# TRANSFERRING INFORMATION BETWEEN NEURAL NETWORKS

*Christopher Ehmann* and *Wojciech Samek*

Fraunhofer Heinrich Hertz Institute
Machine Learning Group
10587 Berlin, Germany

## ABSTRACT

This paper investigates techniques to transfer information between deep neural networks. We demonstrate that a *student network*, which has access to information computed by a *teacher network* on the training data, learns faster, can be less deep and requires less labeled examples to achieve a given performance level. For that we force the student to mimic the teacher by adding a penalty term to the student's objective. We evaluate different penalty terms: (1) mean squared error between the cost gradients, (2) the Jacobian of the pre-softmax layer, (3) its row-summed version, (4) the cost gradient differences to standard double backpropagation and (5) a targeted double backpropagation via gradient derived masks. The Jacobian method improves the accuracy proportional to the difference in training examples, in contrast to the cost gradient. If the difference in accuracy between teacher and student is large enough, we find an improvement from the Jacobian information, even if both had seen the same training data. This indicates that information transfer has a regularization effect.

***Index Terms—*** Neural networks, attention transfer, student-teacher learning, network compression

## 1. INTRODUCTION

Due to increased computational capacities, availability of open source datasets and advancements in theoretical research, Deep Neural Networks (DNNs) currently achieve excellent performance in a wide range of applications, e.g., image classification [8] and quality assessment [3], natural language processing [4], genomics [16] or strategic game playing [19]. Though they perform well on their respective measures, DNNs suffer from a high computation cost during inference, as architectures may contain billions of trainable parameters [5], and from interpretability issues. This limits their usability on certain tasks, for example offline speech recognition on a mobile device, or transcriptomics, where one would like to know, which DNA motif led the protein to bind. The lack of understanding the decision process also prohibits the user from implementing already known information, be it from another classifier or from a domain expert.

In recent years researchers have started to tackle these problems. For instance, there have been efforts to speed up inference by engineering new chip architectures, e.g. Google's Tensor Processing Unit [10], by compressing the network [7] or by introducing fast fourier transform convolutions [13]. Similarly, different techniques to explain a networks decision process have been investigated. The authors of [20] interpreted neural network predictions by computing the partial derivatives of the logits with respect to input, more recent approaches involve the calculation of each neuron's contribution to the decision [2, 14]. A comparison of methods for explaining DNN predictions can be found in [18, 15].

This paper addresses both problems by investigating different ways for training of a shallower architecture (*student network*) with additional information from the explanations of a deeper one (*teacher network*). Hinton et al. [9] proposed knowledge distillation, where the student network is trained additionally to the labels on the logits of deeper architectures, representing the unnormalized probabilities of class membership. Zagoruyko and Komodakis [21] used the minimization of the $L^2$ distance between the gradients of the cost function with respect to the input to transfer information between two image recognition architectures trained of CIFAR-10. Ross et al. [17] used the sum of the logits' gradients to force the network into a certain decision strategy, by adding matrices, which specify at which entries of the input the summed gradients should be close to zero.

Building on these works, we investigate different techniques to transfer information between a teacher and student network. We compare (1) minimizing the mean squared error (MSE) between the cost gradient, (2) the gradients of the logits and (3) their summed version. Additionally, we compare (4) minimizing the cost gradient distance to standard double backpropagation (DB) and (5) a targeted DB with target matrices derived from the teacher's gradients. We find that information transfer between neural networks with the Jacobian of the pre-softmax layer improves performance, even when teacher and student had seen exactly the same training data.

This paper is organized as follows. The next section intro-

This work was supported by the Fraunhofer Society through the MPI-FhG collaboration project "Theory & Practice for Reduced Learning Machines".

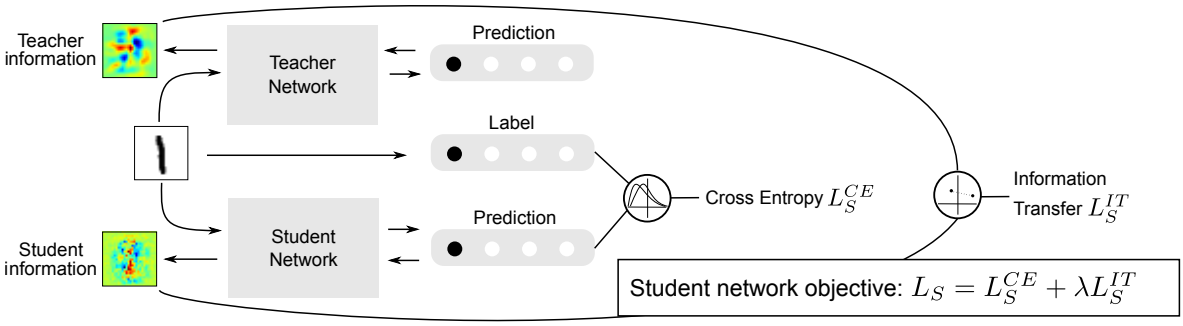



**Fig. 1**: Schema of student learning from teacher.

duces the information transfer methods. Section 3 reports the experimental results and Section 4 concludes the paper with a brief discussion.

## 2. INFORMATION TRANSFER BETWEEN NEURAL NETWORKS

In student-teacher learning, a teacher network T provides additional information to a student network S in order to facilitate (or speed up) the learning process of the student. Since both networks can have different architectures, the teacher cannot simply share its weights, but rather has to share a quantity which is present at both sides. Here we use gradient information, i.e., so-called sensitivity maps, for the information transfer. These maps are computed for each training sample by both networks and, abstractly speaking, provide information about the networks' decision processes. Through the minimization of the MSE between the student's and teacher's sensitivity map, the student network is forced to mimic the teacher network. This teacher's guidance helps the student to focus on the important (i.e., sensitive) locations in the input, which significantly facilitates and speeds up the learning process. Fig. 1 summarizes the student-teacher learning. The student optimizes an objective

$$L_S = L_S^{CE} + \lambda L_S^{IT} \tag{1}$$

which consists of the standard cross entropy loss $L_S^{CE}$, capturing the error between the predictions and the true labels, and the information transfer loss $L_S^{IT}$, specifying the mismatch between the information derived on the student and on the teacher side. The parameter $\lambda$ trades-off the two losses.

In this paper we compare different information transfer losses. The first method computes partial derivatives of the student's and teacher's cost function. With that the information transfer loss is given by

$$L_S^{IT} = \frac{1}{N * D} \sum_{n=1}^{N} \sum_{j=1}^{D} \left( \frac{\partial L_T^{CE}}{\partial x_j^n} - \frac{\partial L_S^{CE}}{\partial x_j^n} \right)^2 \tag{2}$$

where $D$ is the size of the $n$-th input sample $x^n$ and $N$ denotes the number of samples in a batch. This term indicates where a change in the input would lead to a shift in the similiarity between network prediction and label.

The second method computes the full Jacobian matrix of the pre-softmax layer. The rows of this matrix correspond to the subfunction of each output node of this layer, the columns to each entry of the input vector. As the softmax function only normalizes these outputs, also called logits, to a probability distribution, we ignore it. This can be interpreted as seperately creating a sensitivity map for each class, that indicates, where a slight shift in the input would lead to a increase, decrease or no change of the predicted class probability. Thus the information transfer loss is given by

$$L_S^{IT} = \frac{1}{N * C * D} \sum_{n=1}^{N} \sum_{i=1}^{C} \sum_{j=1}^{D} \left( \frac{\partial y_i^T}{\partial x_j^n} - \frac{\partial y_i^S}{\partial x_j^n} \right)^2 \tag{3}$$

where $C$ is the number of classes and $\frac{\partial y_i}{\partial x_j}$ denotes the entries of the Jacobian with dimension $C \times D$.

The third method sums the rows of the Jacobian before minimization. Then, the information transfer loss is given by

$$L_S^{IT} = \frac{1}{N * D} \sum_{n=1}^{N} \sum_{j=1}^{D} \left( \sum_{i=1}^{C} \frac{\partial y_i^T}{\partial x_j^n} - \sum_{i=1}^{C} \frac{\partial y_i^S}{\partial x_j^n} \right)^2 \tag{4}$$

This can be interpreted as creating a single map, that shows, which parts are important or unimportant for class prediction.

In the fourth method the squared gradient is minimized by itself, which is akin to minimizing the distance to the null gradient, i.e., the information transfer loss is given by

$$L_S^{IT} = \frac{1}{N * D} \sum_{n=1}^{N} \sum_{j=1}^{D} M_{n,j} \left( \frac{\partial L_S^{CE}}{\partial x_j^n} \right)^2 \tag{5}$$

with $M_{n,j} = 1$ for all $n, j$, This is known as the double backpropagation method [6].

The fifth method penalizes different parts of the sum in eq. 5 by multiplying the squared gradient of a single input with matrix $M_{n,j}$ before minimizing it. In our experiments we create two matrices from the gradients of the teacher, one

that only minimizes the squared gradient, if the squared gradient of the teacher at this entry was below a certain percentage of the maximum, and one, that penalizes the gradient everwhere, but stronger, if the corresponding value was below the teacher's threshold value.

$$M_{n,j} = \begin{cases} 1 & \text{if } \frac{\partial L_T^{CE}}{\partial x_j^n}^2 < p \max\left(\frac{\partial L_T^{CE}}{\partial x^n}^2\right) \\ 0 & \text{else} \end{cases} \quad (6)$$

$$M_{n,j} = \begin{cases} 2 & \text{if } \frac{\partial L_T^{CE}}{\partial x_j^n}^2 < p \max\left(\frac{\partial L_T^{CE}}{\partial x^n}^2\right) \\ 1 & \text{else} \end{cases} \quad (7)$$

## 3. EXPERIMENTAL EVALUATION

### 3.1. Models & Setup

Four networks are used in the experiments: A fully-connected (FC) student and teacher with two and four trainable layers, respectively, and a convolutional student and teacher with two and four trainable layers, respectively. All networks use ReLU activation functions and a final softmax layer. For the FC networks we choose output dimensions of [1000,10] for the student and [1000,500,200,10] for the teacher. Both convolutional networks had a stride of 1 and valid padding. For the convolutional student network, we set kernel size and output depth to [(15,10),(14,10)], for the convolutional teacher to [(10,10),(9,25),(8,100),(4,10)]. We omitted pooling, as the input size is small enough.

All networks were trained of the MNIST dataset for 50000 steps with a batch size of 50 and an initial learning rate of 0.0001 with ADAM [11] in standard setting ($\beta_1$ = 0.9, $\beta_2$ = 0.999, $\epsilon = 1e-8$). All biases were initialized with 0 and all weights were initialized from a truncated normal distribution with a standard deviation of 0.01. To simplify, we dropped techniques such as batch normalization or dropout and did not use weight decay. $\lambda$ was set to 0.1 for the summed and full Jacobian methods and $10^7$ for the cost gradient method; these values were chosen in an initial experiment such that the initial value of the information transfer term was around 0.01% to 1% of the initial cross entropy term. All experiments were implemented in TensorFlow 1.0.0 [1], the networks were constructed with the LRP TF wrapper [12].

### 3.2. Comparison of information transfer methods

The initial experiment investigated, if using the gradient information increases the student's accuracy, even if student and teacher have seen the same amount of data. We trained the convolutional and FC teacher on 20000 randomly sampled and balanced images and computed the sensitivity maps. Then we trained the convolutional student on the information from the convolutional teacher and the FC student on the information from the FC teacher. We limited the number of training examples for the students to 50, 1000 and 20000 images. If information transfer is to occur, we would expect a bigger increase of the student's accuracy, when the difference between the number of training examples is higher. For comparison we added student baseline networks, which only had the label information. See Table 1 for the accuracies on the whole test set achieved at various training set sizes averaged over 7 runs each. On average the FC teacher network achieved an accuracy of 0.973, the convolutional teacher network had an accuracy of 0.987.

**Table 1**: Average test set accuracy of different methods.

| # examples | 50 | 1000 | 20000 |
|---|---|---|---|
| FC | | | |
| only labels | 0.6849 | 0.8882 | 0.9760 |
| cost | 0.6900 | 0.8981 | 0.9816 |
| full Jacobian | 0.8056 | 0.9239 | 0.9761 |
| summed Jacobian | 0.6928 | 0.8879 | 0.9763 |
| convolutional | | | |
| only labels | 0.7186 | 0.9160 | 0.9814 |
| cost | 0.7137 | 0.9272 | 0.9854 |
| full Jacobian | 0.7540 | 0.9359 | 0.9819 |
| summed Jacobian | 0.7250 | 0.9216 | 0.9816 |

For both architectures, the full Jacobian method improved the accuracy of the student over the only label baseline, if the network had only been trained on 50 examples. This performance increase was significant with $p < 0.05$ for the FC model according to the Wilcoxon rank-sum test. At the 1000 examples mark the results were significant for both types, but when the training set size had been increased to 20000, the improvement became smaller and we found no significant differences. Interestingly, even though both the convolutional teacher and baseline student achieved a higher accuracy at the 50 examples mark, the FC student's accuracy gain with additional information was higher compared to the convolutional student's. Thus, the FC maps seem to contain more accessible information. Adding the summed Jacobian term had little effect overall: While it increased the accuracy slightly in the beginning, at the 20000 images mark there was no effect of improving the student's performance. The cost gradient method showed a reverse effect: While it had a negative effect on the convolutional student and a slight positive effect on the FC student at the 50 images mark, it was the only method that improved the networks accuracy significantly, when student and teacher had seen the same data. This indicates that this type of information transfer may have a regularization effect.

### 3.3. Across architecture transfer

In a second experiment we tested, if the difference in accuracy gain between convolutional and FC student network was due to the character of sensitivity maps and if a network could

**Table 2**: Average test set accuracy of the crossover experiment.

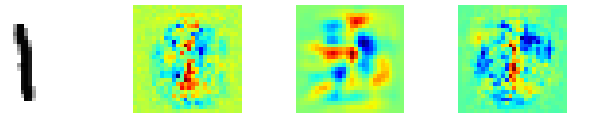
| # examples | 50 | 1000 | 20000 |
|---|---|---|---|
| FC to convolutional | | | |
| only labels | 0.7186 | 0.9160 | 0.9814 |
| cost | 0.7109 | 0.9179 | 0.9846 |
| full Jacobian | 0.8101 | 0.9336 | 0.9815 |
| convolutional to FC | | | |
| only labels | 0.6849 | 0.8882 | 0.9760 |
| cost | 0.6945 | 0.8985 | 0.9817 |
| full Jacobian | 0.7531 | 0.9221 | 0.9780 |

imitate different sensitivity maps without a loss of accuracy. While we reused the teachers and the baseline students, we dropped the summed Jacobian condition and this time the FC student network received the sensitivity maps from the convolutional teacher and vice versa.

Table 2 summarizes the results. The full Jacobian method showed the same behavior as in the previous experiment. However, while at the 50 images mark the FC network with additional information earlier achieved a higher test set accuracy than the convolutional one, after switching the teacher's gradients, this ranking changed, indicating that indeed the FC maps were more useful. Although, this time at the 20000 images mark the results for the convolutional to FC setup were significant. Thus, we find that even if student and teacher had seen the same training data, if the difference between the accuracy of the teacher and the baseline student was high enough, adding the logits' gradients improved the student's performance. Once more, the cost gradient method's impact at the 50 examples mark was positive for the FC student's performance and negative for the convolutional student's accuracy. Comparing this to results of the first experiment, we see that the difference in the direction is not due to the convolutional teacher's maps. As the training set size increased to 1000 images, though it was positive for both, only the effect of the convolutional to FC setup was significant, while at 20000 training images results were statistically significant for both. The gain from the cost gradient still seemed to be independent of the difference in information. In Figure 2 we see the sensitivity maps for class one of a FC teacher, a convolutional student without information and a convolutional student with the added information from the FC teacher's Jacobian. After adding these, the map of the student clearly resembles more that of the teacher.

### 3.4. Targeted double backpropagation

With the knowledge from the previous experiment, we investigated if the effect from minimizing the differences between student and teacher cost gradient was from penalizing the gradient or due to a real information transfer. Thus, we compared



Input FC teacher Only labels Conv student

**Fig. 2**: The images from left to right show the input sample, the map for class one of the FC teacher, the baseline student and the student with the Jacobian information

**Table 3**: Average test set accuracy of the targeted double backpropagation experiment.

| M=1 | cost diff | cost diff + M=1 |
|---|---|---|
| 0.9854 | 0.9850 | 0.9852 |

| | p=0.2 | p=0.5 | p=0.7 | p=0.9 |
|---|---|---|---|---|
| $M \in \{0,1\}^D$ | 0.9845 | 0.9845 | 0.9844 | 0.9848 |
| $M \in \{1,2\}^D$ | 0.9848 | 0.9842 | 0.9838 | 0.9843 |

it to standard DB and our targeted DBs, yet also created a combination of minimizing the cost gradient differences and standard DB. For all methods we set $\lambda = 1e7$. We trained the convolutional teacher as before with an average test set accuracy of 0.988. Table 3 shows the average test set accuracy of 6 runs of the convolutional student. Neither the MSE cost difference, nor the cost difference plus standard DB method could outperform standard double backpropagation, while the combination even leads to a worse performance than just standard DB. Similar results were obtained for the targeted DB: None of the $M \in \{0,1\}^D$ maps achieved better results and there was no discernible trend when we changed p. Unexpectedly, also none of the thresholds of the $M \in \{1,2\}^D$ binary mask reached the effect of the standard DB, but we have to note, only the $M \in \{1,2\}^D$, p = 0.7 results differed significantly from the best result.

## 4. CONCLUSION

We showed, that information transfer between neural networks with the Jacobian of the pre-softmax layer is possible, even when teacher and student had seen exactly the same training images. As the cost gradient's effect was indistiguishable from DB, the Jacobian seems to have better prospects as a means of not only improving accuracy of the student, but as well as changing the decision rules the network is using, though the effectiveness might depend on the network's architecture. A comparison could be to knowledge distillation [9] as one could possibly improve performance further or force the same decision strategy, when the gradient constraints were added. As the gradients are only one possible choice of explaining a network's decision, one could likewise investigate the use of other methods like LRP [2].

## 5. REFERENCES


[1] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, et al. Tensorflow: A system for large-scale machine learning. In *Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, 2016.

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

[3] S. Bosse, D. Maniry, K.-R. Müller, T. Wiegand, and W. Samek. Deep neural networks for no-reference and full-reference image quality assessment. *IEEE Transactions on Image Processing*, 27(1):206–219, 2018.

[4] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, page 17241734, 2014.

[5] A. Coates, B. Huval, T. Wang, D. Wu, B. Catanzaro, and N. Andrew. Deep learning with cots hpc systems. In *International Conference on Machine Learning*, pages 1337–1345, 2013.

[6] H. Drucker and Y. Le Cun. Improving generalization performance using double backpropagation. *IEEE Transactions on Neural Networks*, 3(6):991–997, 1992.

[7] S. Han, H. Mao, and W. J. Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015.

[8] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[9] G. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[10] N. P. Jouppi, C. Young, N. Patil, D. Patterson, G. Agrawal, R. Bajwa, S. Bates, S. Bhatia, N. Boden, A. Borchers, et al. In-datacenter performance analysis of a tensor processing unit. In *Proceedings of the 44th Annual International Symposium on Computer Architecture*, pages 1–12. ACM, 2017.

[11] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[12] S. Lapuschkin, A. Binder, G. Montavon, K.-R. Müller, and W. Samek. The lrp toolbox for artificial neural networks. *Journal of Machine Learning Research*, 17(114):1–5, 2016.

[13] M. Mathieu, M. Henaff, and Y. LeCun. Fast training of convolutional networks through ffts. *arXiv preprint arXiv:1312.5851*, 2013.

[14] G. Montavon, S. Lapuschkin, A. Binder, W. Samek, and K.-R. Müller. Explaining nonlinear classification decisions with deep taylor decomposition. *Pattern Recognition*, 65:211–222, 2017.

[15] G. Montavon, W. Samek, and K.-R. Müller. Methods for interpreting and understanding deep neural networks. *Digital Signal Processing*, 73:1–15, 2018.

[16] D. Quang and X. Xie. Danq: A hybrid convolutional and recurrent deep neural network for quantifying the function of dna sequences. *Nucleic Acids Research*, 44(11):e107–e107, 2016.

[17] A. S. Ross, M. C. Hughes, and F. Doshi-Velez. Right for the right reasons: training differentiable models by constraining their explanations. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pages 2662–2670. AAAI Press, 2017.

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

[19] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

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

[21] S. Zagoruyko and N. Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. *arXiv preprint arXiv:1612.03928*, 2016.