Comparing Rectified Linearity Performance on Image Data

Jason Smith

Lecturer, Boise State University Mathematics Department

Email: jasonsmith5@boisestate.edu

Abstract: Two recent variants of the popular ReLU activation function, leaky ReLU and parametric ReLU, have been adopted to solve the dying ReLU problem. Using metrics that quantify the quality of information representation in deep neural networks, we compare the performance of ReLU, leaky ReLU, and parametric ReLU on two benchmark image recognition datasets using simple convolutional neural networks.

Keywords: Deep Neural Network, Image Recognition, Activation Function

1. Introduction

When encoding information, the number of active neurons in the human brain has been estimated to be at most 4% (Lennie et al.). Similarly, *sparse* representations in Deep Neural Networks (DNNs) increase performance and invariance to input perturbation (Bengio et al.). However, sparseness is not the only measure of quality information representation. Neurons in the human brain represent information in both a sparse and distributed (*disperse*) way (Attwell et al.).

Once the hallmark of DNN architectures, sigmoidal hidden units have recently given way to the rectified linear unit (ReLU) as the unit of choice for modern DNNs (Figure 1). This is due, in part, to ReLU's ability to provide both sparse and disperse hidden layer representations (Maas et al.).

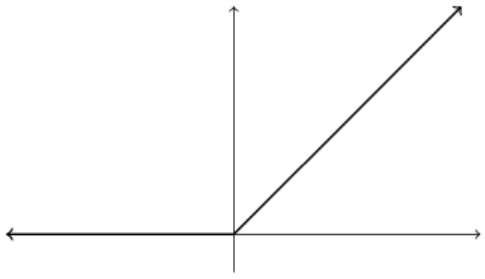

Figure 1: ReLU Activation Function

Another factor is the greater susceptibility of sigmoidal hidden units to the *vanishing gradient* problem (Bengio et al., 1994) than ReLU hidden units. Vanishing, or minimal, gradients occur when the partial derivative of enough hidden unit activations in a lower layer are close to 0. For sigmoidal activation functions, $\sigma(z)$, this happens when $|z| \geq 5$ whereas for rectified linear activations, $g(z)$, $g'(z) = 0$ only when $z \leq 0$.

Still, a variant of the vanishing gradient problem, the *dying ReLU* problem (Karniadakis et al.), can affect ReLU hidden units when the *learning rate* or *bias term* is too large and z becomes negative during backpropogation or training, respectively. Three broad DNN modifications have been taken to address this problem: architecture, training, and initialisation.

In this paper, we address a small piece of the first of these modifications, namely the performance of ReLU variants on image recognition problems. These variants are *leaky ReLU* (LReLU) (Figure 2) and *parametric ReLU* (PReLU). Section 2 discusses the contributions of two other similar works. Section 3 defines the two variants of ReLU that we will study. Section 4 delves into the experiments and presents an empirical comparison of the performance of the three rectifier hidden units mentioned above. Lastly, Section 5 analyzes the experiments of Section 4.

2. Related Works

Focusing more on the effects of activation function choice (hyperbolic tangent (tanh), ReLU, and LReLU) then model performance, Mass et al. train both ReLU, LReLU, and Tanh acoustic DNNs to model a large vocabulary continuous speech recognition (LVCSR) task and show that using rectified linear units decreases word

error rates by 2% and increases model accuracy 4% as compared to their sigmoidal counterparts. Further, Maas et al. show that the sparsity in the last hidden layer in their ReLU acoustic model is six times better than the sparsity in the last hidden layer of their tanh acoustic model. Both the ReLU and LReLU acoustic models perform the same.

Addressing the lack of impact of LReLU over ReLU, He et al. proposed the PReLU activation function which improves upon LReLU by treating the constant fixed negative value in LReLU as a learnable parameter. After tuning a 14-layer convolutional neural network (CNN) to model the ImageNet dataset using all ReLU hidden units, He et al. retrained the same model with all ReLUs replaced by PReLUs. Their PReLU model obtained a 1.2% increase in accuracy over the ReLU baseline.

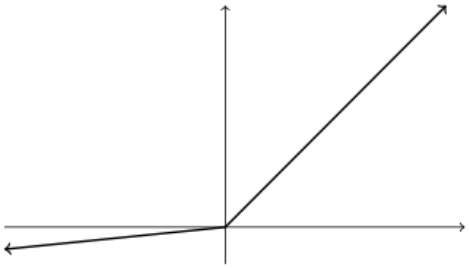

Figure 2: LReLU Activation Function

These results lead one to wonder if PReLU will always outperform ReLU and LReLU? In this work, we train three CNNs on two benchmark image recognition data sets (Fashion MNIST and CIFAR10) using hanh and each of the three rectified linear units described above. To compare results we use testing accuracy, *lifetime sparsity*, and *dispersity* in an attempt to quantify the performance of each activation function.

3 Rectified Linear Units

Given a hidden layer prior to activation, $z = W^T x + b$, the *generalized* ReLU activation function is given by

$$g(z_i) = \max(0, z_i) + \alpha_i \cdot \min(0, z_i).$$

When $\alpha_i = 0$ we have ReLU (Figure 1). When $\alpha_i = 0.1$, (or another small number) we have the LReLU (Figure 2). When α_i is treated as a learnable parameter, we have the Parametric ReLU.

PReLU has two variants: channel-wise and channel-shared. In the channel-wise variant z_i is the input of g on the ith channel. In the channel-shared variant the coefficient $\alpha_i = \alpha$ is shared by all channels of one layer.

PReLU is trained using backpropagation and optimized simultaneously with the layers of a DNN. The gradient of α_i for one layer is

$$\frac{\partial L}{\partial \alpha_i} = \sum_{z_i} \frac{\partial L}{\partial g(z_i)} \cdot \frac{\partial g(z_i)}{\partial \alpha_i},$$

where L is the loss function. The $\frac{\partial L}{\partial g(z_i)}$ term is the gradient from a deeper layer whose gradient is

$$\frac{\partial L}{\partial g(z_i)} = \begin{cases} 0 & \text{if } z_i > 0 \\ z_i & \text{if } z_i \leq 0 \end{cases}.$$

In the channel-shared variant,

$$\frac{\partial L}{\partial \alpha} = \sum_i \sum_{z_i} \frac{\partial L}{\partial g(z_i)} \cdot \frac{\partial g(z_i)}{\partial \alpha},$$

where the outer sum is over all channels of the layer. The momentum method is used for updating α_i. He et al. suggest initializing $\alpha_i = 0.25$. In this paper, we study the channel-wise variant only.

4 Experiments

4.1 Data

We use two standard image recognition data sets: Fashion-MNIST and CIFAR-10.

The Fashion-MNIST consists of $60,000$ training samples and $10,000$ testing samples. Each Fashion-MNIST sample is a 28×28 grayscale image from one of the following 10 classes: T-shirt, pants, pullover, dress, coat, sandal, shirt, sneaker, bag, and ankle boot.

The CIFAR-10 data set also consists of $60,000$ training samples and $10,000$ testing samples. Each CIFAR-10 sample is a 32×32 color image from one of the following 10 classes: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck.

4.2 Metrics

To assess model performance we use three metrics applied to the testing data: accuracy, lifetime sparsity, and dispersion. Accuracy is the ratio of correctly classified samples. To define sparsity and dispersion we first need to define the *activation probability* of a hidden unit.

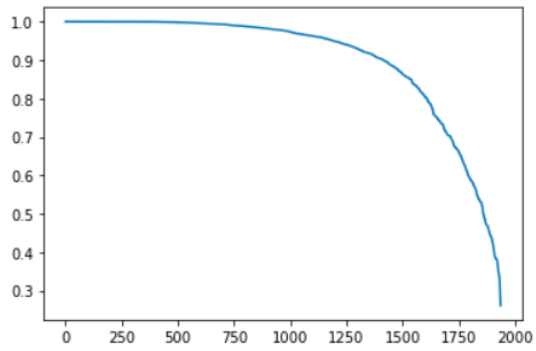

Figure 3: Activation probabilities for tanh.

Given a set of testing samples, the *activation probability* of a hidden unit is the ratio of samples for which the unit is not saturated. We define rectified linear units as saturated when $g(z) \leq 0$ and hyperbolic tangent units as saturated when $\tanh(z) \leq -0.95$ or $\tanh(z) \geq 0.95$. Figure 3 shows the activation probabilities, sorted in decreasing order, for the second convolutional layer in a 3-layer DNN that models the Fashion-MNIST data set using the tanh activation function.

The *lifetime sparsity* of a hidden layer in a DNN is the average of the activation probabilities of all units in that layer (Willmore and Tollhurst, 2001). The *dispersion*, which measures whether the set of active units is different for each sample (Willmore et al. 2000), of a hidden layer is defined to be the standard deviation of all activation probabilities for all units in that layer. If this standard deviation is 0, then the layer is perfectly disperse, meaning that all units activate equally.

As an example, consider two hidden layers consisting of five units with activation probabilities: $[0.5, 0, 0.5, 0, 0]$ and $[0.2, 0.2, 0.2, 0.2, 0.2]$. The former layer has a lifetime sparsity of 0.2 but a dispersity of 0.24 while the latter hidden layer has lifetime sparsity 0.2 and is perfectly disperse (std= 0).

Informally, dispersion is the slope of the curve defined by the activation probabilities for a hidden layer, when sorted in decreasing order. In Figure 3 we can see that tanh has low dispersion but high lifetime sparsity.

4.3 Methods

In an effort to isolate hidden activation performance, we do not focus on obtaining high classification accuracy. Rather, we focus on differences in classification accuracy, lifetime sparsity, and dispersion between the models on each data set.

To this end, we trained twelve different CNNs, each of which uses one of the ReLU, LReLU, PReLU, or tanh activation functions for all convolution layers. For each of these four choices of activation function,we trained a 4-layer CNN on the Fashion-MNIST data set, a 4-layer CNN on the CIFAR-10 data set, and a slightly deeper CNN using three VGG blocks (Simonyan et al.) on the CIFAR-10 data set. All twelve of these models use the same two activation functions, ReLU and softmax, in their final two fully-connected layers.

Table 1: Model Specifications

3-layer CNN	3-VGG block CNN
conv-32, 3×3	conv-32, 3×3
max pool-2×2	conv-32, 3×3
conv-32, 3×3	max pool-2×2
max pool-2×2	conv-64, 3×3
flatten	conv-64, 3×3
dense-128	max pool-2×2
dense-10	conv-128, 3×3
	conv-128, 3×3
	max pool-2×2
	flatten
	dense-128
	dense-10

The latter deeper model was used in an effort to give PReLU a chance to better learn its parameter α_i. To determine the number of epochs necessary to achieve the best performance and eliminate overfitting in this deeper model, (3-VGG Blocks) we first trained with the PReLU activation function for 10 epochs. Since it was clear that overfitting had begun in the third epoch, we then trained all models using three epochs.

To obtain the lifetime sparsity and dispersion for each of these models we computed the activation probability of all hidden units in the last convolutional layer of each model. This layer was chosen over all other layers since, at this point, the richest representation by the convolutional layers has been obtained.

We implemented the aforementioned models using Tensorflow and Keras. All of these models use the Adam optimizer and the categorical cross entropy loss function. Since these models

Table 2: 4-Layer Fashion-MNIST Results

Function	Accuracy	Lifetime Sparsity	Dispersion
tanh	0.897	0.931	0.145
ReLU	0.906	0.265	0.161
LReLU	0.900	0.270	0.153
PReLU	0.908	0.219	0.194

Table 3: 4-Layer CIFAR-10 Results

Function	Accuracy	Lifetime Sparsity	Dispersion
tanh	0.684	0.960	0.078
ReLU	0.685	0.203	0.136
LReLU	0.685	0.178	0.116
PReLU	0.703	0.167	0.110

are not very deep, the default Glorot Uniform initialisation is sufficient.

4.4 Results

Tables 2, 3, and 4 show the accuracy, lifetime sparsity, and dispersion for each of the four types of hidden units described above on the Fashion-MNIST 3-layer CNN, the CIFAR-10 3-layer CNN, and the CIFAR-10 3-VGG Block CNN, respectively.

Figures 2, 3, and 4 show the activation probabilities for each of the four types of hidden units described above that were collected in the final convolution layer of the Fashion-MNIST 3-layer CNN, the CIFAR-10 3-layer CNN, and the CIFAR-10 3-VGG Block CNN, respectively. These figures display the activation probabilities sorted in decreasing order. Note that the slope of these curves gives a rough estimate of the dispersion.

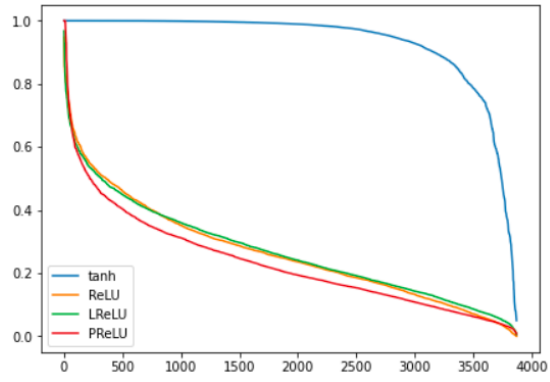

Figure 4: 4-Layer Fashion-MNIST Activation Probabilities

Table 4: 3-VGG Block CIFAR-10 Results

Function	Accuracy	Lifetime Sparsity	Dispersion
tanh	0.699	0.217	0.089
ReLU	0.747	0.093	0.096
LReLU	0.741	0.118	0.074
PReLU	0.763	0.143	0.083

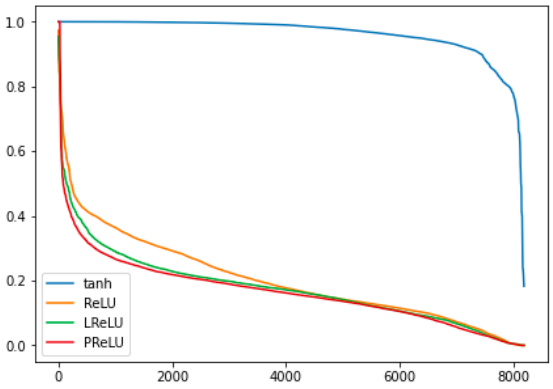

Figure 5: 4-Layer CIFAR-10 Activation Probabilities

5. Analysis

The results above provide do not provide evidence that PReLU will always outperform ReLU and LReLU on image recognition problems. However, these results do indicate that PReLU does not perform worse, and often performs better than ReLU and LReLU when classifying images. Moreover, these results indicate that, although sparsity and dispersion are good metrics for understanding the quality of information representation in a DNN, these metrics do not give one a complete picture of how DNNs comprehend images.

PReLU gave its best performance with the 4-layer CNN modelling the CIFAR-10 dataset. This model gave the best performance of all four models with this architecture and this dataset on all three metrics.

PReLU performed about the same as ReLU and LReLU with the 4-layer CNN modelling the Fashion-MNIST dataset. While PReLU performed the best on lifetime sparsity, it had the same accuracy and slightly worse dispersion as ReLU and LReLU.

While PReLU obtained the best accuracy of the four models using 3-block VGG architecture, it scored about 2% and 1% worse than LReLU on lifetime sparsity and dispersion, respectively. Also, PReLU scored about 5% worse but 1% better than ReLU on lifetime sparsity and dis-

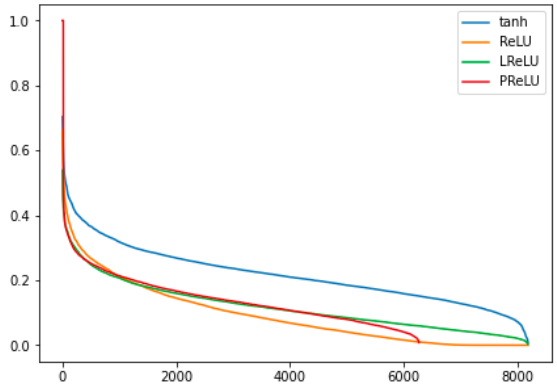

Figure 6: 3-VGG Block CIFAR-10 Activation Probabilities

persion, respectively. Even though these inconsistencies are minimal, they make one question the completeness of the picture captured by lifetime sparsity and dispersion.

References

Attwell, D. , Laughlin, S. An Energy Budget for Signaling in the Grey Matter of the Brain. *The Journal of Cerebral Blood Flow and Metabolism*, 21(10, 1133-1145, 2001.

Bengio, Y.,Bordes, A., Glorot, X. Deep Sparse Rectifier Neural Networks. In *AISTATS'2011.*

Bengio, Y., Simard, P., Fransconi, P., Learning longterm dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2), 1994.

He, K., Zhang, X., Ren, S., Sun, J.. Delving Deep Into Rectifiers: Surpassing Human Level Performance on ImageNet Classification. arXiv 1502.01852, 2015.

Karniadakis, G., Lu, L., Yanhui, S., Yeonjong, S., Dying ReLU and Initialization: Theory and Numerical Examples.arXiv: 1903.06733, 2019.

Lennie, P. The cost of cortical computation. *Current Biology*, 14, 493-497.

Maas, A., Hannun, A., Ng, A. Rectifier Nonlinearities Improve Neural Network Acoustic Models. In *ICML Workshop on Deep Learning for Audio, Speech, and Language Processing*, 2013.

Simonyan, K., Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv: 1409.1556, 2014.

Willmore, B., Tolhurst, D. Characterizing the Sparseness of Neural Codes. *Network: Computation in Neural Systems*, 12(3): 255-270, 2001.

Willmore, B., Watters, P., Tolhurst, D. Acomparison of Natural-Image-Based Models of Simple-Cell Coding. *Perception*, 29(9): 1017-1049, 2000.