UNIVERSIDADE FEDERAL DE UBERLÂNDIA - CAMPUS SANTA MÔNICA
CIENCIA DA COMPUTACAO
INTELIGÊNCIA ARTIFICIAL
CARLOS CÉSAR MORAIS - 11611BCC037

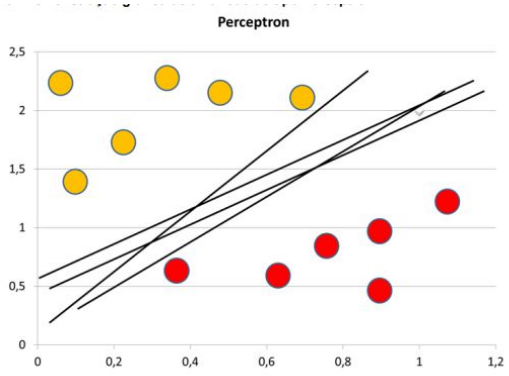
TRABALHO FINAL - RELATÓRIO

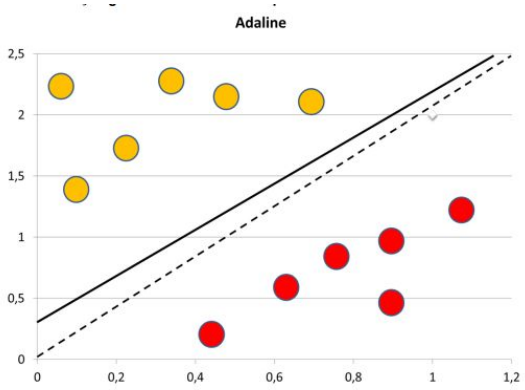
UBERLÂNDIA, 2019

Como motivação das apresentações feitas em sala de aula, e todo conhecimento já adquirido no curso, no caso, de inteligência computacional, optei por implementar um classificador de letra com detecção de ruído.

A ideia é utilizar uma rede de chamada Adaline, que se parece muito ao Perceptron. Porém, sua diferença está na forma de treinamento da rede. O Adaline utiliza como treinamento a regra delta, a fim de minimizar o erro além de utilizar a correção no potencial de ação e não na saída propriamente dita.

Abaixo temos ilustrando a diferença entre as duas redes:





Pode-se perceber que como a Adaline procura um ponto ótimo, qualquer solução para a separação entre as duas classes será próxima do hiperplano ótimo de separação.

Para treinamento da rede Adaline, foi utilizado um pseudo-código encontrado no artigo de pesquisa retratado no fim deste relatório. Segue o pseudo-código logo ao lado.

Sendo

x_n o vetor de entrada com elementos distribuídos por $[-1, x_1, x_2, \ldots x_n]$;
w_n o vetor de peso com elementos distribuídos por $[\theta_n, w_1, w_2, \ldots. w_n]$;
θ_n o limiar de ativação;
d_n a resposta desejada;
$g(u)$ é a função de ativação da saída do neurônio;
y_n a resposta real após a função de ativação e;
η a taxa de aprendizagem.
E a função para o erro quadrático médio de p amostras dado por

$\frac{1}{p}\sum_{k=1}^{p}(d_n - u_n)^2$; e

ε a precisão do erro quadrático médio.

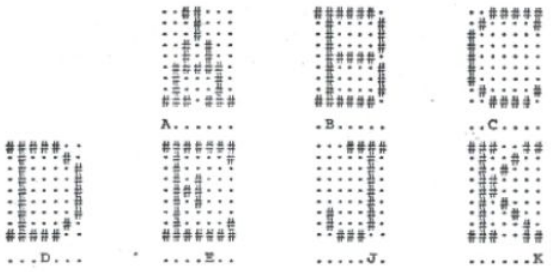
Início do treinamento
1- Obter o conjunto de amostras de treinamento x_n e sua respectiva saída d_n para cada amostra.
2- Inicializar o vetor w_n com valores aleatórios pequenos, a taxa de aprendizagem e o contador de épocas.
3-Repetir até o erro ser menor que a precisão ou estoure a quantidade de épocas:
3.1- Atualizar o erro quadrático médio $E_{an} \leftarrow E_w$
3.2- Para todas as amostras de treinamento:
3.2.1- Calcular o potencial de ativação por $u = w \cdot x$
3.2.2- Atualizar os pesos por $w \leftarrow w + \eta\,(d_n - u_n).\, x_n$
3.3- Incrementar contador de épocas: época ← época +1;
3.4- Atualizar o erro quadrático médio $E_{at} \leftarrow E_w$
3.5- Calcular precisão até $|E_{at} - E_{an}| \leq \varepsilon$;
Fim do treinamento.

Depois de apresentado os conceitos básicos usado no desenvolvimento deste programa, pode-se definir que este programa nada mais é do que um classificador de letras com detecção de ruído.

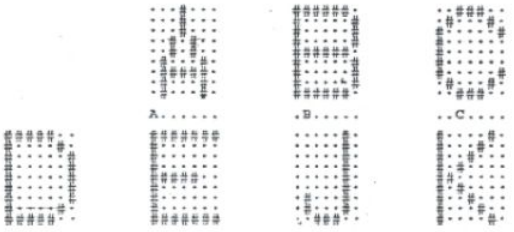
Funciona da seguinte forma, é apresentado um conjunto inicial de letras do alfabeto em forma de matriz. Com isso, é feito um treinamento de uma rede neural, no caso a utilizada nesse projeto foi a Adeline, e com isso aplicar a regra Delta, como já dito logo acima, para que seja feito a análise da entrada e poder fornecer a quantidade de ruído que a letra possui e de fato identificar a letra.

Neste trabalho a rede deverá reconhecer as letras do alfabeto A, B, C, D, E, J e K. Também é estabelecido três formatos de fontes diferentes da letra.

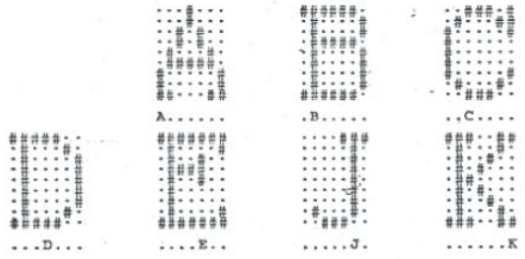
- Fonte 1



- Fonte 2



- Fonte 3



Considerando que cada entrada é uma matriz de 7X9, ou melhor, um vetor de 63 posições; então a rede terá 63 neurônios de entrada, mais o bias. E ainda, a rede terá 7 neurônios de saída: cada neurônio faz o reconhecimento de uma letra. Para o treinamento da rede, utilizou - se a regra Adaline, 63 neurônios de entrada (mais o Bias), erro mínimo de 0,001 e desvio Padrão = 0,02. Para testar a rede o usuário deve aplicar o carácter “#” onde queira que seja preenchido e aplicar “0” onde queira aplicar ruído.

O PEAS deste programa pode ser estabelecido da seguinte forma:

Agente : Classificador de letras
Performance: Conseguir classificar a letra desenhada
Sensores: Entrada pelo teclado para aplicação de ruídos, treinamento da rede
Atuadores: Usuário, computador, tela do computador, teclado, mouse.
Ambiente: Rede neural Adaline

- **Parcialmente observável**: devido ao ruído e a sensores imprecisos;
- **Multi-agente competitivo:** o usuário tenta confundir o agente aplicando ruídos, assim, sendo capaz de competir com olho humano;
- **Estocástico:** ao fato de toda hora pode haver um ruído diferente;
- **Episódico:** o agente recebe uma percepção e em seguida executa uma única ação;
- **Dinâmico:** o ambiente pode se alterar;
- **Discreto:** o ambiente possui um limite;
- **Conhecido:** é possível retornar uma resposta, no caso, qual porcentagem de ruído em uma letra;

Referências

FAUSETT,L. Fundamentals of neural networks, Architectures, algorithms and applications Prentice Hall, Englewood Cliffs, p. 100, 1994

MATOS, M. & OLENIK, S. IMPLEMENTAÇÃO DE REDES NEURAIS PERCEPTRON E ADALINE EM AMBIENTE LABVIEW, 2018