example (see Fig. 7), the visual semantics of different car parts generate subgroups and are located in different places, such as car roof centered at the 281st neuron, car bottom body (405th neuron as a representative), car window grouped in Fig. 7b, car wheel in Fig. 7c, and the general car semantic of the neuron 462.

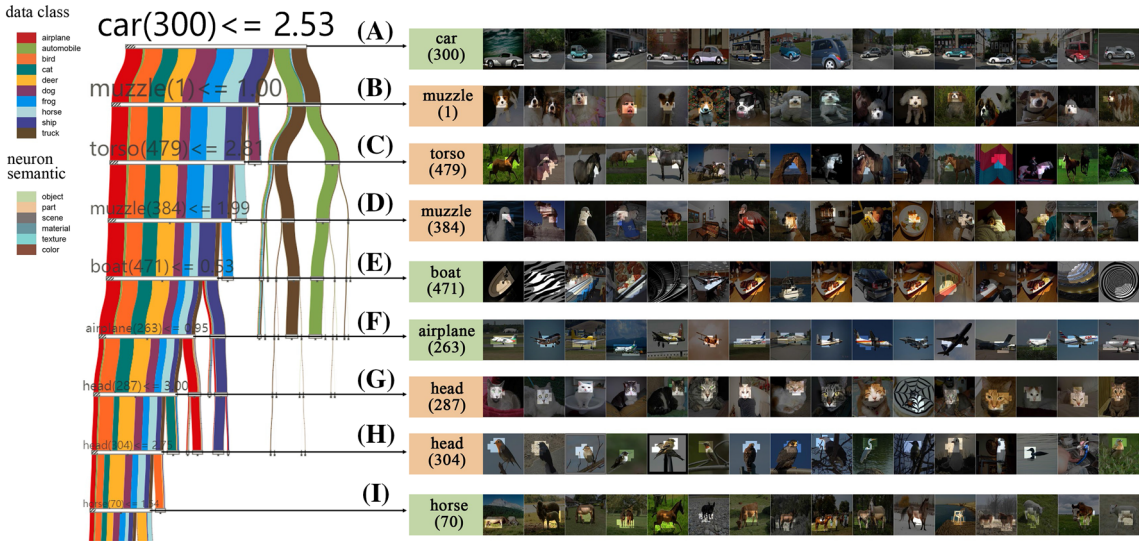


**Fig. 9** TreeFlow of surrogate decision tree extracted from VGG16 with salience maps (A–I) to interpret each visual semantic of decision nodes

Neuron 300 labeled as a car is surrounded by wheel neurons, such as neuron 451. This makes sense because the neuron 300 focuses on the wheels. We can judge the mislabeled neurons via neighbor neurons. For example, neuron 191 is labeled as honeycombed because the features it learned are grid-like. After exploring the semantic map, we learned the complexity of the relationships among neurons and the richness of the same visual semantics. This situation may point out that quantifying the visual semantics of each neuron is fascinating, but interpreting the visual semantics still requires visualization.

*Interpret decision process using TreeFlow (T1).* TreeFlow shows that each decision rule applies a condition in which one semantic is less than or equal to one threshold, which means that distributed data in the right branches of decision nodes are the images that activate the neurons. One interesting pattern (Fig. 9) is that the mainstream of TreeFlow, located at the leftmost part, splits one or two main classes to the right branches using one semantic, and the right branches have a high degree of purity. That is, the right branches separate the most different classes from the left.

Node A applies neuron 300 as a decision rule labeled as a car. It separates automobile and truck apart from the left branches. Note that automobile and truck are the only two classes that belong to cars. The features show that the salience maps focus on cars exactly, and on wheels more accurately. Node B separates dog images apart using neuron 1 labeled as the muzzle. The neuron focuses on the dog's face. A child's face has sneaked into the image sequence because it has similar features to a dog's face. Node C differentiates images of horse and others using torso neuron, or the intersection part between the horse head and neck accurately. An interesting observation in the salience maps is that one image containing something pink and blue just like a roof is mixed in. The reason is evident because all their shapes are similar to an upside-down V.

Although the previously discussed decision nodes make sense, the decisions of Node D are not intuitive. Finding the similarity of the salience maps through the node labeled as a muzzle is difficult because Broden lacks the visual semantics of frogs. And adding the human labeled semantics will enhance the interpretability. Node E labeled as boat divides ships apart, but only one ship image appears in the sequence. It seems that the neuron looks for curves from the bottom left to the top right. We open the decision view from ship brands (Fig. 1b) and find that images are classified by the direction of the boat bottom border, which confirms the assumption.

*Interpret decision rules using decision view (T3).* After the overview, users may want to obtain more insights into the decision rules. Users can reorganize the structure of TreeFlow to filter the irrelevant branches by toggling the classes. For example, TreeFlow only highlights the streams of airplanes and ships and makes other steams gloomy or closed as shown in Fig. 4c. The decision view is updated (Fig. 6) when users click on the detail button (Fig. 4e).

TreeFlow uses neuron 86 labeled as a road to separate airplane and ship images and draws a decision boundary between the border of two streams (Fig. 6c). Most of the ship images are divided below the

decision boundary and airplane images on the top. This neuron seems to search for the interfaces between boats and water (see the salience maps of neuron 86 in Fig. 6). Notice that all images below the decision boundary have the features exactly.

*Interpret the reasons for TP and FP decisions (T4).* Take instances of Fig. 6a, b for further exploration and evidence. Image (a) is labeled as ship by CNN correctly and obtains the maximum activation of neuron 86 compared with others. Image (b) is labeled as airplane correctly. The activation map of this image shows the dominant neurons, such as neuron 46 labeled as sky and neuron 499 labeled as airplane. We can compare the features in Image (b) and the salience maps of neuron 46 and find that neuron 46 actually searches for spikes in the sky and not the color of the sky.

Throughout the aforementioned case, using CNN2DT cannot only provide global (T1) and local (T3) interpretations of the decision process of CNN but also explanations for single images. CNN2DT can interpret the misclassifications (T4) with the assistance of multi-view visualizations, take images (d) and (c) as examples. Image (c) is mislabeled as airplane although it slightly activates neuron 86. The reason is that it has the features of airplanes because it activates neurons 499 the most. Image (d) is mislabeled as truck because the higher activations come from the neurons that searching for features of cars and buses.

## 5.3 Use case 2: comparing surrogate decision trees

*Compare surrogate decision trees of VGG16 and AlexNet.* The structure of the surrogate decision tree of AlexNet is different from that of VGG16. Surrogate decision tree of AlexNet has two main streams apart from neuron 386 and has more branches with deep depth and low purity (Fig. 10). Although AlexNet and VGG16 use the wheel semantics to separate images at the first layer, the neuron chosen by AlexNet focuses explicitly on the wheels of automobiles, whereas VGG16 uses a general neuron labeled as the wheel to separate both automobiles and trucks.

AlexNet uses neuron 386 labeled as torso to separate the left stream into two groups. The left mainly contains vehicles, whereas the right is primarily composed of animals. Thus, AlexNet chooses several neurons (wheel, head, torso, and car) which are similar to those of VGG16 to separate classes gradually. For example, neuron 408 divides deer apart from the animal group using head features, and the salience maps in view G show that it focuses on the regions that contain the structures of bifurcations including tree branches and cacti. It makes sense given that deers have antlers on their heads, which is similar to the features of tree branches and succulents.

We found that AlexNet uses extra texture and color neurons to separate classes in the first nine prominent neurons(A-I), whereas VGG16 uses object and part neurons. For example, neuron 165 uses a perforated texture to separate frogs. AlexNet uses blue color (neuron 150, see I in Fig. 10) to distinguish most of the ships. This may be effective in this context but may be rough and dangerous sometimes, for example, in the field of self-driving. Besides, some neurons AlexNet used are less discriminative compared with those of

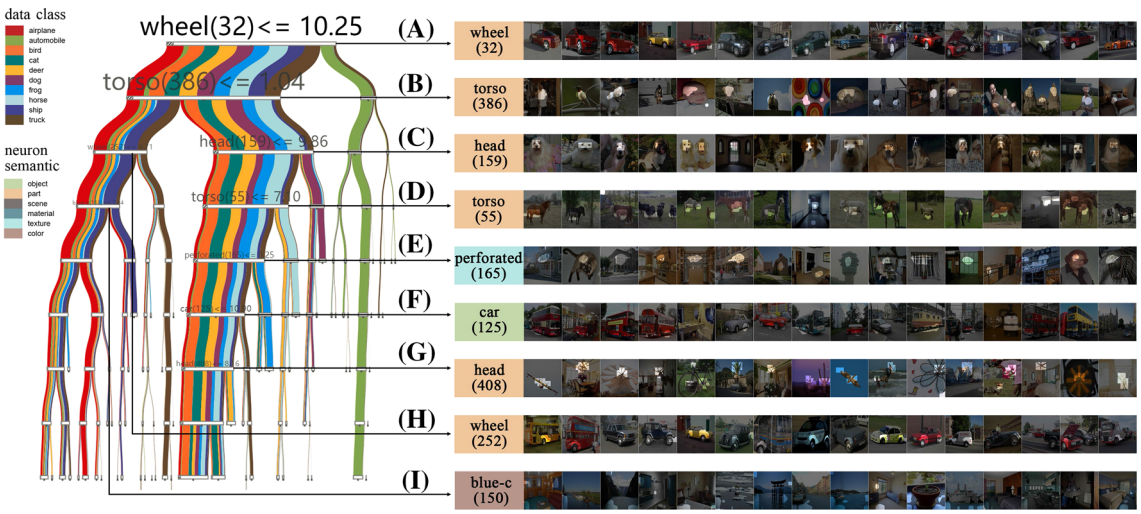


**Fig. 10** TreeFlow of surrogate decision tree extracted from AlexNet with salience maps (A–I) to interpret each visual semantic of decision nodes

VGG16. VGG16 use more accurate and specified neurons to separate images, whereas AlexNet sometimes uses lower-level and general features, which makes the decision paths tortuous.

### 5.4 Use case 3: train samples versus accuracy and fidelity and rule size

This section evaluates the stability of our method when using different train samples to extract rules. We randomly select 10k, 20k, 30k, 40k, and 50k samples from Cifar10 as train data, and the others as test data with each group 100 times. The five groups are used to evaluate the fidelity (true positive rate according to predicted labels of CNN), accuracy (true positive rate according to the true labels of images), and node size of the extracted decision trees (see Figs. 11a, b).

Fidelity and accuracy grow slowly almost linearly when samples size increases, and both are between 0.7 and 0.8 in Fig. 11a, whereas the node size grows faster linearly. The amplification is approximately 20k. Fidelity, accuracy, and tree size all have low deviations in each group. Increasing the sample size has a low effect on the fidelity and accuracy but can increase the tree size significantly. Therefore, we choose 10k samples as train data in our experiments.

We choose five results for contrast, in which the number of train samples is 10k, as shown in Fig. 12. The result in Fig. 12c is the same as that in Fig. 9 as a benchmark. The main structures of TreeFlows are much similar, and all have the mainstream leftmost with one sub-stream apart as the right branch with high purity. They divide automobiles and trucks first using neuron 300 labeled as a car and separate dogs apart using the first neuron labeled as a muzzle or neuron 14 labeled as a dog. Following the two levels, the TreeFlows use slightly different orders to separate images with almost the same neurons, and some exchange one or two neurons to separate. For example, (c) and (d) trees use the same neurons in the first nine layers but with different orders, whereas (c) and (b) trees use almost the same neurons except the second and the third layers, in which neuron 479 extracts the intersection part between horse head and neck as shown in Fig. 9c, whereas neuron 412 with the same label seems to extract the belly or forelegs of a horse.

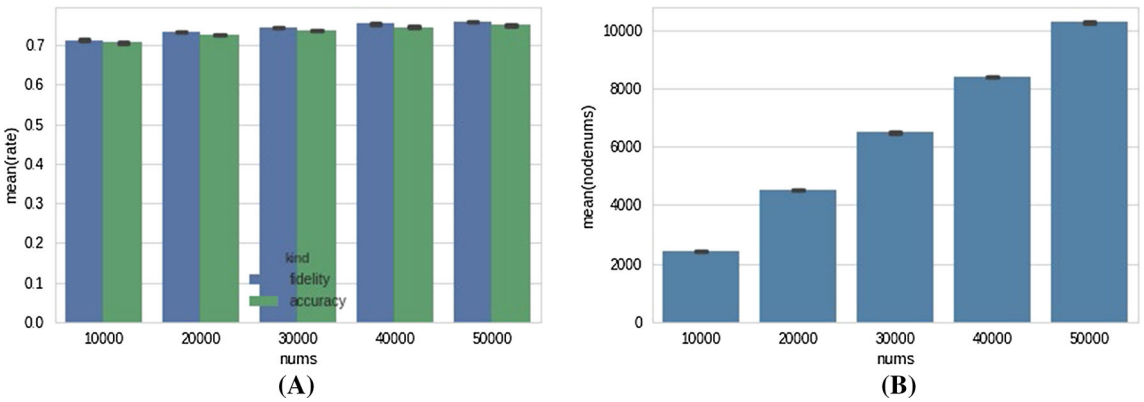


**Fig. 11** **a** Fidelity and accuracy influenced by train samples. **b** Node number influenced by train samples

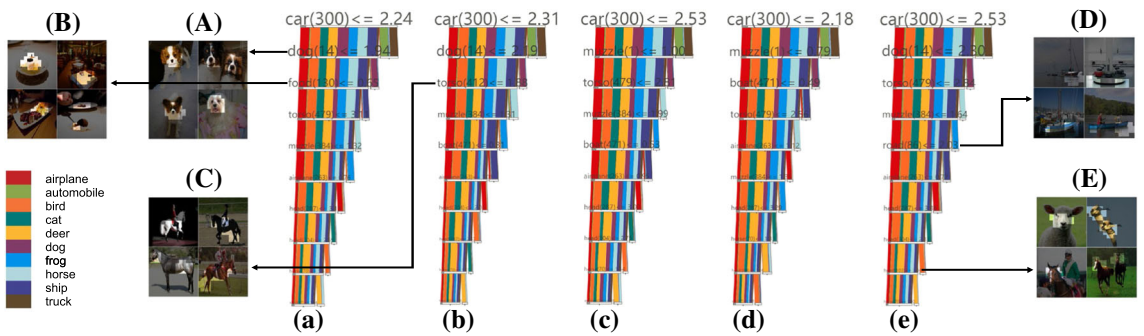


**Fig. 12** Five results **a–e** from VGG16 using random 10k samples and extra neurons **A–E** different from those in Fig. 9

## 6 User study

### 6.1 Study description

We conduct a user study to evaluate the usability of CNN2DT. We invite ten participants with different levels of experience of ML knowledge including: A1–A2 (two Ph.D. students who do DL-based computer vision research), B1-B3 (three ML masters who have two or three years developing ML models but are not experts in DL), and C1-C5 (five undergraduates who have little knowledge of ML).

There are four females and six males with their age range from 20 to 32. Each participant receives a flash disk as a reward for the study. Before the study, each person should fill in an intro questionnaire to provide the demographic data and information about how much ML knowledge they have. Then, we give each participant a video to introduce the background, motivation, and the primary usage of our system. This video lasts about ten minutes. During this process, each person is allowed to raise questions about the CNN2DT. Then, each participant logs in our system deployed in the cloud and gets familiar with the system as a warm-up. We give each person ten minutes to warm-up. Then, we provide each participant 30 minutes to explore the interpretations of VGG16 and AlexNet using CNN2DT freely and records what insights they have found during the session. At the end of the study, participants completed an exit questionnaire to provide feedback on CNN2DT.

### 6.2 Key observations

*TreeFlow for overview, decision view for details, neuron projection for hypothesis testing.* All participants explored the TreeFlow first. They clicked the rules one by one to check the mask images in the neuron ranking and tried to understand what the rules mean. Often, they only checked the first ten layers of the surrogate decision tree and quickly understood the whole decision process. Most participants mentioned TreeFlow is compact and easy to understand. C3 mentioned before the study, he even did not know what decision tree was, but after we explained it using an example, he quickly got it and said it quite follows the human-thinking. After checking the whole rules, most participants began to choose their interests to focus on specific classes. B2 selected the dog and cat and she found VGG16 confuses these two species a little bit. She clicked the corresponding details buttons to check the data distributions of each node. She found that VGG16 can classify dog and cat images with clear full-frontal faces in the center of images but confuse them when they are sideways. Some participants noticed that some labels are not matched well with the mask images. Instead, they usually choose neighbor semantics as interpretations in the neuron projection.

*Application to model comparison.* When switching to AlexNet, all participants mentioned the logic was messer than that of VGG16. B2 said that AlextNet had three mainstreams and lots of thin branches. He said it seems Alex does not classify the images well. A1 and A2 commented that they loved the comparison of two models. They said researchers usually use confusion matrixes to compare the performances of different models, but they do not know why models make different decisions. They continued that CNN2DT transforms this problem into the comparison of structures of trees, and users can quickly grasp the logic uncertainty of different models.

### 6.3 User feedback

Overall, all participants mentioned that they enjoyed using CNN2DT, and it was well-designed and enabled them to explore freely. Most participants commented that visualizing a CNN model into a decision tree makes the logic of decisions much clear. After the study, we asked participants to critique CNN2DT and suggest improvements.

*Suggest interesting rules to investigate.* B2 and B3 both mentioned that although exploring the large surrogate decision tree is impressive, examining all the rules by humans will take patience. It would be better if the system automatically suggests which rules should investigate and generates possible interesting reports to users and leave users to check.

*Enable users to add test images.* A1, B2, C3, and C5 all mentioned that sometimes the labels are not consistent with the mask images. It is not clear whether it is due to the un-interpretability of neurons or the lack of semantics of test images. It would be better if the system can enable users to add their images to test the hypothesis.

## 7 Discussion

*Interpretability*. Two differences between CNNs and decision trees should be noted because these cause limitations using our method.

(i) CNN uses nonlinear transformation, which can learn the complex decision planes, whereas decision tree is limited by the axis-parallel splits, which can only approximate the decision planes.
(ii) CNN can classify data considering all the attributes in a parallel manner, whereas decision tree separates data based on one attribute at each decision node.

The working mechanism of decision tree is similar to the way humans think, and it is different from that of CNN. Therefore, there is still a long way to explore how far can a simplified decision tree representation to convey the workings of a high-dimensional and nonlinear CNN model. Moreover, semantic database (Broden) may lack some visual semantics in that ambiguity can emerge when interpreting what neurons have learned. For example, in Fig. 9, the salience maps seem to be less interpretable because Broden lacks the images for frog parts, although neuron 384 can differentiate the frog apart from the other classes. Researchers can add more semantics to enrich Broden, which then enhances the interpretability of CNN2DT.

*Scalability* CNN2DT in this paper only visualizes the rules extracted from the last convolutional layers of basic CNNs, and it uses color to encode a small number of data classes. Rules will be incredible abstract and complex if we extract them from low-level layers of more complex CNNs with more data classes. It is unclear whether users can still get a whole idea of extracted rules. We leave further explorations as the furture work.

*Stability* The visual results can be influenced by train samples shown in Sect. 5.4, in which as train samples grow linearly, node size of decision trees grows at the same time. However, when the same amount of train samples is used to extract rules, the results have relatively stable structures. The main reason that causes the instability comes from CART algorithm. Therefore, stable tree-based rule extraction algorithms are essential when we use decision trees to interpret CNNs.

## 8 Conclusion

We propose CNN2DT to train surrogate decision trees from CNNs and build a visual analysis system for exploration. We conduct three use cases and one user study to evaluate our method using the classic CNN models including VGG16 and AlexNet. The results show that CNN2DT provides global and local interpretations of the decision process of CNNs. Users can reason about the misclassification for single images and find the discriminating capacity of different models.

One drawback of our method is that the structures of decision trees are slightly unstable when using different amounts of train samples, which is mainly due to the rule extraction algorithm. Future work will first focus on the evaluation when using different tree-based rule extraction algorithms. Besides, more case studies would be conducted to further evaluate the usability of CNN2DT. Toward full validation, we will deeply collaborate with domain experts to meet their requirements as a further investigation.

**Acknowledgements** We would like to thank David Bau, Bolei Zhou, Aditya Khosla, Aude Oliva, Antonio Torralba, the authors of network dissection. Partial work of this paper is based on the results of their hard work and open source. Besides, we greatly appreciate the feedback from anonymous reviews. This work was supported by National NSF of China (No. 61702359).

## References

Asahi T, Turo D, Shneiderman B (1995) Visual decision-making: using treemaps for the analytic hierarchy process. In: Bederson BB, B Shneiderman (eds) The craft of information visualization. Elsevier, pp 405–406. https://doi.org/10.1016/B978-1-55860-915-0.X5000-8

Bau D, Zhou B, Khosla A, Oliva A, Torralba A (2017) Network dissection: quantifying interpretability of deep visual representations. In: CVPR

Breiman L, Friedman J, Stone CJ, Olshen RA (1984) Classification and regression trees. CRC Press, Boca Raton

Choo J, Liu S (2018) Visual analytics for explainable deep learning. IEEE Comput Graph Appl 38:84–92

Collins C, Penn G, Carpendale S (2009) Bubble sets: revealing set relations with isocontours over existing visualizations. IEEE Trans Vis Comput Graph 15:1009,1016–1009,1016

Craven M, Shavlik JW (1994) Using sampling and queries to extract rules from trained neural networks. In: International joint conference on neural networks, pp 37–45

Craven M, Shavlik JW (1996) Extracting tree-structured representations of trained networks. In: Advances in neural information processing systems, pp 24–30
Dabkowski P, Yarin G (2017) Real time image saliency for black box classifiers. Adv Neural Inf Process Syst 2017:6967–6976
Ellson J, Gansner ER, Koutsofios E, North SC, Woodhull G (2004) Graphviz and dynagraph—static and dynamic graph drawing tools. In: Graph drawing software. Springer, Berlin, Heidelberg, pp 127–148
Haipeng Z (2017) Towards better understanding of deep learning with visualization. Foundations of Science
Han J, Cercone N (2001) Interactive construction of decision trees. In: Cheung D, Williams GJ, Li Q (eds) Advances in knowledge discovery and data mining, PAKDD 2001. Lecture notes in computer science, vol 2035. Springer, Berlin, Heidelberg, pp 575–580. https://doi.org/10.1007/3-540-45357-1_61
Hohman FM, Kahng M, Pienta R, Chau DH (2018) Visual analytics in deep learning: an interrogative survey for the next frontiers. IEEE Trans Vis Comput Graph
Huysmans J, Baesens B, Vanthienen J (2006) Using rule extraction to improve the comprehensibility of predictive models. Social science Electronic Publishing, Rochester
Kahng M, Andrews PY, Kalro A, Polo DC (2017) ACTIVIS: visual exploration of industry-scale deep neural network models. IEEE Trans Vis Comput Graph 24:1–1
Karpathy A, Toderici G, Shetty S, Leung T, Sukthankar R, Fei-Fei L (2014) Large-scale video classification with convolutional neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1725–1732
Krizhevsky A, Hinton G (2009) Learning multiple layers of features from tiny images
Krizhevsky A, Sutskever I, Hinton GE (2012) Imagenet classification with deep convolutional neural networks. In: Proceedings of the 25th international conference on neural information processing systems, vol 1. Curran Associates Inc, Lake Tahoe, Nevada, pp 1097–1105
Lakkaraju H, Bach SH, Leskovec J (2016) Interpretable decision sets: a joint framework for description and prediction. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining pp 1675–1684
Liu Y, Salvendy G (2007) Design and evaluation of visualization support to facilitate decision trees classification. Int J Hum Comput Stud 65:95–110
Liu M, Shi J, Cao K, Zhu J, Liu S (2017) Analyzing the training processes of deep generative models. IEEE Trans Vis Comput Graph 24:1–1
Liu S, Xiao J, Liu J, Wang X, Jing W, Zhu J (2017) Visual diagnosis of tree boosting methods. IEEE Trans Vis Comput Graph 24:1–1
Lvd Maaten, Hinton G (2008) Visualizing data using t-SNE. J Mach Learn Res 9:2579–2605
Mansmann F, Göbel T, Cheswick W (2012) Visual analysis of complex firewall configurations. In: Proceedings of the ninth international symposium on visualization for cyber security, pp 1–8
Muhlbacher T, Linhardt L, Moller T, Piringer H (2017) TreePOD: sensitivity-aware selection of pareto-optimal decision trees. IEEE Trans Vis Comput Graph 24:1–1
Pedregosa F et al (2011) Scikit-learn: machine learning in Python. J Mach Learn Res 12:2825–2830
Pezzotti N, Höllt T, Gemert JV, Lelieveldt BPF, Eisemann E, Vilanova A (2017) DeepEyes: progressive visual analytics for designing deep neural networks. IEEE Trans Vis Comput Graph 24:1–1
Rauber Paulo E, Fadel Samuel G, Falcao Alexandre X, Telea Alexandru C (2017) Visualizing the hidden activity of artificial neural networks. IEEE Trans Vis Comput Graph 23:101–110. https://doi.org/10.1109/TVCG.2016.2598838
Reingold EM, Tilford JS (1981) Tidier drawings of trees. IEEE Trans Softw Eng 2:223–228
Ribeiro MT, Singh S, Guestrin C (2016) Why should I trust you?: Explaining the predictions of any classifier. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining pp 1135–1144
Sacha D, Kraus M, Keim DA, Chen M (2019) VIS4ML: an ontology for visual analytics assisted machine learning. IEEE Trans Vis Comput Graph 25:385–395
Sato M, Tsukimoto H (2001) Rule extraction from neural networks via decision tree induction. In: International joint conference on neural networks. Proceedings (Cat. No. 01CH37222), vol 3. IEEE, pp 1870–1875
Shixia Liu, Xiting Wang, Liu Mengchen, Jun Zhu (2017) Towards better analysis of machine learning models: a visual analytics perspective. Vis Inform 1:48–56
Simonyan K, Zisserman A (2013) Very deep convolutional networks for large-scale image recognition. In: ICLR 2015
Tzeng FY, Ma KL (2005) Opening the black box—data driven visualization of neural networks. In: VIS 05. IEEE Visualization, vol 2005. IEEE, pp 383–390
Van Den Elzen S, van Wijk JJ (2011) Baobabview: interactive construction and analysis of decision trees. In: IEEE conference on visual analytics science and technology, pp 151–160
Wang J, Gou L, Zhang W, Yang H, Shen H-W (2019) DeepVID: deep visual interpretation and diagnosis for image classifiers via knowledge distillation. IEEE Trans Vis Comput Graph 25:2168–2180
Wongsuphasawat K et al (2017) Visualizing dataflow graphs of deep learning models in tensorflow. IEEE Trans Vis Comput Graph 24:1–1
Wu M, Hughes MC, Parbhoo S, Zazzi M, Roth V, Doshi-Velez F (2017) Beyond Sparsity: tree regularization of deep models for interpretability. IIPS TIML workshop
Yao Ming HQ, Bertini E (2019) RuleMatrix: visualizing and understanding classifiers with rules. IEEE Trans Vis Comput Graph 25:342–352
Zeiler MD, Fergus R (2014) Visualizing and understanding convolutional networks. In: European conference on computer vision, pp 818–833
Zhang Q-s, Zhu S-C (2018) Visual interpretability for deep learning: a survey. Front Inf Technol Electron Eng 19:27–39
Zhang Q, Yang Y, Wu YN, Zhu S-C (2018) Interpreting CNNs via Decision Trees CoRR. arXiv:abs/1802.00121