



# Redes Neurais Artificiais

---



# Objetivo

---

- Apresentar os modelos tradicionais de redes neurais artificiais, seus métodos de aprendizagem e exemplos de aplicações.



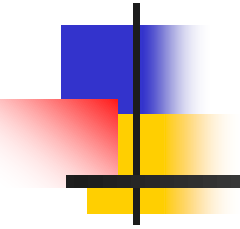
# Conteúdo

---

- Histórico e aplicações;
- Modelo Perceptron e o problema da separabilidade linear;
- Perceptron de multi-camadas MLP (treinamento por retropropagação);
- Ferramenta NNTOOL (Matlab);
- Aplicação de MLP em OCR

# Introdução às Redes Neurais Artificiais

---





# Motivação

---

- Os trabalhos em redes neurais artificiais (RNA) têm sido motivados pelo reconhecimento que o cérebro humano realiza “tarefas computacionais” complexas de maneira inteiramente diferente dos computadores digitais convencionais.
- O cérebro é um computador paralelo, altamente complexo e não linear, e com capacidade de aprendizagem.

# O que é uma Rede Neural Artificial



---

- Uma RNA é um sistema paralelo distribuído, inspirado no sistema nervoso de seres vivos, composto por unidades de processamento simples (*neurônios*), os quais armazenam conhecimento experimental e tornam o mesmo disponível para uso. As unidades são dispostas em uma ou mais camadas e interligadas por conexões associadas a pesos.
- Uma RNA se parece com o cérebro em dois aspectos:
  - O conhecimento adquirido pela rede é obtido do ambiente em que a mesma está inserida, através de um procedimento de aprendizagem.
  - Os pesos das conexões, são usados para o armazenamento do conhecimento adquirido.



# Propriedades e Capacidades

---

- Adaptação por experiência;
- Generalização;
- Mapeamento entrada-saída;
- Aprendizagem;
- Organização de dados (**determinação de agrupamentos**);
- Tolerância a falhas e armazenamento distribuído;
- Possibilidade de implementação em hardware e software;
- Inspiração biológica.



# Breve Histórico das RNA

---

- 1943 - Warren McCulloch e Walter Pitts apresentam uma teoria geral baseada em teorias do funcionamento biológico conhecidas na época. Baseava-se em elementos de decisão, chamados "neurônios". Cada elemento poderia obter um valor de saída -1 ou 1 para sua ativação ou não. O modelo não apresentava uma regra de aprendizagem para determinação dos "pesos" de interligação entre os "neurônios".
- 1949 - Donald O. Hebb define um método de atualizar os "pesos" na aprendizagem. Apresenta contribuições para a teoria de redes neurais, afirmando que a informação obtida pela rede neural é armazenada nos pesos. "Quanto mais uma ligação entre neurônios é utilizada, mais forte é a intensidade dessa ligação".
- 1959 - Frank Rosenblatt e seus colaboradores pesquisam um tipo específico de rede neural chamada "Perceptron", que apresenta duas camadas separadas de conjuntos de "neurônios": entrada e saída. O grupo introduz um algoritmo iterativo para determinação dos "pesos" que interligam os "neurônios".





# Breve Histórico das RNA (cont.)

---

- 1960 - Widrow e Hoff apresentam um novo modelo chamado ADALINE "Adaptive Linear Element", introduzindo um algoritmo de aprendizagem mais rápido do que o apresentado pelo grupo de Rosenblatt, baseado no método dos mínimos quadrados. Eles mostraram que a forma com que ajustavam os "pesos", minimizava o erro para todos os padrões de treinamento. O erro é a diferença entre o que a saída do "Adaline" deveria ser e a saída da somatória atual.
- 1969 - Marvin Minsky e Seymour Papert criticam a teoria das redes neurais, afirmando que elas serviam apenas para resolução de problemas insignificantes. Mostram que o "Perceptron" não apresentava solução para alguns problemas simples, como por exemplo, a função lógica "ou exclusivo".
- 1970 - Stephen Grossberg propõe a função de ativação do neurônio do tipo sigmoidal.
- 1972 - Teuvo Kohonen apresenta um neurônio linear e contínuo diferente do descrito por McCulloch e Pitts em 1943 e Widrow e Hoff em 1960.



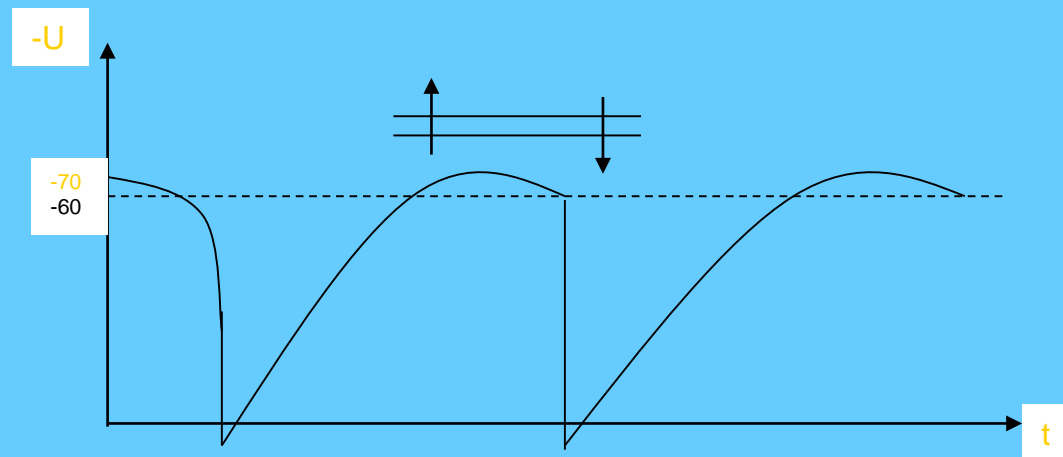
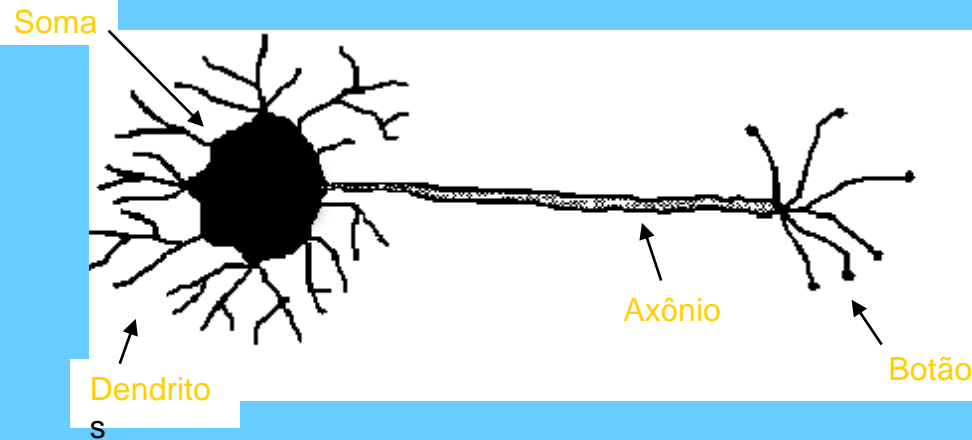
# Breve Histórico das RNA (cont.)

---

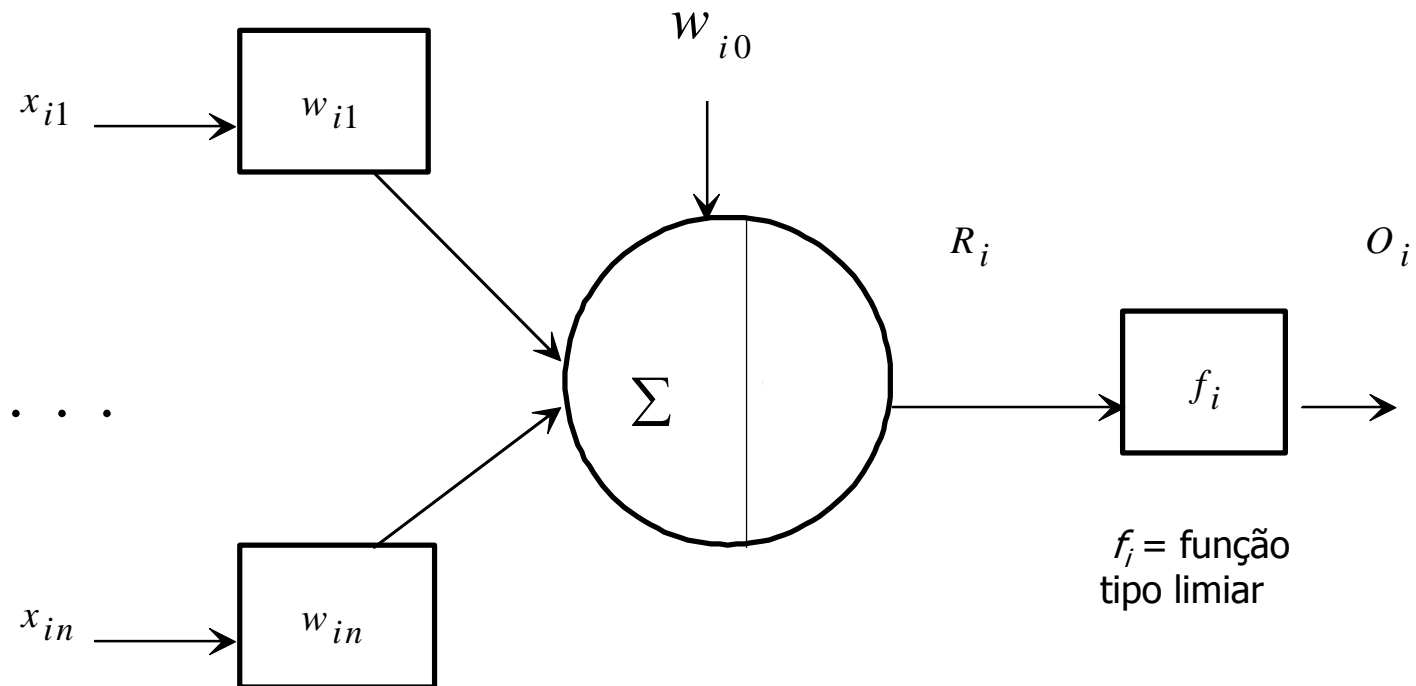
1982 - John Hopfield realiza estudos relativos ao armazenamento e restabelecimento de informação em redes neurais (memórias auto-associativas).

1986 - David E. Rumelhart, James L. McClelland e um grupo de pesquisadores retomam estudos em redes neurais baseadas nos "Perceptrons". Isso foi iniciado pela descoberta de um eficiente algoritmo de aprendizagem anteriormente pesquisado por Paul Werbos em 1974, para determinação dos "pesos" de ligação entre os neurônios. Surgimento do Perceptron multicamadas.

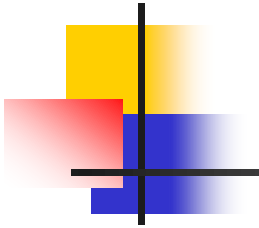
# Modelo Biológico do Neurônio



# Modelo Artificial de McCulloch e Pitts



# O Perceptron e o Problema "XOR"



$w_{ij} = [w_{i1}, w_{i2}, \dots, w_{in}]$  é o vetor de pesos das entradas do i-ésimo neurônio;

$x_{ij} = [x_{i1}, x_{i2}, \dots, x_{in}]$  é o vetor de entrada do i-ésimo neurônio;

$R_i = \sum w_{ij} x_{ij}$  é a ativação do neurônio i;

$O_i = f\left(\sum_{j=1}^n w_{ij} \cdot x_{ij} + w_0\right)$  é a saída do neurônio i;

$T_i$  é o padrão desejado de saída;

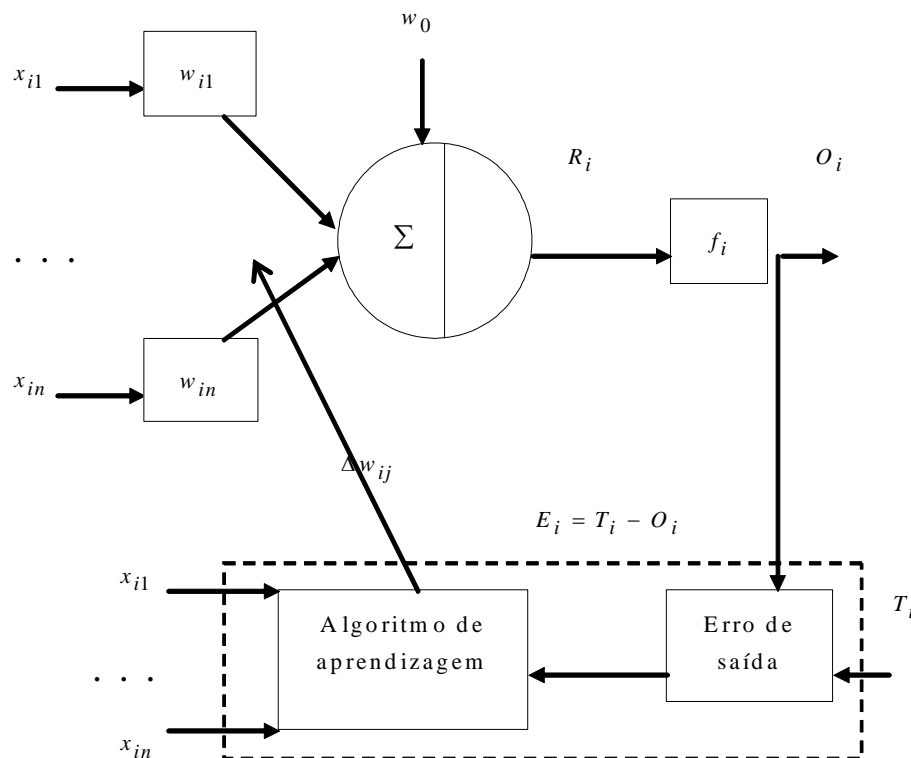
$E_i = T_i - O_i$  é o erro de saída utilizado durante a aprendizagem;

$\Delta w_{ij}$  é a variação dos pesos durante a aprendizagem.

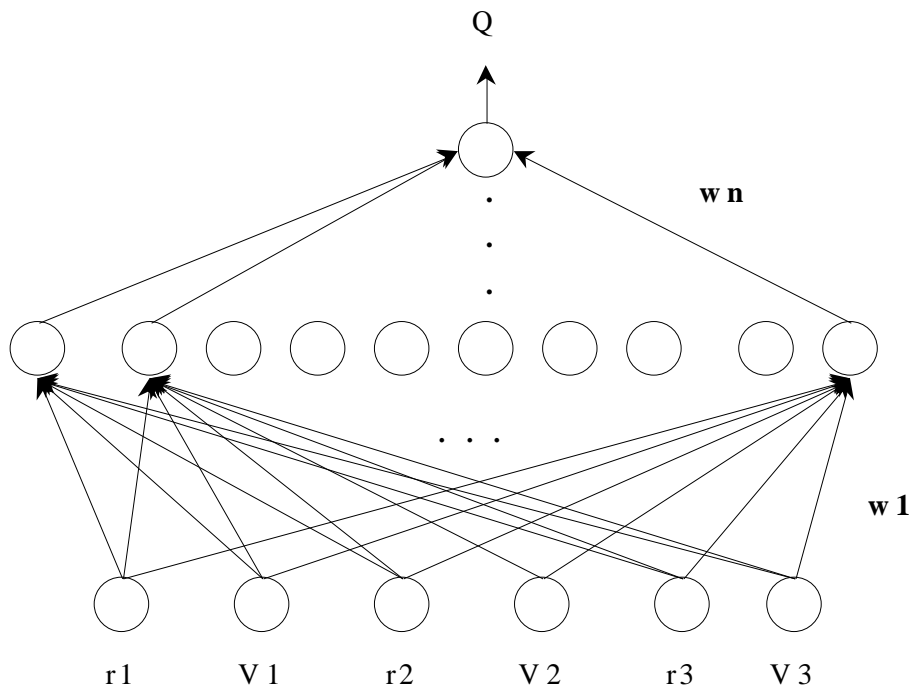
A regra de aprendizagem pode ser expressa como:

$$w_{ij}(k+1) = w_{ij}(k) + \text{correção}$$

$k$  indica o passo do processo iterativo.



# O Perceptron e o Problema "XOR"



Problema "X-Or": Problema da separabilidade linear do Perceptron ("Decision Boundaries")

Solução: MLP e o algoritmo "backpropagation"



# Áreas de Aplicação

---

- **Aproximador universal de funções:** mapeamento funcional entre variáveis;
- **Controle e Identificação de processos:** robótica, aeronaves;
- **Reconhecimento/Classificação de padrões:** reconhecimento de voz, escrita;
- **Agrupamento de dados ("clustering"):** identificação automática de classes e garimpagem de dados;
- **Sistemas de previsão:** séries temporais, previsões climáticas;
- **Otimização:** otimizar custos (Ex: programação dinâmica);
- **Memórias associativas:** recuperação de padrões (transmissão de sinais, processamento de imagens).



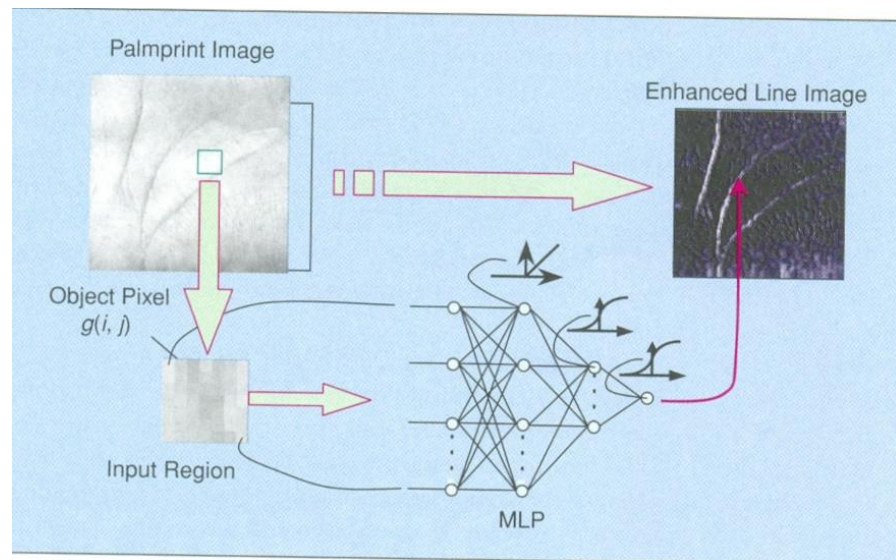
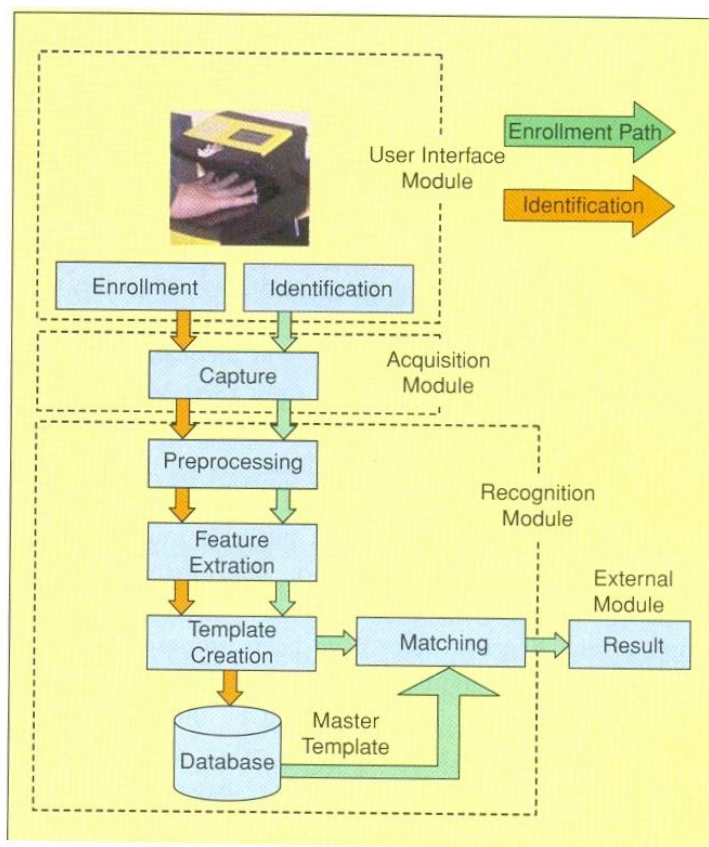
# Exemplos de Aplicações das RNA

---

- ADALINE e MADALINE de Widrow, B., com aplicações em filtragem de sinal adaptativa;
- "Backpropagation Perceptron" de Rumelhart, D., Parker, D. e Werbos, P. com aplicações em reconhecimento e classificação de padrões, filtragem de sinal, compressão de dados, etc.;
- "Adaptive Resonance Theory" (ART) de Grossberg, S. e Carpenter, G., aplicada em reconhecimento de padrões;
- "Brain-State-in-a-Box" (BSB) de Anderson, J., aplicada em otimização;
- "Self-organization and Associative Memory" (Quantização do vetor de aprendizagem) de Kohonen, T.;
- Rede de Hopfield, J., aplicado em otimização;
- Neocognitron de Fukushima, K., aplicado em reconhecimento de caracteres manuscritos e outras figuras;
- Máquinas de Boltzmann e Cauchy de Hinton, G., Sejnowski, T., Ackley, D. Szu, H., usadas em reconhecimento de padrões de imagens e otimização;
- BAM (Memória Associativa Bidirecional) de Kosko, B.;
- "Redes de Funções de Base Radial" de vários pesquisadores, usadas em classificação de padrões;
- "Time-delay" de Tank, D. e Hopfield, J., aplicada em reconhecimento da fala;
- "Recurrent" de Pineda, para controle robótico.
- Aplicação no controle e modelagem de sistemas dinâmicos.



# Aplicação em Biometria



IEEE Computational Intelligence v.2 N.2, 2007



# O Sistema LAP - ZTW

---

- Reconhecimento de placas automotivas.
- Derivado de TG desenvolvido na EEM.

# Sistema LAP ZTW

## ➔ Finalidade e Aplicações



Teste IR ZTW marginal pinheiros Afastamento  
Quarta-feira, dia 25/04/2007, às 22:06:12  
ANI-5518



Teste IR ZTW marginal pinheiros Afastamento Fujon 50mm F1.6 DN AI  
Quarta-feira, dia 25/04/2007, às 22:06:12 Imagem: 1111 Placa de Fundo Claro  
ANI-5518



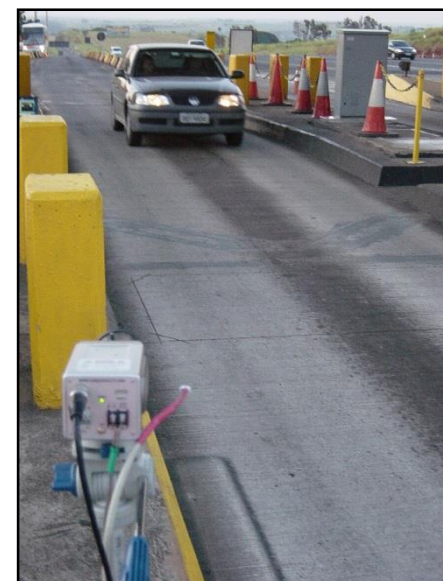
Cod. Equip: 7400 Agente: 108464 Enquadramento: 57.462 Imagem: 5777  
Av. Brigadeiro Faria Lima, 4433 Segunda-feira, dia 3/4/2006, às 07:37:23



CAPTURA DE IMAGEM PARA GERAÇÃO DE PADRÕES  
Cod. Equip: 7400 Imagem: 119565  
DQO-3266



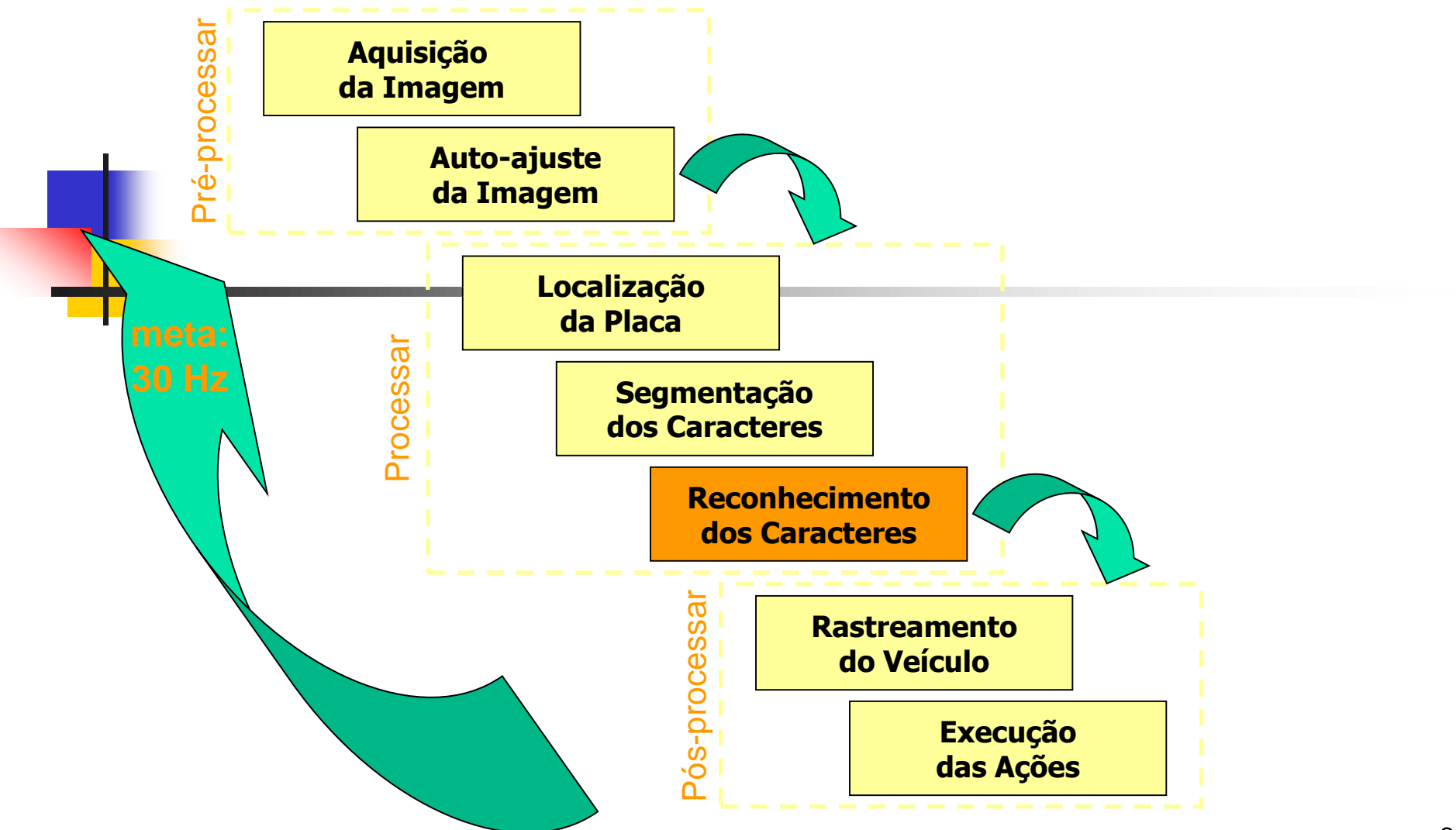
CAPTURA DE IMAGEM PARA GERAÇÃO DE PADRÕES Terça-feira, dia 18/4/2006, às 16:36:24  
Cod. Equip: 7400 Imagem: 119565  
DQO-3266



Cod. Equip: 7400 Agente: 108464 Enquadramento: 57.462 Imagem: 5777  
Av. Brigadeiro Faria Lima, 4433 Segunda-feira, dia 3/4/2006, às 07:37:23

# Sistema LAP ZTW

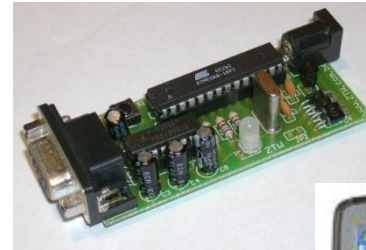
## → Etapas de Processamento





# Sistema LAP ZTW

## → Componentes do Sistema



# Sistema LAP ZTW

## ➔ Aquisição e Tratamento da Imagem

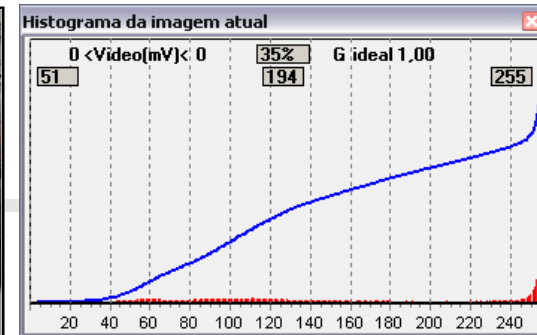
**Aquisição  
da Imagem**

```
SwitchCamera;  
ImgArray:=grab(handle_FG);
```

**Auto-ajuste  
da Imagem**

```
SetVideoLevel;  
SetContrast;  
SetBrightness;
```

**Localização  
da Placa**

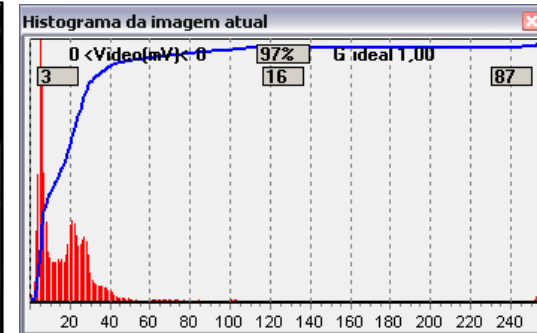


**Segmentação  
dos Caracteres**

**Reconhecimento  
dos Caracteres**

**Rastreamento  
do Veículo**

**Execução  
das Ações**



# Sistema LAP ZTW

## ➔ Localização da Placa

**Aquisição  
da Imagem**

**Auto-ajuste  
da Imagem**

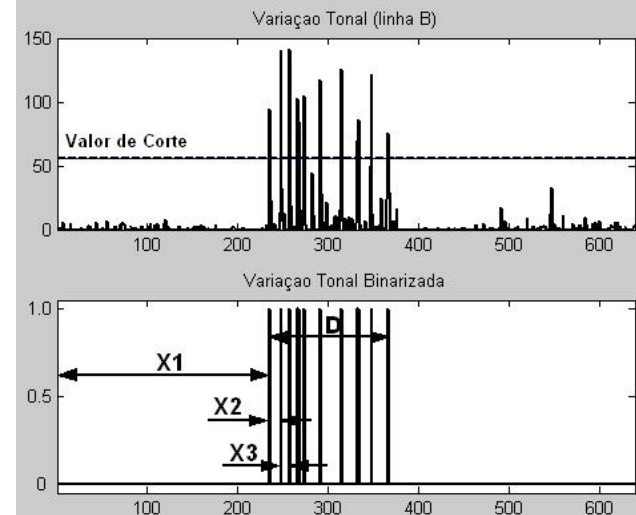
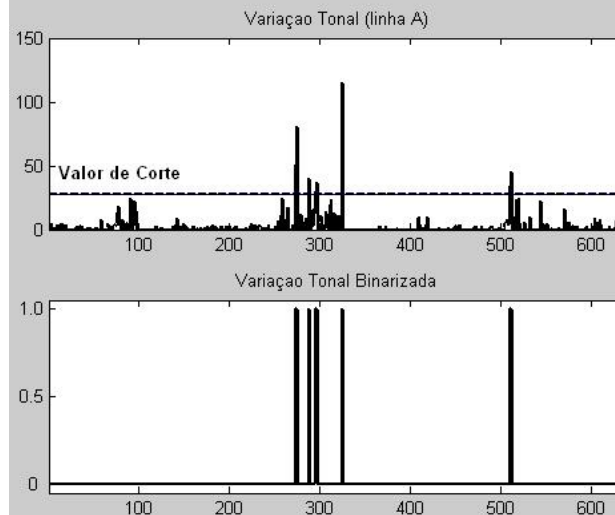
**Localização  
da Placa**

**Segmentação  
dos Caracteres**

**Reconhecimento  
dos Caracteres**

**Rastreamento  
do Veículo**

**Execução  
das Ações**



# Sistema LAP ZTW

➔ Extração da posição dos caracteres na imagem

Aquisição  
da Imagem

Auto-ajuste  
da Imagem

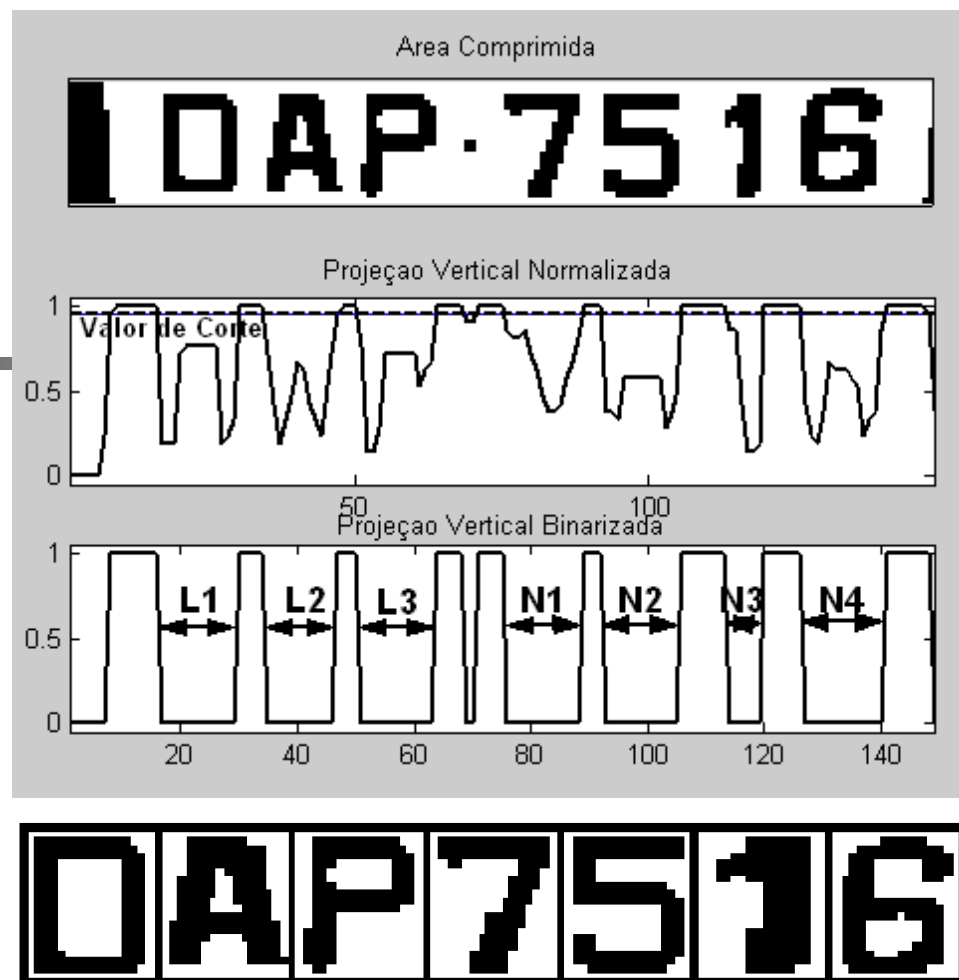
Localização  
da Placa

Segmentação  
dos Caracteres

Reconhecimento  
dos Caracteres

Rastreamento  
do Veículo

Execução  
das Ações





# Sistema LAP ZTW

## → Codificação

Aquisição  
da Imagem

Auto-ajuste  
da Imagem

Localização  
da Placa

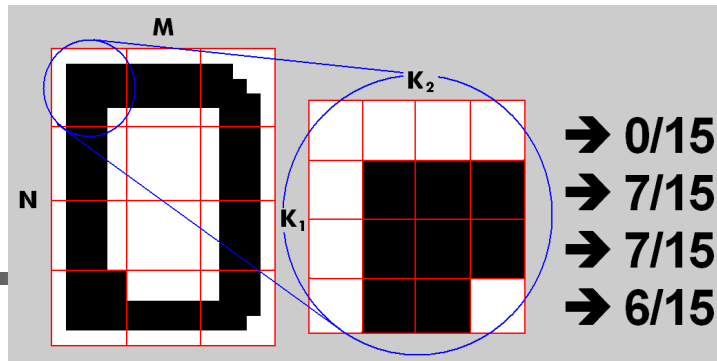
Segmentação  
dos Caracteres

**Reconhecimento  
dos Caracteres**

Rastreamento  
do Veículo

Execução  
das Ações

Caractere com  $M=12$  e  $N=16$  pixels



Redimensionar;  
GerarEntradas;  
FeedForward;

**Transformada ZTW:**  
Reduz entradas de  
192 para 48 (25%)

	0000	0	0/15
	0001	1	1/15
	0010	2	2/15
	0011	3	3/15
	0100	4	4/15
	0101	5	5/15
	0110	6	6/15
	0111	7	7/15
	1000	8	8/15
	1001	9	9/15
	1010	10	10/15
	1011	11	11/15
	1100	12	12/15
	1101	13	13/15
	1110	14	14/15
	1111	15	15/15

# Sistema LAP ZTW

## ➔ Classificação dos caracteres

**Aquisição da Imagem**

**Auto-ajuste da Imagem**

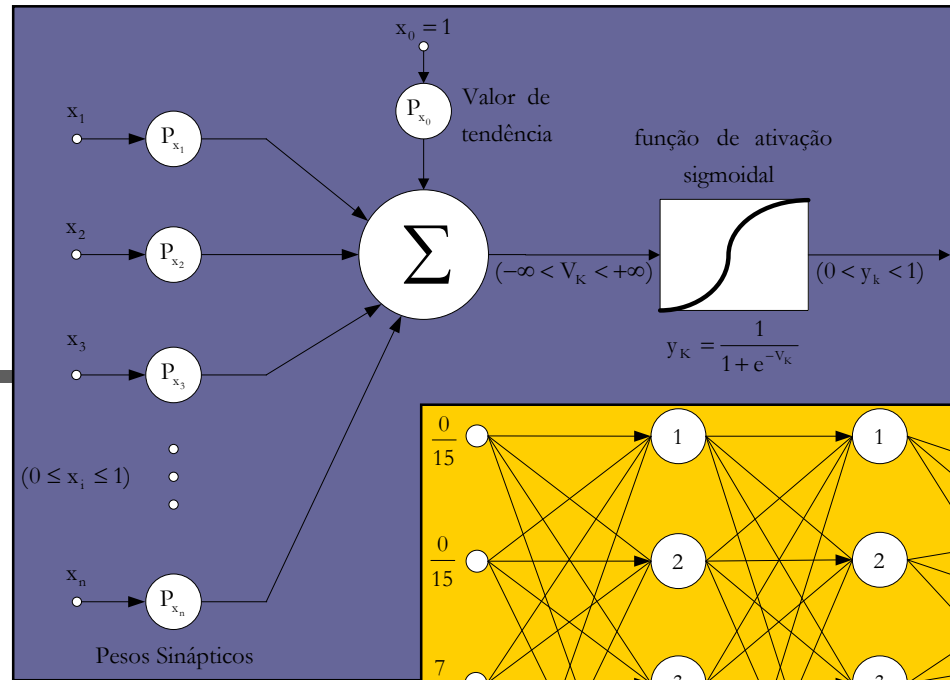
**Localização da Placa**

**Segmentação dos Caracteres**

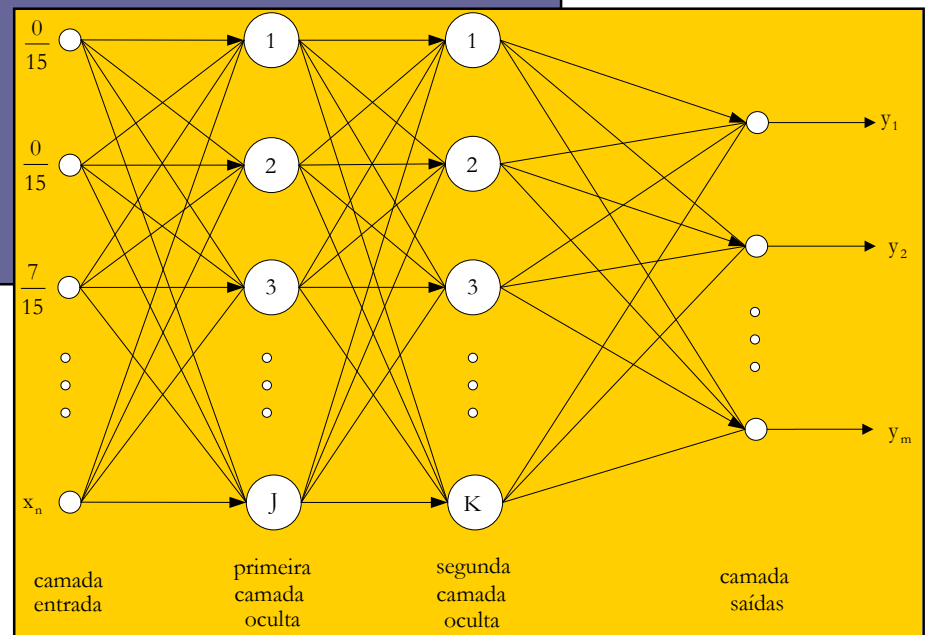
**Reconhecimento dos Caracteres**

**Rastreamento do Veículo**

**Execução das Ações**



Multi Layer Perceptron  
Error Backpropagation





# Desenvolvimento da RNA

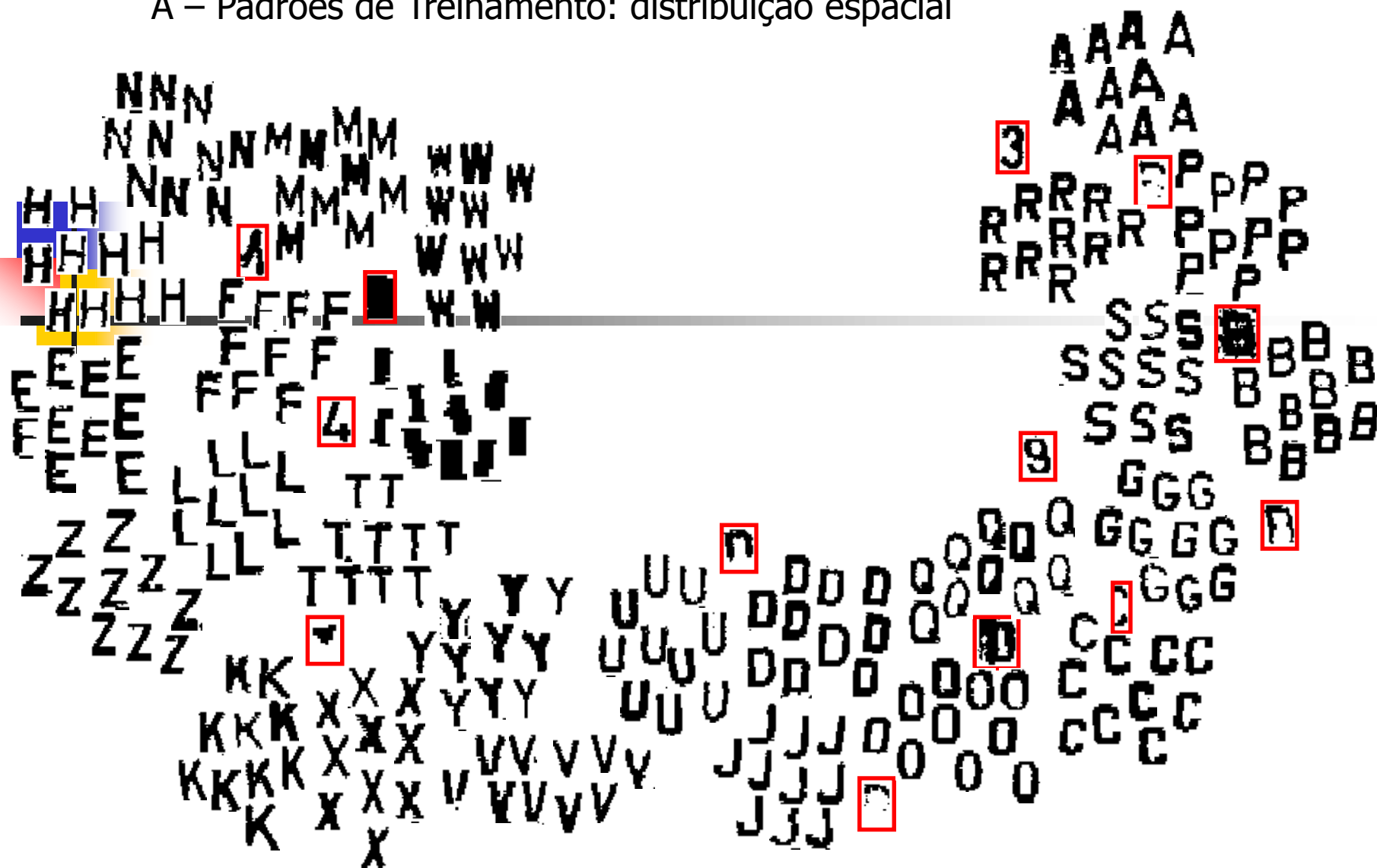
---

## **Etapas**

- A. Padrões de Treinamento**
- B. Projeto**
- C. Treinamento**
- D. Teste de Generalização**
- E. Reprodução da Rede em Delphi**
- F. Teste em Campo**

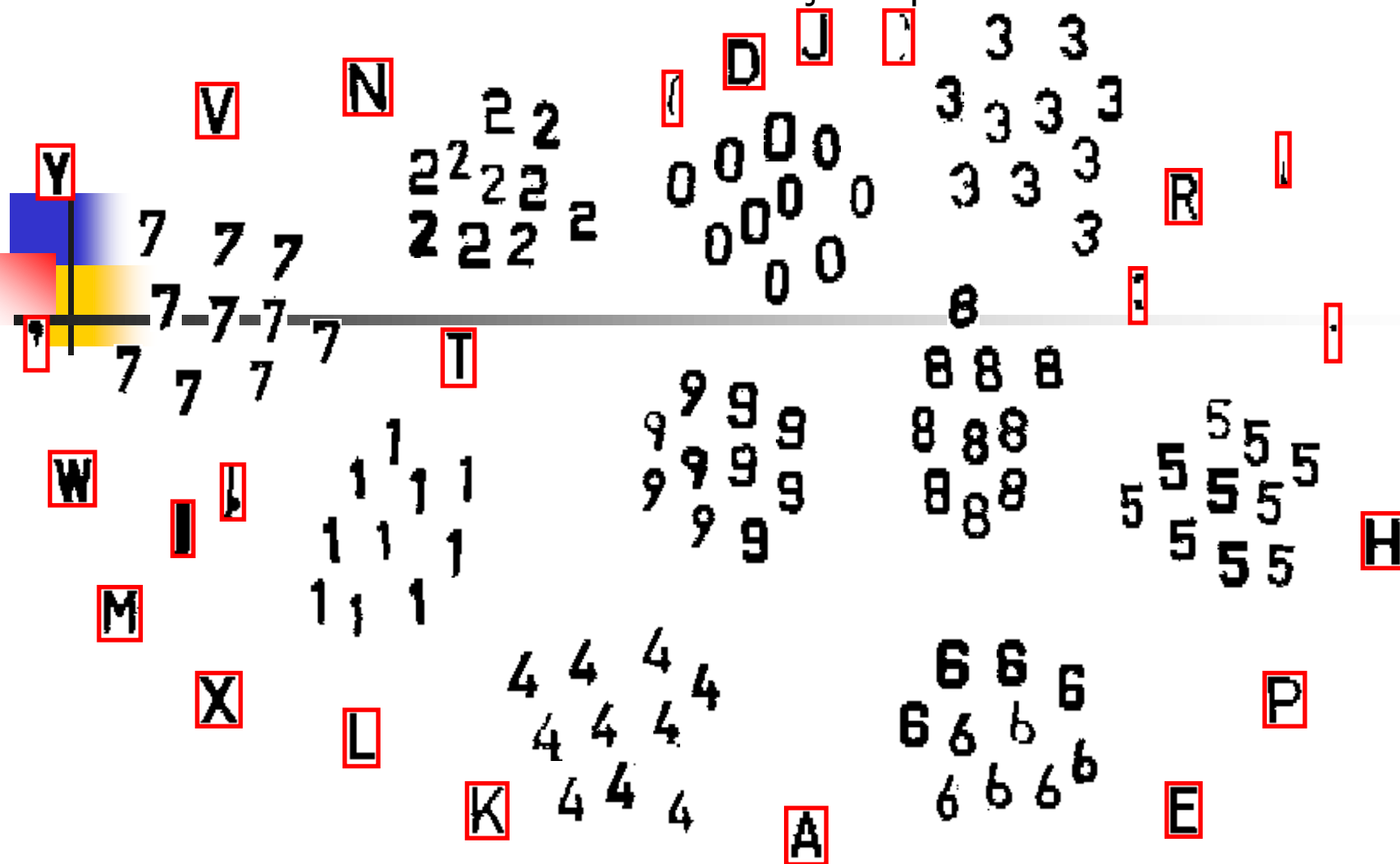
# Desenvolvimento de RNAs

A – Padrões de Treinamento: distribuição espacial



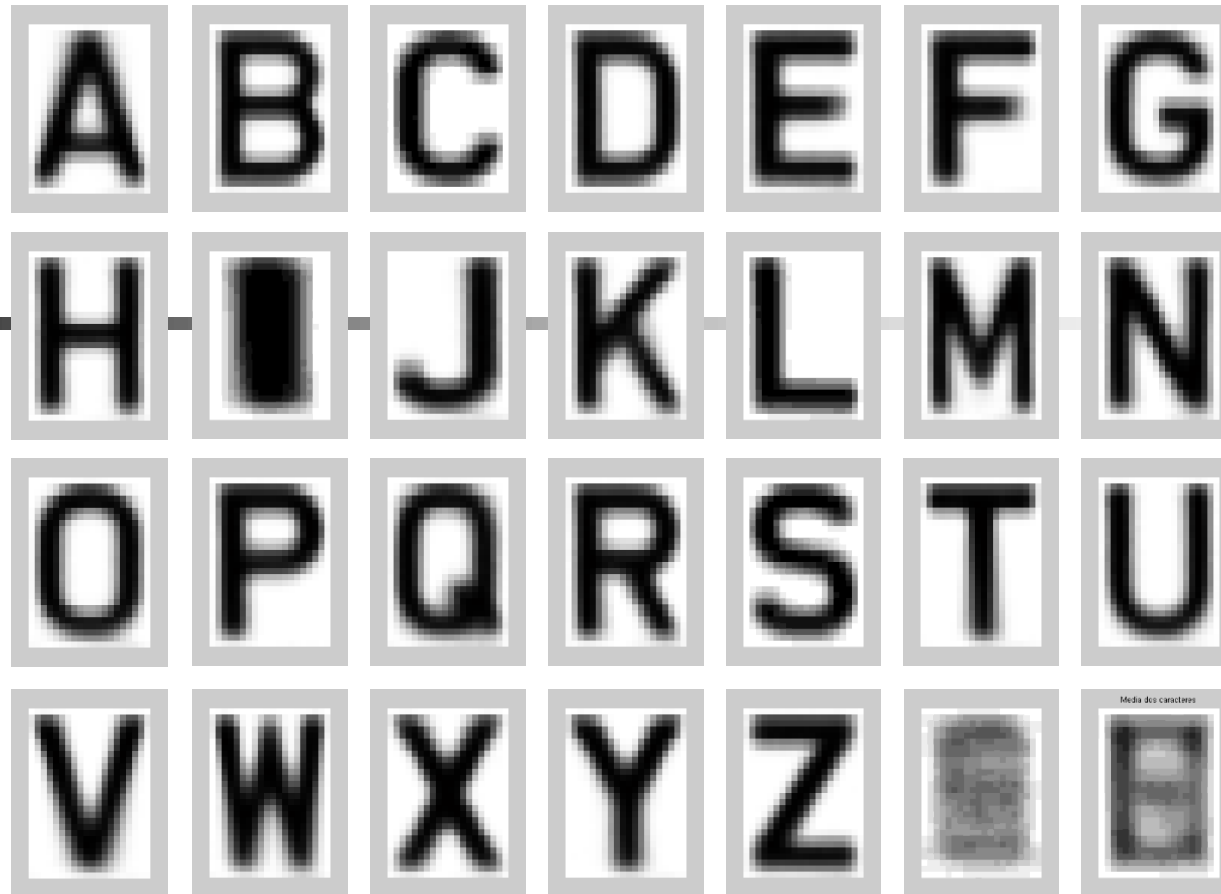
# Desenvolvimento de RNAs

A – Padrões de Treinamento: distribuição espacial



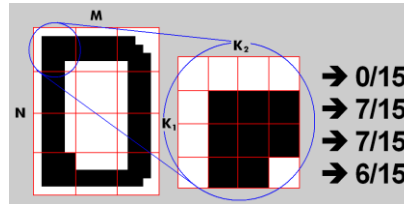
# Desenvolvimento de RNAs

A – Padrões de Treinamento: média de 500 padrões por classe



# Desenvolvimento de RNAs

## A – Criação do Conjunto de Treino



0000	0	0/15
0001	1	1/15
0010	2	2/15
0011	3	3/15
0100	4	4/15
0101	5	5/15
0110	6	6/15
0111	7	7/15
1000	8	8/15
1001	9	9/15
1010	10	10/15
1011	11	11/15
1100	12	12/15
1101	13	13/15
1110	14	14/15
1111	15	15/15

**Entradas** (x 15, com colunas omitidas)

## Saídas Desejadas

[illegible]



# Projeto: Considerações

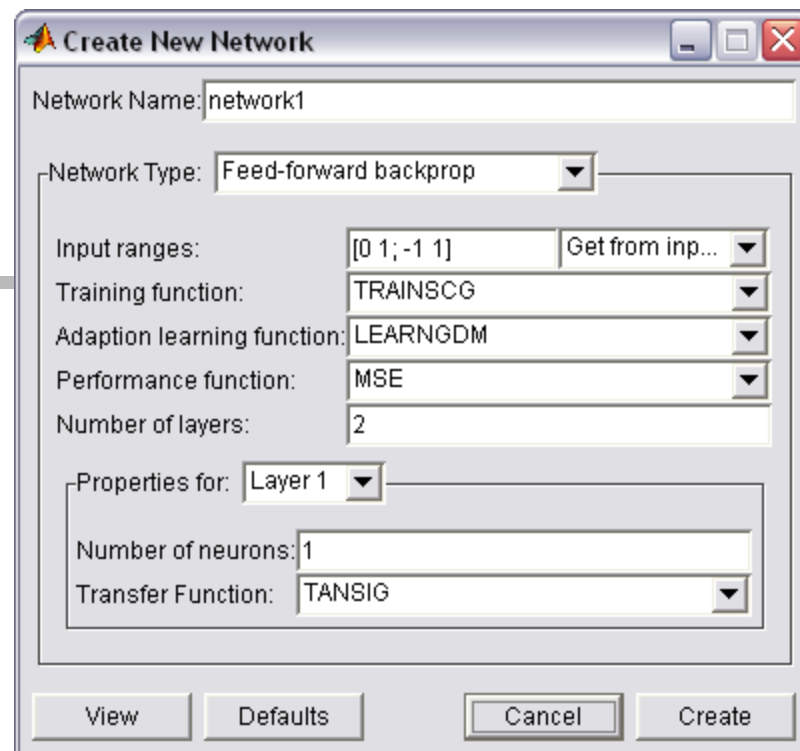
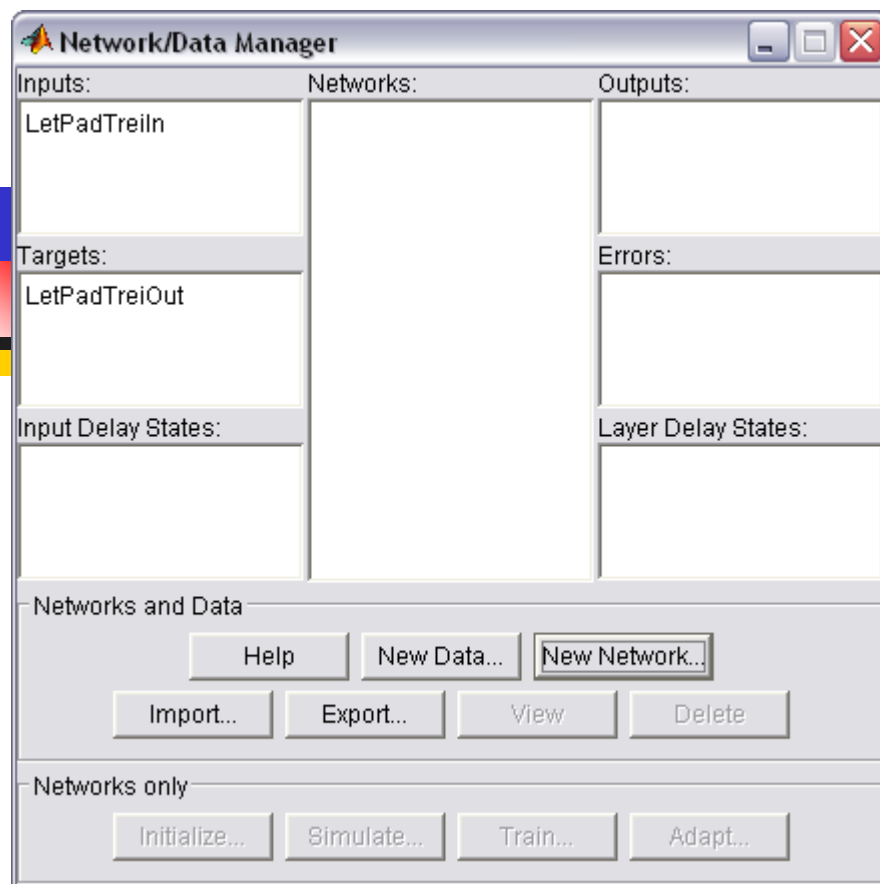
---

- **Tipo de rede usada: MLP**
- **Função de treinamento: backpropagation**
  - ✓ **TRAINGDM: Gradient descent with momentum**
  - ✓ **TRAINGDA: Gradient descent with adaptive learning rate**
  - ✓ **TRAINGDX: Gradient descent with momentum & adaptive learning rate**
  - ✓ **TRAINSCG: Scaled conjugate gradient**
- **Topologia**
  - ✓ **Número de camadas**
  - ✓ **Neurônios por camada**
  - ✓ **Conectividade**
  - ✓ **Realimentação**
- **Função de ativação: sigmóide**



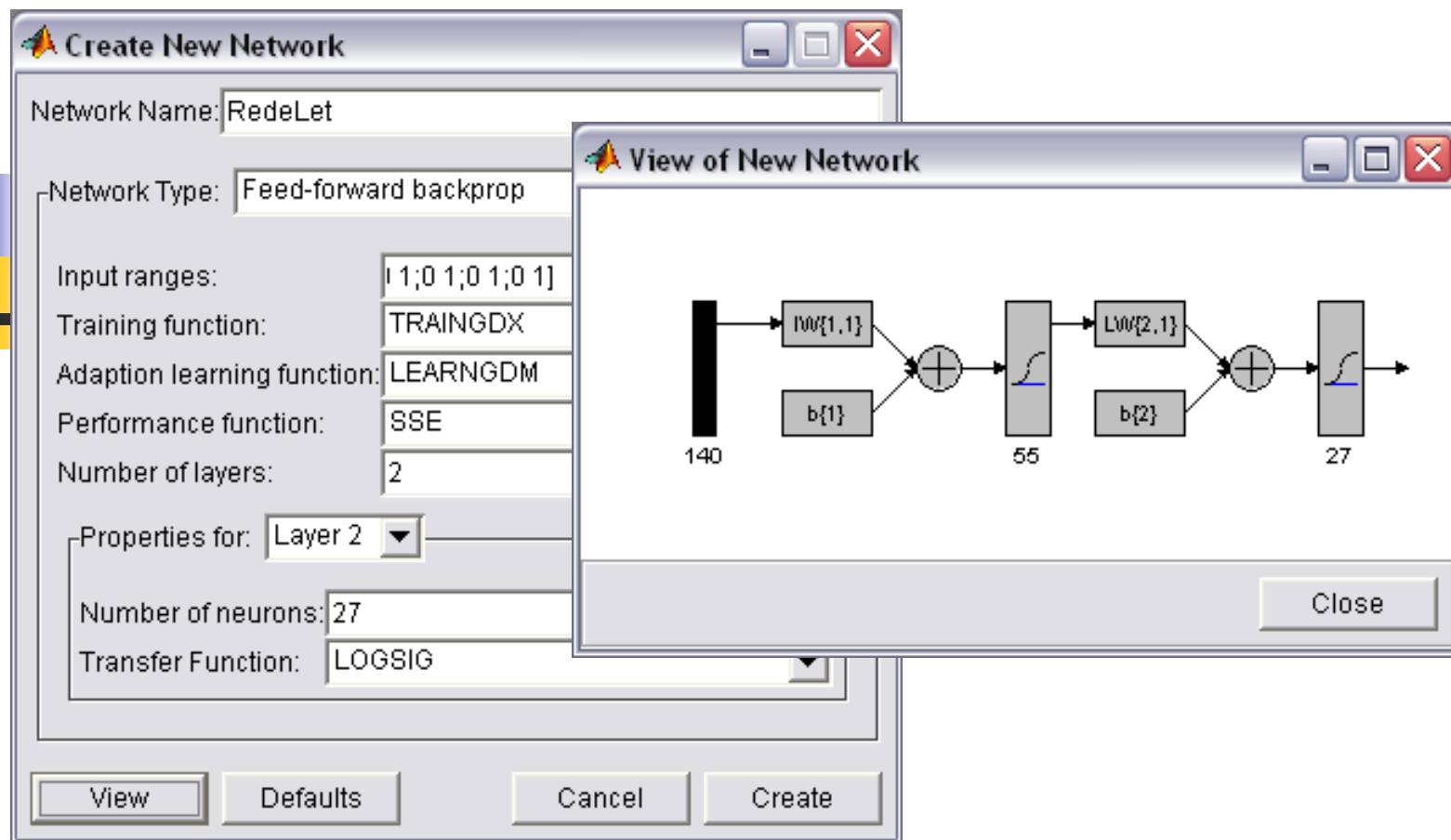
# Desenvolvimento de RNAs

## B – Projeto: nntool (Neural Network Tool)



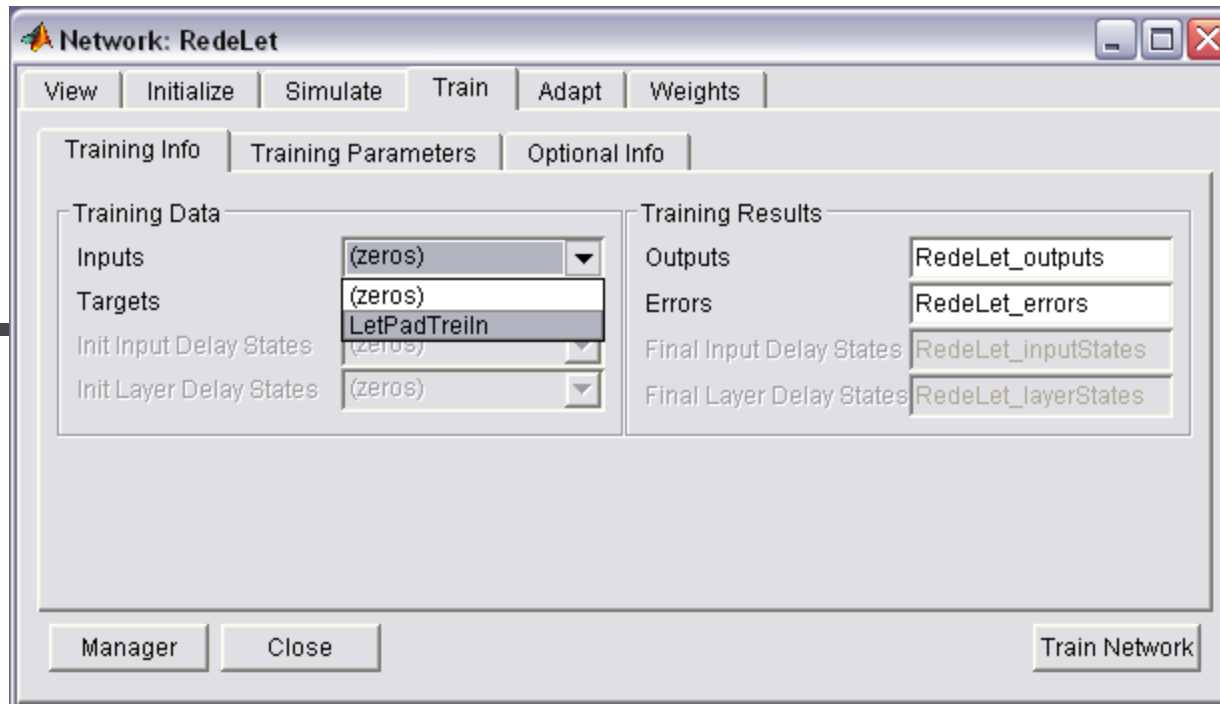
# Desenvolvimento de RNAs

## B – Projeto: visualização da rede criada



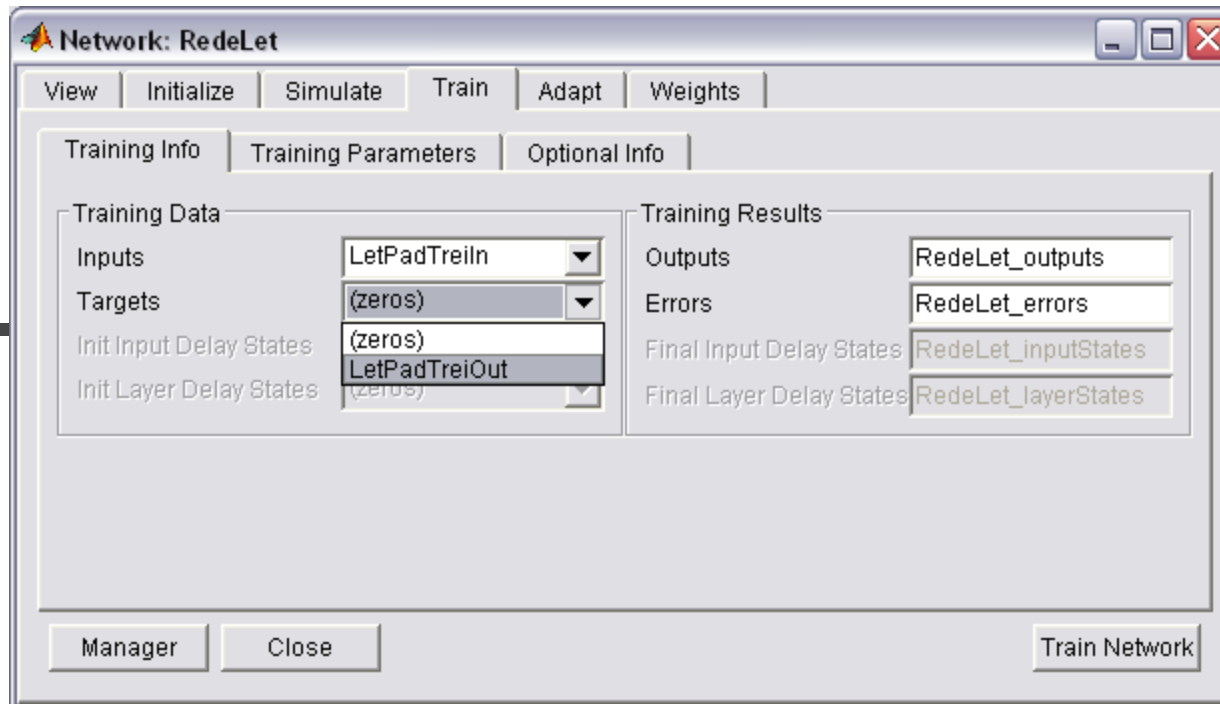
# Desenvolvimento de RNAs

C – Treinamento: escolha do vetor de “entradas”



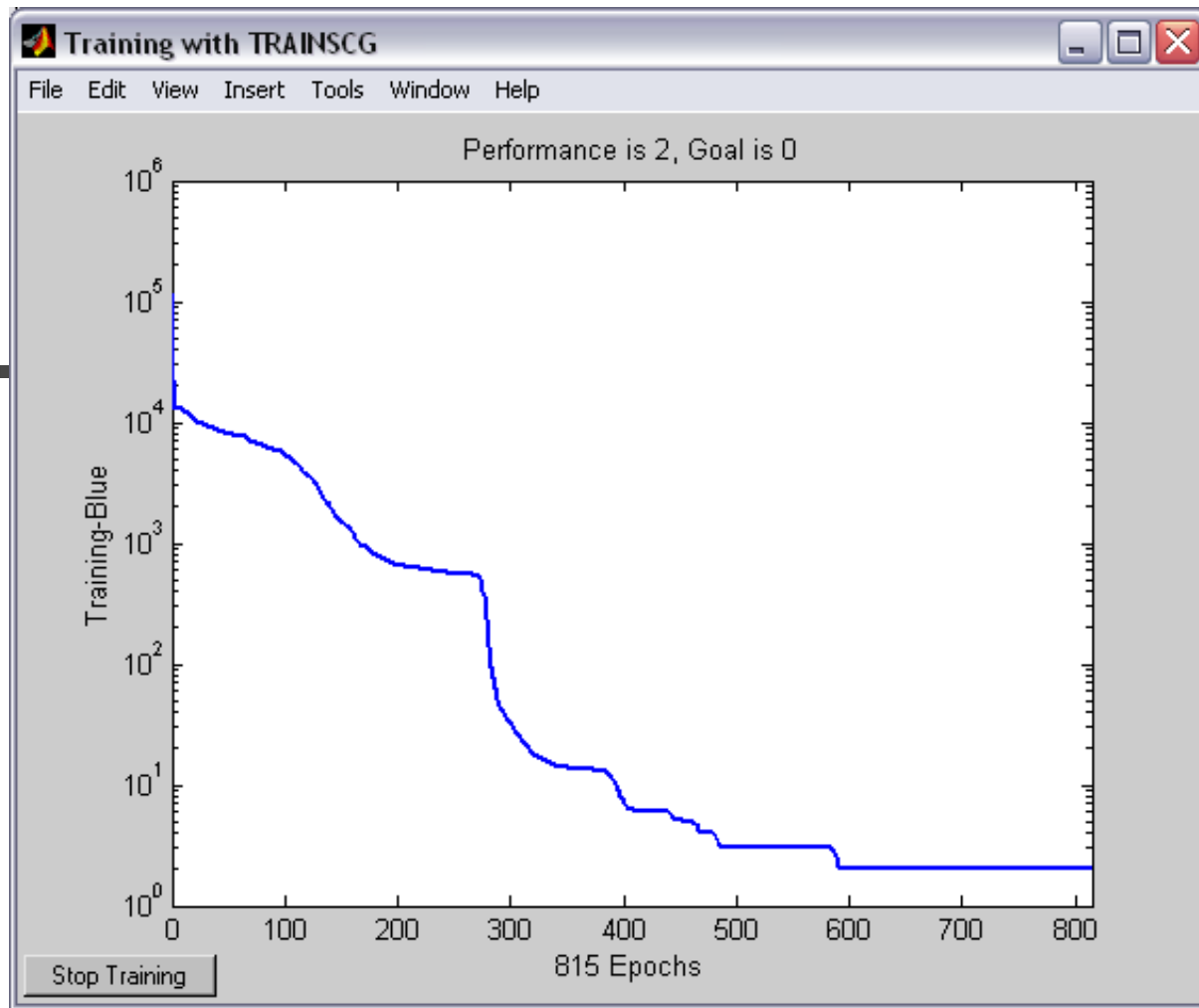
# Desenvolvimento de RNAs

C – Treinamento: escolha do vetor de “saídas desejadas”



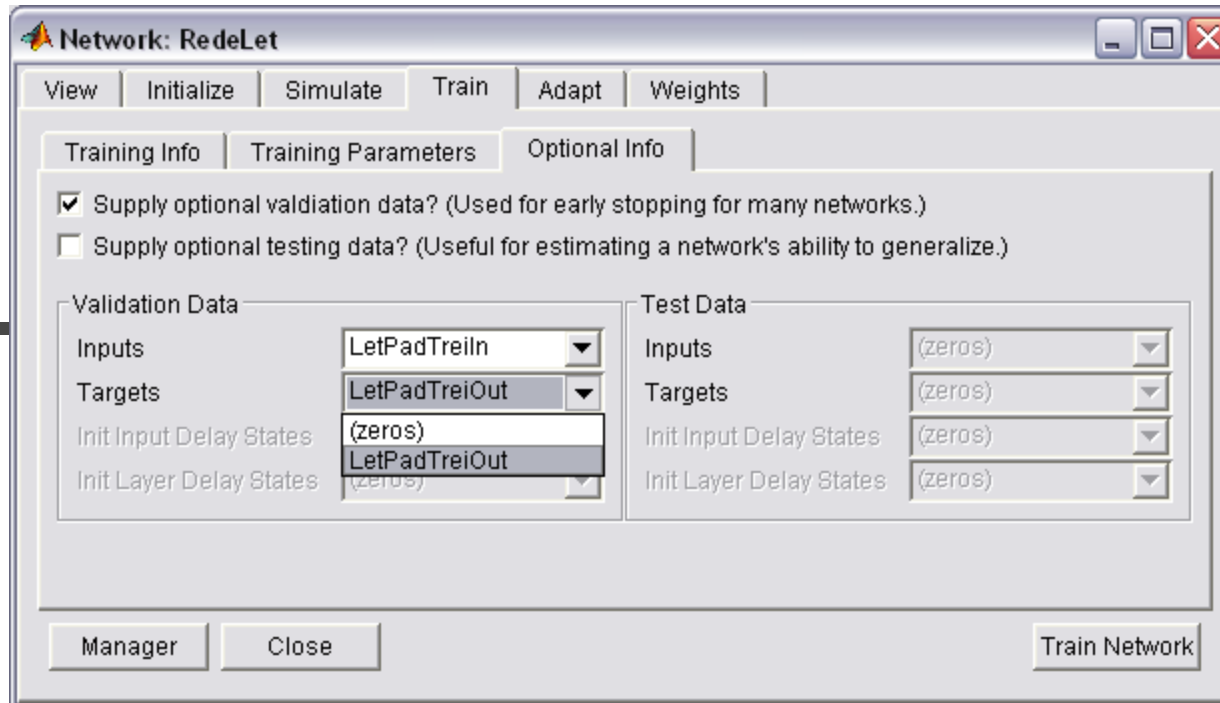
# Desenvolvimento de RNAs

C – Treinamento: curva de erro da RedeLet (com 13k padrões)



# Desenvolvimento de RNAs

## D – Teste de Generalização: conjunto de validação



# Desenvolvimento de RNAs

## D – Teste de Generalização: apuração dos resultados

Índice geral de acerto da RedeLet (padrões com menos de 75% de confiabilidade foram ignorados)

Erro estimado na identificação de uma letra: **3,87%**

Acerto de uma letra individualmente: **96,13%**

Acerto geral das 3 letras da placa simultaneamente: **88,84%**

ERROS		Antiga identificação: f(nome)																										Qtdd de Erros	% de Erro	Total de Padrões
		A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z			
Nova identificação: f(pasta)	A	26	0	0	1	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	29	0,56%	5.284	
	B	0	34	0	19	0	0	0	0	0	0	0	0	0	0	0	0	1	4	0	0	0	0	0	0	0	58	1,01%	5.693	
	C	1	1	105	13	1	1	18	0	0	0	0	2	0	0	5	0	1	0	1	0	0	0	0	0	0	149	0,99%	15.100	
	D	0	18	2	1.347	0	0	2	0	0	0	0	1	0	0	71	0	4	3	0	0	0	0	0	0	0	1.448	13,52%	10.717	
	E	0	3	3	0	3	7	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	18	0,56%	3.277	
	F	0	0	1	1	30	86	0	1	0	0	0	4	0	0	0	4	0	0	0	0	0	0	0	0	0	127	3,78%	3.368	
	G	1	4	2	2	1	0	9	0	1	0	0	0	1	1	1	0	0	1	1	0	0	0	0	0	0	25	0,80%	3.165	
	H	0	6	0	2	0	0	1	15	0	0	0	0	8	0	0	0	0	11	0	0	1	0	0	0	0	44	1,91%	2.302	
	I	16	0	2	1	1	4	0	0	189	0	0	6	0	1	0	0	1	0	0	7	0	0	2	1	2	233	5,90%	3.948	
	J	0	0	0	2	0	0	0	0	3	15	0	0	0	0	1	0	0	0	0	0	2	0	0	0	0	23	0,76%	3.054	
	K	0	0	0	0	0	1	0	1	0	0	21	0	2	0	0	0	0	1	0	0	0	0	0	0	0	26	1,10%	2.370	
	L	1	0	35	1	8	1	0	0	0	0	0	64	0	0	0	0	0	0	0	0	0	0	0	0	0	110	2,71%	4.073	
	M	0	0	0	0	0	0	0	0	0	0	0	0	52	0	0	0	0	0	0	0	0	0	2	0	0	54	1,34%	4.009	
	N	0	0	0	0	0	0	0	0	0	0	0	0	1	8	0	0	0	1	0	0	0	0	1	0	0	11	0,40%	2.668	
	O	0	11	2	317	0	0	21	0	0	0	0	0	0	1	394	0	11	0	0	0	3	0	0	0	0	760	13,40%	5.670	
	P	0	5	1	5	0	1	0	2	0	0	0	0	2	0	0	7	0	9	0	0	0	0	0	0	0	32	0,96%	3.289	
	Q	0	1	6	83	0	0	7	0	0	0	0	1	0	0	34	0	70	0	1	0	0	0	0	0	0	203	13,61%	1.488	
	R	0	2	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	8	0	0	0	0	0	0	0	11	0,27%	3.913	
	S	0	15	1	2	0	0	6	0	0	0	0	0	0	1	0	0	2	0	10	0	0	0	0	0	0	37	1,49%	2.451	
	T	0	0	0	0	0	0	0	0	11	0	0	0	0	0	0	0	0	0	13	0	0	0	0	0	0	24	1,29%	1.838	
	U	0	1	0	136	0	0	1	0	0	0	0	0	0	1	17	1	0	0	0	0	38	0	0	0	0	195	13,95%	1.398	
	V	0	0	0	0	0	0	0	0	0	0	0	1	3	0	0	0	0	0	0	0	0	4	0	0	2	10	0,49%	2.071	
	W	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	17	0	0	0	18	1,98%	908	
	X	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	2	1	5	0,27%	1.773	
	Y	0	1	0	0	0	0	0	0	4	1	0	0	3	0	0	0	0	0	0	2	0	6	0	0	58	0	75	4,31%	1.746
	Z	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	18	0,95%	2.014	
	nada	3	2	2	5	0	3	0	0	13	1	0	2	0	0	1	3	0	1	0	6	0	1	0	0	0	0	43	12,76%	337
	Total de Padrões	5.435	6.067	15.273	15.661	3.373	3.454	3.385	2.166	4.346	3.069	2.398	4.171	4.217	2.813	3.474	3.342	1.420	3.989	2.456	1.851	1.336	2.180	1.077	1.754	1.702	2.022	3.786	3,87%	102.431

OBS: A diagonal principal foi apurada por amostragem: quantidade de erros nos 1000 primeiros padrões.



# Desenvolvimento de RNAs

---

## **E – Reprodução da Rede em Delphi: considerações**

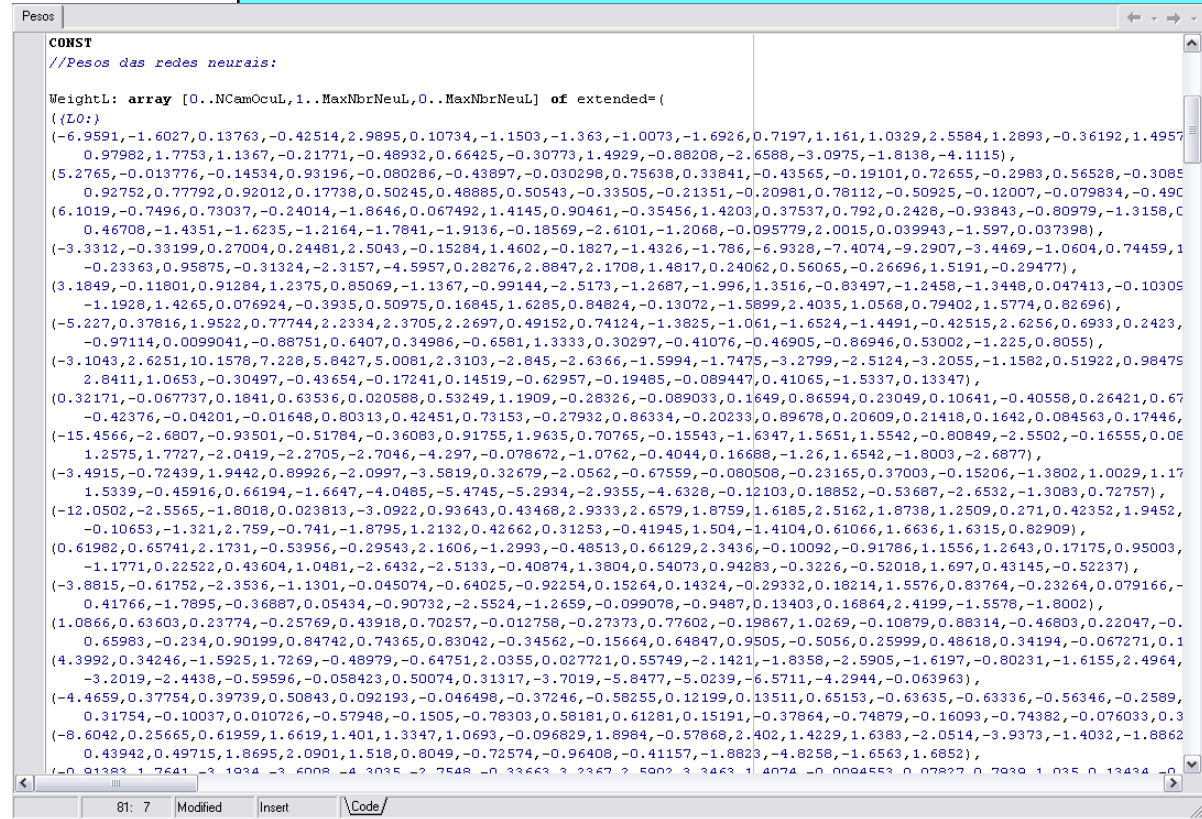
- **Copiar o valor dos pesos e “bias” (viés) do Matlab;**
- **Reproduzir o algoritmo.**



# Desenvolvimento de RNAs

## E – Reprodução da Rede em Delphi: algoritmo

```
procedure FeedForward;
var i, j, k: integer; //contadores de looping
    Campo: extended; //campo local induzido
begin
    //muda de camada
    for j:=0 to NCamOcuL do
    begin
        //muda de neurônio
        for k:=1 to NeuronsL[j+2] do
        begin
            //inicializa o Campo com o bias = Weight[j,k,0]
            Campo:=WeightL[j,k,0];
            //muda de entrada
            for i:=1 to NeuronsL[j+1] do
            begin
                Campo:=Campo+WeightL[j,k,i]*LayerOut[j,i];
                //Sigmoide(Campo);
                LayerOut[j+1,k]:=(1+exp(-Campo));
                //filtragem anti-saturação
                if abs(LayerOut[j+1,k]) >= LimiteAntiSaturacao then
                begin
                    if Campo < 0 then
                        LayerOut[j+1,k]:=Campo/abs(Campo)
                    else
                        LayerOut[j+1,k]:=0;
                end;
            end;
        end;
        //inicializa a variável do resultado final
        for i:=1 to NOPcoesMax do ResultCompet[i].Confia:=0;
        //competir
        for i:=1 to NeuronsL[NCamOcuL+2] do
        begin
            if LayerOut[NCamOcuL+1,i] > ResultCompet.Confia then
            begin
                //novo Máximo
                ResultCompet.Confia:=LayerOut[NCamOcuL+1,i];
                ResultCompet.Carac:=i;
            end;
        end;
    end;
end;
```



```
CONST
//Pesos das redes neurais:
WeightL: array [0..NCamOcuL, 1..MaxNbrNeuL, 0..MaxNbrNeuL] of extended = (
  (LO:
    (-6.9591,-1.6027,0.13763,-0.42514,2.9895,0.10734,-1.1503,-1.363,-1.0073,-1.6926,0.7197,1.161,1.0329,2.5584,1.2893,-0.36192,1.4957,
      0.97982,1.7753,1.1367,-0.21771,-0.48932,0.66425,-0.30773,1.4929,-0.88208,-2.6588,-3.0975,-1.8138,-4.1115),
    (5.2765,-0.013776,-0.14534,0.93196,-0.080286,-0.43897,-0.030298,0.75638,0.33841,-0.43565,-0.19101,0.72655,-0.2983,0.56528,-0.3085,
      0.92752,0.77792,0.92012,0.17738,0.50245,0.48885,0.50543,-0.33505,-0.21351,-0.20981,0.78112,-0.50925,-0.12007,-0.079834,-0.49C
      6.1019,-0.7496,0.73037,-0.24014,-1.8646,0.067492,1.4145,0.90461,-0.35456,1.4203,0.37537,0.792,0.2428,-0.93843,-0.80979,-1.3158,C
      0.46708,-1.4351,-1.6235,-1.2164,-1.7841,-1.9136,-0.18569,-2.6101,-1.2068,-0.095779,2.0015,0.039943,-1.597,0.037398),
    (-3.3312,-0.33199,0.27004,0.24481,2.5043,-0.15284,1.4602,-0.1827,-1.4326,-1.786,-6.9328,-7.4074,-9.2907,-3.4469,-1.0604,0.74459,1
      -0.23363,0.95875,-0.31324,-2.3157,-4.5957,0.28276,2.8847,2.1708,1.4817,0.24062,0.56065,-0.26696,1.5191,-0.29477),
    (3.1849,-0.11801,0.91284,1.2375,0.85069,-1.1367,-0.99144,-2.5173,-1.2687,-1.996,1.3516,-0.83497,-1.2458,-1.3448,0.047413,-0.10305
      -1.1928,1.4265,0.076924,-0.3935,0.50975,0.16845,1.6285,0.84824,-0.13072,-1.5899,2.4035,1.0568,0.79402,1.5774,0.82696),
    (-5.227,0.37816,1.9522,0.77744,2.2334,2.3705,2.2697,0.49152,0.74124,-1.3825,-1.061,-1.6524,-1.4491,-0.42515,2.6256,0.6933,0.2423,
      -0.97114,0.0099041,-0.88751,0.6407,0.34986,-0.6581,1.3333,0.30297,-0.41076,-0.46905,-0.86946,0.53002,-1.225,0.8055),
    (-3.1043,2.6251,10.1578,7.228,5.8427,5.0081,2.3103,-2.845,-2.6366,-1.5994,-1.7475,-3.2799,-2.5124,-3.2055,-1.1582,0.51922,0.98475
      2.8411,1.0653,-0.30497,-0.43654,-0.17241,0.14519,-0.62957,-0.19485,-0.089447,0.41065,-1.5337,0.13347),
    (0.32171,-0.067737,0.1841,0.63536,0.020588,0.53249,1.1909,-0.28326,-0.089033,0.1649,0.86594,0.23049,0.10641,-0.40558,0.26421,0.67
      -0.42376,-0.04201,-0.01648,0.80313,0.42451,0.73153,-0.27932,0.86334,-0.20233,0.89678,0.20609,0.21418,0.1642,0.084563,0.17446,
      -15.4566,-2.6807,-0.93501,-0.51784,-0.36083,0.91755,1.9635,0.70765,-0.15543,-1.6347,1.5651,1.5542,-0.80849,-2.5502,-0.16555,0.0E
      1.2575,-1.7727,-2.0419,-2.2705,-2.7046,-4.297,-0.078672,-1.0762,-0.4044,0.16688,-1.26,1.6542,-1.8003,-2.6877),
    (-3.4915,-0.72439,1.9442,0.89926,-2.0997,-3.5819,0.32679,-2.0562,-0.67559,-0.080508,-0.23165,0.37003,-0.15206,-1.3802,1.0029,1.17
      1.5339,-0.45916,0.66194,-1.6647,-4.0485,-5.4745,-5.2934,-2.9355,-4.6328,-0.12103,0.18852,-0.53687,-2.6532,-1.3083,0.72757),
    (-12.0502,-2.5565,-1.8018,0.023813,-3.0922,0.93643,0.43468,2.9333,2.6579,1.8759,1.6185,2.5162,1.8738,1.2509,0.271,0.42352,1.9452,
      -0.10653,-1.321,2.759,-0.741,-1.8795,1.2132,0.42662,0.31253,-0.41945,1.504,-1.4104,0.61066,1.6636,1.6315,0.82909),
    (0.61982,0.65741,2.1731,-0.53956,-0.29543,2.1606,-1.2993,-0.48513,0.66129,2.3436,-0.10092,-0.91786,1.1556,1.2643,0.17175,0.95003,
      -1.1771,0.22522,0.43604,1.0481,-2.6432,-2.5133,-0.40874,1.3804,0.54073,0.94283,-0.3226,-0.52018,1.697,0.43145,-0.52237),
    (-3.8815,-0.61752,-2.3536,-1.1301,-0.045074,-0.64025,-0.92254,0.15264,0.14324,-0.29332,0.18214,1.5576,0.83764,-0.23264,0.079166,-
      0.41766,-1.7895,-0.36887,0.05434,-0.90732,-2.5524,-1.2659,-0.099078,-0.9487,0.13403,0.16864,2.4199,-1.5578,-1.8002),
    (1.0866,0.63603,0.23774,-0.25769,0.43918,0.70257,-0.012758,-0.27373,0.77602,-0.19867,1.0269,-0.10879,0.88314,-0.46803,0.22047,-0.
      6.5983,-0.234,0.90199,0.84742,0.74365,0.83042,-0.34562,-0.15664,0.64847,0.9505,-0.5056,0.25999,0.48618,0.34194,-0.067271,0.1
      4.3992,0.34246,-1.5925,1.7269,-0.48979,-0.64751,2.0355,0.027721,0.55749,-2.1421,-1.8358,-2.5905,-1.6197,-0.80231,-1.6155,2.4964,
      -3.2019,-2.4438,-0.59596,-0.058423,0.50074,0.31317,-3.7019,-5.8477,-5.0239,-6.5711,-4.2944,-0.063963),
    (-4.4659,0.37754,0.39739,0.50843,0.092193,-0.046498,-0.37246,-0.58255,0.12199,0.13511,0.65153,-0.63635,-0.63336,-0.56346,-0.2589,
      0.31754,-0.10037,0.010726,-0.57948,-0.1505,-0.78303,0.58181,0.61281,0.15191,-0.37864,-0.74879,-0.16093,-0.74382,-0.076033,0.3
      (-8.6042,0.25665,0.61959,1.6619,1.401,1.3347,1.0693,-0.096829,1.8984,-0.57868,2.402,1.4229,1.6383,-2.0514,-3.9373,-1.4032,-1.8862
      0.43942,0.49715,1.8695,2.0901,1.518,0.8049,-0.72574,-0.96408,-0.41157,-1.8823,-4.8258,-1.6563,1.6852),
    (-0.91383,1.7641,-3.1934,-3.6008,-4.3035,-2.7548,-0.33663,3.2367,2.5902,3.3463,1.4074,-0.089453,0.07827,0.7939,1.035,0.13434,-0
```

# Desenvolvimento de RNAs

## F – Teste em Campo: influência de outros fatores

### ➔ Estado da Placa

- Apagada
- Suja
- Amassada
- Enferrujada



### ➔ Luminosidade

- Falta de contraste
- Excesso de ruído
- Reflexo
- Sombra



### ➔ Enquadramento

- Pequena
- Torta
- Escondida





# Análise da Aplicação LAP-ZTW

---

- Maior esforço está concentrado no tratamento das imagens;
- Embora a teoria de RNAs seja complexa, aspectos práticos de implementação são facilitados com o uso de ferramentas como o Matlab (para o caso de redes/algoritmos típicos e básicos);
- A definição da rede é mais arte do que ciência: tentativa e erro (baseada em heurísticas).