Journal of Machine Learning Research 17 (2016) 1-5 Submitted 12/15; Revised 5/16; Published 6/16

The LRP Toolbox for Artificial Neural Networks

Sebastian Lapuschkin SEBASTIAN.LAPUSCHKIN@HHI.FRAUNHOFER.DE
Fraunhofer Heinrich Hertz Institute, Video Coding and Analytics
10587 Berlin, Germany

Alexander Binder ALEXANDER_BINDER@SUTD.EDU.SG
Singapore University of Technology, ISTD
Singapore 487372, Singapore

Grégoire Montavon GREGOIRE.MONTAVON@TU-BERLIN.DE
Berlin Institute of Technology, Machine Learning Group
10623 Berlin, Germany

Klaus-Robert Müller KLAUS-ROBERT.MUELLER@TU-BERLIN.DE
Berlin Institute of Technology, Machine Learning Group
10623 Berlin, Germany
Korea University, Department of Brain and Cognitive Engineering
Seoul 02841, Korea

Wojciech Samek WOJCIECH.SAMEK@HHI.FRAUNHOFER.DE
Fraunhofer Heinrich Hertz Institute, Video Coding and Analytics
10587 Berlin, Germany

Editor: Geoff Holmes

Abstract

The Layer-wise Relevance Propagation (LRP) algorithm explains a classifier's prediction specific to a given data point by attributing *relevance scores* to important components of the input by using the topology of the learned model itself. With the LRP Toolbox we provide platform-agnostic implementations for explaining the predictions of pre-trained state of the art Caffe networks and stand-alone implementations for fully connected Neural Network models. The implementations for Matlab and python shall serve as a playing field to familiarize oneself with the LRP algorithm and are implemented with readability and transparency in mind. Models and data can be imported and exported using raw text formats, Matlab's `.mat` files and the `.npy` format for numpy or plain text.

Keywords: layer-wise relevance propagation, explaining classifiers, deep learning, artificial neural networks, computer vision

1. Introduction

Classification of images has become a key ingredient in many computer vision applications, with nonlinear methods such as Deep Neural Networks (DNNs) being the gold standard in the fields of vision (Krizhevsky et al., 2012; Ciresan et al., 2012b; Szegedy et al., 2014; Ciresan et al., 2012a), natural language processing (Collobert et al., 2011; Socher et al., 2013), speech recognition (Yu and Deng, 2014) or physics (Montavon et al., 2013); see also (Montavon et al., 2012). Although performing incredibly well, they lack transparency. In the sciences, however, understanding a learning machine in order to gain scientific insight on

the problem analysed is often as important as record prediction performance. This is also one of the reasons for the popularity of linear modeling as interpretation is straight forward. In this sense DNNs so far held a disadvantage in practice over simpler but interpretable models. Especially DNNs act as black boxes due to their multilayer nonlinear structure. However, interest in gaining insight into the decision process of nonlinear methods has peaked recently. Several works have been using sensitivity maps (Baehrens et al., 2010; Rasmussen et al., 2012) for visualization of classifier predictions which were based on using partial derivatives at the prediction point $\boldsymbol{x}$.

For DNNs promising approaches for opening the black box are deconvolution networks (Zeiler and Fergus, 2014) highlighting activated input patterns for object detected during the forward pass of a convolutional neural network, saliency maps (Simonyan et al., 2013) visualizing local sensitivities at the input point and LRP (Bach et al., 2015) producing explanatory input patterns that indicate evidence for or against a prediction target of choice in given data. The latter – for which this toolbox is provided and (Samek et al., 2015) have measured and verified meaningfulnes of the computed heatmaps – takes a pre-trained model and an evaluation data point and explains the classifier's decision, e.g. LRP decomposes the decision function such that for each input dimension d a *relevance score* $R_d^{(1)}$ is computed, which indicates that the state of x_d speaks for the presence of the prediction target if $R_d^{(1)} > 0$ and against it if $R_d^{(1)} < 0$. By allowing both signs for $R_d^{(1)}$ also class background information can be modelled in a natural manner.

To this end, the model output $f(\boldsymbol{x})$ is backwards-propagated through the model to the input layer as *relevance* R, by taking into account the model's reaction to the input $\boldsymbol{x}$ in all its intermediate representations at all computational layers l. Due to the relevance conservation principle as a constraint to LRP (Bach et al., 2015), no relevance is lost or gained in between layers of computation, e.g.

$$\sum_i R_i^{(l)} = \sum_j R_j^{(l+1)} \text{ and thus } \sum_d R_d^{(1)} = f(\boldsymbol{x}).$$

Note that to this end, the propagated relevances are always normalized. The relevance values $R_d^{(1)}$ can then be visualized as a heatmap, providing valuable insight to the classifier's decision process. For a theoretical view on LRP we refer the reader to Montavon et al. (2015)

2. Capabilities of the LRP Toolbox for Artificial Neural Networks

The LRP Toolbox provides platform-independant stand-alone implementations of the LRP algorithm for python and Matlab, as well as adapted `.cpp` modules to support LRP for the Caffe deep learning framework (Jia et al., 2014) and operates on pre-trained neural network models. The functionality of the framework and a demo thereof is accessible via command line or an IDE of choice for Matlab, python and Caffe (C++). The Caffe version is based on the caffe-master branched on 3rd October 2015. The latest official release is available from `http://www.heatmapping.org`.

We provide three implementations due to the different strengths and advantages of the three implementations. Caffe is an established toolbox for neural networks, written in C++, and has the big advantage that it comes with many pretrained neural networks in the Caffe Model Zoo. The Caffe version works out of the box with the BVLC reference, the

GoogLeNet model and the VGG CNN models from Chatfield et al. (2014). On the other hand, many scientists are unfamiliar with programming C++ interfaces, thus creating a barrier for experimentation when one is interested in modifying and trying out different LRP implementations. Note that LRP is a principle which permits multiple solutions. The python and Matlab implementations allow an easier access and for easy modification of LRP rules, in particular because python and Matlab are popular among machine learners and are taught in many undergrad university curricula. Furthermore the python and Matlab implementations require less external libraries which is practical in restricted setups where the user does not have full control over the installed packages. These trade-offs lead us to the choice of implementing LRP in three ways.

2.1 Platforms and Requirements

The python and Matlab implementations are available for systems running Linux, Windows and OSX due to the platform agnostic nature of python and Matlab. Caffe runs on Linux and OSX, however Caffe ports for Windows became available very recently on github (e.g. `https://github.com/happynear/caffe-windows`). Known and successfully tested minimum package requirements are available in the toolbox manual.

2.2 User Interface

The example implementations of the LRP pipeline can be executed and modified with the provided files `matlab/lrp_demo.m` and `python/lrp_demo.py` by navigating to the respective folders and executing the scripts using the chosen language's interpreter, e.g. for unix-based systems

```
cd <toolbox_location>/matlab
matlab -nodesktop -r lrp_demo
```

or

```
cd <toolbox_location>/python
python lrp_demo.py
```

The C++-based Caffe implementation of LRP uses a configuration file with its settings being explained in the manual. It takes as further input a text file which contains in each line the path to one image and an integer denoting the class for which the heatmap is to be computed. The third input is a path prefix.

```
cd <toolbox_location>/caffe-master-lrp/demonstrator
./lrp_demo ./config_sequential.txt ./testfilelist.txt ./
```

will then compute heatmaps for the images listed in `./testfilelist.txt`.

2.3 Documentation and Examples

The user manual guides through the installation and use of the toolbox. It also explains the LRP algorithm in detail and instructs on how to compute and interpret the results.

2.4 Developer Access, Licensing and Availability

The source code is provided under FreeBSD (2-Clause) License and is available in a separate archive for each released version. The latest official release of the toolbox code is available from `http://heatmapping.org`. More recent and work-in-progress versions can be found at `https://github.com/sebastian-lapuschkin/lrp_toolbox`.

3. Conclusion

The presented LRP package provides a simple and easy to use implementation that allows a user to explore LRP in Matlab and python or run more involved applications within the Caffe framework. We would like to emphasize that the user can readily take an existing network to apply the LRP procedure. In other words also DNNs from past projects could be retrospectively explained by LRP. Furthermore note that the usage of LRP is not limited to DNNs, in fact, as shown in (Bach et al., 2015) also kernel methods and other learning machines, e.g. using Bag of Words representations can yield heatmaps through LRP. Future work will use the LRP framework and software package in the sciences and for general applications that require explanation.

Acknowledgments

The work was funded in part by the Federal Ministry for Education and Research (BMBF) under Grant 01IS14013A-E and Grant 01GQ1115, as well as by the Deutsche Forschungsgesellschaft (DFG) under Grant MU 987/19-1 and the Brain Korea 21 Plus Program by the Korean Government. Correspondence to KRM and WS.

References

Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLOS ONE*, 10(7):e0130140, 2015.

David Baehrens, Timon Schroeter, Stefan Harmeling, Motoaki Kawanabe, Katja Hansen, and Klaus-Robert Müller. How to explain individual classification decisions. *JMLR*, 11: 1803–1831, 2010.

Ken Chatfield, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Return of the devil in the details: Delving deep into convolutional nets. In *BMVC*, 2014.

Dan Ciresan, Ueli Meier, and Jürgen Schmidhuber. Multi-column deep neural networks for image classification. In *CVPR*, pages 3642–3649. IEEE, 2012a.

Dan C. Ciresan, Alessandro Giusti, Luca Maria Gambardella, and Jürgen Schmidhuber. Deep neural networks segment neuronal membranes in electron microscopy images. In *NIPS*, pages 2852–2860, 2012b.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel P. Kuksa. Natural language processing (almost) from scratch. *JMLR*, 12:2493–2537, 2011.

Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, pages 1106–1114, 2012.

Grégoire Montavon, Geneviève B. Orr, and Klaus-Robert Müller, editors. *Neural networks: Tricks of the trade, reloaded*, volume 7700 of *LNCS*. Springer, 2nd edition, 2012.

Grégoire Montavon, Matthias Rupp, Vivekanand Gobre, Alvaro Vazquez-Mayagoitia, Katja Hansen, Alexandre Tkatchenko, Klaus-Robert Müller, and O Anatole von Lilienfeld. Machine learning of molecular electronic properties in chemical compound space. *NJP*, 15 (9):095003, 2013.

Grégoire Montavon, Sebastian Bach, Alexander Binder, Wojciech Samek, and Klaus-Robert Müller. Explaining nonlinear classification decisions with deep taylor decomposition. *CoRR*, abs/1512.02479, 2015. URL `http://arxiv.org/abs/1512.02479`.

Peter M Rasmussen, Tanya Schmah, Kristoffer H Madsen, Torben E Lund, Stephen C Strother, and Lars K Hansen. Visualization of nonlinear classification models in neuroimaging. In *BIOSIGNALS*, 2012.

Wojciech Samek, Alexander Binder, Grégoire Montavon, Sebastian Bach, and Klaus-Robert Müller. Evaluating the visualization of what a deep neural network has learned. *CoRR*, abs/1509.06321, 2015. URL `http://arxiv.org/abs/1509.06321`.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *CoRR*, abs/1312.6034, 2013.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *EMNLP*, pages 1631–1642, 2013.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *CoRR*, abs/1409.4842, 2014.

Dong Yu and Li Deng. *Automatic speech recognition - a deep learning approach*. Springer, October 2014. URL `http://research.microsoft.com/apps/pubs/default.aspx?id=230891`.

Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *ECCV*, pages 818–833, 2014. doi: 10.1007/978-3-319-10590-1_53.