The Journal of Open Source Software

Delve: Neural Network Eigenspace Computation and Visualization

Justin Shenk[*1,2], **Mats L. Richter**[†2], **and Wolf Byttner**[3]

1 VisioLab, Berlin, Germany **2** Institute of Cognitive Science, University of Osnabrueck, Osnabrueck, Germany **3** Rapid Health, London, England, United Kingdom

DOI: DOIunavailable

Software

- Review
- Repository
- Archive

Editor: Pending Editor

Reviewers:

- @Pending Reviewers

Submitted: N/A
Published: N/A

Summary

Designing neural networks is a complex task.

Several tools exist which allow analyzing neural networks after and during training. These techniques can be characterized by their focus on either data or model as well as their general abstractness. Examples for abstract model oriented techniques are tools for analyzing the sharpness of local optima (Keskar et al., 2016; Novak et al., 2018), which can be an indicator for the generalizing capeabilities of the trained models. In these scenarios the complex of dataset and model is reduced to the error surface, allowing for insights into the differences between different setups. A less abstract data-centric technique GradCam by Selvaraju et al. (Chattopadhay et al., 2018; Selvaraju et al., 2019), which reduce the model to a set of class-activation maps that can be overlayed over individual data points to get an intuitive understanding of the inference process. SVCCA (Morcos et al., 2018; Raghu et al., 2017) can be considered model centric and a middle ground in terms of abstractness, since it allows the comparative analysis on the features extracted by specific layers. SVCCA is also relevant from a functional perspective for this work, since it uses singular value decomposition as a core technique to obtain the analysis results. Another model centric tool that allows for a layer by layer analysis are logistic regression probes (Alain & Bengio, 2016), which utilize logistic regressions trained on the output of a hidden layer to measure the linear seperability of the data and thus the quality of the intermediate solution quality of a classifier model.

The latter is of great importance for this work, since Logistic Regression Probes are often used to compare models and identify the contribution of layers to overall performance (Richter, Byttner, et al., 2021; Richter, Schöning, et al., 2021; Shenk et al., 2020) and to demonstrate that the saturation metric can is capable of showing parameter-inefficienies in neural netwoork architectures.

However, the aforementioned tools have significant limitation in terms of their usefulness in practical application scenarios, where these tools are to be used to improve the performance of a given model. In case of data centric tools like GradCam the solution propagates back to the data, which makes it hard to derive decisions regarding the neural architecture. However, the biggest concern in all aforementioned tools are the cost in computational resources and the integration of the analysis into the workflow of a deep learning practicioner. Tools like SVCCA and Logistic Regression Probes require complex and computationally expensive procedures that need to be conducted after training. This naturally limits these techniques to small benchmarks and primarily academic datasets like Cifar10 (Shenk et al., 2020). A analysis tool that is to be used during the development of a deep learning based model needs to be able to be used with little computational and workflow overhead as possible. Ideally the analysis can be done life while the training is in progress, allowing the researcher to interupt potentially long running training session to improve the model. Saturation was proposed in 2018 (Shenk, 2018) and later refined (Shenk et al., 2020) and is the only known analysis technique known

[*] co-first author
[†] co-first author

to the authors that has this capability while also allowing to identify parameter-inefficiencies in the setup (Richter, Byttner, et al., 2021; Richter, Schöning, et al., 2021; Shenk et al., 2020). In order to make saturation usable an application scenario, it is necessary to provide a easy-to-use framework that allows for an integration of the tool into the normal training and inference code with only minimaly invasive changes. It is also necessary that the computation and analysis can be done online as part of the regular forward pass of the model, to make the integration as seemless as possible.

The Python package Delve provides a framework for allowing a seemless and minimal overhead integration for saturation and other statistical analysis of neural network layer eigenspaces. Delve hooks into PyTorch (Paszke et al., 2019) models and allows saving statistics via TensorBoard (Abadi et al., 2015) events or CSV writers. A comprehensive source of documentation is provided on the home page (http://delve-docs.readthedocs.io).

Statement of Need

Research on changes in neural network representations has exploded in the past years (Alain & Bengio, 2016; Montavon et al., 2010; Morcos et al., 2018; Raghu et al., 2017; Selvaraju et al., 2019; Zhou et al., 2016). Furthermore, researchers who are interested in developing novel algorithms must implement from scratch much of the computational and algorithmic infrastructure for analysis and visualization. By packaging a library that is particularly useful for extracting statistics from neural network training, future researchers can benefit from access to a high-level interface and clearly documented methods for their work.

Overview of the Library

The software is structured into several modules which distribute tasks. Full details are available at https://delve-docs.readthedocs.io/.

The TensorBoardX `SummaryWriter` (Abadi et al., 2015) is used to efficiently save artifacts like images or statistics during training with minimal interruption. A variety of layer feature statistics can be observed:

Statistic
intrinsic dimensionality
layer saturation (intrinsic dimensionality divided by feature space dimensionalit
the covariance-matrix
the determinant of the covariance matrix (also known as generalized variance)
the trace of the covariance matrix, a measure of variance of the data
the trace of the diagonal matrix, another way of measuring the dispersion of the data.
layer saturation (intrinsic dimensionality divided by feature space dimensionality)

Several layers are currently supported:

- Convolutional
- Linear
- LSTM

Additional layers such as PyTorch's ConvTranspose2D are planned for future development (Issue #43).

Eigendecomposition of the feature covariance matrix

Saturation is a measure of the rank of the layer feature eigenspace introduced by (Shenk, 2018; Shenk et al., 2019) and further explored in(Shenk et al., 2020).

Covariance matrix of features is computed online as described in (Shenk et al., 2020):

$$Q(Z_l, Z_l) = \frac{\sum_{b=0}^{B} A_{l,b}^T A_{l,b}}{n} - (\bar{A}_l \bigotimes \bar{A}_l)$$

for B batches of layer output matrix A_l and n number of samples.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., ... Zheng, X. (2015). *TensorFlow: Large-scale machine learning on heterogeneous systems*. https://www.tensorflow.org/

Alain, G., & Bengio, Y. (2016). Understanding intermediate layers using linear classifier probes. *ArXiv*, *abs/1610.01644*.

Chattopadhay, A., Sarkar, A., Howlader, P., & Balasubramanian, V. N. (2018). Grad-CAM++: Generalized gradient-based visual explanations for deep convolutional networks. *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*. https://doi.org/10.1109/wacv.2018.00097

Keskar, N. S., Mudigere, D., Nocedal, J., Smelyanskiy, M., & Tang, P. T. P. (2016). On large-batch training for deep learning: Generalization gap and sharp minima. *CoRR*, *abs/1609.04836*.

Montavon, G., Müller, K.-R., & Braun, M. L. (2010). Layer-wise analysis of deep networks with gaussian kernels. In J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, & A. Culotta (Eds.), *Advances in neural information processing systems 23* (pp. 1678–1686). Curran Associates, Inc. http://papers.nips.cc/paper/4061-layer-wise-analysis-of-deep-networks-with-gaussian-kernels.pdf

Morcos, A. S., Raghu, M., & Bengio, S. (2018). *Insights on representational similarity in neural networks with canonical correlation*. http://arxiv.org/abs/1806.05759

Novak, R., Bahri, Y., Abolafia, D. A., Pennington, J., & Sohl-Dickstein, J. (2018). Sensitivity and generalization in neural networks: An empirical study. *International Conference on Learning Representations*. https://openreview.net/forum?id=HJC2SzZCW

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., ... Chintala, S. (2019). PyTorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlché-Buc, E. Fox, & R. Garnett (Eds.), *Advances in neural information processing systems 32* (pp. 8024–8035). Curran Associates, Inc. http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

Raghu, M., Gilmer, J., Yosinski, J., & Sohl-Dickstein, J. (2017). *SVCCA: Singular vector canonical correlation analysis for deep learning dynamics and interpretability*. http://arxiv.org/abs/1706.05806

Richter, M. L., Byttner, W., Krumnack, U., Schallner, L., & Shenk, J. (2021). Size matters. *CoRR*, *abs/2102.01582*. https://arxiv.org/abs/2102.01582

Richter, M. L., Schöning, J., & Krumnack, U. (2021). Should you go deeper? Optimizing convolutional neural network architectures without training by receptive field analysis. *CoRR*, *abs/2106.12307*. https://arxiv.org/abs/2106.12307

Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., & Batra, D. (2019). Grad-CAM: Visual explanations from deep networks via gradient-based localization. *International Journal of Computer Vision*, *128*(2), 336–359. https://doi.org/10.1007/s11263-019-01228-7

Shenk, J. (2018). *Spectral Decomposition for Live Guidance of Neural Network Architecture Design* [Master's Thesis]. University of Osnabrück.

Shenk, J., Richter, M. L., Arpteg, A., & Huss, M. (2019). Spectral analysis of latent representations. *CoRR*, *abs/1907.08589*. http://arxiv.org/abs/1907.08589

Shenk, J., Richter, M. L., Byttner, W., Arpteg, A., & Huss, M. (2020). Feature space saturation during training. *CoRR*, *abs/2006.08679*. https://arxiv.org/abs/2006.08679

Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., & Torralba, A. (2016). Learning deep features for discriminative localization. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2921–2929.