Computational Analysis of Learned Representations in Deep Neural Network Classifiers

Tomáš Kuzma and Igor Farkaš
Department of Applied Informatics, Comenius University in Bratislava
Mlynská dolina, 84248 Bratislava, Slovak Republic
Email: farkas@fmph.uniba.sk

***Abstract*—When a neural network is trained for a specific task, activations of the hidden units encode internal representations of the inputs. Models formulated in a layer-wise fashion are believed to structure such representations in a hierarchical fashion, increasing in complexity and abstractness towards the output layer, in an analogy to both biological neural networks and artificially constructed computational models. This paper examines how the structure of classification tasks manifests itself in these internal representations, using a variety of ad hoc metrics. The results, based on feedforward neural networks trained on moderately complex datasets MNIST and SVHN, confirm our hypothesis that the hidden neurons become more correlated with class information towards the output layer, providing some evidence for an increasing bottom-up organization in representations. While various activation functions lead to noticeably different internal representations as measured by each of the methods, the differences in overall classification accuracy remain minute. This confirms the intuition that there exist qualitatively different solutions to the complex classification problem imposed by nonlinearities in the hidden layers.**

I. Introduction

Deep neural networks (DNNs) [1] have demonstrated excellent performance in complex machine learning tasks such as image classification, natural language processing, or speech recognition [2]. However, due to their multilayer nonlinear structure, they are not transparent and so it is hard to understand their behavior. Various types of investigations have been performed focusing on different aspects of DNNs. Several methods have been introduced to provide understanding of what a DNN has learned from the perspective of hidden-layer representations. For instance, in [3] the tools were developed that make it possible to analyze DNNs in more depth and to accomplish the tracing of the so-called invariance manifolds learned by the network, for each of the hidden units.

Alternatively, kernel methods have been used, decoupling learning algorithms from the datasets used. In [4] the evolution of hidden-layer representations in a DNN is analyzed by building a sequence of deeper and deeper kernels that subsume the mapping performed by more and more layers of the DNN, and measure how the increasingly complex kernels fit the learning problem. It is observed that DNNs create increasingly better representations of the learning problem and that the network structure controls how fast the representation of the task is formed layer after layer.

In image classification tasks, a large body of work was dedicated to the visualization of particular neurons or neuron layers; see, e.g. [5] and references therein. They introduce two useful tools, the first visualizing internal activations during input processing (image or video) by a trained convolution net, providing intuitive insights into the model functioning. The second tool enables to visualize features at each layer of a DNN via regularized optimization methods that provide qualitatively clearer, more interpretable visualizations, compared to earlier attempts. In [6] the focus is on methods that visualize the impact of particular regions of a given and fixed single image for a prediction of this image. The authors show that the recently proposed layer-wise relevance propagation algorithm qualitatively and quantitatively provides a better explanation of DNN classification decisions, compared to other methods (the sensitivity-based approach or the deconvolution method).

A deal of work has been dedicated to understanding how DNNs learn, by providing several intuitive arguments that could help trying to identify the contributing factors [7, 8]. One of the ideas, the manifold hypothesis, views the learning process as an unfolding of the manifold-shaped data towards higher layers, hence simplifying the mapping towards the targets. In [9], it is shown how to quantitatively validate the unfolding (flattening) hypothesis, by proposing new quantities for measuring this process, and demonstrating their usefulness on several experiments (with both synthetic and real-world datasets).

In a recent paper [10], a powerful tool (SVCCA), combining Singular Value Decomposition and Canonical Correlation Analysis, was proposed for an analysis and comparison of learned deep representations, revealing answers to various questions, related to convergence of hidden-layer representations, intrinsic dimensionality of representations, or unit sensitivity to different classes. In addition, it has been shown useful for suggesting new training regimes that simultaneously save computation and are less prone to overfitting.

In this paper, we aim to enrich this portfolio of investigative methods by focusing our analysis on classification tasks. We propose three conceptually and computationally simple methods of examining the class structure of internal representations: correct-output correlation, per-class statistics and sparsity of class selectivity, and base our pilot analysis on two well-known image classification tasks of medium complexity.

The remainder of the paper is organized as follows: Section II presents different models and the dataset used for the analysis of the trained DNNs on the classification task. Section III introduces three new approaches to studying the internal representations. Section IV presents results of the experiments. Section V summarizes the paper contributions.

978-1-5090-6014-6/18/$31.00 ©2018 IEEE

II. Models

For assessing the suitability of our devised methods of analysing the internal representations, we need to concern ourselves with both problems and models of intermediate difficulty. While the state-of-the-art neural networks employing convolutions, dropout, regularizations, renormalizations etc. are too complex to analyse with simple methods, models very limited in complexity construct no meaningful internal representations. Conversely, the complexity of the model must also match the difficulty of the task. An easy task would underutilise the model, resulting in inelaborate representations, while an intermediately complex network would fail to learn a very challenging task, leaving no room for analysis.

A. Activation functions

In this paper, we will omit recurrent and convolutional neural networks and will only consider simple feedforward neural networks. While being elementary models, they still incorporate most of the relevant principles of the more intricate variants. We will further restrict the networks to a simple architecture of several fully-connected layers of 100 neurons with various activation functions:

- logistic sigmoid:
$$\operatorname{logsig}(x) := \frac{1}{1 + \exp(-x)}$$
- hyperbolic tangent:
$$\tanh(x) := \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)}$$
- softsign function (introduced in [11]):
$$\operatorname{softsign}(x) := \frac{x}{1 + |x|}$$
- rectified linear units (ReLU):
$$\operatorname{relu}(x) := \begin{cases} x, & \text{if } x \geq 0 \\ 0, & \text{otherwise} \end{cases}$$

Each network has the *softmax* output layer representing the classification in a conventional *one-hot encoding*. We will train the networks using a simple first-order gradient descent, augmented using the momentum method, using constant rates and no regularization or input preprocessing.

B. Datasets

While there is a wealth of challenging tasks, largely focusing on image recognition, the same cannot be said about tasks of suitable moderate difficulty. We will perform out experiments on two image classification datasets. The primary one, the MNIST database [12] of hand-written digits, is a common benchmark for machine learning algorithms, consisting of a training set of 60 000 and a testing set of 10 000 grey-scale 28×28 pixel large images, produced from larger originals by weighted centering and cropping. We will further split the training set into an estimation set of 50 000 and a validation set of 10 000 samples, with each training run having an independent partitioning of the training set.

For verification purposes, we will employ a secondary task, the StreetView House Numbers (SVHN) dataset [13]. The preprocessed variant of this dataset contains full-color 32×32 pixel large segments of photographs taken in uncontrolled conditions from moving vehicles, each centered on a single digit to classify. This represents an increase in both task and data complexity over the MNIST dataset.

Unless otherwise noted, the network is first trained on the estimation set for 100 epochs, then a retrospective *early-stopping* selects the epoch with the lowest error on the validation set. Final testing and analysis is performed using this selected model on the provided testing set. For each combination of the network depth and the activation function used, the testing classification error is in the range of 1–2% for the MNIST task, yielding well-trained models for subsequent analysis.

III. Methods

To gain insight into the inner working of neural networks used for classification, we will devise several methods of extracting macro-information from the breadth of underlying neuronal activations.

A. Correlation

The first metric we will introduce is *class-correlation* which assigns a number from the interval $\langle 0; 1 \rangle$ to each hidden neuron in the network – how much useful information for the correct classification can be *linearly* extracted from that neuron. A unit whose activations bear no linear relation to a membership of the input in any class will receive a zero score, while a unit that has a direct relationship with such a membership – such as an output neuron in a one-hot trained network – will achieve a value close to one.

The class-correlation metric is based on the Pearson correlation coefficient which describes linear relations between random variables. One variable of this relation is the activation a_i of a particular i-th hidden unit (as a response to input p), the other is the class-k membership indicator variable $c^{(k)}$, computed with respect to correct classification of the input p:

$$c^{(k)} := \begin{cases} 1, & \text{if } \operatorname{class}(p) = k \\ 0, & \text{otherwise} \end{cases}$$

Both the activation a_i and the indicator variables $c^{(k)}$ are understood as random variables, with each of their realizations corresponding to a particular testing sample of the studied task.

To quantify the useful information content of a single i-th hidden neuron, we will compute the final correlation metric as

$$\rho_i := \max_k \left| \operatorname{corr}(\{a_i\}, \{c^{(k)}\} \right| \qquad (1)$$

A unit that correlates positively with a membership in a particular class is equally useful[1] to a neuron that has the same but negative relationship; therefore we only take absolute value of the correlation into account. Finally, to obtain a single metric for each unit, we take the maximum of the correlation values for all of the classes.

[1] Network weights are not constrained in any fashion and can be positive or negative with no inherent asymmetry in the computation.

B. Selectivity

While the simple correlation-based metric reveals some level of hierarchic organization, this method is limited by its linearity and by the single scalar value assigned to each neuron. To get a more detailed insight, we need to use methods that are either greater in scope or more sophisticated. Our next method looks at individual sensitivity of neurons to inputs of different classes.

For each hidden unit and for each input class, we will collect statistical information of the representations – the average, standard deviation and the extrema of the activations. This offers a more detailed view of the individual neurons and their selectivity, which facilitates some interesting observations.

C. Sparsity

Our final method to quantify the usefulness of individual neurons is based on the concept of *sparsity* of the preceding selectivities. Intuitively, a neuron that reacts with the same (expected) strength to inputs of any class will tend to provide no useful information. As with the correlational approach, each neuron will be assigned a single number with a similar interpretation. An equi-responsive neuron, which has the same *expected* activation for any input class, will be assigned a zero value, while a neuron that strongly (either positively or negatively) responds to members of one particular class, such as a one-hot output neuron, will acquire a high score.

1) General sparsity measures: In other words, we want to grade class-*imbalance* of units' responses. One such conventional measure is the entropy, or Kullback–Leibler divergence, applicable on probability distribution (which unit activations are not, without further transformation). The merits of various sparsity measures are discussed in depth in [14], with the most suitable being (for a vector of non-negative numbers, $x_i \in \mathbb{R}_0^+$):

- L_1-normalized negative entropy:

$$1 + \frac{1}{\log n} \sum_i \frac{x_i}{\|\mathbf{x}\|_1} \log \frac{x_i}{\|\mathbf{x}\|_1}$$

- L_2-normalized negative entropy:

$$1 + \frac{1}{\log n^2} \sum_i \frac{x_i^2}{\|\mathbf{x}\|_2^2} \log \frac{x_i^2}{\|\mathbf{x}\|_2^2}$$

- *Hoyer* metric:

$$\left(\sqrt{n} - \frac{\sum_i x_i}{\|\mathbf{x}\|_2} \right) / \left(\sqrt{n} - 1 \right)$$

- *Gini* index (applicable to ascendingly sorted x_i):

$$1 - \frac{2}{n} \sum_{i=1}^{n} \left(\frac{x_i}{\|\mathbf{x}\|_1} \cdot \left(n - i + \tfrac{1}{2} \right) \right)$$

Among other favorable properties, these measures[2] fit into the convention for the information value – zero represents no useful information, while a single-class membership detector would be quantified by one.

2) Sparsity in classification: To visualize the structure of the class selectivities of hidden neurons, we will plot a *sparsity profile* for all the layers. In the testing phase, a sparsity for the vector of class-averaged activations is computed for each hidden unit. As the sparsity metrics are only defined for vectors of positive values, we will suitably[3] shift the values for activations that can yield negative numbers. The resulting values are sorted in non-increasing order, representing the ordered distribution of information density in internal representations.

IV. Experiments

Analyzing every trained network variant with each of the introduced methods yields an impractical number of results, therefore we will single out the clearest examples of the observable trends.

A. Correlation

The simplest metric – *class-correlation* – assigns an information content value (real value from 0 to 1) to each hidden neuron (Eq. 1). As the network learns to perform the task, the information content of at least some of the neurons should increase, more so towards the output layer. To showcase this effect, we will plot the change of these values in time, across epochs, i.e. number of presentations of the complete training dataset. For each layer of units we construct a *quantile plot*, which depicts the distribution of the correlation values for each training epoch. The average value is charted by the thick black line, while the shading represents the density of the values' distribution. This is equivalent to a series of superimposed box plots of narrowing quantiles and can be understood as a color-encoded histogram with an additional time axis. We can see such a visualization on the top of Fig. 1 in the case of ReLU units.

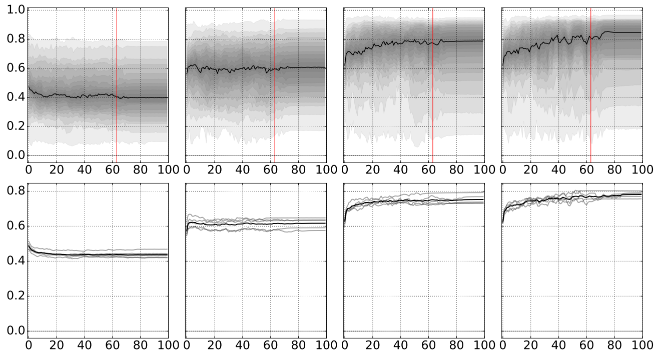


Fig. 1: Progression of the correlation values during the training of networks on the MNIST dataset, comprising four layers of *ReLU* hidden neurons. The first row represents a single training session while the second aggregates multiple runs. Red vertical lines represent the cutoff epoch, when the model performed best on the validation set. For interpretation, see the text.

As we can see, higher layers generally exhibit a greater amount of linearly-available useful information. While this effect is not clear-cut, it is stable in terms of multiple independent runs (as seen on the bottom half of Fig. 1).

[2] Entropy measures were rescaled to gain this property.

[3] i.e. $+1$ for $\tanh(x)$ or $\mathrm{softsign}(x)$, resulting in the range of $(0; +2)$. Note that all four used sparsity measures are scale-invariant.

The second important observation is that as the network learns to perform the classification task better, the correlation values increase. This effect is more pronounced closer to the output layer, as the reduced number of intervening layers forces the partial output mapping to be less non-linear (the class-correlation coefficients are essentially a linear measure).

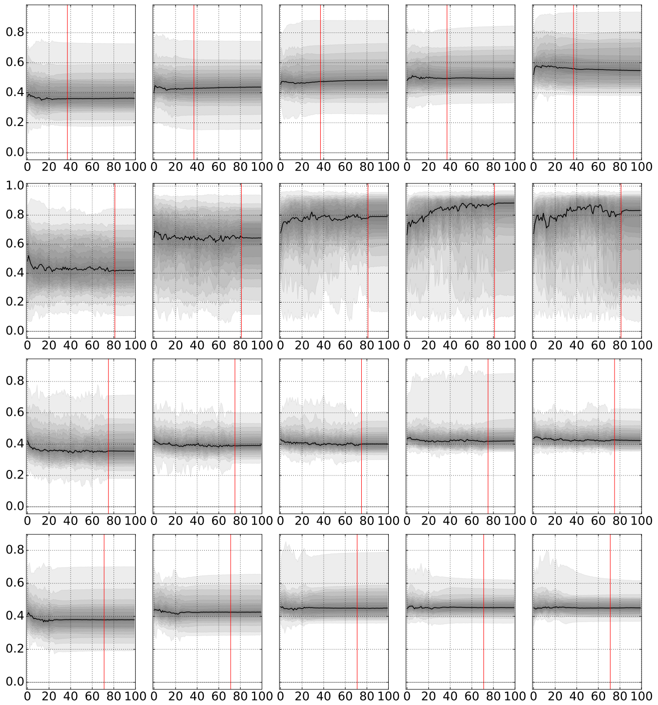


Fig. 2: Correlation during training in networks with five hidden layers using different activations – from top to bottom: *logistic sigmoid*, *rectified linear units*, *hyperbolic tangent* or *softsign*.

activation	1-vs-2	2-vs-3	3-vs-4	4-vs-5
logistic sigmoid	0.0001	0.0013	0.0322	0.0003
rectified linear unit	0.0008	0.0000	0.0120	0.0208
hyperbolic tangent	0.0010	0.1885	0.0030	0.2442
softsign	0.0000	0.0105	0.2457	0.3402

TABLE I: Significance of the differences between consecutive hidden layers as p-values of a Mann–Whitney–Wilcoxon test for independent samples, computed for a single training session. (Compare the results to the matching Fig. 2.)

As a next step, we look at how the choice of activation function influences both the scale of the values of the correlation and the scale of the aforementioned observed effects – Fig. 2 presents the results for various activation functions. Table I illustrates that the increases in useful information, as measured by the correlation values, are statistically significant for most layer-to-layer transitions, with lower layers and unipolar activation functions resulting in a more clear-cut effect. We compare the distributions of the correlation values for each pair of consecutive hidden layers using the Mann–Whitney–Wilcoxon rank-sum test (for independent samples), with the p-values indicating the significance level at which we reject the null hypothesis that the two distributions are equal.

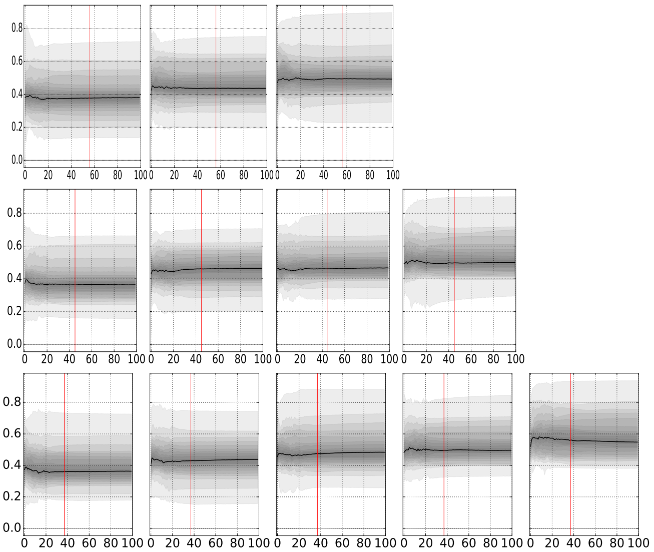


Fig. 3: Correlations during training in networks with *logistic sigmoid* neurons in three, four or five hidden layers.

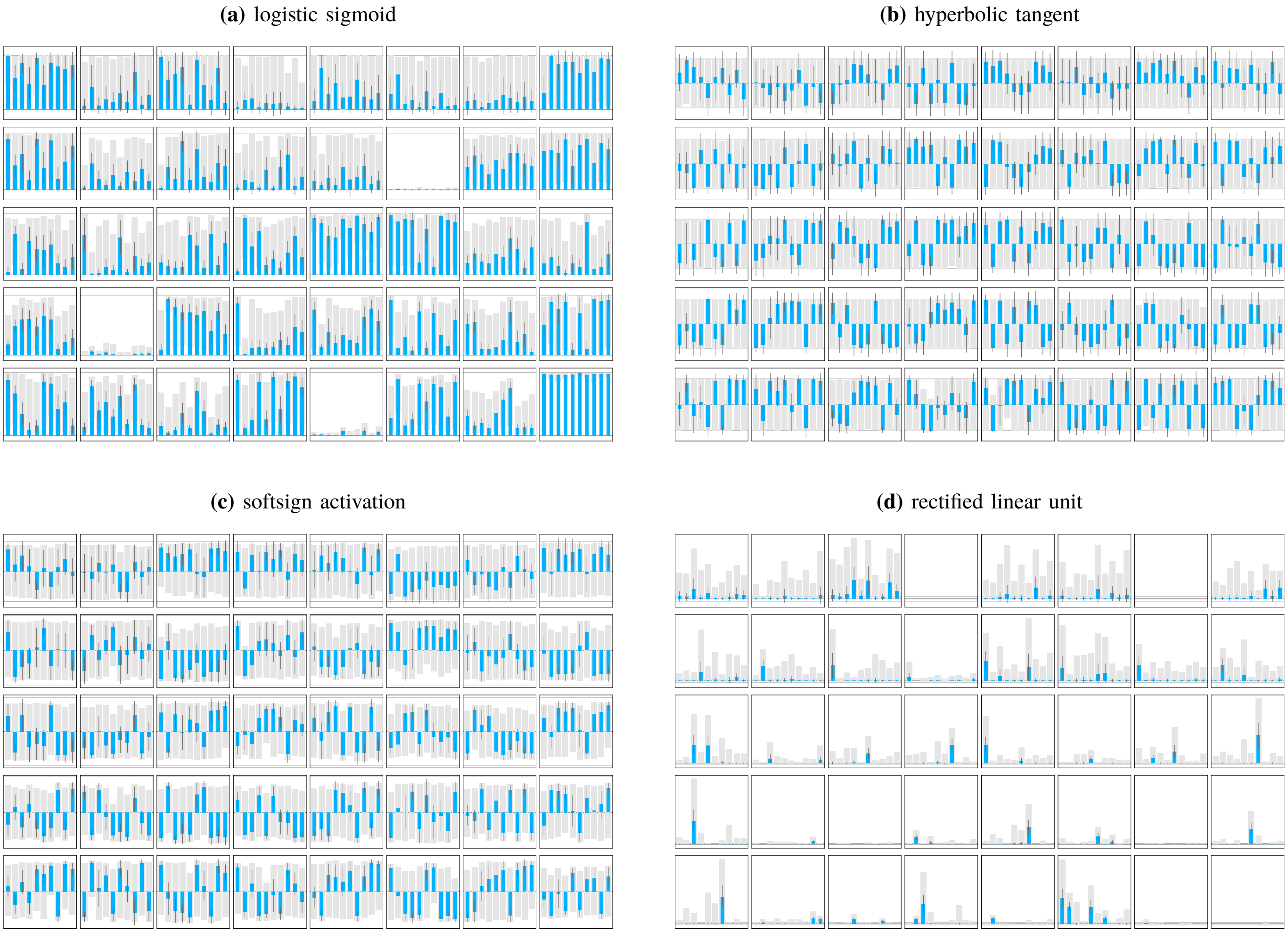
Rectified linear units lead to the most apparent growth of correlations towards higher layers (supported by the lowest p-values), whereas for other functions this growth is less visible, albeit sometimes still significant or completely insignificant. We performed several trials with each model, observing small changes in significance, but the overall pattern remained the same – rectifier units reveal the strongest and most dynamically evolving correlations on all layers.

Last but not least in this section, we wanted to see how correlations change if the number of hidden layers is manipulated. As more layers are added to a network, both the prediction accuracy and the correlation values increase, as shown in Fig. 3. This hints at hierarchical organization of the internal representations and the advantages of a deeper networks, that can accommodate a complex hierarchy and express more intricate mappings.

B. Selectivity

A more detailed view (see Fig. 4) using the class selectivities of the hidden neurons yields several interesting observations. We portray the activation statistics for a few randomly selected individual neurons from each hidden layer (in rows) by composite bar plots. For each of the ten output classes, activation statistics are displayed separately: the most conspicuous blue bar represents the average (from a zero baseline), the larger gray bar in the background depicts the absolute range of activation values and, finally, the per-class variance of activations is expressed as a thin black error bar.

In a network using *logistic sigmoid* units (see Fig. 4a), the neuronal activations become more focused on the higher layers, as evidenced by the shrinking per-class variance of the activations. No units are strongly sensitive to any particular class – logistic sigmoid units lead to distributed internal representations. Worth noting is also the presence of several



(a) logistic sigmoid

(b) hyperbolic tangent

(c) softsign activation

(d) rectified linear unit

Fig. 4: Class selectivity diagrams for trained five-layer networks with different activation functions, using the MNIST dataset. In each diagram, the k-th row visualizes eight randomly selected neurons from the k-th hidden layer. The figure reveals evident differences among hidden-layer representations, regarding their sparsity and unit class-correlations, for various activation functions. More detailed description is provided in the text.

"stuck" units that react either strongly or weakly to *each* class, therefore not providing much useful information. This hints at certain redundancy on the higher levels of the network, which may be exploited for efficiency by selecting a tapering model architecture.

Replacing the unipolar logistic sigmoid activation function by a bipolar but otherwise similar *hyperbolic tangent* yields similar results (see Fig. 4b) – the internal representation also tends to be distributed and the uncertainty of the network diminishes towards the output (see the per-class variance bars). The neuron responses are strongly divided into positive and negative per-class (more so than the low/high values in the logistic sigmoid case) – the hidden neurons serve to partition the input space into two mostly balanced groups of classes. Using the more smoothly non-linear *softsign* activation function (which was devised by [11] as a softer, less *saturating* alternative to the hyperbolic tangent function) produces very similar properties, but with less extreme values (see Fig. 4c).

Models with *rectified linear units* structure their internal representations in a completely different way (see Fig. 4d). While the previous three activation functions are continuous, changing post-synaptic activations smoothly in response to pre-synaptic net input, the rectifier is discontinuous at zero, with any negative net input resulting in a zero output, and positive net input having strictly linear response. In a trivial analysis with random normally distributed inputs and weights, this would lead to output being zero with a 50% probability. The actual measurements point towards the output being even more *sparse*, and the sparsity increased towards the output. Neurons mostly act as very selective *feature detectors*, only responding to inputs from a limited number of classes. In contrast to the distributed representations found when employing continuous activations, the discontinuous rectifiers lead to *localistic encoding* in their sparse internal representations.

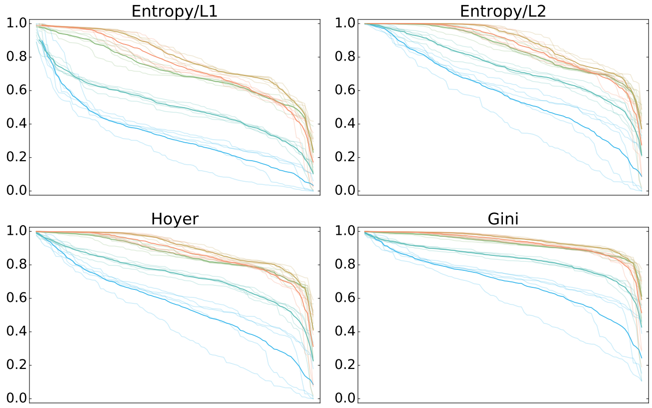


Fig. 5: Comparison of the four different sparsity metrics on trained networks with five layers of rectified linear units. The x-axis represents the 100 hidden units in each layer, sorted in non-increasing order.

C. Sparsity

While the previous approach provides us with a wealth of pictorial evidence to form insights and intuitions, it does not lend itself easily to rigorous testing of the formed hypotheses. Computing the *sparsity* of the average per-class activations is one of the possible methods for summarizing this information in a useful way. Figure 5 portrays the per-layer sparsity profiles computed using four different sparsity measures on multiple runs of the same task. As the calculated profiles are essentially in agreement on the magnitudes, trends and mutual ordering of the layers, we will restrict ourself therafter to visualizing the *Hoyer* metric of sparsity.

Figure 6 visualizes the sparsity profiles for various depths of a network employing rectified linear neurons. The diagram displays the results of five independent simulations: the individual runs are portrayed by thin lines and their average is plotted with a thick line. The characteristics of the sparsity profiles are robust across separate trials. For three hidden layers, there is a clear ordering of the sparsity profiles – neurons on a higher layer are more specialized than neurons on a lower layer (average-wise and quantile-wise, but not for *any* pair of neurons). Adding further layers to the model, however, results in almost no such separation for the added layers.

It is important to notice that the per-class activations of the trained rectified units are very sparse, in agreement with Fig. 4d. In fact, for every layer, there is a neuron that is at or near 100% sparsity – such a neuron only responds to inputs from a single class. The sparsity profile then declines quickly for lower layers, indicating that higher layers contain more strongly selective units.

For continuous activation functions (see Fig. 7), the calculated values are considerably lower – representations for networks with continuous activations are less sparse. This parallels the findings in subsection IV-B concerning localistic versus distributed representations. While there still is an apparent ordering among the sparsity profiles for the hidden layers, the differences are lower than in the case of the rectified units.

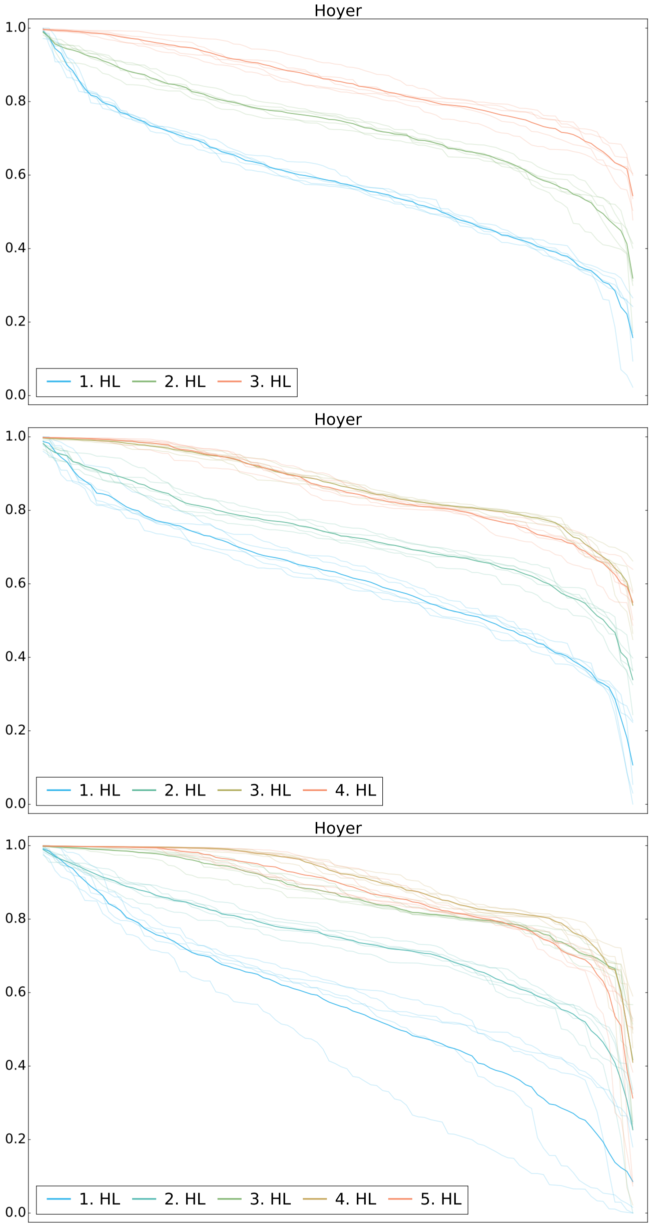


Fig. 6: *Hoyer* sparsity profiles of trained networks with three, four and five layers of rectified linear units, using the MNIST dataset.

D. SVHN

To validate our previous results, the same experiments were also performed on the StreetView House Numbers (SVHN) dataset. This more complex task can also benefit from more hidden layers, but the classification accuracy, given our focus on simple multi-layer feedforward networks, is still far from the current state of the art.

We observed the same general trend of increasing correlation values during training as well as on a layer-wise basis, as visualized in Fig. 8. Of note could be the fact that the correlation values start at (or near) zero, as the more

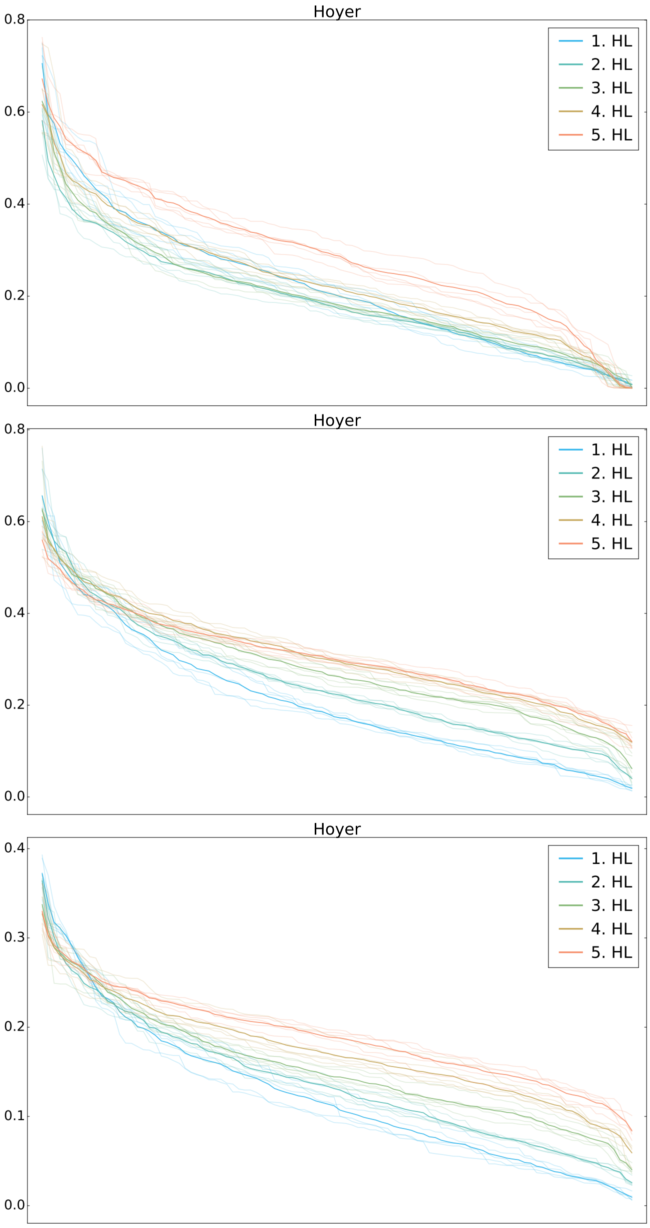


Fig. 7: *Hoyer* sparsity profiles of networks with five hidden layers of either logistic sigmoid (top), hyperbolic tangent (middle) or softsign units (bottom), using the MNIST dataset.

challenging task has no trivial input-to-class correlations.

The specificity and sparsity profiles also subtly, but not fundamentally differ from the results for the MNIST dataset. As seen in Fig. 9, while the actual sparsity values differ, the profiles still have the same ordering and general characteristics as before. The only exception is the first hidden layer for two of the activation functions. This may be connected to the fact that the inputs in SVHN frequently include distractors, off-centre digits which are not related to the desired classification, while the MNIST dataset includes no such irrelevant inputs.

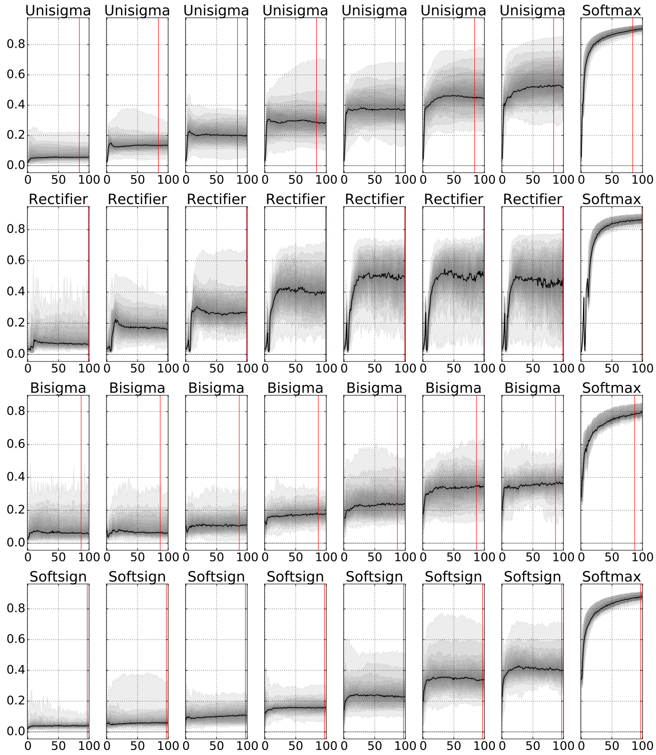


Fig. 8: Correlations during training on the SVHN task in networks with seven hidden layers using different activations – from top to bottom: *logistic sigmoid*, *rectified linear units*, *hyperbolic tangent* or *softsign*.

V. Conclusion

Our goal in this paper was to shed light on knowledge representation principles on the hidden layers of feedforward network classifiers trained on a chosen dataset of medium complexity, with a sufficient number of classes. Regarding the hidden unit correlations with the output classes, the simulation experiments confirmed our hypothesis of an increasing order towards the output, which could be revealed even with linear measures such as correlation. This increase was shown to depend on the activation function, resulting in qualitatively different solutions to the same classification problem when looking at distributions and sparsity of the internal representations. The results of this pilot study were based on just two datasets and could be extended to other datasets. In addition, it would be worth exploring how the ad hoc measures introduced here are related to other quantitative measures proposed and tested in related papers. In any case, there is still a long way to go towards fully uncovering the complexity of deep neural network models.

Acknowledgment

This work was supported by the Slovak Grant Agency for Science (VEGA), project 1/0796/18 and the KEGA project 017UK-4/2016. The authors thank the anonymous reviewers for the comments.

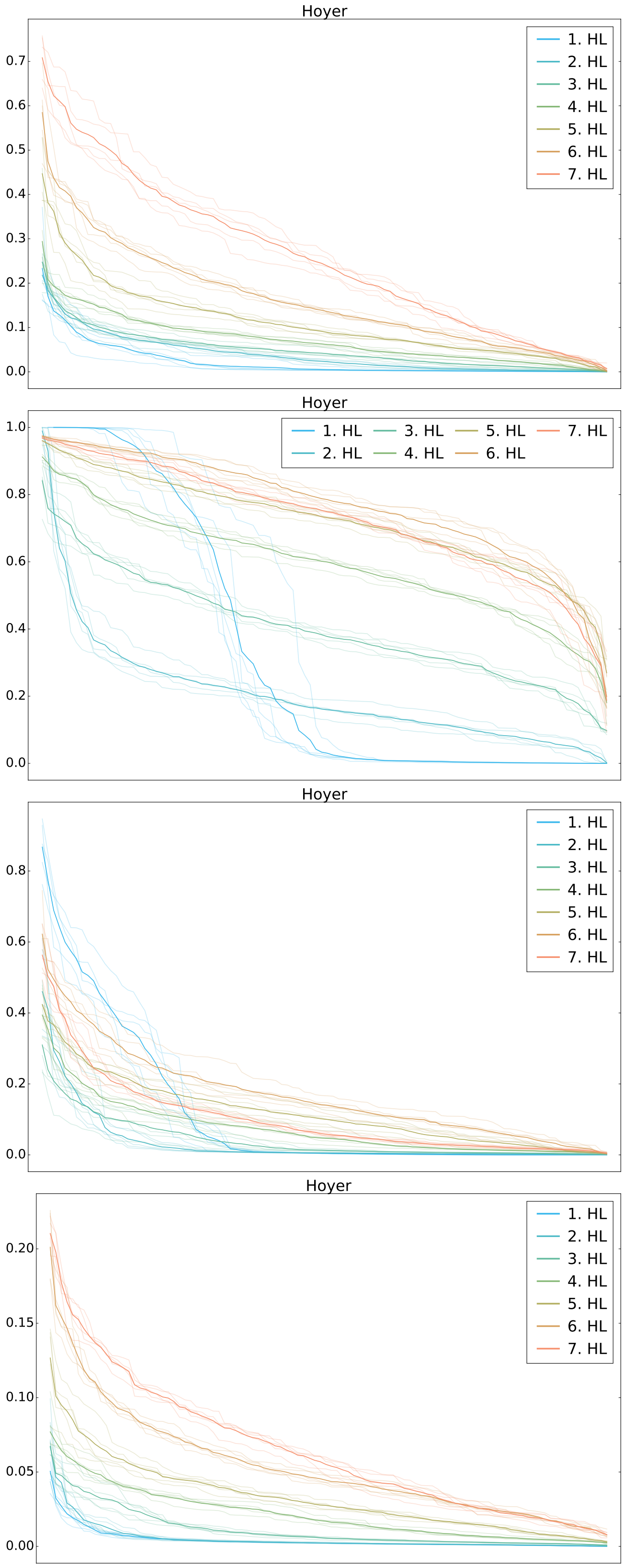


Fig. 9: *Hoyer* sparsity profiles of networks with seven hidden layers trained on the SVHN task using different activations – from top to bottom: *logistic sigmoid*, *rectified linear units*, *hyperbolic tangent* or *softsign*.

References

[1] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks*, vol. 61, no. 11, pp. 85–117, 2015.

[2] L. Deng and D. Yu, "Deep learning: Methods and applications," *Foundations and Trends in Signal Processing*, vol. 7, no. 3–4, pp. 197–387, 2013.

[3] D. Erhan, A. Courville, and Y. Bengio, "Understanding representations learned in deep architectures," Tech. Rep. 1355, 2010.

[4] G. Montavon, M. L. Braun, and K.-R. Mëller, "Kernel analysis of deep networks," *Journal of Machine Learning Research*, vol. 12, pp. 2563–2581, 2011.

[5] J. Yosinski, J. Clune, A. Nguyen, T. Fuchs, and H. Lipson, "Understanding neural networks through deep visualization," in *31st International Conference on Machine Learning*, 2015.

[6] 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*, vol. 28, no. 11, pp. 2660–2673, 2017.

[7] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[8] Y. Bengio, G. Mesnil, Y. Dauphin, and S. Rifai, "Better mixing via deep representations," in *Proceedings of 30th Int. Conf. Machine Learning*, 2013, pp. 552–560.

[9] P. P. Brahma, D. Wu, and Y. She, "Why deep learning works: A manifold disentanglement perspective," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, no. 10, pp. 1997–2008, 2016.

[10] M. Raghu, J. Gilmer, J. Yosinski, and J. Sohl-Dickstein, "SVCCA: Singular vector canonical correlation analysis for deep learning dynamics and interpretability," in *31st Conference on Neural Information Processing Systems (NIPS)*, 2017.

[11] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the International conference on artificial intelligence and statistics*, 2010, pp. 249–256.

[12] Y. LeCun and C. Cortes. (2010) MNIST handwritten digit database. http://yann.lecun.com/exdb/mnist/.

[13] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning," in *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

[14] N. Hurley and S. Rickard, "Comparing measures of sparsity," *IEEE Transactions on Information Theory*, vol. 55, no. 10, pp. 4723–4741, October 2009.