Multi Layer Perceptron
MLP

Junior R. Ribeiro
jrodrib@usp.br

17 de setembro de 2020

Conteúdo

1 Multicamadas

Considere as camadas $\ell = 0, ..., L$ em que os neurônios de uma camada somente se comunicam com os neurônios em camadas vizinhas. Quando isso acontece, dizemos que o MLP é sem atalhos.

Considere os padrões de entrada $\bar{x}(n) \in \mathbb{R}^{m_0 - 1}$ para cada $n = 0, ..., N - 1$ (um total de N padrões de entrada), com suas respectivas saídas desejadas $d(n) \in \Omega^{m_L}$ em que $\Omega = [0, 1]$ ou $\Omega = [-1, 1]$, dependendo do problema. As camadas $\ell = 0$ e $\ell = L$ são as camadas de entrada e saída da rede, e todas as camadas $0 < \ell < L$ são camadas ocultas.

As dimensões das variáveis usadas para cômputo do fluxo da rede são $m_0, m_1, ..., m_L$. Por causa dos *biases* as variáveis do fluxo principal entre os neurônios (exceto os *biases*) têm dimensões $m_0 - 1, m_1 - 1, ..., m_{L-1} - 1, m_L$, pois na camada de saída não há *bias* (note a dimensão de $d(n)$)

As variáveis que efetuam o fluxo da rede são y^ℓ para cada camada $\ell = 0, ..., L$ (os superíndices indicam a camada em questão). Essas variáveis representam o valor de saída de cada nó.

1.1 Esboço do fluxo

Todos os índices neste texto foram pensados como iniciando em zero. Destacamos em roxo as principais partes para implementação.

$$y^0(n) = \begin{bmatrix} 1 \\ \bar{x}_0(n) \\ \vdots \\ \bar{x}_{m_0-2}(n) \end{bmatrix} =: \begin{bmatrix} 1 \\ v_0^0(n) \\ \vdots \\ v_{m_0-2}^0(n) \end{bmatrix} \in \mathbb{R}^{m_0}$$

e para as diversas camadas $\ell = 0, ..., L - 1$, temos

$$w^\ell = \begin{bmatrix} b_0^\ell & W_{0,0}^\ell & \cdot\cdot & W_{0,m_\ell-2}^\ell \\ \vdots & & & \\ b_{m_{\ell+1}-2}^\ell & W_{m_{\ell+1}-2,0}^\ell & \cdot\cdot & W_{m_{\ell+1}-2,m_\ell-2}^\ell \end{bmatrix} \in \mathbb{R}^{(m_{\ell+1}-1)\times m_\ell};$$

$$v^{\ell+1}(n) = \begin{bmatrix} v_0^{\ell+1}(n) \\ \vdots \\ v_{m_{\ell+1}-2}^{\ell+1}(n) \end{bmatrix} = w^\ell y^\ell(n) = \begin{bmatrix} b_0^\ell & W_{0,0}^\ell & \cdot\cdot & W_{0,m_\ell-2}^\ell \\ \vdots & & & \\ b_{m_{\ell+1}-2}^\ell & W_{m_{\ell+1}-2,0}^\ell & \cdot\cdot & W_{m_{\ell+1}-2,m_\ell-2}^\ell \end{bmatrix} \begin{bmatrix} 1 \\ v_0^\ell(n) \\ \vdots \\ v_{m_\ell-2}^\ell(n) \end{bmatrix} \in \mathbb{R}^{(m_{\ell+1}-1)},$$

$$y^\ell(n) = \begin{bmatrix} 1 \\ \text{logistic}(v_0^\ell(n)) \\ \vdots \\ \text{logistic}(v_{m_\ell-2}^\ell(n)) \end{bmatrix} \in \mathbb{R}^{m_{\ell+1}};$$

e por fim (não tem *bias* na camada de saída)

$$y^L(n) = \begin{bmatrix} \varphi(v_0^L(n)) \\ \vdots \\ \varphi(v_{m_L-1}^L(n)) \end{bmatrix} \in \mathbb{R}^{m_L}$$

Para cada problema (classificação, regressão), usamos uma função $\varphi(\cdot)$ diferenciada. As variáveis $w^\ell = [b^\ell \;\; W^\ell]$ são os pesos da rede entre as camadas $\ell = 0, ..., L - 1$.

Temos o erro

$$e_k(n) = d_k(n) - y_k^L(n), \qquad k = 0, ..., m_L - 1.$$

ou vetorialmente,

$$e(n) = d(n) - y(n) \in \mathbb{R}^{m_L},$$

e o erro quadrático acumulado

$$E(n) = 0.5 \sum_{k=0}^{m_L-1} e_k^2(n).$$

ou vetorialmente

$$E(n) = 0.5 e(n)^T e(n) \in \mathbb{R}$$

2 Gradiente

Considere os nós i na camada $L-2$, j na camada $L-1$ e k na camada L. Vamos derivar o erro quadrático acumulado em relação aos pesos ajustáveis w^{L-1}. Como a dependência entre essa svariáveis não é direta, aplicamos a regra da cadeia.

A Figura 2 ilustra em laranjado w_{kj}^{L-1} para $k = 0$ e $j = 2$. Note que o erro E depende de w_{02}^{L-1} apenas através de uma única parcela que consta no nó $k = 0$.

Esquema:

$$\begin{array}{r} \text{camada} \rightarrow \\ \text{fluxo} \rightarrow \end{array} \begin{pmatrix} L-1 & L-1 & L & & L & & d_k \downarrow \\ y_j & \xrightarrow{w_{kj}} & v_k & \xrightarrow{\varphi} & y_k & \xrightarrow{-1} & e_k \end{pmatrix}$$

Conforme visto no esquema acima, fazemos

$$\frac{\partial E(n)}{\partial w_{kj}^{L-1}} = \frac{\partial E(n)}{\partial e_k(n)} \frac{\partial e_k(n)}{\partial y_k^L(n)} \frac{\partial y_k^L(n)}{\partial v_k^L(n)} \frac{\partial v_k^L(n)}{\partial w_{kj}^{L-1}} = e_k(n) \cdot (-1) \cdot \varphi'(v_k^L(n)) \cdot y_j^{L-1}(n)$$

Vamos definir o produto "ponto-a-ponto" entre matrizes de mesma dimensão, como sendo

$$[A \bullet B]_{ij} = A_{ij} B_{ij}.$$

Defina

$$\delta_k^{L-1}(n) = -\frac{\partial E(n)}{\partial v_k^L(n)} = e_k(n)\varphi'(v_k^L(n)), \quad k = 0, ..., m_L - 1, \tag{1}$$

ou vetorialmente

- quando $\varphi(z) = \text{logistic}(z)$,

$$\delta^{L-1}(n) = e(n) \bullet \text{logistic}(v^L(n)) \bullet \Big(\mathbf{1} - \text{logistic}(v^L(n))\Big);$$

- quando $\varphi(z) = z$,

$$\delta^{L-1}(n) = e(n);$$

- quando $\varphi(z) = \tanh(z)$,

$$\delta^{L-1}(n) = e(n) \bullet \Big(\mathbf{1} - \tanh(v^L(n)) \bullet \tanh(v^L(n))\Big).$$

Com essa definição, podemos escrever

$$\frac{\partial E(n)}{\partial w_{kj}^{L-1}} = -\delta_k^{L-1}(n) y_j^{L-1}(n), \quad k = 0, ..., m_L - 1. \tag{2}$$

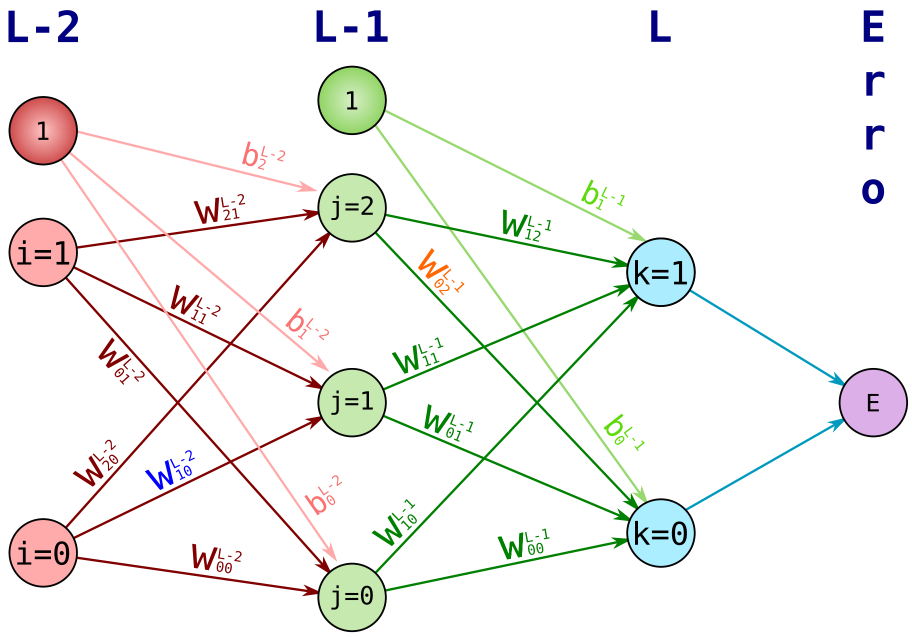

Figura 1: Camadas $\ell = L-2, L-1, L$ e o erro $E(n)$. Os pesos w_{10}^{L-2} e w_{02}^{L-1} foram destacados.

Para calcular a taxa de variação de $E(n)$ em relação a w_{ji}^{L-2}, precisamos seguir o fluxo na rede no sentido “do final para o começo” ou *backward*. Portanto, note que precisamos fazer um somatório, pois as parcelas de todos os nós k são alteradas quando alteramos w_{ji}^{L-2}, o que não acontecia na situação anterior.

$$\begin{aligned}
\frac{\partial E(n)}{\partial w_{ji}^{L-2}} &= \sum_{k=0}^{m_L-1} \frac{\partial E(n)}{\partial e_k(n)} \frac{\partial e_k(n)}{\partial y_k^L(n)} \frac{\partial y_k^L(n)}{\partial v_k^L(n)} \frac{\partial v_k^L(n)}{\partial y_j^{L-1}(n)} \frac{\partial y_j^{L-1}(n)}{\partial v_j^{L-1}(n)} \frac{\partial v_j^{L-1}(n)}{\partial w_{ji}^{L-2}} \\
&= \sum_{k=0}^{m_L-1} e_k(n) \cdot (-1) \cdot \varphi'(v_k^L(n)) \cdot w_{kj}^{L-1} \cdot \text{logistic}'(v_j^{L-1}(n)) \cdot y_i^{L-2}(n) \\
&= -\,\text{logistic}'(v_j^{L-1}(n)) \cdot y_i^{L-2}(n) \sum_{k=0}^{m_L-1} \underbrace{e_k(n) \cdot \varphi'(v_k^L(n))}_{\delta_k^{L-1}(n)} \cdot w_{kj}^{L-1} \\
&= -\,\text{logistic}'(v_j^{L-1}(n)) y_i^{L-2}(n) \sum_{k=0}^{m_L-1} \delta_k^{L-1}(n) w_{kj}^{L-1}.
\end{aligned}$$

Defina

$$\delta_j^{L-2}(n) = -\frac{\partial E(n)}{\partial v_j^{L-1}(n)} = \text{logistic}'(v_j^{L-1}(n)) \sum_{k=0}^{m_L-1} \delta_k^{L-1}(n) w_{kj}^{L-1}, \quad j = 0, ..., m_{L-1} - 1, \tag{3}$$

ou vetorialmente, para a camada $\ell = 0, ..., L-2$,

$$\delta^\ell(n) = \Big[\text{logistic}(v^{\ell+1}(n)) \bullet \big(\mathbf{1} - \text{logistic}(v^{\ell+1}(n))\big)\Big] \bullet \big[(w^{\ell+1})^T \delta^{\ell+1}(n)\big].$$

Com essa definição, escrevemos

$$\frac{\partial E(n)}{\partial w_{ji}^{L-2}} = -\delta_j^{L-2}(n) y_i^{L-2}(n), \quad j = 0, ..., m_{L-1} - 1. \tag{4}$$

O gradiente do erro em relação aos pesos, ou seja, a derivada de primeira ordem da função erro $E(n)$ em relação a cada w_{rs}^ℓ de cada camada $\ell = 0, ..., L-1$ é a coleção de matrizes w^ℓ. Perceba que não é um vetor, nem uma matriz, mas uma coleção de matrizes.

2.1 Direção de descida

Em um método de minimização de primeira ordem, a ideia principal é caminhar na direção oposta à do gradiente. Portanto, o incremento $\Delta w_\rho(n)$ da iteração/ciclo ρ que atualiza cada um dos pesos w, na forma $w_{\rho+1}^\ell \leftarrow w_\rho^\ell + \Delta w_\rho^\ell(n)$, com tamanho de passo $0 < \eta \leq 1$ é dado por

$$\Delta w_{sr}^\ell(n) = \eta \delta_s^\ell(n) y_r^\ell(n), \quad r = 0, ..., m_\ell - 1 \quad s = 0, ..., m_{\ell+1} - 1, \quad \ell = 0, ..., L-1, \tag{5}$$

ou vetorialmente, para $\ell = 0, ..., L-1$

$\Delta w_\rho^\ell(n) = \eta \delta^\ell(n) (y^\ell(n))^T$.

Se quisermos adicionar Momentum, precisamos armazenar $\Delta w_{\rho-1}^\ell(n)$ da iteração $\rho - 1$ e dar uma constante $0 < \alpha < 1$ com

$\Delta w_\rho^\ell(n) = \eta \delta^\ell(n) (y^\ell(n))^T + \alpha \Delta w_{\rho-1}^\ell(n)$.

O termo Momentum nada mais é do que o gradiente da iteração anterior. Para inicializar o termo Momentum, comece-o em zero.

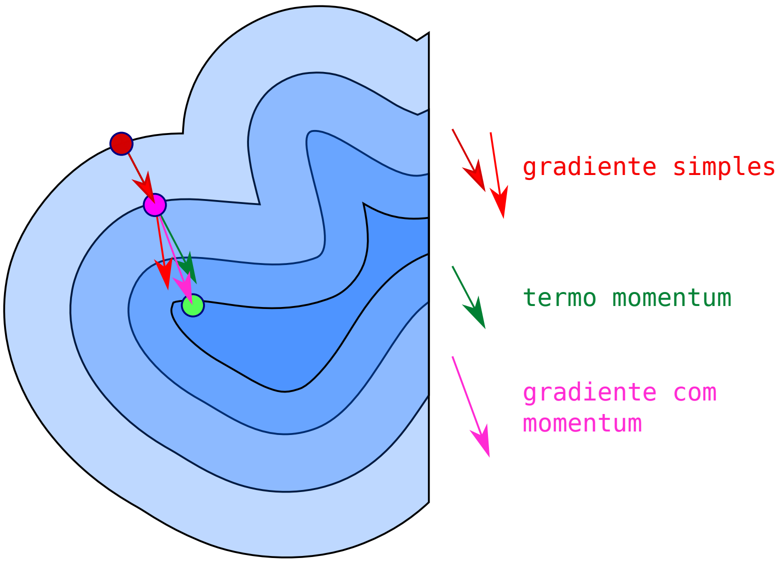

Figura 2: Ilustração do termo Momentum afetando a direção de descida.

3 Função de ativação

Em problemas de regressão segundo [1], usamos uma função linear, $\varphi(z) = z$ e portanto a derivada é a função constante $\varphi'(z) = 1$, vide equação (1).

Em problemas de classificação, usamos $\varphi(z) = \text{logistic}(z)$, uma função com imagem $(0, 1)$:

$$\text{logistic}(z) = \frac{1}{1 + \exp(-z)}$$

$\text{logistic}'(z) = \text{logistic}(z)(1 - \text{logistic}(z))$, vide equações (1) e (3).

Uma alternativa à função logística é usar $\varphi(z) = \tanh(z)$, uma função com imagem $(-1, 1)$:

$$\tanh(z) = \frac{\exp(2z) - 1}{\exp(2z) + 1}$$

$\tanh'(z) = 1 - \tanh^2(z)$, vide equações (1) e (3).

4 Modo cíclico

Agora que já sabemos como calcular o erro E e as taxas de variação de E em função de w em todas as camadas, podemos aplicar qualquer método de minimização de primeira ordem. Na seção "Direção de descida" é dado um método com tamanho de passo fixo η, mas podemos fazer inúmeras variações de algoritmo de forma a torná-lo adaptativo, ou mesmo estocástico. Escrevendo o problema de otimização, temos

$$\underset{W}{\text{minimizar}}\ E(W, n),$$

$$W = \{w^\ell_{i,j} \quad i = 0, ..., m_{\ell+1} - 1, \quad j = 0, ..., m_\ell - 1, \quad \ell = 0, ..., L - 1\}$$

W é a coleção de todos os parâmetros ajustáveis da rede, ou seja, os pesos. O erro $E(W, n)$ depende dos pesos da W rede e do padrão n apresentado.

5 Modo *batch*

Defina o erro quadrático total

$$\mathbf{E} = \sum_{n=0}^{N-1} E(n).$$

Assim, o gradiente total é

$$\frac{\partial \mathbf{E}}{\partial w_{sr}^{\ell}} = \sum_{n=0}^{N-1} \frac{\partial E(n)}{\partial w_{sr}^{\ell}}.$$

A direção de descida global é então a soma das direções de descida para cada padrão. Com isso, podemos aplicar a otimização nessa direção.

O problema de otimização agora é

$$\underset{W}{\text{minimizar}}\ \mathbf{E}(W),$$

$$W = \{w_{i,j}^{\ell} \quad i = 0, ..., m_{\ell+1} - 1, \quad j = 0, ..., m_{\ell} - 1, \quad \ell = 0, ..., L - 1\}.$$

Perceba que a função $\mathbf{E}(W)$ depende apenas dos pesos W da rede e não depende do padrão n, pois já leva em consideração todos os padrões (ou parte deles).

Referências

[1] Riedmiller, Martin. *Machine learning: multi layer perceptrons*. Disponível aqui.

[2] Haykin, Simon. *Neural networks: a comprehensive fondation*. 2a.ed. Singapore: Prentice Hall, 1999. Disponível aqui.

[3] Haykin, Simon. *Neural networks and learning machines*. 3a.ed. New Jersey: Prentice Hall, 2008. Disponível aqui.