Activation Function

Junkun Yuan
Zhejiang University
yuanjk@zju.edu.cn

1. Introduction

In artificial neural networks, each neuron forms a weighted sum of its inputs and passes the resulting scalar value through a function referred to as an **activation function** or transfer function.[1]

We summarize the activation functions with key phrases:

- **Sigmoid:** normalize to $(0, 1)$, sensitive to extremely large/small input. Sec. 2
- **ReLU:** sparsity, alleviate gradient vanishing, died weights, gradient explosion, AlexNet, VGG, ResNet, MobileNet, DenseNet, SENet, ShuffleNet. Sec. 3
- **LReLU:** avoid died wights, GANs. Sec. 4
- **ELU:** robust to noise, push the mean of activations closer to zero, GAT. Sec. 5
- **SELU:** normalize outputs, avoid gradient vanishing/explosion. Sec. 6
- **GELU:** relate nonlinearities and stochastic regularizers, random zero-one mask, BERT, GPT, ViT. Sec. 7
- **Swish:** automatic search, EfficientNet, MobileNetV3. Sec. 8

2. Sigmoid

Sigmoid function normalizes input to $(0, 1)$. Its expression Eq. (1) and gradient Eq. (2) are given below.

$$\text{Sigmoid}(x) = \sigma(x) = \frac{1}{1 + e^{-x}} \tag{1}$$

$$\sigma'(x) = \frac{e^{-x}}{(1 + e^{-x})^2} = \sigma(x)(1 - \sigma(x)) \tag{2}$$

However, when x becomes extremely large/small, $\sigma(x)$ would tend to 1/0, then the gradient $\sigma'(x)$ could tend to 0, hence the network weights would not be updated.

[1] From wikipedia Activation Function.

3. ReLU

Rectified Linear Unit (ReLU) [5] is widely used in the representative networks, like AlexNet, VGG, ResNet, MobileNet, DenseNet, SENet, ShuffleNet, etc. Its expression (Eq. (3)) and gradient (Eq. (4)) are shown below.

$$\text{ReLU}(x) = \max(0, x) \tag{3}$$

$$\text{ReLU}'(x) = \begin{cases} 1 & \text{if } x > 0 \\ 0 & \text{if } x \leq 0 \end{cases} \tag{4}$$

It introduces sparsity into the networks, which is benefit for training. Also, it alleviates gradient vanishing since the gradient could be either 1 or 0. However, some weights would not be updated, and gradient explosion is still not be solved.

4. LReLU

The expression (Eq. (5)) and gradient (Eq. (6)) of **Leaky Recified Linear (LReLU)** [4] is given below.

$$\text{LReLU}(x) = \begin{cases} x & \text{if } x > 0 \\ \alpha x & \text{if } x \leq 0 \end{cases} \tag{5}$$

$$\text{LReLU}'(x) = \begin{cases} 1 & \text{if } x > 0 \\ \alpha & \text{if } x \leq 0 \end{cases} \tag{6}$$

It avoids died weights in ReLU, and is widely used in GANs such as DCGAN, SNGAN, ACGAN, and StackGAN, etc.

5. ELU

The expression (Eq. (7)) and gradient (Eq. (8)) of **Exponential Linear Unit (ELU)** [1] are given below.

$$\text{ELU}(x) = \begin{cases} x & \text{if } x > 0 \\ \alpha(\exp(x) - 1) & \text{if } x \leq 0 \end{cases} \tag{7}$$

$$\text{ELU}'(x) = \begin{cases} 1 & \text{if } x > 0 \\ \text{ELU}(x) + \alpha & \text{if } x \leq 0 \end{cases} \tag{8}$$

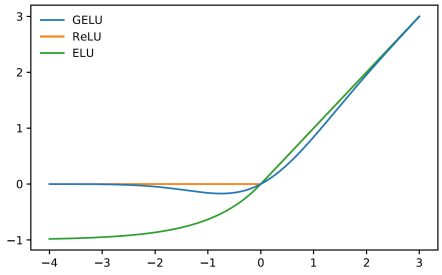

Figure 1. **GELU($\mu = 0, \sigma = 1$).** (From Fig. 1 of [2])

$\alpha > 0$ controls the value to which ELU saturates for negative inputs. It codes the degree of presence of phenomema in the inputs, but does not quantitatively model the degree of their absence. Thus, ELU is more robust to noise. In contrast to ReLU, ELU have negative values which pushes the mean of activations closer to zero, enabling faster learning as they bring the gradient closer to the natural gradient. (From Sec. 1 and Sec. 3 of [5]) It is used in some graph neural networks like graph attention networks (GAT).

6. SELU

Expression (Eq. (9)) and gradient (Eq. (10)) of **Scaled Exponential Linear Unit (SELU)** [3] are given below.

$$\text{SELU}(x) = \lambda \begin{cases} x & \text{if } x > 0 \\ \alpha(\exp(x) - 1) & \text{if } x \leq 0 \end{cases} \tag{9}$$

$$\text{SELU}'(x) = \lambda \begin{cases} 1 & \text{if } x > 0 \\ \alpha \exp(x) & \text{if } x \leq 0 \end{cases} \tag{10}$$

λ and α are set to 1.0507 and 1.67326, respectively. It normalizes outputs and avoids gradient vanishing/explosion.

7. GELU

Nonlinearities (ReLU, ELU, etc.) and stochastic regularizers (Dropout) determine a neuron's output together, yet the two innovations have remained distinct. **Gaussian Error Linear Unit (GELU)** [2] (Fig. 1) relates them by multiplying the input with random zero-one mask (dropout and zoneout). It multiplies the neuron input x by $m \sim \text{Bernoulli}(\Phi(x))$, where $\Phi(x) = P(X \leq x), X \sim \mathcal{N}(0, 1)$. We choose this distribution since neuron inputs tend to follow a normal distribution, especially with Batch Normalization. The transformation is $\Phi(x) \times Ix + (1 - \Phi(x)) \times 0x = x\Phi(x)$. Its formation Eq. (11) and approximation Eq. (12) are given below. (From Sec. 1 and Sec. 2 of [2])

$$\text{GELU}(x) = xP(X \leq x) = x\Phi(x) = x \cdot \frac{1}{2}\left[1 + \text{erf}(\frac{2}{\sqrt{2}})\right] \tag{11}$$

$$0.5x\left(1 + \tanh\left[\sqrt{2/\pi}\left(x + 0.044715x^3\right)\right]\right) \text{ or } x\sigma(1.702x). \tag{12}$$

It is widely used in BERT, GPT, Vision Transformer, etc.

8. Swish

Swish [6] is discovered via automatic search. Its expression (Eq. (13)) and gradient (Eq. (14)) are given below.

$$\text{Swish}(x) = x \cdot \sigma(\beta(x)) \tag{13}$$

$$\text{Swish}'(x) = \beta\text{Swish}(x) + \sigma(\beta(x))(1 - \beta\text{Swish}(x)) \tag{14}$$

σ is a sigmoid function, β is either a constant or a trainable parameter. It is used in EfficientNet and MobileNetV3.

References

[1] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289*, 2015. 1

[2] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016. 2

[3] Günter Klambauer, Thomas Unterthiner, Andreas Mayr, and Sepp Hochreiter. Self-normalizing neural networks. *Advances in neural information processing systems*, 30, 2017. 2

[4] Andrew L Maas, Awni Y Hannun, Andrew Y Ng, et al. Rectifier nonlinearities improve neural network acoustic models. In *Proc. icml*, volume 30, page 3. Citeseer, 2013. 1

[5] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Icml*, 2010. 1, 2

[6] Prajit Ramachandran, Barret Zoph, and Quoc V Le. Searching for activation functions. *arXiv preprint arXiv:1710.05941*, 2017. 2