Reconhecimento de Caracteres com Variações de Escala, Rotação e Translação

Luiz Eduardo Seabra Varella [1,2]
Emmanuel Piceses Lopes Passos [2]
Márcio Azevedo Santos [1]
Ricardo Lomba de Araujo [1]

[1] PETROBRÁS-Petróleo Brasileiro S.A.
Depex-Departamento de Exploração
Av. Chile 65, sala 1402
20035 Rio de Janeiro, RJ, Brasil
g073@c53000.petrobras.anrj.br

[2] IME-Instituto Militar de Engenharia
Seção de Sistemas e Computação/Cartografia
Praça General Tibúrcio, 80
22290 Rio de Janeiro, RJ, Brasil
pass@lncc.bitnet

Abstract: This paper presents a neural network based approach for distortion (translation, scale and rotation) invariant character recognition. To achieve invariancy, six distortion-invariant features are extracted from each image. Results of experimentation with different networks are also described.

1.0-Introdução

O desenvolvimento tecnológico dos últimos anos têm produzido avanços significativos nas diversas áreas da Ciência da Computação, em especial, na Computação Gráfica, no Processamento Digital de Imagens, no Reconhecimento de Padrões, e mais recentemente, nas Redes Neurais. A utilização de técnicas destas quatro áreas têm produzido efeitos de grande expressão na automatização de escritórios, na editoração eletrônica e no reconhecimento de caracteres. No que se refere a Sistemas de Informações Geográficas, nota-se um *gargalo* na aquisição de dados, pois a digitalização manual de documentos cartográficos é bastante lenta e a digitalização via *scanners* ainda exige um grande trabalho manual de edição dos dados. Neste aspecto, este trabalho apresenta uma contribuição significativa na redução deste tempo de edição, com um método de reconhecimento automático de caracteres que utiliza técnicas de Processamento Digital de Imagens e introduz as Redes Neurais como uma ferramenta para o Reconhecimento de Padrões.

O texto foi organizado de maneira a apresentar o preparo inicial da imagem, a fundamentação teórica para a descrição do modelo de redes destinado a classificação dos caracteres, e finalmente, os testes da implementação da solução.

2.0-Estrutura Geral da Solução

A estruturação apresentada em [Varella (1992)], no trabalho de Reconhecimento de Elementos Gráficos Digitalizados via *Scanners*, aborda as etapas: Pré-Processamento, Afinamento, Codificação, Classificação, Reconhecimento Geométrico e Reconhecimento Não Geométrico. Partindo de uma imagem binária, o Pré-Processamento consiste na eliminação de ruídos desta imagem, o Afinamento gera o esqueleto, a Codificação vetoriza os dados, a Classificação separa os elementos gráficos em geométricos e não geométricos, o Reconhecimento Geométrico gera curvas de isovalores, retas e poligonais e o Reconhecimento Não Geométrico codifica textos e símbolos.

De acordo com o enfoque deste trabalho o Pré-Processamento englobará as quatro primeiras etapas citadas, o Reconhecimento Geométrico não será abordado e o Não Geométrico será dividido em duas fases distintas : Extração de Características e Classificação da imagem. A Extração de Características se constitui numa função que mapeia um espaço de medidas (imagem) em um outro, denominado espaço de características, de dimensões reduzidas. O objetivo principal desta transformação não é apenas a redução da dimensionalidade do espaço de medidas, mas também a descoberta de

características principais que permitam a classificação da imagem. Isto é feito através do cálculo dos seis primeiros momentos invariantes da imagem. A Classificação é realizada via um sistema de redes neurais fundamentado no modelo *back-propagation*. Cada uma destas etapas será analisada nas seções subseqüentes.

3.0-Pré-Processamento

Com base na resolução do *Scanner* (espessura mínima das linhas e distância entre linhas), aplicam-se, inicialmente, os filtros conhecidos por Remoção de Ruídos e Preenchimento de Vazios. O passo seguinte é a Remoção de Pixels Isolados, a Reconstrução de Linhas Fragmentadas (Suavização Direcional), e por último a Suavização da Imagem (Suavização pela Maioria).

O Afinamento é realizado pelo método clássico, que se baseia na busca aos pixels de contorno, para posteriormente eliminar aqueles que não fazem parte do esqueleto da imagem.

A Codificação tem como objetivo armazenar apenas os pixels acesos, produzindo uma compactação inicial dos dados (vetorização) e eliminando as conectividades indesejáveis entre os elementos e temas gráficos.

Por fim, explorando-se atributos do tipo área, densidade de pixels acesos e razão altura/largura dos componentes conectados, tem-se a Classificação dos elementos em geométricos e não geométricos. Estes últimos é que são de interesse do trabalho.

4.0-Momentos Invariantes

O uso dos momentos invariantes para extração de características fundamentais é uma técnica que remonta a alguns anos, sendo introduzida inicialmente por [Hu (1962)]. Sua utilização neste trabalho foi motivada pelas publicações de [Dudani (1977)] e [Khotanzad (1988)].

Conceitua-se por momentos invariantes o conjunto de funções não lineares invariantes à escala, translação e rotação, obtidas a partir dos momentos geométricos da imagem. Dada uma imagem digital g(x,y) de dimensões M x M, $\{g(x,y), x,y = 0,\ldots M\text{-}1\}$, o (p+q)-ésimo momento geométrico é dado por:

$$m_{pq} = \sum_{x=0}^{M-1} \sum_{y=0}^{M-1} x^p y^q g(x,y) \quad \text{para p,q} = 0,1,2,\ldots .$$

Como são tratados caracteres de diferentes tamanhos, faz-se necessário a normalização do domínio de m_{pq}, que é feito mapeando-se a imagem plana M x M em um quadrado definido por $x \in [-1,+1]$ e $y \in [-1,+1]$.
Conseqüentemente, a definição de m_{pq} fica:

$$m_{pq} = \sum_{x=-1}^{+1} \sum_{y=-1}^{+1} x^p y^q g(x,y).$$

Fazendo os momentos invariantes à translação, têm-se a seguinte formulação para os momentos centrais da imagem:

$$\mu_{pq} = \sum_{x=-1}^{+1} \sum_{y=-1}^{+1} (x-\bar{x})^p (y-\bar{y})^q g(x,y), \text{ onde}$$

$$\bar{x} = \frac{m_{10}}{m_{00}} \quad \text{e} \quad \bar{y} = \frac{m_{01}}{m_{00}}.$$

Normalizando os momentos centrais a fim de torná-los invariantes à escala chega-se a:

$$\eta_{pq} = \frac{\mu_{pq}}{(\mu_{00})^{\gamma}}, \text{ onde } \gamma = \frac{p+q}{2} + 1.$$

O conjunto de seis funções não lineares, invariantes à escala, rotação e translação, é definido sobre η_{pq} da seguinte maneira:

$$\phi_1 = \eta_{20} + \eta_{02} ;$$

$$\phi_2 = (\eta_{20} - \eta_{02})^2 + 4(\eta_{11})^2 ;$$

$$\phi_3 = (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2 ;$$

$$\phi_4 = (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2 ;$$

$$\phi_5 = (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12}) [(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] + (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03}) [3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] ;$$

$$\phi_6 = (\eta_{20} - \eta_{02}) [(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{03} + \eta_{21}) .$$

Como os valores de ϕ_1 a ϕ_6 são extremamente pequenos, é comum, ao final do cálculo dos momentos, aplicar-se a função logarítmica a fim de se evitar problemas de precisão numérica. Logo, as características extraídas dos caracteres são definidas por: $\log |\phi_i|$, $i = 1,\ldots,6$.

Vale ressaltar que as propriedades de invariância dos momentos apenas se confirmam quando aplicados em funções contínuas. Para o

caso de imagens digitais os resultados mostram variações expressivas, especialmente, quando se efetuam rotações maiores que 45 graus. A Tabela 1 ilustra o fato, apresentando os valores obtidos para os dois primeiros caracteres da Figura 1, inclinados de 0, 45 e 90 graus, onde MI = $-\log|\phi_i|$. Apesar desta limitação, os momentos funcionam, para o caso em estudo, como razoáveis extratores de características. A Figura 2 mostra a distribuição bi-dimensional de 36 caracteres horizontais *plotados* segundo os dois primeiros momentos.

5.0-Conceitos de *Backpropagation*

A regra de aprendizado mais difundida é a *back-propagation*, e a arquitetura em camadas, para a qual *backpropagation* se aplica, é sem dúvida o modelo de rede neural mais empregado na atualidade. Aparentemente, *backpropagation* foi desenvolvida independentemente na década de 80 por [Rumelhart-Hinton-Williams (1986)] do grupo PDP, e por [Parker-Le Cun (1989)]. Todavia, descobriu-se recentemente que um trabalho de [Bryson-Ho (1969)] e outro de [Werbos (1974)] já apresentavam versões deste algoritmo. Ao que tudo indica, estas descobertas se deram independentemente.

A importância de *backpropagation* reside no fato de que redes com unidades binárias com função limiar do tipo Perceptron, de apenas uma camada de pesos são capazes apenas de implementar funções linearmente separáveis. Isto como se sabe é uma grande limitação, já que funções simples, do tipo ou exclusivo, não podem ser obtidas. Redes com múltiplas camadas são capazes de implementar funções multidimensionais bastante gerais, e o algoritmo de *backpropagation* permite que tais redes aprendam a executar este tipo de mapeamento.

É extremamente conveniente, neste ponto do trabalho, que se defina o que vem a ser *backpropagation*. Vários autores referem-se a *backpropagation* como sendo uma regra de aprendizado, implementada em arquiteturas do tipo Perceptron com múltiplas camadas. No entanto, a título de simplificação e até mesmo visando enfatizar a regra de aprendizado, alguns autores definem *backpropagation* como sendo um modelo de rede neural, composto de uma arquitetura própria e aprendizado *backpropagation*.

Tabela 1 : Momentos Invariantes.

-	B/0	B/45	B/90	F/0	F/45	F/90
M1	6.56	6.58	6.55	6.25	6.42	6.21
M2	14.96	14.95	15.09	13.27	13.55	13.35
M3	24.67	23.29	19.42	22.18	18.79	17.16
M4	25.56	25.30	21.58	20.97	20.94	18.95
M5	50.68	50.42	42.19	42.94	41.42	37.09
M6	33.43	32.96	29.19	28.54	30.05	25.67

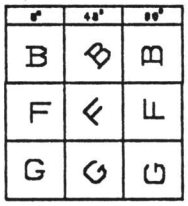

Figura 1: Caracteres para o Cálculo dos Momentos.

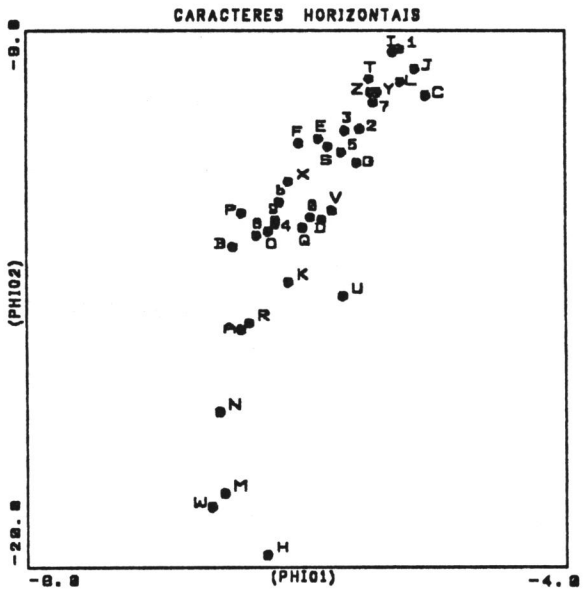

Figura 2: Distribuição de Caracteres Segundo os Momentos.

5.1-Arquitetura do Modelo

A arquitetura mais elementar do modelo é composta de três camadas: entrada, interna e saída. Cada camada é totalmente conectada com as camadas anterior e posterior. Deste modo, cada unidade da camada de entrada envia sua saída para todas as unidades da camada intermediária e cada

unidade da camada intermediária se liga com todas as unidades da camada de saída. Dentro de uma mesma camada, em geral, não é feita nenhuma conexão.

5.2-Funcionamento da Rede

Antes de se iniciar o processo de treinamento, os pesos das conexões devem ser fixados aleatoriamente, excetuando-se os pesos das conexões com o mundo exterior que são fixados em + 1. O processo de aleatorização deve garantir que os pesos tenham valores iniciais distintos.

O aprendizado neste tipo de rede se dá no modo supervisionado, fornecendo-se pares de entrada e saída. Para cada padrão de entrada, os pesos são ajustados de acordo com a regra de treinamento.

O valor de propagação para todas as unidades da rede é dado pela soma ponderada das entradas das unidades.

Uma vez apresentada à rede um estímulo, a camada de entrada o transmite diretamente para todas as unidades da camada intermediária, o que significa dizer que a função de ativação e a função de saída da camada de entrada correspondem à função identidade. As funções de ativação das camadas interna e saída são do tipo sigmoidal, contínua, na forma de S, monotonicamente crescente e assintótica para valores fixos, à medida que a entrada se aproxima de mais ou menos infinito. Normalmente, o limite superior da função é estabelecido em + 1 e o limite inferior em -1 ou 0. Um exemplo desta função é dada pela expressão abaixo, onde p é o valor de propagagação da unidade.

$$A(p) = \frac{1}{(1 + e^{-p})}$$

As funções de saída das camadas interna e de saída geralmente são a função identidade. Depois de processados os sinais de entrada da camada interna os sinais produzidos são propagados para a camada de saída, que os processam, gerando a resposta para o estímulo de entrada da rede.

Durante o aprendizado, o sinal de saída da rede é comparado com o sinal desejado. Existindo diferença, o erro é retropropagado, ajustando-se os pesos das conexões segundo a regra de aprendizado *backpropagation*, como será visto a seguir.

5.3-Aprendizado *Backpropagation*

O algoritmo *backpropagation* consiste numa generalização da regra delta (por esta razão alguns autores o chamam de Regra Delta Generalizada). A regra Delta não é aplicável em redes de múltiplas camadas por duas razões. A primeira, é que ela normalmente emprega unidades de saídas lineares. No entanto, qualquer rede com apenas unidades lineares possui uma rede equivalente com apenas uma camada de pesos. Desta forma, não faz sentido, a princípio, utilizar redes com várias camadas de unidades lineares. Um motivo mais importante se deve ao mecanismo de operação do algoritmo. Na regra Delta, os pesos são ajustados de acordo com as diferenças entre os valores obtidos e os desejados. Numa rede de múltiplas camadas, os valores das unidades intermediárias não são conhecidos, já que no aprendizado supervisionado tradicional apenas os padrões de entrada e saída são fornecidos. O problema é que, apesar de ser possível definir uma função erro do sistema, não é, a princípio, claro, determinar como cada conexão contribui para o erro, uma vez que existem várias conexões intermediárias e os erros podem se somar. Como não se sabe o que as unidades intermediárias devem fazer, não é imediato computar o sinal de erro para estas unidades. Como, então, ajustar os pesos das ligações para minimizar o erro do sistema? *Backpropagation* fornece um mecanismo para solucionar o problema.

Supondo uma rede com três camadas, vide Figura 3, e aplicando-se um estímulo de entrada X, obtêm-se na camada de saída um vetor O que corresponde a resposta da rede a este estímulo de entrada. O problema consiste em se obter uma aproximação $O = \phi'(X)$ para $Y = \phi(X)$, onde Y é o valor de saída desejado. Como o problema não é de natureza linear e se apresenta de forma multidimensional, em geral, utiliza-se uma versão iterativa do método do erro médio quadrático, também conhecido como método do gradiente descendente.

O estímulo introduzido na rede é propagado para a camada intermediária, cuja função de propagação é dada por:

$$P_j = \sum_{i=1}^{N} W_{ji} X_i + \theta_j \; ; \qquad (1)$$

onde P_j é o valor de propagação do elemento j da camada intermediária, W_{ji} corresponde ao peso da conexão entre o elemento i da camada de entrada e o elemento j da camada intermediária, X_i o estímulo de entrada do i-ésimo elemento e θ_j é a parcela do *bias* que atua sobre o elemento j. Como a função de saída da camada intermediária é a identidade, o valor de saída para o j-ésimo elemento intermediário é dado por:

$$A_j = g_j(P_j) \,, \tag{2}$$

onde g_j é a função de ativação do neurônio j.

Analogamente, para um elemento de processamento K da camada de saída, têm-se a seguinte formulação para a função de propagação:

$$P_k = \sum_{j=1}^{L} W_{kj} A_j + \theta_k \,. \tag{3}$$

Como a função de saída da camada de saída também é a identidade, o sinal produzido pelo k-ésimo elemento, em termos de sua função de ativação, é dado pela seguinte expressão:

$$A_k = g_k(P_k) \,. \tag{4}$$

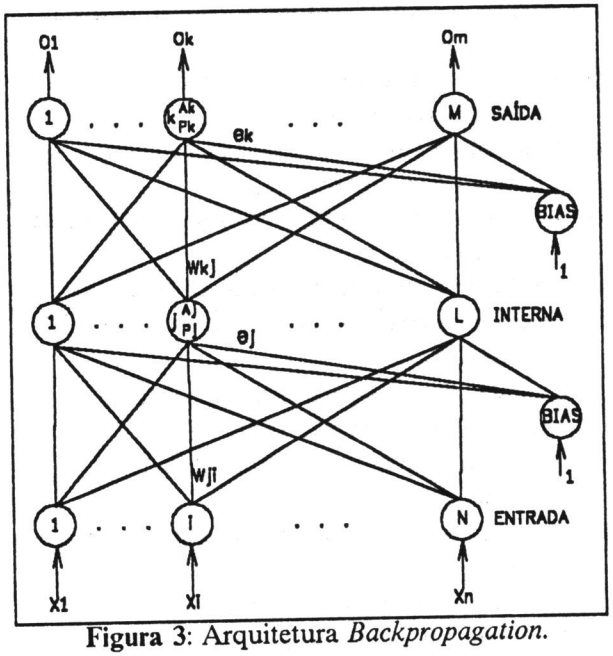

Figura 3: Arquitetura *Backpropagation*.

5.3.1-Atualização dos Pesos da Camada de Saída

Seja $\delta_k = (Y_k - O_k)$ o erro do k-ésimo elemento da camada de saída. O problema consiste em se minimizar o erro total, E_p, via soma dos quadrados dos erros de cada elemento de saída:

$$E_p = \frac{1}{2} \sum_{k=1}^{M} \delta_k^2 \,. \tag{5}$$

O cálculo é feito segundo o gradiente negativo, $-\nabla E_p$, com respeito aos pesos W_{kj}, de maneira a se buscar o ponto de mínimo da superfície de erro. Tomando separadamente cada componente de E_p, tem-se:

$$E_{pk} = \frac{1}{2} (Y_k - O_k)^2 \,, \text{ e} \tag{6}$$

$$\frac{\partial E_{pk}}{\partial W_{kj}} = -(Y_k - O_k) \frac{\partial [g_k(P_k)]}{\partial P_k} \frac{\partial P_k}{\partial W_{kj}} \,, \tag{7}$$

onde $\dfrac{\partial [g_k(P_k)]}{\partial P_k} = g'_k(P_k)$. Sabendo que $\dfrac{\partial P_k}{\partial W_{kj}} = A_j$, chega-se a:

$$-\frac{\partial E_{pk}}{\partial W_{kj}} = (Y_k - O_k) g'_k A_j \,. \tag{8}$$

Como a regra tenta garantir que o erro quadrático médio seja minimizado na rede, os pesos são ajustados adicionando-se ao vetor peso corrente o vetor delta, dado pela expressão (8). Formalizando a atualização do vetor peso corrente em função de um dado instante de treinamento, tem-se:

$$W_{kj}(t+1) = W_{kj}(t) + \Delta W_{kj}(t) \,, \text{ onde} \tag{9}$$

$$\Delta W_{kj}(t) = \eta \, (Y_k - O_k) \, g'_k \, A_j \,. \tag{10}$$

A constante η que aparece na expressão, também conhecida como constante de aprendizado, tem por objetivo agilizar a convergência da rede. Seu valor deve ser positivo de modo a direcionar o vetor delta para o vetor ideal. Deve ser menor do que 1 para garantir a estabilidade da rede, pois do contrário, o ajuste dos pesos pode levar o vetor delta a um valor superior ao da entrada, levando a resultante para a direção errada. O valor desta constante é profundamente dependente dos dados de entrada, devendo ser escolhida de acordo com a aplicação. De modo geral, valores entre 0.8 e 0.9 garantem rapidez de convergência, podendo levar a

expressivas oscilações em torno do ponto de mínimo. Os valores na faixa de 0.1 a 0.2 evitam grandes oscilações, mas podem levar a uma convergência demorada.

Sabendo que $g'_k(P_k)$ corresponde à derivada da função de ativação, do tipo sigmoidal ($g_k(P_k) = (1 + e^{-P_k})^{-1}, g'_k(P_k) = g_k(1 - g_k)$), das camadas intermediária e de saída, e fazendo $\delta_k^o = (Y_k - O_k)\, g'_k(P_k) = \delta_k\, g'_k(P_k)$, pode-se escrever a equação de ajuste dos pesos da camada de saída independente da forma da função de ativação:

$$W_{kj}(t+1) = W_{kj}(t) + \eta\, \delta_k^o\, A_j\,. \qquad (11)$$

5.3.2-Atualização dos Pesos da Camada Intermediária

Utilizando a equação de E_p, conclui-se que o erro total encontra-se relacionado com os valores de saída da camada intermediária,

$$\begin{aligned} E_p &= \frac{1}{2}\sum_k (Y_k - O_k)^2 \\ &= \frac{1}{2}\sum_k (Y_k - g_k(P_k))^2 \\ &= \frac{1}{2}\sum_k \left(Y_k - g_k\left(\sum_j W_{kj}A_j + \theta_k\right)\right)^2 , \end{aligned} \qquad (12)$$

uma vez que A_j depende dos pesos desta camada. Explorando este fato para o cálculo do gradiente de E_p em relação aos pesos da camada intermediária, chega-se a:

$$\begin{aligned} \frac{\partial E_p}{\partial W_{ji}} &= \frac{1}{2}\sum_k \frac{\partial}{\partial W_{ji}} (Y_k - O_k)^2 \\ &= -\sum_k (Y_k - O_k) \frac{\partial O_k}{\partial P_k} \frac{\partial P_k}{\partial A_j} \frac{\partial A_j}{\partial P_j} \frac{\partial P_j}{\partial W_{ji}}\,. \end{aligned} \qquad (13)$$

Escrevendo a equação com os termos definidos anteriormente, tem-se:

$$-\frac{\partial E_p}{\partial W_{ji}} = \sum_k (Y_k - O_k)\, g'_k(P_k)\, W_{kj}\, g'_j(P_j)\, X_i\,. \qquad (14)$$

Logo, a atualização dos pesos segundo o gradiente negativo é dado por:

$$\begin{aligned} \Delta W_{ji} &= \eta\, g'_j(P_j)\, X_i \sum_k (Y_k - O_k) g'_k(P_k)\, W_{kj} \\ &= \eta\, g'_j(P_j)\, X_i \sum_k \delta_k^o\, W_{kj}\,. \end{aligned} \qquad (15)$$

Analisando a expressão de atualização dos pesos da camada intermediária, conclui-se que o valor a ser adicionado ao valor corrente depende fortemente de δ_k^o da camada de saída. Em decorrência deste fato, ou seja, a retropropagação do erro, é que advém o nome *backpropagation.*

De maneira análoga ao que foi feito para se expressar a atualização dos pesos da camada de saída em termos de δ_k^o, define-se $\delta_j^h = g'_j(P_j) \sum_k \delta_k^o W_{kj}$. Conseqüentemente, a equação final de atualização dos pesos da camada intermediária fica:

$$W_{ji}(t+1) = W_{ji}(t) + \eta\, \delta_j^h\, X_i\,. \qquad (16)$$

O algoritmo *backpropagation*, assim como vários outros, pode apresentar problemas de convergência quando atingir um mínimo local. Para reduzir este problema e acelerar a convergência, adiciona-se nas equações de atualização dos pesos o termo: $\alpha(W_{kj}(t) - W_{kj}(t-1))$, que na verdade atua como a "lembrança" da rede, forçando a atualização corrente para a direção anterior.

5.3.3-Algoritmo de Aprendizado

a - Aplicar o vetor X em todos os elementos da camada de entrada.
b - Calcular os valores de propagação da camada intermediária, equação (1).
c - Calcular os valores de saída da camada intermediária, equação (2).
d - Calcular os valores de propagação da camada de saída, equação (3).
e - Calcular as respectivas saídas, equação (4).
f - Calcular os erros para cada elemento de saída, equação (8).
g - Calcular os erros para cada elemento intermediário, equação (15).
h - Atualizar os pesos da camada de saída, equação (11).
i - Atualizar os pesos da camada intermediária, equação (16).
j - Calcular o erro total, equação (5).
k - Se o erro for aceitável encerrar, senão passar para outro par de treinamento, retornando ao passo a.

6.0-Descrição do Sistema de Classificação

Antes de se descrever a solução, é de grande importância que se analise alguns aspectos. O primeiro deles refere-se à geometria e disposição dos padrões. Os textos de um documento gráfico apresentam-se de diversas formas, tamanhos e inclinações, portanto, deve-se chegar a uma solução que contemple tais características. O segundo diz respeito às espessuras. Vale lembrar que a imagem encontra-se representada por seu esqueleto, com todos os elementos apresentando espessuras unitárias. Esta particularidade deve ser levada em conta, uma vez que, a esqueletização da imagem produz sensíveis alterações na forma dos caracteres com inclinações não múltiplas de 45 graus. O terceiro, e último, refere-se à similaridade dos padrões. É importante que se desenvolva uma técnica que discrimine adequadamente caracteres semelhantes, por exemplo, I do 1, O do 0, B do 8.

Como em quase todos os sistemas neurais, baseados no modelo *Backpropagation*, faz-se necessário a realização de um conjunto de testes para se chegar a uma arquitetura definitiva. A título de simplificação para a implementação dos testes, foi digitalizado apenas um fonte de tamanho 0.4 cm, constituído de 36 caracteres. Para cada caracter foram geradas 13 imagens com inclinações múltiplas de 15 graus, variando de -90 à 90 graus, vide Figura 4. A seguir são descritos os testes realizados em um computador IBM/3090.

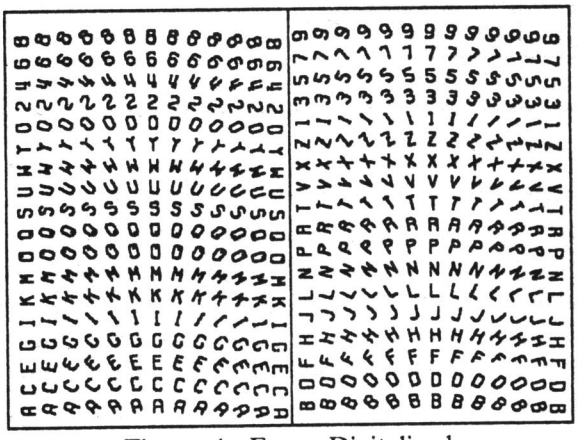

Figura 4 : Fonte Digitalizado.

6.1-Teste 1

O primeiro sistema que se pensou foi de apenas uma rede *Backpropagation* com seis elementos de processamento na camada de entrada, um elemento para cada momento invariante, e trinta e seis elementos na camada de saída, cada elemento respondendo por um caracter, vide Figura 5. Com intuito de se estimar um número inicial para as unidades de processamento da camada intermediária, foi treinado um conjunto de redes para o reconhecimento de caracteres com inclinação igual a 0 graus. Para este teste foram experimentadas 22 arquiteturas de redes, treinadas 20000 vezes com os padrões a 0 graus.

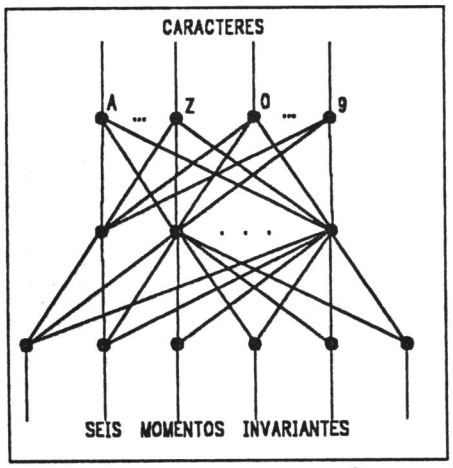

Figura 5 : Teste 1 - Rede Única.

A tabela 2 mostra para cada número de elementos da camada intermediária o respectivo percentual de reconhecimento dos padrões treinados. Foi adotada para a taxa de aprendizado, η, o valor 0.2 e para a constante *momentum*, α, o valor de 0.9. A realização deste teste consumiu aproximadamente 2h 10min de CPU.

Tabela 2 : Resultados do Teste 1.

Unid. Inter.	% Rec.	Unid. Inter.	% Rec.	Unid. Inter.	% Rec.
10	2.56	108	61.53	216	71.79
16	2.56	120	66.66	230	82.05
32	10.25	130	79.48	250*	97.43*
40	7.69	140	71.79	260	87.17
56	25.64	150	74.35	280	84.61
68	30.76	170	79.48	300	79.48
80	51.28	186	71.79	-	-
92	48.71	200	82.05	-	-

De posse desse resultado, foram treinadas três redes com 250, 270 e 300 unidades de processamento na camada intermediária, visando o reconhecimento dos caracteres com todas as inclinações. Do conjunto de treinamento, 468 amostras, Figura 4, foram eliminados os dois W da extremidade (-90 e 90 graus) por serem idênticos aos padrões da letra M (90 e -90 graus). O treinamento destas redes consumiu mais de 32h de CPU, sem contudo apresentar resultados que justificassem o aprimoramento da solução. Este longo tempo de treinamento, sem dúvida, inviabiliza qualquer solução neste sentido.

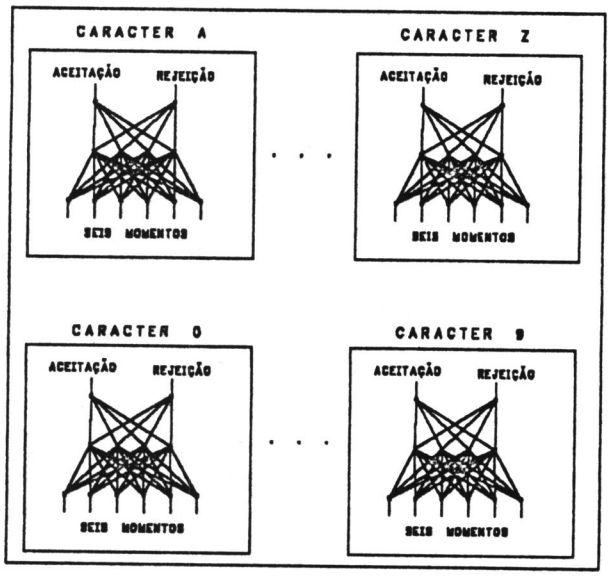

Figura 6: Teste 2 - 36 Redes.

6.2-Teste 2

Uma segunda alternativa foi, então, a implementação de 36 redes, uma para cada caracter, com seis elementos de processamento na camada de entrada (seis momentos invariantes) e dois elementos na camada de saída, indicando a aceitação ou rejeição do padrão de entrada, vide Figura 6. Segundo esta arquitetura, o reconhecimento é feito aplicando-se os seis momentos invariantes do padrão em todas as redes. Conseqüentemente, a classificação é definida pela rede que apresentar o maior valor de ativação do elemento de saída responsável pela aceitação. Para o teste, as 36 redes foram treinadas com 466 amostras, sendo apresentadas 5000 vezes. A taxa de aprendizado (0.2) e o *momentum* (0.9) foram os mesmos do teste anterior. Para cada uma das redes foi pesquisado o melhor número de elementos da camada intermediária.

Tabela 3 : Resultados do Teste 2.

Rede	Un. Int.	% Ac.	% Rej.	Rede	Un. Int.	% Ac.	% Rej.
A	8	100	99	S	8	84	99
B	8	100	98	T	8	76	92
C	8	100	100	U	8	100	100
D	24	100	97	V	8	84	98
E	9	76	100	W	8	100	99
F	9	100	98	X	8	100	98
G	9	100	100	Y	24	76	96
H	8	100	99	Z	8	100	99
I	8	100	98	0	8	100	98
J	9	53	96	1	8	100	98
K	9	100	99	2	8	100	98
L	8	38	94	3	8	92	98
M	8	100	98	4	10	46	99
N	8	100	100	5	8	100	98
O	10	84	98	6	8	92	99
P	8	84	98	7	10	92	83
Q	8	84	95	8	8	84	99
R	8	100	100	9	8	61	100

A tabela 3 mostra o resumo do teste, apresentando, para cada rede, o número de unidades escondidas e os percentuais *aproximados* de reconhecimento, indicados pelos escores de aceitação e rejeição dos padrões. Nesta tabela, os escores de aceitação e rejeição não expressam o resultado global do reconhecimento. Estes valores indicam que, por exemplo, para a rede que corresponde a letra A, 100% dos padrões A são reconhecidos e 99% dos demais padrões são ditos não A. Este índice de rejeição indica que existem alguns padrões não A que são classificados como A. Como apenas quatro redes apresentaram os percentuais ideais, ou seja, 100% para aceitação e 100% para rejeição, e algumas delas mostraram

valores inferiores a 70%, é de se esperar que o resultado final aponte para a busca de uma outra solução. Apesar disso, 68% das amostras treinadas foram reconhecidas e o tempo de treinamento das 36 redes foi bem inferior (1h 25min) ao do teste de uma única rede, mostrando que é mais rápido treinar um conjunto de pequenas redes do que apenas uma com elevado número de elementos de processamento.

6.3-Teste 3

Continuando nesta linha, com o propósito de se aumentar o percentual de reconhecimento, foi treinado um conjunto de 466 redes (36 caracteres * 13 inclinações de 15 em 15 graus - 2 W), com arquiteturas idênticas: seis elementos de processamento na camada de entrada, quatro elementos na camada intermediária e dois na camada de saída. Cada uma destas redes é responsável pelo reconhecimento de apenas uma inclinação de um dado caracter. A Figura 7 mostra o esquema de arquiteturas do sistema neural.

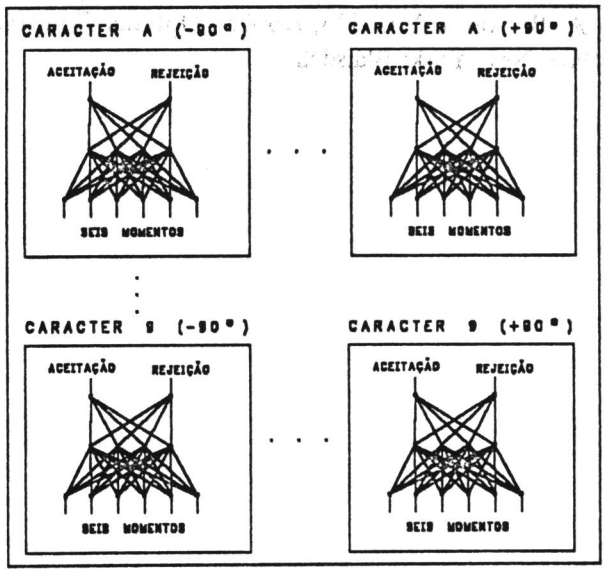

Figura 7: Teste 3 - 466 Redes.

Quanto à estratégia de reconhecimento, o sistema segue o mesmo procedimento do teste anterior, ou seja, aplicam-se os momentos invariantes em todas as redes, e aquela que apresentar o maior valor de ativação no elemento de aceitação do padrão, classificará o caracter. O treinamento deste sistema consumiu 6h 25min de CPU. Para cada rede foram apresentadas 5000 vezes o conjunto de amostras, com a taxa de aprendizado igual a 0.2 e *momentum* igual a 0.9.

O resultado deste modelo, com 466 redes, mostrou avanços significativos, reconhecendo 97% dos padrões treinados. De maneira a se medir a capacidade de generalização do sistema, isto é, o reconhecimento de padrões não treinados, foram realizados outros testes descritos a seguir.

6.4-Teste de Generalização

Foi então gerada uma massa de padrões com caracteres rotacionados de 7.5 graus a partir dos padrões treinados. Para este conjunto de padrões, "não treinados", o sistema respondeu reconhecendo 41% dos caracteres. Este índice deveu-se ao elevado intervalo de rotação aplicado em cada caracter (15 graus) do conjunto de treinamento. De maneira análoga, foi realizado um segundo teste de generalização com intervalos de 10 graus para o treinamento e de 5 graus para o reconhecimento. A generalização deste modelo atingiu percentuais da ordem de 59%. Estima-se que com intervalos de 5 graus para o treinamento, a generalização atinja valores superiores a 80%. A tabela 4 sintetiza os resultados destes testes de reconhecimento dos padrões não treinados, mostrando para cada intervalo de rotação os respectivos percentuais de generalização.

Tabela 4 : Teste de Generalização.

Int. Trein.	Int. Gener.	% de Rec.
15°	7.5°	41
10°	5°	59
$5°_{\times}$	$2.5°_{\times}$	> 80

Com relação ao reconhecimento de caracteres de diferentes tamanhos (escala), os percentuais se mantiveram no mesmo nível para variações até a metade do padrão treinado. Isto significa dizer que para o fonte utilizado nos testes, de tamanho igual a 0.4cm, pode-se reconhecer caracteres que variam de 0.2 a 0.6cm.

7.0-Conclusões

Concluindo, o método aqui proposto para o reconhecimento de caracteres com variações de

escala, rotação e translação se dá através de um conjunto de pequenas redes neurais, do tipo *Back-propagation*, onde cada uma delas é responsável pela classificação de uma única posição do caracter. Todas as redes são constituídas de seis elementos de processamento na camada de entrada, quatro elementos na camada intermediária e dois elementos na camada de saída.

É importante ressaltar que a acuracidade do reconhecimento é função da similaridade do conjunto de caracteres, da deformação produzida pela esqueletização, da resolução do dispositivo de aquisição e da capacidade de generalização introduzida (quantidade de redes), o que nos dá a certeza de que resultados ainda melhores podem ser obtidos.

Referências

- L.E.S. Varella, Reconhecedor de Elementos Gráficos Digitalizados via Scanners, Dissertação de Mestrado, Instituto Militar de Engenharia, RJ, 1992.
- M. Hu, Visual Pattern Recognition by Moment Invariants, IRE Trans. Inform. Theory, 1962, Vol.IT-8, PP.179-187.
- S.A. Dudani "et alli", Aircraft Identification by Moment Invariants, IEEE Trans. on Computers, 1977, Vol.C-26, Num.1, PP.39-46.
- A. Khotanzad & J. Lu, Distortion Invariant Character Recognition by a Multi-Layer Perceptron and Backpropagation Learning, IEEE International Conference on Neural Networks, 1988, Vol.1, PP.625-632.
- P.K. Simpson, Artificial Neural Systems: Foundations, Paradigms, Applications, and Implementations, Pergamon Press, 1990.
- Y. Le Cun "et alli", Handwritten Digit Recognition Applications of Neural Chips and Automatic Learning, IEEE Commun. Mag., 1989, Vol. 27, Num. 11, PP. 41-46.
- D.E. Rumelhart, G.E. Hinton & R.J. (1986 b), Learning Internal Representations by Error Propagation, em D.E. Rumelhart & J.L. McClelland, Parallel Distributed Processing, 1988, Vol. 1, Cambridge, M.A., The MIT Press.
- P. Werbos, Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences, 1974, Ph.D. Dissertation, Harvard University.
- A. Bryson & Y.C. Ho, Applied Optimal Control, 1969, New York: Blaisdell.