Inteligência Artificial

Raoni F. S. Teixeira

Aula 9 - Redes Neurais - Treinamento

1 Introdução

Esta aula apresenta o método backpropagation, usado para treinar redes neurais artificiais. O termo 'backpropagation' ('propagação para trás dos erros') refere-se à aplicação da regra da cadeia para ajustar os parâmetros da rede e reduzir os erros de previsão.

2 Backpropagation

Uma rede neural possui dois componentes principais:

1. **Arquitetura da rede:** Número de camadas e de neurônios e tipo de conexões.
2. **Parâmetros:** valores ajustáveis que a rede aprende durante o treinamento, conhecidos também como pesos.

Os componentes fixos são definidos pela arquitetura da rede, enquanto os parâmetros ajustáveis são aprendidos durante o treinamento.

2.1 Etapas do Treinamento

1. **Inicialização:** Os parâmetros são inicializados com valores pequenos e aleatórios, geralmente seguindo uma distribuição normal $\mathcal{N}(0, 0.1)$. A inicialização evita valores nulos que inviabilizam o aprendizado.
2. **Previsão (forward pass):** Para cada amostra de treinamento (x, y), os dados de entrada x passam pela rede camada por camada. As ativações $a^{[k]}$ e pré-ativações $z^{[k]}$ são calculadas até gerar a previsão $\hat{y}$.

3. **Função de custo:** A previsão $\hat{y}$ é comparada com a saída esperada y por meio da função custo. Para classificação, utiliza-se a função *log-loss*:

$$J(\hat{y}, y) = -\left[y \log(\hat{y}) + (1-y)\log(1-\hat{y})\right].$$

O valor de J é monitorado, e sua redução indica melhora no modelo.

4. **Propagação de erros (backward pass):** Nesta etapa, os parâmetros de cada camada ℓ são ajustados para minimizar a função de custo J. A atualização segue as equações:

$$W^{[\ell]} = W^{[\ell]} - \alpha \frac{\partial J}{\partial W^{[\ell]}}, \quad b^{[\ell]} = b^{[\ell]} - \alpha \frac{\partial J}{\partial b^{[\ell]}}, \tag{1}$$

em que α é a taxa de aprendizado.

Os gradientes são obtidos pela regra da cadeia, considerando o erro na camada ℓ:

$$\delta^{[\ell]} = \frac{\partial J}{\partial z^{[\ell]}}. \tag{2}$$

O cálculo do erro e dos gradientes segue três passos principais:

- **Passo 1:** Calcular o erro na camada de saída:

$$\delta^{[L]} = a^{[L]} - y. \tag{3}$$

- **Passo 2:** Para cada camada $\ell \in \{L-1, L-2, \ldots, 2\}$, propagar o erro para trás:

$$\delta^{[\ell]} = (W^{[\ell+1]} \circ \delta^{[\ell+1]})\sigma'(z^{[\ell]}), \tag{4}$$

onde $\circ$ representa a multiplicação elemento a elemento, e σ' é a derivada da função de ativação.

- **Passo 3:** Calcular os gradientes para atualização dos parâmetros:

$$\frac{\partial J}{\partial W^{[\ell]}} = \delta^{[\ell]} a^{[\ell+1]T}, \quad \frac{\partial J}{\partial b^{[\ell]}} = \delta^{[\ell]}. \tag{5}$$

As Equações 4 e 5 derivam do reuso de cálculos da regra da cadeia, com os gradientes sendo propagados camada por camada:

$$\frac{\partial J}{\partial W^{[\ell]}} = \frac{\partial J}{\partial z^{[\ell+1]}} \frac{\partial z^{[\ell+1]}}{\partial a^{[\ell]}} \frac{\partial a^{[\ell]}}{\partial z^{[\ell]}} \frac{\partial z^{[\ell]}}{\partial W^{[\ell]}} \tag{6}$$

e

$$\frac{\partial J}{\partial b^{[\ell]}} = \frac{\partial J}{\partial z^{[\ell+1]}} \frac{\partial z^{[\ell+1]}}{\partial a^{[\ell]}} \frac{\partial a^{[\ell]}}{\partial z^{[\ell]}} \frac{\partial z^{[\ell]}}{\partial b^{[\ell]}}. \tag{7}$$

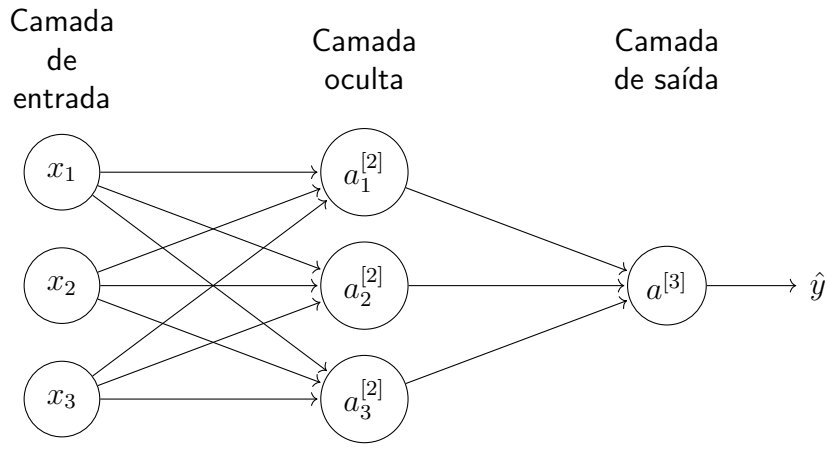

Figura 1: Rede neural com três camadas e ativação sigmoide.

Esses gradientes são definidos como:

$$\frac{\partial J}{\partial z^{[\ell+1]}} = \delta^{[\ell+1]}, \quad \frac{\partial z^{[\ell+1]}}{\partial a^{[\ell]}} = W^{[\ell+1]}, \quad \frac{\partial a^{[\ell]}}{\partial z^{[\ell]}} = \sigma'(z^{[\ell]}), \quad \frac{\partial z^{[\ell]}}{\partial W^{[\ell]}} = a^{[\ell+1]T}, \quad \frac{\partial z^{[\ell]}}{\partial b^{[\ell]}} = 1. \tag{8}$$

Para verificar a consistência do método de propagação de erros com a regra da cadeia, seguimos os passos abaixo: A Figura 1 mostra uma rede neural com três camadas e ativação sigmoide. Os parâmetros dessa rede são definidos como:

- $W^{[2]} \in \mathbb{R}^{3\times3}$, $b^{[2]} \in \mathbb{R}^{1\times3}$,
- $W^{[3]} \in \mathbb{R}^{1\times3}$, $b^{[3]} \in \mathbb{R}^{1\times1}$.

As ativações $a^{[k]}$ e pré-ativações $z^{[k]}$ são definidas como:

1. $z^{[2]} = W^{[2]}x + b^{[2]}$,
2. $a^{[2]} = \sigma(z^{[2]})$,
3. $z^{[3]} = W^{[3]}a^{[2]} + b^{[3]}$,
4. $a^{[3]} = \sigma(z^{[3]})$.

Usando o método de propagação de erros, temos:

$$\frac{\partial J}{\partial W^{[2]}} = (W^{[3]} \circ \delta^{[3]})\sigma'(z^{[2]}) = (W^{[3]^T} \circ \sigma'(z^{[2]}))(a^{[3]} - y)x^T. \tag{9}$$

Aplicando a regra da cadeia o gradiente é:

$$\frac{\partial J}{\partial W^{[2]}} = \frac{\partial J}{\partial a^{[3]}} \frac{\partial a^{[3]}}{\partial z^{[3]}} \frac{\partial z^{[3]}}{\partial a^{[2]}} \frac{\partial a^{[2]}}{\partial z^{[2]}} \frac{\partial z^{[2]}}{\partial W^{[2]}}. \tag{10}$$

Sabendo que $\sigma'(z) = \sigma(z)(1 - \sigma(z))$, o erro na camada de saída é:

$$\begin{aligned}
\delta^{[3]} &= \frac{\partial J}{\partial z^{[3]}} \\
&= -\frac{\partial}{\partial z^{[3]}} \left[(1-y)\log(1-\hat{y}) + y\log\hat{y}\right] \\
&= -(1-y)\frac{\partial}{\partial z^{[3]}}\log(1-\sigma(z^{[3]})) - y\frac{\partial}{\partial z^{[3]}}\log(\sigma(z^{[3]})) \\
&= -(1-y)\frac{1}{1-\sigma(z^{[3]})}(-1)\sigma'(z^{[3]}) - y\frac{1}{g(z^{[3]})}g'(z^{[3]}) \\
&= (1-y)a^{[3]} - y(1-a^{[3]}) \\
&= a^{[3]} - y. \qquad (11)
\end{aligned}$$

Os demais gradientes são definidos como:

$$\begin{aligned}
\frac{\partial z^{[3]}}{\partial a^{[2]}} &= W^{[3]^T}, \\
\frac{\partial a^{[2]}}{\partial z^{[2]}} &= \sigma'(z^{[2]}), \\
\frac{\partial z^{[2]}}{\partial W^{[2]}} &= x^T.
\end{aligned}$$

Substituindo os termos, temos:

$$\frac{\partial J}{\partial W^{[2]}} = (a^{[3]} - y)W^{[3]^T}\sigma'(z^{[2]})x^T. \qquad (12)$$

Para verificar a consistência, considere as dimensões:

$$\underbrace{\frac{\partial J}{\partial W^{[2]}}}_{3\times3} = \underbrace{(a^{[3]} - y)}_{1\times1}\underbrace{W^{[3]^T}}_{3\times1} \circ \underbrace{\sigma'(z^{[2]})}_{3\times1}\underbrace{x^T}_{1\times3}. \qquad (13)$$

Reorganizando:

$$\frac{\partial J}{\partial W^{[2]}} = (W^{[3]^T} \circ \sigma'(z^{[2]}))(a^{[3]} - y)x^T, \qquad (14)$$

confirmando a consistência com o método de propagação de erros.

3 Técnicas de Atualização de Parâmetros

Há três maneiras principais de aplicar a regra da Equação 1 em um conjunto com múltiplas amostras:

1. **Atualização em lote:** considera o conjunto inteiro a cada atualização.
2. **Atualização estocástica:** atualiza os parâmetros após cada amostra.
3. **Atualização em mini-lotes:** atualiza os parâmetros com pequenos subconjuntos de tamanho B.

A diferença principal é que a descida do gradiente calcula o gradiente exato com todos os exemplos, enquanto a descida estocástica aproxima esse valor com uma única amostra, introduzindo ruído. Apesar do ruído, a abordagem estocástica é eficiente para grandes conjuntos de dados.

Por outro lado, a descida em mini-lotes equilibra as duas técnicas. Ela oferece gradientes mais estáveis que os da abordagem estocástica e requer menos recursos computacionais do que a descida em lote.

4 Regularização e Controle de Overfitting

Seja W o conjunto de parâmetros $W^{[\ell]}$ de cada camada da rede. Apenas esses pesos são regularizados na rede neural. A regularização L2 modifica a função de custo ao adicionar um termo:

$$\begin{aligned} J_{\text{L2}} &= J + \frac{\lambda}{2}\|W\|^2 \\ &= J + \frac{\lambda}{2}\sum_i W_i^2, \end{aligned}$$

onde J é a função de custo padrão, λ controla o peso da regularização (quanto maior λ, maior a regularização), e W contém todos os pesos do modelo.

A regra de atualização com regularização L2 é:

$$\begin{aligned} W &= W - \alpha\frac{\partial J}{\partial W} - \alpha\lambda W \\ W &= (1-\alpha\lambda)W - \alpha\frac{\partial J}{\partial W}, \end{aligned}$$

onde α é a taxa de aprendizado. Durante a descida do gradiente, minimizamos J enquanto adicionamos uma penalização proporcional a W. Essa penalização aumenta J, restringindo os pesos a valores menores. Como resultado, reduzimos o *overfitting*, limitando a capacidade do modelo de se ajustar excessivamente aos dados de treinamento.

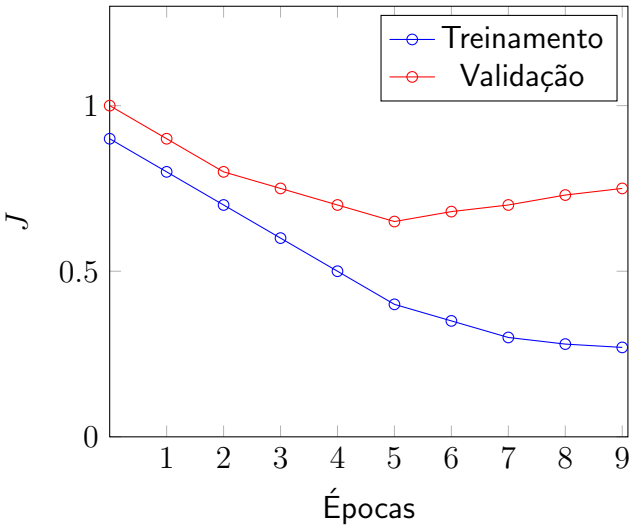

Figura 2: Função log-loss medida ao longo de 10 épocas.

Exercícios

1. Explique por que a inicialização dos parâmetros com valores iguais a zero inviabiliza o aprendizado da rede neural. Descreva também como isso impactaria o treinamento da rede.

2. Calcule $\frac{\partial J}{\partial W^{[1]}}$, $\frac{\partial J}{\partial b^{[1]}}$ e $\frac{\partial J}{\partial b^{[2]}}$ para a rede neural apresentada na Figura 1. Compare os resultados obtidos com os calculados pelo método de propagação de erros.

3. Para a função de custo utilizada na regressão linear, $J(\hat{y}, y) = \frac{1}{2}(\hat{y} - y)^2$, calcule $\frac{\partial J}{\partial z^{[3]}}$ e explique como essa derivada é usada no treinamento do modelo.

4. Durante o treinamento de uma rede neural, o valor da função de custo J é monitorado e registrado tanto para o conjunto de treinamento quanto para o conjunto de validação. A Figura 2 mostra os valores de J medidos ao longo de 10 épocas.

   (a) Identifique no gráfico o ponto em que o modelo começa a apresentar overfitting. Justifique sua resposta.

   (b) Descreva como a regularização L2 poderia influenciar o comportamento das curvas de treinamento e validação, considerando o impacto da penalização nos pesos.