High Capacity Neural Network

Domenic Donato

2016-10-06

1 Introduction

TODO(domenic): Write this section

2 Related Literature

TODO(domenic): Write this section

2.1 Forward Computation

The forward computation of a fully connected feedforward neural network with N layers can be described as follows:

$$x_{n-1} = a_n(t_n(x_n)) \tag{1}$$

$\{N \in \mathbb{N} | N > 0\}$
$n \in N$
x_n is a vector
x_N is the input vector
$\hat{y} = x_1$ is the output vector
$t_n(x)$ is a transform function
$a_n(x)$ is an activation function

The standard transform function, $t_n(x)$, is:

$$t_n(x) = f_n(x) \tag{2}$$

$$f_n(x) = x \cdot W_n + b_n \tag{3}$$

$W_n \in \mathbb{R}^2$
$b_n \in \mathbb{R}$

The transform function proposed in this paper is:

$$t_n(x) = f_n(x) \cdot g(x) \tag{4}$$

$$g_n(x) = x \cdot Y_n + c_n \tag{5}$$

$Y_n \in \mathbb{R}^2$
$c_n \in \mathbb{R}$

Rather than using the standard transform, (2), a quadratic transform, (4), is performed. The quadratic transform is a superset of the standard design, which becomes apparent when $Y_n \in \{0, ..., 0\}$ and $c_n \in \{1, ..., 1\}$. Using a quadratic transformation enables the neural network to represent any bounded degree polynomial over an infinite domain using a finite number of nodes. The standard design would require an infinite number of nodes to do the same.

2.2 Backward Computation

The total network error given a cost function $c(y, \hat{y})$ is:

$$E = \sum_i c(y_i, \hat{y}_i) \tag{6}$$

A squared error cost function was used for the experiments in this paper.

$$c(y, \hat{y}) = \frac{(y - \hat{y})^2}{2} \tag{7}$$

$$c'(y, \hat{y}) = y - \hat{y} \tag{8}$$

Back propagation was used to train the networks. The key thing about back propagation is that its a dynamic program and caches part of the gradient calculation so it can be reused upstream.

C_n is the matrix of cached calculations at level n

The cache is seeded with the derivative of the cost function:

$$C_1 = c'(y, \hat{y}) \tag{9}$$

At each layer the cache is updated as follows:

$$C_{n+1} = a'_n(x_n) \cdot (C_n \cdot t'_n(x_n)^\top) \tag{10}$$

While visiting a layer, the gradients for each parameter in θ_n is calculated:

$$\delta\theta_n = \theta'_n \cdot C_n \tag{11}$$

3 Experiments

TODO(domenic): Write this section

4 Conclusion

TODO(domenic): Write this section