Understanding Locally Competitive Networks

Rupesh Kumar Srivastava, Jonathan Masci,
Faustino Gomez, Jürgen Schmidhuber
IDSIA, USI-SUPSI
Manno–Lugano, Switzerland
{rupesh, jonathan, tino, juergen}@idsia.ch

Abstract

Recently proposed neural network activation functions such as rectified linear, maxout, and local winner-take-all have allowed for faster and more effective training of deep neural architectures on large and complex datasets. The common trait among these functions is that they implement local competition between small groups of units within a layer, so that only part of the network is activated for any given input pattern. In this paper, we attempt to visualize and understand this self-modularization, and suggest a unified explanation for the beneficial properties of such networks. We also show how our insights can be directly useful for efficiently performing retrieval over large datasets using neural networks.

A version of this paper was submitted to NIPS 2014 on 06-06-2014

1 Introduction

Recently proposed activation functions for neural networks such as rectified linear (ReL;[1]), maxout [2] and LWTA [3] are quite unlike sigmoidal activation functions. These functions depart from the conventional wisdom in that they are not continuously differentiable (and sometimes non-continuous) and are piecewise linear. Nevertheless, many researchers have found that such networks can be trained faster and better than sigmoidal networks, and they are increasingly in use for learning from large and complex datasets [4, 5]. Past research has shown observational evidence that such networks have beneficial properties such as not requiring unsupervised training for weight initialization [1], better gradient flow [2] and mitigation of catastrophic forgetting [3, 6]. Recently, the expressive power of deep networks with such functions has been theoretically analyzed [7]. However, we are far from a complete understanding of their behavior and advantages over sigmoidal networks, especially during learning. This paper sheds additional light on the properties of such networks by interpreting them as *models of models*.

A common theme among the ReL, maxout and LWTA activation functions is that they are locally competitive. Maxout and LWTA utilize explicit competition between units in small groups within a layer, while in the case of the rectified linear function, the weighted input sum competes with a fixed value of 0. Related activation techniques have been studied in the past decades, including recurrent networks with locally competitive units [8]. Self-delimiting recurrent networks with competitive units [3, 9] can in principle learn to decide their own run time and effective number of parameters, thus learning their own computable regularizers. In this paper, we restrict our analysis to networks trained with gradient-based algorithms which are often trained with the dropout regularization technique [10, 11] for improved generalization.

We start from the observation that in locally competitive networks, a subnetwork of units has non-zero activations for each input pattern. Instead of treating a neural network as a complicated highly nonlinear function approximator, the expressive power of the network can be interpreted to be coming from its ability to activate different subsets of linear units

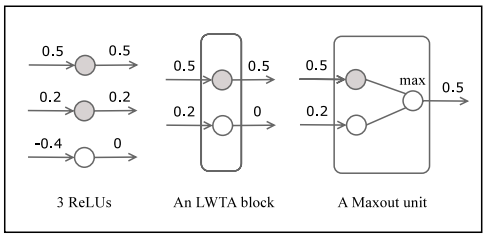


Figure 1: Comparison of rectified linear units (ReLUs), local winner-take-all (LWTA), and maxout activation functions. The pre- and post-synaptic activations of the units are shown on the left and right side of the units respectively. The shaded units are 'active' – non-zero activations and errors flow through them. The main difference between maxout and LWTA is that the post-synaptic activation can flow through connections with different weight depending on the winning unit in LWTA. For maxout, the outgoing weight is the same for all units in a block.

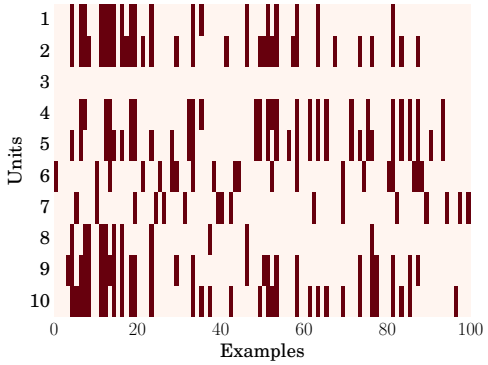


Figure 2: Subnetworks for 100 examples for 10 ReLUs. The examples activate many different possible subsets of the units, shown in dark. In this case, unit number 3 is inactive for all examples.

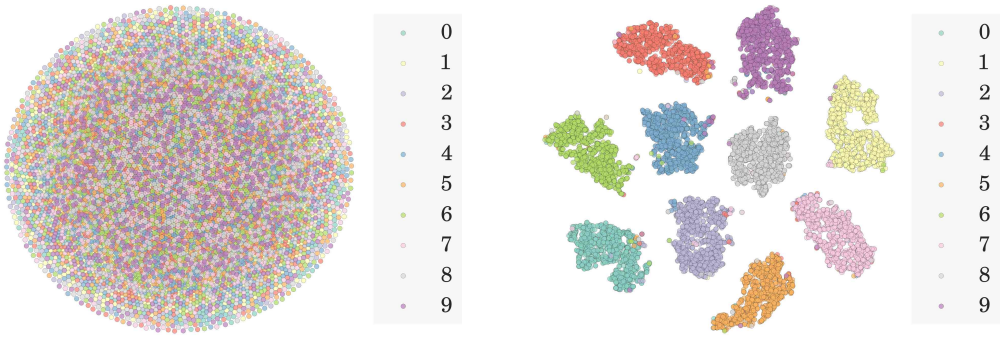
for different patterns. We hypothesize that the network acts as a model that can switch between "submodels" (subnetworks) such that *similar submodels respond to similar patterns*, and the relative ease of this switching behavior for locally competitive networks makes them easier to train. As evidence of this behavior, we analyze the activated subnetworks for a large subset of a dataset (which is not used for training) and show that the subnetworks activated for different input patterns exhibit a structure consistent with our hypothesis. These observations provide a unified explanation for improved credit assignment in locally competitive networks during training, which is believed to be the main reason for their success. Our new point of view suggests a link between these networks and competitive learning approaches of the past decades. We also show that a simple encoding of which units in a layer are activated for a given example (its subnetwork) can be used to represent the example for retrieval tasks. Experiments on MNIST, CIFAR-10, CIFAR-100 and the ImageNet dataset show that promising results are obtained for datasets of varying size and complexity.

2 Locally Competitive Neural Networks

Neural networks with activation functions like rectified linear, maxout and LWTA are locally competitive. This means that local competition among units in the network decides which parts of it get activated or trained for a particular input example. For each unit, the total input or presynaptic activation z is first computed as $z = \mathbf{w}\mathbf{x} + b$, where $\mathbf{x}$ is the vector of inputs to the unit, $\mathbf{w}$ is a trainable weight vector, and b is a trainable bias. For the rectified linear function, the output or postsynaptic activation of each unit is simply $max(z, 0)$, which can be interpreted as competition with a fixed value of 0. For LWTA, the units in a layer are considered to be divided into blocks of a fixed size. Then the output of each unit is Iz where I is an indicator which is 1 if the unit has the maximum z in its group and 0 otherwise. In maxout, the inputs from a few units compete using a max operation, and the block output is the maximum z among the units[1]. A maxout block can also be interpreted as an LWTA block with shared outgoing weights among the units. A comparison of the 3 activation functions is shown in Figure 1.

In each of the three cases, there is a local gating mechanism which allows non-zero activations (and errors during training) to propagate only through part of the network, i.e. a subnetwork. Consider the activation of a neural network with ReLUs in a single hidden layer. For each

[1]In our terminology, the terms *unit* and *block* correspond to the terms *filter* and *units* in [2].



(a) (b)

Figure 3: 2-D visualization of submasks from the penultimate layer of a 3 hidden layer network with ReLUs on the MNIST test set. (a) shows the submasks from an *untrained* network layer which lacks any discernable structure. (b) shows submasks from a trained network layer, showing clearly demarcated clusters relevant to the supervised learning task. 'Mistakes' made by the network can also be observed, such as mistaking 4's for 9's.

input pattern, the subset of units with non-zero activations in the hidden layer form a subnetwork, and an examination of the subnetworks activated for several examples shows that a large number of different subnetworks are activated (Figure 2). The result of training the network can interpreted in the following way: when training a single network with a local gating mechanism, a large number of linear subnetworks with shared parameters are trained on the dataset such that different examples are gated to different subnetworks, each getting trained to produce the desired output. At test time, the system generalizes in the sense that the appropriate subnetwork for a given example is activated.

Note that this *model of models* interpretation is different from the model averaging perspective of dropout [11] which posits that using dropout is equivalent to averaging the predictions of many subnetworks on the same data. Our interpretation is related to mixtures of local experts [12], where a gating network was trained to select one of many subnetworks or modules in a system. We come back to this relationship in Section 5.

3 Subnetwork Analysis

This section investigates how the model of models that is implemented though local competition self-organizes due to training. In order to visualize the organization of subnetworks as a result of training, they are encoded as bit strings called *submasks*. For the input pattern i, the submask $s_i \in \{0,1\}^u$, where u is the number of units in the full network, represents the corresponding subnetwork by having a 0 in position j, $j = 1..u$, if the corresponding unit has zero activation, and 1 otherwise. The submasks uniquely and compactly encode each subnetwork in a format that is amenable to analysis through clustering, and, as we show in Section 4.2, facilitates efficient data retrieval.

In what follows, the subnetworks that emerge during training are first visualized using the t-SNE [13] algorithm. This dimensionality reduction technique enables a good visualization of the relationship between submasks for several examples in a dataset by preserving the local structure. Later in this section, we examine the evolution of subnetworks during training, and show that the submasks obtained from a trained network can directly be used for classification using a simple nearest neighbors approach. All experiments in this section are performed on the MNIST dataset [14]. This familiar dataset was chosen because it is relatively easy, and therefore provides a tractable setting in which to verify the repeatability of our results. Larger, more interesting datasets are used in section 4 to demonstrate the utility of techniques developed in this section for classification and retrieval.

3.1 Visualization through Dimensionality Reduction

For visualizing the relationship between submasks for a large number of input patterns, we trained multiple networks with different activation functions on the MNIST training set, stopping when the error on a validation set did not improve. The submasks for the entire test set (10k examples) were then extracted and visualized using t-SNE. Since the competition between subnetworks is local and not global, subsets of units in deeper (closer to the output) layers are activated based on information extracted in the shallow layers. Therefore, like unit activations, submasks from deeper layers are expected to be better related to the task since deeper layers code for higher level abstractions. For this reason, we use only submasks extracted from the penultimate network layers in this paper, which considerably reduces the size of submasks to consider.

Figure 3b shows a 2D visualization of the submasks from a 3 hidden layer ReL network. Each submask is a bitstring of length 1k (the size of the network's penultimate layer). Ten distinct clusters are present corresponding to the ten MNIST classes. It is remarkable that, irrespective of the actual activation values, the subnetworks which are active for the testing examples can be used to visually predict class memberships based on their similarity to each other. The visualization confirms that the subnetworks active for examples of the same class are much more similar to each other compared to the ones activated for the examples of different classes.

Visualization of submasks from the same layer of a randomly initialized network does not show any structure (Figure 3a), but we observed some structure for the untrained first hidden layer (Appendix A). For trained networks, similar clustering is observed in the submasks from shallow layers in the network, though the clusters appear to be less separated and tight. The visualization also shows many instances where the network makes mistakes. The submasks for some examples lie in the cluster of submasks for the wrong class, indicating that the 'wrong' subnetwork was selected for these examples. The experiments in the next sections show that the organization of subnetworks is indicative of the classification performance of the full network.

Other locally competitive activation functions such as LWTA and maxout result in similar clustering of submasks (visualizations included in Appendix A). For LWTA layers, the submasks can be directly constructed from the activations because there is no subsampling when going from presynaptic to postsynaptic activations, and it is reasonable to expect a subnetwork organization similar to that of ReL layers. Indeed, in a limited qualitative analysis, it has been shown previously [3] that in trained LWTA nets there are more units in common between subnetworks for examples of the same class than those for different class examples.

For maxout layers, the situation is trickier at a first glance. The unit activations get pooled before being propagated to the next layer, so it is possible that the maximum activation value plays a much more important role than the identity of the winning units. However, using the same basic principle of credit assignment to subnetworks, we can construct submasks from maxout layers by binarizing the unit activations such that only the units producing the maximum activation are represented by a 1. Separation of subnetworks is necessary to gain the advantages of local competition during learning, and the visualization of the generated submasks produces results similar to those for ReLU and LWTA (included in Appendix A).

3.2 Behavior during Training

In order to measure how the subnetworks evolve over the course of training, the submasks of each sample in the training set were recorded at each epoch. Figure 4 characterizes the change in the subnets over time by counting the number of input patterns for which a unit flips from being on to being off, or vice-versa, from one epoch to the next. The curve in the figure shows the fraction of patterns for which an inter-epoch flip occurred, averaged across all units in the network. Higher values indicate that the assignment of subnets to patterns is not stable. The batch size for this experiment was 100, which means that each pass over the training set consists of 500 weight updates. For the run shown, the average fraction of flips starts at 0.2, but falls quickly below 0.05 and keeps falling as training continues, indicating that during training, the assignment of subnetworks to individual examples stabilizes quickly

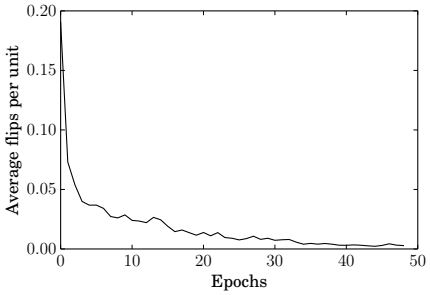


Figure 4: The plot shows the mean fraction of examples (total 10k) for which units in the layer flip (turn from being active to inactive or vice-versa) after every pass through the training set. The units flip for upto 20% of the examples on average in the first few epochs, but quickly settle down to less than 5%.

Network	No. test errors	
	Softmax	kNN
ReL (no dropout)	161	158
LWTA (dropout)	142	154
Maxout (dropout)	116	131

Table 1: Some examples of classification results on the permutation invariant MNIST test set using softmax layer outputs vs. kNN on the submasks. All submasks are extracted from the penultimate layer. kNN results are close to the softmax results in each case. The maxout network was additionally trained on the validation set. Results vary slightly across experimental runs and were not cherry-picked for reporting.

in this case. After a brief ($\sim$3 epochs) transient period, a fine-tuning period follows where the selected subnetworks keep getting trained on their corresponding training patterns.

3.3 Classification/Retrieval using Submasks

Since the visualization of submasks for the test set shows task-relevant structure, it is natural to ask: how well can the submask represent the data that produced it? If the submasks for similar examples are similar, perhaps they can be used as data descriptors for tasks such as similarity-based retrieval. Sparse binary codes enable efficient storage and retrieval for large and complex datasets due to which learning to produce them is an active research area [15, 16, 17]. This would make representative submasks very attractive since no explicit training for retrieval would be required to generate them.

To evaluate if examples producing similar binary codes are indeed similar, we train locally competitive networks for classification and use a simple k nearest neighbors (kNN) algorithm for classifying data using the generated submasks. This approach is a simple way to examine the amount of information contained in the submasks (without utilizing the actual activation values). Further experiments using submasks in a more conventional retrieval setting are currently underway.

We trained networks with fully connected layers on the MNIST training set, and selected the value of k with the lowest validation error to perform classification on the test set. Results are shown in Table 1. In each case, the kNN classification results are close to the classification result obtained using the network's softmax layer. For comparison, using the (non-pooled) unit activations from the maxout network instead of submasks for kNN classification results in 121 errors.

Submasks can also be obtained from convolutional layers. Using a convolutional maxout network, we obtained 52 errors on the MNIST test set when we reproduced the model from [2]. Since the penultimate layer in this model is convolutional, the submasks were constructed using the presynaptic unit activations from this layer for all convolutional maps. Visualization of these submasks showed similar structure to that obtained from fully connected layers, kNN classification on the submasks resulted in 65 errors. As seen before, for a well-trained network the kNN performance is close to the performance of the network's softmax layer.

3.4 Effect of Dropout

The dropout [10] regularization technique has proven to be very useful and efficient at improving generalization for large models, and is often used in combination with locally competitive activation functions [2, 4, 5]. We found that networks which were trained with dropout (and thus produced lower test set error) also yielded better submasks in terms of kNN classification performance. To observe the effect of dropout in more detail, we

Dataset	Softmax	kNN (activations)	kNN (submasks)
CIFAR-10	9.61%	9.79%	11.19%
CIFAR-100	34.54%	41.22%	42.97%

Table 2: Classification results on CIFAR datasets comparing network performance, kNN on continuous activation values, and kNN on binary submasks.

trained a 3 hidden layer network with 800 ReLUs in each hidden layer without dropout on MNIST starting from 5 different initializations until the validation set error did not improve. The networks were then trained again from the same initialization with dropout until the validation error matched or fell below the lowest validation error from the non-dropout case. In both cases, minibatch gradient descent with momentum was used for training the networks. A comparison of kNN classification error for the dropout and non-dropout cases shows that when the validation errors are similar, the organization of subnetworks is not significantly better or worse. If dropout training is stopped at a point when validation error is similar to a no-dropout network, the submasks from both cases give similar results, but as dropout improves generalization (lowers validation set error), the organization of subnetworks also improves.

This supports the interpretation of dropout as a regularization technique which prevents "co-adaptation of feature detectors" (units) [10], leading to better representation of data by the subnetworks. Dropout improves generalization by injecting noise in the organization of subnetworks, making them more robust. Due to this effect, the remaining results and visualizations in this paper are all derived from networks trained with dropout.

4 Experimental Results

The following experiments apply the methods described in the previous section to more challenging benchmark problems: CIFAR-10, CIFAR-100, and ImageNet. For the CIFAR experiments, we used the models described in [2] since they use locally competitive activations (maxout), are trained with dropout, and good hyperparameter settings for them are available [18]. We report the classification error on the test set obtained using the softmax output layer, as well kNN classification on the penultimate layer unit activations and submasks. The best value of k is obtained using a validation set, though we found that $k = 5$ with distance weighting usually worked well.

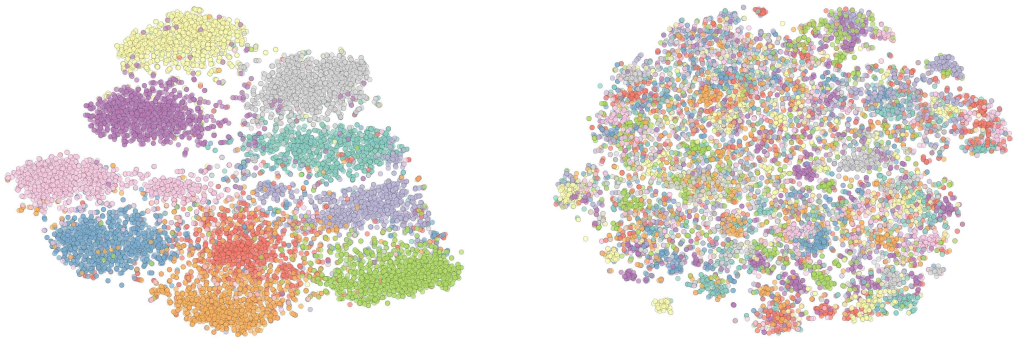
4.1 CIFAR-10 & CIFAR-100

CIFAR-10 is a dataset of 32×32 color images of 10 classes split into a training set of size 50k and testing set of size 10k (6k images per class) [19]. CIFAR-100 is a similar dataset of color images but with 100 classes and 600 images per class, making it more challenging. The results obtained on these datasets are summarized in Table 2. The models from [2] for these dataset utilize preprocessing using global contrast normalization and ZCA whitening as well as data augmentation using translational and horizontal reflections.

We find that when comparing nearest neighbor classification performance with submasks to unit activation values, we lose an accuracy of 1.4% on the CIFAR-10 dataset, and 1.75% on the CIFAR-100 dataset. This indicates a good extent of subnetwork organization according to different classes. Figure 5a shows the 2-D visualization of the test set submasks for CIFAR-10. Some classes can be seen to have highly representative submasks, while confusion between classes in the lower half is observed. The clusters of subnetworks are not as well-separated as in the case of MNIST, reflecting the relatively worse classification performance. Submask visualization for CIFAR-100 (Figure 5b) reflects the high error rate in this dataset. Although any visualization with 100 classes can be hard to interpret, many small clusters of submasks can still be observed.

4.2 ImageNet

The results of kNN classification and t-SNE visualization using submasks on small datasets of varying complexities show that the submasks contain substantial information about the data relevant to the task. In this section, the utility of the submasks obtained for a large



(a) (b)

Figure 5: 2-D visualizations of the submasks from the penultimate layer of the trained maxout networks reproduced from [2]. (a) The CIFAR-10 test set. The 10-cluster structure is visible, although the clusters are not as well separated as in the case of MNIST. This corresponds to the higher error rates obtained using both kNN and the network's output layer. (b) The CIFAR-100 test set. This is a more difficult task with 100 classes, but several clusters are still visible. The separation between clusters is much worse, which is reflected in the high classification error obtained.

convolutional network trained on the ImageNet Large Scale Visual Recognition Challenge 2012 (ILSVRC-2012) [20] dataset is evaluated.

ILSVRC-2012 is a dataset of over a million natural images split into 1000 classes. An implementation of the network in [4], with some differences [21], is available publicly. For the experiments in this section, the penultimate-layer activations obtained using this model were downloaded from CloudCV [22]. The activations were obtained using the center-only option, meaning that only the activations for the central, 224×224 crop of each image were used. Each image in the training and validation set can thus be represented using a 4096 dimensional submask, which is much more efficient for storage and retrieval than a floating-point vector of activations. Using these submasks, the top-1 and top-5 classification errors obtained using kNN classification for the validation set are **56.7%** and **29.2%**. For each validation set example, 100 examples from the training set with the closest submasks were weighted by the inverse of the distance, then the classes with top-1 or top-5 weighted sums were returned as predictions. The results obtained using the network's softmax outputs[2] are 42.9% and 19.2%. We only report errors on the validation set since the true labels for the test set are not public. Our ongoing experiments compare the utility of these submasks to other recently proposed algorithms which are designed to obtain binary data descriptors.

The classification results show that the performance on this hard classification task is very good, considering the gain in efficiency obtained from binarization of the activations. For instance, submasks for the full ILSVRC-2012 training set can be stored in about 0.5 GB. Moreover, our experiments indicate that submasks obtained from a better trained network will result in even better performance, since quality of submasks improves as network training continues. In [4], it was first shown that the activations from the penultimate layer of the deep network can be used to retrieve similar images. Since retrieval using real-valued vectors is inefficient, it was suggested that the activations can be compressed to binary codes using auto-encoders. However, the submasks can be directly utilized for quick and efficient retrieval of data based on high level similarity. Sample retrieval results for examples from the ILSVRC-2012 dataset are shown in Figure 6, where the first image in each row is a query image from the validation set and the rest are images from the training set with the most similar submasks to the query image. The returned images are often very relevant to the query images, suggesting a new recipe for retrieval tasks in general: one can train a

[2]The network's error is reported for classification using 5 crops for each image and their horizontal reflections, not with the center-only option which we used. This improves the network's error by about 1-2% [4].

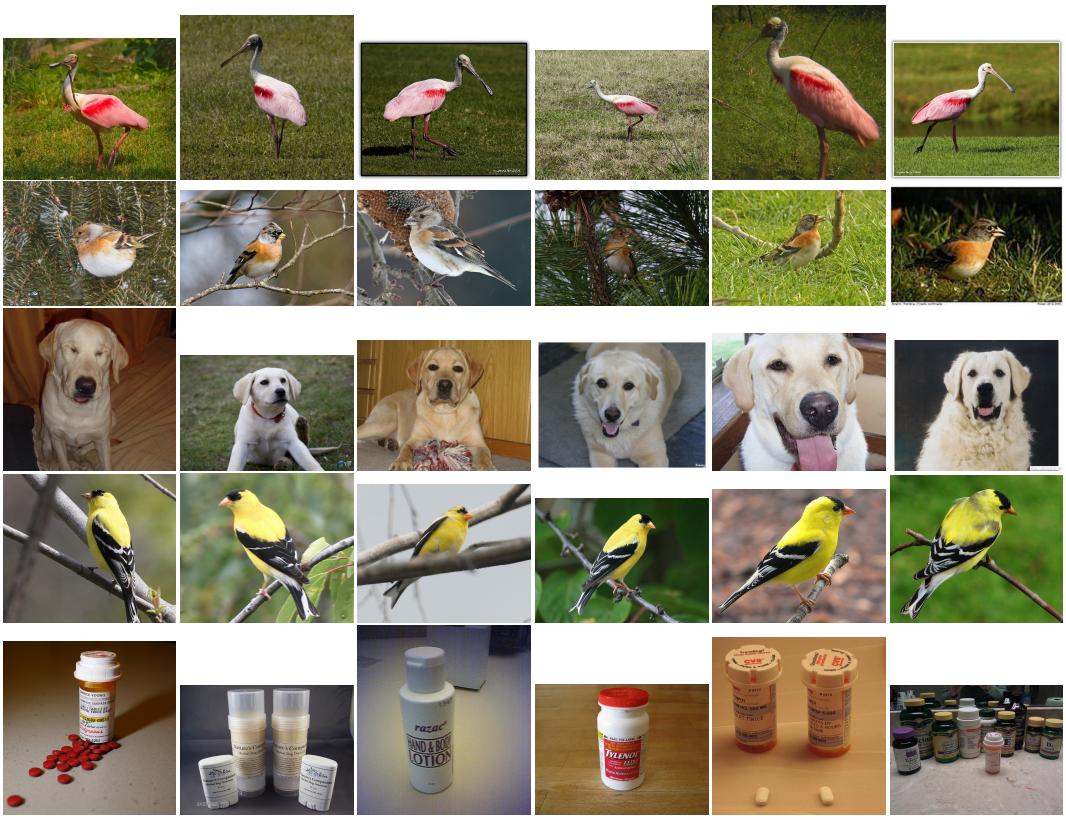


Figure 6: Retrieval based on subnetworks on the ILSVRC-2012 dataset. The first image in each row is the query image; the remaining 5 are the responses retrieved using submasks.

neural network with locally competitive units on a dataset for a high level task, extract the submasks from the penultimate layers, and directly utilize them for retrieval.

5 Discussion

Training a system of many networks on a dataset such that they specialize to solve simpler tasks can be quite difficult without combining them into a single network with locally competitive units. Without such local competition, one needs to have a global gating mechanism as in [12]. The training algorithm and the objective function also need modifications such that competition between networks is encouraged. On the other hand, a locally competitive neural network can behave like a model composed of many subnetworks, and massive sharing of parameters between subnetworks enables better training. Stochastic gradient descent can be used to minimize the desired loss function, and the implementation is so simple that one does not even realize that a model of models is being trained.

Figure 4 suggests that during optimization, the subnetworks get organized during an early transient phase such that subnetworks responding to similar examples have more parameters in common than those responding to dissimilar examples. This allows for better training of subnetworks due to reduced interference from dissimilar examples and shared parameters for similar examples. In the later fine-tuning phase, the parameters of subnetworks get adjusted to improve classification and much less re-assignment of subnetworks is needed. In this way, the gating mechanism induced by locally competitive activation functions accomplishes the purpose of global competition efficiently and no modifications to the error function are required.

Due to above advantages, networks with such activation functions can usually be trained faster and better compared to networks with sigmoidal or similar activation functions for complex pattern recognition tasks. The nature of organization of subnetworks is reminiscent

of the data manifold hypothesis for classification [23]. Just like data points of different classes are expected to concentrate along sub-manifolds, we expect that the organization of subnetworks that respond to the data points reflects the data manifold being modeled.

An important take-away from these results is the unifying theme between locally competitive architectures, and its relation to past work on competitive learning. Insights from past literature on this topic can be utilized to develop improved learning algorithms and architectures for locally competitive learning. To the best of our knowledge, this paper is the first to show that simply training a deep network for classification results in binary descriptors that are useful for retrieval. These descriptors are not just results of a thresholding trick or unique to a particular activation function, but arise as a direct result of the way the network learns and processes information. Our experiments on datasets of increasing complexity show that when the network performance (softmax classification) improves, the performance gap to submask-based retrieval closes. This suggests that in the near future, as training techniques continue to advance and yield lower errors on larger datasets, submasks will perform as well as activation values for retrieval and transfer learning tasks. Importantly, these binary representation will always be far more efficient for storage and retrieval than continuous activation vectors.

Acknowledgments

This research was funded by EU projects WAY (FP7-ICT-288551) and NASCENCE (FP7-ICT-317662). We thank Jan Koutník and Klaus Greff for their helpful comments.

References

[1] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier networks. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics. JMLR W&CP*, volume 15, page 315–323, 2011.

[2] Ian Goodfellow, David Warde-Farley, Mehdi Mirza, Aaron Courville, and Yoshua Bengio. Maxout networks. In *Proceedings of The 30th International Conference on Machine Learning*, pages 1319–1327, 2013.

[3] Rupesh K. Srivastava, Jonathan Masci, Sohrob Kazerounian, Faustino Gomez, and Jürgen Schmidhuber. Compete to compute. In *Advances in Neural Information Processing Systems*, page 2310–2318, 2013.

[4] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, 2012.

[5] Matthew D. Zeiler, M. Ranzato, R. Monga, M. Mao, K. Yang, Q. V. Le, P. Nguyen, A. Senior, V. Vanhoucke, and J. Dean. On rectified linear units for speech processing. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, page 3517–3521. IEEE, 2013.

[6] Ian J. Goodfellow, Mehdi Mirza, Xiao Da, Aaron Courville, and Yoshua Bengio. An empirical investigation of catastrophic forgetting in gradient-based neural networks. In *International Conference on Learning Representations*, 2014.

[7] Razvan Pascanu, Guido Montufar, and Yoshua Bengio. On the number of response regions of deep feed forward networks with piece-wise linear activations. *arXiv:1312.6098 [cs]*, December 2013. arXiv: 1312.6098.

[8] Jürgen Schmidhuber. A local learning algorithm for dynamic feedforward and recurrent networks. *Connection Science*, 1(4):403–412, 1989.

[9] Jürgen Schmidhuber. Self-delimiting neural networks. *arXiv preprint arXiv:1210.0118*, 2012.

[10] Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv:1207.0580 [cs]*, July 2012.

[11] Pierre Baldi and Peter Sadowski. The dropout learning algorithm. *Artificial Intelligence*, 210:78–122, May 2014.

[12] Robert A. Jacobs, Michael I. Jordan, Steven J. Nowlan, and Geoffrey E. Hinton. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87, 1991.

[13] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(11), 2008.

[14] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[15] Yunchao Gong, Sanjiv Kumar, Henry A. Rowley, and Svetlana Lazebnik. Learning binary codes for high-dimensional data using bilinear projections. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, page 484–491. IEEE, 2013.

[16] Jonathan Masci, Alex M. Bronstein, Michael M. Bronstein, Pablo Sprechmann, and Guillermo Sapiro. Sparse similarity-preserving hashing. In *International Conference on Learning Representations*, 2014. arXiv: 1312.5479.

[17] Kristen Grauman and Rob Fergus. Learning binary hash codes for large-scale image search. In *Machine Learning for Computer Vision*, page 49–87. Springer, 2013.

[18] Ian J. Goodfellow, David Warde-Farley, Pascal Lamblin, Vincent Dumoulin, Mehdi Mirza, Razvan Pascanu, James Bergstra, Frédéric Bastien, and Yoshua Bengio. Pylearn2: a machine learning research library. *arXiv:1308.4214 [cs, stat]*, August 2013.

[19] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. *Computer Science Department, University of Toronto, Tech. Rep*, 2009.

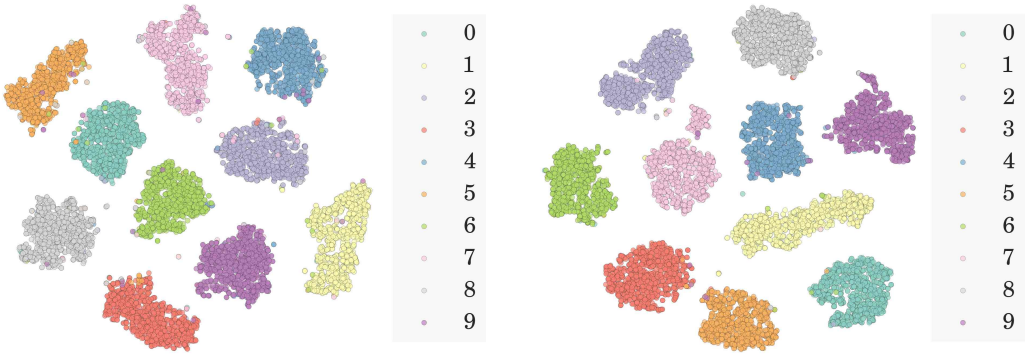
[20] Jia Deng, Alex Berg, Sanjeev Satheesh, Su Hao, Aditya Khosla, and Fei-Fei Li. ImageNet large scale visual recognition competition 2012 (ILSVRC2012). http://www.image-net.org/challenges/LSVRC/2012/, 2012.

[21] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. DeCAF: a deep convolutional activation feature for generic visual recognition. *arXiv:1310.1531 [cs]*, October 2013.

[22] D. Batra, H. Agrawal, P. Banik, N. Chavali, and A. Alfadda. CloudCV: Large-Scale distributed computer vision as a cloud service. http://www.cloudcv.org, September 2013.

[23] Salah Rifai, Yann Dauphin, Pascal Vincent, Yoshua Bengio, and Xavier Muller. The manifold tangent classifier. In *Advances in Neural Information Processing Systems*, page 2294–2302, 2011.

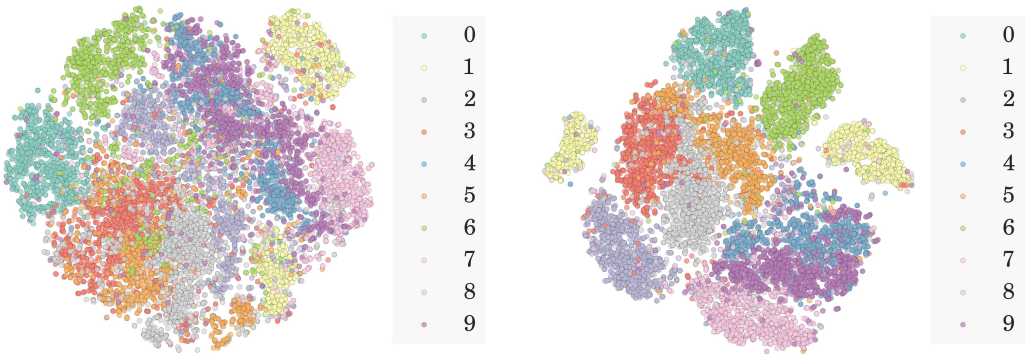
A Extra visualizations



(a) Trained LWTA layer.

(b) Trained Maxout layer.

Figure 7: 2-D visualization of submasks from the penultimate layer of 3 hidden layer LWTA and maxout networks on MNIST test set. Organization of submasks into distinct class specific clusters similar to ReL networks is observed.



(a) Untrained 1st LWTA layer.

(b) Untrained 1st ReL layer.

Figure 8: 2-D visualization of submasks obtained before training from the 1st (closest to the input) hidden layer of 3 hidden layer LWTA and ReL networks on MNIST test set.