Journal of Machine Learning Research 23 (2022) 1-6 Submitted 7/21; Revised 5/22; Published 6/22

InterpretDL: Explaining Deep Models in PaddlePaddle

Xuhong Li LIXUHONG@BAIDU.COM
Haoyi Xiong* XIONGHAOYI@BAIDU.COM
Xingjian Li LIXINGJIAN@BAIDU.COM
Xuanyu Wu V_WUXUANYU@BAIDU.COM
Zeyu Chen CHENZEYU01@BAIDU.COM
Dejing Dou DOUDEJING@BAIDU.COM
Baidu, Inc., Beijing, China

Editor: Alexandre Gramfort

Abstract

Techniques to explain the predictions of deep neural networks (DNNs) have been largely required for gaining insights into the black boxes. We introduce *InterpretDL*, a toolkit of explanation algorithms based on *PaddlePaddle*, with uniformed programming interfaces and "plug-and-play" designs. A few lines of codes are needed to obtain the explanation results without modifying the structure of the model. *InterpretDL* currently contains 16 algorithms, explaining training phases, datasets, global and local behaviors of post-trained deep models. *InterpretDL* also provides a number of tutorial examples and showcases to demonstrate the capability of *InterpretDL* working on a wide range of deep learning models, e.g., Convolutional Neural Networks (CNNs), Multi-Layer Preceptors (MLPs), Transformers, etc., for various tasks in both Computer Vision (CV) and Natural Language Processing (NLP). Furthermore, *InterpretDL* modularizes the implementations, making efforts to support the compatibility across frameworks. The project is available at `https://github.com/PaddlePaddle/InterpretDL`.

Keywords: Explanation, Interpretation Algorithms, Trustworthiness, Deep Models

1. Background

Deep neural networks (LeCun et al., 2015) are well-known for their excellent performance in handling various machine learning and artificial intelligence tasks. Meanwhile, they are also known for their black-box natures (Castelvecchi, 2016); it is often difficult for humans to understand the prediction results of these deep models. In recent years, many interpretation tools (Ribeiro et al., 2016; Selvaraju et al., 2017; Smilkov et al., 2017; Sundararajan et al., 2017; Toneva et al., 2019; Wang et al., 2020) have been proposed to explain or reveal the ways that deep models make decisions, in order to identify *whether the deep models work* and *why they work*. While most interpretation algorithms for explaining deep models have been already open-sourced, there are still several issues and technical burdens for industrial and academic practices.

First of all, deep models in various domains (e.g., structural data, images, and texts) desire varying types of explanations. For example, one could easily explain the prediction results of deep models with structural data through finding the important features/variables

*. Corresponding Author.

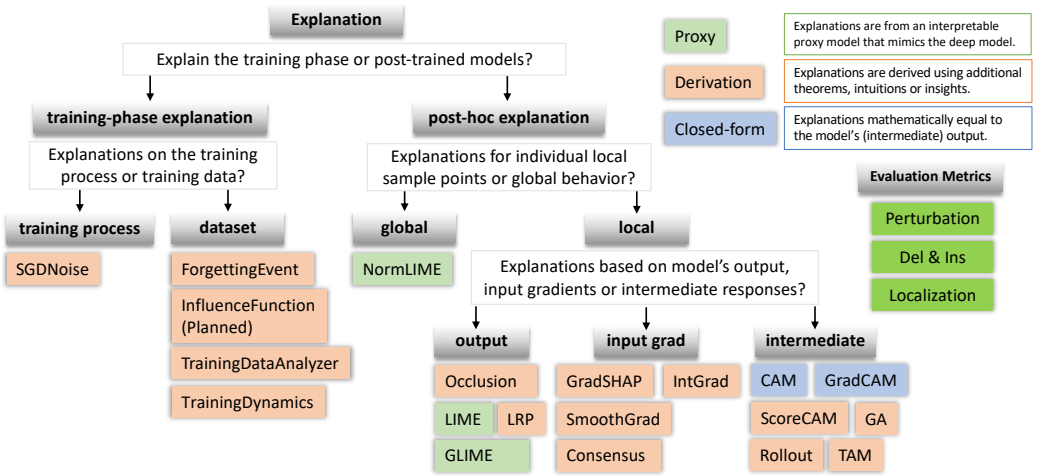


Figure 1: The tree-based taxonomy for Algorithms implemented in *InterpretDL*.

in the given data point, while the interpretation algorithms for image classification models use discriminative pixels (Smilkov et al., 2017) or superpixels (Ribeiro et al., 2016) for visual explanations. Then, it lacks a unified abstraction layer (e.g., a standard set of Application Programming Interfaces (APIs)) for the existing interpretation algorithms. Thus, there is a need to retrieve, read, and comprehend a long list of documents and even source codes to use an interpretation algorithm. Finally, most interpretation algorithms were implemented independently, in an ad-hoc manner, based on different programming frameworks of deep learning (e.g., Pytorch and Tensorflow). Though some efforts (Alber et al., 2019; tfe; Kokhliyan et al.) have been done to collect and unify a number of explanation algorithms for Tensorflow or Pytorch, still, it would be heavy to explain deep learning models that are trained on other frameworks.

2. The Proposed Toolkit *InterpretDL*

To cope with these issues, we report *InterpretDL*, an open-source library with mainstream explanation algorithms for deep models' explanations based on PaddlePaddle (Ma et al., 2019), which is a deep learning framework with over 4 million developers currently and numerous users. Model explanations are one of the most desirable features for the PaddlePaddle ecosystem. To avoid unexpected and unnecessary switch-framework efforts, we provide the easy-to-use tool *InterpretDL* for explaining the behaviors of PaddlePaddle deep models. We follow the plug-and-play fashion and use universal and straightforward APIs, with at least one detailed tutorial for each algorithm, towards a user-friendly library for both industrial and academic practices.

The proposed toolkit *InterpretDL* also considers the compatibility across frameworks, by modularizing the explanation algorithms where the implementations concerning the framework are wrapped into one single module. In this way, any other module that returns desired outputs would be compatible with *InterpretDL*. We have currently supported this feature for most gradient-based algorithms (Smilkov et al., 2017; Lundberg and Lee, 2017) and model-agnostic ones (Ribeiro et al., 2016; Ahern et al., 2019). We will continuously support this compatibility across frameworks for more explanation algorithms.

Furthermore, we provide a new categorization for explanation algorithms. As shown in Figure 1, we show a tree-based taxonomy for the 18 current (and planned) algorithms

```python
from interpretdl import SmoothGradInterpreter

# Load a pretrained model from PaddlePaddle model zoo.
from paddle.vision.models import resnet50
paddle_model = resnet50(pretrained=True)

# Or load a custom model.
# paddle_model = a_custom_model()
# model_path = "path/to/trained_weights.pdparams"
# paddle_model.set_dict(paddle.load(model_path))

sg = SmoothGradInterpreter(paddle_model, device="gpu:0")
gradients = sg.interpret("test.jpg", visual=True, save_path=None)
```

Listing 1: Codes for SmoothGrad-based explanations of ResNet-50 based on a given image.

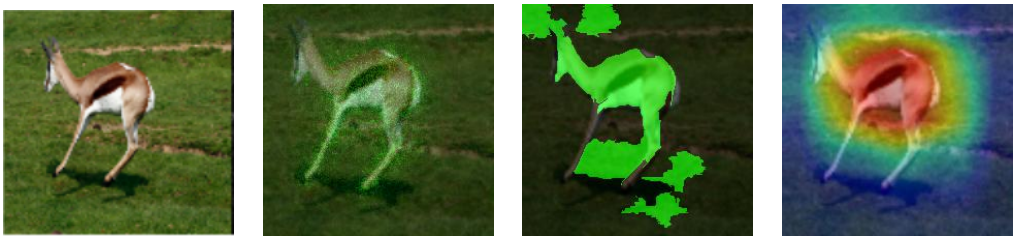
in *InterpretDL* with the properties of algorithms, i.e., "proxy", "derivation" and "closed-form", following Li et al. (2021). The algorithms were chosen according to their popularity, efficiency, effectiveness and potentials. Compared to existing deep learning interpretability libraries (tfe; Kokhliyan et al.; Alber et al., 2019), *InterpretDL* considers a wider concept of explanations. Training-phase explanations are included for investigating the insights to understand the training process and the influences or contributions of the training set to the trained model. As for post-hoc explanations, we provide both global and local explanation algorithms. More specifically, we provide three different sub-categories of local ones beyond using vanilla gradients and activations.

The trustworthiness is also one important property of explanation algorithms that assures the explanation results be faithful to the model. *InterpretDL* currently provides three methods to validate the trustworthiness, based on perturbation tests (Samek et al., 2016; Petsiuk et al., 2018) and localization ability (Zhang et al., 2018).

Comparisons to Existing Interpretation Libraries Compared to the existing open-sourced libraries of interpretation/explanation algorithms for machine learning in general, e.g., InterpretML (Nori et al., 2019), AIX360 (Arya et al., 2020) and Alibi (Klaise et al., 2021), *InterpretDL* is designed and optimized for interpreting deep neural networks, which would be computational intensive with the design and implementation of specific deep models. For interpreting deep models, Captum (Kokhliyan et al.) provided post-hoc local explanation algorithms based on Pytorch (Paszke et al., 2019), tf-explain (tfe) for Tensorflow (Abadi et al., 2016) models, and similarly iNNvestigate (Alber et al., 2019) for Keras with TensorFlow backend. Compared to them, the proposed toolkit *InterpretDL* considers a wider concept for explanations, containing both training-phase explanations (Toneva et al., 2019) and post-hoc ones (Ahern et al., 2019; Wang et al., 2020), and focuses on the PaddlePaddle ecosystem. Meanwhile, *InterpretDL* supports cross-framework compatibility, through modularized designs and simple, clear, and unified APIs.

3. *InterpretDL* Examples with Codes and Results

To demonstrate the efficiency and effectiveness of using *InterpretDL* to interpret the prediction results of deep models, we provide two examples, including code and visualization results. Listing 1 shows lines of codes for interpreting a trained ResNet-50 model, where the classification result of the model based on the image stored in "test.jpg" is explained using SmoothGrad algorithm (Smilkov et al., 2017). Figure 2 (b) shows the SmoothGrad



(a) Original Image (b) SmoothGrad (c) LIME (d) GradCAM

Figure 2: Explanations of a ResNet-50 model based on a given image using three algorithms: SmoothGrad (b), where the explanation scores are added to the "green" channel; LIME (c), where a threshold of top 20% superpixels is applied; GradCAM (d), where the low-resolution explanation is resized to the original image scale and the "JET" color map is applied.

True Label	Predicted Label (Prob)	Target Label	Word Importance
1	1 (1.00)	1	it ' s a charming and often affecting journey .
1	1 (1.00)	1	the movie achieves as great an impact by keeping these thoughts hidden as . . . (quills) did by showing them .
0	0 (0.93)	0	this one is definitely one to skip , even for horror movie fanatics .
0	0 (0.97)	0	in its best moments , resembles a bad high school production of grease , without benefit of song .

Figure 3: Interpretation results of a sentiment classifier based on texts.

results and under the same settings, we include the results of LIME (Ribeiro et al., 2016) and GradCAM (Selvaraju et al., 2017) in Figures 2 (c) and (d). The example shows that only with two lines of codes (e.g., lines 12 and 13 in Listing 1), *InterpretDL* can explain the prediction result of a given model based on a given sample, where the user can choose to both visualize the interpretation result immediately and store the result with a given file path. Similar example based on NLP models is given in Figure 3, where the model has been trained to classify the sentiment of a given sentence, and the interpretation result prefers the words that might affect the classification results.

4. Conclusions and Future Plans

InterpretDL is hosted on GitHub[1] under Apache License 2.0. The package can be easily installed from Python Package Index (PyPI) or from the source repository for development/developer usages. *InterpretDL* provides the documentation[2] for each implemented function, which is compatible to the sphinx documentation generator, enabling the automatic documentation generation. Interactions with the community, including pull requests and proposals of issues, can be done within the GitHub website with regular updates. *InterpretDL* follows the PEP8 style for Python codes. Unit tests with a line coverage of 93% of the code and the continuous integration guarantee the code quality. In the future, we would continue implementing and integrating new interpretation algorithms into *InterpretDL*, with the same APIs and designs. An interface is also planned to facilitate visual comparisons. Other valuable components are also in consideration.

1. Available online: `https://github.com/PaddlePaddle/InterpretDL`
2. Available online: `https://interpretdl.readthedocs.io/en/latest/interpretdl.html`

Acknowledgments

The first two authors are supported in part by National Key R&D Programs of China under the grant No. 2021ZD0110303.

References

tf-explain: Interpretability methods for tf.keras models with tensorflow 2.0. In *GitHub repository, https://github.com/sicara/tf-explain*.

Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th USENIX symposium on operating systems design and implementation (OSDI 16)*, pages 265–283, 2016.

Isaac Ahern, Adam Noack, Luis Guzman-Nateras, Dejing Dou, Boyang Li, and Jun Huan. Normlime: A new feature importance metric for explaining deep neural networks. *arXiv preprint arXiv:1909.04200*, 2019.

Maximilian Alber, Sebastian Lapuschkin, Philipp Seegerer, Miriam Hägele, Kristof T. Schütt, Grégoire Montavon, Wojciech Samek, Klaus-Robert Müller, Sven Dähne, and Pieter-Jan Kindermans. iNNvestigate neural networks! *Journal of Machine Learning Research*, 2019.

Vijay Arya, Rachel KE Bellamy, Pin-Yu Chen, Amit Dhurandhar, Michael Hind, Samuel C Hoffman, Stephanie Houde, Q Vera Liao, Ronny Luss, Aleksandra Mojsilovic, et al. AI explainability 360: An extensible toolkit for understanding data and machine learning models. *Journal of Machine Learning Research*, 2020.

Davide Castelvecchi. Can we open the black box of AI? *Nature News*, 538(7623):20, 2016.

Janis Klaise, Arnaud Van Looveren, Giovanni Vacanti, and Alexandru Coca. Alibi explain: Algorithms for explaining machine learning models. *Journal of Machine Learning Research*, 22(181):1–7, 2021. URL `http://jmlr.org/papers/v22/21-0017.html`.

Narine Kokhliyan, Edward Wang, Vivek Miglani, and Orion Richardson. Captum. In *GitHub repository, https://github.com/pytorch/captum*.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553): 436–444, 2015.

Xuhong Li, Haoyi Xiong, Xingjian Li, Xuanyu Wu, Xiao Zhang, Ji Liu, Jiang Bian, and Dejing Dou. Interpretable deep learning: Interpretation, interpretability, trustworthiness, and beyond. *arXiv preprint arXiv:2103.10689*, 2021.

Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems (Neurips)*, 2017.

Yanjun Ma, Dianhai Yu, Tian Wu, and Haifeng Wang. PaddlePaddle: An open-source deep learning platform from industrial practice. *Frontiers of Data and Computing*, 2019.

Harsha Nori, Samuel Jenkins, Paul Koch, and Rich Caruana. InterpretML: A unified framework for machine learning interpretability. *arXiv preprint arXiv:1909.09223*, 2019.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 2019.

Vitali Petsiuk, Abir Das, and Kate Saenko. Rise: Randomized input sampling for explanation of black-box models. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2018.

Marco Tulio Ribeiro, Sameer Singh, and Carlos 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.

Wojciech Samek, Alexander Binder, Grégoire Montavon, Sebastian Lapuschkin, and Klaus-Robert Müller. Evaluating the visualization of what a deep neural network has learned. *IEEE transactions on neural networks and learning systems*, 2016.

Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-CAM: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE International Conference on Computer Vision*, 2017.

Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. SmoothGrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825*, 2017.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International Conference on Machine Learning (ICML)*, 2017.

Mariya Toneva, Alessandro Sordoni, Remi Tachet des Combes, Adam Trischler, Yoshua Bengio, and Geoffrey J Gordon. An empirical study of example forgetting during deep neural network learning. *International Conference on Learning Representations (ICLR)*, 2019.

Haofan Wang, Zifan Wang, Mengnan Du, Fan Yang, Zijian Zhang, Sirui Ding, Piotr Mardziel, and Xia Hu. Score-cam: Score-weighted visual explanations for convolutional neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, 2020.

Jianming Zhang, Sarah Adel Bargal, Zhe Lin, Jonathan Brandt, Xiaohui Shen, and Stan Sclaroff. Top-down neural attention by excitation backprop. *International Journal of Computer Vision*, 2018.