INTERPRETING DEEP MODELS BY EXPLAINING THEIR PREDICTIONS

Toon Meynen *Hamed Behzadi-Khormouji* *José Oramas*

University of Antwerp, imec-IDLab

ABSTRACT

We propose a method that exploits the feedback provided by visual explanation methods combined with pattern mining techniques in order to identify the relevant class-specific and class-shared internal units. In addition, we put forward a patch extraction approach to find faithfully class-specific and class-shared visual patterns. Contrary to the common practice in literature, our approach does not require pushing augmented visual patches through the model. Experiments on different CNN architectures show the effectiveness of the proposed method.

***Index Terms*—** Model Interpretation, Data Mining.

1. INTRODUCTION

Interpretation methods [1, 2, 3, 4, 5] aim at the study of the representations internally encoded by Convolutional Neural Networks (CNNs). Despite the effectiveness of these methods at providing insights on this internal representation, they suffer from two weaknesses. First, except for [6, 7, 8], it has not yet been demonstrated, from which relevant and critical internal units of a model, the visual feedback are drawn. Second, in some works [9, 10, 1], it is a common practice to generate a pool of visual patches per class from a limited set of images as part of the interpretation procedure. To achieve this, the patches need to be resized to the input size expected by the model. This results in the model observing patterns with size and aspect ratios that significantly differ from those seen during training [11]. Thus, the provided interpretation feedback cannot guarantee that it reflects the actual representations encoded within the model.

To address the above weaknesses, we propose a new perspective on interpreting CNNs by exploiting the capabilities of the well-studied model explanation methods (Fig. 1). This type of methods produce visualizations which highlights important region(s) of the input determining the prediction made by the model. To efficiently identify critical internal units, we first utilize a model explanation method to generate visual explanation per input image. Second, a faithful patch generation procedure is proposed to extract visual patches containing the highlighted regions in the visual explanation. The faithfulness refers to the process where extraction operation utilizes the actual internal representations computed from images with the size and aspect ratios used during the training phase. Third, the method utilizes indices of the internal units corresponding to the visual patches from the highlighted regions to create a transaction dataset. Finally, frequent itemset mining and association rule mining techniques are applied to the transaction dataset. This results in a set of class-specific and class-shared rules referring to the relevant internal units to the group of visual patches with similar patterns.

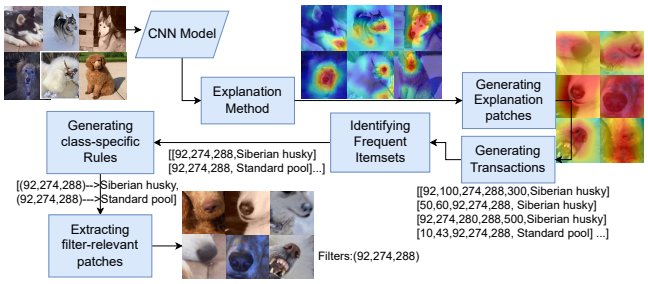

Fig. 1. Proposed Interpretation method. Visual explanations are generated by an explanation method, followed by extracting explanation patches showing highlighted parts. Transaction dataset is created via the patches. Finally, a pattern mining algorithm extracts class-specific and class-shared convolutional filters and their corresponding visual patches.

2. RELATED WORK

Regarding the capability of identifying relevant internal units, [7, 8] utilizes external datasets with annotated visual attributes in order to link internal units w.r.t. such annotations. [6] follows a similar methodology in [7], but instead of looking for relationships w.r.t visual attributes, it looks for relationships with the classes of interest themselves. Different from [7], our method is independent of pre-defined attributes. Also, opposite to [6], our method is able to identify both class-shared and class-specific internal units. [10] conduct association rule mining, via the *Apriori* algorithm [12], on activations from the fully-connected layer computed from cropped image patches fed to the model. Differently, we use the activations produced by the last convolutional layer from complete images. Moreover, we extract the visual patterns from visualizations generated in an explanation-based approach. Furthermore, we use the *eclat* algorithm [13], which

has been proved faster than *Apriori* [14, 15], for frequent itemset mining. Finally, our procedure is conducted in a hierarchical manner to obtain the class-specific and class-shared frequent itemsets.

3. PROPOSED METHOD

We summarize the proposed method in the following steps: First, image patches are extracted from the visualizations produced by an explanation algorithm. Second, a transaction database from the extracted patches is created. Each record of this database contains indices of the internal units(s) and the predicted class label per image patch. Third, a set of association rules are learned by finding frequent itemsets from the internal units indices in the database. These rules assist with the interpretation of the model by identifying relevant class-specific and class-shared units. Finally, we generate visualizations of the input features encoded by these rules.

3.1. Patch Extraction

Consider an image dataset $D{=}\{X^i, Y^i\}_{i=1}^n$ containing n pairs of image examples X^i with their corresponding class label Y^i. Also, consider A^i, with shape $w{\times}h{\times}d$, as the activation maps produced by d filters in the last convolutional layer and $\hat{Y}^i$ as the predicted label by a model for the input example X^i. In the first step, we utilize Grad-CAM [16] to find input regions relevant to the class $\hat{Y}^i$ predicted by the model. More specifically, given an input image X^i, the explanation heatmap H^i is obtained from the last convolutional layer of the model via Grad-CAM. This is followed by resizing H^i to the size of the input image. Next, the method extracts a set of image patches with the highest importance to the given prediction. To do so, a two-step patch extraction procedure w.r.t the explanation heatmap H^i is applied. We will now describe these two steps.

Candidate patch extraction. By applying a cropping window W with size $\omega{\times}\omega$ at random locations we generate a set of random patches, $P^i{=}\{p^{i,1}...p^{i,m}\}$, pertaining to the input X^i. Next, through Eq. 1, the patches are scored based on the summation of elements at every location (u, v) in the region W from the explanation heatmap H^i.

$$S^i = \{s^{i,j} | s^{i,j} = \sum_{u,v \in [1...\omega]} H[W(u,v)]\}, \quad \forall j \in [1,m] \quad (1)$$

where $s^{i,j}$ indicates the score of the patch $p^{i,j}$ and S^i is the set of scores for the patches of the input X^i. In the next step, the patch $p^{i,j}$ with the highest score is selected and added to a list named P'^i. This procedure of extracting candidate patches is repeated k times. At each time, we update the explanation heatmap H^i by setting the region corresponding to the selected high-scoring patch $p^{i,j}$, to zero (i.e., $H^i[p^{i,j}]{=}0$). This is done to avoid extracting duplicate patches in each iteration. Finally, this procedure results in a set P'^i containing k candidate patches.

Relevant Patch Selection. To select the relevant patches, our method follows an incremental perturbation procedure. Considering the obtained set P'^i containing the k candidate patches and a set of relevant patches $P^{*i}{=}\emptyset$, the image X^i is perturbed by setting its pixel values corresponding to a patch to zero. Then, the perturbed image is pushed through the model to obtain the output score for class label Y^i.

More specifically, the input image X^i is perturbed independently by each of the patches $p^{i,b}{\in}P'^i$ until the difference of the output score from class Y^i between two sequential perturbation operations reaches below a threshold τ_{prt}. In parallel, the list P^{*i} is gradually grown by adding the new patches $p^{i,b}$ considered in each iteration. Through this two-step procedure, the method can identify patches closely centred in the highlighted region(s) from explanation heatmap H^i. Moreover, it allows to extract multiple relevant patches that might occur when an explanation heatmap H^i highlights multiple non-overlapping regions in the input image. This procedure is repeated for all images X^i in the dataset. As a result, the method outputs a list $P{=}\{P^{*1}...P^{*n}\}$ including the subsets P^{*i} that contain the important patch(es) for each example i.

3.2. Transaction Database Generation

Transactions are sets of discrete items usually received as input by association rule mining algorithms. Hence, we aim to convert each image patch $p^{i,b}{\in}P^{*^i}$ into a representation of N discrete values $T^{i,b}{=}\{t^{i,1}...t^{i,N}\}$, namely patch-transaction. The first $N{-}1$ elements of this transaction refer to the index of convolutional filters, while the last element indicates the predicted class label $\hat{Y}^i$. To identify $N{-}1$ convolutional filters per image patch, we utilize the activation maps A^i. These activation maps A^i are resized to the size of the input image X^i. Then, the window corresponding to the image patch $p^{i,b}$ on the activation maps is cropped and considered as $A^{p^{i,b}}$ with size $\omega{\times}\omega{\times}d$, where d indicates the number of the filters in the convolutional layer. Next, following Eq. 2, we calculate the activation response $a_f^{i,b}$ from the activation map $A_f^{p^{i,b}}$ for each filter f.

$$a_f^{i,b} = \sum_{u,v=0}^{\omega} A_f^{p^{i,b}}(u,v), \quad \forall f \in [1..d] \quad (2)$$

Next, the index of the $(N{-}1){-}top$ filters with the highest activation responses $a_f^{i,b}$ are selected as the element in the transaction $T^{i,b}$. This procedure is repeated for the patches of images from each class in the dataset. This leads to a transaction dataset T^c containing transactions $T^{i,b}$ pertaining to the patch $p^{i,b}$ of the example X^i from predicted class label $\hat{Y}^c$. Accordingly, $T{=}\{T^1...T^c\}$ where $c = [1...C]$ indicates the set of transaction dataset related to each class c.

3.3. Frequent Itemsets Mining

In this step, the method mines the transaction database $T=\{T^1...T^c\}$ to identify frequent itemsets of convolutional filter indices. The mined itemsets are both class-specific as well as commonly occurring among different classes, i.e. class-shared. Moreover, their *support* measurement [13] must be higher than a threshold $\tau_{support}$. We achieve this by using *eclat* algorithm [13]. This algorithm uses the *support* metric to identify the importance of an itemset. The common practice in the literature is applying eclat once on the entire transaction dataset. Different from this, we apply it in a hierarchical manner as the transactions in the database T are related to the different classes.

In the first level, we identify the frequent itemsets for each transaction subset T^c. This step results in a set of class-specific frequent itemsets, i.e., $I=\{I^1, ..., I^c\}$. It is worth noting that the identified frequent itemsets in each set I^c contain the class label c as one item in the itemset. Hence, the *support* measurement of the itemsets in I is biased to each class. Also, there are some frequent itemsets, including just convolutional filter indices, which appear in different classes with different support measurements. To cope with this issue, in the second level, we re-calculate the *support* of all identified frequent itemsets in I w.r.t. the transaction database T, instead of considering the transaction database T^c. That is, we drop the class label item from the database. This results in a new set I' with new support. Its frequent itemsets are not directly class-specific, but refer to the convolutional filter indices which frequently occur in the transaction database for different class labels.

3.4. Association Rule Mining

In this section we describe how association rule mining algorithms [17] are used to mine a set of class-specific rules R^c in the form of $(A \rightarrow B)$ from the identified frequent itemsets (Sec. 3.3). The notation A and B refer to the antecedent and the consequent of a rule, respectively. In this algorithm [17], any item of frequent itemset can be in the antecedent or the consequent part. Different from this, we apply the constraint on the followed association rule mining apporach, where A indicates only the frequent itemset of the convolutional filters indices and B refer only to the class label. The reason for that is to find the relation between convolutional filters and a class label and not between filters and filters. The association rule mining calculates *support* and *confidence* values for each generated rule. Finally, we select the $q-top$ rules for each class c following $R^c=\{R_e^c | R_e^c \in \{\arg\max_q(Supp(R_e^c) * Conf(R_e^c))\} \quad \forall c \in [1...C] \ \& \ e \in [1...E-1]\}$ where, the term $(Supp(R_e^c) * Conf(R_e^c))$ indicates the score of a rule. $Supp$ and $Conf$ stand for the support and confidence values of the rule R_e^c. E indicates a set of generated rules for each class. Finally, we form set $R=\{R^1...R^c\}$ that contains information about the critical internal convolutional filters relevant to each class.

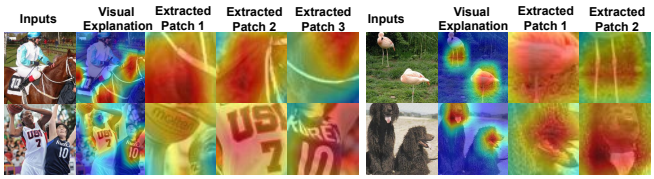

Fig. 2. Example of multiple visual patches extracted from visual explanation, proposed by the patch generation operation.

Class and filter-relevant visual patterns. Given the association rule set R, the method produces two outcomes: (1) G_R a set containing image patches relevant to a group of class-specific filters, and (2) G_F a set containing a group of image patches relevant to a group of filters. The antecedent and the consequent of a rule represent the convolutional filter indices and class label, respectively. Also, these rules refer to a set of frequent itemsets in I which have been mined from transaction dataset T. Since, each transaction was created from an image patch, then we are able to retrieve a group of image patches indexed to the convolutional filter indices in the generated association rules in the set R. Similarly, the antecedent of a rule can appear in the rules related to different classes. Hence, we can retrieve the image patches encoded by a group of filters, i.e, visual patterns shared among classes.

4. EVALUATION

We evaluate our method on the VGG16 [18] and Resnet50 [19] architectures, pre-trained on the *ImageNet* dataset (1000 classes). Results are reported on the validation set, we compare our performance w.r.t. [6]. Based on internal tests, we set variables q=10, τ_{prt}=0.1, and $\tau_{support}$=0.5.

4.1. Qualitative Analysis: Visual Patterns

The *Patch Extraction* operation of the proposed method (Sec.3.1) is able to extract visual patches that are closely centered in the highlighted region(s) from explanation heatmap (Fig. 2). It prevents the model from computing activation maps for massive number of re-scaled patches as a common practice in the literature [9, 10, 1].

Visual Patterns from Class-shared Units. We aim to illustrate the visual pattern related to a group of filters shared among classes according to Sec. 3.4. The illustration in Fig. 3 over Resnet50 and VGG16 reveals two interesting observations. First, consider Fig. 3 (bottom-left) which shows examples of visual patterns from classes *Siberian husky*, *Alaskan malamute*, *Eskimo dog*, and *Greater Swiss Mountain dog* encoded by the convolutional filters *1583* and *1634* in Resnet50. This suggests that classes with visual similarities share convolutional filters. We have observed a similar trend on VGG19 (Fig. 3 bottom-right).

Table 1. Model accuracy comparison when internal units identified by each of the methods are perturbed.

Models/Methods	Ours-5 Acc. / avg. perturb.	Ours-10 Acc. / avg. perturb.	Ours-25 Acc. / avg. perturb.	[6] Acc. / avg. perturb.	Random Acc. / avg. perturb.	Baseline Acc. / avg. perturb.
VGG16	55.17%±0.20 / 6.79	**45.67%±0.20 / 10.20**	53.91%±0.20 / 8.80	50.53%±0.20 / 5.40	71.39%±0.17 / 10.00	71.59%±0.17 / 0.00
ResNet50	71.11% ±0.17 / 6.79	**68.63% ±0.18 / 10.20**	71.26% ±0.17 / 8.80	69.74% ±0.18 / 5.44	76.08% ±0.16 / 10.00	76.13%±0.16 / 0.00

Fig. 3. Example of visual patterns from different classes relevant to a group of filters in Resnet50(left) and VGG16 (right).

Second, similar patterns in Fig. 3 (top-right) are related to classes with dissimilar objects such as *cardoon*, *globe artichoke*, *pineapple*, *fig*, *sea anemone*, and *bee*. A similar trend can be seen in Fig. 3 (top-left). Despite visual patterns showing a fish, there are two examples from the class *Ambulance* (fourth row). This reveals that this group of filters have encoded similar patterns from classes with completely different objects. To sum up, the proposed method can identify filters capable of grouping classes based on visual similarities.

Visual Patterns from Class-specific Units. We show the weighted average of patches relevant to a specific class encoded by a given filter. To do so, first, we collect the image patches from each rule for a given filter appearing in the antecedent. Second, we aggregate the elements in the activation region corresponding to a patch and consider it as the weight for that patch. Finally, we compute a weighted average of the patches. The resulting visualization illustrates the class-specific pattern mostly focused on by the filter. Fig. 4 illustrates examples from VGG16 over different classes.

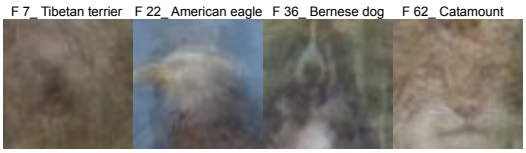

Fig. 4. Weighted average of class-specific images patches encoded with filters in VGG16.

4.2. Quantitative Analysis: Filter Importance

We investigate the model accuracy trends when we perturb the relevant filters identified by each method. We first perturb the identified class-relevant filters by setting their computed activation map to zero. Second, we compute the model accuracy for the considered class. This procedure is repeated for all classes in the dataset. Finally, we report in Table 1 the mean accuracy across all the classes as a function of the number of identified units. This should indirectly assess the performance of the interpretation method at identifying critical internal units. We report performance of the original model (*baseline*), a randomly perturbed model (*Random*), and the method from [6] modified to only consider the convolutional layer. Since the proposed method considers the last convolutional layer, we then modify [6] to consider the last convolutional layer. We consider variants of our method where 5, 10, and 25 top filter indices are selected as the elements of the transactions (Sec. 3.2). As can be seen, *Ours-10* leads to the lowest model accuracy in VGG16 and ResNet50 showing the efficiency of the proposed method in identifying higher number of filters.

5. CONCLUSION

We introduced an interpretation method to identify class-specific/shared internal units by mining explanations of its predictions. We proposed a patch generation method to find visual patterns encoded in the internal units. Experiments showed the efficiency of the proposed method at interpretation task. Future work will focus on identifying units by considering the whole architecture and the use of meta-heuristic approaches for this purpose.

Acknowledgements: This work is partially supported by the UAntwerp BOF DOCPRO4 Project (id 41612) "Multimodal Relational Interpretation for Deep Models", and the FWO Project G0A4720N "Design and Interpret". We would like to thank Salma Haidar for the feedback provided during the preparation of this paper.

6. REFERENCES

[1] Amirata Ghorbani, James Wexler, James Zou, and Been Kim, "Towards automatic concept-based explanations," *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 32, 2019.

[2] Chih-Kuan Yeh, Been Kim, Sercan Ömer Arik, Chun-Liang Li, Pradeep Ravikumar, and Tomas Pfister, "On concept-based explanations in deep neural networks," *In International Conference on Learning Representations (ICLR)*, 2020.

[3] Chaofan Chen, Oscar Li, Alina Barnett, Jonathan Su, and Cynthia Rudin, "This looks like that: deep learning for interpretable image recognition," *Advances in neural information processing systems (NeurIPS)*, vol. 32, 2019.

[4] Vidhya Kamakshi, Uday Gupta, and Narayanan C Krishnan, "Pace: Posthoc architecture-agnostic concept extractor for explaining cnns," in *2021 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2021, pp. 1–8.

[5] Zhi Chen, Yijie Bei, and Cynthia Rudin, "Concept whitening for interpretable image recognition," *Nature Machine Intelligence*, vol. 2, no. 12, pp. 772–782, 2020.

[6] José Oramas, Kaili Wang, and Tinne Tuytelaars, "Visual explanation by interpretation: Improving visual feedback capabilities of deep neural networks," *In International Conference on Learning Representations (ICLR)*, 2019.

[7] Victor Escorcia, Juan Carlos Niebles, and Bernard Ghanem, "On the relationship between visual attributes and convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1256–1264.

[8] Ruth Fong and Andrea Vedaldi, "Net2vec: Quantifying and explaining how concepts are encoded by filters in deep neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8730–8738.

[9] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba, "Object detectors emerge in deep scene cnns," *The International Conference on Learning Representations (ICLR)*, 2015.

[10] Yao Li, Lingqiao Liu, Chunhua Shen, and Anton van den Hengel, "Mining mid-level visual patterns with deep cnn activations," *International Journal of Computer Vision*, vol. 121, no. 3, pp. 344–364, 2017.

[11] Johanna Vielhaben, Stefan Blücher, and Nils Strodthoff, "Sparse subspace clustering for concept discovery (ssccd)," *arXiv preprint arXiv:2203.06043*, 2022.

[12] Rakesh Agrawal, "Fast algorithms for mining association rules in large databases," *VLDB, 1994*, pp. 487–499, 1994.

[13] Mohammed Javeed Zaki, Srinivasan Parthasarathy, Mitsunori Ogihara, Wei Li, et al., "New algorithms for fast discovery of association rules.," in *KDD*, 1997, vol. 97, pp. 283–286.

[14] Pedro F. Felzenszwalb, Ross B. Girshick, David McAllester, and Deva Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 9, pp. 1627–1645, 2010.

[15] V Srinadh, "Evaluation of apriori, fp growth and eclat association rule mining algorithms," *International Journal of Health Sciences*, , no. II, pp. 7475–7485, 2022.

[16] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," in *Proceedings of the IEEE international conference on computer vision (ICCV)*, 2017, pp. 618–626.

[17] Rakesh Agrawal, Tomasz Imieliński, and Arun Swami, "Mining association rules between sets of items in large databases," in *Proceedings of the 1993 ACM SIGMOD international conference on Management of data*, 1993, pp. 207–216.

[18] Karen Simonyan and Andrew Zisserman, "Very deep convolutional networks for large-scale image recognition," *In International Conference on Learning Representation (ICLR)*, 2014.

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," *In Computer Vision and Pattern Recognition Conference (CVPR)*, 2015.