INTERPRETABILITY OF DEEP LEARNING MODELS FOR VISUAL DEFECT DETECTION: A PRELIMINARY STUDY

Mehdi Elion, Sonia Tabti, Julien Budynek
FieldBox.ai, Bordeaux, France
{melion, stabti, jbudynek}@fieldbox.ai

ABSTRACT

How relevant are interpretability methods designed for deep learning models in the context of visual defect detection? Beyond their actual output, to what extent can these methods be used in a production environment? We study and evaluate interpretability methods for convolutional neural networks (CNN) and vision transformers (ViT) on image classification datasets designed for defect detection.

1 Introduction

Computer vision models based on deep learning have been increasingly successful in numerous fields, including industrial applications such as quality control and visual inspection. However, such models suffer from a lack of trust from end users as they are often considered as "black boxes" with not enough insights regarding their decision process. Hence, the need for model interpretability methods has concomitantly grown, not only to facilitate user trust and adoption, but also because such techniques can help detect model biases, validate the relevance of the decision processes underlying deep learning models and therefore make a first step towards AI certification.

1.1 Brief literature search on interpretability methods for computer vision deep learning models

To meet the need for model interpretability, several methods have been developed to provide users with visual insights pertaining to model predictions. Perturbation-based methods such as LIME [1], SHAP [2] or occlusion sensitivity [3] are model-agnostic and provide interesting insights, but they usually are computationally expensive, and perturbed data can be outside of the training distribution. On the other end, methods based on gradient back-propagation such as Integrated Gradients [4] or GradCAM (class activation maps) [5] are widely used and usually work well with CNNs. However, such methods are not necessarily adapted to alternate architectures such as Vision Transformers. Some methods, eg: attention-rollout [6], exploit the attention mechanism in ViT architectures in order to visualize which features they can extract from input images. However this method is not class-dependent, making it difficult to use for defect detection. Several methods have thus been developed specifically to better interpret ViT predictions [7], for instance gradient-attention-rollout, which is a class-dependant improvement of attention-rollout, or layer-wise relevance propagation (LRP), based on Deep Taylor Decomposition [8], which has proved to be a suitable choice for ViTs.

1.2 Problem Statement

Live defect detection applied to pictures of products taken at regular time intervals on a production line is a common industrial use case. In this study, we approach it through an image classification task. More precisely, we will focus on two formulations for this problem. Binary classification on the one hand, to detect whether or not there is a defect on an image of a product. Multi-class classification on the other hand, to categorize among several defect types which one appears on a product image. In this study, CNN and ViT classifiers will be compared in terms of classification metrics, computation efficiency and effectiveness of some of the most suitable interpretability frameworks for them.

2 Approach and implementation

In this section, we describe the above-mentioned deep learning models and their respective interpretability methods selected for this preliminary study. Then, we provide training and implementation details.

2.1 Models and interpretability methods description

The selected deep image classification networks for this study are VGG16 [9] (pretrained on ImageNet) for CNN classifiers, and ViT-small16 [10] (pretrained on ImageNet using DINO [11]) for ViT-based ones. The VGG16 architecture provides a good balance in terms of classification performance, computation efficiency and adaptability to many interpretability methods. In this study, the interpretabilty frameworks compared for the CNN model are Occlusion sensitivity and GradCAM. Occlusion sensitivity is a perturbation-based method that is model agnostic. It occludes iteratively image regions to assess how the CNN's confidence is affected. GradCAM is a gradient-based method. It uses the feature maps produced by the last convolutional layer to understand which regions of an image were relevant to the CNN.

Vision Transformers is the second type of classification model selected for this study as it has shown impressive results in multiple computer vision applications with high robustness to various types of perturbations (eg: image occlusion, domain shift) [12], which is valuable in an industrial context. The simplest interpretability method to exploit the ViTs' multi-head self-attention mechanism is the attention-rollout method which combines the attention maps from all the heads. As a result, this method is not class-specific, which can be an issue when one wants to inspect an interpretability map for a specific class to understand a model's decisions and errors. More formally, the attention-rollout boils down to the following equation:

$$\hat{\mathbf{A}}^{(b)} = I + \mathbb{E}_h \mathbf{A}^{(b)}, \quad \text{rollout} = \hat{\mathbf{A}}^{(1)} \cdot \hat{\mathbf{A}}^{(2)} \cdot ... \cdot \hat{\mathbf{A}}^{(B)} \tag{1}$$

where $\mathbf{A}^{(b)}$ is the attention map, $b = \{1, ..., B\}$ the transformer block index, $\mathbb{E}_h$ the mean across "heads" dimension and $(\cdot)$ the matrix multiplication.

The layer-wise relevance propagation (LRP) [7] provides class-specific maps. It computes relevance scores based on the Deep Taylor Decomposition principle for each attention head in each layer of a Transformer model. Then, it back-propagates these relevancy scores through the layers. In short, the output of the method, noted $\mathbf{C}$, is a combination of weighted attention relevance and is computed as follows:

$$\bar{\mathbf{A}}^{(b)} = I + \mathbb{E}_h(\nabla \mathbf{A}^{(b)} \odot R^{(n_b)})^+, \quad \mathbf{C} = \bar{\mathbf{A}}^{(1)} \cdot \bar{\mathbf{A}}^{(2)} \cdot ... \cdot \bar{\mathbf{A}}^{(B)} \tag{2}$$

where $\nabla \mathbf{A}^{(b)}$ is the gradient of the attention map, $R^{(n_b)}$ is the layer's relevance with respect to a target class (see [7] for more details), $\odot$ is the Hadamard product and $(.)^+$ denotes the positive part function.

2.2 Training procedure and implementation details

In order to obtain classifiers that are specifically trained on our target tasks, we use transfer learning and fine-tuning as it allows us to reach good results while using a reasonable amount of resources and time. More precisely, the transfer learning phase consists in removing the classifier head that was specific to the source task, replacing it with a new one that is specific to our target task and training it while leaving original backbone weights frozen. Then, during the fine-tuning phase, backbone weights are partially or totally unfrozen before launching a second training phase. Models are trained using gradient descent with Cross Entropy as a training criterion.

3 Experimental study

3.1 Datasets

The models and the interpretability methods implemented are evaluated on two image classification datasets. The first one is the casting defect dataset available on Kaggle[13], which contains a total of 7348 images of size 300×300, labeled as showing a defective (*def-front*) or non-defective (*ok-front*) product. It is divided into training and test sets. The training set contains 3758 defective images and 2875 non-defective images, while the test set contains 453 defective images and 262 non-defective images. The second one is the NEU-DET dataset [14], which contains 1800 images showing six types of surface defects of a hot-rolled steel strip, which are Crazing (Cr), Inclusion (In), Patches (Pa), Pitted Surface (PS), Rolled-in Scale (RS), and Scratches (Sc). Each type of sample has 300 grayscale images of size 200×200. Note that, for each dataset, images were resized to 224×224 before being used as model inputs, and we also keep 20% of the training set for validation in order to avoid overfitting.

3.2 Results

First, classification metrics listed on table 1 show that combining transfer learning and fine-tuning to train a classifier yields successful results, especially on such clean datasets where defects are easily identified. It should be noted that ViTs perform slightly better than VGG16 for most metrics and both datasets. Note that, the training procedure being very successful on NEU-DET, both models end-up misclassifying the same unique test image, hence the identical metrics.

	CNN			ViT		
	Accuracy	Precision	Recall	Accuracy	Precision	Recall
Casting defect	98.74	99.01	99.55	99.58	99.66	99.33
NEU_DET	99.72	99.73	99.72	99.72	99.73	99.72

Table 1: Classification metrics obtained on test sets. For the casting defect dataset, metrics were computed by considering defective products as positive cases. For NEU-DET, shown metrics were averaged over all classes.

Second, table 2 shows different relevant time measurements for each deep learning classifier and each interpretability method, namely: inference time and time necessary to generate interpretability heatmaps for one image. The hardware used is an NVIDIA GeForce RTX 2080 Ti GPU with 11 Go of RAM. It is clear that generating those heatmaps is computationally more intensive than a simple inference, by a factor of about three to ten, indifferently on both datasets. For the CNN classifier, GradCAM is twice as fast as occlusion sensitivity. For the ViT classifier, rollout and LRP have roughly the same performance. And, inference with ViT is twice as fast as inference with CNN.

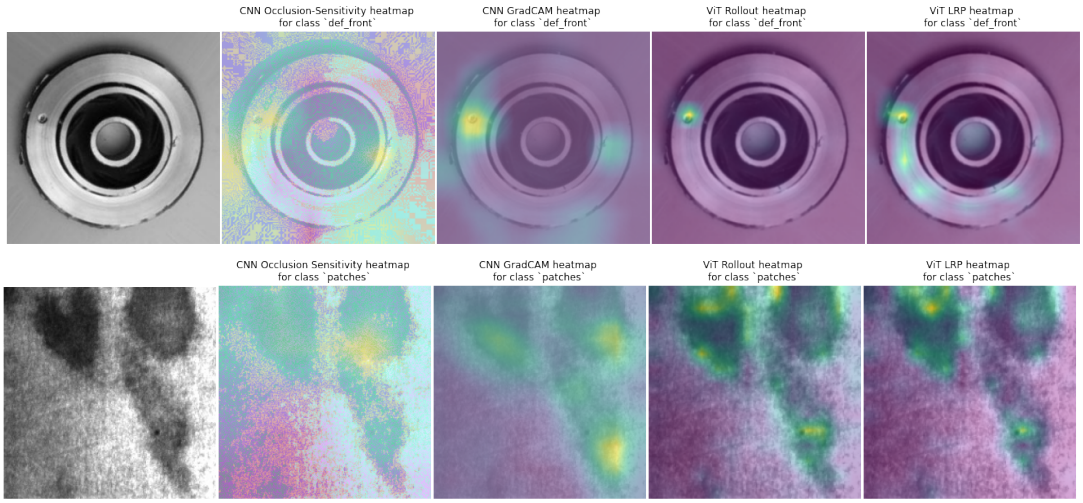

Figure 1: Example interpretability maps computed for defective products from the casting defect dataset (top line) and from the NEU-DET dataset (bottom line). The first image is the raw picture, then the columns show heatmaps for CNN Occlusion, CNN GradCAM, ViT attention-rollout and ViT LRP.

Finally, based on some visual examples, we evaluate the quality of interpretability maps output by the models. Figure 1 shows such examples on correctly classified samples. For class-specific methods, the interpretability map of the predicted class is shown. One can observe that occlusion sensitivity provides less accurate results than other methods. Indeed GradCAM, attention rollout and LRP heatmaps tend to highlight tighter areas in the image. However, we note that, on the casting defect example, GradCAM seems more "exhaustive" in terms of highlighted class-related areas, while LRP seems more selective than GradCAM. Attention rollout, on the other hand, by design, doesn't highlight class-related areas but salient elements in the image, which, in that case, can be defects themselves but in other cases can be harmful to model interpretability.

We also see the benefit of using interpretability to better understand classification errors. For instance, figure 2 shows an example on the casting dataset where the classifier seems to wrongly consider a product as defective because of particular lighting conditions. An example is shown as well on the NEU-DET dataset where an inclusion defect is confused with a scratch. It is understandable since some scratches are similar to inclusions.

4 Conclusion

To conclude this preliminary study, combining the measurements in tables 1 and 2 with the qualitative analysis of the interpretability maps, we recommend practitioners the use of GradCAM over occlusion sensitivity for CNNs and LRP over attention-rollout for ViTs. However, if the computational efficiency required once a defect detection solution is deployed on a production line is strong, one might consider using ViTs and an interpretability method as an offline tool

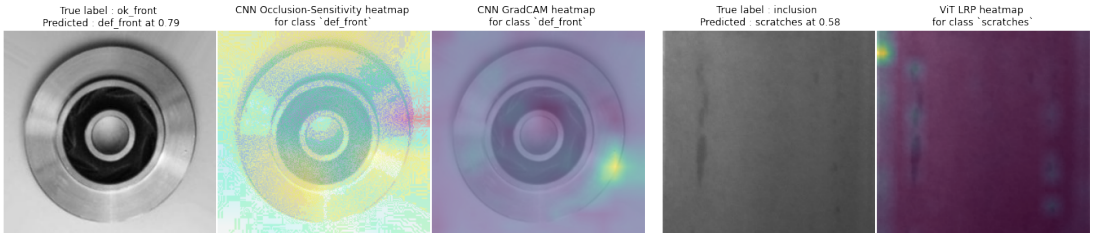

Figure 2: Example interpretability maps computed on misclassified images, first from the casting defect dataset (left side: raw picture, then heatmaps for CNN Occlusion and CNN GradCAM), and from the NEU-DET dataset (right side: raw picture, then heatmap from ViT LRP).

	CNN			ViT		
	Forward-pass	Occlusion	GradCAM	Inference	Rollout	LRP
Casting defect	51.0 ± 1.6	160.0 ± 9.4	89.2 ± 4.7	24.7 ± 8.2	291.2 ± 27.3	266.4 ± 26.1
NEU_DET	58.2 ± 13.0	181.0 ± 44.0	92.3 ± 7.8	25.6 ± 5.0	249.2 ± 23.6	246.3 ± 17.2

Table 2: Computation times (in ms). Provided numbers correspond to mean and standard deviation of elapsed time during single-image operations: forward-pass or computation of intepretability maps.

to monitor the model's predictions behavior. Future works will include a deeper comparative study. For instance, other sizes of ViTs could be used and other interpretability methods tested. Also, a quantitative analysis of the relevance of interpretability maps using datasets with segmentation maps could be done.

References

[1] M.T Ribeiro, S. Singh, and 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*, pages 1135–1144, 2016.

[2] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30, 2017.

[3] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.

[4] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks, 2017.

[5] 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. *International Journal of Computer Vision*, 128(2):336–359, oct 2019.

[6] Samira Abnar and Willem Zuidema. Quantifying attention flow in transformers. *arXiv preprint arXiv:2005.00928*, 2020.

[7] Hila Chefer, Shir Gur, and Lior Wolf. Transformer interpretability beyond attention visualization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 782–791, June 2021.

[8] Grégoire Montavon, Sebastian Lapuschkin, Alexander Binder, Wojciech Samek, and Klaus-Robert Müller. Explaining nonlinear classification decisions with deep Taylor decomposition. *Pattern Recognition*, 65:211–222, May 2017.

[9] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[10] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[11] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers, 2021.

[12] Muhammad M Naseer, Kanchana Ranasinghe, Salman H Khan, Munawar Hayat, Fahad Shahbaz Khan, and Ming-Hsuan Yang. Intriguing properties of vision transformers. *Advances in Neural Information Processing Systems*, 34:23296–23308, 2021.

[13] Ravirajsinh Dabhi. Casting product image data for quality inspection, 2020.

[14] Kechen Song and Yunhui Yan. A noise robust method based on completed local binary patterns for hot-rolled steel strip surface defects. *Applied Surface Science*, 285:858–864, November 2013.