Towards Neural Network Interpretability Using Commonsense Knowledge Graphs

Youmna Ismaeil[1,2](✉), Daria Stepanova[1], Trung-Kien Tran[1], Piyapat Saranrittichai[1], Csaba Domokos[1], and Hendrik Blockeel[2]

[1] Bosch Center for Artificial Intelligence, Renningen, Germany
{youmna.ismaeil,daria.stepanova,trung-kien.tran,piyapat.saranrittichai, csaba.domokos}@de.bosch.com
[2] KU Leuven, Leuven, Belgium
{youmna.ismaeil,hendrik.blockeel}@kuleuven.be

Abstract. Convolutional neural networks (CNNs) classify images by learning intermediate representations of the input throughout many layers. In recent work, latent representations of CNNs have been aligned with semantic concepts. However, for generating such alignments, the majority of existing methods predominantly rely on large amounts of labeled data, which is hard to acquire in practice. In this work, we address this limitation by presenting a framework for mapping hidden units from CNNs to semantic attributes of classes extracted from external commonsense knowledge repositories. We empirically demonstrate the effectiveness of our framework on copy-paste adversarial image classification and generalized zero-shot learning tasks.

Keywords: Interpretability · Image classification · Knowledge graphs

1 Introduction

Convolutional neural networks (CNNs) are well-known for their capacity to learn different powerful representations of the input in their successive layers. It is also well-known that they process images in a way that is not always intuitive to humans [11]. The lack of interpretability of CNNs is clearly undesirable, especially in safety-critical applications such as medical diagnosis or autonomous driving. This has led to an increased interest in methods that make the behavior of a trained CNN more interpretable by trying to assign human-understandable concepts (*e.g.*, face) to the neurons in the intermediate layers, often without explicit supervision [2,6,16,18,23,29].

An important class of methods [2,16,25] proposes to align neurons with class attributes by using images in which segments are labeled. More specifically, one tries to find out which neurons are activated by particular image segments and, in that manner, associate these neurons with the label of the segment. For instance, if, over multiple images, a particular neuron tends to be active for the image segments labeled with *table*, one could argue that this neuron recognizes tables,

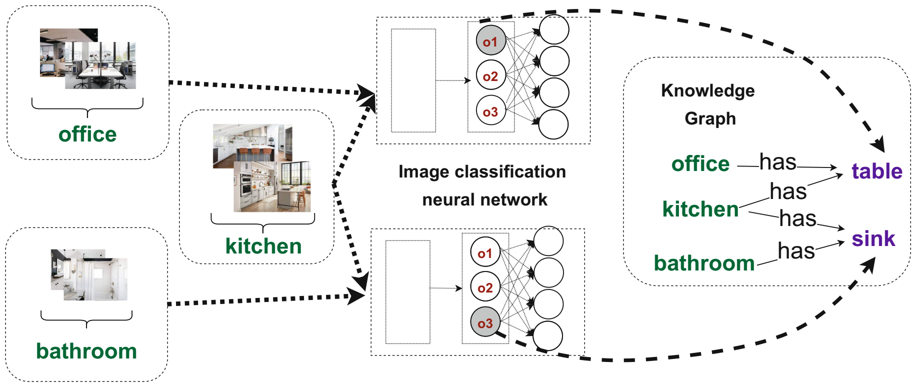
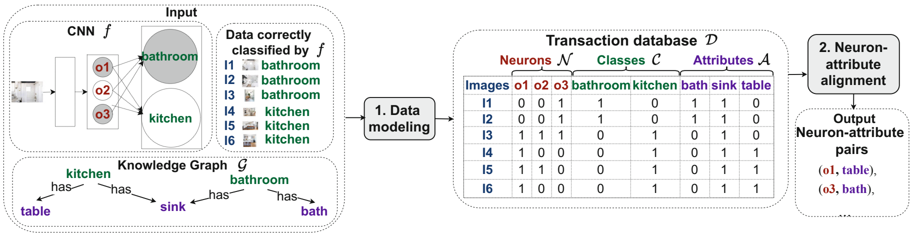


Fig. 1. The goal of our work is to assign meaning to neurons using semantic properties of classes from a knowledge graph.

and neurons in later layers essentially use this high-level information to decide whether the image shows, *e.g.*, an office. However, an important limitation of such methods is that during training they require fine-grained semantic labels of the images that are often not readily available and can be expensive to construct.

In this paper, we provide an alternative. Instead of using semantically labeled images, we assume that external knowledge extracted from a knowledge graph (KG) is available that contains symbolic descriptions of objects. Extensive KGs of this kind exist. For example, ConceptNet [26] or WebChild [27] store semantic (including visual) information about concepts (*e.g.*, *offices contain tables*, *kitchens contain ovens*, *etc.*) acquired using crowd sourcing or information extraction from the Web. We develop methods that exploit such KGs by linking the class label of images to typical visible attributes of this class, and then trying to correlate neuron activation with these attributes. For example, if a particular neuron tends to be active for pictures of offices and kitchens, but not for pictures of bathrooms, and the KG states that offices and kitchens tend to contain tables while bathrooms do not, then this may be an indication that the neuron reflects the presence of a table (see Fig. 1).

We demonstrate experimentally that the methods we propose successfully interpret neurons to the extent that they enable zero-shot learning of new classes with comparable performance to existing methods, but with the advantage of interpreting neurons in human-understandable terms. In the zero-shot learning setting, a model is expected to classify images from classes it has never encountered during training. Earlier works in this direction that likewise make use of external knowledge about classes [17,20] propose to exploit such knowledge during training. While natural, these methods typically assume that prior to training, the knowledge of both seen and unseen classes is available. This implies that whenever the source of knowledge (e.g., KG) is updated with information about new unseen classes, training needs to be done completely from scratch, which might be undesirable. Removing knowledge about unseen classes during



	Neurons $\mathcal{N}$			Classes $\mathcal{C}$		Attributes $\mathcal{A}$		
Images	o1	o2	o3	bathroom	kitchen	bath	sink	table
I1	0	0	1	1	0	1	1	0
I2	0	0	1	1	0	1	1	0
I3	1	1	1	0	1	0	1	0
I4	1	0	0	0	1	0	1	1
I5	1	1	0	0	1	0	1	1
I6	1	0	0	0	1	0	1	1

Fig. 2. Workflow of the proposed framework.

training makes the respective methods less effective (see Sect. 5). In contrast, our approach is advantageous in that we require the knowledge about unseen classes to be present in the KG only at the inference phase.

Our main contributions are summarized as follows:

- We propose a framework for mapping neurons in a fully connected layer of a CNN to attributes of classes from an external knowledge graph.
- On the task of copy-paste adversarial classification [16], we show that KGs indeed contain important semantic attributes of classes, which are helpful for CNNs.
- We experimentally demonstrate the usefulness of our framework for zero-shot learning and show how it can be effectively exploited for retrieving class predictions using reasoning over multiple networks.

2 Preliminaries

Image Classification CNN. Assuming a set of object classes $\mathcal{C}$, for a given (RGB) image $I\colon \Omega \rightarrow \mathbb{R}^3$, where Ω denotes the pixel space, we consider a function $f\colon (\Omega \rightarrow \mathbb{R}^3) \rightarrow \mathcal{C}$ for a parameter vector w. The function f, providing image classification, is defined as an L-layer *convolutional neural network* (CNN), namely $f(I) = \arg\max_{k\in\mathcal{C}} (\text{softmax}(f_L \circ \dots \circ f_2 \circ f_1(I))_k)$, where f_l defines the l^{th} layer of the network.

Knowledge Graphs. We will assume a *knowledge graph* (KG) encoding relations between object classes and attributes. Let $\mathcal{V}$ and $\mathcal{P}$ denote a set of entities (*a.k.a.* constants) and so-called predicates, respectively. A KG $\mathcal{G} \subseteq \mathcal{V} \times \mathcal{P} \times \mathcal{V}$ represents collections of factual information encoded by triplets $\langle \textit{subject}, \textit{predicate}, \textit{object} \rangle$. More formally, $\mathcal{G} = \{\langle s, p, o\rangle \mid s, o \in \mathcal{V} \text{ and } p \in \mathcal{P}\}$[1]. In this work, we focus on *commonsense KGs* (CSKG), that is, KGs that describe visual and physical properties of object classes (*e.g.*, $\langle \textit{bathroom}, \textit{has}, \textit{bath} \rangle$, $\langle \textit{kitchen}, \textit{has}, \textit{table} \rangle$). Examples of such knowledge graphs include, e.g., ConceptNet [26] or WebChild [27].

[1] Alternatively, a KG $\mathcal{G} = (\mathcal{V}, \{\mathcal{E}_p \subseteq \mathcal{V} \times \mathcal{V}\}_{p\in\mathcal{P}})$ can be viewed as a directed supergraph (*i.e.* a composition of directed graphs $\mathcal{G}_p = (\mathcal{V}, \mathcal{E}_p), \forall p \in \mathcal{P}$, where the edges are labeled by the predicates p.

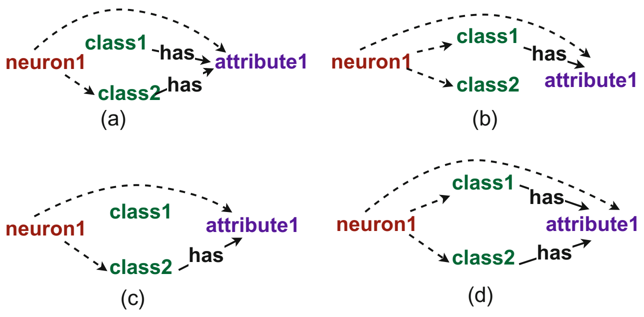


Fig. 3. Illustration of neuron-attribute alignments, where dashed lines show correlations between items in $\mathcal{D}$. (a), (b) present invalid alignments; (c), (d) reflect desirable alignments.

Association Rules. Association rules are widely used in the context of data mining. We will use association rules to model relations between neurons and attributes. First, we define a *transaction database* $\mathcal{D} = \{(i, \mathcal{X}) \mid i \in \mathcal{U} \text{ and } \mathcal{X} \subseteq \mathcal{I}\}$, where $\mathcal{I}$ stands for a set of items, and $\mathcal{U}$ is a set of IDs. A *transaction* $(i, \mathcal{X})$ has a (unique) ID i and $\mathcal{X} \subseteq \mathcal{I}$ denoting an itemset. Furthermore, we introduce the *support* of an itemset $\mathcal{X}$ in a given transaction database $\mathcal{D}$, which is the frequency of transactions in $\mathcal{D}$ containing the itemset $\mathcal{X}$:

$$\text{supp}(\mathcal{X}) = |\{(i, \mathcal{X}') \in \mathcal{D} \mid \mathcal{X} \subseteq \mathcal{X}')\}| \;/\; |\mathcal{D}| \;.$$

We consider bi-directional *association rules*, which are expressions of the form $\mathcal{X} \Leftrightarrow \mathcal{Y}$, where $\mathcal{X}, \mathcal{Y} \subseteq \mathcal{I}$. Association rules can be ranked relying on certain *interestingness metrics* [1]. In this work, we focus on Jaccard index (*a.k.a.* intersection over union), which for a given association rule $\mathcal{X} \Leftrightarrow \mathcal{Y}$ computes the ratio of co-occurrences to all occurrences of $\mathcal{X}$ and $\mathcal{Y}$ in the transaction dataset:[2]

$$J(\mathcal{X} \Leftrightarrow \mathcal{Y}) = \frac{\text{supp}(\mathcal{X} \cup \mathcal{Y})}{\text{supp}(\mathcal{X}) + \text{supp}(\mathcal{Y}) - \text{supp}(\mathcal{X} \cup \mathcal{Y})} \;. \tag{1}$$

3 Generating Neuron-Attribute Alignments

Our goal is to interpret neurons' behavior[3] in human-interpretable terms. In this paper, we consider image classification as an application. In contrast to existing methods [2,16], we aim at developing a framework that can be qualitatively and quantitatively evaluated and that is not limited by the availability of semantically labeled data.

[2] We also write $J(\mathcal{X}, \mathcal{Y})$ for conciseness.

[3] A neuron can also be understood as an element of the vector of activation output for a given layer.

We propose to interpret neurons' behavior in terms of the semantic properties (*a.k.a.* attributes, encoded by a set $\mathcal{A}$) from pre-constructed KGs, which store human knowledge about classes from $\mathcal{C}$. To this end, we aim at answering the following key questions:

Q1. Do the extracted alignments comply with the behavior of the CNN from which they were extracted?
Q2. Do networks learn in terms of attributes defined by the knowledge graph?
Q3. Beyond explainability, how can we utilize the extracted explanations?

Framework Overview. First, we present our framework, depicted in Fig. 2, for aligning individual neurons with high-level attributes from a KG.

Let us consider a neural network f trained for image classification on a labeled image dataset $\mathcal{T} = \{(I, c) \mid I \colon \Omega \to \mathbb{R}^3 \text{ and } c \in \mathcal{C}\}$. Moreover, assume we are given a knowledge graph $\mathcal{G} = \{\langle c, p, a\rangle \mid c \in \mathcal{C} \text{ and } a \in \mathcal{A}\}$ storing semantic properties (*a.k.a.* attributes) $\mathcal{A}$ of classes from $\mathcal{C}$, where the predicate p reflects a visual property, e.g., *hasColor*, *hasShape*, *hasPart*. While any kind of relation can be used in this context, we select those that likely provide visual attributes because naturally they are more effective for vision tasks than other relations such as *capableOf*, *isA*, etc.

We aim at aligning the neurons with the nodes $a \in \mathcal{A}$ from the KG $\mathcal{G}$. While, in principle, any set of neurons can be interpreted by our framework, we propose to specifically focus on the neurons from fully-connected layers, since these are known to reflect high-level abstract visual features [22]. Therefore, we consider a layer $f_l \colon \mathbb{R}^m \to \mathbb{R}^n$, $x \mapsto \sigma(Wx + b)$, where m and n stand for the fan-in and fan-out, respectively, for the given layer indexed by l and $\sigma \colon \mathbb{R}^n \to \mathbb{R}^n$ is an activation function. For $f_l(x) = \begin{bmatrix} o_1 \; o_2 \; \dots \; o_n \end{bmatrix}^\top$, we will use the notation $o_1, o_2, \dots, o_n$ for the individual neurons.

In the first step of our framework, we model the input data as a transaction database (Sect. 3.1). We then compute the neuron-attribute alignments using data mining-based methods (Sect. 3.2). The output of our framework is a set of neuron-attribute pairs ρ of the form (o, a), where o is an individual neuron of f, and $a \in \mathcal{A}$ is an entity in $\mathcal{G}$ corresponding to an attribute of a class in $\mathcal{C}$. In Sect. 4 and Sect. 5, we empirically provide answers to **Q1–Q3**.

3.1 Data Modeling

Suppose we are given a dataset $\mathcal{S} = \{(I, c) \mid f(I) = c\} \subseteq \mathcal{T}$ of images that are correctly classified by f. Given the neural network f, and the set $\mathcal{N} = \{o_1, \dots, o_n\}$ of individual neurons from the target layer f_l we proceed with constructing the transaction dataset $\mathcal{D}$ with items $\mathcal{I} = \mathcal{C} \cup \mathcal{N} \cup \mathcal{A}$. For every $(I, c) \in \mathcal{S}$, $\mathcal{D}$ stores a transaction $(i, \mathcal{X}_i)$, where i is the unique ID of the image I, and $\mathcal{X}_i \subseteq \{c\} \cup \mathcal{N} \cup \mathcal{A}$. For $a \in \mathcal{A}$, we have that $a \in \mathcal{X}_i$ iff $\langle c, p, a\rangle \in \mathcal{G}$, that is, the class of the image c and all of its attributes from the KG are in $\mathcal{X}_i$.

Intuitively, for every neuron $o \in \mathcal{N}$, it holds that $o \in \mathcal{X}_i$ iff o has high value before softmax when I is passed through f. To detect neurons from $\mathcal{N}$ with high value, the continuous values of the neurons are simply thresholded to a binary value $v_o \in \{0, 1\}$. This can be done *a priori* (*e.g.*, for post-ReLU activations) or by dynamically thresholding above neuron-specific percentile [16,24]. This way, for each image $I_i \in \mathcal{S}$ we identify a set of neurons from $\mathcal{N}$ with high activation value for the given image, and collect them into $\mathcal{X}_i$.

3.2 Neuron-Attribute Alignment

We then rely on the constructed transaction data $\mathcal{D}$ to compute the alignments between neurons and attributes, *i.e.*, items in $\mathcal{N}$ and those in $\mathcal{A}$, respectively. We propose methods for computing such alignments that we describe next.

Direct Method. Intuitively, a neuron o is *correlated* with an attribute a if the following two conditions hold: 1) it is highly probable that the attribute a is visually present in an image given that the neuron o is active for it; 2) it is highly probable that the neuron o is active, given that the attribute a is visually present in the image. Note that we cannot straightly compute the respective probabilities since the images are not explicitly labeled with the attributes. Therefore, instead, we estimate such probabilities by relying on the assumption that an attribute a is likely visible in an image I belonging to the class c if $\langle c, p, a\rangle \in \mathcal{G}$.

The first method that we propose (referred to as *direct*) is to *directly* construct the target alignments by identifying correlated pairs (o, a), where $o \in \mathcal{N}$, and $a \in \mathcal{A}$, such that $J(o, a) \geq \theta$ for a predefined threshold θ. The computed pairs are collected into the set ρ.

Example 1. Given $\mathcal{D}$ from Fig. 2 and $\theta = 0.7$, we have $J(o_1, \mathit{sink}) = 4/6$, $J(o_1, \mathit{table}) = 3/4$ and $J(o_3, \mathit{bath}) = 2/3$, $J(o_3, \mathit{sink}) = 3/6$. Thus, we obtain only (o_1, table) as the resulting alignment.

Constrained Method. While natural, the main drawback of the above *direct* method is that it only considers correlated pairs of neurons and attributes but ignores the knowledge about classes, like both bathrooms and kitchens have sinks, while offices and bedrooms do not. This information is important, especially when the dataset is unbalanced, to ensure that all meaningful alignments are computed.

Example 2. Reconsider Example 1. Looking closer at $\mathcal{D}$, one can observe that o_1 is highly correlated with the class *kitchen*, as it is active for all images of this class. Similarly, o_3 is highly correlated with the class *bathroom*. Since *bath* is the attribute which is relevant only for the class *bathroom* but not for *kitchen*, it would be expected that (o_3, bath) is also included in the resulting set of alignments along with (o_1, table). Decreasing the threshold θ to a lower value (e.g., 0.65) would resolve this, but would also lead to (o_1, sink) being in the result, which is counter-intuitive, since *sink* is an attribute which is relevant both for *bathroom* and *kitchen*, but *o1* is active only for images of the latter class.

Intuitively, an alignment is deemed meaningless if it fits into the cases (a) or (b) depicted in Fig. 3 (more formally defined via constrains below). An alignment in Fig. 3 (a) is undesirable, because we consider the neuron's activity as a sign of the existence or absence of the attribute in the input image. Consequently, if a neuron is active frequently for images of only one particular class, it might be a sign that this neuron is triggered by some attribute that only belongs to this class but not shared with other classes. Similarly, following Fig. 3 (b), if a neuron is active frequently for images of a set of classes, then it might be an indication that it is triggered by attributes that are shared among these classes.

For example, if o is aligned with the sink, then we expect o with high probability to be active for images of classes that typically contain sink (*e.g.*, bathrooms and kitchens), but not those that do not have sink (*e.g.*, bedrooms and offices). To alleviate the threshold's rigidity in the *direct* method, we establish formal constraints that allow us to filter out those and only those alignments that become completely meaningless when the class information is taken into account. The respective constraints are presented below:

(1) if $|\{c_i \in \mathcal{C} \mid \langle c_i, p, a\rangle \in \mathcal{G}\}| \geq 2$, and $|\{c_j \in \mathcal{C} \mid \langle c_j, p, a\rangle \in \mathcal{G}$ and $J(o, c_j) \geq \beta\}| < 2$, then (o, a) is an *invalid* alignment (see Fig. 3 (a)).
(2) if $\langle c_i, p, a\rangle \in \mathcal{G}$, for all $c_j \in \mathcal{C} \setminus \{c_i\}$, $\langle c_j, p, a\rangle \notin \mathcal{G}$ and $|\{c_j \in \mathcal{C} \setminus \{c_i\} \mid J(o, c_j) \geq \beta\}| \geq k-1$, then (o, a) is an *invalid* alignment, where $2 \leq k \leq |\mathcal{C}|$ is a parameter (see Fig. 3 (b) for $k = 2$).

Intuitively, the first constraint (1) states that if at least two classes have an attribute a, but only less than two out of them are correlated with the neuron o, then the alignment (o, a) is invalid. The second constraint (2) reflects that if a is relevant for a single class only, and the number of other classes correlated with o is larger than k, then (o, a) is invalid.

In the *constrained-k* method, after computing the alignments relying on the Jaccard similarity for a given threshold θ, we post-process the results by removing alignments that violate the above constraints. The *constrained-k* method is illustrated by the following example.

Example 3. We have $\langle c_i, has, sink\rangle \in \mathcal{G}$ for $c_i \in \{kitchen, bathroom\}$ in Example 1, *i.e.*, *sink* is an attribute that is relevant for at least two classes. Moreover, for $\beta = \theta$, we have $J(o_1, kitchen) > \beta$, but $J(o_1, bathroom) < \beta$, namely, the neuron o_1 is only frequently active for images of kitchen, but rarely for those of bathroom. Hence, based on the constraint (1), we remove $(o_1, sink)$ from the list of alignments computed by the *direct* method. Analogously, $(o_3, sink)$ is removed.

4 Evaluation and Applications

We now discuss various strategies for evaluating the computed neuron-attribute alignments as well possible applications, where they can be useful.

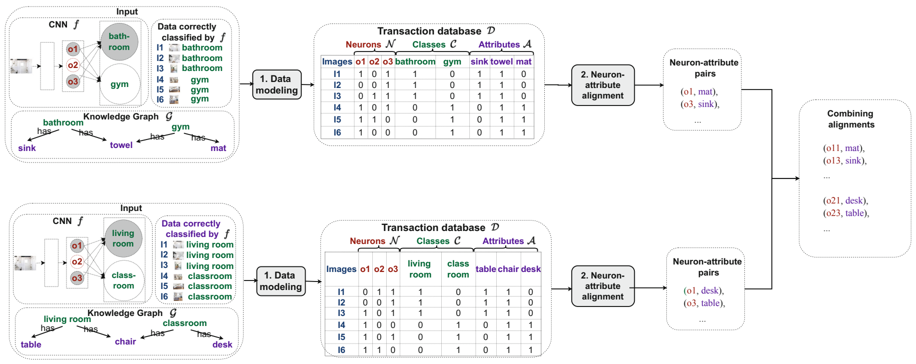
Copy-Paste Adversarial Examples. Adversarial images are images that are intentionally perturbed to confuse and deceive visual models. Changes to the image can be made in different ways, one of which is by copying patches from images of one class and pasting them into images of another class. Images with this type of perturbation are known as copy-paste adversarial images (as defined in [3]). Our hypothesis is that, if the network learns in terms of the attributes used in our work for interpreting neurons, then copying an image patch of an attribute relevant only for single class and pasting it into an image from another class should result in confusing the CNN to predict the class of the given image as the one to which the patch belongs. An example of a copy-paste adversarial image can be obtained by copying the image patch representing the bed from a bedroom image and pasting it into a bathroom image. Intuitively, if passing the bathroom image with a bed through a CNN trained on images of bathrooms and bedrooms results in confusing the network to classify the input image as a bedroom, then one can confirm that the network learns in terms of the attributes associated with the classes. We exploit the copy-paste adversarial images to in Sect. 5 to address **(Q1)** and **Q2**.

Towards addressing **(Q3)**, next we propose applications, in which the computed neuron-attribute alignments could be useful.

Zero-Shot Learning. The first application concerns zero-shot learning, *i.e.*, a popular task in image classification, in which images of new (*i.e.*, unseen) classes that do not exist in the training set need to be classified. For that, we develop an image classifier from the obtained neuron-attribute alignments. More specifically, given an image I of an unseen class, the trained network f, the pre-computed neuron-attribute pairs $\rho = \{(o, a) \mid o \in \mathcal{N}, a \in \mathcal{A}\}$, as well as the KG $\mathcal{G}$ storing the semantic information about the (un)seen classes, our goal is to derive the most likely class to which the target image I belongs. First, we pass I through f, and collect the set of activated neurons among $o \in \mathcal{N}$. We then exploit the neuron-attribute pairs ρ to retrieve the list of attributes, with which the active neurons are aligned, *i.e.*, $\mathcal{A}_I = \{a \in \mathcal{A} \mid (o, a) \in \rho \text{ and } o \text{ is active by } I \text{ in } f\}$. Finally, relying on $\mathcal{G}$, the classes $c \in \mathcal{C}$ are ranked based on the following scoring function:

$$\text{Score}(c) = \sum_{a \in \mathcal{A}_I} w_a \ / \ |\{a : \langle c, p, a \rangle \in \mathcal{G}\}| \tag{2}$$

where w_a is the ratio of neurons activated by the given image that are aligned with the attribute a to the total number of neurons activated by the image. The more active neurons are aligned with an attribute a, the more certain we are that a exists in the image and subsequently the greater is w_a. The candidate classes are ranked based on the formula from Eq. 2 and the top-ranked class is selected as the final prediction. Intuitively, the proposed technique referred to as *attribute-based classifier* allows one to reduce the image classification task to reasoning over the KG. The *attribute-based classifier* acts as an evaluator of the extracted alignments; if it gives high classification accuracy, then the extracted alignments reflect what the network learns.



Transaction database $\mathcal{D}$

	Neurons $\mathcal{N}$			Classes $\mathcal{C}$		Attributes $\mathcal{A}$		
Images	o1	o2	o3	bathroom	gym	sink	towel	mat
I1	1	0	1	1	0	1	1	0
I2	0	0	1	1	0	1	1	0
I3	0	1	1	1	0	1	1	0
I4	1	0	1	0	1	0	1	1
I5	1	1	0	0	1	0	1	1
I6	1	0	0	0	1	0	1	1

Transaction database $\mathcal{D}$

	Neurons $\mathcal{N}$			Classes $\mathcal{C}$		Attributes $\mathcal{A}$		
Images	o1	o2	o3	living room	class room	table	chair	desk
I1	0	1	1	1	0	1	1	0
I2	0	0	1	1	0	1	1	0
I3	1	0	1	1	0	1	1	0
I4	1	0	0	0	1	0	1	1
I5	1	0	1	0	1	0	1	1
I6	1	1	0	0	1	0	1	1

Fig. 4. The figure illustrates how to combine knowledge acquired by two networks. The aggregate alignments are later employed to reason about images from unseen classes.

Reasoning over Multiple Networks. The described *attribute-based classifier* can also be exploited in the setting, where multiple neural networks trained on non-intersecting sets of classes are used to solve tasks that are outside their initial scope (i.e., classify images into classes unseen by either of the networks). The systematic way of combining knowledge extracted from multiple networks is beneficial, as it allows one to avoid massive retraining on a larger number of classes while preserving high accuracy of predictions.

The procedure for attribute-based zero-shot classification can be naturally extended to handle several networks. First, we collect activated neurons for a given image from a number of networks, and then use the neuron-attribute alignments pre-computed for each network separately to detect attributes in the image. Finally, we merge the acquired knowledge using KG to make a decision regarding the most likely class of the image exactly in the same way as described above.

Example 4. Consider two convolutional neural networks f_1 and f_2 trained on the classes $C_1 = \{livingRoom, classroom\}$ and $C_2 = \{gym, bathroom\}$, respectively. Assume that the classes from C_1 have the following attributes $A_1 = \{table, chair, desk\}$, while those from C_2 contain the set of attributes $A_2 = \{mat, sink, towel\}$. Combining the knowledge acquired by the two networks would allow us to classify images into a new class, e.g., *kitchen*, which contains the set of attributes $A_3 = \{table, sink\}$ that were seen separately by the respective networks (see Fig. 4).

5 Experiments

We evaluate the proposed method for aligning neurons of a network with attributes of classes from a knowledge graph by empirically analyzing **(Q1)-(Q3)** from Sect. 3.

5.1 Experimental Setup

Datasets. We consider the popular *MITScenes* [19] and *AwA2* [28] datasets, and use *ConceptNet* [26] knowledge graph as a knowledge source.

MITScenes: We have selected images from the MIT scene dataset [19] belonging to classes whose visual properties are well covered in ConceptNet, which resulted in 10,475 images labelled with 15 classes. We use 3,840 images from 10 classes for *training*. For testing the performance on seen classes, we have 2,320 images belonging to the same 10 classes. For testing on the unseen classes, we have 4,675 images from the remaining 5 classes. On average, for each class, we get 645 images.

AwA2: We also consider a subset of the *AwA2* [28] dataset, in which the semantic information about image classes is well covered in the KG. We get 17,746 images spread across 20 classes. For training, we have 7,842 images from 15 classes. For testing on seen classes, we have 5,229 images belonging to the same 15 classes. For testing on unseen classes, we have 4,675 images from the remaining 5 classes.

ConceptNet KG: The *AwA2* and *MITScenes* datasets come with attributes already; however, their coverage is rather low (7.2 attributes per class at most). Our goal is to demonstrate the usefulness of commonsense KGs as sources for acquiring further class knowledge. For that, we have extracted attributes from a popular commonsense KG, ConceptNet [26]. For *MITScenes*, we collect the attributes connected to the classes via the inverse of *atLocation* relation (e.g., $\langle table, atLocation, kitchen \rangle$). In total, we get 1,680 attributes which on average amounts to 112 attributes per class. For *AwA2*, we use the predicate *has* to get 3,352 attributes, which yields 167 attributes per class on average.

CNN Training. We adopt ResNet50 [12] pre-trained on ImageNet [7] as the backbone, which in some experiments is fine-tuned on the considered datasets, while in others trained from scratch as described separately in each subsection. We replaced the fully connected layer before the last one in ResNet50 with a fully connected layer that has 2,048 neurons, which we aim at aligning with the attributes from ConceptNet.

Baselines. We compare our direct (*dir*) and constrained (*con*) methods (with fine-tuned parameters θ and β) for the attribute-based classification against the state-of-the-art methods that likewise make use of KGs (but do not map neurons to the KG entities), namely Dense Graph Propagation method (*DGP*) [14], Attentive Zero-Shot Learning method (*AZSL-D*) [10], and *ZSL-KG* [17]. *AZYL-D* and *ZSL-KG* rely on *DGP*, which is a framework that proposes a dense connection scheme of a knowledge graph to optimize the knowledge propagation between distant nodes in shallow networks such as graph convolution networks.

We run the methods proposed in [10] and [17], ensuring that no semantic information about unseen classes in the KG is used during training.

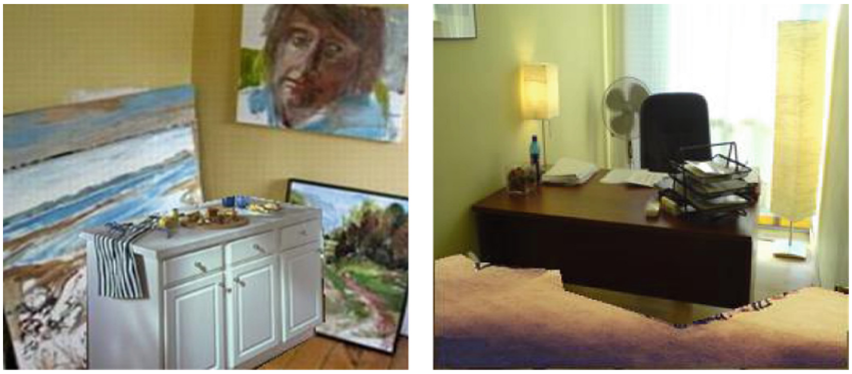


Fig. 5. Copy-paste adversarial examples: *art studio* with an *island* attribute relevant only for *kitchen* (left) and *office* with a *rug* relevant to classes *bedroom* and *office*.

Table 1. Class samples of adversarial copy-paste examples.

Original class	Inserted attribute	Resulting class
Dining room	Wall with doors	Corridor
Classroom	Shower	Bathroom
Office	Bed	Bedroom
Kitchen	Furniture	Office
Living room	Painting	Art Studio

Evaluation Metrics. We use the standard *hit@1* metric, which reflects the percentage of test images for which the method returned the correct class prediction in the top-1.

5.2 Copy-Paste Adversarial Examples

To answer **(Q1)-(Q2)**, following [16] we first generate the copy-paste adversarial examples using the *MITScenes* dataset, which comes with labeled semantic segments, as follows. Out of all labels, we select those (set $\mathcal{A}$) which are present in ConceptNet. Then, for each class $c \in \mathcal{C}$, we construct the set $\mathcal{A}_c = \{a \mid \langle c, p, a\rangle \in \mathcal{A} \text{ and } \forall c' \in \mathcal{C}, \langle c', p, a\rangle \notin \mathcal{G}\}$. For every pair of images I, I' from the test set belonging to different classes c and c' respectively, we insert the visualization of a randomly selected attribute $a \in \mathcal{A}_{c'}$ from the image I' into the image I, and label it with c'. For example, given an image I of an art studio and I' of a kitchen, as a copy-paste adversarial example, we generate an image I with the kitchen island from I' inserted into I (see Fig. 5 for illustration). The resulting image is labeled as a kitchen. The attributes to be inserted for every class are chosen randomly while making sure that they exist visually in at least one image (see Table 1 for class and attribute examples).

Table 2. Results for copy-paste adversarial classification on the *MITScenes* dataset. The alignment score is the percentage of images on which the attribute-based classifier (i.e., *con-all*, *con-2*) made the same prediction as *ResNet50*.

Method	hit@1(%)	Alignment(%)
ResNet50	78.2	–
con-all	**82.3**	96.14
con-2	82.1	96.64

This way, we obtain 2,320 adversarial copy-paste examples. We then analyze whether the considered CNN ResNet50 and our attribute-based classifier described in Sect. 4 misclassify the adversarial images relying on the inserted attribute. For instance, in the above example, we expect the network to misclassify the art studio as a kitchen. To perform such an evaluation, we pass every copy-paste adversarial example through the network and compute the *hit@1* score.

The results are presented in Table 2. High misclassification and alignment scores demonstrate that the KG attributes that distinguish classes from each other are indeed important for classification [5].

We have also repeated the same experiment on examples, constructed by inserting attributes relevant for multiple classes into the image (e.g., inserting a kitchen door into the art studio). We observe that in this case, the misclassification score in *hit@1* drops to around 2% for *ResNet50.* This witnesses that not all parts of an image are equally important for the network to make decisions, but only those that are distinguishing a given class from others based on the KG.

5.3 Zero-Shot Learning Task

To answer **(Q1)** and **(Q3)**, we compare the introduced methods for attribute-based classification to the baselines *AZSL-D* [10], *ZSL-KG* [20], and *DGP* [14] with respect to their performance on the zero-shot learning task.

For this task, we trained *ResNet50* on 10 classes of the *MITScene* dataset. We then used the trained network to compute the neuron-attribute alignment pairs for the classes from the training set. The other 5 classes are used for testing.

Since the knowledge graph stores the semantic information about both seen and unseen classes, we effectively exploit this knowledge along with the neuron-attribute alignments computed by our methods to classify the images from the unseen classes as described in Sect. 4. Importantly, the attributes of unseen classes are only used at the inference phase, but not during training.

Table 3 presents the results for the zero-shot learning tasks for *MITScenes* and *AwA2* datasets, respectively. Importantly, we report the performance both when using the attributes based on the semantic labels that accompany the datasets, as well as those from the ConceptNet KG.[4] For the *MITScenes* dataset,

[4] We also experimented with the WebChild [27] KG, but the results for ConceptNet are more promising.

Table 3. Zero-shot learning results on *MITScenes* and *AwA2*.

	Attribute source	Method	Unseen hit@1	Seen hit@1
MITScenes		AZSL-D	8.26	8.33
	–	DGP	19.87	9.73
	–	ZSL-KG	9.31	11.14
	Labels	*direct*	20.0	9.4
		con-2	20.0	19.1
		con-all	23.8	30.4
	Concept Net	*direct*	31.7	10.2
		con-2	38.4	50.0
		con-all	37.6	50.8
AwA2		AZSL-D	23.9	6.5
	–	DGP	24.1	6.6
	–	ZSL-KG	9.38	14.18
	Labels	*direct*	29.2	12.0
		con-2	38.6	41.5
		con-all	24.6	23.4
	Concept Net	*direct*	32.7	9.5
		con-2	40.2	47.6
		con-all	23.1	19.1

the attribute-based classifier that exploits attributes from *ConceptNet* outperforms all baselines including the attribute-based classifier that makes use of the labels coming with the dataset. This is due to the fact that, among visual attributes, the KG also provides a set of non-visual attributes that help in linking semantically similar classes via alignments. Moreover, the attribute labels coming from the dataset are shared among different classes, which leaves only a few attributes discriminatively describing each class.

Example Alignments. We present the alignments computed by our method for the considered datasets in Fig. 7. One can observe that the alignments indeed contain attributes visually relevant to the respective images.

5.4 Reasoning over Multiple Networks

We analyze the usefulness of the neuron-attribute alignments for the task of joint reasoning over multiple networks without having to retrain or fine-tune them.

In this experiment we trained from scratch two ResNet50 networks net_1 and net_2, on *AwA2* as follows. net_1 was trained for 15 epochs on 8 classes, and net_2 for 15 epochs on the other 7 classes. Moreover, we trained another network net on all 15 classes for 16 epochs. We get for net_1 a test accuracy on seen classes of 74.2%, for net_2 of 86.4%, and for net 80.5%. For the test set, we have images from 5 unseen classes.

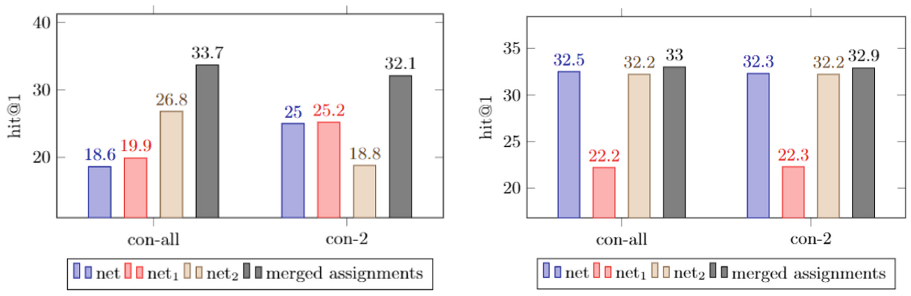


Fig. 6. Zero-shot learning using neuron-attribute alignments over multiple networks on *AwA2* (left) and *MITScenes* (right).

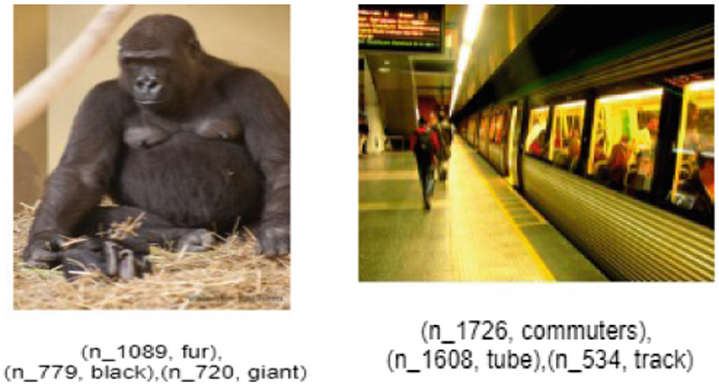


Fig. 7. Examples of neuron-attribute pairs computed by our method.

Figure 6 shows the results of the presented methods for reasoning over knowledge learned by multiple networks. For *AwA2* dataset, the attribute-based classifier that jointly considers neuron-attribute alignments from both net_1 and net_2 outperforms the attribute-based classifiers constructed separately for net_1, net_2 and net respectively. For the *MITScenes* dataset, a similar trend is observed with the exception that the attribute-based classifier constructed relying on net_2 significantly outperforms the one based on net_1 and has comparable performance to other classifiers. This is due to the fact that net_1 (resp. net_2) was trained on classes with many (resp. few) attributes in common with the unseen classes.

6 Related Work

Recently, there has been an increasing interest in understanding what deep learning models learn. Our work extends the earlier proposals on explaining individual neurons in deep representations [2,6,9,16,25]. However, in contrast to existing work, we do not rely on large amounts of training data, but instead exploit external knowledge graphs. In [25], semantic knowledge (in the form of ontologies) has also been considered for interpreting CNNs by generating explanations, yet this method again makes use of labeled and segmented images, unlike we do.

Our approach for aligning neurons and KG attributes is in the spirit of [8, 13], where data mining has been exploited for interpreting neural networks or forming concepts based solely on neurons, but KGs have not been considered in this context. Recently, many works have motivated the exploitation of KGs for enhancing the performance of image classifiers, e.g., [10,14,17,20] (see [4] for an overview). The direction of explaining the behavior of CNNs with semantic technologies has been also discussed as a valuable research stream in several works [15,21]. However, to the best of our knowledge, no concrete proposals for aligning individual neurons with KG entities exist to date.

Several works have considered graph neural networks (GNNs) for zero-shot learning [10,14,17,20]. These methods make use of graph-structured external knowledge, in which each class is represented by a single node and each inter-class link is represented by an edge. Given the external knowledge graph, its embedding representation is first computed using a GNN, and then exploited for the zero-shot learning task.

Some techniques [10,14] utilize convolutional layers with an additional dense connection layer to propagate features to distant nodes within the same network. The work [10] further introduced weighted aggregation as a method for emphasizing more significant neighboring nodes for class nodes. A key difference between these approaches and our work is that they do not aim at aligning neurons with KG entities like we do. Additionally, they explicitly include KG-based information about unseen classes during training, whereas we only exploit this knowledge at the inference phase. We observed that when knowledge about unseen classes is omitted from the information used for training, the performance of AZSL-D [10], DGP [17], and ZSL-KG [20] drops significantly (see Table 3).

7 Conclusion

This paper proposes a framework for aligning neurons of a neural network to attributes defined by external commonsense knowledge graphs. These alignments not only make NNs more interpretable (see Fig. 7 for examples), but are also useful in various applications, such as zero-shot classification (using multiple networks). Our framework does not require the knowledge about unseen classes to be used during training, but rather exploits it at inference stages. Our results demonstrate that commonsense KGs contain distinctive attributes relying on which CNNs tend to perform classification. This demonstrates the importance and usefulness of commonsense KGs for computer vision tasks.

Although we relied on ConceptNet KG in the experiments, our work is certainly not bound to it, and other KGs (or combinations of them) can likewise be exploited. We believe that our method has a broader impact, as it offers an interesting perspective for reducing machine learning tasks to those of reasoning over KGs.

Supplemental Material Statement: Supplemental material is available at https://github.com/boschresearch/feature-attribute-association.

Acknowledgements. We would like to thank Dr. Volker Fischer from Bosch Center for AI for providing helpful feedback on initial versions of this work.

References

1. Agrawal, R., Imielinski, T., Swami, A.N.: Mining association rules between sets of items in large databases. In: SIGMOD 1993, pp. 207–216 (1993)
2. Bau, D., Zhou, B., Khosla, A., Oliva, A., Torralba, A.: Network dissection: quantifying interpretability of deep visual representations. In: CVPR, pp. 3319–3327 (2017)
3. Brunner, T., Diehl, F., Knoll, A.: Copy and paste: a simple but effective initialization method for black-box adversarial attacks. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW) (2019)
4. Chen, J., Geng, Y., Chen, Z., Horrocks, I., Pan, J.Z., Chen, H.: Knowledge-aware zero-shot learning: survey and perspective. In: IJCAI, pp. 4366–4373. ijcai.org (2021)
5. Cheng, X., Lu, J., Feng, J., Yuan, B., Zhou, J.: Scene recognition with objectness. Pattern Recogn. **74**, 474–487 (2018)
6. Dalvi, F., Nortonsmith, A., Bau, A., et al.: Neurox: a toolkit for analyzing individual neurons in neural networks. In: AAAI 2019, pp. 9851–9852 (2019)
7. Deng, J., Dong, W., Socher, R., Li, L., et al.: Imagenet: a large-scale hierarchical image database. In: CVPR 2009, pp. 248–255 (2009)
8. Endres, D., Földiák, P.: Interpreting the neural code with formal concept analysis. In: NIPS, pp. 425–432 (2008)
9. Fong, R., Vedaldi, A.: Net2vec: quantifying and explaining how concepts are encoded by filters in deep neural networks. In: CVPR, pp. 8730–8738 (2018)
10. Geng, Y., Chen, J., Zhiquan Ye, e.: Explainable zero-shot learning via attentive graph convolutional network and KGs. SW 12 (2021)
11. Goodfellow, I.J., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples. In: ICLR 2015 (2015)
12. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
13. Horta, V.A.C., Mileo, A.: Towards explaining deep neural networks through graph analysis. In: DB and Expert Systems Applications, pp. 155–165 (2019)
14. Kampffmeyer, M., Chen, Y., Liang, X., Wang, H., Zhang, Y., Xing, E.P.: Rethinking knowledge graph propagation for zero-shot learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2019)
15. Lécué, F.: On the role of knowledge graphs in explainable AI. Semant. Web **11**(1), 41–51 (2020)
16. Mu, J., Andreas, J.: Compositional explanations of neurons. Adv. Neural Inf. Process. Syst. **33**, 17153–17163 (2020)
17. Nayak, N.V., Bach, S.H.: Zero-shot learning with common sense knowledge graphs. CoRR abs/2006.10713 (2020)

18. Nguyen, A.M., Dosovitskiy, A., Jason Yosinski, e.: Synthesizing the preferred inputs for neurons in neural networks via deep generator networks. In: Neurips 2016, pp. 3387–3395 (2016)
19. Quattoni, A., Torralba, A.: Recognizing indoor scenes. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition, pp. 413–420 (2009). https://doi.org/10.1109/CVPR.2009.5206537
20. Roy, A., Ghosal, D., Cambria, E., Majumder, N., Mihalcea, R., Poria, S.: Improving zero shot learning baselines with commonsense knowledge. CoRR abs/2012.06236 (2020)
21. Sarker, M.K., Xie, N., Doran, D., Raymer, M., Hitzler, P.: Explaining trained neural networks with semantic web technologies: first steps. In: NeSy (2017)
22. Selvaraju, R.R., et al.: Choose your neuron: incorporating domain knowledge through neuron-importance. In: ECCV (13), pp. 540–556 (2018)
23. Simonyan, K., Vedaldi, A., Zisserman, A.: Deep inside convolutional networks: visualising image classification models and saliency maps. In: ICLR 2014 (2014)
24. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: ICLR 2015 (2015)
25. de Sousa Ribeiro, M., Leite, J.: Aligning artificial neural networks and ontologies towards explainable AI. In: AAAI 2021, pp. 4932–4940 (2021)
26. Speer, R., Chin, J., Havasi, C.: Conceptnet 5.5: an open multilingual graph of general knowledge. In: AAAI 2017, pp. 4444–4451 (2017)
27. Tandon, N., de Melo, G., Suchanek, F.M., Weikum, G.: Webchild: harvesting and organizing commonsense knowledge from the web. In: WSDM. ACM (2014)
28. Xian, Y., Lampert, C.H., Schiele, B., Akata, Z.: Zero-shot learning-a comprehensive evaluation. IEEE Trans. Pattern. Anal. Mach. Intell. **41**(9), 2251–2265 (2019)
29. Zhou, B., Khosla, A., Lapedriza, À., Oliva, A., Torralba, A.: Object detectors emerge in deep scene cnns. In: ICLR 2015 (2015)