MLVis: Machine Learning Methods in Visualisation for Big Data (2019)
D. Archambault, I. Nabney and J. Peltonen (Editors)

Visual Ensemble Analysis to Study the Influence of Hyper-parameters on Training Deep Neural Networks

Sagad Hamid, Adrian Derstroff, Sören Klemm, Quynh Quang Ngo, Xiaoyi Jiang, Lars Linsen

Westfälische Wilhelms-Universität Münster, Germany

Abstract

A good deep neural network design allows for efficient training and high accuracy. The training step requires a suitable choice of several hyper-parameters. Limited knowledge exists on how the hyper-parameters impact the training process, what is the interplay of multiple hyper-parameters, and what is the interrelation of hyper-parameters and network topology. In this paper, we present a structured analysis towards these goals by investigating an ensemble of training runs. We propose a visual ensemble analysis based on hyper-parameter space visualizations, performance visualizations, and visualizing correlations of topological structures. As a proof of concept, we apply our approach to deep convolutional neural networks.

1. Introduction

Successful training of a Neural Network (NN) for a specific machine learning task essentially comprises two steps. First, a suitable NN architecture has to be identified. Second, a set of hyper-parameters for training this architecture, given the available data, has to be found. This is an iterative process of training a specific architecture and evaluating its performance after training with a set of hyper-parameters. Common evaluation criteria are, among others, the final accuracy on a previously unseen testing data set and training time needed to achieve a certain output-quality threshold.

While the tuning of NN parameters (i.e., the weights) is done automatically using gradient-descent methods, finding correct hyper-parameters is a computationally expensive and cumbersome task, especially for deep NN [GSK*17]. Despite new developments in hyper-parameter tuning (HPT) strategies, it usually boils down to train the given NN architecture using the hyper-parameter set proposed for that architecture and to compare the performance of the different settings [HKV18]. To reduce the effort, predictive early stopping criteria are desired. These criteria allow for a premature abort of a training run, if successful training is unlikely.

The impact of the hyper-parameter choices on the training process of a NN is not fully understood yet. The main goals are to observe (G1) how the choices affect the performance of the NN, (G2) how multiple hyper-parameters influence each other during the training process, and (G3) how topological structures change their behavior when changing the hyper-parameter values.

In this paper, we propose a structured analysis towards these goals by visually analyzing an ensemble of training runs with different hyper-parameter settings. We propose performance measures for accuracy and efficiency to visually analyze their evolution during the training process, see Sec. 4.1. Moreover, we propose correlation measures of topological structures such as filters and layers to visually analyze their behavior, see Sec. 4.2. Coordinated views of hyper-parameter space visualizations, performance visualizations, and correlation visualizations allow for the interactive visual analysis of the influence of hyper-parameters on the NN's performance during the training process and on the behavior of topological structures. As a proof of concept, we apply our approach to analyzing the training hyper-parameters *learning rate* and *momentum* of deep convolutional NNs using the well-known examples *MNIST* and *CIFAR-10* and present respective findings, see Sec. 5.

2. Related Work

Visualization of NN training process. Designing an optimal training model is one of the most important tasks in the field of deep learning [PHvG*18] and wrong decisions are irreversible [BCV13]. Recently, visual tools that support designing and debugging a training model have been proposed. A recent survey provides a good overview of the various endeavors [HKPC18]. However, none of these approaches fully address goals (G1-G3) listed above.

Visual performance analysis. Computing accuracy or loss, respectively, of a NN is a standard approach to judge the quality of a NN. Recently, a visualization approach that displays the loss function in the form of a landscape (using two random directions) was proposed to discover the relationship between network structure and the loss landscape [LXT*18]. Hyper-parameters are only briefly discussed. We, instead, perform a structured analysis using an ensemble.

Visual tools for topology optimization. A number of approaches focus on the optimization of the network structure [RSLMR17,

DOI: 10.2312/mlvis.20191160

RFFT17], where the correlation or similarity of filter activations is analyzed. A progressive optimization approach was proposed by Pezzotti et al. [PHvG*18], where they observe correlation or activation of filters to provide a hint whether topological structures shall be enhanced or may be reduced [PHvG*18]. We build upon this idea by also computing correlations between filters and even enhance the idea to larger topological structures in the form of layers. Moreover, we analyze ensembles of training runs, not only a single run.

Effect of hyper-parameters. Concepts for theoretical searches for optimal hyper-parameter settings for training NN exist [BB12, CDM15]. The authors state the issue of automatic searches for hyper-parameters of training algorithms to prevent a predefined structural model from being over- or under-fitted after training. Breuel [Bre15] investigated the effect of training hyper-parameters on accuracy and loss. We, instead, investigate the performance during the evolution of the training process. If the quality of training runs can be judged early on, this would reduce substantially the computational burden.

3. Background

The current gold standard of fitting NN parameters (i.e., weights and biases) to a specific problem is stochastic gradient descent (SGD). Labeled data samples are propagated through the network and a scalar measure of difference between ground truth and network output, called *loss*, is calculated. SGD calculates partial derivatives of the loss with respect to every parameter of the network, called gradients. After each training step, the weights are updated according to the calculated gradients. The *learning rate* defines the length of those update steps. While a large learning rate causes larger steps and might lead to quicker convergence, it can also cause overshooting effects and hence slow down convergence or, in the worst case, cause divergence. A further hyper-parameter, called *momentum*, is used to partially overcome this problem by calculating the average of the last update steps and weighting them with an exponential decay. While loss is used to train the network, *accuracy* is a measure often used to evaluate performance on balanced data sets, as it is easier to interpret. Accuracy is calculated as the ratio of correctly assigned labels across all classes. In general, loss and accuracy are negatively correlated.

When classifying image data, convolutional neural networks (CNNs) are commonly used, which exploit the fact that neighbouring pixels are often highly correlated and belong to the same object. To reduce the complexity of the model to be trained, neurons are aggregated to *filters*, i.e., patches of weights, which are convolved over the input. Every convolutional layer comprises a set of filters of equal size, which are evaluated in parallel and extract image features. Multiple layers can be applied consecutively to increase the complexity of image features learned by the NN. For a given number of filters, the information contained in the output of a layer is maximal, if the filter responses do not correlate with each other, i.e., every filter detects a different property. To find an optimal NN, we are hence looking for uncorrelated filters in every layer.

4. Visual Ensemble Analysis

To allow for the analysis of the influence of hyper-parameters on the NN training process, we generate an ensemble of training runs by sampling the hyper-parameter space. Targeting goals (G1)-(G3) listed in Sec. 1, we developed visual analysis methods of the ensemble addressing the influence of the hyper-parameter choices on the NN performance (including the interplay of multiple hyper-parameters), see Sec. 4.1, and on the behavior of topological structures, see Sec. 4.2.

4.1. Performance Analysis

The performance of a NN is judged by the quality of the output and the efficiency of the training process. To judge the latter, we need to investigate the evolution of quality during the training process. While loss and accuracy judge the NN quality at a fixed point in time during or after training, we capture the evolution by introducing three **performance measures**: (P1) Given a point in time, we compute the accuracy at that time. (P2) Given a time interval, we compute the maximum accuracy within the time interval. (P3) Given a time interval, we compute the standard deviation from the mean accuracy within the time interval. The latter is normalized to range $[0,1]$, where low standard deviation (meaning high stability) is set to high values. Thus, all three performance measures are within the range $[0,1]$ with high values meaning better performance. The training times used for computing the measures can be chosen interactively to investigate different time intervals. For example, we can use (P2) to judge which training runs produce high accuracy early on in the training process and we can use (P3) to judge how stable the training process is during that time interval.

To investigate how hyper-parameters affect the performance (Goal (G1)), we have to simultaneously visualize all hyper-parameters and all performance measures, which leads to a multi-dimensional data visualization task. Since our aim is to observe correlations between the dimensions in addition to having a visual representation that scales well with the number of dimensions, a **Parallel Coordinate Plot** (PCP) is the most suitable choice. The first axes represent the hyper-parameters, which the later axes represent the performance measures. Brushing for interactive selection allows for filtering and highlighting training runs with specific properties like low/high performance with respect to some performance measures or low/high hyper-parameter values, see Figure 1(a). Moreover, the user can restrict the considered values for performance measures to certain intervals. The values outside the chosen intervals are then clamped to the minimum and maximum respectively. A common visual representation for NN performance is to show a graph that plots loss and accuracy as a function of the training time. It allows for observing the convergence speed of the training process. We also support such a **Loss-accuracy Graph** and link it to the PCP. Thus, only the training runs selected in the PCP are displayed in the graph. Multiple selections using different colors are possible, see Figure 1(c).

Goal (G2) postulated to investigate what the interplay of different hyper-parameters for the performance is. To provide a respective overview of all pairwise hyper-parameter combinations, we propose a hyper-parameter space visualization. Using a grid-based sampling of the parameter space, the performance of all

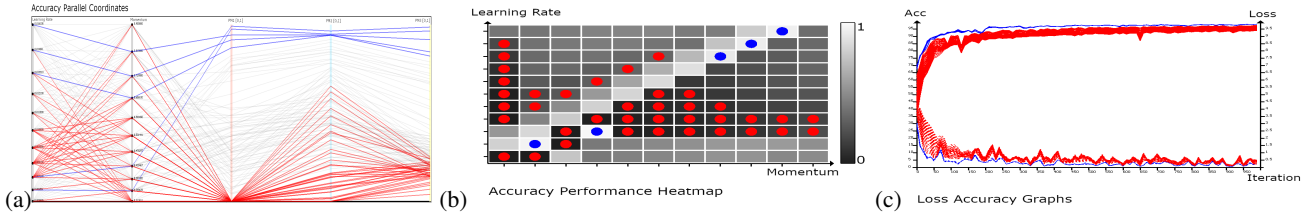

Figure 1: *Performance analysis of MNIST data set using coordinated views: (a) PCP for analyzing correlation of hyper-parameters (axes 1-2) and performance measures (axes 3-5). Brushing on third axis is used for selecting training runs with low (red) and high accuracy (blue). (b) Hyper-parameter space visualization via Performance Heat map allows for analyzing the interplay of hyper-parameters. PCP selections are shown as colored dots. The brightness of the cells indicates the accumulated accuracy performances from the PCP. (c) Loss-accuracy Graph shows evolution for selected runs (red and blue). Accuracy curves (top) are solid while loss curves (bottom) are stippled.*

pairwise hyper-parameter combinations can be visualized using a **Performance Heat Map**, see Figure 1(b). A selected performance measure or the average of multiple selected performance measures is color-coded using a luminance color map. Individual hyper-parameter value pairs can be selected by clicking at the respective cells of the heat map and are highlighted by colored dots. Interactions with the Performance Heat Map are coordinated with the other views (PCP and Loss-accuracy Graph). For example, the highlighted cells in Figure 1(b) correspond to the training runs selected in Figure 1(a).

4.2. Analyzing Topological Structures

As mentioned in Sec. 3, high correlations among filters in a CNN indicate redundancy and hint to a possible reduction of a layer. However, it has not been examined systematically yet, how hyper-parameters influence such correlations of topological structures (Goal (G3)).

Given a time step of training run, we can compute pairwise correlations among the activation vectors of all filters of a layer using the Pearson correlation coefficient [BCHC09]. These pairwise correlations can be stored in a (symmetric) correlation matrix. Since correlation and anti-correlation may equally reflect redundancies, we use its magnitude. The correlation matrix can be visualized using a **Filter Correlation Heat Map** representation using a luminance color map, see Figures 2(b,c). To visually convey clusters of correlated filters, we re-order the matrices using a hierarchical clustering approach [ZKF05]. The hierarchical clustering delivers a dendrogram that clusters filters of high correlation magnitude, but does not impose a unique order yet. We enforce a unique order by sorting the clusters by their size in decreasing order, which will be of utmost importance for subsequent steps (see below).

The Filter Correlation Heat Map only reflects a single time of a single run. To visually compare the correlations in multiple training runs and convey the evolution during the training process, we compute the number of high correlations of each time step of each training run and plot one graph per run in a **Filter Correlation Graph**, see Figure 2(d). The number of high correlations is estimated by comparing against a user-defined threshold.

Plotting the correlation graphs of all training runs leads to visual clutter. Hence, we need a means to select runs of interest. We propose to compute pairwise similarities among different simulation runs and visually encode the similarities in a similarity plot. We propose a novel **similarity measure for training runs**. The similarity is computed by interpreting the correlation matrix as a vector and computing the Pearson correlation of the vectors. This only works, if the vectors store corresponding structures of different runs at the same entries. In NN, however, filters may take over different roles in different training runs. Hence, using the order of the filters in the layer would not work. However, when using the order of filters based on hierarchical clustering as described above, we have comparable structures at same locations within the vectors and computing correlations is meaningful.

Given the correlation matrix of all training runs for a selected point in time, we want to visualize the correlations in an **Ensemble Similarity Plot**. Such similarity matrices can be used to generate similarity plots in the form of scatterplots in a 2D embedding, see Figure 2(a). Different embeddings exist. Since we are interested in conveying the correlations (or similarities), the embedding should reflect dissimilarities by distances. This is the objective function of a Multidimensional Scaling (MDS) embedding [BG05]. Alternatively, we also support t-SNE embeddings [vdMH08], which preserve neighborhoods. We enhance the scatterplot by assigning to each point a color to indicate the hyper-parameter settings of each training run. For a 2D hyper-parameter space, we propose to use the a^* and b^* channels of the CIE $L^*a^*b^*$ color space to map hyper-parameter settings to colors as shown in the legend of Figure 2(a).

The Ensemble Similarity Plot provides an overview over all training runs in the ensemble. Interesting runs can be selected interactively (highlighted by increased point size) and analyzed in more detail in the coordinated views of the Filter Correlation Graph and the Filter Correlation Heat Map.

5. Application Scenarios

We applied our visual ensemble analysis tool in two scenarios with CNNs using the well-known MNIST and CIFAR data sets. The MNIST data set contains 2 convolutional layers, where the first layer (Conv1) consists of 20 filters and the second layer (Conv2) of 50 filters. The CIFAR-10 data set contains 3 convolutional layers with 32 (Conv1), 32 (Conv2), and 64 (Conv3) filters. As hyper-parameters we considered learning rate and momentum. To generate training ensembles with 121 runs, the hyper-parameter space is

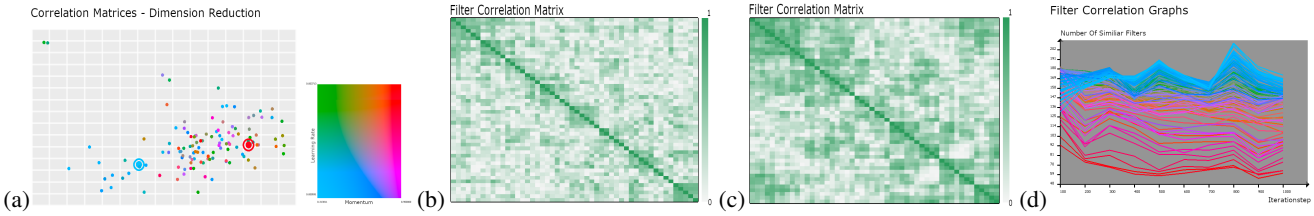

Figure 2: *Analyzing hyper-parameter influence on topological structures for second layer (Conv2) of MNIST data set using coordinated views: (a) Ensemble Similarity Plot, which is the result of MDS, provides an overview over all training runs for time step 500 by applying a 2D color map to hyper-parameter space. (b) Filter Correlation Heat Map of red selection in (a) exhibits low correlations. (c) Filter Correlation Heat Map of cyan selection in (a) exhibits high correlations. Color in both Heat Maps indicates the absolute Pearson correlation coefficient between two filters. (d) Filter Correlation Graph shows the evolution of correlations during training.*

sampled on a 2D grid. For learning rate we used values $0.01 \cdot 0.9^i$ for $i = 0,\ldots,5$ and $0.01 \cdot 1.1^i$ for $i = 1,\ldots,5$, while for momentum we used values 0.9^i for $i = 1,\ldots,11$. We computed $1,000$ (MNIST) and $4,000$ (CIFAR-10) training steps for each run.

MNIST. We start the MNIST data set analysis by considering the entire ensemble to compute performance measures and visualize them in the PCP. Figure 1(a) represents hyper-parameters learning rate and momentum in the first two axes and performance measures (P1)-(P3) in the last three axes. For the performance measures, we picked time step 100 for (P1) and time interval $[100,600]$ for (P2) and (P3). We interactively adjust the displayed accuracy range for (P1) to $[0.85,0.93]$ and for (P2) to $[0.93,0.98]$, while the displayed standard deviation is set to $[1.5,3]$ for the full range of observed values. We brush on the (P1) axis to select high (green) and low values (red). We observe that there is a high correlation in the performance measures for the selections (green lines indicate high values for all measures, red lines low values). However, we do not see a clear pattern in the hyper-parameter axes. To get a better understanding of the distribution of performance values for different hyper-parameter settings, we use the coordinated hyper-parameter space view of the Performance Heat Map using (P1) and (P2), see Figure 1(b). The selections made in the PCP are shown as colored dots. The green dots appear in the brightest cells and the red dots in the darkest cells. We observe that there is a clear 2D pattern, where best performances are showing up as a bright diagonal stripe. Hence, we conclude that the two hyper-parameters are strongly inter-dependent and the hyper-parameters cannot be optimized separately. We also observe that there is no monotonic trend in the heat map.Finally, we also wanted to see the evolution of the accuracy during the training process. Figure 1(c) shows the Loss-accuracy Graph for the selected runs.The selected runs with higher accuracy (and lower loss) have higher accuracy (and lower loss) throughout the training process. This is an important finding, as we can judge the quality early on in the training process.

Next, we want to investigate how topological structures are affected by the hyper-parameter choices. We choose time step 500 of the second layer (Conv2) and generate an overview of all training runs using the Ensemble Similarity Plot based on MDS, see Figure 2(a). We observe some green outliers to the upper left and a cyan cluster to the lower left, but other colors seem to be rather mixed. To understand how the training runs differ, we pick one of the runs from the cyan cluster on the left (learning rate 0.005905, momentum 0.313811) and one of the runs to the right (red, learning rate 0.016105, momentum 0.9). Figures 2(b) and (c) show the corresponding Filter Correlation Heat Maps.The heat maps exhibit stronger filter correlations for the red dot in (b). Finally, we want to analyze whether there is a global pattern during the evolution of the training process. We use the Filter Correlation Graph displaying the number of filters with correlations above threshold 0.7 in Figure 2(d). We observe that, after an initial phase, the training runs with low values in both hyper-parameters (cyan) have the largest filter correlations, while the runs with high values in both hyper-parameters (red) have lowest filter correlations. We also observe that the colors in the graph mix less with increasing training times. Hence, we conclude that the hyper-parameters, indeed, influence the correlation of topological structures.

CIFAR-10. We perform a similarly structured analysis for the CIFAR-10 data set to investigate whether we can make similar observations, see video and appendix. For performance analysis we make similar observations as for the MNIST data set. One striking and surprising difference is that the high accuracy values exhibit an anti-correlation to the standard deviation values. Another surprising observation is that the observed pattern of the Performance Heat Map is strikingly similar to the one for the MNIST data set. When analyzing the influence of the hyper-parameters on topological structures, we immediately see smooth color transition.When compared to MNIST, filter correlations are generally lower for CIFAR-10, which may indicate lower redundancy. The Filter Correlation Graph computed for threshold 0.7 shows that the training runs colored red have low correlations after an initial phase, while the ones colored cyan exhibit high correlations until very late in the training process.

6. Discussion and Conclusion

In this paper, we shed light on how hyper-parameters influence the training process by visually analyzing a training ensemble. We proposed suitable visual encodings in coordinated views to address the formulated analysis goals. The visual encodings scale to higher numbers of hyper-parameters, performance measures, filters, and training runs. Only the 2D color map and the parameter-space visualization assumed two hyper-parameters. With higher number of hyper-parameters, one would use multiple Performance Heat Maps, one for each hyper-parameter pair, to analyze pairwise correlations.

References

[BB12] BERGSTRA J., BENGIO Y.: Random search for hyper-parameter optimization. *Journal of Machine Learning Research 13*, Feb (2012), 281–305. 2

[BCHC09] BENESTY J., CHEN J., HUANG Y., COHEN I.: Pearson correlation coefficient. In *Noise reduction in speech processing*. Springer, 2009, pp. 1–4. 3

[BCV13] BENGIO Y., COURVILLE A., VINCENT P.: Representation learning: A review and new perspectives. *IEEE Trans. Pattern Anal. Mach. Intell. 35*, 8 (Aug. 2013), 1798–1828. URL: http://dx.doi.org/10.1109/TPAMI.2013.50, doi:10.1109/TPAMI.2013.50. 1

[BG05] BORG I., GROENEN P. J.: *Modern multidimensional scaling: Theory and applications*. Springer Science & Business Media, 2005. 3

[Bre15] BREUEL T. M.: The effects of hyperparameters on sgd training of neural networks. *arXiv preprint arXiv:1508.02788* (2015). 2

[CDM15] CLAESEN M., DE MOOR B.: Hyperparameter search in machine learning. *arXiv preprint arXiv:1502.02127* (2015). 2

[GSK*17] GREFF K., SRIVASTAVA R. K., KOUTNÍK J., STEUNEBRINK B. R., SCHMIDHUBER J.: LSTM: A search space odyssey. *IEEE Trans. Neural Netw. Learning Syst. 28*, 10 (2017), 2222–2232. doi:10.1109/TNNLS.2016.2582924. 1

[HKPC18] HOHMAN F. M., KAHNG M., PIENTA R., CHAU D. H.: Visual analytics in deep learning: An interrogative survey for the next frontiers. *IEEE Transactions on Visualization and Computer Graphics* (2018), 1–1. doi:10.1109/TVCG.2018.2843369. 1

[HKV18] HUTTER F., KOTTHOFF L., VANSCHOREN J. (Eds.): *Automatic Machine Learning: Methods, Systems, Challenges*. Springer, 2018. In press, available at http://automl.org/book. 1

[LXT*18] LI H., XU Z., TAYLOR G., STUDER C., GOLDSTEIN T.: Visualizing the loss landscape of neural nets. In *Advances in Neural Information Processing Systems 31*, Bengio S., Wallach H., Larochelle H., Grauman K., Cesa-Bianchi N., Garnett R., (Eds.). Curran Associates, Inc., 2018, pp. 6389–6399. URL: http://papers.nips.cc/paper/7875-visualizing-the-loss-landscape-of-neural-nets.pdf. 1

[PHvG*18] PEZZOTTI N., HÖLLT T., VAN GEMERT J., LELIEVELDT B., EISEMANN E., VILANOVA A.: Deepeyes: Progressive visual analytics for designing deep neural networks. *IEEE Transactions on Visualization and Computer Graphics (Proceedings of IEEE VAST 2017) 24*, 1 (2018), 98 – 108. doi:10.1109/TVCG.2017.2744358. 1, 2

[RFFT17] RAUBER P. E., FADEL S. G., FALCAO A. X., TELEA A. C.: Visualizing the hidden activity of artificial neural networks. *IEEE transactions on visualization and computer graphics 23*, 1 (2017), 101–110. 1

[RSLMR17] ROYCHOWDHURY A., SHARMA P., LEARNED-MILLER E., ROY A.: Reducing duplicate filters in deep neural networks. In *NIPS workshop on Deep Learning: Bridging Theory and Practice* (2017), vol. 1. 1

[vdMH08] VAN DER MAATEN L., HINTON G.: Visualizing data using t-SNE. *Journal of Machine Learning Research 9* (2008), 2579–2605. URL: http://www.jmlr.org/papers/v9/vandermaaten08a.html. 3

[ZKF05] ZHAO Y., KARYPIS G., FAYYAD U.: Hierarchical clustering algorithms for document datasets. *Data mining and knowledge discovery 10*, 2 (2005), 141–168. 3