Notas de Aula - Semana 2

Introdução às Redes Neurais

Curso de Redes Neurais

Eduardo Adame

20 de agosto de 2025

Sumário

1 Introdução

Nesta segunda aula, fazemos a transição dos métodos tradicionais de machine learning para as redes neurais artificiais. Veremos como o conceito de regressão logística, que estudamos na semana passada, se generaliza para criar modelos muito mais poderosos e flexíveis.

As redes neurais representam um paradigma fundamental no aprendizado de máquina moderno. Inspiradas pelo funcionamento do cérebro humano, essas estruturas computacionais são capazes de aprender representações hierárquicas complexas dos dados, tornando-se a base para os avanços recentes em visão computacional, processamento de linguagem natural e muitas outras áreas.

1.1 Objetivos de Aprendizagem

Ao final desta aula, o estudante será capaz de:

- Compreender a inspiração biológica das redes neurais
- Implementar um neurônio artificial (perceptron)
- Entender a relação entre neurônios e regressão logística
- Construir redes neurais feedforward simples
- Realizar o forward pass usando operações matriciais
- Identificar a necessidade de funções de ativação não-lineares

1.2 Conexão com a Aula Anterior

Na semana passada, estudamos:

- Regressão linear: $y = \mathbf{w}^T\mathbf{x} + b$
- Descida de gradiente: $\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \alpha\nabla J(\mathbf{w}^{(t)})$
- Função de custo MSE: $J(\mathbf{w}) = \frac{1}{2n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$

Hoje veremos como esses conceitos se estendem para redes neurais, onde:

- Cada neurônio realiza uma operação similar à regressão
- Múltiplos neurônios são organizados em camadas
- A descida de gradiente continua sendo o método de otimização

2 Inspiração Biológica

2.1 O Neurônio Biológico

O neurônio biológico é a unidade fundamental do sistema nervoso. Suas principais componentes são:

- **Dendritos:** Recebem sinais de outros neurônios
- **Corpo celular (soma):** Processa e integra os sinais recebidos
- **Axônio:** Transmite o sinal de saída
- **Sinapses:** Conexões com outros neurônios

O neurônio biológico funciona de forma simplificada assim:

1. Recebe múltiplos sinais através dos dendritos
2. Integra esses sinais no corpo celular
3. Se a soma dos sinais excede um limiar, dispara um potencial de ação
4. Transmite o sinal através do axônio para outros neurônios

2.2 Do Biológico ao Artificial

O neurônio artificial é uma abstração matemática inspirada no neurônio biológico:

Neurônio Biológico	**Neurônio Artificial**
Dendritos	Entradas $(x_1, x_2, ..., x_n)$
Força sináptica	Pesos $(w_1, w_2, ..., w_n)$
Corpo celular	Função de agregação ($\sum$)
Limiar de ativação	Bias (b)
Potencial de ação	Função de ativação (f)
Axônio	Saída (y)

3 O Neurônio Artificial

3.1 Modelo Matemático

Definição 3.1 (Neurônio Artificial) *Um neurônio artificial é uma unidade computacional que:*

1. *Recebe um vetor de entrada* $\mathbf{x} = [x_1, x_2, ..., x_n]^T$
2. *Calcula uma combinação linear ponderada:* $z = \mathbf{w}^T\mathbf{x} + b$
3. *Aplica uma função de ativação:* $a = f(z)$

Matematicamente:

$$a = f\left(\sum_{i=1}^{n} w_i x_i + b\right) = f(\mathbf{w}^T\mathbf{x} + b) \tag{1}$$

onde:

- $\mathbf{x} \in \mathbb{R}^n$ é o vetor de entrada
- $\mathbf{w} \in \mathbb{R}^n$ é o vetor de pesos

- $b \in \mathbb{R}$ é o bias (viés)
- $z = \mathbf{w}^T\mathbf{x} + b$ é chamado de "net input"ou entrada líquida
- $f : \mathbb{R} \to \mathbb{R}$ é a função de ativação
- a é a saída (ativação) do neurônio

3.2 Relação com Regressão Logística

Teorema 3.1 (Equivalência com Regressão Logística) *Um neurônio artificial com função de ativação sigmoid é matematicamente equivalente à regressão logística.*

Demonstração: Na regressão logística, temos:

$$P(y = 1|\mathbf{x}) = \sigma(\mathbf{w}^T\mathbf{x} + b) = \frac{1}{1 + e^{-(\mathbf{w}^T\mathbf{x}+b)}} \tag{2}$$

Um neurônio com ativação sigmoid calcula:

$$a = \sigma(z) = \sigma(\mathbf{w}^T\mathbf{x} + b) = \frac{1}{1 + e^{-(\mathbf{w}^T\mathbf{x}+b)}} \tag{3}$$

Portanto, são idênticos! A diferença está na interpretação:

- Regressão logística: modelo probabilístico para classificação binária
- Neurônio: unidade computacional que pode ser combinada em rede

4 Funções de Ativação

4.1 Por que Funções de Ativação?

Funções de ativação introduzem não-linearidade na rede neural. Sem elas, múltiplas camadas lineares colapsariam em uma única transformação linear.

Teorema 4.1 (Composição de Funções Lineares) *A composição de qualquer número de transformações lineares é ainda uma transformação linear.*

Prova: Sejam duas transformações lineares: $f_1(\mathbf{x}) = W_1\mathbf{x} + \mathbf{b}_1$ e $f_2(\mathbf{x}) = W_2\mathbf{x} + \mathbf{b}_2$. A composição é:

$$\begin{aligned} f_2(f_1(\mathbf{x})) &= W_2(W_1\mathbf{x} + \mathbf{b}_1) + \mathbf{b}_2 && (4)\\ &= W_2W_1\mathbf{x} + W_2\mathbf{b}_1 + \mathbf{b}_2 && (5)\\ &= W_{comp}\mathbf{x} + \mathbf{b}_{comp} && (6) \end{aligned}$$

onde $W_{comp} = W_2W_1$ e $\mathbf{b}_{comp} = W_2\mathbf{b}_1 + \mathbf{b}_2$.

4.2 Função Sigmoid

A função sigmoid é uma das funções de ativação mais clássicas:

Definição 4.1 (Função Sigmoid)

$$\sigma(z) = \frac{1}{1 + e^{-z}} \tag{7}$$

Propriedades:

- Domínio: $(-\infty, +\infty)$
- Imagem: $(0, 1)$
- Interpretação probabilística
- Diferenciável em todo domínio
- Saturação para valores extremos

Teorema 4.2 (Derivada da Sigmoid) *A derivada da função sigmoid tem uma forma elegante:*

$$\sigma'(z) = \sigma(z)(1 - \sigma(z)) \tag{8}$$

Demonstração:

$$\begin{aligned}
\sigma'(z) &= \frac{d}{dz}\left(\frac{1}{1 + e^{-z}}\right) && (9)\\
&= \frac{d}{dz}(1 + e^{-z})^{-1} && (10)\\
&= -(1 + e^{-z})^{-2} \cdot (-e^{-z}) && (11)\\
&= \frac{e^{-z}}{(1 + e^{-z})^2} && (12)\\
&= \frac{1}{1 + e^{-z}} \cdot \frac{e^{-z}}{1 + e^{-z}} && (13)\\
&= \sigma(z) \cdot (1 - \sigma(z)) && (14)
\end{aligned}$$

Esta propriedade será extremamente útil quando calcularmos gradientes na retropropagação.

4.3 Outras Funções de Ativação Comuns

4.3.1 Tangente Hiperbólica (tanh)

$$\tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}} = \frac{2}{1 + e^{-2z}} - 1 \tag{15}$$

Propriedades:

- Imagem: $(-1, 1)$
- Centrada em zero
- Derivada: $\tanh'(z) = 1 - \tanh^2(z)$

4.3.2 ReLU (Rectified Linear Unit)

$$\text{ReLU}(z) = \max(0, z) = \begin{cases} z & \text{se } z > 0 \\ 0 & \text{caso contrário} \end{cases} \tag{16}$$

Propriedades:

- Computacionalmente eficiente
- Não satura para valores positivos
- Problema do "dying ReLU"
- Derivada: $\text{ReLU}'(z) = \begin{cases} 1 & \text{se } z > 0 \\ 0 & \text{caso contrário} \end{cases}$

5 Redes Neurais Feedforward

5.1 Arquitetura em Camadas

Uma rede neural feedforward organiza neurônios em camadas sequenciais:

Definição 5.1 (Rede Neural Feedforward) *Uma rede neural feedforward é composta por:*

1. ***Camada de entrada:*** *Recebe os dados (não realiza computação)*
2. ***Camadas ocultas:*** *Uma ou mais camadas de neurônios que processam informação*
3. ***Camada de saída:*** *Produz o resultado final*

A informação flui em uma única direção: entrada $\rightarrow$ ocultas $\rightarrow$ saída.

5.2 Notação

Para uma rede com L camadas:

- l denota o índice da camada, $l \in \{1, 2, ..., L\}$
- $n^{(l)}$ é o número de neurônios na camada l
- $W^{(l)} \in \mathbb{R}^{n^{(l-1)} \times n^{(l)}}$ é a matriz de pesos da camada $l - 1$ para l
- $\mathbf{b}^{(l)} \in \mathbb{R}^{n^{(l)}}$ é o vetor de bias da camada l
- $\mathbf{z}^{(l)}$ é o vetor de net inputs da camada l
- $\mathbf{a}^{(l)}$ é o vetor de ativações da camada l

5.3 Forward Pass

O forward pass (propagação direta) calcula a saída da rede dado uma entrada:

Exemplo 5.1 (Forward Pass) *Para uma entrada* $\mathbf{x}$*, o forward pass calcula iterativamente:*

$$\mathbf{a}^{(0)} = \mathbf{x} \tag{17}$$

$$\mathbf{z}^{(l)} = W^{(l)T}\mathbf{a}^{(l-1)} + \mathbf{b}^{(l)} \quad \text{para } l = 1, ..., L \tag{18}$$

$$\mathbf{a}^{(l)} = f^{(l)}(\mathbf{z}^{(l)}) \quad \text{para } l = 1, ..., L \tag{19}$$

$$\hat{\mathbf{y}} = \mathbf{a}^{(L)} \tag{20}$$

5.4 Representação Matricial

Para processar múltiplas amostras simultaneamente, usamos notação matricial:
Seja $X \in \mathbb{R}^{m \times n^{(0)}}$ uma matriz com m amostras (linhas).

$$A^{(0)} = X \tag{21}$$

$$Z^{(l)} = A^{(l-1)}W^{(l)} + \mathbf{1}_m \mathbf{b}^{(l)T} \tag{22}$$

$$A^{(l)} = f^{(l)}(Z^{(l)}) \tag{23}$$

$$\hat{Y} = A^{(L)} \tag{24}$$

onde $\mathbf{1}_m$ é um vetor coluna de uns de tamanho m.

6 Exemplo Prático

6.1 Rede com 2 Camadas Ocultas

Considere uma rede neural com:

- Entrada: 3 features ($n^{(0)} = 3$)
- Primeira camada oculta: 4 neurônios ($n^{(1)} = 4$)
- Segunda camada oculta: 4 neurônios ($n^{(2)} = 4$)
- Saída: 3 classes ($n^{(3)} = 3$)

Exemplo 6.1 (Cálculo do Forward Pass) *Dado* $\mathbf{x} = [0.9, 0.2, 0.3]^T$ *e pesos inicializados aleatoriamente, vamos calcular a saída.*
Primeira camada oculta:

$$\mathbf{z}^{(1)} = W^{(1)T}\mathbf{x} + \mathbf{b}^{(1)} \tag{25}$$

$$\mathbf{a}^{(1)} = \sigma(\mathbf{z}^{(1)}) \tag{26}$$

Se $W^{(1)} \in \mathbb{R}^{3\times 4}$ *e* $\mathbf{b}^{(1)} \in \mathbb{R}^4$*, então* $\mathbf{a}^{(1)} \in \mathbb{R}^4$*.*

Segunda camada oculta:

$$\mathbf{z}^{(2)} = W^{(2)T}\mathbf{a}^{(1)} + \mathbf{b}^{(2)} \tag{27}$$

$$\mathbf{a}^{(2)} = \sigma(\mathbf{z}^{(2)}) \tag{28}$$

Camada de saída:

$$\mathbf{z}^{(3)} = W^{(3)T}\mathbf{a}^{(2)} + \mathbf{b}^{(3)} \tag{29}$$

$$\hat{\mathbf{y}} = \mathit{softmax}(\mathbf{z}^{(3)}) \tag{30}$$

6.2 Função Softmax

Para problemas de classificação multiclasse, usamos softmax na camada de saída:

Definição 6.1 (Função Softmax)

$$\mathit{softmax}(\mathbf{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \tag{31}$$

onde K é o número de classes.

Propriedades:

- Saída forma uma distribuição de probabilidade
- $\sum_{i=1}^{K} \text{softmax}(\mathbf{z})_i = 1$
- $0 < \text{softmax}(\mathbf{z})_i < 1$ para todo i

7 Por que Redes Profundas?

7.1 Limitações de um Único Neurônio

Um único neurônio (como na regressão logística) só pode aprender fronteiras de decisão lineares. Isso limita severamente os problemas que podem ser resolvidos.

Exemplo 7.1 (Problema XOR) *O problema XOR não é linearmente separável:*

x_1	x_2	*XOR*
0	*0*	*0*
0	*1*	*1*
1	*0*	*1*
1	*1*	*0*

Não existe uma linha reta que separe as classes 0 e 1.

7.2 Poder das Redes Multicamadas

Teorema 7.1 (Teorema da Aproximação Universal) *Uma rede neural feedforward com uma única camada oculta contendo um número finito de neurônios pode aproximar qualquer função contínua em subconjuntos compactos de $\mathbb{R}^n$, sob suposições brandas sobre a função de ativação.*

Este teorema, provado por Cybenko (1989) e Hornik (1991), garante que redes neurais são aproximadores universais de funções.

7.3 Profundidade vs. Largura

Embora uma única camada oculta seja teoricamente suficiente, redes profundas (múltiplas camadas) são mais eficientes:

- **Redes rasas e largas:** Precisam de exponencialmente mais neurônios
- **Redes profundas:** Aprendem representações hierárquicas
- **Composicionalidade:** Camadas sucessivas aprendem features cada vez mais abstratas

8 Conclusão

Nesta segunda aula, estabelecemos os fundamentos das redes neurais artificiais, fazendo a transição natural dos métodos tradicionais de machine learning para arquiteturas mais sofisticadas e poderosas.

8.1 Principais Conceitos Abordados

Exploramos os conceitos fundamentais que formam a base das redes neurais modernas:

- A inspiração biológica dos neurônios artificiais e como ela se traduz em modelos matemáticos elegantes
- A equivalência entre um neurônio artificial e a regressão logística, demonstrando a continuidade conceitual
- A importância crítica das funções de ativação não-lineares para conferir poder expressivo às redes
- A arquitetura feedforward e sua representação matricial eficiente
- O forward pass como processo computacional fundamental