Layered Explanations: Interpreting Neural Networks with Numerical Influence Measures

Xuan-Vinh Ho
National University of Singapore
hxvinh@comp.nus.edu.sg

Abstract

Deep learning is currently the leading paradigm in machine learning applications, mostly due to its effectiveness in a variety of application domains; however, deep learning models are effectively "black boxes", whose behavior is notoriously difficult to explain. This lack of transparency is fast becoming a pressing issue, especially since neural networks are deployed in high-stakes domains, which require human interpretable explanations. We explain the decisions of neural networks using layered explanations: using layerwise selection criteria, we identify an influential layer L_k, containing the most explanatory power; next, we compute influence scores for each neural unit in L_k, and backpropagate them to the input domain as explanations. Preliminary results show that our explanations match human intuition and act consistently throughout different layers. Combining the efficacy of numerical influence measures with backpropagation techniques allows us to maintain a valid influence model, while exploiting the underlying network structure.

Overview

In recent years neural networks have seen widespread use in a variety of increasingly high-stakes domains (Hinton et al., 2006; Krizhevsky et al., 2012; LeCun et al., 2015). The algorithms that generate neural networks are, by and large, easy to understand; however, their output is extremely difficult to interpret, especially if they were trained using large volumes of data. Our goal is thus to *design sound explanation frameworks for neural networks, that work well on various network architectures, utilizing state-of-the-art numerical influence measures*.

Our framework is part of a fast-growing body of work on *algorithmic transparency*: tools that devise faithful explanations of a learning model. Such explanations can take many forms: influence scores (Sliwinski et al., 2017; Datta et al., 2016), heatmaps (Springenberg et al., 2014; Simonyan et al., 2013), model approximations (Ribeiro et al., 2016), datapoint-based (Koh and Liang, 2017), and more. We combine numerical influence measures and backpropagation techniques, formulating a general explanation framework for neural networks.

We are given a datapoint $\vec{x} \in \mathbb{R}^n$, which was assigned a label $f(\vec{x})$ by some neural network classifier (say, $\vec{x}$ is an individual applying for a job, and $f(\vec{x})$ is a decision on whether to invite them for an interview). A *numerical influence measure* computes a vector of values $\vec{\phi}(\vec{x}) \in \mathbb{R}^n$, where $\phi_i(\vec{x})$ should roughly correspond to the relative importance of the i-th feature in deciding the label $f(\vec{x})$. The measure $\vec{\phi}(\vec{x})$ may use a known labeled dataset $\langle \mathcal{X}, f \rangle$, as well as additional domain knowledge to compute $\phi_i(\vec{x})$; indeed, several works (Baehrens et al., 2010; Ribeiro et al., 2016; Datta et al., 2016; Sliwinski et al., 2017) offer methods for computing influence scores, and can be readily adapted to our method. Backpropagation-based visualizations produce a heatmap that highlights areas relevant to the predictions of a neural network. We use DeconvNet (Zeiler and Fergus, 2014) — a well-known backpropagation technique applicable for convolutional neural networks (CNNs) — but other backpropagation methods are as applicable (Bach et al., 2015; Springenberg et al., 2014).

Measuring the numerical influence of points in the input layer directly often results in low influence attributed to all features; this is particularly true in high-dimensional domains (e.g. image classification tasks), where neural networks are often used. This motivates us to explore hidden layers instead, which possibly capture high-level feature and co-influence of feature subsets.

Model

Fig. 1 depicts the three main steps in proposed method:

1. **Identify influential layer**: we choose a layer for which, informally, one can well-approximate using a linear model. Alternatively, we compute influence scores for each unit in the k-th layer, and choose a layer for which there is a significant difference between the influence of individual units. Note that by definition, the output layer will have high influence on the output, thus our approach is guaranteed to identify at least one important layer. Intuitively, if the influential layer is too close to the input, its explanatory power should be relatively low ("this is an image of a dog because of the pixels in the top-right region"); similarly, influential layers close to the output are also undesirable ("this is a dog because it's a dog"). Ideally, one would identify a "mid-network" layer.

2. **Apply numerical influence measures**: Given a chosen layer L_k and previously retrieved labels of the dataset, we apply MIM (Sliwinski et al., 2017) to compute influence

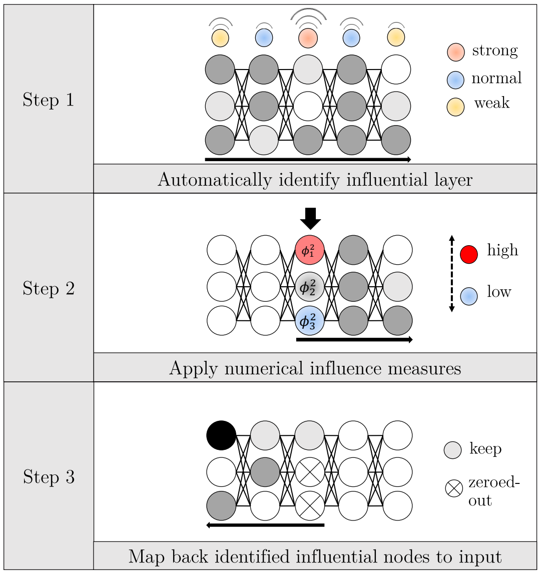
Step 1
strong
normal
weak
Automatically identify influential layer
Step 2
ϕ_1^2
ϕ_2^2
ϕ_3^2
high
low
Apply numerical influence measures
Step 3
keep
zeroed-out
Map back identified influential nodes to input

Figure 1: The proposed three-step method.

scores.

3. **Reconstruct relevant area**: We sort the values of $\vec{\phi}(\vec{x})$ in decreasing absolute value, and use them to systematically select the top influential nodes and backpropagate their raw values to input layer with DeconvNet.

Discussion

We train a typical CNN on the MNIST dataset reduced to data points labeled zero and seven. As a baseline, we create the reconstructed image by single maximum value node (Max) and compare this to selecting the top influential nodes identified by MIM (Fig. 2). Max is inconsistent across different convolutional-maxpooling layers, while MIM outputs agree on the lower left curve as the region that distinguishes this zero instance from the class seven. The reconstructed image also starts to converge from top 1.3%, implying influential units are indeed picked by MIM.

Besides the efficacy of utilized methods, our method also unavoidably inherits weaknesses of both: MIM is currently not immediately applicable in multiclass domains, whereas ReLU activation in CNN filters out pixels contributing negatively in the forward phase. In future work, we intend to evaluate our methods based on the framework proposed by (Samek et al., 2017).

References

Bach, S., Binder, A., Montavon, G., Klauschen, F., Müller, K.-R., and Samek, W. (2015). On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLOS ONE*, 10(7):e0130140.

Baehrens, D., Schroeter, T., Harmeling, S., Kawanabe, M., Hansen, K., and MÃžller, K.-R. (2010). How to explain individual classification decisions. *Journal of Machine Learning Research*, 11(Jun):1803–1831.

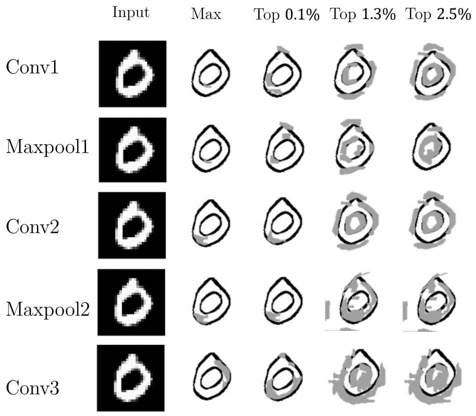

Figure 2: Reconstructed image provided by DeconvNet and our method. **Rows**: the first five layers in the pretrained CNN. **Columns**: the original input, backpropagated value of single maximum value node, top 0.1%, 1.3% and 2.5% influential nodes selected by MIM masking $|\vec{\phi}(\vec{x})|$.

Datta, A., Sen, S., and Zick, Y. (2016). Algorithmic transparency via quantitative input influence: Theory and experiments with learning systems. In *Security and Privacy (SP), 2016 IEEE Symposium on*, pages 598–617. IEEE.

Hinton, G. E., Osindero, S., and Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554.

Koh, P. W. and Liang, P. (2017). Understanding black-box predictions via influence functions. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pages 1885–1894.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Proceedings of the 25th Annual Conference on Neural Information Processing Systems (NIPS)*, pages 1097–1105.

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature*, 521(7553):436.

Ribeiro, M. T., Singh, S., and Guestrin, C. (2016). Why should i trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 1135–1144. ACM.

Samek, W., Binder, A., Montavon, G., Lapuschkin, S., and Müller, K.-R. (2017). Evaluating the visualization of what a deep neural network has learned. *IEEE Transactions on Neural Networks and Learning Systems*, 28(11):2660–2673.

Simonyan, K., Vedaldi, A., and Zisserman, A. (2013). Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*.

Sliwinski, J., Strobel, M., and Zick, Y. (2017). A characterization of monotone influence measures for data classification. *arXiv preprint arXiv:1708.02153*.

Springenberg, J. T., Dosovitskiy, A., Brox, T., and Riedmiller, M. (2014). Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*.

Zeiler, M. D. and Fergus, R. (2014). Visualizing and understanding convolutional networks. In *Proceedings of the 13rd European Conference on Computer Vision (ECCV)*, pages 818–833. Springer.