Multi-Layer Perceptron (MLP)- Parte II

Profa. Dra. Roseli Aparecida Francelin Romero
SCC - ICMC - USP

2022

Sumário

Teorema da Aproximação Universal (TAU)

- Qual é o número mínimo de camadas em uma MLP que fornece uma aproximação para qualquer mapeamento contínuo?
- Cybenko [1989] mostrou pela primeira vez que uma rede com **uma única camada intermediária** é suficiente para aproximar uniformemente qualquer função contínua definida em um hipercubo unitário.

Teorema da Aproximação Universal (TAU)

- **Teorema:** seja $g(\cdot)$ uma função contínua limitada estritamente crescente. Seja I_p um hipercubo unitário p-dimensional e $C(I_p)$ o espaço das funções contínuas em I_p. Então, dada qualquer função $f \in C(I_p)$ e $\epsilon > 0$, existe um inteiro M e constantes reais α_i, θ_i e w_{ji}, onde $i = 1, 2, \cdots, M$ e $j = 1, 2, \cdots, p$, tal que se pode definir:

$$F(x_1, \cdots, x_p) = \sum \alpha_i g\left(\sum w_{ji} x_j - \theta_i\right) \qquad (1)$$

com

$$|F(x_1, \cdots, x_p) - f(x_1, \cdots, x_p)| < \epsilon \qquad \{x1, \cdots, x_p\} \in I_p$$

Teorema da Aproximação Universal (TAU)

- As funções *sigmoid* ou logística são contínuas, estritamente crescentes e limitadas, portanto satisfazem as condições impostas para a função $g(\cdot)$.
- A equação (1) representa a saída da MLP.
  - A rede tem p nós de entrada e uma única camada intermediária de M nós.
- O neurônio i tem pesos $w_{1i}, \cdots, w_{pi}$ e limiar θ_i.
- A saída da rede é uma combinação linear das saídas dos neurônios intermediários com α_i.

Teorema da Aproximação Universal (TAU)

- Trata-se de um teorema de **existência**, visto que fornece uma justificativa para a aproximação de funções contínuas. $\Longrightarrow$ SUFICIENTE
- Entretanto, ele não afirma que uma única camada é um número **ótimo**.
- Na prática, nem sempre se dispõe de uma função contínua e nem de uma camada intermediária de tamanho qualquer.
- Chester [1990] e Funahashi [1989] defendem o uso de duas camadas intermediárias, tornando a aproximação mais maleável.

Teorema da Aproximação Universal (TAU)

- Características locais são extraídas na primeira camada.
  - Alguns neurônios na primeira camada são usados para particionar o espaço em várias regiões, e outros aprendem as características locais daquelas regiões.
- Características globais são extraídas na segunda camada.
  - Um neurônio na segunda camada combina as saídas de neurônios da primeira que estão operando numa região particular do espaço de entrada e assim aprende características globais daquela região.

Sumário

Sumário

Velocidade de aprendizado

- O algoritmo BP fornece uma aproximação para a trajetória no espaço dos pesos.
- Quanto menor o valor de η, menores as mudanças nos pesos e mais suave será a trajetória.
  - **Aprendizado lento.**
- Se η é muito grande, o aprendizado torna-se rápido, porém a rede pode tornar-se **instável**.

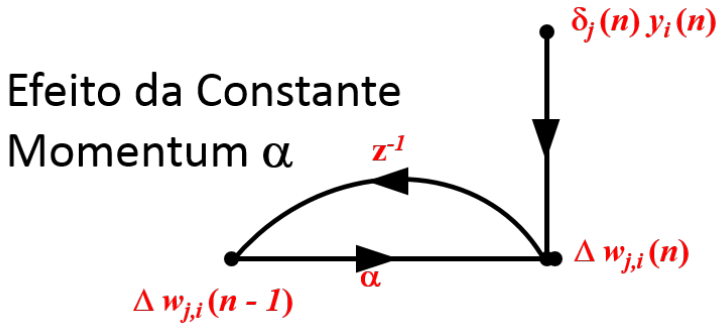
Sumário

Efeito da constante α

- É um método simples de aumentar a velocidade do aprendizado e evitar o perigo de instabilidade, como mostrado por Rumelhart *et al.*, 1986.

$$\Delta w_{ji}(n) = \eta \delta_j(n) y_i(n) + \alpha \Delta w_{ji}(n-1) \qquad (2)$$

- Onde α é geralmente um número positivo chamado **constante momentum**.



A equação (α) é chamada *REGRA DELTA GENERALIZADA*. Se α =0 $\Rightarrow$ REGRA DELTA

Gradiente

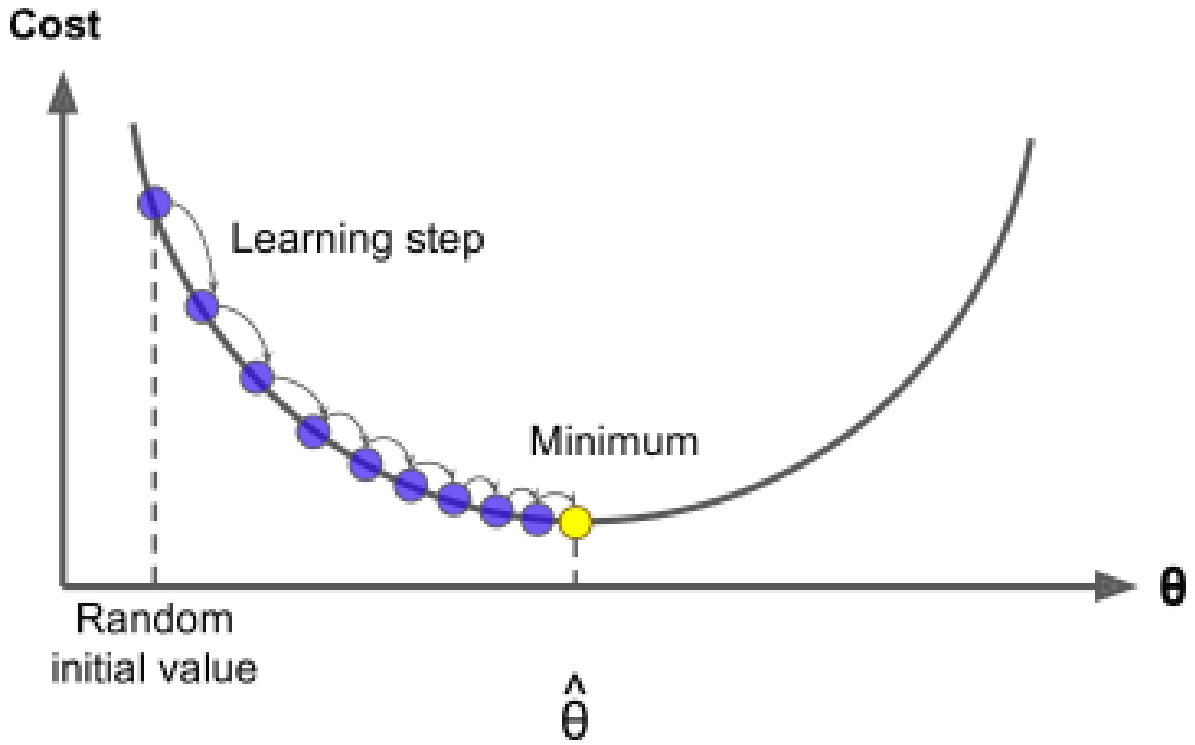


Figura 1: Gradiente Descent(Geron$_2$017)

Efeito da constante α

- Vamos considerar uma série de tempo com índice t (de 0 a n).
- A equação 2 pode ser vista como uma equação diferencial de primeira ordem em relação a $\Delta w_{ji}(n)$. Resolvendo:

$$\Delta w_{ji}(n) = \eta \sum_{t=0}^{n} \alpha^{n-t} \delta_j(t) y_i(t) \tag{3}$$

- Que representa uma série de tempo comprimido $n+1$. Mas:

$$\delta_j(n) y_i(n) = -\frac{\partial E(n)}{\partial w_{ji}(n)}$$

$$\therefore \Delta w_{ji}(n) = -\eta \sum_{t=0}^{n} \alpha^{n-t} \frac{\partial E(t)}{\partial w_{ji}(t)} \tag{4}$$

Efeito da constante α

1. O ajuste atual $\Delta w_{ji}(n)$ representa a soma de uma série temporal ponderada exponencialmente convergente $\Longrightarrow$ $0 \leq |\alpha| < 1$
2. Quando $\frac{\partial E(t)}{\partial w_{ji}(t)}$ tem o mesmo sinal algébrico em iterações consecutivas, então a série cresce em magnitude e os pesos são ajustados por uma quantidade grande. Portanto, o BP tende a acelerar a "descida"nas regiões de descida da superfície do erro.
3. Quando $\frac{\partial E(t)}{\partial w_{ji}(t)}$ tem sinais opostos em iterações sucessivas, então a série diminui em magnitude, e $\Delta w_{ji}(n)$ é atualizado por uma quantidade pequena. Então, a inclusão do termo *momentum* tem o **efeito de estabilização** nas direções em que o sinal oscila.

Efeito da constante α

- Portanto, o termo *momentum* pode ter efeitos benéficos no comportamento do aprendizado do algoritmo. Ele pode evitar que o processo termine em um mínimo local na superfície do erro.
- **Observação:** o parâmetro η foi considerado constante.
  1. η_{ji} **dependente da conexão**: fatos interessantes ocorrem se η_{ji} é tomado diferente em diferentes partes do algoritmo.
  2. **Restringir o número de pesos a serem ajustados**: $\eta_{ji} = 0$ para o peso w_{ji}.

Efeito da constante α

- Modo segundo o qual as camadas ocultas são interconectadas: no procedimento, supomos que cada camada recebe entradas apenas das unidades da **camada anterior**.
- Não existe uma razão para isso. Se esse não for o caso, existem dois tipos de sinais de erro:
  - Um sinal de erro que resulta de uma comparação direta do sinal de saída daquele neurônio como uma resposta desejada.
  - Um sinal de erro que é passado através de outras unidades cuja ativação ele afeta.'

Sumário

Modos de treinamento

- Aprendizado BP resulta de muitas apresentações de um conjunto de treinamento de exemplos.
- Uma apresentação **completa** do conjunto de treinamento corresponde a 1 ciclo (*epoch*).
- O processo de aprendizado é repetido ciclo após ciclo, até que os pesos sinápticos e níveis *threshold* se **estabilizem**.
- Tomar os pesos em uma forma aleatória $\rightarrow$ pesquisa no espaço dos pesos **estocástica**.

Modo padrão

- **(1) Modo padrão:**
  - Atualização nos pesos é feita após a apresentação de cada exemplo de treinamento.
  - Um ciclo consistindo de N exemplos de treinamento, arranjados na ordem:

$$\{[\mathbf{x_1}, \mathbf{d_1}], [\mathbf{x_2}, \mathbf{d_2}], \cdots, [\mathbf{x_N}, \mathbf{d_N}]\}$$

  - $[\mathbf{x_1}, \mathbf{d_1}] \rightarrow$ cálculos*forward/backward* e atualização dos pesos.
  - $[\mathbf{x_2}, \mathbf{d_2}] \rightarrow$ cálculos*forward/backward* e atualização dos pesos.

$$\vdots$$

  - $[\mathbf{x_N}, \mathbf{d_N}] \rightarrow$ cálculos*forward/backward* e atualização dos pesos.

Modo padrão

- Dessa forma, a variação média nas mudanças dos pesos é:

$$\Delta\hat{w}_{ji} = \frac{1}{N}\sum_{n=1}^{N}\Delta w_{ji}(n)$$

$$= -\frac{\eta}{N}\sum_{n=1}^{N}\frac{\partial E(n)}{\partial w_{ji}(n)} \implies$$

$$\boxed{\Delta\hat{w}_{ji} = -\frac{\eta}{N}\sum_{n=1}^{N} e_j(n)\frac{\partial e_j(n)}{\partial w_{ji}(n)}} \quad (5)$$

Modo *batch*

- **(2) Modo batch**:
  - Atualização dos pesos é feita depois da apresentação de todos os exemplos de treinamento que constituem um ciclo.
  - Para um ciclo, função custo com o erro quadrático médio:

$$\mathcal{E}_{av} = \frac{1}{2N} \sum_{n=1}^{N} \sum_{j \in C} e_j^2(n) \tag{6}$$

  - Onde C denota o conjunto de índices correspondentes aos neurônios da camada de saída e e_j é o sinal do erro do neurônio j correspondente ao exemplo de treinamento w.

$$\Delta w_{ji} = -\eta \frac{\partial \mathcal{E}_{av}}{\partial w_{ji}} \implies \boxed{\Delta w_{ji} = \frac{\eta}{N} \sum_{n=1}^{N} e_j(n) \frac{\partial e_j(n)}{\partial w_{ji}}}$$

Modos de treinamento - comparação

- Claramente, $\Delta\hat{w}_{ji}$ é diferente de Δw_{ji}.
  - $\Delta\hat{w}_{ji}$ representa uma **estimativa** de Δw_{ji}.
- Do ponto de vista *online*, o **modo padrão** é preferido. Além disso, os exemplos de treinamento são aleatoriamente apresentados (atualização nos pesos é **estocástica**) $\rightarrow$ menos provável o algoritmo BP estacionar em um mínimo local.
- Por outro lado, o **modo batch** fornece uma estimativa mais precisa do **vetor gradiente**.
- De qualquer forma, a eficiência dos dois modos depende do problema que se tem em mãos (Hertz, 1991).

Sumário

Critério de parada

- Não se pode, em geral, mostrar a convergência do algoritmo BP, tampouco existem critérios bem definidos para encerrar seu processamento.
- Para formular um critério, devem-se considerar propriedades de mínimo local ou global da superfície de erro.

Critério de parada

- Seja $\mathbf{w}^*$ o vetor mínimo local ou global.

1. Uma condição necessária para $\mathbf{w}^*$ ser mínimo:
   - O gradiente (derivada de primeira ordem) da superfície de erro em relação a $\mathbf{w}$ seja zero em $\mathbf{w} = \mathbf{w}^*$, isto é, $\nabla g(w) = 0$ em $\mathbf{w} = \mathbf{w}^*$.
   - Diz-se que o algoritmo BP convergiu se a norma do vetor gradiente é menor que um certo ϵ pequeno arbitrário.
2. Função custo $\mathcal{E}_{av}(w)$ é estacionária em $\mathbf{w} = \mathbf{w}^*$.
   - Diz-se que o algoritmo BP convergiu se a taxa de mudança no erro quadrático médio por ciclo é suficientemente pequena.
   - Tipicamente, são consideradas pequenas taxas de mudanças no erro de 0.1% a 1% ou de 0.01%.

Critério de parada

- Kramer e Sangiovanni-Vicentelli(1989) sugerem um critério de convergência:

O algoritmo BP termina no vetor peso w_{final} *quando* $\| g(w_{final}) \| \leq \varepsilon$, onde ε é suficiente pequeno, ou $|| \varepsilon_{av}(\text{ final }) || \leq \tau$ onde τ é suficiente pequeno.

Sumário

Generalização

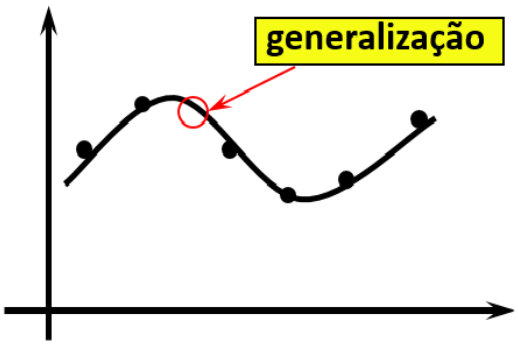
Aprendizado **BP**

conjunto treinamento + algoritmo BP $\Rightarrow$ pesos sinápticos

GENERALIZAR

"GENERALIZAÇÃO": termo da psicologia.

Processo de aprendizado pode ser visto como um Método de Aproximação de Funcões



: efeito de uma boa aproximação não linear dos dados de entrada, tamanho e eficiência do conjunto treinamento, arquitetura da rede, complexidade física do problema

Complexidade da rede

- **Problema:** determinar o melhor número de nós na camada intermediária.
- Estatisticamente, esse problema é equivalente a determinar o tamanho do conjunto de parâmetros usado para modelar o conjunto de dados. Existe um limite no tamanho da rede.
- Esse limite deve ser tomado lembrando que é melhor treinar a rede para **produzir a melhor generalização** do que treinar a rede para representar perfeitamente um conjunto de dados.
- Isso pode ser feito usando **validação cruzada**.

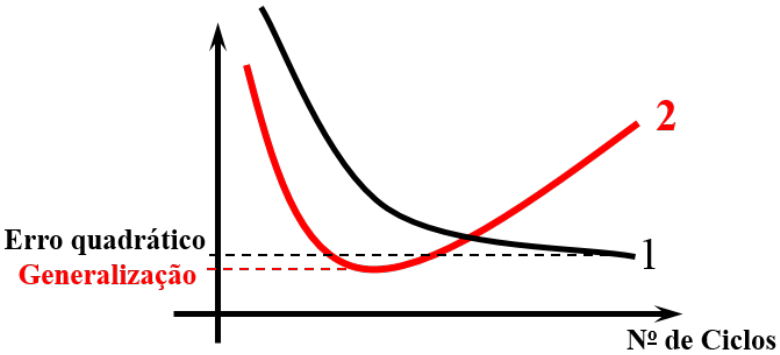
Validação cruzada

- Conjunto de dados:
  - Treinamento (75%)
  - Teste (25%)
- Conjunto de treinamento:
  - Um subconjunto para validação do modelo.
  - Um subconjunto para treinamento.
- Validar o modelo em um conjunto diferente do usado para estimá-lo.

Validação cruzada

- Usa-se o subconjunto de validação para avaliar o desempenho de diferentes candidatos do modelo (diferentes topologias) e, então, escolhe-se uma delas.
- O modelo escolhido é treinado sobre o conjunto de treinamento inteiro e a capacidade de generalização é medida no conjunto de teste.
- A validação cruzada pode ser usada para decidir quando o treinamento de uma rede deve ser encerrado.

Tamanho do conjunto de treinamento



Curva 1: poucos parâmetros (*underfitting*)

Curva 2: muitos parâmetros (*overfitting*)

- Em ambos os casos:
  1. O desempenho do erro na generalização exibe um mínimo.
  2. O mínimo no caso *overfitting* é menor e mais definido.
- Pode-se obter boa generalização se a rede é projetada com muitos neurônios, contanto que o treinamento seja cessado após um número de ciclos correspondente ao mínimo da curva do **erro** obtida na **validação cruzada**.

Sumário

Inicialização

- O primeiro passo do algoritmo BP é a inicialização da rede.
- Uma boa escolha para os parâmetros livres (pesos sinápticos e *threshold*) podem contribuir significativamente no sucesso do aprendizado.

Inicialização

- **Informação disponível**
- **Nenhuma informação disponível?**
  - Pesos inicializados aleatoriamente, isto é, inicializar os pesos com valores uniformemente distribuídos em um intervalo pequeno.
- **Escolha errada $\Longrightarrow$ saturação prematura**
  - Esse fenômeno se refere a uma situação na qual o erro quadrático permanece constante por um período de tempo, porém continua a diminuir depois que este período é concluído.

Inicialização

- Fatos interessantes podem ocorrer:

1. Suponha que, para um particular padrão de treinamento, o nível de ativação interna de um neurônio saída tenha um valor cuja magnitude é grande (como a função é *sigmoid*, trata-se de um caso em que $y = 1$ ou $y = -1$). Em tal caso , diz-se que o neurônio está em **saturação**.
2. Se y está mais próximo de 1 quando a saída desejada é -1, ou vice-versa, o neurônio está **incorretamente saturado**.
   - Quando isso ocorre, o ajuste nos pesos será pequeno, embora o erro seja de magnitude grande, e a rede levará um longo tempo para corrigir essa situação (Lee,1991).
3. No estágio inicial do BP, podem existir neurônios não-saturados ou incorretamente saturados.

Inicialização

- Para os não-saturados $\rightarrow$ os pesos mudam rapidamente.
- Para os incorretamente saturados $\rightarrow$ permanecem saturados por algum tempo.
- **Fenômeno da saturação prematura** pode ocorrer, com $\mathcal{E}$ permanecendo constante.

Inicialização

- Em Lee(1991), uma fórmula para a probabilidade de **saturação prematura** foi obtida para o **modo batch**.
- A essência dessa fórmula pode ser: [Haykin, 1994]
  1. **Saturação incorreta** é evitada escolhendo valores iniciais dos pesos sinápticos e níveis *threshold*, uniformemente distribuídos em um intervalo pequeno.
  2. É menos provável quando o número de neurônios intermediários é mantido baixo.
  3. Raramente ocorre quando os neurônios da rede operam em sua regiões lineares.
- Segundo [Haykin,1994], para o **modo padrão** de atualização dos pesos, os resultados mostram uma tendência similar ao **modo batch**.