Optimization Problems: Fully-connected Networks

Chih-Jen Lin
National Taiwan University

Last updated: April 23, 2021

Multi-class Classification I

- Our training set includes $(\boldsymbol{y}^i, \boldsymbol{x}^i)$, $i = 1, \ldots, l$.
- $\boldsymbol{x}^i \in R^{n_1}$ is the feature vector.
- $\boldsymbol{y}^i \in R^K$ is the label vector.
- As label is now a vector, we change (label, instance) from

$$(y_i, \boldsymbol{x}_i) \text{ to } (\boldsymbol{y}^i, \boldsymbol{x}^i)$$

- K: # of classes
- If $\boldsymbol{x}^i$ is in class k, then

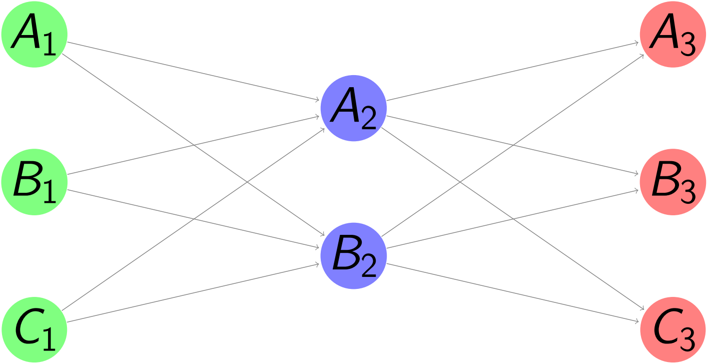
$$\boldsymbol{y}^i = [\underbrace{0, \ldots, 0}_{k-1}, 1, 0, \ldots, 0]^T \in R^K$$

Multi-class Classification II

- A neural network maps each feature vector to one of the class labels by the connection of nodes.

Fully-connected Networks

- Between two layers a weight matrix maps inputs (the previous layer) to outputs (the next layer).



Operations Between Two Layers I

- The weight matrix W^m at the mth layer is

$$W^m = \begin{bmatrix} w_{11}^m & w_{12}^m & \cdots & w_{1n_m}^m \\ w_{21}^m & w_{22}^m & \cdots & w_{2n_m}^m \\ \vdots & \vdots & \vdots & \vdots \\ w_{n_{m+1}1}^m & w_{n_{m+1}2}^m & \cdots & w_{n_{m+1}n_m}^m \end{bmatrix}_{n_{m+1} \times n_m}$$

- n_m : # input features at layer m
- n_{m+1} : # output features at layer m, or # input features at layer $m+1$
- L: number of layers

Operations Between Two Layers II

- $n_1 = \#$ of features, $n_{L+1} = \#$ of classes
- Let $\boldsymbol{z}^m$ be the input of the mth layer, $\boldsymbol{z}^1 = \boldsymbol{x}$ and $\boldsymbol{z}^{L+1}$ be the output
- From mth layer to $(m+1)$th layer

$$\boldsymbol{s}^m = W^m \boldsymbol{z}^m,$$
$$z_j^{m+1} = \sigma(s_j^m),\ j = 1, \ldots, n_{m+1},$$

$\sigma(\cdot)$ is the activation function.

Operations Between Two Layers III

- Usually people do a bias term

$$\begin{bmatrix} b_1^m \\ b_2^m \\ \vdots \\ b_{n_{m+1}}^m \end{bmatrix}_{n_{m+1} \times 1},$$

so that

$$\boldsymbol{s}^m = W^m \boldsymbol{z}^m + \boldsymbol{b}^m$$

Operations Between Two Layers IV

- Activation function is usually an

$$R \to R$$

  non-linear transformation.
- There are various reasons of using an activation function. An important one is to introduce the non-linearity.

Operations Between Two Layers V

- If without an activation function, all

$$W^L \cdots W^2 W^1$$

becomes a single matrix and we end up with having only a linear mapping from the input feature to the output layer

Operations Between Two Layers VI

- We collect all variables:

$$\boldsymbol{\theta} = \begin{bmatrix} \text{vec}(W^1) \\ \boldsymbol{b}^1 \\ \vdots \\ \text{vec}(W^L) \\ \boldsymbol{b}^L \end{bmatrix} \in R^n$$

n : total # variables $= (n_1+1)n_2+\cdots+(n_L+1)n_{L+1}$

- The vec$(\cdot)$ operator stacks columns of a matrix to a vector

Optimization Problem I

- We solve the following optimization problem,

$$\min_{\boldsymbol{\theta}} \quad f(\boldsymbol{\theta}), \quad \text{where}$$

$$f(\boldsymbol{\theta}) = \frac{1}{2}\boldsymbol{\theta}^T\boldsymbol{\theta} + C\sum\nolimits_{i=1}^{l} \xi(\boldsymbol{z}^{L+1,i}(\boldsymbol{\theta}); \boldsymbol{y}^i, \boldsymbol{x}^i).$$

  C: regularization parameter

- $\boldsymbol{z}^{L+1}(\boldsymbol{\theta}) \in R^{n_{L+1}}$: last-layer output vector of $\boldsymbol{x}$.
  $\xi(\boldsymbol{z}^{L+1}; \boldsymbol{y}, \boldsymbol{x})$: loss function. Example:

$$\xi(\boldsymbol{z}^{L+1}; \boldsymbol{y}, \boldsymbol{x}) = ||\boldsymbol{z}^{L+1} - \boldsymbol{y}||^2$$

Optimization Problem II

- The formulation is same as linear classification
- However, the loss function is more complicated
- Further, it's non-convex
- Note that in the earlier discussion we consider a single instance
- In the training process we actually have for $i = 1, \ldots, l$,

$$\boldsymbol{s}^{m,i} = W^m \boldsymbol{z}^{m,i},$$
$$z_j^{m+1,i} = \sigma(s_j^{m,i}),\ j = 1, \ldots, n_{m+1},$$

This makes the training more complicated