A Patch-Masking Approach to Explain CNNs and Vision Transformers through Class-Specific Impacts

Jean-Marc Boutay[1,*,†], Damian Boquete[1], Deniz Köprülü[1], Ludovic Pfeiffer[1] and Guido Bologna[1,†]

[1]*Department of Computer Science, University of Applied Sciences and Arts of Western Switzerland, Rue de la Prairie 4, 1202 Geneva, Switzerland*

Abstract

Artificial intelligence models are widely used for image classification, notably through complex architectures such as convolutional neural networks (CNNs) and Vision Transformers (ViTs), which are now broadly considered state-of-the-art. In many domains, it is important to be able to understand and explain a model's decision, particularly when these models contain tens of millions, or even billions, of parameters, which greatly complicates the interpretation of their predictions. In this paper, we present a new method for explaining CNNs and ViTs image classification. This method provides a global explanation of the dataset through a set of interpretable rules composed of one or more antecedents. These rules combine pixel-level properties and patch-level features, analyzing the impact of each region on the model's classification. We apply this method to two reference image datasets: MNIST and CIFAR-10. The extracted rules provide faithful and useful explanations, highlighting the important regions of the image during classification, while maintaining classification performance comparable to the standard results obtained with CNNs and ViTs.

Keywords

Image Patches, Model Explanation, Vision Transformers, Convolutional Neural Networks, Explainable Artificial Intelligence

1. Introduction

Artificial intelligence models are playing an increasingly important role in society and in our daily lives. Notable examples include the recent emergence of large language models (LLMs), as well as the widespread use of computer vision models in domains such as medical research, autonomous driving, satellite imagery, security, and many more. In these contexts, convolutional neural networks (CNNs) and Vision Transformers (ViTs) have established themselves as effective and widely adopted solutions. However, in many domains, obtaining a correct prediction is no longer sufficient: it is also crucial to be able to understand and explain the decision of a model.

Popular explainability techniques are feature relevance methods, such as Shapley values [1] and LIME [2]. They assess the contribution of each input feature to the output prediction. While Shapley values assume feature independence, LIME creates a local surrogate model (linear or tree-based) around the instance that needs to be explained. In addition, methods that produce heatmaps, such as GradCam [3], which highlight regions relevant to classification, are often used to understand CNNs. Nevertheless, when classifying objects in images, heatmaps usually show the pixels corresponding to the object. However, this emphasis is the same for all classes and does not distinguish between them.

Rule-based explanations present decisions in a logical way, making them easier to understand. However, their application to deep models is not widely explored in the existing literature due to the complexity of the problem [4]. For example, in [5], each kernel is associated with a specific concept. Specifically, the output of each kernel is quantized as a binary value, and the kernels are labeled using

Late-breaking work, Demos and Doctoral Consortium, colocated with the 4th World Conference on eXplainable Artificial Intelligence: July 01–03, 2026, Fortaleza, Brazil

*Corresponding author.

†These authors contributed equally.

jean-marc.boutay@hesge.ch (J. Boutay)

0009-0000-4138-204X (J. Boutay); 0000-0002-6070-3459 (G. Bologna)

manually established symbols to generate rules. Similarly, in [6], the kernels in the final convolutional layer of a CNN can be associated with multiple concepts rather than just one.

In this study, we present a novel approach to explaining image classification that is applicable to both CNNs and ViTs. Our approach aims to produce comprehensive explanations of the model's behavior in the form of interpretable rules that link pixel properties and different areas of the image to their impact on classification. These areas consist of small patches of size $P \times Q$.

We created a rule extraction algorithm comprising local and global versions, which can be applied to several models. It is called Fidex and FidexGlo and was presented in [7]. In this work, we use the FidexGlo global algorithm[1]. Unlike the methods described above([5] and [6]), which fall within the concept learning domain, our method does not require us to determine concepts in advance or to make strong approximations, such as binarization of convolutional kernels. In fact, our method is flexible enough to also be applied to Vision Transformers.

We previously investigated the usefulness of patches for explaining CNN classification in an earlier work [8]. This first approach was constructive, as we computed the probabilities of each patch contained in the image with respect to the final classification and used them to train another model that aimed to explain the entire image. In the present work, we address the problem using a destructive approach based on patch masking. Specifically, we measure the effect of removing information through masking, allowing us to analyze the direct impact of each patch on a given class within the image classification process. The next sections describe the methods and models, then we present the experiments performed on two benchmark problems, followed by the conclusion.

2. Methods and Models

2.1. Fidex and FidexGlo rule extraction algorithms

We present our two rule extraction algorithms introduced in [7], which operate on a discretized input space defined by axis-parallel hyperplanes induced by the Quantized Interpretable Layer (QIL), a dedicated discretization layer described in [9, 7]. These algorithms primarily rely on the notion of fidelity, which measures how well the rules replicate the model's predictions and is defined as T/S, where S denotes the total number of test samples and T the number of samples for which the rules and the model produce identical responses. Fidex, our local algorithm, generates a rule that explains the model's decision for a given sample. A rule is defined as a conjunction of antecedents of the form

$$A_1, A_2, \ldots, A_n \rightarrow C, \tag{1}$$

where A_i denotes the i^{th} antecedent and C the class predicted by the rule. Each antecedent corresponds to a threshold condition of the form $f \geq t$ or $f < t$, where f represents a feature, and t a threshold defined by a hyperplane.

At each step of the algorithm, we select as a new antecedent the hyperplane and feature that maximize fidelity with respect to the training set while covering the considered sample. The algorithm terminates once the perfect training fidelity is reached. We also define a dropout mechanism that randomly removes candidate hyperplanes and features at each iteration. Each hyperplane and feature are independently dropped with probability equal to the dropout parameter (e.g., a value of 0.8 results in 80% of candidates being removed on average).

FidexGlo is the global version of this algorithm. It applies Fidex to each training sample to generate an initial set of rules, which is subsequently simplified to obtain a compact set that generalizes over the entire dataset. For a new unseen sample, we can therefore search for an explanation within this ruleset. If no rule matches the sample, a new one can be easily generated using Fidex.

[1] https://github.com/Jean-Marc-B/dimlpfidex_Hepia

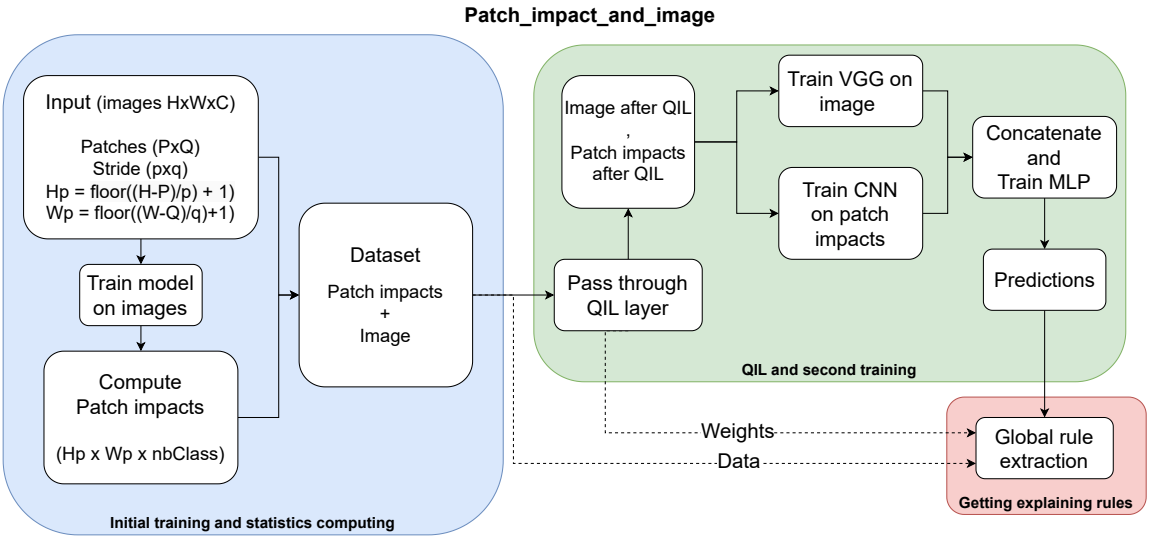

Figure 1: Diagram of our methodology for rule extraction, composed of a two-phase training process and patch impact computation.

2.2. Our Train Methodology for Rule Extraction

We use FidexGlo to extract rules explaining the model's decisions. We propose a rule extraction pipeline that operates identically for both CNNs and ViTs, producing meaningful explanations regardless of the underlying architecture. This methodology is illustrated in Figure 1. The extracted rules identify the regions of the image that contribute to the classification. The rules are composed of two different types of antecedents. The first is the impact of an image region p on the prediction score of a specific class. This impact is computed for the class C as follows:

$$I_{C,p} = P(original\_image) - P(masked\_image)$$

where P denotes the probability obtained for class C, and $masked\_image$ refers to the image in which the pixels in the selected region have been replaced by black pixels, this masking value being chosen arbitrarily. This results in an antecedent of the form: $Impact\_of\_area[15-21] \times [13-19]\_on\_class\_3 < 0.2$, which indicates that the impact of this specific area on class 3 must be lower than 0.2. In general, the absolute value of this score reflects the strength of the patch's impact on the considered class. In contrast, values close to zero indicate a negligible influence. A positive value indicates a positive contribution to the class score, whereas a negative value indicates that the patch opposes this class. In addition to this metric, our methodology produces antecedents directly related to the pixel values of the original image, for example: $Pixel\_6 \times 22 \times 2 >= 0.52$. This means that the value of the blue channel of pixel 6×22 has to be greater or equal to 0.52.

We now provide a more detailed analysis of the pipeline used to generate these rules. We first train a model on our images, which can be either a CNN or a ViT. Then, we define patches according to their desired size $P \times Q$, and to the defined stride $p \times q$. The stride defines the number of pixels by which the sliding window is shifted horizontally and vertically when generating patches. In the diagram shown in Figure 1, Hp denotes the number of patches defined horizontally for an image of size $H \times W \times C$, and Wp denotes the number of patches defined vertically. For each patch, we compute its impact on each class. These scores are gathered to form a dataset, to which we concatenate the image itself. This dataset is then passed through the QIL layer, after which a second training phase is performed. The image is processed by a VGG network, while the patch impacts are handled by a custom CNN. The two outputs are then concatenated and fed into a final Multi Layer Perceptron (MLP). Finally, the predictions of the second training and the weights obtained from the QIL layer are fed into FidexGlo to extract the rules.

3. Experiments

3.1. Datasets and Models

We apply our method to two different image datasets:

- MNIST : 60000 training samples and 10000 testing samples of handwritten digit images of size 28×28 pixels, resulting in 784 features and ten classes.
- CIFAR-10 : 50000 training samples and 10000 testing samples of 32×32 RGB natural images belonging to ten object categories, corresponding to 3072 features.

We trained two different models on each dataset. The first uses a VGG-16 backbone pre-trained on ImageNet-1k. The images are resized to 224×224 pixels before being fed into the model. A task-specific classification head is added, consisting of a flattening operation followed by a fully connected layer with 256 units, dropout (p = 0.3), batch normalization, and a final softmax layer.

The second model is a Vision Transformer (ViT-B/16) with a patch size of 16×16 and an input resolution of 384×384 pixels. The model is re-implemented in Keras and initialized with weights pre-trained on ImageNet-1k from the PyTorch timm library using the `vit_base_patch16_384` configuration. The original classification head is replaced with a task-specific output layer.

For both models, the entire network is fully fine-tuned on each target dataset, with no layers frozen during training. Following the pipeline described in Section 2.2, the second training phase relies on a two-branch architecture. The VGG-16 model is supplied with image pixels, whereas a custom CNN is trained using patch impact features comprising three convolutional layers with 64, 128 and 256 filters. Each layer is followed by batch normalisation, LeakyReLU activation and max pooling. This block is followed by a flattening layer and a fully connected layer with 256 units, batch normalization, LeakyReLU activation, and dropout (p = 0.4).

The outputs of both branches are concatenated and passed through an MLP composed of fully connected layers with 256, 128, and 64 units (ReLU activations), followed by a final softmax layer for classification. We use a GPU for training and for the computation of patch impacts, and 35 CPUs for the rules extraction. It takes at most six days to generate the rules.

3.2. Metrics

In the following tables, we present the results of our experiments. The reported parameters include the patch size and its stride, the dropout rate, the prediction accuracy of the second model (test set accuracy), the fidelity, the rule prediction accuracy (with respect to the ground-truth), the number of extracted rules, the average number of antecedents per rule, and the average rule coverage. This last metric measures the mean number of training samples covered by each rule. Since the rules produce predictions, we can measure their fidelity, and FidexGlo ensures a training fidelity of 100% by construction.

For each dataset, we evaluated different dropout values as well as several patch sizes and stride configurations. These parameters represent a trade-off between rule quality and computational cost. A high dropout rate reduces the number of features and hyperplanes analyzed at each iteration, significantly decreasing computation time, but potentially degrading rule quality by limiting the amount of information exploited. Larger patch sizes result in broader and therefore less precise explanatory regions, while reducing the number of areas to process, which accelerates the computation of statistics and rule generation. Finally, increasing the stride decreases the number of analyzed regions, simplifying the process at the risk of discarding potentially informative areas.

3.3. The MNIST Handwritten Digit Dataset

The first dataset analyzed is MNIST. The results obtained on with the VGG-16 training when training VGG-16 as the first model are reported in Table 1. The best training configuration is achieved using patches of size 5×5 with a stride of 1, reaching 99.59% accuracy on the test set. The rules generated

Table 1
Rule extraction results on MNIST with varying patch sizes, strides, and dropouts. The first training uses VGG16.

Patch size(stride)	Dropout	Pred. Acc (%)	Fidelity (%)	Rule Acc (%)	Nb. Rules	Avg. Nb. Ant.	Avg. Cov.
$7 \times 7(1)$	0.95	99.53	99.87	99.53	**212**	4.89	**1631.1**
$7 \times 7(1)$	0.80	99.53	99.90	99.55	270	**3.19**	1443.1
$5 \times 5(1)$	0.95	**99.59**	99.93	99.61	407	5.32	884.2
$5 \times 5(1)$	0.80	**99.59**	**99.94**	**99.62**	471	3.41	803.0
$4 \times 4(2)$	0.95	99.58	99.84	99.44	1011	6.21	345.6

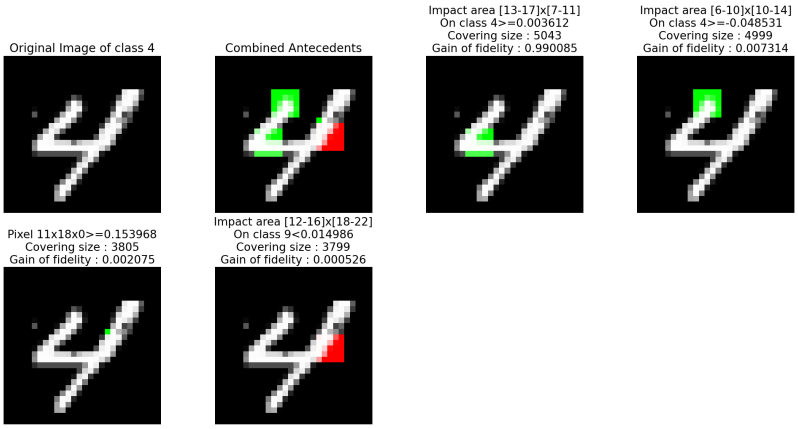

Figure 2: Representation of a rule composed of four antecedents explaining a VGG-16 model on a MNIST sample of class 4. Each antecedent is specified with the rule's coverage and fidelity gain when added.

with a dropout rate of 0.8 instead of 0.95 are slightly better. This configuration also achieves the best fidelity and rule accuracy. The latter is even higher than the score obtained by the model. The set of 407 extracted rules therefore constitutes a classifier that slightly outperforms the original model on the test set. The lowest number of rules and highest average coverage are obtained with patches of size 7×7. As a result, these rules generalize the training set better. By decreasing the dropout rate, we observe a lower average number of antecedents per rule, making the rules simpler. We also observe that excessively reducing patch size and increasing the stride negatively impacts rule quality.

We illustrate in Figure 2 a rule explaining a sample of class *4*. It contains three antecedents highlighting the most important regions for classifying the digit 4, and a fourth involving a pixel value. These regions are located along the upper edges of the digit. Two of them involve a positive impact on class *4*, indicating that these regions are essential for correctly classifying a *4*. The last antecedent ensures that class *9* does not have too strong an influence on the right side. We observe that a 4 can look like a 9 if the top of the digit is closed. It is therefore important to eliminate this possibility to ensure that the digit is classified as a *4* rather than a *9*. The first antecedent alone already achieves 99% fidelity. The remaining 1% is obtained by adding the other antecedents. In summary, this rule states that if two specific regions have a positive impact on class *4*, another region has a rather negative impact on class *9*, and pixel (11,18) has a value greater than 0.15, then the image is classified as class *4*. This rule is global as it covers 3,799 samples, and achieves 100% fidelity and 100% accuracy on the training set. On the test set, it reaches 99.85% fidelity and covers 664 samples. Its accuracy remains at 100%, suggesting that it may generalize better than the model. Note that the confidence is 100% on the training set and 99.85% on the test set. It represents the mean output prediction score of covered samples.

3.4. The CIFAR-10 Color Image Dataset

The results obtained on the CIFAR-10 dataset when training VGG-16 as the first model are presented in Table 2. We observe that the best configuration consists of using 7×7 patches with a stride of 1 and a dropout rate of 0.85. This setting achieves 91.77% accuracy on the test set, a fidelity of 98.62%, and a rule

Table 2
Rule extraction results on CIFAR with varying patch sizes, strides, and dropouts. The first training uses VGG16.

Patch size(stride)	Dropout	Pred. Acc (%)	Fidelity (%)	Rule Acc (%)	Nb. Rules	Avg. Nb. Ant.	Avg. Cov.
$7 \times 7(1)$	0.95	91.77	98.13	91.18	3245	5.76	129.8
$7 \times 7(1)$	0.85	**91.77**	**98.62**	**91.71**	**2757**	**4.47**	**148.1**
$5 \times 5(1)$	0.95	89.81	96.49	88.31	7412	5.81	47.5
$4 \times 4(2)$	0.95	89.98	93.49	85.51	10741	6.15	28.9

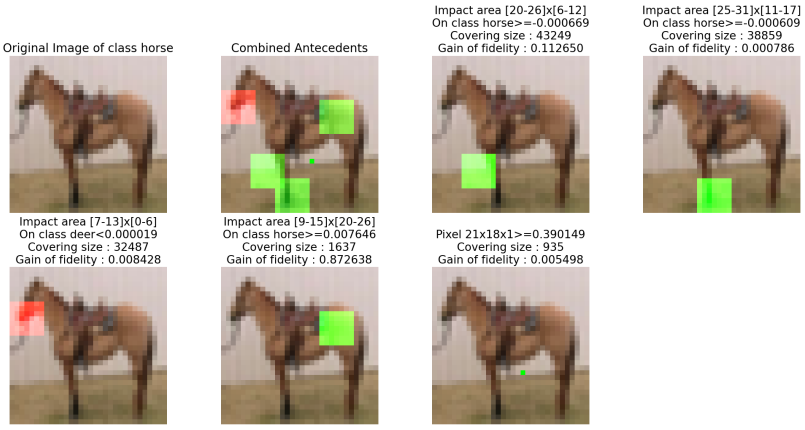

Figure 3: Representation of a rule composed of five antecedents explaining a VGG-16 model on a CIFAR-10 sample of class horse. Each antecedent is specified with the rule's coverage and fidelity gain when added.

Table 3
Rule extraction results on CIFAR with different patch sizes, strides, and dropouts. The first training uses ViT-timm.

Patch size(stride)	Dropout	Pred. Acc (%)	Fidelity (%)	Rule Acc (%)	Nb. Rules	Avg. Nb. Ant.	Avg. Cov.
$7 \times 7(2)$	0.95	**96.96**	83.74	81.77	17574	6.44	10.5
$7 \times 7(2)$	0.70	**96.96**	**90.79**	**89.04**	**17190**	**4.46**	**13.8**
$4 \times 4(1)$	0.95	96.58	82.95	80.90	18348	6.46	9.3

accuracy almost as high as the model accuracy, reaching 91.71%. The average coverage is significantly lower compared to MNIST, as CIFAR-10 is a problem more complex and harder to generalize, which also explains the much larger number of extracted rules and the slightly higher number of antecedents. We further observe the benefit of reducing the dropout rate, even though this slows down the rule generation process.

Figure 3 illustrates a rule explaining a sample from the class *horse*. The important areas for classification are located on the face, the back and the front legs of the animal. The rule requires a positive impact on class *horse*, and a negative impact on class *deer*, which is visually similar to horse. The fidelity and accuracy of this rule are 100% and the confidence is 99% on 935 covered training samples. On the test set, the fidelity is 97.99%, the accuracy is 97.32%, and the confidence is 97.25% for 149 samples.

3.5. Rule Extraction on a Vision Transformer

The results of the rule extraction when training the ViT timm as the first model on CIFAR-10 are presented in Table 3. Compared to VGG-16, we observe a clear improvement in the model's prediction accuracy, increasing from 91.77% to 96.96%. However, rule quality and fidelity decrease significantly. The average coverage drops substantially, while the number of extracted rules increases considerably. This is likely due to the higher complexity of the ViT compared to VGG-16. Nevertheless, we still obtain interesting results and meaningful explaining rules for a ViT. The best scores are again obtained with 7×7 patches with lower dropout.

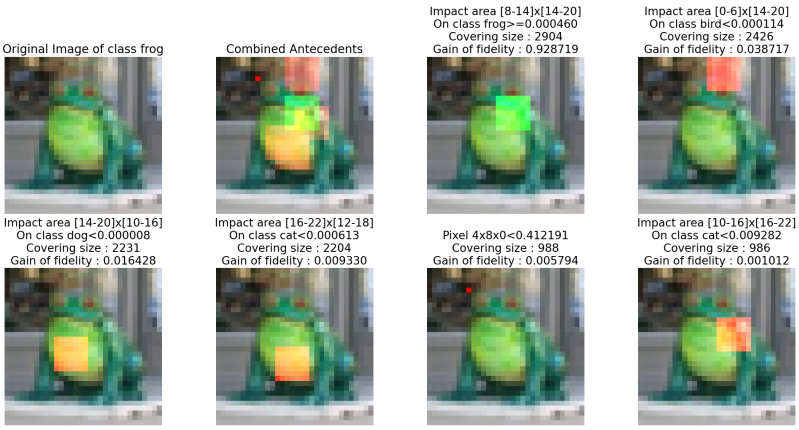

Figure 4: Representation of a rule composed of five antecedents explaining a ViT model on a CIFAR-10 sample of class frog. Each antecedent is specified with the rule's coverage and fidelity gain when added.

Figure 4 illustrates a rule from class *frog*. The rules highlight the most important regions for classification with the ViT. It is interesting to see that for the *frog*, the model needs to ensure a positive impact of the *frog* class while enforcing a negative impact on classes *bird*, *dog* and *cat*, which are also small animals and may share similar backgrounds.

3.6. Comparison with Our Previous Constructive Approach

Table 4
Rule extraction results on MNIST and CIFAR with the previous constructive method (0.95 dropout, VGG-16), compared to the best results of the destructive approach. Values are reported as constructive / destructive.

Dataset	Pred. Acc (%)	Fidelity (%)	Rule Acc (%)	Nb. Rules	Avg. Nb. Ant.	Avg. Cov.
MNIST	**99.61** / 99.59	99.05 / **99.94**	98.82 / **99.62**	1825 / **212**	**3.18** / 3.19	473.05 / **1631.1**
CIFAR-10	**92.65** / 91.77	76.55 / **98.62**	72.79 / **91.71**	22233 / **2757**	**3.98** / 4.47	6.22 / **148.1**

We now compare our new destructive mask-based approach with our previous constructive method introduced in [8]. The results are presented in Table 4. The prediction accuracy of the new method decreases for both datasets. Aside from this, both fidelity and rule accuracy improve significantly, the most striking example being the fidelity on CIFAR-10, which increases from 76.55% to 98.62%. The number of extracted rules decreases substantially, with a nearly 90% reduction for CIFAR-10. This represents a considerable simplification of the rule set. The type of explanations produced by the rules differs. While the constructive approach produces explanations based on the probability scores of a class over a given region, the destructive approach generates explanations based on the impact scores of a region on a class's classification. In both cases, the rules identify the critical regions involved in the image classification process.

4. Discussion and Conclusion

We presented a new method for extracting rules that explain the image classification decisions of both CNNs and Vision Transformers (ViTs). The method consists of a first training phase used to obtain the impact of each image region on the model's classification. These impacts, together with the image, are then passed through a QIL layer and a second model, after which rules are generated using FidexGlo. We apply our method to two different datasets, using both a VGG-16 and a ViT as the initial trained model. The obtained results show strong performance in terms of fidelity and rule accuracy, and yield meaningful rules that are themselves capable of classifying new samples. To the best of our knowledge, no other algorithm is capable of extracting global rules composed of both patch-level and pixel-level

information for CNNs and ViTs. We plan to evaluate this methodology on additional datasets. We also aim to investigate alternative methodologies focused on explaining ViTs in order to further improve fidelity for this type of model. Finally, we intend to replace pixel-level information with patch-based statistics within the methodology, with the objective of accelerating the overall process.

Acknowledgments

This work was conducted in the context of the Horizon Europe project PRE-ACT (Prediction of Radiotherapy side effects using explainable AI for patient communication and treatment modification). It was supported by the European Commission through the Horizon Europe Program (Grant Agreement number 101057746), by the Swiss State Secretariat or Education, Research and Innovation (SERI) under contract number 2200058, and by the UK government (Innovate UK application number 10061955).

Declaration on Generative AI

During the preparation of this work, the authors used ChatGPT, Grammarly in order to: Grammar and spelling check, Paraphrase and reword, and Text Translation. Using these tools, the authors reviewed and edited the content as needed and assume full responsibility for the content of the publication.

References

[1] H. Chen, S. Lundberg, S.-I. Lee, Explaining models by propagating shapley values of local components, Explainable AI in Healthcare and Medicine: Building a Culture of Transparency and Accountability (2021) 261–270. doi:10.1007/978-3-030-53352-6_24.

[2] M. T. Ribeiro, S. Singh, C. Guestrin, "why should i trust you?" explaining the predictions of any classifier, in: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, 2016, pp. 1135–1144. doi:10.1145/2939672.2939778.

[3] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, D. Batra, Grad-cam: Visual explanations from deep networks via gradient-based localization, in: Proceedings of the IEEE international conference on computer vision, 2017, pp. 618–626. doi:10.1109/ICCV.2017.74.

[4] F. Sabbatini, Four decades of symbolic knowledge extraction from sub-symbolic predictors. a survey, ACM Computing Surveys 58 (2025) 1–36. doi:10.1145/37490.

[5] J. Townsend, T. Kasioumis, H. Inakoshi, Eric: Extracting relations inferred from convolutions, in: Proceedings of the Asian Conference on Computer Vision, Springer, Cham, 2020, pp. 206–222. doi:10.1007/978-3-030-69535-4_13.

[6] P. Padalkar, H. Wang, G. Gupta, Nesyfold: a framework for interpretable image classification, in: Proceedings of the AAAI Conference On Artificial Intelligence, volume 38, 2024, pp. 4378–4387. doi:10.1609/aaai.v38i5.2823.

[7] G. Bologna, J.-M. Boutay, D. Boquete, Q. Leblanc, D. Köprülü, L. Pfeiffer, Fidex and fidexglo: From local explanations to global explanations of deep models, Algorithms 18 (2025). URL: https://www.mdpi.com/1999-4893/18/3/120. doi:10.3390/a18030120.

[8] J. Boutay, Q. Leblanc, B. D. Boquete, D. Köprülü, L. Pfeiffer, G. Bologna, Explaining CNN classifications using small patches, in: Proceedings of the 1st International Workshop on Advanced Neuro-Symbolic Applications (ANSyA), Co-located with the European Conference on Artificial Intelligence (ECAI) 2025, Bologna, Italy, 2025, p. Paper 9. URL: https://ansya-workshop.github.io/proceedings-2025/paper_9.pdf.

[9] G. Bologna, C. Pellegrini, Constraining the mlp power of expression to facilitate symbolic rule extraction, in: 1998 IEEE International Joint Conference on Neural Networks Proceedings. IEEE World Congress on Computational Intelligence (Cat. No. 98CH36227), volume 1, IEEE, 1998, pp. 146–151. doi:10.1109/IJCNN.1998.682252.