ELU $\gg$ ReLU

Aidan Rocke

May 31, 2017

Abstract

While the choice of activation function in the hidden layers of a feedforward neural network is essential for controlling the rate and stability of learning, a principled understanding of the relative strengths and weaknesses of different activation functions appears to be lacking.

1 Introduction

1.1 Why are activation functions important?

The choice of activation function in the hidden layers has a profound impact on every aspect of neural network training. In particular, I would emphasise the following:

1. Rate of learning:

   Ideally, learning would be fast and computationally efficient.

2. Stability:

   We want all components to learn at similar rates and we want the derivative of the activation function σ to be Lipschitz-stable to input perturbations $\alpha \in \mathbb{R}$:

$$\exists \lambda \in \mathbb{R}_+ \forall \alpha \in \mathbb{R}, \|\dot{\sigma}(x+\alpha) - \dot{\sigma}(x)\| \leq \lambda \alpha \tag{1}$$

3. Internal covariate shift:

   Small changes to the inputs of a neural networks hidden layers get amplified as we go deeper into the network. This leads to internal covariate set shift within the neural network and slows down the process of learning an approximation to the joint distribution $P(X, Y)$ [4].

1.2 neural networks are complex systems:

1. Hierarchical organisation: Higher-order features in the hidden layers are derived from zero-order features(i.e. input data) in a compositional manner.

2. Highly non-linear: The aggregate behaviour of a deep neural network is highly non-linear and can't be derived from the activity of individual components.

Now, given that activation functions σ_i are fixed during training and these must satisfy the constraints mentioned in the previous section it makes sense that these must be carefully chosen. For a quick review of feedforward neural networks I would refer the reader to the Appendix.

2 Appendix:

2.1 feedforward neural networks:

In general, a feedforward neural network f is a non-linear function consisting of affine transformations(W_i) interleaved with differentiable activation functions(σ_i):

$$\begin{aligned} f &: \widetilde{X} \to Y \\ f(x) &= \sigma_n W_n ... \sigma_2 W_2 \sigma_1 W_1 x \end{aligned} \tag{2}$$

It's useful to note that f represents the composition of differentiable functions so it's also fully differentiable.

2.2 function approximation:

The task of training a feedforward neural network is essentially to obtain successively better approximations f_i to a desired but unknown mapping g:

$$g : X \to Y \tag{3}$$

where $\widetilde{X} \subset X$.

Given a training set $D_{train} = (X_{train}, Y_{train}) = (x_j, y_j)_{j=1}^{n}$, this is done via a sequence of applications of the mini-batch back-propagation algorithm which uses a gradient-based optimiser to simultaneously update all the matrices W_i while leaving the σ_i fixed[1]. This essentially means that the desired output of N passes of mini-batch backprop is to obtain $(f_i)_{i=1}^{N}$ such that:

$$\|f_N - g\| = \min_i \|f_i - g\| \tag{4}$$

where $\|\cdot\|$ is an appropriately chosen metric on a function space G including all parametric approximations of g.

Now, given that g is unknown it follows that the best we can do is to find $f_j \in (f_i)_{i=1}^{N}$ that minimises the empirical risk $\tilde{R}$ on the validation set $D_{validate} = (x_j, y_j)_{j=1}^{m}$:

$$\tilde{R}(f_j) = \frac{1}{m} \sum_{i=1}^{m} L(f_j(x_i), y_i) \tag{5}$$

where L is a suitably chosen loss function.

3 References:

1. J.G Makin. Backpropagation. 2006.
2. V. Vapnik Empirical Risk Minimization for Learning Theory. 1991.
3. X. Glorot & Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. 2010.

4. S. Ioffe & C. Szegedy. Batch Normalisation: Accelerating Deep Network Training by Reducing Internal Covariate Shift. 2015.

5. D. Clevert, T. Unterthiner & S. Hochreiter. Fast and Accurate Deep Network Learning by Exponential Linear Units. 2016.