Interactive xAI-dashboard for Semantic Segmentation

Finn Schürmann[1] and Sibylle D. Sager-Müller[1,*]

[1] *Lucerne University of Applied Sciences and Arts, Suurstoffi 1, CH-6343 Rotkreuz, Switzerland*

Abstract

This article proposes an interactive dashboard for analyzing semantic image segmentation models using eXplainable AI (xAI) methods. It integrates open-source xAI packages with segmentation models from PyTorch and TensorFlow Keras, focusing on road traffic images. Through model-based and post hoc explanation methods, users gain insights into model perceptions. The dashboard facilitates user interaction by allowing selection of model, label, and xAI method, with visualizations displaying segmented images and explanations. The implementation uses Python's Dash library, complemented by PyTorch and external xAI libraries. A demo app showcases model comparisons and xAI method outputs, enhancing transparency and trust in AI systems for safety-critical applications like autonomous driving.

Keywords

Computer vision, Semantic segmentation, explainable AI, Human-Machine interaction

1. Introduction

xAI is a set of methods and procedures which help humans to understand and trust results created by artificial intelligence (AI) algorithms. Especially in safety-critical applications, xAI is a key requirement [1]. This can be achieved through various validation algorithms. To obtain an unbiased and comprehensive understanding of the outcomes produced by existing algorithms for neural networks (NNs) at different depths, this article proposes a novel interactive dashboard containing a subset of common xAI methods of the most prominent open source xAI packages. These packages can be used to analyze data and segmentation models, e.g., from road traffic. The dashboard also allows for a selection of data and trained segmentation models from common libraries like PyTorch [2].

In autonomous driving the perception of environment is an important aspect. Image segmentation, a frequently used technique in this application, assigns a label to every pixel in an image so that pixels with the same label share certain characteristics. It is important that these segmentation models have a high accuracy and efficiency as they are included in

Late-breaking work, Demos and Doctoral Consortium, colocated with The 2nd World Conference on eXplainable Artificial Intelligence: July 17–19, 2024, Valletta, Malta

*Corresponding author

finn.schuermann@kasel.ch (F. Schürmann), sibylle.sager@hslu.ch (S. Sager-Müller)

https://orcid.org/0000-0003-2375-5601 (F. Schürmann), https://orcid.org/0009-0000-4857-5514 (S. Sager-Müller)

driver assistance systems. xAI enables the user to analyze the performance of the segmentation models. Thus, it helps the AI developer – in the example of machine vision for autonomous vehicles – to check if the model evaluates the situation shown in the image correctly. However, the segmentation models and xAI methods are not yet optimised for autonomous driving. While they can assist in the resolution of issues, their development in this field is not yet sufficiently advanced.

From 2015 to 2021, more than 150 xAI related tools have been published [3]. In general the tools are implemented for image classification, because the xAI-methods were initially developed for image classification [4]. For both machine vision tasks, image classification and semantic segmentation, heatmaps can be employed to help the user to find out if the model learned what it was expected to learn. During the analysis of the widely used xAI-tools, we did not come across a specific tool dedicated solely to image segmentation. The Neuroscope framework [4] has been one approach towards xAI analysis for image segmentation (plus image classification). However, it is important to note that its implementation is platform-dependent and there are currently no plans for further development [5]. Therefore, we intend to fill the gap by implementing a novel platform-independent dashboard for image segmentation specifically. Our goal is to insert the most common segmentation models and xAI-methods in a dashboard with user-friendly interaction possibility.

2. Models and xAI-Methods

2.1. Models

The right choice of model for computer vision tasks is crucial. The library from PyTorch specifies which model is preferable for which task. Table 1 shows an overview of the models for image segmentation used in the dashboard. The models can be categorised in fully convolutional networks (FCN) and DeepLabV3 networks.

Table 1
Overview of used models for semantic image segmentation in PyTorch

task	model	specification
Semantic segmentation	FCN	FCN_ResNet_50
		FCN_ResNet_101
	DeepLabV3	DeepLabV3_MobileNet_V3_Large
		DeepLabV3_ResNet_50
		DeepLabV3_ResNet_101

The first model class comprises FCN ResNet models which take inputs of different sizes and produce outputs of corresponding sizes. Segmentation networks are based on

classification networks with little adaption. ResNet is a deep convolutional neural network proposed by Microsoft. With residual blocks that help optimise a residual function, this architecture allows accuracy to be increased by increasing the depths of layers [6]. The number "50", for example in FCN ResNet 50, represents the number of layers in the network. All models in Table 1 are pretrained with the PASCAL VOC dataset.

The second model class is the DeepLabV3 model, which is based on the Resnet 50, Resnet 101 or MobileNet3 backbone. The difference is that DeepLabV3 uses atrous convolution, also called dilated convolution. The models based on atrous convolution are actively researched for in semantic segmentation [7]. DeepLabV3 uses the MobileNetV3-Large model, which is 34% faster than its predecessor at the same accuracy level for cityscape segmentation. The reason for the speed increase of mobile models is two-fold: First, MobileNetV3 uses the hard sigmoid function instead of the standard sigmoid function because the hard sigmoid function has much lower latency costs [8]. Second, mobile models employ atrous convolution meaning that the kernel laid over the input has some holes. The size of the holes can be controlled by the hyperparameter *rate*. The default convolution sets the rate to 1. The more the rate increases, the more it is possible to encode the object with multi-scale context [7].

2.2. xAI-Methods

Our dashboard in its current implementation uses Layer GradCAM, LIME, Feature Ablation, and Saliency as xAI-methods. xAI-methods can be categorized in model-specific and model-agnostic methods. Model-specific methods calculated the effect of changes in the input features to the output using the model itself, while model-agnostic methods work by manipulating input data and analyzing the respective model predictions without knowledge of the model. Within the subclasses of specific and agnostic, one can further distinguish between local or global methods. Local methods explain the individual predictions of models, while global methods explain the behavior of the model averaged over all samples [9],[10],[11].

Gradient-weighted Class Activation Mapping (GradCAM) [12] is a technique that analyzes gradient information for any convolutional layer of a model and generates a heatmap that highlights important regions in the image. This method operates through forward passes without backpropagation.

Local Interpretable Model Agnostic Explanations (LIME) [13] trains an interpretable surrogate model. The model is evaluated at sampling points around a defined input example to train a simple surrogate model. It is a model-agnostic, perturbation-based approach.

Feature Ablation [9] is perturbation-based and calculates attribution, by replacing each input feature with some reference, and calculating the difference in output. A set of features can be turned off together instead of one at a time.

Saliency [14] calculates the gradients with respect to inputs.

3. Implementation

The goal is to deliver a dashboard that allows the user to interactively check an AI model for semantic segmentation using a variety of xAI methods. This involves finding the trade-off between technical depth on one side and comprehensibility by the ordinary user on the other side. The dashboard is implemented with python using Dash [15]. Dash is a library from Plotly to create web apps without need to write code in JavaScript or HTML. Dash in combination with PyTorch yields good visualization possibilities for segmented models. xAI methods were implemented using the Captum library [9] and the pytorch-grad-cam library from Jacob Gildenblat [16].

4. Function of dashboard

The dashboard can be used to compare xAI methods for different segmentation models. As a first step, a demo app has been implemented with the goal to show the user the main working principles of the dashboard. This will help the user to get accustomed to the tool. The demo app can be opened with the tab “show demo”. In the dropdown menus as shown in Figure 1 the segmentation models for comparison can be chosen. If on both images a model is selected, the difference between the segmented images of the two models gets visible on the bottom left side. This image is obtained by taking the pixelwise difference of the arrays of the two segmented images.

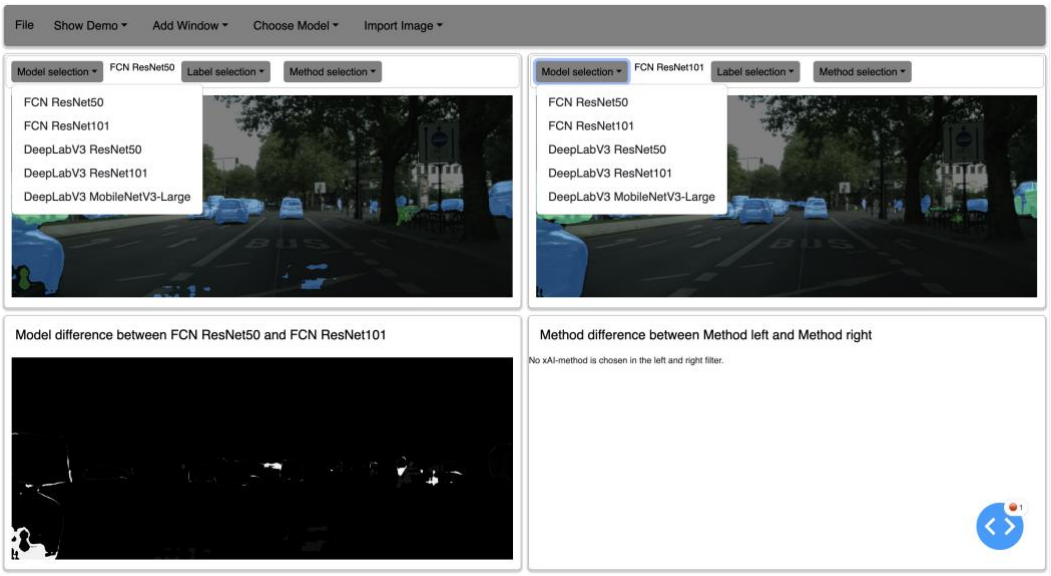

Figure 1: Model selection to compare the segmentation models.

The segmented image is overlaid with the original one so that the user can easily see the quality of the segmentation. By showing the differences, the model quality can be easily compared visually. This helps the user to decide for an appropriate model. After selecting

a model, the user chooses a label from the dropdown menu “Label Selection”, as shown in Figure 2. The labels are, in the preliminary version, predefined since the dashboard only allows the selection of pre-trained models from PyTorch.

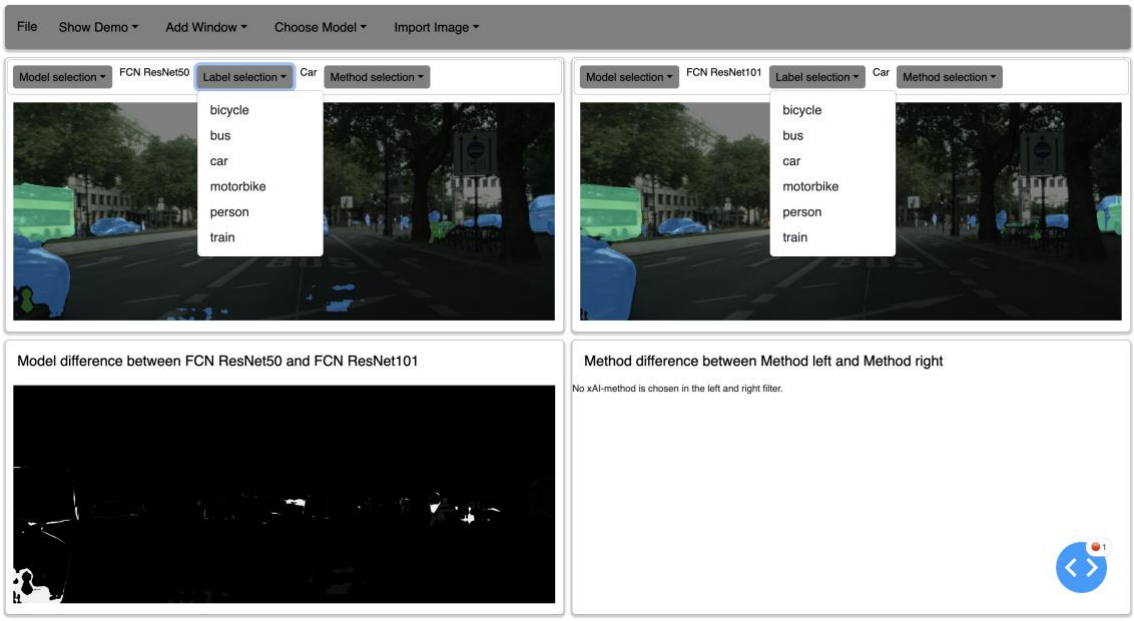

Figure 2: Label Selection.

After the label is selected, up to two xAI methods can be chosen from the dropdown menu, one on the left-hand side, another on the right-hand side. Then, the upper row shows the original image overlaid with the heatmap from the corresponding xAI methods. If two xAI methods are selected, the difference of their corresponding heatmaps is shown in the lower right section to allow for direct visual comparison. On a technical level, the user has to make sure to select the same label for both xAI methods to make the comparison meaningful. The difference of the heatmaps is calculated as soon as the methods are selected in both filter bars. If a method is not selected in one of the filters, the element will indicate which filter needs to have a method chosen, as shown in Figure 3.

This is not the final state of the dashboard: One the lower right section, it is planned to include metrics to evaluate the xAI methods, e.g., from the library Quantus [17]. Currently, Quantus is only applicable to image classification, but with a few modifications, it should be possible to apply it to image segmentation as well. Also, the metrics have to be pre-selected first. For implementation everything is documented on GitHub*[2].

[2] https://github.com/fschurma/xAI_dashboard

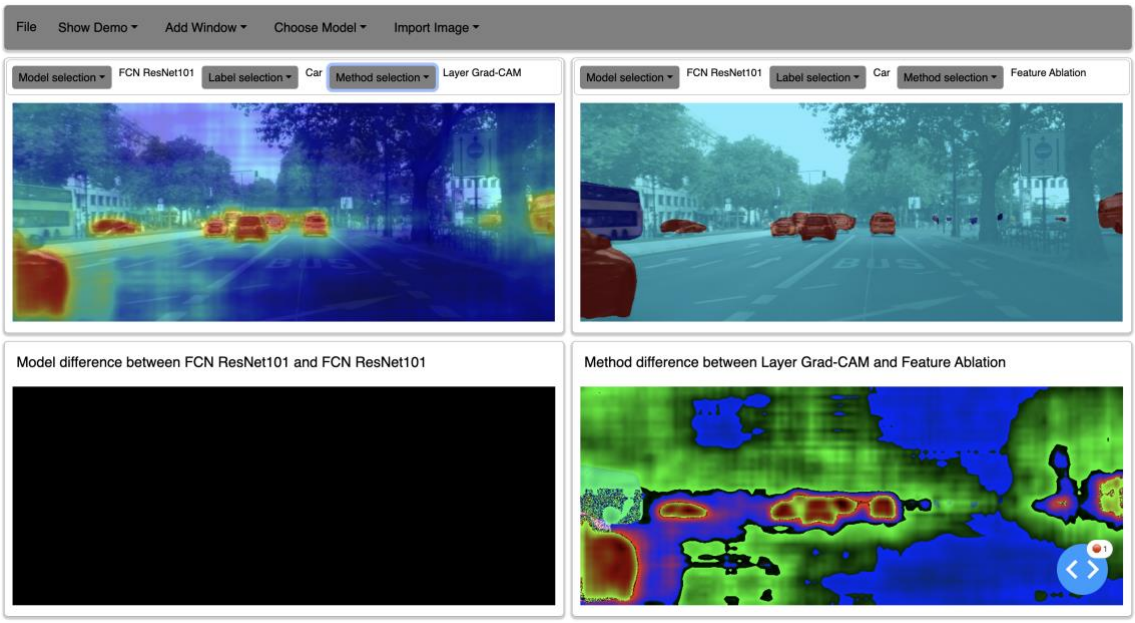

Figure 3: Method selection to compare the xAI methods.

5. Conclusion

The demo app with the functions described above serves as the foundation of the final app currently under construction. The goal of the final version is to allow the users to import their own image(s) and segmentation model(s) to test its/their performance. This enables the user to adjust the models. In the current state, comparison of two models and two methods is possible only visually. It is planned to display evaluation metrics additionally, which would be a benefit compared to similar implementations like Neuroscope [4]. Metrics could be based on those from Quantus [17]. However, this will require first to study which of the metrics can be transferred from classification to segmentation tasks. The xAI methods employed in this demo app are just a small selection which will be extended to a larger subset like, e.g., in Neuroscope. For a better visualization, it is planned to include a color scale to illustrate the magnitude of the image differences. As a last step, the app will be thoroughly tested to make sure it will meet the requirements for user-friendly human-machine interaction.

References

[1] F. Xu, H. Uszkoreit, Y. Du, W. Fan, D. Zhao, and J. Zhu, Explainable AI: A Brief Survey on History, Research Areas, Approaches and Challenges, in *Natural Language Processing and Chinese Computing*, vol. 11839, J. Tang, M.-Y. Kan, D. Zhao, S. Li, and H. Zan, Eds., in Lecture Notes in Computer Science, vol. 11839, Cham: Springer International Publishing, 2019, pp. 563–574. doi: 10.1007/978-3-030-32236-6_51.

[2] Models and pre-trained weights — Torchvision 0.16 documentation'. Accessed: Oct. 24, 2023. [Online]. Available: https://pytorch.org/vision/stable/models.html#semantic-segmentation

[3] N. Uhl, Making It Easier to Compare the Tools for Explainable AI, Partnership on AI. Accessed: Nov. 26, 2023. [Online]. Available: https://partnershiponai.org/making-it-easier-to-compare-the-tools-for-explainable-ai/

[4] C. Schorr, P. Goodarzi, F. Chen, and T. Dahmen, Neuroscope: An Explainable AI Toolbox for Semantic Segmentation and Image Classification of Convolutional Neural Nets, *Appl. Sci.*, vol. 11, no. 5, p. 2199, Mar. 2021, doi: 10.3390/app11052199.

[5] C. Schorr, Personal Communication, 2023.

[6] K. Le, 'A quick overview of ResNet models', MLearning.ai. Accessed: Nov. 07, 2023. [Online]. Available: https://medium.com/mlearning-ai/a-quick-overview-of-resnet-models-f8ed277ae81e

[7] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, Rethinking Atrous Convolution for Semantic Image Segmentation. arXiv, Dec. 05, 2017. Accessed: Nov. 12, 2023. [Online]. Available: http://arxiv.org/abs/1706.05587

[8] A. Howard *et al.*, 'Searching for MobileNetV3'. arXiv, Nov. 20, 2019. Accessed: Nov. 07, 2023. [Online]. Available: http://arxiv.org/abs/1905.02244

[9] Introduction · Captum'. Accessed: Dec. 02, 2023. [Online]. Available: https://captum.ai/

[10] C. Molnar,G. Casalicchi, B. Bischl (2020). Interpretable Machine Learning – A Brief History, State-of-the-Art and Challenges. In: Koprinska, I., *et al.* ECML PKDD 2020 Workshops. ECML PKDD 2020. Communications in Computer and Information Science, vol 1323. Springer, Cham. https://doi.org/10.1007/978-3-030-65965-3_28.

[11] M. Munn and D. Pitman, *Explainable AI for practitioners: designing and implementing explainable ML solutions*. Beijing, Sebastopol, CA: O'Reilly, 2022.

[12] R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, D. Batra, 'Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization', *International Journal Computer Vision*, vol. 128, no. 2, pp. 336–359, Feb. 2020, doi: 10.1007/s11263-019-01228-7.

[13] M. T. Ribeiro, S. Singh, and C. Guestrin, "Why Should I Trust You?": Explaining the Predictions of Any Classifier. arXiv, Feb. 26, 2016. Accessed: May 13, 2024. [Online]. Available: http://arxiv.org/abs/1602.04938.

[14] K. Simonyan, A. Vedaldi, and A. Zisserman, Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps. arXiv, Apr. 19, 2014. Accessed: Mar. 20, 2024. [Online]. Available: http://arxiv.org/abs/1312.6034

[15] 'Dash Documentation & User Guide | Plotly'. Accessed: Apr. 05, 2024. [Online]. Available: https://dash.plotly.com/

[16] J. Gildenblat, 'jacobgil/pytorch-grad-cam'. Apr. 05, 2024. Accessed: Apr. 05, 2024. [Online]. Available: https://github.com/jacobgil/pytorch-grad-cam

[17] A. Hedström *et al.*, ‘Quantus: An Explainable AI Toolkit for Responsible Evaluation of Neural Network Explanations and Beyond’. arXiv, Feb. 14, 2022. Accessed: May 13, 2024. [Online]. Available: http://arxiv.org/abs/2202.06861.