Xaitk-Saliency: An Open Source Explainable AI Toolkit for Saliency

Brian Hu, Paul Tunison, Brandon RichardWebster, Anthony Hoogs

Kitware, Inc.
1712 Route 9, Suite 300
Clifton Park, New York 12065 USA
{brian.hu, paul.tunison, brandon.richardwebster, anthony.hoogs}@kitware.com

Abstract

Advances in artificial intelligence (AI) using techniques such as deep learning have fueled the recent progress in fields such as computer vision. However, these algorithms are still often viewed as "black boxes", which cannot easily explain how they arrived at their final output decisions. Saliency maps are one commonly used form of explainable AI (XAI), which indicate the input features an algorithm paid attention to during its decision process. Here, we introduce the open source *xaitk-saliency* package, an XAI framework and toolkit for saliency. We demonstrate its modular and flexible nature by highlighting two example use cases for saliency maps: (1) object detection model comparison and (2) doppelgänger saliency for person re-identification. We also show how the *xaitk-saliency* package can be paired with visualization tools to support the interactive exploration of saliency maps. Our results suggest that saliency maps may play a critical role in the verification and validation of AI models, ensuring their trusted use and deployment. The code is publicly available at: https://github.com/xaitk/xaitk-saliency.

Introduction

Research in artificial intelligence (AI) has seen significant progress in the past few years, spurring the increasing adoption of AI models in many real-world applications. Despite the success of these AI models, their "black box" nature and lack of interpretability presents a serious barrier to use in domains such as healthcare, criminal justice, and autonomous driving (Holzinger et al. 2017; Doshi-Velez and Kim 2017). The growing field of explainable artificial intelligence (XAI) seeks to develop tools and resources that enable AI models to not only generate results, but also human-understandable explanations of *why* these results were produced (Samek, Wiegand, and Müller 2017; Vilone and Longo 2020). As such, XAI has the potential to help human users better understand, appropriately trust, and effectively manage AI models (Gunning et al. 2021).

The United States Department of Defense recently adopted a set of five ethical principles for the development and deployment of autonomous systems (Board 2019): responsible, equitable, traceable, reliable, and governable. XAI can inform many of these principles, while encompassing various aspects of the AI model lifecycle, including research and development, verification and validation, and user trust and acceptance. In this setting, the success of XAI is dependent on the availability of robust, easy-to-use, and open source tools that can generate meaningful explanations of different types of data-driven AI models.

To address this gap, we previously proposed the Explainable AI Toolkit (XAITK), which contains a searchable repository of XAI capabilities for analytics and autonomy applications (Hu et al. 2021). In this paper, we introduce the *xaitk-saliency* package, which is part of the XAITK and provides an integrated, common software framework focused on saliency. We make the following contributions:

(1) provide an open source, explainable AI toolkit for saliency map computation that can support AI model verification and validation across a wide variety of tasks.

(2) create a set of example notebooks and a novel visualization tool that allows users to interactively explore saliency maps on their own datasets and models.

(3) present two concrete use cases supported by the *xaitk-saliency* package, where saliency maps are shown to be useful in real-world deployment scenarios.

Related Work

Explainable Artificial Intelligence (XAI)

Several different taxonomies of XAI methods have been proposed, with explanations generally falling into different categories based on their scope and required level of model access (Arrieta et al. 2020). Local explanations seek to explain models using individual examples (e.g. one image from a dataset), while global explanations seek to explain models across multiple examples (e.g. at the dataset level). Explanations can either be white-box or black-box, depending on the amount of access to the model the explanation requires. White-box methods typically require internal model access and the computation of model gradients, while black-box methods are model agnostic and can often be applied more generally. As an alternative to post-hoc explanation methods, some have argued that models should be made more

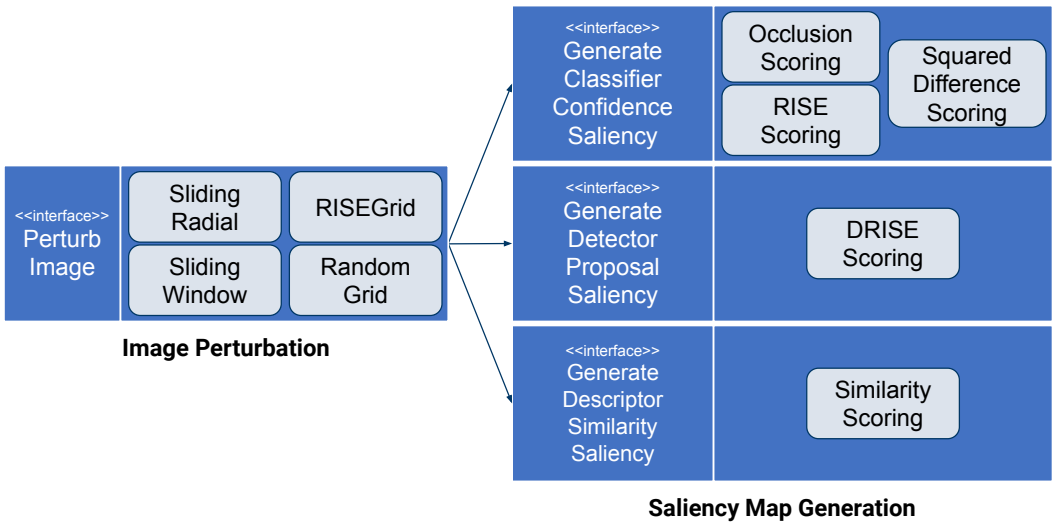

Figure 1: Overview of the *xaitk-saliency* package. Black-box saliency map computation involves the separate steps of image perturbation and saliency map generation. Standardized interfaces specify required inputs and outputs (left side of each module), while specific implementations (gray boxes on right side of each module) must adhere to the defined interfaces.

interpretable in the first place (Rudin 2019).

Saliency Maps as Visual Explanations

Saliency maps are a form of visual explanation that indicate which input features were used by an AI model to generate its output decisions. For example, the ears and whiskers of a cat might be highlighted in the saliency maps of a model trained to classify cat images. While most XAI techniques involving saliency have been developed for image classification tasks (Zeiler and Fergus 2014; Selvaraju et al. 2017), there has been an increasing push to create explanations for other image understanding tasks, including object detection (Petsiuk et al. 2021) and image similarity (Dong, Collins, and Hoogs 2019; Hu, Vasu, and Hoogs 2022). More recently, saliency has also been extended to other data modalities, such as text (Danilevsky et al. 2020). Recent work has also shown that extreme care must be taken in the interpretation of saliency maps, which can sometimes be misleading and may not always faithfully reflect the underlying model being explained (Adebayo et al. 2018).

XAI Frameworks

Several XAI frameworks already exist, such as AIX360 (Arya et al. 2022), Captum (Kokhlikyan et al. 2020), and InterpretML (Nori et al. 2019). There are also dedicated visualization tools for exploring datasets and models such as the What-If Tool (Wexler et al. 2019). These approaches are often centered around specific machine learning frameworks, such as Tensorflow (Abadi et al. 2015) or Pytorch (Paszke et al. 2019), making them harder to use with incompatible models. In contrast, the *xaitk-saliency* package is framework agnostic and supports a wide variety of machine learning and deep learning frameworks. This is done through the use of black-box saliency algorithms, which are model agnostic and only require access to the inputs and outputs of a model. As with other frameworks, we focus on data scientists, researchers, and machine learning practitioners as our main users, providing demonstrations of how to use different saliency algorithms via Jupyter notebooks. We describe the design and usage of the *xaitk-saliency* package in the following sections.

Xaitk-Saliency Design and Usage

The *xaitk-saliency* package is built using Python, which supports the large existing ecosystem of Python-based tools for data science and machine learning. We use a standard strategy and adapter design pattern through the creation of base-level interfaces and more specific implementations. This approach enables the modularity and flexibility needed to support a wide variety of saliency algorithms. At a high level, we split the black-box saliency map computation process into two sequential steps: 1) image perturbation and 2) saliency map generation (Figure 1).

Image perturbation consists of perturbing the input image multiple times (e.g. by sliding an occluding window across the image). Image perturbation is handled by the PerturbImage interface, with current implementations including RandomGrid, RISEGrid, SlidingRadial, and SlidingWindow. We also provide helper functions to perform either batch or streaming image perturbation, which allows users to trade-off runtime and memory usage.

Saliency map generation involves appropriately weighting the resulting perturbed model outputs in order to compute the final saliency maps. Saliency map generation is handled by task-specific interfaces, which include GenerateClassifierConfidenceSaliency (for image classification), GenerateDetectorProposalSaliency (for object detection), and GenerateDescriptorSimilaritySaliency (for image retrieval). Specific implementations of these interfaces are described in more detail in the sections below.

Task	Saliency Algorithm(s)
Image classification	Occlusion-based saliency (Zeiler and Fergus 2014); Randomized Input Sampling for Explanation (RISE) (Petsiuk, Das, and Saenko 2018)
Object detection	Detector-RISE (D-RISE) (Petsiuk et al. 2021)
Image retrieval	Similarity Based Saliency Maps (SBSM) (Dong, Collins, and Hoogs 2019)
Reinforcement learning	Perturbation-based Saliency (Greydanus et al. 2018)

Table 1: Currently implemented saliency algorithms in the *xaitk-saliency* package. Algorithms are black-box and span multiple tasks, such as image classification, object detection, image retrieval, and deep reinforcement learning.

Splitting the saliency map computation process into separate steps allows the individual component steps to be reused or flexibly combined. Having standardized interfaces also facilitates algorithm interoperability, ensuring that the inputs and outputs of each component step are compatible with each other. Users can access saliency algorithms through either a "low-level" API by configuring the image perturbation and saliency map generation steps separately, or through a "high-level" API, which conveniently handles the gluing together of these steps using standard implementations. Example code showing the API is shown in Listing 1.

Listing 1: *xaitk-saliency* Code Example

```
import matplotlib.pyplot as plt
from xaitk_saliency.impls.
    gen_object_detector_blackbox_sal.
    drise import DRISEStack

# Instantiate the D-RISE algorithm
# "high-level" implementation
g = DRISEStack(n=500, s=8, p1=0.5)

# Generate saliency maps for some
# image detections
saliency_maps = g(ref_image, det_boxes,
                  det_scores, det_model)

# Visualize a saliency map over image
plt.figure()
plt.imshow(ref_image)
plt.imshow(saliency_maps[0],
           cmap='jet_r', alpha=0.3)
```

Saliency Algorithms

The currently implemented saliency algorithms are shown in Table 1. These algorithms are split by task on the left, with the corresponding algorithm name (along with its reference) shown on the right. For *image classification*, occlusion-based saliency (Zeiler and Fergus 2014) and RISE (Petsiuk, Das, and Saenko 2018) are implemented through OcclusionScoring and RISEScoring, respectively. For *object detection*, D-RISE (Petsiuk et al. 2021) is implemented through DRISEScoring. For *image retrieval*, similarity-based saliency maps (Dong, Collins, and Hoogs 2019) are implemented through SimilarityScoring. We also have support for perturbation-based saliency of *deep reinforcement learning* agents (Greydanus et al. 2018) through the SquaredDifferenceScoring implementation.

We expect the set of implemented algorithms to grow as the field of XAI advances and new algorithms are introduced. We also note that all currently supported algorithms are black-box in nature (in support of our model agnostic framework), but this could be enhanced in the future to also include support for commonly used white-box saliency methods, such as Grad-CAM (Selvaraju et al. 2017).

Example Notebooks

The *xaitk-saliency* package also includes a comprehensive set of example Jupyter notebooks which showcase its functionality, while highlighting the diversity of potential use cases. In addition to the option of running these notebooks locally on a Jupyter server, we have made it possible to run all notebooks in the cloud using Google Colab. This allows users to quickly test out example notebooks without having to set up an environment locally. Screenshots from a selected set of these notebooks are shown in Figure 2. The chosen examples span different domains such as medical imaging (explanations for a chest X-ray classifier), underwater imagery (explanations for a fish classifier), and deep reinforcement learning (explanations for an agent trained in a game environment). These examples and others can be found at: https://github.com/xaitk/xaitk-saliency/examples.

Visualization Tools

The use of saliency maps as a form of XAI is often embedded into user workflows that involve the evaluation of AI models. Towards this end, we have developed an interactive, web-based visualization tool that allows users to explore the outputs of different models and saliency algorithms. Figure 3 shows the graphical user interface (GUI) of our *xaitk-saliency-web-demo* tool, which allows a user to input data, select a model to explain, and configure parameters for a chosen saliency algorithm. The GUI visualizes the model's output predictions, as well as saliency maps highlighting the evidence for different classes. The user can also set visualization parameters for the saliency map in order to adjust how it is overlaid on the original image. In the image classification example shown in Figure 3, the model predicts the class 'German shepherd' and highlights positive evidence for this class in blue (while negative evidence from the cat's face is highlighted in red). This visualization tool is highly portable, being able to run locally or via a containerized web interface using the *trame* framework (Jourdain et al. 2022). The code for this visualization tool can be found at: https://github.com/xaitk/xaitk-saliency-web-demo.

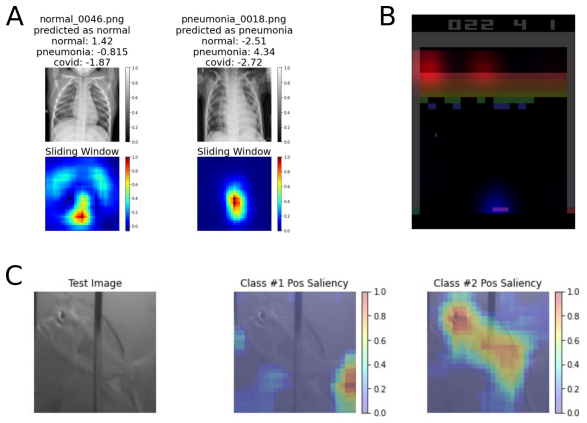

Figure 2: Potential applications of saliency maps available as example notebooks. (A) Chest X-ray classification using MONAI (MONAI Consortium 2020). (B) XAI for a deep reinforcement learning agent trained on the *Breakout-v0* Atari game in OpenAI Gym (Brockman et al. 2016). (C) Fish species classification using VIAME (Dawkins et al. 2017).

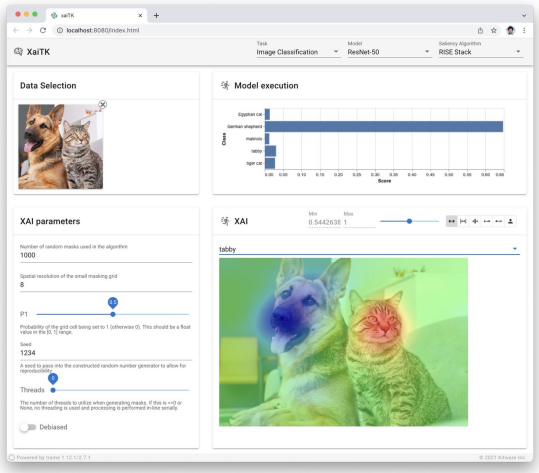

Figure 3: Interactive visualization of saliency maps (an image classification task is shown here). A user can upload an image, choose a particular model to explain, and configure the parameters of the saliency algorithm. Model predictions and saliency maps are shown together, with the ability to also dynamically adjust the visualization.

Example Use Case: Object Detection Model Comparison

A critical question related to the real-world deployment of AI models involves their verification and validation, which goes beyond standard test and evaluation protocols. How does a user know that their model will perform as expected on new data? What level of assurance or justified trust can a user place in their model, and what tools can be used to assess this? Take for example the following hypothetical scenario: imagine two AI models have been trained to detect different types of aircraft in overhead imagery. One model learns to use the correct image features in order to determine detections (e.g. aircraft wing or engine features). The other model incorrectly latches onto spurious correlations in the training data (e.g. the shadow cast by a particular aircraft). This is an example of unintended dataset bias, which will cause the model to fail to generalize when these aircraft shadows are no longer present in the image.

Both models might achieve similar performance on a held-out dataset with similar statistics as the training data. How would a user discover this undesired bias in the data, and correctly choose to use the model that generalizes better? We believe that saliency maps can provide an additional axis of information upon which to evaluate AI models. Saliency maps can provide a more human-understandable view of models compared to raw performance metrics, which may actually be quite similar for different models. In the example above, saliency maps for the biased model might show that the model incorrectly focuses on shadows rather than the aircraft itself. Similarly, saliency maps might be able to help identify edge cases in the data and flag these data for further review by subject matter experts.

To test this hypothesis, we studied object detection models trained on the VisDrone dataset (Zhu et al. 2021a). The VisDrone dataset consists of about 10K images, with over 540K annotated bounding boxes covering 10 different object classes. Challenges with the VisDrone dataset include its long-tailed distribution, the presence of small objects, and visually similar classes. Two pretrained models with publicly available implementations were chosen for comparison: TPH-YOLOv5[1] and CenterNet[2]. TPH-YOLOv5 is a Transformer-based extension of the original YOLO model (Zhu et al. 2021b), while CenterNet is an anchor-less, single-stage detection model (Zhou, Wang, and Krähenbühl 2019). After training, both models achieved similar performance on the test dataset, with the TPH-YOLOv5 model performing slightly better (30.8 vs. 25.9 mAP).

As a form of model verification and validation, we examined the predicted detections of each model on sample images. We also computed class-specific saliency maps on predicted detections in order to identify the image features used by each model. This was done using the RandomGridStack implementation in the *xaitk-saliency* package, which implements a form of black-box saliency for object detectors (Petsiuk et al. 2021). Example detections and their corresponding saliency maps are shown in Figure 4. While both models made similar high-confidence predicted detections, the saliency maps reveal that the two models learned to use slightly different features. For example, for the class 'pedestrian', the TPH-YOLOv5 model learned to focus on the head and feet while the CenterNet model focused more on the torso. We also observed differences in the saliency maps for other object classes, such as 'car' and 'bus' (Figure 4). An open research question is whether the quality of the generated saliency maps (as quantified by metrics such as entropy, etc.)

[1]https://github.com/cv516Buaa/tph-yolov5

[2]https://github.com/GNAYUOHZ/centernet-visdrone

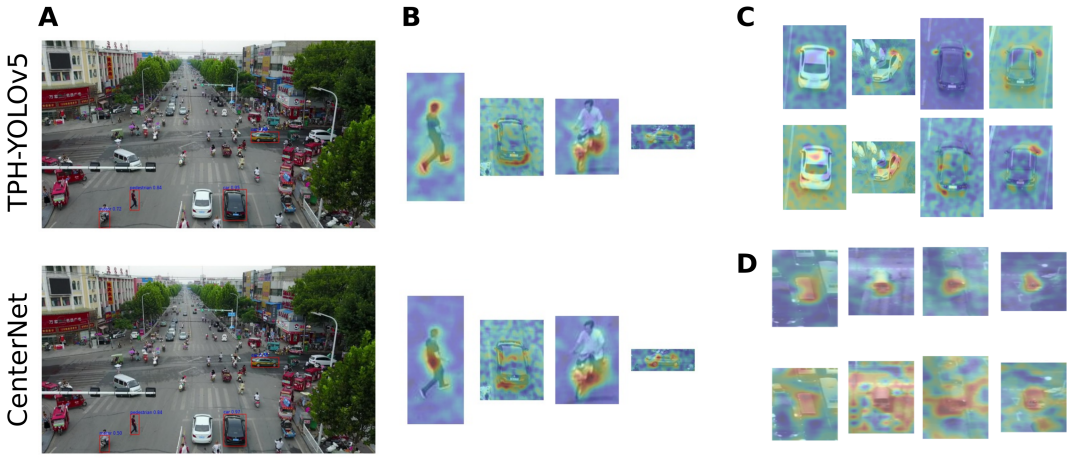

Figure 4: Object detection model comparison using saliency maps. We compared the predicted detections and object-specific saliency maps of two different models trained on the VisDrone aerial dataset (Zhu et al. 2021a). In each panel, the TPH-YOLOv5 model (Zhu et al. 2021b) is shown in the top row and the CenterNet model (Zhou, Wang, and Krähenbühl 2019) is shown in the bottom row. (A) Both models produce similar high-confidence detections (shown with red bounding boxes). (B) Despite having similar detections, the saliency maps corresponding to these detections reveal subtle differences in the input features used by the two models, e.g. TPH-YOLOv5 (top) focuses on the head and feet of pedestrians, while CenterNet (bottom) focuses on the torso. (C) For the 'car' class, TPH-YOLOv5 (top) seems to focus more on side view mirrors compared to CenterNet (bottom). (D) For the relatively rare 'bus' class, TPH-YOLOv5 (top) shows better localized saliency maps compared to CenterNet (bottom).

is a good indicator of model generalization or robustness. The example notebook that reproduces these results can be found here: https://github.com/XAITK/xaitk-saliency/blob/master/examples/ModelComparisonWithSaliency.ipynb.

Example Use Case: Doppelgänger Saliency for Person Re-Identification

As a second example use case, we study the role of AI models for person re-identification (ReID). Person ReID seeks to accurately identify individuals over time and across various environmental changes such as different camera views, in-door/outdoor settings, etc. Modern surveillance systems have become increasingly dependent on AI to provide actionable information for real-time decision making. A critical question relates to how these systems handle difficult ethical dilemmas, such as the ReID of similar looking individuals (which we here on out refer to as doppelgängers). To address this issue, we previously developed a novel saliency-based technique to help users identify discriminative features between doppelgängers (RichardWebster et al. 2022).

For these experiments, we used the video-based MARS (Motion Analysis and Re-identification Set) dataset (Zheng et al. 2016). The dataset consists of 1,261 different pedestrians, spanning more than 20,000 tracklets collected from six near-synchronized cameras. Each of the videos contains significant variations in pose, color, and illumination, along with the resolution of different pedestrians. To generate salient differences, RichardWebster et al. first trained a support vector machine (SVM) on features computed from the tracklets of two different individuals — a tracklet is a sequence of chips that have been cropped to an individual from a source video. To ensure the robustness of this approach, features from three different person ReID models with varying architectures were used. The SVM classifier is trained to discriminate between the two tracklets and computes a probability that each chip in the tracklet belongs to one of the individuals. This can be viewed as a simple two-class image classification problem, where each class is one of the two individuals in the doppelgänger pair.

RichardWebster et al. then used the SlidingRadial and OcclusionScoring implementations provided by the *xaitk-saliency* package to compute the doppelgänger saliency. Intuitively, image regions that are highlighted by the saliency map are critical for the classifier's prediction (i.e. removing them impacts the predicted class probability). In other words, the region of pixels that has the strongest signal is also the region which most strongly discriminates it from its doppelgänger counterpart. We further quantified this by using the set of insertion and deletion metrics introduced in Petsiuk, Das, and Saenko (2018), along with a comparison to random saliency maps as a form of sanity check. Figure 5 shows example doppelgänger pairs (top) along with potentially discriminating features in the resulting saliency maps (bottom). Each saliency map highlights potential differences between the two individuals, such as shirt and shoe color. We note that a region does not have to be highlighted in both images to be considered a difference.

The computed saliency maps can alert human users of the presence of doppelgängers and provide important visual evidence to reduce the potential of false matches in these high-stakes situations. The results of the paper suggest that this novel use of visual saliency can improve overall outcomes by helping human users in the person re-identification setting, while assuring the ethical and trusted operation of

Figure 5: Example doppelgänger saliency. (A) Image regions that differ between the two individuals (e.g. face, shirt logo, pants, and shoes) are highlighted in green. For illustration purposes, colored arrows pointing to corresponding image regions are shown. (B) Additional doppelgänger saliency examples. The four pairs shown highlight at least one key difference, in order from left to right: missing logo, shorts instead of dress, different shoes, and different face. In a full person re-identification system, the user can view the highlighted regions to quickly spot visual differences in the doppelgänger pair. Figure adapted from RichardWebster et al. (2022).

surveillance systems. RichardWebster et al. presented their paper and results in the IEEE CVPR Workshop on Fair, Data Efficient and Trusted Computer Vision. We refer the reader to this workshop paper for more detailed information.

Discussion

We have introduced the open source *xaitk-saliency* package, which is a general framework and toolkit for saliency map computation. Alongside the toolkit, we have also developed a novel visualization tool that enables users to interactively explore saliency maps in support of model verification and validation tasks. Through several examples and use cases, we have demonstrated how easy *xaitk-saliency* is to use and extend in order to address different real-world scenarios. We have also shown how the current design and use of black-box saliency algorithms enables both framework agnostic and model agnostic explanations, which is useful for supporting multiple deep learning or machine learning frameworks and when internal model access is not always possible.

Future work should explore quantitative metrics for the goodness or quality of saliency maps, which can be used to study whether saliency maps are predictive of model generalization or robustness. More careful thought should also be given to the design of interfaces and implementations for white-box saliency algorithms, since model inference and gradient computation is often handled differently by different deep learning frameworks. Finally, while the *xaitk-saliency* package is currently focused on image understanding tasks, it should also be extended for use with other data modalities such as tabular or text data.

The availability of big data and compute is helping drive forward progress towards the realization of novel AI technologies. However, the growing use of AI in many applications also creates a need for explanation of the underlying AI models driving this progress. With more and more data being used to build these AI models, new ways to interpret, understand, and distill the knowledge that is learned by these models becomes critical. We believe that saliency maps as a form of visual explanation will play an important role in the verification, validation, and eventual use of AI models. Importantly, novel visualization tools and techniques can also help support the use of saliency maps throughout the entire AI model lifecycle. This will allow data scientists, researchers, and machine learning practitioners to develop new models, while assuring the use of trusted, reliable, and robust AI models. We believe that the *xaitk-saliency* package will be of broad interest to anyone who deploys AI capabilities in operational settings and needs to validate, characterize, and trust AI performance across a wide range of real-world conditions and application areas.

Acknowledgments

This material is based upon work supported by the United States Air Force and DARPA under Cooperative Agreement number FA875021C0130. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the United States Air Force and DARPA.

References

Abadi, M.; et al. 2015. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. https://www.tensorflow.org/. Accessed: 2022-11-11.

Adebayo, J.; Gilmer, J.; Muelly, M.; Goodfellow, I.; Hardt, M.; and Kim, B. 2018. Sanity checks for saliency maps. In *Advances in Neural Information Processing Systems*, 9505–9515.

Arrieta, A. B.; et al. 2020. Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*, 58: 82–115.

Arya, V.; et al. 2022. AI Explainability 360: Impact and

Design. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, 12651–12657.

Board, D. I. 2019. AI Principles: Recommendations on the Ethical Use of Artificial Intelligence by the Department of Defense. *Supporting document, Defense Innovation Board*.

Brockman, G.; Cheung, V.; Pettersson, L.; Schneider, J.; Schulman, J.; Tang, J.; and Zaremba, W. 2016. Openai gym. *arXiv preprint arXiv:1606.01540*.

Danilevsky, M.; Qian, K.; Aharonov, R.; Katsis, Y.; Kawas, B.; and Sen, P. 2020. A Survey of the State of Explainable AI for Natural Language Processing. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, 447–459.

Dawkins, M.; Sherrill, L.; Fieldhouse, K.; Hoogs, A.; Richards, B.; Zhang, D.; Prasad, L.; Williams, K.; Lauffenburger, N.; and Wang, G. 2017. An open-source platform for underwater image and video analytics. In *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 898–906. IEEE.

Dong, B.; Collins, R.; and Hoogs, A. 2019. Explainability for Content-Based Image Retrieval. In *CVPR Workshops*, 95–98.

Doshi-Velez, F.; and Kim, B. 2017. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*.

Greydanus, S.; Koul, A.; Dodge, J.; and Fern, A. 2018. Visualizing and understanding atari agents. In *International conference on machine learning*, 1792–1801. PMLR.

Gunning, D.; Vorm, E.; Wang, J. Y.; and Turek, M. 2021. DARPA's explainable AI (XAI) program: A retrospective. *Applied AI Letters*, 2(4): e61.

Holzinger, A.; Biemann, C.; Pattichis, C. S.; and Kell, D. B. 2017. What do we need to build explainable AI systems for the medical domain? *arXiv preprint arXiv:1712.09923*.

Hu, B.; Tunison, P.; Vasu, B.; Menon, N.; Collins, R.; and Hoogs, A. 2021. XAITK: The explainable AI toolkit. *Applied AI Letters*, 2(4): e40.

Hu, B.; Vasu, B.; and Hoogs, A. 2022. X-MIR: EXplainable Medical Image Retrieval. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 440–450.

Jourdain, S.; Avery, P.; Harris, C.; Schroeder, W.; Berger, D.; and Tunison, P. 2022. trame. https://kitware.github.io/trame/. Accessed: 2022-11-11.

Kokhlikyan, N.; et al. 2020. Captum: A unified and generic model interpretability library for pytorch. *arXiv preprint arXiv:2009.07896*.

MONAI Consortium. 2020. Project MONAI. https://monai.io/. Accessed: 2022-11-11.

Nori, H.; Jenkins, S.; Koch, P.; and Caruana, R. 2019. InterpretML: A Unified Framework for Machine Learning Interpretability. *arXiv preprint arXiv:1909.09223*.

Paszke, A.; et al. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In Wallach, H.; Larochelle, H.; Beygelzimer, A.; d'Alché-Buc, F.; Fox, E.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 32*, 8024–8035. Curran Associates, Inc.

Petsiuk, V.; Das, A.; and Saenko, K. 2018. Rise: Randomized input sampling for explanation of black-box models. *arXiv preprint arXiv:1806.07421*.

Petsiuk, V.; Jain, R.; Manjunatha, V.; Morariu, V. I.; Mehra, A.; Ordonez, V.; and Saenko, K. 2021. Black-box explanation of object detectors via saliency maps. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 11443–11452.

RichardWebster, B.; Hu, B.; Fieldhouse, K.; and Hoogs, A. 2022. Doppelganger Saliency: Towards More Ethical Person Re-Identification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2847–2857.

Rudin, C. 2019. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5): 206–215.

Samek, W.; Wiegand, T.; and Müller, K.-R. 2017. Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models. *arXiv preprint arXiv:1708.08296*.

Selvaraju, R. R.; Cogswell, M.; Das, A.; Vedantam, R.; Parikh, D.; and Batra, D. 2017. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, 618–626.

Vilone, G.; and Longo, L. 2020. Explainable Artificial Intelligence: a Systematic Review. *arXiv preprint arXiv:2006.00093*.

Wexler, J.; Pushkarna, M.; Bolukbasi, T.; Wattenberg, M.; Viégas, F.; and Wilson, J. 2019. The what-if tool: Interactive probing of machine learning models. *IEEE transactions on visualization and computer graphics*, 26(1): 56–65.

Zeiler, M. D.; and Fergus, R. 2014. Visualizing and understanding convolutional networks. In *European conference on computer vision*, 818–833. Springer.

Zheng, L.; Bie, Z.; Sun, Y.; Wang, J.; Su, C.; Wang, S.; and Tian, Q. 2016. Mars: A video benchmark for large-scale person re-identification. In *European conference on computer vision*, 868–884. Springer.

Zhou, X.; Wang, D.; and Krähenbühl, P. 2019. Objects as points. *arXiv preprint arXiv:1904.07850*.

Zhu, P.; Wen, L.; Du, D.; Bian, X.; Fan, H.; Hu, Q.; and Ling, H. 2021a. Detection and Tracking Meet Drones Challenge. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1–1.

Zhu, X.; Lyu, S.; Wang, X.; and Zhao, Q. 2021b. TPH-YOLOv5: Improved YOLOv5 based on transformer prediction head for object detection on drone-captured scenarios. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2778–2788.