Sparse Rectifier Neural Networks

Xavier Glorot and Yoshua Bengio
Dept. IRO, U. Montreal

Abstract. Rectifying neurons are more biologically plausible than sigmoid neurons, which are more biologically plausible than hyperbolic tangent neurons (which work better for training multi-layer neural networks than sigmoid neurons). We show that networks of rectifying neurons yield generally better performance than sigmoid or tanh networks while creating highly sparse representations with true zeros, in spite of the hard non-linearity and non-differentiablity at 0.

Introduction.
Despite their original connection, there is an important gap between the common artificial neural network models used in machine learning (such as those used in the recent surge of papers on deep learning, see (Bengio, 2009) for a review) and several neuroscience observations:

- Various studies on brain energy expense suggest that neurons encode information in a sparse and distributed way (Attwell and Laughlin, 2001), estimating the percentage of neurons active at the same time from 1 to 4% (Lennie, 2003).
- There are also important divergences regarding the non-linear activation functions assumed in learning algorithms and in computational neuroscience. For example, with 0 input, the sigmoid has an output of $\frac{1}{2}$, therefore, after initializing with small weights, all neurons fire at half their saturation frequency. This is biologically implausible and also hurts gradient-based optimization (LeCun et al., 1998; Bengio and Glorot, 2010). The hyperbolic tangent has an output of 0 at 0, and is therefore preferred from the optimization standpoint (LeCun et al., 1998; Bengio and Glorot, 2010), but it forces a symmetry around 0 that is not present in biological neurons. Neuroscience models of neurons spiking rate in function of their input current are one-sided, have a strong saturation near 0 for their threshold current, and a slow saturation to the maximum firing rate at important currents. In addition, the neuroscience literature (Bush and Sejnowski, 1995; Douglas and al., 2003) indicates that cortical neurons are rarely in their saturation regime and can be approximated as rectifiers.

We propose to explore the use of rectifying non-linearities as alternatives to the sigmoidal (or hyperbolic tangent) ones, in deep artificial neural networks, using an L_1 sparsity regularizer to prevent potential numerical problems with unbounded activation.

From the computational point of view, sparse representations have advantageous mathematical properties, like **information disentangling** (different explanatory factors do not have to be compactly entangled in a dense representation) and **efficient variable-size representation** (the number of non-zeros may vary for different inputs). Sparse representations are also **more likely to be linearly separable** (or more easily separable with less non-linear machinery). Learned sparse representations have been the subject of much previous work (Olshausen and Field, 1997; Doi, Balcan and Lewicki, 2006; Ranzato et al., 2007; Ranzato and LeCun, 2007; Ranzato, Boureau and LeCun, 2008; Mairal et al., 2009), and this work is particularly inspired by the sparse representations learned in the context of auto-encoders variants, since auto-encoders have been found to be very useful to train deep architectures (Bengio, 2009). In our experiments, we explore denoising auto-encoders (Vincent et al., 2008) for unsupervised pre-training, but using rectifying non-linearities in the hidden layers. Note that for an equal number of neurons, sparsity may hurt performance because it reduces the effective capacity of the model.
The rectifier function $\max(0, x)$ is one-sided and therefore does not enforce a sign symmetry (like does the absolute value non-linearity $|x|$ used in (Jarrett et al., 2009)) or antisymmetry (like does a $\tanh(x)$ non-linearity). Nevertheless, we can still obtain symmetry or antisymmetry by combining two rectifier units.
The rectifier activation function has the benefit of being linear by parts, so the computation of activations is computationally cheaper, and the propagation of gradients is easier on the active paths (there is no gradient vanishing

effect on the active paths, as may be seen with deep networks of sigmoid or tanh units). On the other hand, the hard saturation at 0 may completely block the gradients and make optimization harder. To evaluate the importance of this potential problem, the softplus (softplus$(x) = \log(1 + e^x)$), a smooth version of the rectifier, will also be investigated.

Finally, the activation of the rectifier being unbounded, the activations and gradients could get to be arbitrary large. Hence, the L_1 sparsity penalty on the activations does not have only the role to enforce sparsity but also to keep the weights and the activations small. The universal approximation theorem for multi-layer networks concerns only bounded functions, but we can show that it applies in this case because we can easily create a bounded activation function by adding two rectifier units.

Experimental results.

Three different classification experiments have been performed. For all of them the model has three hidden layers with 1000 units per layer (selected, along with learning rates and L1 penalty using a validation set). The supervised cost is the negative log likelihood of the softmax logistic regression (the output layer). The first experiment is with 10000 training examples from *Shapeset*, a dataset of grey-level images of simple 2-D shapes (triangles, rectangles and ellipses) (Bengio et al., 2009). 10000 examples are used for model selection and 10000 as a test set. The second and the third experiments are with the *MNIST* digit images (50000/10000/10000 used for training, model selection and test), respectively with and without unsupervised pre-training (with a denoising auto-encoder (Vincent et al., 2008)).

Table 1: Left: average sparsity (fraction of zeros) of the hidden layers representation with rectifier units for MNIST without pre-training, with or without L1 penalty. Right: test error on *Shapeset* and *MNIST* without pre-training.

Layer	**With L1**	**w/o L1**
Layer 1	93.94%	52.23%
Layer 2	80.92%	43.86%
Layer 3	76.66%	37.18%

Neuron type	**Shapeset**	**MNIST**
Rectifier + L1 penalty	28.62%	1.56%
Rectifier	29.86%	1.7%
Softplus	36.58%	1.55%
Tanh	36.64%	1.66%

Table 2: Average error of the 5 best models on MNIST with denoising auto-encoder pretraining

Neuron type	**Test**	**Valid**
Rectifier + L1 penalty	1.16±0.07%	1.15±0.07%
Tanh	1.30±0.12%	1.18±0.05%

First of all, on *Shapeset* we have a considerable gain using the rectifying activation function (Table 1 (right)). *Shapeset* images have variations in contrast and foreground/background grey levels, and we hypothesize that this is more easily handled with rectifier units, which can propagate continuous-valued quantities, by opposition to sigmoids or hyperbolic tangents, which saturate and are better suited to represent binary features (or probabilities over binary events).

An advantage of using the rectifying activation function is that we easily obtain a sparse representation with real zeros. As shown in Table 1 (left) the amount of sparsity is quite important, with around 80 or 90% zeros. Surprisingly, for the same number of hidden units, the sparsity constraint usually gives better generalization error, even though it should reduce capacity, and even though the tanh network was not overfitting.

The soft variant of the rectifier (*softplus*, i.e. with continuous gradients) does not seem to help more than the hard rectifier (in fact it is usually worse) and it does not produce a sparse representation.

Performance can be further improved by unsupervised pre-training of each layer. The input to the denoising auto-encoder (with a single hidden layer that is the code layer) is corrupted (here by iid Gaussian noise) and the reconstruction is compared (by squared error, to be minimized) with the uncorrupted input (Vincent et al., 2008). Each layer of the deep network is initialized by pre-training it in this way, taking the output code of the previous layer as its input. The encoder and decoder weights are forced to be the transpose of each other, forcing the encoder to have a roughly norm-preserving Jacobian, a condition that we believe to be useful for gradient propagation in deep networks. The corruption process is only used during training of each auto-encoding layer. Substantially better results are obtained (Table 2) with pre-training, as expected. Interestingly, when using pre-training, a large amount of sparsity is achieved by rectifier units, even without L1 penalty.

References

Attwell, D. and Laughlin, S. (2001). An energy budget for signaling in the grey matter of the brain. *Journal of Cerebral Blood Flow and Metabolism*, 21(10):1133–1145.

Bengio, Y. (2009). Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, 2(1):1–127. Also published as a book. Now Publishers, 2009.

Bengio, Y. and Glorot, X. (2010). Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of AISTATS 2010*.

Bengio, Y., Louradour, J., Collobert, R., and Weston, J. (2009). Curriculum learning. In Bottou, L. and Littman, M., editors, *Proceedings of the Twenty-sixth International Conference on Machine Learning (ICML'09)*. ACM.

Bush, P. C. and Sejnowski, T. J. (1995). *The cortical neuron*. Oxford university press.

Doi, E., Balcan, D. C., and Lewicki, M. S. (2006). A theoretical analysis of robust coding over noisy overcomplete channels. In Weiss, Y., Schölkopf, B., and Platt, J., editors, *NIPS 18*, pages 307–314. MIT Press, Cambridge, MA.

Douglas, R. and al. (2003). Recurrent excitation in neocortical circuits. *Science*, 269(5226):981–985.

Jarrett, K., Kavukcuoglu, K., Ranzato, M., and LeCun, Y. (2009). What is the best multi-stage architecture for object recognition? In *Proc. International Conference on Computer Vision (ICCV'09)*. IEEE.

LeCun, Y., Bottou, L., Orr, G., and Muller, K. (1998). Efficient backprop.

Lennie, P. (2003). The cost of cortical computation. *Current Biology*, 13:493–497.

Mairal, J., Bach, F., Ponce, J., Sapiro, G., and Zisserman, A. (2009). Supervised dictionary learning. In Koller, D., Schuurmans, D., Bengio, Y., and Bottou, L., editors, *NIPS 21*, pages 1033–1040. NIPS Foundation.

Olshausen, B. A. and Field, D. J. (1997). Sparse coding with an overcomplete basis set: a strategy employed by V1? *Vision Research*, 37:3311–3325.

Ranzato, M., Boureau, Y.-L., and LeCun, Y. (2008). Sparse feature learning for deep belief networks. In Platt, J., Koller, D., Singer, Y., and Roweis, S., editors, *NIPS 20*, pages 1185–1192, Cambridge, MA. MIT Press.

Ranzato, M. and LeCun, Y. (2007). A sparse and locally shift invariant feature extractor applied to document images. In *Proc. ICDAR'07*, pages 1213–1217, Washington, DC, USA. IEEE Computer Society.

Ranzato, M., Poultney, C., Chopra, S., and LeCun, Y. (2007). Efficient learning of sparse representations with an energy-based model. In Schölkopf, B., Platt, J., and Hoffman, T., editors, *NIPS 19*, pages 1137–1144. MIT Press.

Vincent, P., Larochelle, H., Bengio, Y., and Manzagol, P.-A. (2008). Extracting and composing robust features with denoising autoencoders. In Cohen, W. W., McCallum, A., and Roweis, S. T., editors, *Proceedings of the Twenty-fifth International Conference on Machine Learning (ICML'08)*, pages 1096–1103. ACM.