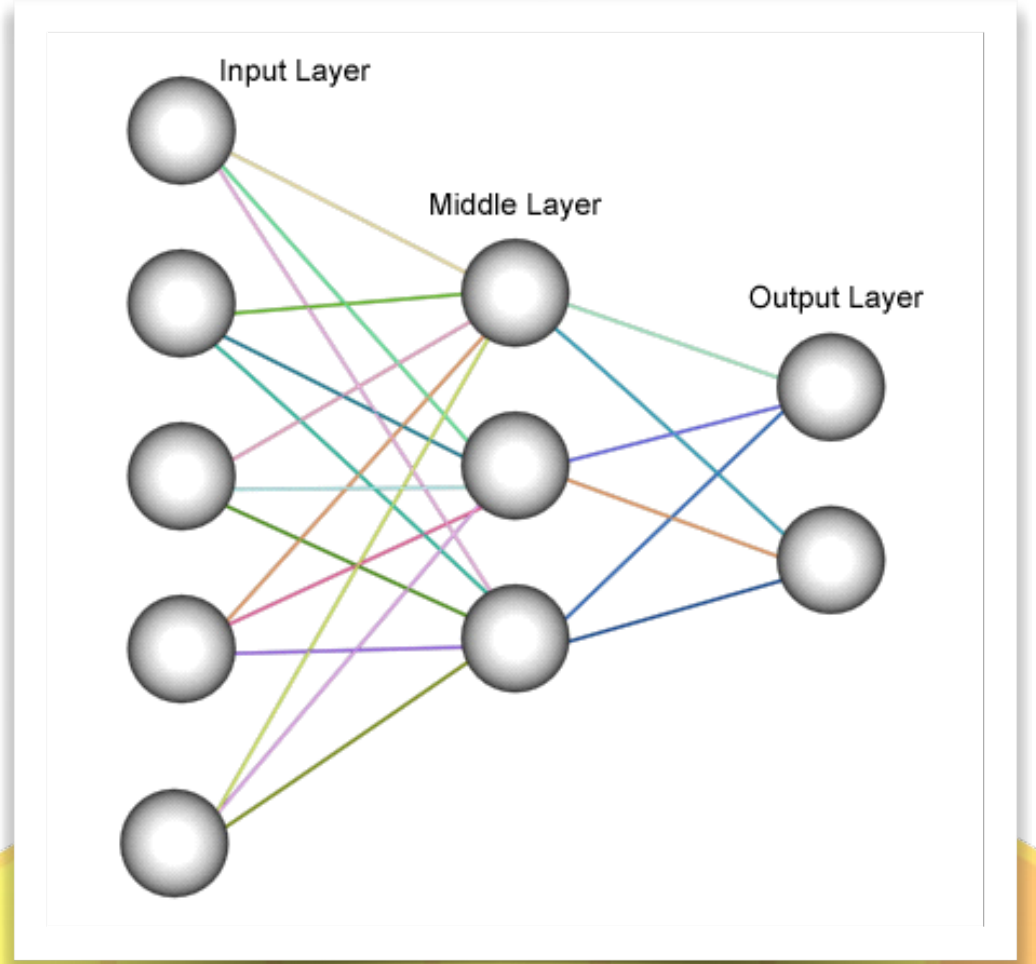
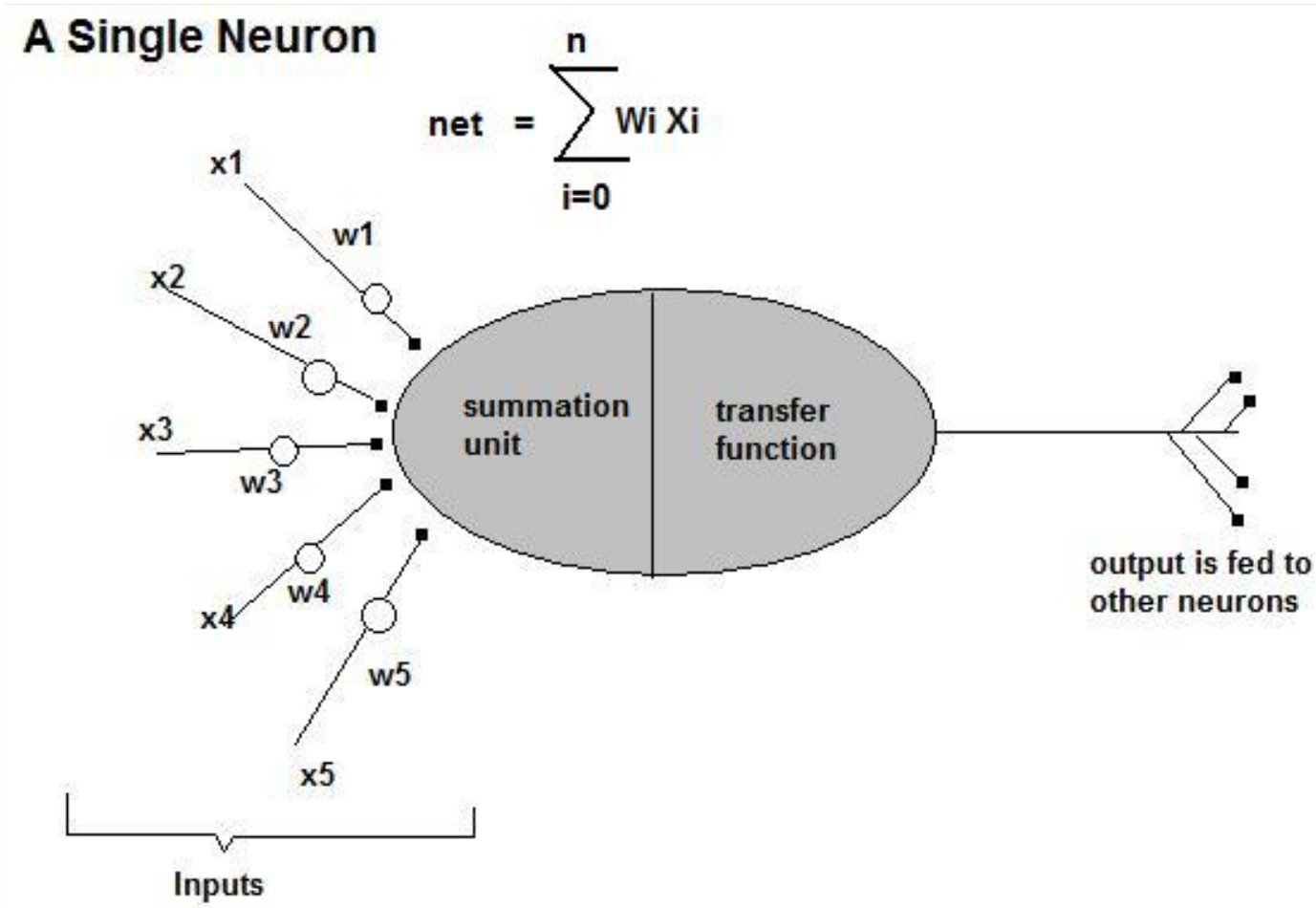
Algoritmo Backpropagation

Profª Carine G. Webber

Rede Neural de 3 camadas



Exemplo de um neurônio

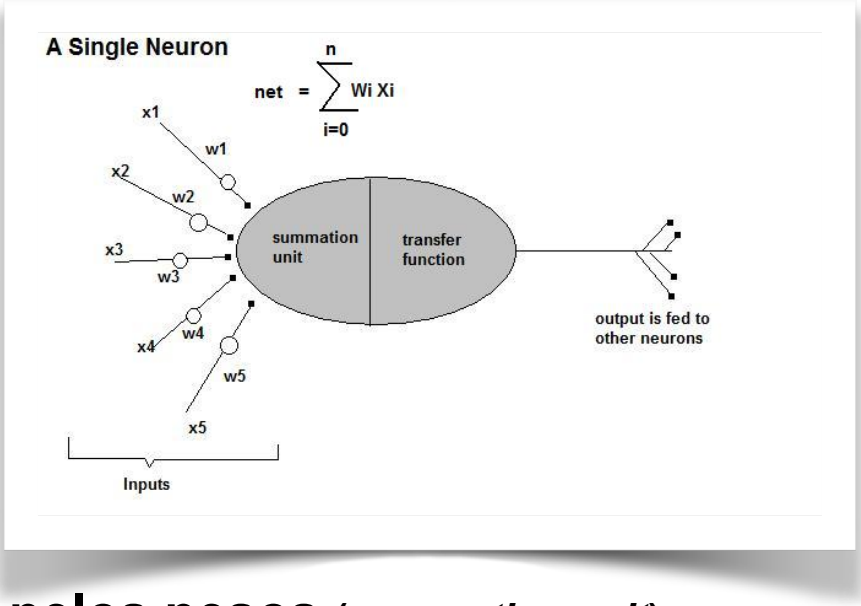


Neurônio

O processamento de um neurônio passa por duas fases:

1. FASE 1 : Somatório das entradas multiplicadas pelos pesos (*summation unit*)
2. FASE 2: Geração da saída do neurônio (*transfer function*)

Fase 1



1) Somatório das entradas multiplicadas pelos pesos (*summation unit*)

Suponha um neurônio contendo as entradas (x1, x2, x3, x4, x5) e os pesos (w1, w2, w3, w4, w5).

As entradas do neurônio e os pesos de cada conexão devem estar armazenadas em vetores. Os pesos devem ser inicializados aleatoriamente.

O somatório é calculado como segue:

```
somatorio=0
for i=0 to neuronio.entradas.count-1
    somatorio=somatorio + x(i) * w(i)
```

Fase 2

2) Geração da saída do neurônio (*transfer function*)

A função de transferência (ou função de ativação) é uma função simples que usa o valor do somatório para gerar a saída do neurônio. Esta saída é então propagada para os neurônios da próxima camada.

Tem-se vários tipos de função de transferência: sigmoidal, tanh, RELU.

Função Sigmoidal

Gera um valor entre 0 e 1.

```
saida = 1 / (1 + Exp(-somatorio))
```

Treinamento

Treinamento é o processo a partir do qual os pesos das conexões entre neurônios são ajustados a fim de que a rede produza o resultado (saída) esperado para todas as entradas fornecidas.

Passo 1– As entradas

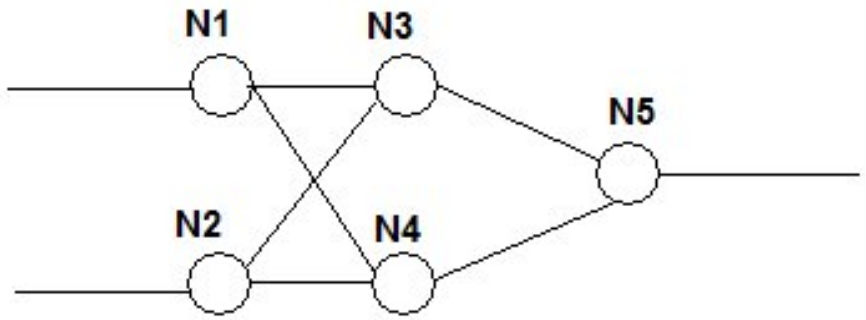
Inicialmente as entradas devem ser fornecidas para a Rede Neural. Na primeira camada, a saída dos neurônio é a própria entrada.

Treinamento

Passo 2 – Saída da rede

Calcule a saída da rede.

Veja um exemplo. Suponha seguinte a rede neural:



somatorio de N3 = (N1.Saida * Peso da conexao de N1 para N3) + (N2.Saida * Peso da conexao de N2 para N3)

somatorio de N4 = (N1.Saida * Peso da conexao de N1 para N4) + (N2.Saida * Peso da conexao de N2 para N4)

Para gerar a saída dos neurônios utiliza-se a função *sigmoidal*.

Treinamento

Passo 3 – Cálculo do Erro

Erro ou Delta é a diferença entre a saída esperada e a saída obtida. Deve ser calculado para os neurônios das camadas intermediárias e de saída.

Primeiro calcula-se o erro da camada de saída. Este valor é utilizado para calcular o erro das camadas intermediárias (retro-propagação do erro).

A equação geral do cálculo do erro delta de um neurônio é:

Neuronio.Erro = Neuronio.Saida * (1 - Neuronio.Saida) * FatorErro

Treinamento

Passo 3 – Cálculo do Erro

Para um neurônio da camada de saída, calcula-se o fator de erro da seguinte maneira:

FatorErro de neuronio na camada de saída=SaídaEsperada–SaídaAtualNeuronio

Treinamento

Para um neurônio da camada intermediária, o fator de erro é calculado de forma diferente.

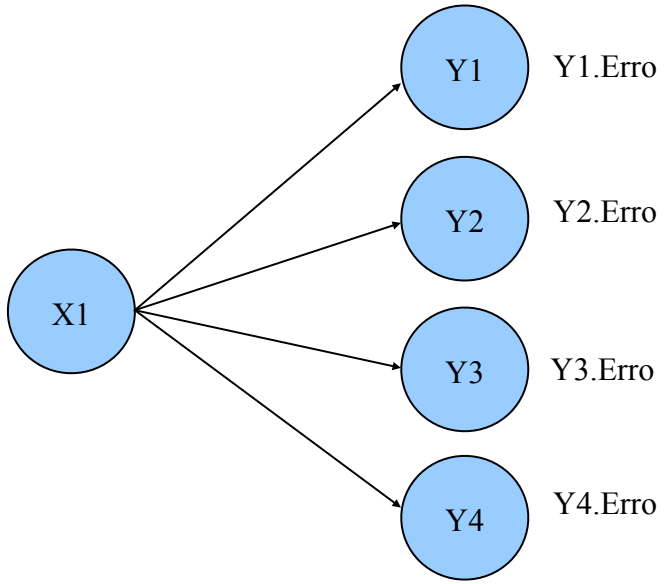
Para ilustrar, considere um neurônio X1 que se encontra na camada escondida. X1 está conectado com os neurônios Y1, Y2, Y3 e Y4 (da camada de saída).

FatorErro de X1 = (Y1.Erro * Peso da conexão de X1 para Y1) +

(Y2.Erro * Peso da conexão de X1 para Y2) +

(Y3. Erro * Peso da conexão de X1 para Y3) +

(Y4. Erro * Peso da conexão de X1 para Y4)



Agora o erro de X1 pode ser calculado como:

X1.Erro = X1.Saida * (1 - X1.Saida) * FatorErro de X1

Treinamento

Passo 4 – Ajuste dos Pesos

Após o cálculo dos erros de todos os neurônios de todas das camadas, deve-se corrigir os pesos a fim de que se possa obter saídas mais próximas das esperadas.

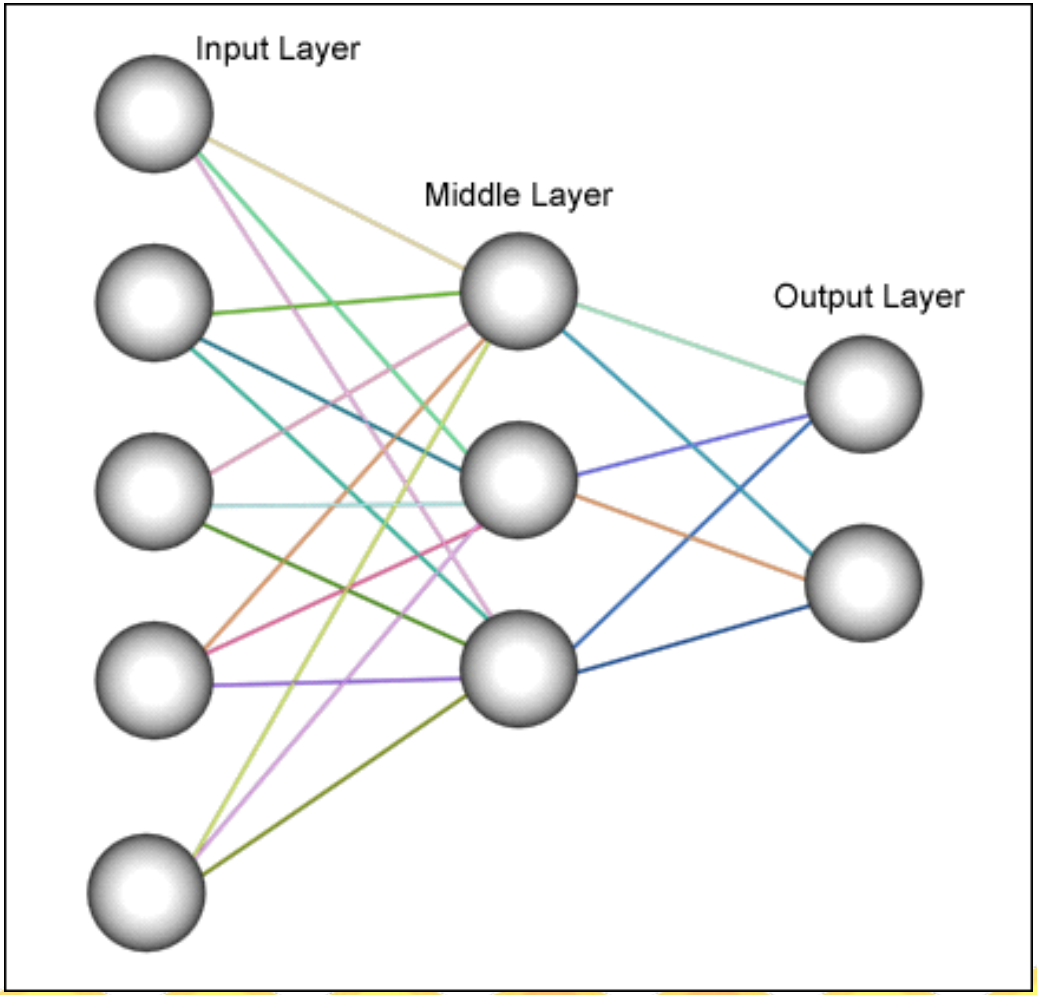
Novo_peso = Peso_anterior + Taxa de aprendizagem * Saída do neurônio anterior * Erro

Treinamento

Encerra-se assim um episódio de Treinamento, ou uma iteração.

Sumário dos Passos:

Passo 1 - propaga entradas
Passo 2 – calcula saídas
Passo 3 – calcula erros
Passo 4 – corrige pesos



Teste

Na etapa de teste, o conjunto de dados de teste deve ser usado para avaliar o desempenho do classificador gerado.

O resultado da classificação da rede neural deve ser comparado com o resultado esperado.

Deve ser gerada uma matriz de confusão e as métricas: acurácia, precisão, recall e f1-score.

Desafio

Problema de Classificação

https://archive-beta.ics.uci.edu/ml/datasets/image+segmentation

Considere o conjunto de dados descrito no link acima.
Você deve construir e treinar uma Rede Neural para reconhecer classes de imagens segmentadas
E representadas pelas suas características de cores (RGB), saturação e intensidade.

Classes (7): {brickface,sky,foliage,cement,window,path,grass}

Projeto de Rede Neural

Por onde começar?

1. Entenda o problema;
2. Defina as estruturas de dados necessárias para armazenar entradas, neurônios, pesos, saídas;
3. Implemente os principais métodos descritos nos passos 1 a 4;
4. Inicie o treinamento da Rede neural e observe o seu comportamento;
5. Uma vez que a rede estiver reconhecendo as classes, mesmo que com baixa acurácia, trabalhe nos seus parâmetros para melhorar o resultado;
6. Se a rede apresentar uma performance baixa, modifique os valores das constantes: taxa de aprendizagem e momentum, reduzindo-a para que os pesos sejam sutilmente alterados a cada correção.
7. A cada execução você poderá observar pequenas mudanças no comportamento da rede, uma vez que os pesos são inicializados aleatoriamente. Entretanto, as execuções sempre devem convergir para valores aproximados (com mais ou menos iterações sendo necessárias).