Convex Neural Network

Jianhao Shen

June 11, 2017

1 Main Idea

The paper: Benjio, Y., Convex Neural Networks, has shown that training a NN can be seen as a convex optimization problem, but has infinite hidden units. By choosing a regularizer that promotes sparse solutions(e.g., L^1 regularizetion), the solution will have a finite number of "active" hidden units, but the problem still consists of a large number of variables. An effective approach to this problem is to add one hidden unit at a time and stop once the global optimum is reached so that not all possible hidden units are visited.

2 convex NN

Neural Network(NN) is a function of the form $\hat{y}(x) = \sum_{i=1}^{m} w_i h_i(x)$ where x is an input vector, $h_i(x)$ are hidden units and w_i are weight values. Hidden units are obtained from a linear discriminant function: $h_i(x) = s(v_i x + b_i)$ where $s(a)$ is an nonlinear function, e.g. $s(a) = tanh(a)$ or $s(a) = \frac{1}{1+e^{-a}}$. A loss function $Q(y, \hat{y})$ is introduced to measure the mismatches between the prediction $\hat{y}(x_i)$ and the observed y. A regularizer $\Omega(w)$ is introduced to favor smaller parameters, e.g. a L^1 regularizer promotes sparse solutions. With notations above, we want to solve the optimization problem:

$$\min_{m,w,v} \Omega(w) + \sum_{i=1}^{n} Q(y_i, \hat{y}_i(x_i)) \tag{1}$$

where $\{(x_i, y_i)\}$ is the observed data set.

This problem is not convex, even if $\Omega(w)$ and $Q(y, \hat{y})$ are convex, since the hidden units $h_i(x) = s(v_i x + b)$ is not convex w.r.t. v_i because of the nonlinear function s(a). Notice that the non-convexity only lies in v_i which determine the chosen hidden units, the problem can be converted into a convex one by including all hidden units in our NN and only tuning w_i to determine both chosen hidden units and weight values(i.e., a hidden unit is chosen iff the corresponding weight value is nonzero). More specifically, we define the "infinite"" NN:

Definition 2.1 *Let $\mathcal{H}$ be a set of functions from an input space $\mathcal{X}$ to $\mathbb{R}$. Elements of $\mathcal{H}$ can be understood as "hidden units" in a NN. Let $\mathcal{W}$ be the Hilbert space of functions from $\mathcal{H}$ to $\mathbb{R}$, with an inner product denoted by $a \cdot b$ for a, b $\in \mathcal{W}$. An element of $\mathcal{W}$ can be understood as the output weights vector in a NN. Let $h(x) : \mathcal{H} \to \mathbb{R}$ the function that maps any element h_i of $\mathcal{H}$ to $h_i(x)$. $h(x)$ can be understood as the vector of activations of hidden units when input x is observed. Let $w \in \mathcal{W}$ represent a parameter(the output weights). The NN prediction is denoted $\hat{y}(x) = w \cdot h(x)$. Let $Q : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ be a cost function convex in its second argument that takes a scalar target value y and a scalar prediction $\hat{y}(x)$ and returns a scalar cost. This is the cost to be minimized on example pair (x, y). Let $D = \{(x_i, y_i : 1 \leq i \leq n)\}$ a training set.Let $\Omega : \mathcal{W} \to \mathbb{R}$ be a convex regularization function that penalizes for the choice of more "complex" parameters. We define the convex NN criterion $C(\mathcal{H}, Q, \Omega, D, w)$ with parameter w as follows:*

$$C(\mathcal{H}, Q, \Omega, D, w) = \Omega(w) + \sum_{t=1}^{n} Q(y_t, w \cdot h(x_t)) \tag{2}$$

It is easy to see that the convex NN cost $C(\mathcal{H}, Q, \Omega, D, w)$ is a convex function of w, but has infinite hidden units, which cannot be trained. However, with regularizer that promotes sparse solutions like L^1 regularizer, it can be proved that the solution has finite hidden units. We show the special case with $Q(y, \hat{y}) = max(0, 1 - y\hat{y})$ and L^1 regularization, and the training criterion is

$$C(w) = K\|w\|_1 + \sum_{t=1}^{n} max(0, 1 - y_t w \cdot h(x_t)) \tag{3}$$

We can rewrite this cost function as:

$$\min_{w,\xi} K\|w\|_1 + \sum_{t=1}^{n} \xi_t \tag{4}$$

such that

$$y_t[w \cdot h(x_t)] \geq 1 - \xi_t \tag{5}$$

$$\xi_t \geq 0, t = 1, ..., n \tag{6}$$

Then we derive the dual problem (P):

$$\max_{\lambda} \sum_{t=1}^{n} \lambda_t \tag{7}$$

such that

$$\lambda \cdot Z_i - K \leq 0, i \in I \tag{8}$$

$$\lambda_t \leq 1, t = 1, ..., n \tag{9}$$

where $(Z_i)_t = y_t h_i(x_t)$,and I is the index set of hidden units. Such a problem follows Theorem 4.2 from (Hettich and Kortanek, 1993), and the following theorem holds:

Theorem 2.1 *The solution of (P) can be attained with constraints $C_2^{'}$ and only $n+1$ constraints $C_1^{'}$(i.e., there exists a subset of n+1 constraints $C_1^{'}$ giving rise to the same maximum as when using the whole set of constraints). Therefore, the primal problem associated is the minimization of the cost function of a NN with n + 1 hidden neurons.*

This convex NN can be optimized by a stepwise algorithm, which is shown as follows:

Algorithm 1 Convex(NN)

Input: training set $D = \{(x_1, y1), ..., (x_n, y_n)\}$, convex loss function Q, and scalar regularization penalty λ.s is either the *sign* function or the *tanh* function.

Set $v_1 = (0, 0, ..., 1)$ and select $w_1 = argmin_{w_1} \sum_t Q(y_t, w_1 s(1)) + \lambda |w_1|$
Set $i = 2$
while True **do**
  Let $q_t = Q^{'}(y_t, \sum_{j=1}^{i-1} w_j h_j(x_t))$
  if s=*sign* **then**
    train linear classifier $h_i(x) = sign(v_i \cdot \tilde{x})$ with examples $\{(x_t, sign(q_t))\}$ and errors weighted by $|q_t|, t = 1, ..., n$.(i.e., $maximize \sum_t q_t h_i(x_t)$)
  else if s=*tanh* **then**
    train linear classifier $h_i(x) = tanh(v_i \cdot \tilde{x})$ to $maximize \sum_t q_t h_i(x_t)$
  end if
  if $\sum_t q_t h_i(x_t) \leq \lambda$ **then**
    stop
  end if
  select $w_1, ..., w_i$(and optimally $v_2, ..., v_i$) minimizing (exactly or approximally) $C = \sum_t Q(y_t, \sum_{j=1}^{i} w_j h_j(x_t)) + \lambda \sum_j |w_j|$ such that $\frac{\partial C}{\partial w_j} = 0$ for $j = 1, ..., i$
  Return the predictor $\hat{y}(x) = \sum_{j=1}^{i} w_j h_j(x)$
end while

And it has been proved that Alogrithm convex NN stops when it reaches the global optimum of $C(w)$. However, finding a linear classifier that minimize the weighted sum of classfication errors is an NP-hard problem, which can only be solved efficiently when the input dimension is low. For high dimension input, we can implement approximate minimization, which turns out to perform well in practice.

In another paper: 2016, Francis Bach, Breaking the Curse of Dimensionality with Convex Neural Networks, the author gives a detailed analysis of generalization bound of different prediction functions, which is listed below:

Model	Function form	Generalization bound
No assumption		$n^{-1/(d+3)}logn$
Affine function	$w^\top x + b$	$d^{1/2} \cdot n^{-1/2}$
Generalized additive model	$\sum_{j=1}^k f_j(w_j^\top x), w_j \in \mathbb{R}^d$	$kd^{1/2} \cdot n^{-1/4}logn$
Single-layer neural network	$\sum_{j=1}^k \mu_j(w_j^\top x + b_j)_+$	$kd^{1/2} \cdot n^{-1/2}$
Projection pursuit	$\sum_{j=1}^k f_j(w_j^\top x), w_j \in \mathbb{R}^d$	$kd^{1/2} \cdot n^{-1/4}logn$
Dependence on subspace	$f(W^\top x), W \in \mathbb{R}^{d\times s}$	$d^{1/2} \cdot n^{-1/(s+3)}logn$

Table 1: Summary of generalization bounds for various models.