



Redes Neurais Artificiais: Raspberry Pi e Python

RF

27 de maio de 2016

O Raspberry Pi

Python

O Campo da Inteligência Computacional

Redes Neurais Artificiais

- Introdução

- O Neurônio

- Arquiteturas de Rede

- Aprendizado de máquina

Biblioteca PyBrain

Desenvolvimento de Sistemas Utilizando RNA's

Hands On

Bibliografia Recomendada

Apresentação:

<http://tinyurl.com/m8js79h>

Exemplo 1:

<http://pastebin.com/88ZsJnmA>

Exemplo 2:

<http://pastebin.com/UaG3nKWD>

Dados do Exemplo 2:

<http://tinyurl.com/oe9947b>

Fonte dos dados do Exemplo 2:

<https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Original%29>

Section 1

O Raspberry Pi

A placa

- ▶ Baixo custo e robusta: projetada para crianças
- ▶ Movimento Maker e computação física
- ▶ ARM11, 700MHz, 256 MB RAM, GPU 250 MHz
- ▶ Suporta Debian GNU/Linux compilado para ARM v6

Periféricos

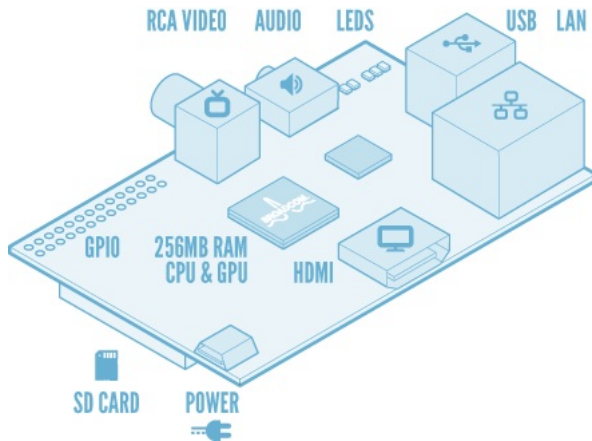


Figura: Fonte: raspyfi.com

Section 2

Python

Linguagem Python

- ▶ Linguagem de programação de alto nível, interpretada e orientada a objetos
- ▶ `import this`

Tutorial de 2 segundos

```
print ‘‘Alô Mundo!!!’’
```

Section 3

O Campo da Inteligência Computacional

O Campo da Inteligência Computacional

Algumas ferramentas que têm amadurecido nos últimos 20 anos tem sido utilizadas com sucesso em problemas até então, intratáveis. A maioria delas é bioinspirada; mimetiza processos que já são conhecidos dos campos da biologia. Podem ser classificadas, basicamente, em 3 áreas:

- ▶ Lógica Fuzzy
- ▶ Computação Evolucionária
- ▶ Redes Neurais

Ao contrário da Inteligência Artificial clássica, possui pouca utilização da abordagem simbólica, ênfase em sistemas com computação simples e aprendizado de máquina e não na simulação de agentes inteligentes.

Section 4

Redes Neurais Artificiais

Conforme Haykin, O cérebro pode ser definido como:

Um sistema de processamento de informação altamente complexo, não-linear e paralelo

Possui as seguintes características:

- ▶ Aprendizagem
- ▶ Plasticidade
- ▶ Acumula experiência
- ▶ É tolerante à falhas
- ▶ Processador universal de informação

O Neurônio Artificial

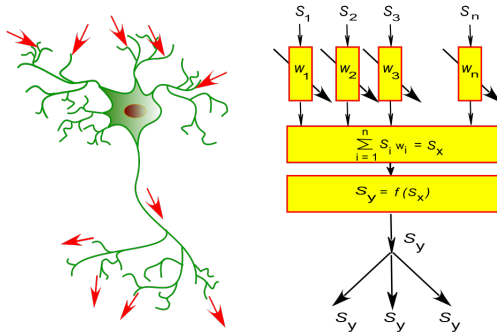


Figura: Fonte: Wikimedia Commons

O Neurônio Artificial

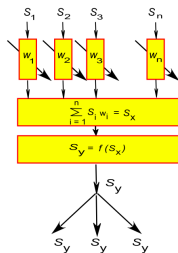


Figura: Fonte: Wikimedia Commons

- ▶ Entradas
- ▶ Pesos
- ▶ Somatório
- ▶ Campo Local induzido
- ▶ Função de Ativação

Funções de Ativação

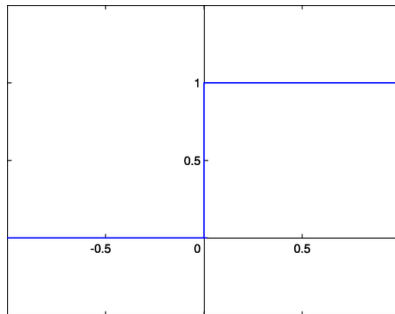


Figura: Função de Limiar. Fonte: Wikimedia Commons

Funções de Ativação

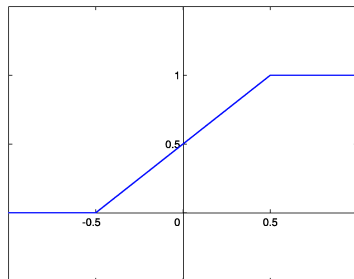


Figura: Função Linear por partes. Fonte: Wikimedia Commons

Funções de Ativação

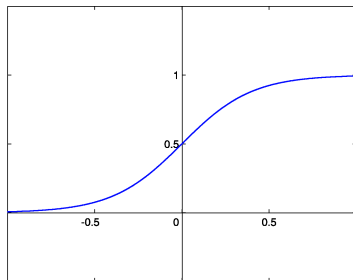


Figura: Função Sigmóide. Fonte: Wikimedia Commons

$$\phi(v) = \frac{1}{1 + e^{-av}}$$

Funções de Ativação

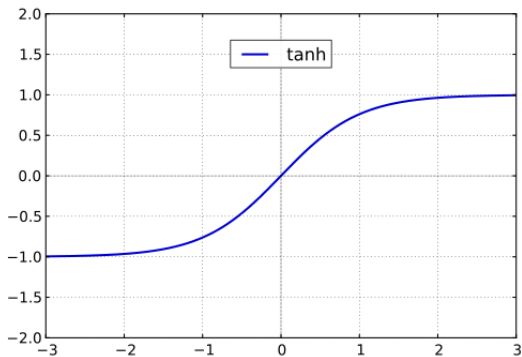


Figura: Função tangente sigmóide hiperbólico. Fonte: Wikimedia Commons

$$\phi(v) = \frac{2}{1 + e^{-2v}} - 1$$

A Rede Neural

Ainda com Haykin, ele afirma que:

Uma Rede Neural é um processador maciçamente paralelamente distribuído constituído de unidades de processamento simples, que tem a propensão natural para armazenar conhecimento experimental e torná-lo disponível para o uso. Ela se assemelha ao cérebro em dois aspectos: O conhecimento é adquirido pela rede a partir de seu ambiente através de um processo de aprendizagem e forças de conexão entre neurônios, conhecidas como pesos sinápticos, são utilizadas para armazenar conhecimento adquirido.

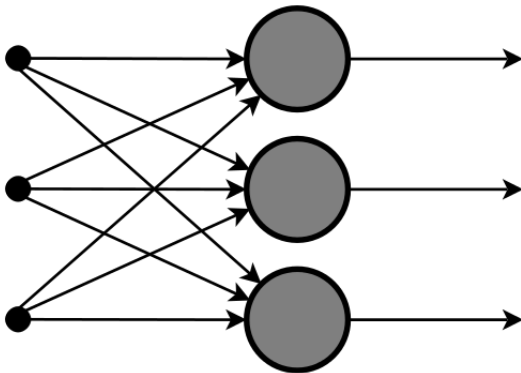


Figura: Redes com uma camada adiante. Fonte: Wikimedia Commons

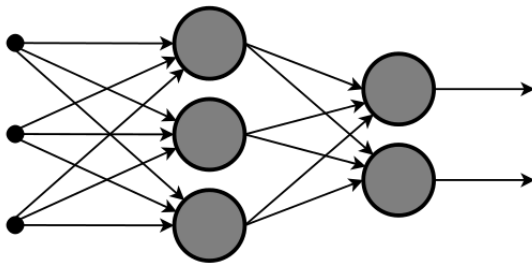


Figura: Redes com múltiplas camadas adiante. Fonte: Wikimedia Commons

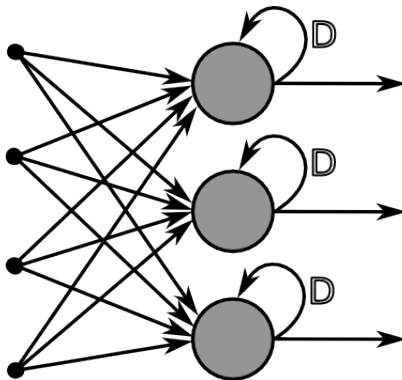


Figura: Redes recorrentes. Fonte: Wikimedia Commons

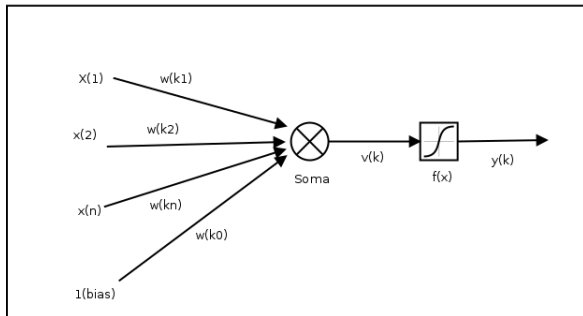
Definição

Pode-se dizer que um programa de computador aprende de alguma experiência E com relação a alguma tarefa T e alguma medida de desempenho P , se seu desempenho em T , tal como medido por P , melhora com a experiência E .

Tom Mitchell

Modos: supervisionado e não-supervisionado

Notação



- ▶ $x_j(n)$: Entrada j do neurônio k
- ▶ $w_{kj}(n)$: peso da entrada j aplicado ao neurônio k no tempo n
- ▶ d_k : saída desejada pelo professor no tempo n .
- ▶ b_k : *bias* do neurônio k .

Aprendizagem por Correção de erro

A comparação entre a saída y_k e a saída apresentada pelo professor d_k gera o erro:

$$e_k = d_k - y_k$$

Define-se uma função de custo:

$$E(n) = \frac{1}{2} e_k^2(n)$$

E realiza-se a minimização de $E(n)$ alterando os pesos da seguinte forma:

$$\begin{aligned} w_{kj}(n+1) &= w_{kj}(n) + \Delta w_{kj}(n+1) \\ \Delta w_{kj}(n) &= \eta e_k(n) x_j(n) \end{aligned}$$

Section 5

Biblioteca PyBrain

<http://pybrain.org/docs/>



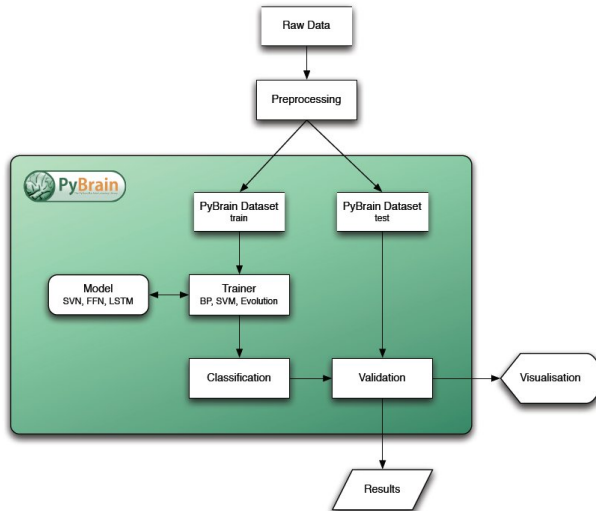



Figura: <http://pybrain.org/docs/tutorial/intro.html>

Section 6

Desenvolvimento de Sistemas Utilizando RNA's

Etapas de desenvolvimento

- ▶ Definição do problema
- ▶ Coleta de dados
- ▶ Pré-tratamento. Normalmente, $\mu = 0$ e $\sigma = 1$.
- ▶ Configuração da rede
- ▶ Treinamento
- ▶ Teste
- ▶ Integração




UCI
Machine Learning Repository
Center for Machine Learning and Intelligent Systems

Iris Data Set

Download: [Data Folder](#), [Data Set Description](#)

Abstract: Famous database; from Fisher, 1936



Data Set Characteristics:	Multivariate	Number of Instances:	150	Area:	Life
Attribute Characteristics:	Real	Number of Attributes:	4	Date Donated	1988-07-01
Associated Tasks:	Classification	Missing Values?	No	Number of Web Hits:	708112

Figura: Repositório de dados

`http://archive.ics.uci.edu/ml/`

Separação dos conjuntos de dados



Figura: Fonte: Wikimedia Commons

Separação dos conjuntos de dados



Figura: Fonte: Wikimedia Commons

Separação dos conjuntos de dados



Figura: Fonte: Wikimedia Commons

Separação dos conjuntos de dados



Figura: Fonte: Wikimedia Commons

Separação dos conjuntos de dados



Figura: Fonte: Wikimedia Commons

Separação dos conjuntos de dados

Não se pode validar ou testar a rede com os dados utilizados para aprendizado. Separamos os dados em 3 conjuntos:

- ▶ Treinamento - Utilizado para realizar o aprendizado através do algoritmo escolhido
- ▶ Validação - Utilizado para verificar a eficácia da rede e capacidade de generalização.
- ▶ Teste - Utilizado para quaisquer testes que o desenvolvedor deseje.

Section 7

Hands On

Referências I

-  John Robert Anderson, Ryszard Stanisław Michalski, Jaime Guillermo Carbonell, and Tom Michael Mitchell, *Machine learning: An artificial intelligence approach*, vol. 2, Morgan Kaufmann, 1986.
-  A de P Braga, ACPLF Carvalho, and Teresa Bernarda Ludermir, *Redes neurais artificiais: teoria e aplicações*, Livros Técnicos e Científicos, 2000.
-  Katti Faceli, *Inteligência artificial: uma abordagem de aprendizado de máquina*, Grupo Gen-LTC, 2011.
-  SIMON Haykin, *Redes neurais: princípios e aplicações*, Bookman, Porto Alegre, Brazil (2000).
-  Isaías Lima, Carlos Pinheiro, and Flávio Santos Oliveira, *Inteligência artificial*, vol. 1, Elsevier Brasil, 2004.
-  Andrew Ng, *Machine Learning*, MOOC Lecture, 2015.

Referências II