Deep Sparse Rectifier Neural Networks

Xavier Glorot, Antoine Bordes and Yoshua Bengio
Dept. IRO
University of Montreal
Montreal, QC
{glorotxa,bordesa,bengioy}@iro.umontreal.ca

Abstract

Rectifying neurons are more biologically plausible than logistic sigmoid neurons, which are themselves more biologically plausible than hyperbolic tangent neurons. However, the latter work better for training multi-layer neural networks than logistic sigmoid neurons. This paper shows that networks of rectifying neurons yield equal or better performance than hyperbolic tangent networks in spite of the hard non-linearity and non-differentiability at zero and create sparse representations with true zeros which are remarkably suitable for naturally sparse data. Even though they can take advantage of semi-supervised setups with extra-unlabeled data, deep rectifier networks can reach their best performance without requiring any unsupervised pre-training on purely supervised tasks with large labeled datasets. Hence, these results can be seen as a new milestone in the attempts at understanding the difficulty in training deep but purely supervised neural networks, and closing the performance gap between neural networks learnt with and without unsupervised pre-training.

1 Introduction

Many differences exist between the neural network models used by machine learning researchers and those used by computational neuroscientists. This is in part because the objective of the former is to obtain computationally efficient learners, that generalize well to new examples, whereas the objective of the latter is to abstract out neuroscientific data while obtaining explanations of the principles involved, providing predictions and guidance for future biological experiments. Areas where both objectives coincide are therefore particularly worthy of investigation, pointing towards computationally motivated principles of operation in the brain that can also enhance research in artificial intelligence. In this paper we show that two common gaps between computational neuroscience models and machine learning neural network models can be bridged by using the following linear by part activation : $\max(0, x)$, called the rectifier (or hinge) activation function. Experimental results will show engaging training behavior of this activation function, especially for *deep architectures* (see Bengio (2009) for a review), i.e., where the number of hidden layers is 3 or more.

Recent theoretical and empirical work in statistical machine learning has demonstrated the importance of learning algorithms for deep architectures. This is in part inspired by observations of the mammalian visual cortex, which consists of a chain of processing elements, each of which is associated with a different representation of the raw visual input. This is particularly clear in the primate visual system (Serre *et al.*, 2007), with its sequence of processing stages: detection of edges, primitive shapes, and moving up to gradually more complex visual shapes. Interestingly, it was found that the features learned in deep architectures resemble those observed in the first two of these stages (in areas V1 and V2 of visual cortex) (Lee *et al.*, 2008), and that they become increasingly invariant to factors of variation (such as camera movement) in higher layers (Goodfellow *et al.*, 2009).

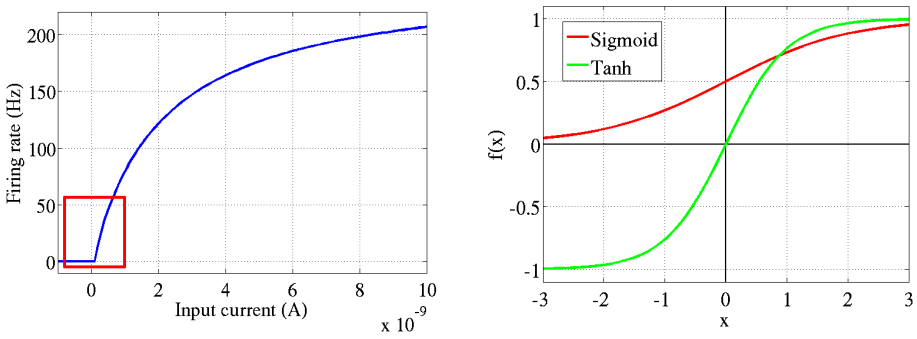

Figure 1: Left: common activation function motivated by biological data. Right: commonly used activation functions in neural networks literature: logistic sigmoid and hyperbolic tangent ($tanh$).

Regarding the training of deep networks, something that can be considered a breakthrough happened in 2006, with the introduction of Deep Belief Networks (Hinton *et al.*, 2006), and more generally the idea of initializing each layer by unsupervised learning (Bengio *et al.*, 2007; Ranzato *et al.*, 2007). Some authors have tried to understand why this unsupervised procedure helps (Erhan *et al.*, 2010) while others investigated why the original training procedure for deep neural networks failed (Bengio and Glorot, 2010). From the machine learning point of view, this paper brings additional results in these lines of investigation.

We propose to explore the use of rectifying non-linearities as alternatives to the hyperbolic tangent or sigmoids in deep artificial neural networks, in addition to using an L_1 regularizer to promote sparsity and prevent potential numerical problems with unbounded activation. Nair and Hinton (2010) present promising results of the influence of such units in the context of Restricted Boltzmann Machines compared to logistic sigmoid activations. Our work extends this for the case of pre-training using denoising auto-encoders (Vincent *et al.*, 2008) and provides an extensive empirical comparison of the rectifying activation function against the hyperbolic tangent on image classification benchmarks as well as an original derivation for the text application of sentiment analysis.

Our experiments indicate that training proceeds better when the artificial neurons are either off or operating mostly in a linear regime. Surprisingly, rectifying activation allows deep networks to achieve their best performance even without unsupervised pre-training. Hence, our work proposes a new contribution to the trend of understanding and merging the performance gap between deep networks learnt with and without unsupervised pre-training (Erhan *et al.*, 2010; Bengio and Glorot, 2010). Interestingly, rectifier networks can still benefit from unsupervised pre-training in the context of semi-supervised learning where large amounts of unlabeled data are provided. Furthermore, as rectifier units naturally lead to sparse networks and are closer to biological neurons' responses in their main operating regime, this work also bridges (in part) a machine learning / neuroscience gap in terms of activation function and sparsity.

2 Background

2.1 Neuroscience Observations

For models of biological neurons, the activation function is the expected firing rate as a function of the total input currently arising out of incoming signals at synapses (Dayan and Abott, 2001). An activation function is termed, respectively *antisymmetric* or *symmetric* when its response to the opposite of a strongly excitatory input pattern is respectively a strongly inhibitory or excitatory one, and *one-sided* when this response is zero. The main gaps that we wish to consider between computational neuroscience models and machine learning models are the following.

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 to be between 1 and 4% (Lennie, 2003). This corresponds to a trade-off between richness of representation and small action potential energy expenditure. Without additional regularization, such as an L_1 penalty, ordinary feedforward neural nets do not have this property. For example, the sigmoid activation has a steady state regime around $\frac{1}{2}$, therefore, after initializing with small weights, all neu-

rons fire at half their saturation regime. This is biologically implausible *and* hurts gradient-based optimization (LeCun *et al.*, 1998; Bengio and Glorot, 2010).

Important divergences between biological and machine learning models concern non-linear activation functions. The leaky integrate-and-fire (or *LIF*) (Dayan and Abott, 2001), gives the following relation between the firing rate and the input current, illustrated in Figure 1 (left):

$$f(I) = \begin{cases} \left[\tau \log\left(\frac{E+RI-V_r}{E+RI-V_{th}}\right) + t_{ref}\right]^{-1}, \\ \qquad \text{if } E + RI > V_{th} \\ 0\,, \qquad \text{if } E + RI \leq V_{th} \end{cases}$$

where t_{ref} is the refractory period (minimal time between two action potentials), I the input current, V_r the resting potential and V_{th} the threshold potential (with $V_{th} > V_r$), and R, E, τ the membrane resistance, potential and time constant. The most commonly used activation functions in the deep learning and neural networks literature are the standard *logistic sigmoid* and the *hyperbolic tangent* (see Figure 1, right). The hyperbolic tangent has a steady state at 0, and is therefore preferred from the optimization standpoint (LeCun *et al.*, 1998; Bengio and Glorot, 2010), but it forces an antisymmetry around 0 which is absent in biological neurons.

2.2 Advantages of Sparsity

Sparsity has become a concept of interest, not only in computational neuroscience and machine learning but also in statistics and signal processing (Candes and Tao, 2005). It was first introduced in computational neuroscience in the context of sparse coding in the visual system (Olshausen and Field, 1997). It has been a key element of deep convolutional networks exploiting a variant of auto-encoders (Ranzato *et al.*, 2007, 2008; Mairal *et al.*, 2009) with a sparse distributed representation, and has also become a key ingredient in Deep Belief Networks (Lee *et al.*, 2008). A sparsity penalty has been used in several computational neuroscience (Olshausen and Field, 1997; Doi *et al.*, 2006) and machine learning models (Lee *et al.*, 2007; Mairal *et al.*, 2009), in particular for deep architectures (Lee *et al.*, 2008; Ranzato *et al.*, 2007, 2008). However, in the latter, the neurons end up taking small but non-zero activation or firing probability. We show here that using a rectifying non-linearity gives rise to real zeros of activations and thus truly sparse representations. From a computational point of view, such representations are appealing for the following reasons:

- **Information disentangling**. One of the claimed objectives of deep learning algorithms (Bengio, 2009) is to disentangle the factors explaining the variations in the data. A dense representation is highly entangled because almost any change in the input modifies most of the entries in the representation vector. Instead, if a representation is both sparse and robust to small input changes, the set of non-zero features is almost always roughly conserved by small changes of the input.
- **Efficient variable-size representation.** Different inputs may contain different amounts of information and would be more conveniently represented using a variable-size data-structure, which is common in computer representations of information. *Varying the number of active neurons allows a model to control the effective dimensionality of the representation for a given input and the required precision*.
- **Linear separability**. Sparse representations are also more likely to be linearly separable, or more easily separable with less non-linear machinery, simply because the information is represented in a high-dimensional space. Besides, this can reflect the original data format. In text-related applications for instance, the original raw data is already very sparse (see Section 4.2).
- **Distributed but sparse**. Dense distributed representations are the richest representations, being potentially exponentially more efficient than purely local ones (Bengio, 2009). Sparse representations' efficiency is still exponentially greater, with the power of the exponent being the number of non-zero features. They may represent a good trade-off with respect to the above criteria.

Nevertheless, forcing too much sparsity may hurt predictive performance for an equal number of neurons, because it reduces the effective capacity of the model.

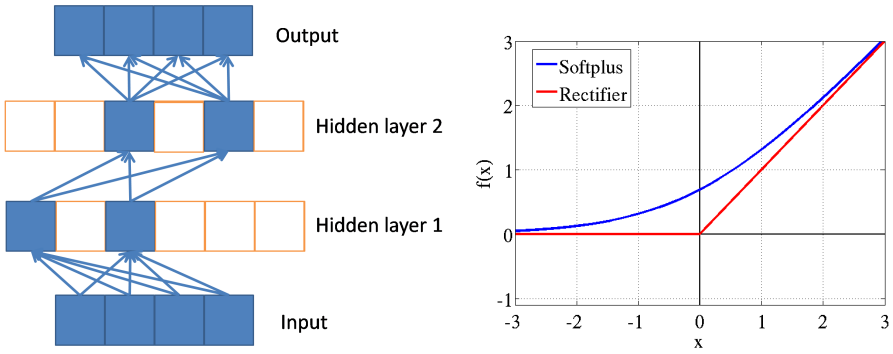

Figure 2: Left: sparse propagation of activations and gradients in a network of rectifier units. The input selects a subset of active neurons and computation is linear in this subset. Right: The rectifier and the softplus activation functions, the second one is a smooth version of the first.

3 Deep Rectifier Networks

3.1 Rectifier Neurons

The neuroscience literature (Bush and Sejnowski, 1995; Douglas and al., 2003) indicates that *cortical neurons are rarely in their maximum saturation regime*, and suggests that their activation function can be approximated by a rectifier. Most previous studies of neural networks involving a rectifying activation function concern recurrent networks (Salinas and Abbott, 1996; Hahnloser, 1998). The rectifier function $\text{rectifier}(x) = \max(0, x)$ is one-sided and therefore does not enforce a sign symmetry[1] or antisymmetry[1]: instead, the response to the opposite of an excitatory input pattern is 0 (no response). However, we can obtain symmetry or antisymmetry by combining two rectifier units sharing parameters.

Advantages The rectifier activation function allows a network to easily obtain sparse representations. For example, after uniform initialization of the weights, around 50% of hidden units continuous output values are real zeros, and this fraction can easily increase with sparsity-inducing regularization. Apart from being more biologically plausible, sparsity also leads to mathematical advantages (see previous section).

As illustrated in Figure 2 (left), the only non-linearity in the network comes from the path selection associated with individual neurons being active or not. For a given input *only a subset of neurons are active*. Computation is *linear* on this subset: once this subset of neurons is selected, the output is a linear function of the input (although a large enough change can trigger a discrete change of the active set of neurons). The function computed by each neuron or by the network output in terms of the network input is thus linear by parts. We can see the model as an *exponential number of linear models that share parameters* Nair and Hinton (2010). Because of this linearity, gradients flow well on the active paths of neurons (there is no gradient vanishing effect due to activation non-linearities of deep networks of sigmoid or tanh units), and mathematical investigation is easier. Computations are also cheaper: there is no need for computing the exponential function in activations, and sparsity can be exploited.

Potential Problems One may hypothesize that the hard saturation at 0 may hurt optimization by blocking gradient back-propagation. To evaluate the potential impact of this effect we also investigate the softplus activation: $softplus(x) = log(1 + e^x)$ (Dugas *et al.*, 2001), a smooth version of the rectifying non-linearity. We lose the exact sparsity, but may hope to gain easier training. However, experimental results (see Section 4.1) tend to contradict that hypothesis, suggesting that hard zeros can actually help supervised training. We hypothesize that the hard non-linearities do not hurt *so long as the gradient can propagate along some paths*, i.e., that some of the hidden units in each layer are non-zero. With the *credit and blame assigned to these ON units* rather than distributed

[1]The hyperbolic tangent absolute value non-linearity $|\tanh(x)|$ used by Jarrett *et al.* (2009) enforces sign symmetry. A $\tanh(x)$ non-linearity enforces sign antisymmetry.

more evenly, we hypothesize that optimization is easier. Another problem could arise due to the unbounded behavior of the activations; one may thus want to use a regularizer to prevent potential numerical problems. Therefore, we use the L_1 penalty on the activation values, which also promotes additional sparsity. Also recall that, in order to efficiently represent symmetric/antisymmetric behavior in the data, a rectifier network would need twice as many hidden units as a network of symmetric/antisymmetric activation functions.

Finally, rectifier networks are subject to ill-conditioning of the parametrization. Biases and weights can be scaled in different (and consistent) ways while preserving the same overall network function. More precisely, consider for each layer of depth i of the network a scalar α_i, and scaling the parameters as $\mathbf{W}'_i = \frac{\mathbf{W}_i}{\alpha_i}$ and $\mathbf{b}'_i = \frac{\mathbf{b}_i}{\prod_{j=1}^{i} \alpha_j}$. The output units values then change as follow: $\mathbf{s}' = \frac{\mathbf{s}}{\prod_{j=1}^{n} \alpha_j}$. Therefore, as long as $\prod_{j=1}^{n} \alpha_j$ is 1, the network function is identical.

3.2 Unsupervised Pre-training

This paper is particularly inspired by the sparse representations learned in the context of auto-encoder variants, as they have been found to be very useful in training deep architectures (Bengio, 2009), especially for unsupervised pre-training of neural networks (Erhan *et al.*, 2010).

Nonetheless, certain difficulties arise when one wants to introduce rectifier activations into stacked denoising auto-encoders (Vincent *et al.*, 2008). First, the hard saturation below the threshold of the rectifier function is not suited for the reconstruction units. Indeed, whenever the network happens to reconstruct a zero in place of a non-zero target, the reconstruction unit can not backpropagate any gradient.[2] Second, the unbounded behavior of the rectifier activation also needs to be taken into account. We have experimented with several strategies, and propose the following two (the first strategy has proven to be the most efficient on image experiments and the second one on text ones):

1. Use a softplus activation function for the reconstruction layer, along with a quadratic reconstruction cost.
2. Scale the rectifier activation values coming from the previous encoding layer to bound them between 0 and 1, then use a sigmoid activation function for the reconstruction layer, along with a cross-entropy reconstruction cost.

4 Experimental Study

4.1 Image Recognition

Experimental setup We considered the image datasets detailed below. Each of them has a training set (for tuning parameters), a validation set (for tuning hyper-parameters) and a test set (for reporting generalization performance). They are presented according to their number of training/validation/test examples, their respective image sizes, as well as their number of classes:

- MNIST (LeCun *et al.*, 1998): 50k/10k/10k, 28×28 digit images, 10 classes.
- CIFAR10 (Krizhevsky and Hinton, 2009): 50k/5k/5k, $32 \times 32 \times 3$ RGB images, 10 classes.
- NISTP: 81920k/80k/20k, 32×32 character images from the NIST database 19, with randomized distortions (Anonymous, 2010), 62 classes. This dataset is much larger and more difficult than the original NIST (Grother, 1995).
- NORB: 233172/58428/58320, taken from Jittered-Cluttered NORB (LeCun *et al.*, 2004). Stereo-pair images of toys on a cluttered background, 6 classes. The data has been preprocessed similarly to (Nair and Hinton, 2010): we subsampled the original $2 \times 108 \times 108$ stereo-pair images to $2 \times 32 \times 32$ and scaled linearly the image in the range $[-1,1]$. We followed the procedure used by Nair and Hinton (2010) to create the validation set.

For all experiments except on the NORB data (LeCun *et al.*, 2004), the models we used are stacked denoising auto-encoders (Vincent *et al.*, 2008) with three hidden layers and 1000 units per layer.

[2] Why is this not a problem for hidden layers too? we hypothesize that it is because gradients can still flow through the active (non-zero), possibly helping rather than hurting the assignment of credit.

Table 1: **Test error on networks of depth 3.** Bold results represent statistical equivalence between similar experiments, with and without pre-training, under the null hypothesis of the pairwise test with $p = 0.05$.

Neuron	MNIST	CIFAR10	NISTP	NORB
***With* unsupervised pre-training**				
Rectifier	**1.20%**	**49.96%**	**32.86%**	**16.46%**
Tanh	**1.16%**	**50.79%**	35.89%	17.66%
Softplus	**1.17%**	**49.52%**	**33.27%**	19.19%
***Without* unsupervised pre-training**				
Rectifier	**1.43%**	**50.86%**	**32.64%**	**16.40%**
Tanh	1.57%	52.62%	36.46%	19.29%
Softplus	1.77%	53.20%	35.48%	17.68%

The architecture of Nair and Hinton (2010) has been used on NORB: two hidden layers with respectively 4000 and 2000 units. We used a cross-entropy reconstruction cost for tanh networks and a quadratic cost over a softplus reconstruction layer for the rectifier and softplus networks. We chose masking noise as the corruption process: each pixel has a probability of 0.25 of being artificially set to 0. The unsupervised learning rate is constant, and the following values have been explored: $\{.1, .01, .001, .0001\}$. We select the model with the lowest reconstruction error. For the supervised fine-tuning we chose a constant learning rate in the same range as the unsupervised learning rate with respect to the supervised validation error. The training cost is the negative log likelihood $-\log P(\text{correct class}|\text{input})$ where the probabilities are obtained from the output layer (which implements a softmax logistic regression). We used stochastic gradient descent with mini-batches of size 10 for both unsupervised and supervised training phases.

To take into account the potential problem of rectifier units not being symmetric around 0, we use a variant of the activation function for which half of the units output values are multiplied by -1. This serves to cancel out the mean activation value for each layer and can be interpreted either as inhibitory neurons or simply as a way to equalize activations numerically. Additionally, an $L1$ penalty on the activations with a coefficient of 0.001 was used during pre-training and fine-tuning in order to increase the amount of sparsity in the learned representations.

Main results Table 1 summarizes the results on networks of 3 hidden layers of 1000 hidden units each, comparing all the neuron types on all the datasets, with or without unsupervised pre-training. In the latter case, the supervised training phase has been carried out using the same experimental setup as the one described above for fine-tuning. The main observations we make are the following:

- Despite the hard threshold at 0, networks trained with the rectifier activation function can find local minima of greater or equal quality than those obtained with its smooth counterpart, the softplus. Rectifiers are not only biologically plausible, they are also computationally efficient.
- There is almost no improvement when using unsupervised pre-training with rectifier activations, contrary to what is experienced using tanh or softplus. Purely supervised rectifier networks remain competitive on all 4 datasets, even against the pretrained tanh or softplus models.
- Rectifier networks are truly deep sparse networks. There is an average exact sparsity (fraction of zeros) of the hidden layers of 83.40% on MNIST and 72.00% on CIFAR10. Figure 3 (left) provides a better understanding of the influence of sparsity. It displays the MNIST test error of deep rectifier networks (without pre-training) according to different average sparsity obtained by varying the $L1$ penalty on the activations. Networks appear to be quite robust to it as models with 70% to almost 85% of true zeros can achieve similar performances.

With labeled data, deep rectifier networks appear to be attractive models. They are biologically credible, and, compared to their standard counterparts, do not seem to depend as much on unsupervised pre-training, while ultimately yielding sparse representations.

This last conclusion is slightly different from those reported in (Nair and Hinton, 2010) in which is demonstrated that unsupervised pre-training with Restricted Boltzmann Machines and using rectifier units is beneficial. In particular, the paper reports that pre-trained rectified Deep Belief Networks can achieve a test error on NORB below 16%. However, we believe that our results are compatible with those: we extend the experimental framework to a different kind of models (stacked denoising auto-encoders) and different datasets (on which conclusions seem to be different). Furthermore,

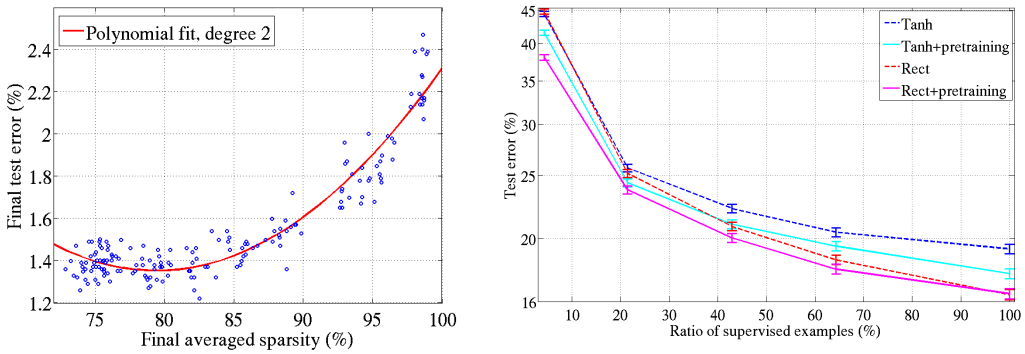

Figure 3: Left: influence of final sparsity on accuracy. 200 randomly initialized deep rectifier networks were trained on MNIST with various L_1 penalties (from 0 to 0.001) to obtain different sparsity levels. Right: effect of unsupervised pre-training. On NORB, we compare hyperbolic tangent and rectifier networks, with or without unsupervised pre-training, and fine-tune only on subsets of increasing size of the training set.

note that our rectified model without pre-training on NORB is very competitive (16.4% error) and outperforms the 17.6% error of the non-pretrained model from Nair and Hinton (2010), which is basically what we find with the non-pretrained softplus units (17.68% error).

Semi-supervised setting Figure 3 (right) presents results of semi-supervised experiments conducted on the NORB dataset. We vary the percentage of the original labeled training set which is used for the supervised training phase of the rectifier and hyperbolic tangent networks and evaluate the effect of the unsupervised pre-training (using the whole training set, unlabeled). Confirming conclusions of Erhan *et al.* (2010), the network with hyperbolic tangent activations improves with unsupervised pre-training for any labeled set size (even when all the training set is labeled). However, the picture changes with rectifying activations. In semi-supervised setups (with few labeled data), the pre-training is highly beneficial. But the more the labeled set grows, the closer the models with and without pre-training. Eventually, when all available data is labeled, the two models achieve identical performance. Rectifier networks can maximally exploit labeled and unlabeled information.

4.2 Sentiment Analysis

In text-related applications, data is usually very sparse. Deep rectifier networks, thanks to their naturally sparse behavior, could be an interesting match for this kind of learning task. To validate this intuition, we consider sentiment analysis, a text mining area which aims to determine the judgment of a writer with respect to a given topic (see (Pang and Lee, 2008) for a review). The basic task consists in classifying the polarity of reviews either by predicting whether the expressed opinions are positive or negative, or by assigning them star ratings on either 3, 4 or 5 star scales.

Following a task originally proposed by Snyder and Barzilay (2007), our data consists of restaurant reviews which have been extracted from the restaurant review site `www.opentable.com`. We have access to 10,000 labeled and 300,000 unlabeled training reviews, while the test set contains 10,000 examples. The goal is to predict the rating on a 5 star scale and performance is evaluated using Root Mean Squared Error (RMSE).[3] The review text is treated as a bag of words and transformed into binary vectors encoding the presence/absence of terms. For computational reasons, only the 5000 most frequent terms of the vocabulary are kept in the feature set.[4] The resulting preprocessed data is very sparse: 0.6% of non-zero features on average. Unsupervised pre-training of the networks employs both labeled and unlabeled training reviews while the supervised fine-tuning phase is carried out by 10-fold cross-validation on the labeled training examples.

The model are stacked denoising auto-encoders, with 1 or 3 hidden layers of 5000 hidden units and rectifier or tanh activation, which are trained in a greedy layer-wise fashion. Predicted ratings are defined by the expected star value computed using multiclass (multinomial, softmax) logistic

[3]Even though our tasks are identical, our database is larger than the one of (Snyder and Barzilay, 2007).

[4]Preliminary experiments suggested that larger vocabulary sizes did not markedly change results.

regression output probabilities. For rectifier networks, when a new layer is stacked, activation values of the previous layer are scaled within the interval [0,1] and a sigmoid reconstruction layer with a cross-entropy cost is used. We also add an $L1$ penalty during pre-training and fine-tuning. Because of the binary input, we use a "salt and pepper noise" (i.e. masking some inputs by zeros and others by ones) for unsupervised training of the first layer. A simple zero masking (as in (Vincent *et al.*, 2008)) is used for the higher layers. We selected the noise level of each layer based on the classification performance, the other hyperparameters are selected according to the reconstruction error.

Table 2: **Test RMSE and sparsity level** obtained by 10-fold cross-validation on OpenTable data.

Network	**No hidden layer**	**Rectifier (1-layer)**	**Rectifier (3-layers)**	**Tanh (3-layers)**
RMSE	0.885 ± 0.006	0.807 ± 0.004	$\mathbf{0.746} \pm 0.004$	0.774 ± 0.008
Sparsity	99.4%	$28.9\% \pm 0.2$	$53.9\% \pm 0.7$	$00.0\% \pm 0.0$

Results are displayed in Table 2. Interestingly, the RMSE significantly decreases as we add hidden layers to the rectifier neural net. These experiments confirm that rectifier networks improve after an unsupervised pre-training phase in a semi-supervised setting: with no pre-training, the 3-layers model can not obtain a RMSE lower than 0.833. Additionally, although we can not replicate the original very high degree of sparsity of the training data, the 3-layers network can still attain an overall sparsity of more than 50%. Finally, on data with these particular properties (binary, high sparsity), the 3-layers network with tanh activation function (which has been learnt with the exact same setup) is clearly outperformed. The sparse behavior of the deep rectifier network seems particularly suitable in this case, because the raw input is very sparse and *varies in its number of non-zeros*. The latter can also be achieved with sparse internal representations, not with dense ones.

5 Conclusion

Sparsity and neurons operating mostly in a linear regime can be brought together in more biologically plausible deep neural networks. Rectifier units help to bridge the gap between unsupervised pre-training and no pre-training, which suggests that they may help in finding better minima during training. This finding has been verified for four image classification datasets of different scales and all this in spite of their inherent problems, such as zeros in the gradient, or ill-conditioning of the parametrization. Rather sparse networks are obtained (from 50 to 80% sparsity for the best generalizing models, whereas the brain is hypothesized to have 95% to 99% sparsity), which may explain some of the benefit of using rectifiers. Furthermore, rectifier activation functions have shown to be remarkably adapted to sentiment analysis, a text-based task with a very large degree of data sparsity. This promising result tends to indicate that deep sparse rectifier networks might yield powerful text mining tools in the future.

Acknowledgments

The authors are greatful for the support received from the following organizations: DARPA, NSERC, CBIE, CIFAR, the Canada Research Chairs, and MITACS.

References

Anonymous, A. (2010). Anonymous title. Technical report, Anonymous University.

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*, volume 9, pages 249–256.

Bengio, Y., Lamblin, P., Popovici, D., and Larochelle, H. (2007). Greedy layer-wise training of deep networks. In *NIPS 19*, pages 153–160. MIT Press.

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

Candes, E. and Tao, T. (2005). Decoding by linear programming. *IEEE Transactions on Information Theory*, **51**(12), 4203–4215.

Dayan, P. and Abott, L. (2001). *Theoretical neuroscience*. MIT press.

Doi, E., Balcan, D. C., and Lewicki, M. S. (2006). A theoretical analysis of robust coding over noisy overcomplete channels. In *NIPS'05*, pages 307–314. MIT Press, Cambridge, MA.

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

Dugas, C., Bengio, Y., Belisle, F., Nadeau, C., and Garcia, R. (2001). Incorporating second-order functional knowledge for better option pricing. In *NIPS 13*. MIT Press.

Erhan, D., Bengio, Y., Courville, A., Manzagol, P.-A., Vincent, P., and Bengio, S. (2010). Why does unsupervised pre-training help deep learning? *JMLR*, **11**, 625–660.

Goodfellow, I., Le, Q., Saxe, A., and Ng, A. (2009). Measuring invariances in deep networks. In *NIPS'09*, pages 646–654.

Grother, P. (1995). Handprinted forms and character database, NIST special database 19. In *National Institute of Standards and Technology (NIST) Intelligent Systems Division (NISTIR)*.

Hahnloser, R. L. T. (1998). On the piecewise analysis of networks of linear threshold neurons. *Neural Netw.*, **11**(4), 691–697.

Hinton, G. E., Osindero, S., and Teh, Y. (2006). A fast learning algorithm for deep belief nets. *Neural Computation*, **18**, 1527–1554.

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.

Krizhevsky, A. and Hinton, G. (2009). Learning multiple layers of features from tiny images. Technical report, University of Toronto.

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

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, **86**(11), 2278–2324.

LeCun, Y., Huang, F.-J., and Bottou, L. (2004). Learning methods for generic object recognition with invariance to pose and lighting. In *Proc. CVPR'04*, volume 2, pages 97–104, Los Alamitos, CA, USA. IEEE Computer Society.

Lee, H., Battle, A., Raina, R., and Ng, A. (2007). Efficient sparse coding algorithms. In *NIPS'06*, pages 801–808. MIT Press.

Lee, H., Ekanadham, C., and Ng, A. (2008). Sparse deep belief net model for visual area V2. In *NIPS'07*, pages 873–880. MIT Press, Cambridge, MA.

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 *NIPS'08*, pages 1033–1040. NIPS Foundation.

Nair, V. and Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. In *Proc. 27th International Conference on Machine Learning*.

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.

Pang, B. and Lee, L. (2008). Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, **2**(1-2), 1–135. Also published as a book. Now Publishers, 2008.

Ranzato, M., Poultney, C., Chopra, S., and LeCun, Y. (2007). Efficient learning of sparse representations with an energy-based model. In *NIPS'06*.

Ranzato, M., Boureau, Y.-L., and LeCun, Y. (2008). Sparse feature learning for deep belief networks. In *NIPS'07*, pages 1185–1192, Cambridge, MA. MIT Press.

Salinas, E. and Abbott, L. F. (1996). A model of multiplicative neural responses in parietal cortex. *Neurobiology*, **93**, 11956–11961.

Serre, T., Kreiman, G., Kouh, M., Cadieu, C., Knoblich, U., and Poggio, T. (2007). A quantitative theory of immediate visual recognition. *Progress in Brain Research, Computational Neuroscience: Theoretical Insights into Brain Function*, **165**, 33–56.

Snyder, B. and Barzilay, R. (2007). Multiple aspect ranking using the Good Grief algorithm. In *Proceedings of HLT-NAACL*, pages 300–307.

Vincent, P., Larochelle, H., Bengio, Y., and Manzagol, P.-A. (2008). Extracting and composing robust features with denoising autoencoders. In *ICML'08*, pages 1096–1103. ACM.