Problem Statement

We consider the regression problem of predicting an target variable $y \in \mathbb{R}$ given an object $\mathbf{x}\mathbb{R}^n$. The goal is to build a model f which outcomes a prediction for each input object. We assume that the function $f(\mathbf{x}|\mathbf{w})$ is a feed-forward dense neural network with 1 hidden, and output layer. The hidden layer dimension is denoted by h. The model output is

$$f(\mathbf{x}|\mathbf{w}) = \sigma_2\left(\mathbf{W}_2\sigma_1(\mathbf{W}_1\mathbf{x})\right),$$

where $\mathbf{W}_1 \in \mathbb{R}^{h\times n}$, $\mathbf{W}_2 \in \mathbb{R}^{1\times h}$ are weight matrices, σ_1, σ_2 are activation functions applied to each input component. We omitted the bias terms for simplicity. Let denote by $\mathbf{w} = (\mathrm{vec}(\mathbf{W}_1), \mathrm{vec}(\mathbf{W}_2)) \in \mathbb{R}^p$ a joint weight vector, where $p = h(n+1)$ is the total number of weights.

Let assume that we are given the design matrix $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_m]^\intercal \in \mathbb{R}^{m\times n}$ and the target vector $\mathbf{y} = [y_1, \ldots, y_m]^\intercal \in \mathbb{R}^m$. Each i-th matrix $\mathbf{X}$ row represents an object which is associated with the i-th vector $\mathbf{y}$ element. The goal is to find the optimal weight vector $\mathbf{w}^*$.

The weights $\mathbf{w}$ are fitted by the minimization of an error function:

$$\mathbf{w}^* = \arg\min_{\mathbf{w}\in\mathbb{R}^r} S(\mathbf{w}|\mathbf{X}, \mathbf{y}, f). \tag{1}$$

The most common choice for the error function $S(\mathbf{w}|\mathbf{X}, \mathbf{y}, f)$ is the squared error between real target values and predicted ones

$$S(\mathbf{w}|\mathbf{X}, \mathbf{y}, f) = \|\mathbf{y} - \mathbf{f}(\mathbf{X}|\mathbf{w})\|_2^2 = \sum_{i=1}^{m}(y_i - f(\mathbf{x}_i|\mathbf{w}))^2.$$

The problem 1 could be solved by any neural network optimization method [].

The number of model weights p could be extremely huge. In this case the solution of the problem 1 leads to overfitting. To eliminate this problem we propose to select the subset $\mathcal{A} \subseteq \{1, \ldots, p\}$ of the active weights. The weights which are not active are supposed to be zero. To choose the subset $\mathcal{A}$ from all possible 2^p combinations let introduce a quality criteria $Q(\mathcal{A}|\mathbf{X}, \mathbf{y}, f)$. This function evaluates the quality of a particular active set $\mathcal{A}$. We desire to find the optimal subset $\mathcal{A}^*$ which minimize the function

$$\mathcal{A}^* = \arg\min_{\mathcal{A}\subseteq\{1,\ldots,p\}} Q(\mathcal{A}|\mathbf{X}, \mathbf{y}, f). \tag{2}$$

If the solution $\mathcal{A}^*$ of the 2 is given the next step is to determine the optimal model weights $\mathbf{w}^*$ by solving a problem

$$\mathbf{w}^* = \arg\min_{\mathbf{w}\in\mathbb{R}^r} S(\mathbf{w}|\mathbf{X}, \mathbf{y}, f), \quad \text{subject to } w_j = 0 \text{ for } j \notin \mathcal{A}^*. \tag{3}$$

QPFS

To find the optimal subset $\mathcal{A}^*$ we suggest to use QPFS algorithm. The algorithm solves the linear regression problem, where the model $f(\mathbf{x}|\mathbf{w}) = \mathbf{x}^\intercal\mathbf{w}$

$$S(\mathbf{w}|\mathbf{X}, \mathbf{y}, f) = \|\mathbf{y} - \mathbf{X}\mathbf{w}\|_2^2 \to \min_{\mathbf{w}\in\mathbb{R}^n}.$$

The goal of the QPFS is to select not correlated features which are relevant to target vector. To formalise this approach let introduce two functions: Sim and Rel. The former measures the redundancy between features, the latter contains relevances between each feature and target vector. We want to minimize the Sim function and maximize the Rel simultaneously.

The QPFS method offers the explicit way to construct the functions Sim and Rel. The method minimizes the following functional

$$\underbrace{\mathbf{z}^\top \mathbf{Q}\mathbf{z}}_{\text{Sim}} - \alpha \cdot \underbrace{\mathbf{b}^\top \mathbf{z}}_{\text{Rel}} \to \min_{\mathbf{z}\in[0,1]^n}. \tag{4}$$

This functional is an analogue of the described quality criteria 2. The first term is associated with the Sim function and the second with the Rel. The matrix $\mathbf{Q} \in \mathbb{R}^{n\times n}$ entries measure the pairwise similarities between features. The vector $\mathbf{b} \in \mathbb{R}^n$ expresses the similarities between each feature and the target vector $\mathbf{y}$. The normalized vector $\mathbf{z}$ shows the importance of each feature. This approach penalizes the dependent features and encourages features relevant to the target. The parameter α allows to control the trade-off of the Sim and the Rel terms. To find the optimal feature subset the thresholding for $\mathbf{z}$ is applied

$$j \in \mathcal{A} \Leftrightarrow z_j > \tau.$$

To measure similarity it was proposed to use the absolute value of sample Pearson correlation coefficient or sample mutual information. The problem 4 is convex if the matrix $\mathbf{Q}$ is positive semidefinite. In general it is not always true. To satisfy this condition we replace this matrix by $\mathbf{Q} - \lambda_{\min}\mathbf{I}$, where $\lambda_{\min}$ is a minimal eigenvalue of $\mathbf{Q}$.

Model linearization

Let assume that we have the model $f(\mathbf{x}|\mathbf{w})$ and we want to find the new weights by adding the updates $\Delta\mathbf{w}$ to the existing weights $\mathbf{w}$. Similar to Levenberg-Marquardt algorithm, we use the linear approximation of the model

$$\mathbf{f}(\mathbf{X}|\mathbf{w} + \Delta\mathbf{w}) \approx \mathbf{f}(\mathbf{X}|\mathbf{w}) + \mathbf{J} \cdot \Delta\mathbf{w}$$

where $\mathbf{J} \in \mathbb{R}^{m\times p}$ is a Jacobian matrix

$$\mathbf{J} = \begin{pmatrix} \frac{\partial f(\mathbf{x}_1|\mathbf{w})}{\partial w_1} & \dots & \frac{\partial f(\mathbf{x}_1|\mathbf{w})}{\partial w_p} \\ \dots & \dots & \dots \\ \frac{\partial f(\mathbf{x}_m|\mathbf{w})}{\partial w_1} & \dots & \frac{\partial f(\mathbf{x}_m|\mathbf{w})}{\partial w_p} \end{pmatrix}. \tag{5}$$

In this case the problem 1 is a linear regression problem with the target vector $\mathbf{y} - \mathbf{f}(\mathbf{X}|\mathbf{w})$, matrix $\mathbf{J}$ and weights $\Delta\mathbf{w}$

$$S(\mathbf{w} + \Delta\mathbf{w}|\mathbf{X}, \mathbf{y}, f) = \|\mathbf{y} - \mathbf{f}(\mathbf{X}|\mathbf{w} + \Delta\mathbf{w})\|_2^2 \approx \|(\mathbf{y} - \mathbf{f}(\mathbf{X}|\mathbf{w})) - \mathbf{J} \cdot \Delta\mathbf{w}\|_2^2.$$

To find the most significant set of weights for the current point $\mathbf{w}$ we can apply QPFS algorithm. Significance means the relevance of the weight update to the residual vector and pairwise weights updates independence through training data.

We use backpropagation procedure to update the network weights. The most of the neural network optimization methods use the gradient of the model to update network weights. It allows to get the Jacobian matrix **J** for free from optimization process.

Experiment

In the experiment we used the Boston House Pricing dataset (objects: 506, features: 13).

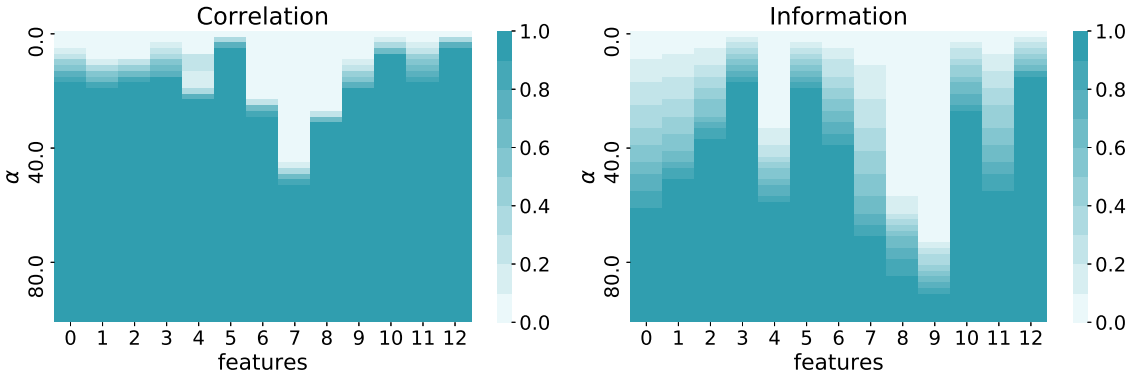

Figure 1: QPFS feature scores with respect to α coefficient

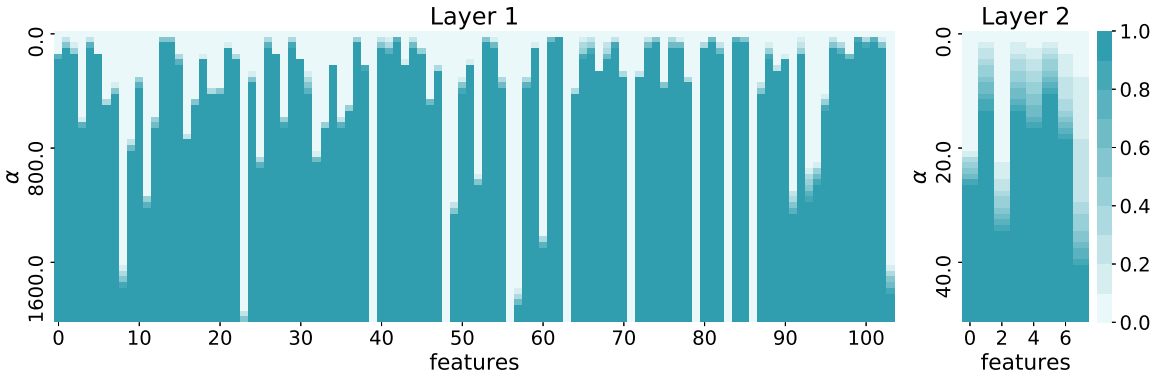

Figure 2: Neural networks weight scores with respect to α coefficient for correlation similarity measure

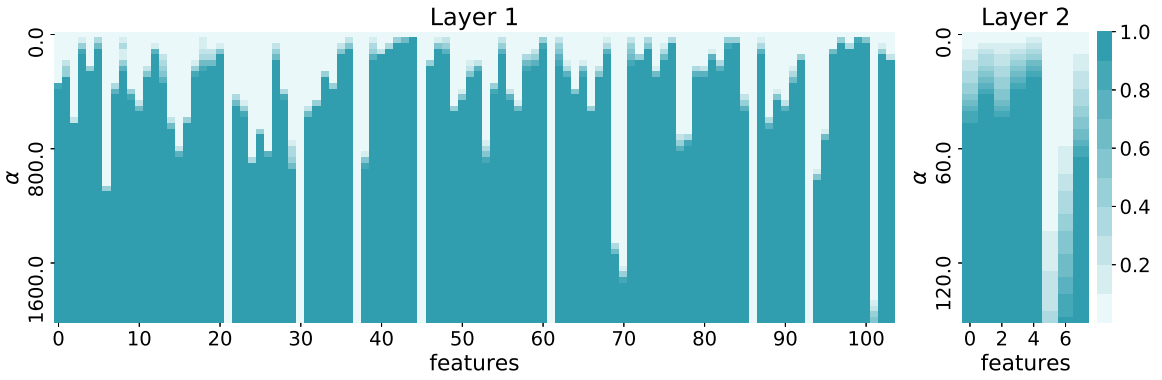

Figure 3: Neural networks weight scores with respect to α coefficient for mutual information similarity measure

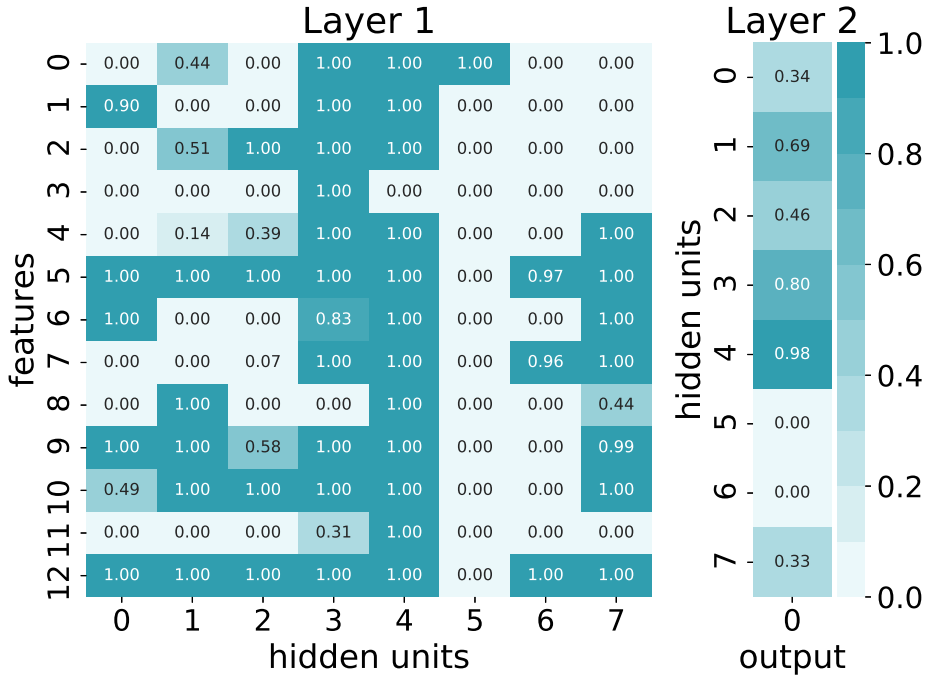

Figure 4: Neural network weight scores maps for mutual information similarity coefficient

---

OLD VERSION

Model weights selection

We propose the algorithm that extends the quadratic programming methodology to the case of model weights selection. Let suppose that the weights from different layers do not interact. This assumption allows to solve the model weights selection problem separately for each layer. We introduce two vectors $\mathbf{z}_1 \in \mathbb{R}_+^{hn}$, $\|\mathbf{z}_1\|_1 = 1$ and

$\mathbf{z}_2 \in \mathbb{R}_+^{rh}, \|\mathbf{z}_2\|_1 = 1$ for matrices $\mathbf{W}_1$ and $\mathbf{W}_2$ respectively. The components of these vectors express the importance of each individual weight. Let introduce the soft version of the problem 2

$$\mathbf{z}_1^* = \underset{\mathbf{z}_1 \in \mathbb{R}_+^{hn} \|\mathbf{z}_1\|_1 = 1}{\arg\min} Q(\mathbf{z}_1 | \mathbf{X}, \mathbf{y}, f) \tag{6}$$

$$\mathbf{z}_2^* = \underset{\mathbf{z}_2 \in \mathbb{R}_+^{h} \|\mathbf{z}_2\|_1 = 1}{\arg\min} Q(\mathbf{z}_2 | \mathbf{X}, \mathbf{y}, f). \tag{7}$$

Since the vectors $\mathbf{z}_1$ and $\mathbf{z}_2$ are obtained, we have to recover the optimal active subset $\mathcal{A}^*$. Thresholds d_1 and d_2 are tuned for both layers. If the weight importance larger than corresponding threshold the weight is assumed to be active.

Similarly to the authors of QPFS we use the quadratic quality criteria for each layer

$$Q(\mathbf{z} | \mathbf{X}, \mathbf{y}, f) = (1 - \alpha)\mathbf{z}^\intercal \mathbf{Q} \mathbf{z} - \alpha \mathbf{b}^\intercal \mathbf{z}. \tag{8}$$

The matrix $\mathbf{Q}$ entries estimate the pairwise interactions between weights. The vector $\mathbf{b}$ estimates the influence of each weight to the target variable.

Weight interactions

To estimate the interactions between weights we measure similarity function $s(\cdot, \cdot)$ between input neurons for these weights. We use correlation coefficient and mutual information as similarity function. Interactions between weights $\mathbf{W}_1$ are measured by similarity between input features. Interactions between weights $\mathbf{W}_2$ are measured by similarity between hidden layer neurons. Since each neuron from one layer is connected to each neuron from the next layer, the interactions between weights from one neuron are the same. The total number of distinct element in matrices $\mathbf{Q}$ will be n^2 and h^2 for both layer respectively.

Weight relevances

The vector $\mathbf{b}$ entries estimate the influence of the weights to the target variable. Let approximate the function $f(\mathbf{x}|\mathbf{w})$ at the point $\mathbf{w}$ by its linearization

$$\mathbf{f}(\mathbf{X}|\mathbf{w} + \Delta\mathbf{w}) \approx \mathbf{f}(\mathbf{X}|\mathbf{w}) + \mathbf{J} \cdot \Delta\mathbf{w}, \tag{9}$$

where $\mathbf{J} \in \mathbb{R}^{m \times r}$ is a Jacobian matrix

$$\mathbf{J} = \begin{pmatrix} \frac{\partial f(\mathbf{x}_1|\mathbf{w})}{\partial w_1} & \cdots & \frac{\partial f(\mathbf{x}_1|\mathbf{w})}{\partial w_r} \\ \cdots & \cdots & \cdots \\ \frac{\partial f(\mathbf{x}_m|\mathbf{w})}{\partial w_1} & \cdots & \frac{\partial f(\mathbf{x}_m|\mathbf{w})}{\partial w_r} \end{pmatrix}. \tag{10}$$

The elements of the vector $\mathbf{b}$ equal the similarity function $s(\cdot, \cdot)$ between the corresponding column of the matrix $\mathbf{J}$ and the target vector $\mathbf{y}$.

1 Final algorithm

Let assume that we already have the solution $\mathbf{w}^*$ of problem 1. The next step is to determine the matrices $\mathbf{Q}$ and the vectors $\mathbf{b}$ for each layer. The way to construct them is described in previous section. We solve the problems 6, 7 to get the weights importances $\mathbf{z}_1$ and $\mathbf{z}_2$. Thresholding of these vectors gives the active set $\mathcal{A}$.

To summarize the described method we derive the pseudocode for the final algorithm 1.

Algorithm 1

Require: $\mathbf{X}, \mathbf{y}$;
Ensure: $\mathcal{A}, \mathbf{w}$;
1: $\mathcal{A}^0 = \{1, \dots, p\}$
2: $\mathbf{w}^* = \arg\min_{\mathbf{w} \in \mathbb{R}^p} S(\mathbf{w}|\mathbf{X}, \mathbf{y}, f)$
3: Compute $\mathbf{Q}$'s and $\mathbf{b}$'s
4: $\mathbf{z}_1^* = \arg\min_{\mathbf{z}_1 \in \mathbb{R}_+^{hn} \|\mathbf{z}_1\|_1 = 1} Q(\mathbf{z}_1|\mathbf{X}, \mathbf{y}, f), \quad \mathbf{z}_2^* = \arg\min_{\mathbf{z}_2 \in \mathbb{R}_+^{h} \|\mathbf{z}_2\|_1 = 1} Q(\mathbf{z}_2|\mathbf{X}, \mathbf{y}, f).$
5: Take the largest components of $\mathbf{z}_1$ and $\mathbf{z}_2$ as $\mathcal{A}$
6: $\mathbf{w}^* = \arg\min_{\mathbf{w} \in \mathbb{R}^r} S(\mathbf{w}|\mathbf{X}, \mathbf{y}, f), \quad$ subject to $w_j = 0$ for $j \notin \mathcal{A}$.

Experiment

To illustrate the proposed method we created the synthetic data. We sampled the subset of matrix $\mathbf{X}$ from normal distribution and generated other columns by linear combination of sampled ones. Fig. 5 shows the absolute correlations between the matrix $\mathbf{X}$ columns. The matrix $\mathbf{X}$ size is 800×100. The data was splitted into train and test in proportion 75%/25%.

The model f is a two-layer neural network with number of hidden units $h = 40$. The number of network parameter is $100 \times 40 + 40 \times 2 = 4080$. We learned this network using SGD optimizer. The learning process is illustrated in Fig. 6. The network achieved 82% accuracy on train set and 78% on test set.

We applied proposed method to this dataset and the model f. The resulting active subset $\mathcal{A}$ contains $82 + 80 = 162$ parameters. It is 25 times less than the original network. We finetuned the network to adjust the weights. The learning process for the reduced neural network is shown in Fig. 7.

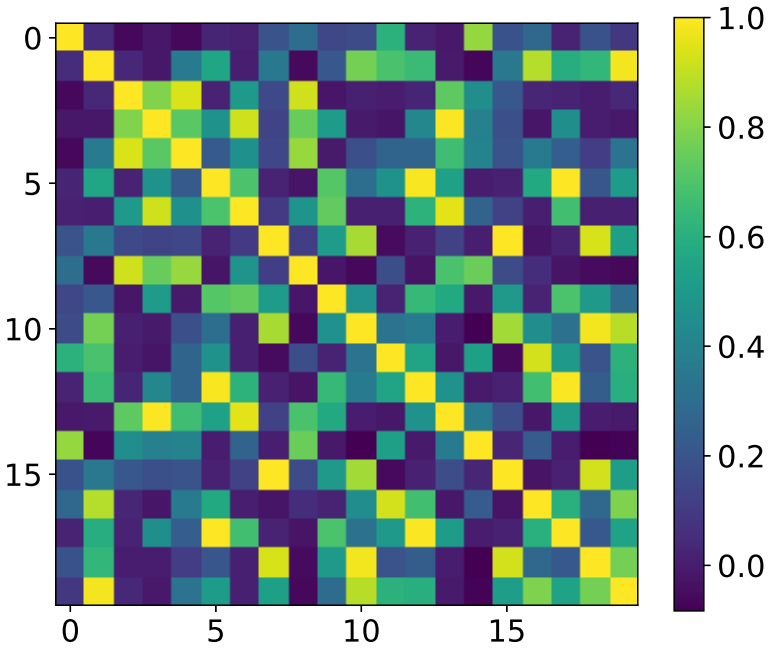

Figure 5: Correlation coefficients between features for synthetic data

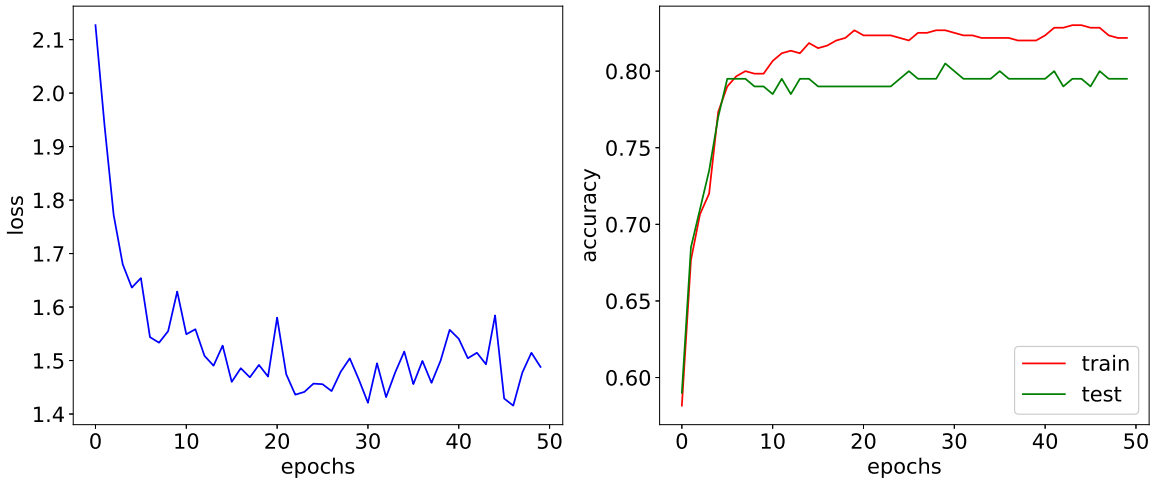

Figure 6: Original neural network learning.

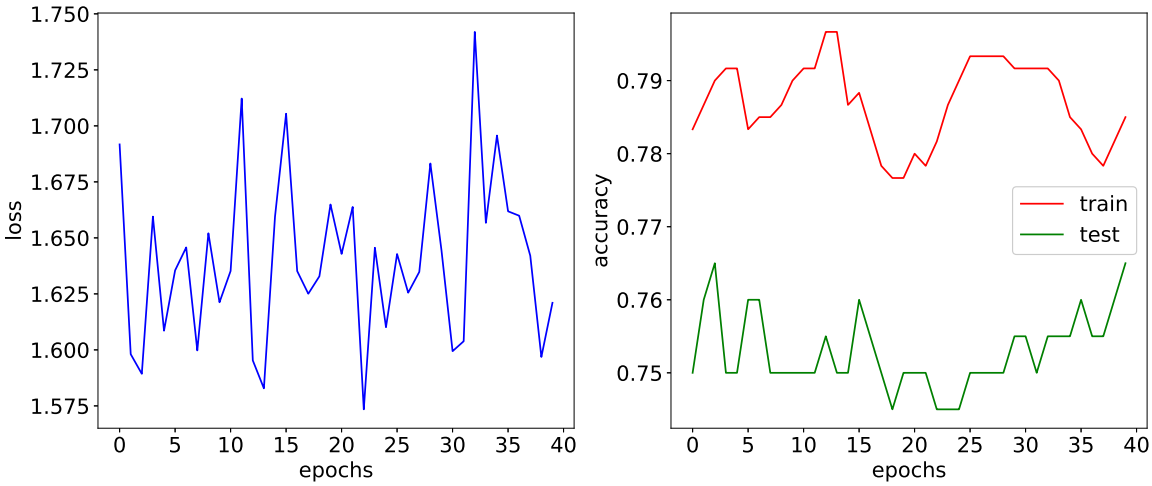

Figure 7: Reduced neural network learning.