POEM: Pattern-Oriented Explanations of Convolutional Neural Networks

Vargha Dadvar
University of Waterloo
vdadvar@uwaterloo.ca

Lukasz Golab
University of Waterloo
lgolab@uwaterloo.ca

Divesh Srivastava
AT&T Chief Data Office
divesh@research.att.com

ABSTRACT

Convolutional Neural Networks (CNNs) are commonly used in computer vision. However, their predictions are difficult to explain, as is the case with many deep learning models. To address this problem, we present POEM, a modular framework that produces patterns of semantic concepts such as shapes and colours to explain image classifier CNNs. POEM identifies patterns such as "if sofa then living room", meaning that if an image contains a sofa and the model pays attention to the sofa, then the model classifies the image as a living room. We illustrate the advantages of POEM over existing work using quantitative and qualitative experiments.

PVLDB Reference Format:
Vargha Dadvar, Lukasz Golab, and Divesh Srivastava. POEM: Pattern-Oriented Explanations of Convolutional Neural Networks . PVLDB, 16(11): 3192 - 3200, 2023.
doi:10.14778/3611479.3611518

PVLDB Artifact Availability:
The source code, data, and/or other artifacts have been made available at https://github.com/vargha67/poem-pipeline.

1 INTRODUCTION

Convolutional neural networks (CNNs), commonly used in computer vision, are deep learning models designed to detect concepts such as shapes, objects, and textures in images. However, CNNs are difficult to explain, as is the case with many deep learning models. This makes them hard to debug and limits their adoption, especially in critical applications such as healthcare and law [20].

Explanations can be *local*, to understand individual predictions, or *global*, to interpret the general decision-making of a model. Local explanations of CNNs identify pixels in an image that influenced the model's decision [2, 19, 21, 22, 27, 29]. Global explanations identify the concepts learned by a CNN. Some global methods additionally summarize the relationships between concepts and predictions using rules or decision trees [8, 15, 24, 28]. We focus our attention on the problem of global CNN explanations.

Global CNN explanation approaches face three challenges: inspecting the model to determine which parts of images it pays attention to during inference, identifying concepts located in those parts, and summarizing how these concepts are related to the model's predictions. In this paper, we propose a three-step process to address these challenges. The first step, *Concept Identification*, finds the concepts learned by the model. The second step, *Concept Attribution*, associates images with the concepts that the model pays attention to when classifying the images. The third step, *Concept Pattern Mining*, summarizes how concepts are linked to prediction decisions. This three-step design allows us to analyze existing methods and identify their limitations: false positives during concept identification and attribution, and pattern verbosity. To overcome these limitations, *we propose POEM: a system that explains image classifier CNNs using concise and informative patterns of concepts.*

POEM takes in a pretrained CNN model and a *target* dataset containing images the CNN is trained to classify. The output is a set of patterns linking concepts occurring in images with the model's predictions. For example, suppose we apply POEM to explain a CNN that classifies images of rooms into living rooms and bedrooms. POEM may identify a pattern "if sofa then living room", indicating that if an image contains a sofa and the model pays attention to the sofa during inference, then the model classifies the image as a living room. As explained in Sections 2 and 3, the novelty of POEM, enabled by the three-step design, consists of using state-of-the-art methods in the first two steps to avoid false positives, and using concise and informative rule mining methods (which have not been used before for CNN explanation) in the third step.

An explanation is a set of such patterns, describing the inner workings of the model to build end user trust and assist experts with model debugging. Specifically, POEM can be used to verify that the CNN is paying attention to the right concepts instead of overfitting. Furthermore, model engineers can use POEM to identify model weaknesses through concepts whose presence is associated with incorrect predictions; one solution may be to collect more examples containing these concepts. Finally, the patterns produced by POEM can serve as data quality tools to identify mislabelled images and bias in training data.

In Section 2, we give an overview of CNNs and existing work on explaining them, along with their limitations. In Section 3, we present the modular design of POEM. Section 4 presents our experimental evaluation, including efficiency improvements due to our filtering of insignificant concepts, and we conclude in Section 5 with directions for future work.

2 BACKGROUND AND RELATED WORK

2.1 Convolutional Neural Networks

A CNN takes an image as input, encoded as matrices of pixel intensities of each of the three primary colours, and outputs a predicted label for the image. During inference, the input passes through the layers of the CNN. Some layers are *convolutional*, whose purpose is to transform the input pixels to detect objects and shapes. Convolutional layers are implemented using *filters*, whose outputs are called *activation maps*: matrices indicating the locations within a

Proceedings of the VLDB Endowment, Vol. 16, No. 11 ISSN 2150-8097.
doi:10.14778/3611479.3611518

given image that were activated by the filter. For example, filters in the first few layers may identify pixels that correspond to straight lines, whereas filters in the deeper layers are more likely to detect complex objects and shapes [3, 4, 18, 21].

During training, labelled images are passed through the model, and the convolutional filters are updated by *backpropagating* the classification errors. The *gradients* of the errors with respect to the filters determine how much and in what direction each filter should be updated. The filters thus learn to identify features in images that minimize the classification error of the model.

2.2 Explaining Convolutional Neural Networks

Some concept identification methods analyze a pretrained model without modifying it [3, 9, 11, 16], and others modify the CNN architecture to reveal the concepts learned [6, 28]. In the former category, which is the focus of POEM, CNN2DT [15] builds a decision tree to predict the model's predictions based on the concepts found in images. CNN2DT uses *Network Dissection* [3], which works as follows. First, a secondary image dataset is assembled in which images are labeled as containing a set of predefined concepts. Second, these images are passed through the CNN to determine which filters (in the last convolutional layer) detect these concepts, producing a filter-to-concept mapping. However, these predefined concepts may not appear in the target dataset or may not be relevant to the classification decisions over the target dataset. For example, detecting a car may not be relevant when distinguishing between living rooms and bedrooms. As we will show in Section 3, POEM addresses this by using only the concepts that exist in the target dataset, leading to 30-60% improvements in concept relevance.

ACDTE [8] proposed a concept identification method to explain a single prediction. First, images similar to the one being explained are identified and segmented. Then, the image segments are clustered according to the filters they activate, assuming that each cluster corresponds to a concept. However, this approach requires human inspection to map the clusters to the corresponding concepts. Moreover, a cluster may include image segments corresponding to multiple concepts. ERIC [24], which computes rules relating filters to model predictions, has the same shortcomings since it requires manual inspection of images to identify filter-concept mappings.

After identifying the concepts, the next step is to tag each image in the target dataset with the concepts that played a role in the model's prediction for that image. A naive solution only checks whether a concept exists in the image, while a more accurate solution also ensures that the model pays attention to the concept during inference. In most of the related work [8, 15, 24], filter-concept mappings from concept identification are used in concept attribution: the concept must exist in the image and must activate the corresponding filter during inference. However, while high activation can indicate that a filter pays attention to a concept, it does not guarantee that the filter has played a significant role in the prediction of the model. Furthermore, the locations activated by the filter may not correspond to the location of the concept in the image. As we explain in Section 4.3.2, POEM addresses these shortcomings and improves concept attribution by a factor of 6-10.

For mining patterns of concepts, related methods use decision trees [8, 15, 24, 28]. Since decision trees are known to be quite verbose, POEM instead uses recently proposed rule mining methods, including decision trees, explanation tables [10] and interpretable decision sets [17], for concise and informative explanations.

3 SYSTEM DESIGN

Figure 1 summarizes the architecture of POEM. The modular design allows us to address the shortcomings of prior work by implementing state-of-the-art solutions in each module.

3.1 Concept Identification

The goal of this step is to find the concepts learned by a CNN model, i.e., concepts detected by the filters in the last convolutional layer. Figure 1 shows three filters mapped to their corresponding concepts: 'cabinet', 'sofa' and 'table'.

POEM uses the latest version of *Network Dissection* [4]. Instead of using a pre-segmented secondary dataset, each image in the target dataset is segmented down to its concepts. This is done using the *UPerNet* semantic segmentation method, which is based on *Unified Perceptual Parsing* [26]. This method is pretrained on a dataset called *Broden* to identify a variety of concepts such as objects, object parts, materials and colours. The Broden dataset itself includes multiple segmented scene and object datasets. This method can potentially be pretrained on other segmented datasets such as medical images for domain-specific explanations.

After identifying the concepts, we pass each image through the CNN and measure the pixel overlap between concepts and filter activation maps. We identify locations in a filter activation map whose values are higher than 99% of this filter's activation map's values over all the images, which is the recommended threshold for network dissection. We call these locations *highly activated*. For this, activation maps need to be *upsampled* (i.e., resized) to the size of the input image.

Figure 2 shows an example of concept identification for a single image and a single filter. On the right, semantic segmentation identifies the 'bed' concept in the image. On the left, we show the filter activation map when this image passes through the CNN (yellow areas are highly activated). After upsampling the activation map, we compute the overlap between the highly activated areas in the filter's activation map and the location of the 'bed' concept.

After repeating this process for every image, we map each filter to the most likely concept it is detecting. This is the concept having the most overlap with the filter's highly activated areas over all the images. Overlap is computed using *Intersection over Union (IoU)*, which measures the ratio of the total intersection between concepts and highly activated areas of the activation map over their union. As in network dissection, we ignore weak filter-concept mappings with an IoU lower than 0.04. While each filter is only mapped to a single concept, multiple filters may map to the same concept.

As explained in Section 2, related work on filter-concept mappings (ERIC [24]) requires manual examination of activation regions in images, and ACDTE [8] finds clusters of image segments that may not correspond to concepts. Furthermore, CNN2DT implements network dissection by passing pre-segmented images from the Broden dataset through the CNN for concept identification, which may lead to concepts that do not exist in the target dataset. The critical difference in POEM is that network dissection is applied

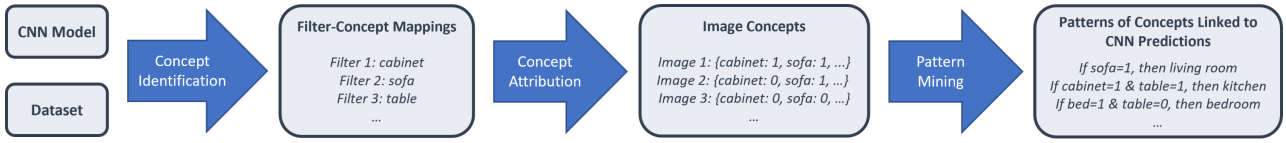

Figure 1: Overview of POEM

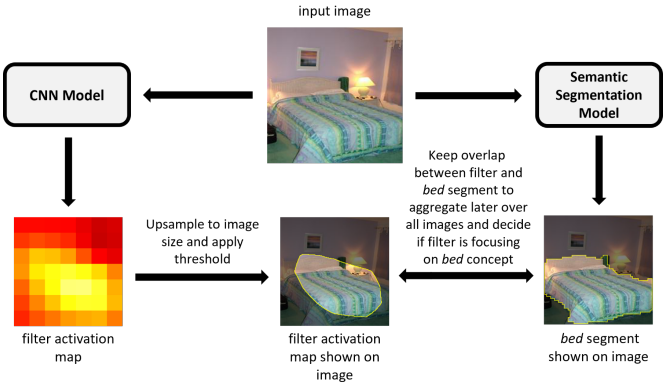

Figure 2: POEM's concept identification process

directly to segment the images in the target dataset, which are then passed through the CNN. As a result, unlike CNN2DT, the concepts used by POEM are a subset of the Broden concepts that actually exist in the target dataset. This allows us to explain a CNN using concepts relevant to the classification task at hand.

3.2 Concept Attribution

The goal of concept attribution is to correlate concepts with the model's decisions. Figure 1 shows examples of the output of this step, in which each image is tagged with the most likely concepts that played a role in the model's prediction.

POEM attributes a concept to an image if the following three conditions are true: 1) the concept exists in the image; 2) a filter mapped to the concept (during concept identification) has played a role in the model's prediction for the image, as indicated by filter activations and gradients; and 3) the area of the image containing the concept overlaps significantly with the corresponding filter's highly activated area.

To check condition 1, we use the semantic segmentation model explained earlier (UPerNet), which allows us to locate the concept mapped to each filter in the image. Specifically, we reuse the same segmentations produced during concept identification.

We check condition 2 using a process inspired by Grad-CAM [21], which is a method for explaining an individual prediction based on model gradients. A CNN classifier outputs a value for each class, and the class with the highest value is selected as the prediction. Similar to Grad-CAM, we compute the gradients of the predicted class values for the image with respect to each filter's activation values. We then multiply the gradients by the corresponding activation values element-wise, to capture the effects of both the gradients and activations. We then apply the *ReLU* function to focus on activation values that have a positive effect on the predicted class. We finally obtain an activation-gradient map for each filter, which we call the *filter saliency map*. The values in the filter saliency map represent importance for the model's prediction.

Our approach differs from Grad-CAM in that we do not aggregate the saliency maps of different filters to create a single saliency map for the image. The reason is that we want to evaluate the importance of each individual filter (concept). Moreover, unlike Grad-CAM, we do not average the gradients of a filter before multiplying by the activations. This is because we need to know the importance of each value in a filter's activation map when we later check the overlap with the concept's location in the image.

To identify the high-importance parts of the saliency map for each filter, we use as a threshold the 95th percentile of all the activation gradient values over all the filters for the given image. Activation gradient values higher than this threshold have the most impact on the prediction of the model. Our experiments show that most values below this threshold are very close to zero and have little impact on the prediction.

Given the high-importance mask for each filter, we check the overlap with the concept mapped to the filter, which corresponds to condition 3. To check this *concept-saliency overlap*, we need to upsample the filter's activation-gradient map to the size of the image. We then check if at least 50% of the high-importance area of the filter is covered by the concept found in the image (our experiments showed that thresholds higher than 50% can be too strict because of errors in upsampling the activation gradient maps). In this case, our conditions for attributing the concept to the image are satisfied, and we set the concept value to *1* for this image. Otherwise, we set the concept to *0*. Note that it may be possible to give different weights to the attributed concepts based on their level of overlap with high-importance filter areas or the number of related filters activated. However, for simplicity, we only consider binary concept attribution in the current version of POEM.

Figure 3 shows an example of concept attribution for an image containing a bed that is predicted as a bedroom. On the left, we show the filter saliency map that checks condition 2 (by multiplying the activation map by the gradients of the predicted class label and passing the output through the *ReLU* function). The filter saliency map is upsampled to check the overlap with the location of the 'bed' concept (found in the concept identification step).

Upon completion of concept attribution for all the images in the target dataset, we discard the weakest concepts, which we define as those associated with less than 1% of images. If no such concepts exist in the given dataset then we can reduce the threshold to, say, 0.5%. Furthermore, we only keep the top 10 concepts having the highest mutual information with the model's prediction. This filtering of weak concepts not only enables faster pattern mining

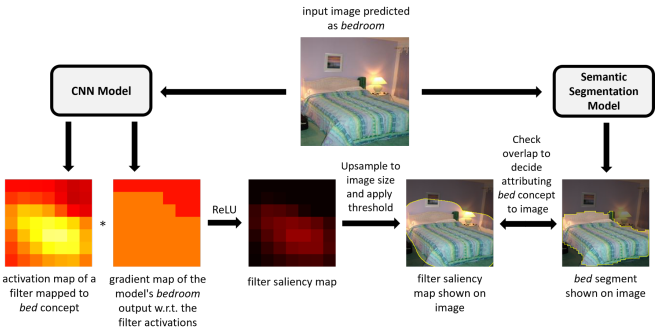

Figure 3: Concept attribution process used in POEM

in the next step but also leads to more concise patterns based on the most important concepts, as we will show experimentally in Section 4.3.

The output of the concept attribution step is a new dataset D, with one row for each image in the target dataset. Every important concept is a binary feature, corresponding to the attribution (or not) of the concept to an image. The labels of D are the predictions of the CNN model for the target images. The ith row of D can be thought of as a *local* explanation for the model's decision-making process on the ith image. In the next step of POEM's design, we go one step further to find patterns over the concept attributions encoded in D to explain the overall decision making of the model.

Finally, we point out that related methods such as CNN2DT and ERIC omit conditions 1 and 3 and only use activations to check condition 2, leading to false positives compared to POEM.

3.3 Concept Pattern Mining

In contrast to related work that uses only decision trees [8, 15, 24, 28], we use a set of rule mining methods to find patterns linking concepts to model predictions: *Classification and Regression Trees (CART)* [5], *Explanation Tables* [10], and *Interpretable Decision Sets (IDS)* [17]. Figure 1 shows three example patterns.

In CART, each node represents a concept and each root-to-leaf path represents a pattern of concepts. We control the size of the tree by setting a minimum number of samples per leaf. This is equivalent to a minimum support of each pattern, which is the number of examples matching the concepts referenced in the pattern.

Explanation tables find a set of patterns that provide the most information about the distribution of an outcome, which is the model's predicted class in our case. Information is defined as the ability to reconstruct the distribution of the outcome from the distributions within the reported patterns. In other words, patterns with unusual or unexpected distributions of the outcome are prioritized.

IDS, originally proposed as an interpretable classification model, finds rules by taking into account conciseness, confidence and support. We selected IDS for our set of rule mining methods because all three of these properties are important for interpretable and accurate explanations of CNNs based on concepts. The minimum support of the candidate patterns and the weights of the different optimization criteria are user-defined parameters.

3.4 Pattern Analysis and Visualization

While patterns reveal how concepts are related to model predictions, visual analysis of the images related to each pattern provides further insights into the weaknesses and strengths of the model and potential data quality issues. For this purpose, POEM displays three categories of images for each pattern: *matching*, *non-matching*, and *wrongly-predicted* (see [7] for details of the POEM front end).

Matching images agree with both the concepts and the prediction stated in the pattern. For example, for "if sofa then living room", matching images are attributed to the concept 'sofa' and predicted as living rooms by the CNN. Inspecting these images, with their high-importance areas highlighted, helps to verify the correct attribution of concepts to images.

Non-matching images agree with the concepts of a pattern, but not with the prediction stated in the pattern. For "if sofa then living room", non-matching images are attributed to the concept 'sofa' but *not* predicted by the model as living rooms. If we define the *confidence* of a pattern as the fraction of images that match its concepts but not the prediction, then non-matching images exist for patterns with confidence below 100%. For instance, we may want to know whether there are rare images of bedrooms including sofas in the target dataset, or if there are images of living rooms that the model predicts incorrectly despite the existence of sofas.

Wrongly predicted images agree with the concepts and the model's prediction in the pattern, but have a label in the target dataset. For "if sofa then living room", wrongly predicted images are attributed to 'sofa', predicted by the model as living rooms, but *not* labelled in the target dataset as living rooms. If we define the *accuracy* of a pattern as the fraction of images matching the concepts and prediction of the pattern that also have the same label as the prediction, then wrongly predicted images exist when the accuracy of a pattern is less than 100%. Inspecting these examples can identify the characteristics of images that the model predicts incorrectly, as defined by their concepts. For instance, some of these images may include concepts found in both bedrooms and living rooms. Including more such examples in the training dataset may improve the model's accuracy. Some wrongly predicted images may be mislabeled as bedrooms – a data quality issue.

4 EXPERIMENTS

4.1 Implementation

The POEM back end is implemented using Python and PyTorch. Our implementation of network dissection is based on its Github code[1]. Similarly, for unified perceptual parsing, we used the code from the project Github page[2]. For rule mining, we used the Scikit-Learn implementation of CART, we obtained a copy of the explanation tables code from the authors, and we used the publicly available implementation of IDS from its Github page[3].

For comparison to POEM, we implemented an approach similar to CNN2DT [15] to represent previous work. Because we did not have access to the code for CNN2DT or any other closely related work, we implemented CNN2DT in our framework as follows. For concept identification, we used the original network dissection

[1] https://github.com/davidbau/dissect
[2] https://github.com/CSAILVision/unifiedparsing
[3] https://github.com/lvhimabindu/interpretable_decision_sets

method [3], with code from the corresponding Github repository[4]. For concept attribution, we only check if a filter mapped to a concept includes a high-activation area when an image is passed through the model. As in network dissection, we use the 99th percentile of the filter's activation values over all images as the threshold for high activation. For pattern mining, we use the CART decision tree.

4.2 Dataset and Evaluation Methodology

We use the Places dataset [30] containing labelled images of indoor places such as bedrooms, kitchens or restaurants. There are 365 classes with 5000 images in each class. The two CNNs we explain are *ResNet-18* [14] (Use Case 1) and *VGG-16* [23] (Use Case 2), which have achieved state-of-the-art accuracy in image classification at some point during the last few years. Each CNN is pretrained using images from all 365 classes of the Places dataset. Next, we modify the classification layer of ResNet-18 to focus on three classes (kitchen, bedroom, living room), fix the non-classification layers of the CNN, and fine tune the classification layer using 70% of the images from these three classes (a total of 0.7 * 3 * 5000 images) in Places, and evaluate/explain on the other 30% of images from these three classes (a total of 0.3 * 3 * 5000 images). Similarly, we modify VGG-16 to focus on two classes: coffee shop vs. restaurant.

To evaluate concept identification, we define *concept relevance* as the fraction of concepts that are relevant to the target dataset. We consider a concept to be relevant if it is present in at least one image in the target dataset that is attributed to that concept.

To evaluate concept attribution, we use the concept-saliency overlap defined in Section 3.2. For both CNN2DT and POEM, we compute the average concept-saliency overlap over the concepts attributed to all the images in the target dataset.

To evaluate the quality of pattern explanations, we borrow criteria used in related work [10, 13, 17]. *Pattern size* is the number of concepts used in a pattern. *Support* is the fraction of images matching the concepts within a pattern. *Confidence* is the fraction of supporting images that also match the CNN prediction stated in the pattern. *Information gain* measures how well a set of patterns captures the distribution of an outcome (computed as in [10]), which for us corresponds to the CNN prediction.

For CNN2DT, we compute these criteria for the patterns found by CART based on the concepts found. For POEM, we evaluate patterns from CART, IDS and Explanation Tables. This allows us to compare CART patterns from CNN2DT with CART patterns from POEM (whose concepts may be different), and also to determine if IDS and explanation tables can provide better patterns than CART.

Because minimum support is a parameter in each of these pattern mining methods, we experiment with a range of minimum support values, including 0.01, 0.03, 0.05, 0.1, 0.15 and 0.2. For each value, we evaluate the same top number of patterns from different methods, for a fair comparison regardless of the number of patterns produced by each method. Finally we report the average value of each criterion over all the tested minimum support values.

4.3 Quantitative Evaluation

4.3.1 Concept Identification. Table 1 shows the concept relevance score computed for CNN2DT and POEM for Use Case 1 (top) and 2 (bottom). Concepts identified by POEM are all relevant to the target dataset, while a significant fraction of the CNN2DT concepts are not. We will examine these concepts more closely in Section 4.4.1.

[4]https://github.com/CSAILVision/NetDissect-Lite

Table 1: Concept relevance and concept-saliency overlap for CNN2DT and POEM for Use Case 1 (top) and 2 (bottom)

	CNN2DT	*POEM*
concept relevance	0.63	**1.0**
concept-saliency overlap	0.07	**0.67**

	CNN2DT	*POEM*
concept relevance	0.77	**1.0**
concept-saliency overlap	0.12	**0.75**

Table 2: Pattern statistics for CNN2DT and POEM for Use Case 1 (top) and 2 (bottom)

	CNN2DT	*POEM CART*	*POEM Exp*	*POEM IDS*
pattern size	2.34	**1.52**	3.61	1.88
support	0.13	0.16	0.16	**0.17**
confidence	**0.94**	0.91	0.89	0.9
information gain	660	1038	**1045**	1044

	CNN2DT	*POEM CART*	*POEM Exp*	*POEM IDS*
pattern size	3.25	**1.79**	4.17	2.97
support	0.14	0.16	0.16	**0.41**
confidence	0.7	0.71	**0.73**	0.67
information gain	62	80	**103**	82

4.3.2 Concept-Saliency Overlap. As Table 1 shows, concepts attributed to images by POEM overlap 6-10 times more highly with the high-importance image areas. In Section 4.4.2, we will show examples of high importance areas highlighted on images to illustrate this performance difference in concept attribution.

4.3.3 Pattern Mining. Table 2 shows the average size, support, confidence and information gain of patterns from CNN2DT and the three methods used in POEM for Use Case 1 (top) and 2 (bottom). When comparing CART patterns from CNN2DT with CART patterns from POEM, we see that POEM patterns are more concise and provide higher support and information gain. CNN2DT has higher confidence, but the concepts found by CNN2DT may not always be relevant, leading to spurious explanations (as we show qualitatively in Section 4.4.3).

Comparing the rule mining methods used in POEM, we see that IDS patterns obtain higher support, while explanation tables provide more informative patterns. This suggests that a set of methods can provide a wide range of explanations along multiple explainability criteria of interest.

4.3.4 Concept Filtering. IDS and Explanation Tables become very slow for more than 15 features (concepts). These rule mining methods filter out weaker concepts naturally by generating patterns that include more informative and high-support concepts. However, they first need to generate and evaluate a large set of candidate patterns, which can potentially lead to poor running times.

Table 3: Concepts identified by CNN2DT (top) and POEM (bottom) in Use Case 1

Objects: building, bed, airplane, person, pool table, car, water, dog, bus, shelf, road, train, house, grass, tree, mountain, windowpane, bird, plant, ceiling, skyscrapper, table, sofa, sky, chair, seat, sea, horse, bathtub, book, cabinet, painting, bottle, stairway, boat, cat, stove, mirror, wardrobe, fence, snow
Parts: screen, wheel, head, coach, torso, body, crosswalk, shop window, hair
Materials: food, tile
Colours: red, orange

Objects: person, work surface, bed, sofa, window, plant, fireplace, chair, stove, curtain, shelf, pane, door, painting, cushion, building, pillow, drawers
Parts: top, footboard, back pillow
Materials: polished stone, fur, tile, brick
Colours: red, blue, green, orange, pink

Table 4: Concepts identified by CNN2DT (top) and POEM (bottom) in Use Case 2

Objects: building, person, tree, road, grass, ceiling, plant, mountain, bed, car, windowpane, airplane, floor, water, bus, chair, table, sky, ground, painting, train, horse, cat, skyscrapper, cabinet, sofa, curtain, book, shelf, pool table, boat, sink, fence, dog, work surface, sidewalk, stairway, seat, stove, rock, house, sea, snow, track, toilet, motorbike, field, chandelier, swivelchair
Parts: wheel, body, screen, head, crosswalk, drawer, torso, hair, roof, leg, seat cushion, shop window, coach, arm, balcony, top, pot, back, column
Materials: food, carpet, wood, glass, painted, tile, metal, fabric, brick
Colours: red, yellow, white, blue, pink, green

Objects: food, ceiling, window, tree, sky, chair, cup, plant, shelf, plate, building, sidewalk, floor, painting, signboard, person, bar, curtain, bottle, case, sea, road, awning, cabinet, umbrella, car, column, chandelier
Parts: top, head, torso, body
Materials: food, hair, carpet, skin, paper, brick, ceramic, tile
Colours: blue, green, yellow, red, white, pink

As we mentioned in Section 3.2, we filter out concepts attributed to less than 1% of data before the pattern mining step. To demonstrate the resulting performance improvements, we evaluate the running time and the pattern quality criteria when we apply the rule mining methods with and without concept filtering. For this experiment, we focus on a smaller subset of data including 500 images for each class in Use Case 1 and 2, so that the case without concept filtering can finish in reasonable time.

The results are shown in Table 5. In Use Case 1, concept filtering reduces the number of concepts from 10 to 5, and improves the running time by a factor of more than 50, while the set of generated patterns does not change at all. For Use Case 2, the number of concepts drops from 11 to 3, leading to more than 17 times better performance. Some of the pattern criteria improve when filtering the concepts, including pattern size (conciseness) for all three methods. Other criteria get worse by factors between 0.1 to 2.5.

In terms of end-to-end running time, POEM was nearly four times faster than CNN2DT in Use Case 1 (676 vs. 2571 seconds) and nearly ten times faster in Use Case 2 (684 vs. 6646 seconds). POEM spent more time on pattern mining (especially on IDS) but much less time on concept identification and attribution.

4.4 Qualitative Evaluation

We now examine the concepts identified by CNN2DT and POEM, and we analyze sample images with their high-importance areas

Table 5: Running time and pattern statistics with and without concept filtering for Use Case 1 (top) and 2 (bottom)

	without filtering			*with filtering*		
method	*CART*	*Exp*	*IDS*	*CART*	*Exp*	*IDS*
running time (seconds)	1	11	541	**0.2**	**8**	**2**
pattern size	2	4.5	1	2	4.5	1
support	0.04	0.34	0.04	0.04	0.34	0.04
confidence	0.85	0.63	0.85	0.85	0.63	0.85
information gain	30.5	24	30.5	30.5	24	30.5

	without filtering			*with filtering*		
method	*CART*	*Exp*	*IDS*	*CART*	*Exp*	*IDS*
running time (seconds)	1	11	185	**0.3**	**7**	**4**
pattern size	2	5	1.6	**1.5**	**1**	**1**
support	0.01	**0.05**	0.33	0.01	0.02	**0.5**
confidence	0.99	0.89	**0.85**	0.99	**0.99**	0.78
information gain	**7.9**	6	**5.7**	5.5	6	3.2

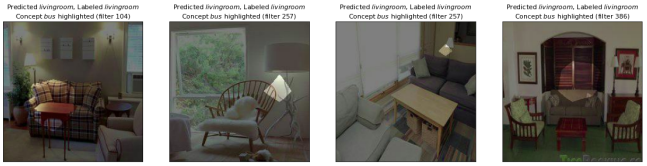

Figure 4: Images from Use Case 1 with 'bus' high-activation areas found by CNN2DT highlighted (pattern 7 in Figure 11)

Figure 5: Images from Use Case 2 with 'toilet' high-activation areas found by CNN2DT highlighted (pattern 8 in Figure 12)

highlighted to determine if these areas correspond to locations of concepts. We focus on the top 10 patterns from CNN2DT and POEM. For POEM, we merge the patterns found by its three rule mining methods and rank them by $\frac{\text{support} \times \text{confidence}^2}{\text{size}^2}$. This ordering emphasizes concise and confident patterns.

4.4.1 Concept Identification. Table 3 shows the concepts identified by CNN2DT (top) and POEM (bottom) for Use Case 1, grouped by their categories. CNN2DT finds some relevant concepts (bed, bottle, cabinet, sofa, stove) but also many others unlikely to appear in the target dataset (airplane, bathtub, bus, car, horse, train). Figure 4 shows sample images associated with the 'bus' concept by CNN2DT, with the corresponding 'bus' filter high-activation areas highlighted. Clearly, buses do not exists in these images. In contrast, all the concepts identified by POEM in Use Case 1 are relevant, as also seen in our quantitative evaluation in Section 4.3.1.

Figure 6: Images from Use Case 2 with 'sea' high-importance areas found by POEM highlighted

Figure 7: Images from Use Case 1 with 'sofa' high-activation areas found by CNN2DT highlighted (pattern 8 in Figure 11)

Table 4 shows the concepts identified by CNN2DT (top) and POEM (bottom) for Use Case 2. Again, CNN2DT finds many concepts that are unlikely to appear in the images of coffee shops and restaurants such as airplane, bed, car, motorbike, mountain, toilet, etc. Figure 5 shows the high activation areas for some images attributed to the 'toilet' concept. None include anything resembling a toilet, though some are showing a coffee cup, which may be similar to the body of a toilet and the water inside it. In fact, the same filters that are activated on cups may be activated when images of toilets from the secondary dataset were passed through them.

On the other hand, the concepts identified by POEM in Table 4 are consistent with coffee shops and restaurants, such as bottle, chair, cup, food, plate, shelf, etc. Some concepts may not seem relevant at first glance, such as sea, sky, and tree. To check this further, in Figure 6 we show sample images from the target dataset with the 'sea' concept highlighted. All of these images show outdoor coffee shop or restaurant views where the sea is part of the landscape and has led to activations of the corresponding filters in the model.

4.4.2 Concept Attribution. We now examine high activation areas related to relevant concepts identified by CNN2DT to check their overlap with the location of the concept in the image. Figure 7 shows sample images from Use Case 1 with their 'sofa' concept high activation areas highlighted. In some images, either a sofa does not exist or the high-activation areas do not match the location of the sofa in the image. These examples imply that the concepts attributed to images using CNN2DT may not be reliable indicators of what the CNN model pays attention to, as we saw in Section 4.3.2.

Figure 8 shows sample images that POEM attributed to the same concept 'sofa', with the related high-importance areas highlighted. The highlighted areas cover sofas either partially or completely.

Figure 9 shows images from Use Case 2 associated with the 'red' concept by CNN2DT. While some of these correctly highlight the red colour, others either do not include a red area, or the high activation areas do not cover any red pixels. Figure 10 shows sample images attributed to the 'red' colour using POEM, where the related high-importance areas correspond to the red locations in images.

Figure 8: Images from Use Case 1 with 'sofa' high-importance areas found by POEM highlighted (pattern 2 in Figure 11)

Figure 9: Images from Use Case 2 with 'red' high-activation areas found by CNN2DT highlighted

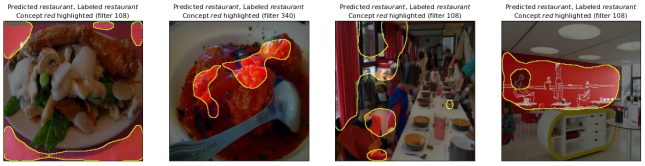

Figure 10: Images from Use Case 2 with 'red' high-importance areas found by POEM highlighted (pattern 5 in Figure 12)

4.4.3 Pattern Mining and Analysis. Figure 11 shows the top 10 patterns mined by CNN2DT (top) and POEM (bottom) for Use Case 1. For each row (pattern), the concepts are labelled with "yes" or "no"; a blank means that the given concept's existence or lack thereof is not relevant. Some patterns produced by CNN2DT are intuitive, e.g., pattern 2 indicating that if the concepts 'bottle', 'cabinet' and 'chair' are attributed to an image, then the CNN classifies it as a kitchen in 98% of cases. We also see patterns with lower confidence levels, which include some of the irrelevant concepts mentioned earlier such as the 'bus' concept in patterns 7 and 9.

Furthermore, all the patterns generated by CNN2DT include two or more concepts, which leads to relatively narrow patterns. For example, there are no single-concept patterns capturing important concepts such as beds or stoves. On the other hand, Figure 11 shows that POEM finds several high-confidence single-concept patterns such as patterns 1, 2, 4 and 7. These patterns highlight the importance of the concept 'bed' for predicting bedrooms, 'sofa' for living rooms, and 'work surface' and 'tile' for kitchens. From these patterns, we learn that the model appears to be paying attention to the right concepts, which can build end user trust and assist model engineers with model validation prior to deployment.

Figure 12 shows the top patterns mined using CNN2DT (top) and POEM (bottom) for Use Case 2. As in Use Case 1, patterns extracted by CNN2DT include several concepts. Furthermore, the issues with concept identification and attribution lead to patterns that may not explain the model accurately. In contrast, patterns found by POEM include fewer concepts, and we learn that the

	bathtub	bed	bottle	bus	cabinet	car	chair	screen	sofa	stove	train	PREDICTION	METHOD
1			yes		no							kitchen	CART
2			yes		yes		yes					kitchen	CART
3		no	yes		yes		no					kitchen	CART
4			no		yes				no	yes		kitchen	CART
5		yes	yes		yes		no					kitchen	CART
6			no		yes				no	no		kitchen	CART
7			no	yes			yes		yes			livingroom	CART
8			no	no		no	yes		yes			livingroom	CART
9			no	yes			no	yes	yes		no	bedroom	CART
10		no	no		no				no			livingroom	CART

	bed	curtain	green	orange	person	red	sofa	tile	window	work surface	PREDICTION	METHOD
1	yes										bedroom	Exp, IDS, CART
2							yes				livingroom	IDS
3	yes		no	no							bedroom	Exp
4										yes	kitchen	IDS
5	no		no							no	livingroom	Exp
6	no						yes				livingroom	CART
7								yes			kitchen	IDS
8	no				no		no		no		kitchen	CART
9	no		no			no	no	no			kitchen	Exp
10	yes	no		no		no	no		no		bedroom	Exp

Figure 11: Top patterns for Use Case 1 for CNN2DT (top) and POEM (bottom)

	balcony	blue	book	chandelier	coach	green	motorbike	painting	pool table	seat	shelf	swivel chair	toilet	work surface	PREDICTION	METHOD
1									yes	no			no	yes	coffeeshop	CART
2	no				no		no			yes					coffeeshop	CART
3			yes		yes			yes		yes	no				restaurant	CART
4									no	no			no		coffeeshop	CART
5		yes							yes	no			no	no	coffeeshop	CART
6					no	no	yes			yes					restaurant	CART
7		yes						yes		no		yes	yes		coffeeshop	CART
8		no			no		no			no			yes		coffeeshop	CART
9			no		yes				yes	yes	no				restaurant	CART
10					yes				no	yes	no				restaurant	CART

	building	ceiling	head	red	shelf	top	PREDICTION	METHOD
1		yes					restaurant	IDS
2						yes	restaurant	IDS, CART
3		no		no		no	coffeeshop	IDS
4					yes		coffeeshop	Exp
5				yes			restaurant	Exp, IDS
6				no		yes	restaurant	Exp
7	no					yes	restaurant	Exp
8	no	no	no			no	coffeeshop	CART
9		yes				no	restaurant	CART
10	no				no	yes	restaurant	Exp

Figure 12: Top patterns for Use Case 2 for CNN2DT (top) and POEM (bottom)

model pays attention to concepts such as 'shelf' for coffee shops and 'top' (table top) for restaurants. Patterns 1 and 5 in Figure 12 also link 'ceiling' and 'red' with restaurant predictions, but with a lower confidence. These concepts are not intrinsic to coffee shops or restaurants, but may indicate that the CNN has seen shelves in coffee shop training images, and table tops, ceilings and red colours in restaurant images. These patterns assist model engineers with identifying model weaknesses and issues in the training dataset.

Figure 13: Images matching POEM pattern 2 in Figure 11, but wrongly predicted by the model

Figure 14: Non-matching images of POEM pattern 4 in Figure 12

In general, the patterns found by POEM are more concise and provide more informative explanations about the CNN, as we also demonstrated quantitatively in Section 4.3.3. This may be related to the more accurate concept identification and attribution steps, as well as the inclusion of patterns from Explanation Tables and IDS.

Using POEM, we can also examine other categories, such as non-matching and wrongly predicted images. Figure 13 shows sample images from Use Case 1 matching pattern 2 in Figure 11, but predicted incorrectly as living rooms instead of bedrooms or kitchens. These examples either include a sofa in the bedroom or a bed similar to a sofa. These patterns can assist model engineers with model debugging and data augmentation by revealing concepts associated with model mispredictions.

To further analyze POEM patterns from Use Case 2, we examine sample images from other categories. Figure 14 shows non-matching images of pattern 4 from Figure 12, which are attributed to 'shelf' but predicted as restaurants rather than coffee shops. The leftmost image is predicted incorrectly as a restaurant. However, the other three examples are less obvious, and may even be mislabeled as restaurants. This illustrates the value of POEM's patterns in finding potential errors in training data labelling.

5 CONCLUSIONS

POEM has several limitations that lead to directions for future work. The network dissection method we use for concept identification assumes that each CNN filter focuses on one concept. Recent work has questioned this assumption [6, 9, 28], suggesting that some filters may be learning multiple concepts, or may jointly learn a single concept. For concept attribution, we combine filter activations and gradients as indicators of what the CNN is paying attention to. An interesting direction for future work is to explore concept pattern mining using *counterfactual* methods. Instead of analyzing filter activations, these methods perturb an image to determine if masking some pixels leads to a different prediction by the model [1, 12, 25]. Finally, we plan to extend POEM's identification, attribution and pattern mining modules to other models such as transformers.

REFERENCES

[1] Arjun R. Akula, Keze Wang, Changsong Liu, Sari Saba-Sadiya, Hongjing Lu, Sinisa Todorovic, Joyce Chai, and Song-Chun Zhu. 2022. CX-ToM: Counterfactual explanations with theory-of-mind for enhancing human trust in image recognition models. *iScience* 25, 1 (2022), 103581.

[2] O. Barkan, O. Armstrong, A. Hertz, A. Caciularu, O. Katz, I. Malkiel, and N. Koenigstein. 2021. GAM: Explainable Visual Similarity and Classification via Gradient Activation Maps. In *CIKM*. 68–77.

[3] D. Bau, B. Zhou, A. Khosla, A. Oliva, and A. Torralba. 2017. Network Dissection: Quantifying Interpretability of Deep Visual Representations. In *CVPR*. 3319–3327.

[4] D. Bau, J.-Y. Zhu, H. Strobelt, A. Lapedriza, B. Zhou, and A. Torralba. 2020. Understanding the role of individual units in a deep neural network. *Proceedings of the National Academy of Sciences* (2020), 30071–30078.

[5] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. 1984. *Classification and Regression Trees*. Wadsworth and Brooks, Monterey, CA.

[6] Z. Chen, Y. Bei, and C. Rudin. 2020. Concept whitening for interpretable image recognition. *Nature Machine Intelligence* 2 (12 2020), 1–11.

[7] V. Dadvar, L. Golab, and D. Srivastava. 2022. POEM: Pattern-Oriented Explanations of CNN Models. *PVLDB* 15, 12 (2022), 3618–3621.

[8] R. El Shawi, Y. Sherif, and S. Sakr. 2021. Towards Automated Concept-based Decision Tree Explanations for CNNs. In *EDBT*. 379–384.

[9] R. Fong and A. Vedaldi. 2018. Net2Vec: Quantifying and Explaining How Concepts are Encoded by Filters in Deep Neural Networks. In *CVPR*. 8730–8738.

[10] K. El Gebaly, G. Feng, L. Golab, F. Korn, and D. Srivastava. 2018. Explanation Tables. *IEEE Data Engineering Bulletin* 41 (2018), 43–51.

[11] A. Ghorbani, J. Wexler, J. Zou, and B. Kim. 2019. Towards Automatic Concept-based Explanations. In *NeurIPS*, Vol. 32.

[12] Yash Goyal, Ziyan Wu, Jan Ernst, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Counterfactual Visual Explanations. In *ICML*. 2376–2384.

[13] Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi. 2018. A Survey of Methods for Explaining Black Box Models. 51, 5, Article 93 (aug 2018), 42 pages.

[14] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *CVPR*. 770–778.

[15] S. Jia, P. Lin, Z. Li, J. Zhang, and S. Liu. 2020. Visualizing Surrogate Decision Trees of Convolutional Neural Networks. *J. Vis.* 23, 1 (feb 2020), 141–156.

[16] B. Kim, M. Wattenberg, J. Gilmer, C. Cai, J. Wexler, F. Viégas, and R. Sayres. 2018. Interpretability Beyond Feature Attribution: Quantitative Testing with Concept Activation Vectors (TCAV). In *ICML*.

[17] H. Lakkaraju, S. Bach, and J. Leskovec. 2016. Interpretable Decision Sets: A Joint Framework for Description and Prediction. In *KDD*. 1675–1684.

[18] A. Mahendran and A. Vedaldi. 2016. Visualizing Deep Convolutional Neural Networks Using Natural Pre-Images. *International Journal of Computer Vision* 120 (12 2016), 233–255.

[19] S. Nakandala, A. Kumar, and Y. Papakonstantinou. 2019. Incremental and Approximate Inference for Faster Occlusion-based Deep CNN Explanations. In *SIGMOD*. 1589–1606.

[20] R. Pradhan, A. Lahiri, S. Galhotra, and B. Salimi. 2022. Explainable AI: Foundations, Applications, Opportunities for Data Management Research. In *ICDE*. 3209–3212.

[21] R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. 2017. Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization. In *ICCV*. 618–626.

[22] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2014. Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps. In *ICLR*.

[23] Karen Simonyan and Andrew Zisserman. 2015. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *ICLR*.

[24] J. Townsend, T. Kasioumis, and H. Inakoshi. 2021. ERIC: Extracting Relations Inferred from Convolutions. In *ACCV*. 206–222.

[25] S Wachter, B Mittelstadt, and C Russell. 2018. Counterfactual explanations without opening the black box: automated decisions and the GDPR. *Harvard Journal of Law and Technology* 31, 2 (2018), 841–887.

[26] T. Xiao, Y. Liu, B. Zhou, Y. Jiang, and J. Sun. 2018. Unified Perceptual Parsing for Scene Understanding. In *ECCV*. 432–448.

[27] Matthew D. Zeiler and Rob Fergus. 2014. Visualizing and Understanding Convolutional Networks. In *ECCV*. 818–833.

[28] Q. Zhang, Y. Yang, H. Ma, and Y. Wu. 2019. Interpreting CNNs via Decision Trees. In *CVPR*. 6254–6263.

[29] Bolei Zhou, Aditya Khosla, Àgata Lapedriza, Aude Oliva, and Antonio Torralba. 2015. Learning Deep Features for Discriminative Localization. *CoRR, arXiv* abs/1512.04150 (2015).

[30] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba. 2017. Places: A 10 million Image Database for Scene Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2017).