

**Instituto Tecnológico de Aeronáutica - ITA**  
**Inteligência Artificial para Robótica Móvel - CT213**  
**Aluno:**

**Relatório do Laboratório 6 - Redes Neurais**

## **1 Breve Explicação em Alto Nível da Implementação**

O objetivo deste laboratório é implementar uma rede neural com tres camadas (1 camada escondida) para realizar a segmentação de cores em imagens. A rede é treinada para classificar cada pixel da imagem em uma das seguintes classes: verde, branco ou outras cores (representadas pela cor preta).

Foi utilizada a função de ativação sigmoide em todas as camadas, o que permite interpretar as saídas como probabilidades. Como se trata de um problema de classificação multiclasse, a função de custo escolhida foi a função de perda logística multiclasse:

$$L(y^{(i)}, \hat{y}^{(i)}) = - \sum_{c=1}^C [(1 - y_c^{(i)}) \log(1 - \hat{y}_c^{(i)}) + y_c^{(i)} \log(\hat{y}_c^{(i)})] \quad (1)$$

A função de custo total da rede é então definida como a média da função de perda individual para cada exemplo de treinamento:

$$J(\theta) = \frac{1}{m} \sum_{i=1}^m L(y^{(i)}, \hat{y}^{(i)}) \quad (2)$$

O treinamento da rede é realizado via back propagation, que calcula os gradientes da função de custo em relação aos parâmetros da rede (pesos e vieses), permitindo a atualização desses parâmetros por meio de algoritmos de otimização, como o gradiente descendente. Os gradientes para cada camada são dados por:

$$\frac{\partial L}{\partial w_{ck}^{[2]}} = \delta_c^{[2]} a_k^{[1]} \quad (3)$$

$$\frac{\partial L}{\partial b_c^{[2]}} = \delta_c^{[2]} \quad (4)$$

$$\frac{\partial L}{\partial w_{kj}^{[1]}} = \delta_k^{[1]} a_j^{[0]} \quad (5)$$

$$\frac{\partial L}{\partial w_{kj}^{[1]}} = \delta_k^{[1]} a_j^{[0]} \quad (6)$$

$$\frac{\partial L}{\partial b_k^{[1]}} = \delta_k^{[1]} \quad (7)$$

onde os termos de erro são definidos como:

$$\delta_c^{[2]} = (\hat{y}_c - y_c) \quad (8)$$

$$\delta_k^{[1]} = \sum_{c=1}^C w_{ck}^{[2]} \delta_c^{[2]} \sigma'(z_k^{[1]}) \quad (9)$$

A etapa de forward propagation foi implementada de forma vetorizada para garantir maior eficiência computacional. O processo segue as equações abaixo:

$$\mathbf{a}^{[0]} = \mathbf{x} \quad (10)$$

De  $l = 1$  até  $L$

$$\mathbf{z}^{[l]} = \mathbf{W}^{[l]} \mathbf{a}^{[l-1]} + \mathbf{b}^{[l]} \quad (11)$$

$$\mathbf{a}^{[l]} = g^{[l]}(\mathbf{z}^{[l]}) \quad (12)$$

$$\hat{\mathbf{y}} = \mathbf{a}^{[L]} \quad (13)$$

## 2 Figuras Comprovando Funcionamento do Código

### 2.1 Função de Classificação *sum\_gt\_zeros*

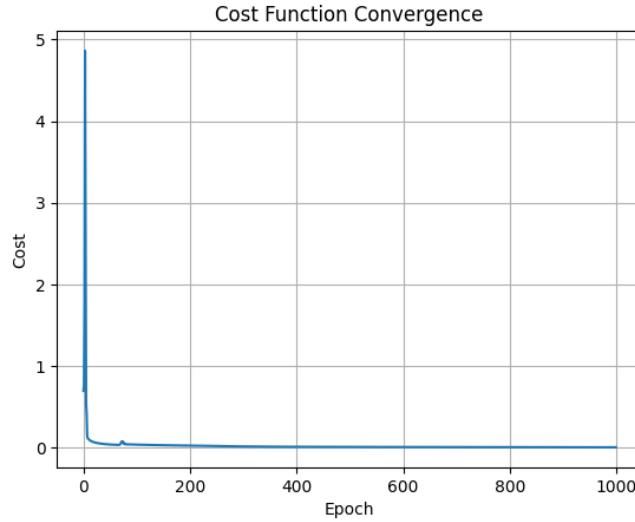


Figura 1: gráfico de convergência da função custo para o *sum\_gt\_zeros*.

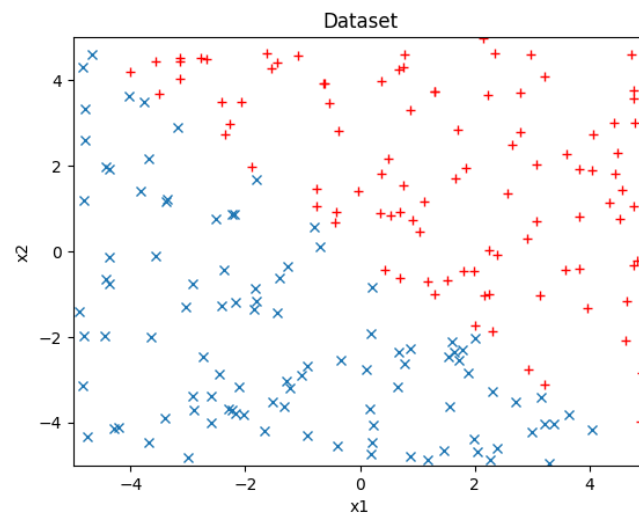


Figura 2: gráfico do dataset para o *sum\_gt\_zeros*.

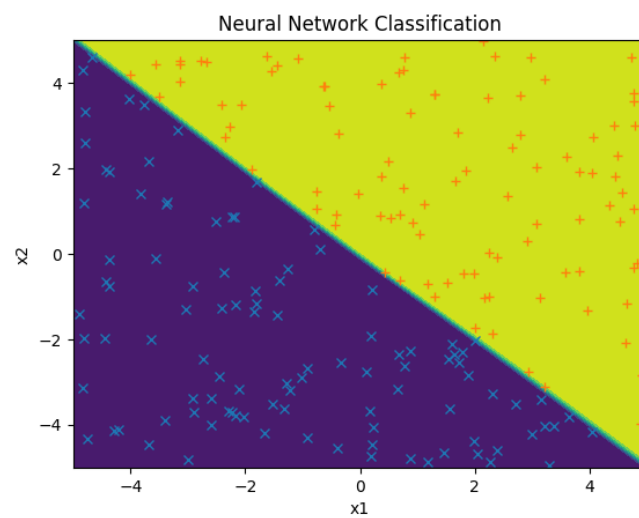


Figura 3: gráfico de classificação da rede neural *fitness* para o *sum\_gt\_zeros*.

## 2.2 Função de Classificação *XOR*

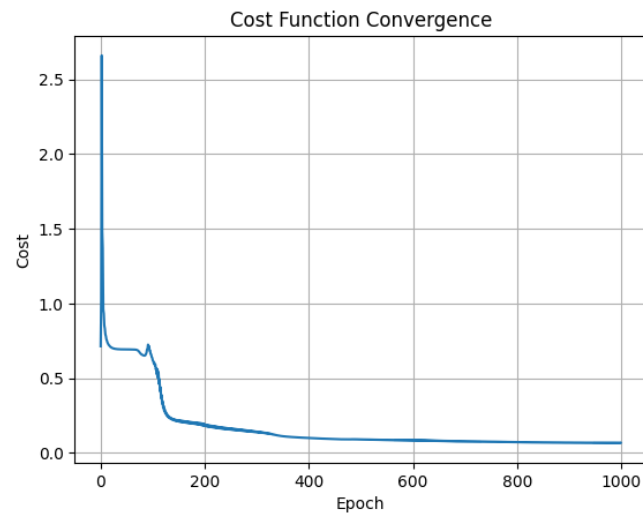


Figura 4: gráfico de convergência da função custo para o *XOR*.

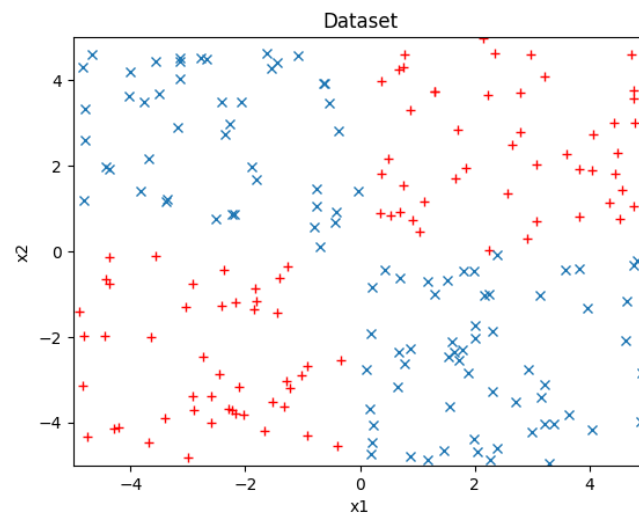


Figura 5: gráfico do dataset para o *XOR*.

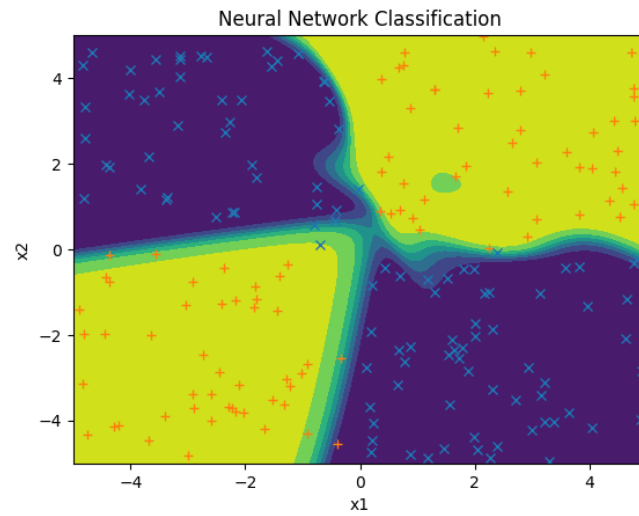


Figura 6: gráfico de classificação da rede neural *fitness* para o *XOR*.

### 2.3 Segmentação de Cores

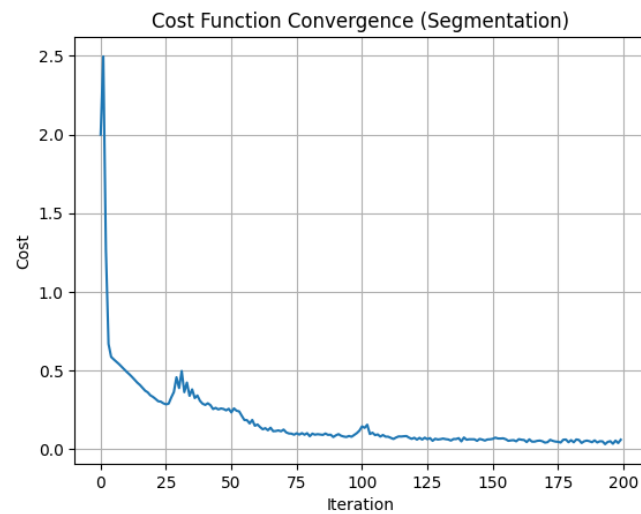


Figura 7: gráfico de convergência da função custo para a segmentação por cores.

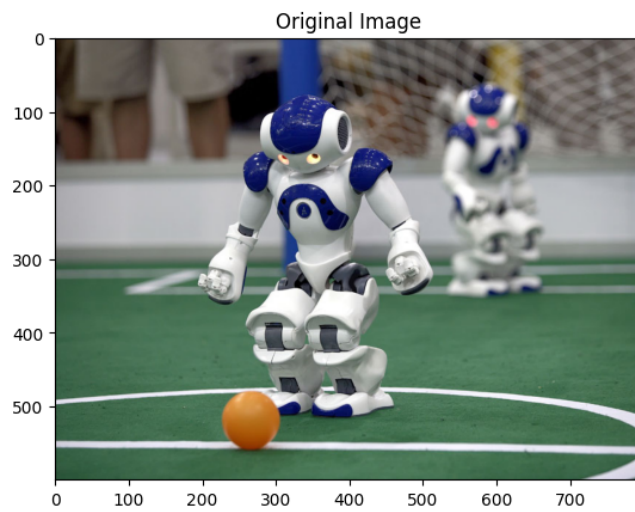


Figura 8: Imagem do robô NAO.

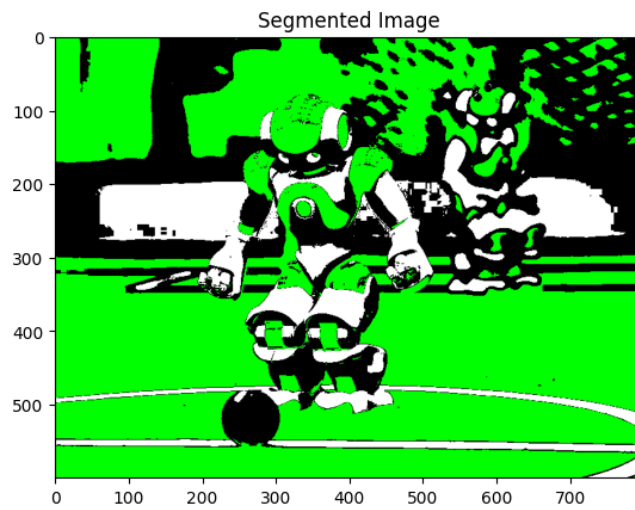


Figura 9: Imagem segmentada por cores do robô NAO.

### 3 Discussões

A rede neural apresentou boa convergência nos testes realizados, como nos problemas *sum\_gt\_zeros* e *XOR*, demonstrando capacidade de modelar relações não lineares e separações complexas no espaço de entrada. Isso indica que os mecanismos de forward propagation e back propagation foram corretamente implementados e que a arquitetura, mesmo simples, é suficiente para tarefas básicas de classificação.

Na segmentação de cores do robô NAO, a rede obteve resultados satisfatórios ao distinguir verde, branco e outras cores. A principal limitação observada foi na diferenciação entre o azul do corpo do robô e a classe verde, o que é esperado devido à proximidade espectral e à capacidade restrita do modelo.