ANÁLISE DE DESEMPENHO DE UM CLASSIFICADOR NEURAL DE ATRIBUTOS DE TEXTURA EM AMBIENTE DISTRIBUÍDO

Kelly Maria Rangel[1], Lamartine Nogueira Frutuoso Guimarães [2]

[1]Instituto Nacional de Pesquisas Espaciais/Laboratórios Associados de Computação e Matemática Aplicada – INPE/LAC, São José dos Campos-SP, kelly.rangel@lac.inpe.br
[2]Instituto de Estudos Avançados/Divisão de Energia Nuclear – IEAv/ENU, São José dos Campos - SP, guimarae@ieav.cta.br

Resumo - Este artigo apresenta alguns resultados do trabalho desenvolvido para melhoria de performance do "Classificador Neural de Atributos de Textura para Busca e Recuperação de Imagens de Sensoriamento Remoto" (Martins, 2003). A implementação em ambiente distribuído do Classificador Neural foi analisada, focando a distribuição do treinamento do Mapa Auto-Organizável de Kohonen, implementado na versão seqüencial do Classificador Neural. Para a implementação distribuída, foram analisadas algumas bibliotecas de troca de mensagens implementadas em Matlab. E, para a avaliação de desempenho em ambiente distribuído foi utilizado o Toolbox ParMatlab, de troca de mensagens, para prover a comunicação entre os nós durante o treinamento da Rede Neural. O speedup e a eficiência encontrados foram satisfatórios para o caso de distribuição de processamento do treinamento do Mapa Auto-Organizável de Kohonen no sistema distribuído semi-dedicado montado no IEAv.

Palavras-chave: Processamento Distribuído e Redes Neurais Artificiais.
Área do Conhecimento: I – Ciências Exatas e da Terra

Introdução

Com o aumento do número de imagens de sensoriamento remoto coletadas diariamente por diferentes sensores, há uma crescente necessidade de se aperfeiçoar o gerenciamento desses dados, envolvendo, entre outras coisas, a constante melhoria de desempenho dos softwares de busca, recuperação e classificação destas imagens.

Neste trabalho são aplicadas determinadas técnicas de processamento distribuído no classificador neural construído por Maurício Pozzobon Martins[1], de modo a aumentar a eficiência através da distribuição do processamento, podendo assim aumentar o número de tarefas realizáveis através do compartilhamento de recursos, aumentar a confiabilidade e diminuir a interferência entre tarefas evitando interações desnecessárias.

A versão seqüencial do Classificador Neural foi desenvolvida em MATLAB e foi desenvolvidas uma rede de natureza supervisionada (*Learning Vector Quantization - LVQ*) e outra não supervisionada (*Self Organizing Map - SOM*).

Durante o desenvolvimento do trabalho foram analisadas algumas bibliotecas de trocas de mensagens desenvolvidas em MATLAB. O intuito principal foi analisar a portabilidade do Classificador Neural de Martins para um ambiente distribuído afim de que posteriormente fosse portado para o Cluster de Computadores, denominado *BELIEVe II* do Laboratório de Engenharia Virtual do Instituto de Estudos Avançados (LEV/IEAv/CTA). Para a versão distribuída do Classificador Neural [4], foi mantido o desenvolvimento em MATLAB utilizando um biblioteca de troca de mensagens denominado ParMatlab [Cetto].

O Classificador Neural de Atributos de Textura para Busca e Recuperação de Imagens de Sensoriamento Remoto (Martins, 2003).

O Classificador Neural de Martins é um sistema de busca de informações em imagens de sensoriamento remoto baseado em modelos visuais, esse tipo de modelo oferece uma forma eficiente de representar e transmitir informações em relação ao paradigma baseado em texto [6]. Os modelos de redes neurais implementados neste Classificador Neural são: Algoritmo do tipo M*apa Auto-Organizável de Kohonen,* conhecidos como *Self-Organizing Map – SOM,* com implementação da sua consciência, que penaliza o neurônio que vence freqüentemente; e Algoritmos de aprendizagem por quantização vetorial, conhecidos como *Learning Vector Quantization – LVQ*. Estes modelos foram implementados em MATLAB, para máquinas de processamento ordinário, e para o treinamento destas redes foi utilizado um conjunto de texturas, conhecido como Álbum Broadtz. Este Classificador Neural pode ser

utilizado no desenvolvimento de sistemas para a administração de grandes coleções de imagens de sensoriamento remoto e é baseado na metodologia utilizada por Manjunath e Ma (1996).

Implementação em Ambiente Distribuído

Foi utilizado o toolbox ParMatlab 1.77, desenvolvido por Lucio Andrade Cetto. Este toolbox utiliza o protocolo TCP/IP para comunicação, e é constituído de scripts Matlab. E para esta abordagem utilizamos um ambiente distribuído semi-dedicado, com máquinas heterogêneas e sistema operacional Windows.

A análise de desempenho do classificador neural em máquinas monoprocessadas apontou dois grandes blocos de processamento como pontos críticos: A extração dos atributos de textura das imagens e o treinamento da rede Kohonen. A extração dos atributos de textura das imagens necessita ser feito apenas uma vez para cada novo grupo de texturas, por isso o foco desta análise foi à distribuição do processamento de treinamento da rede Kohonen.

Para realizar a análise da distribuição de processamento do classificador foi necessário levar em consideração principalmente a arquitetura do ambiente distribuído utilizado. Assim, foi necessária a formalização de algumas definições desta área. A Figura 1 tem por finalidade mostrar a definição de granularidade em processamento paralelo e distribuído.

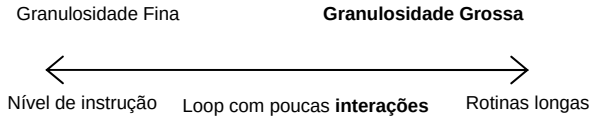

Figura 1 - Granularidade de Processamento

Para a implementação no ambiente de processamento distribuído semi-dedicado, foram feitas modificações no módulo de treinamento da rede Kohonen, de modo que esse treinamento da rede Kohonen fosse solicitado em blocos às máquinas "worker", sendo informados os parâmetros de entrada necessários para o treinamento; e, terminado o treinamento, a máquina worker enviaria à máquina "master" os resultados do treinamento.

Conforme a definição para granularidade em processamento paralelo ou distribuído utilizada neste trabalho, essa implementação do treinamento da rede de Kohonen pode ser classificada como *processamento de grão grosso*, e pode-se levar em consideração que a principal limitação para que o processamento utilizado fosse de média para fina foi o fato de que as redes Kohonen são redes auto-ealimentadas, e isso ocasiona grande dependência de dados.

Foi necessário também retirar a interface gráfica deste módulo, para melhorar o desempenho nas máquinas "worker". As Figuras 2 e 3 possuem os fluxogramas de processamento do treinamento do Mapa Auto-Organizável de Kohonen – SOM para uma máquina monoprocessada e ambiente de processamento distribuído.

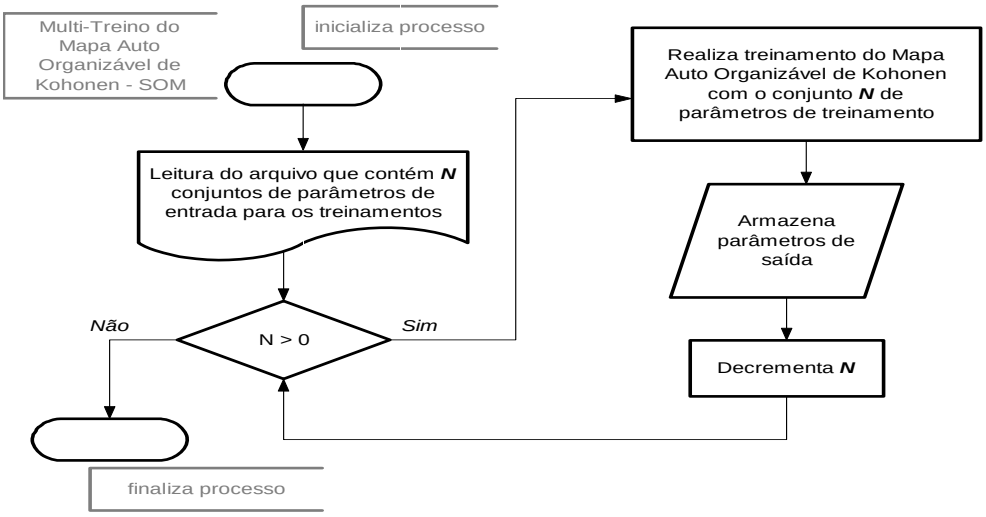

Figura 2 - Fluxograma de processamento seqüencial

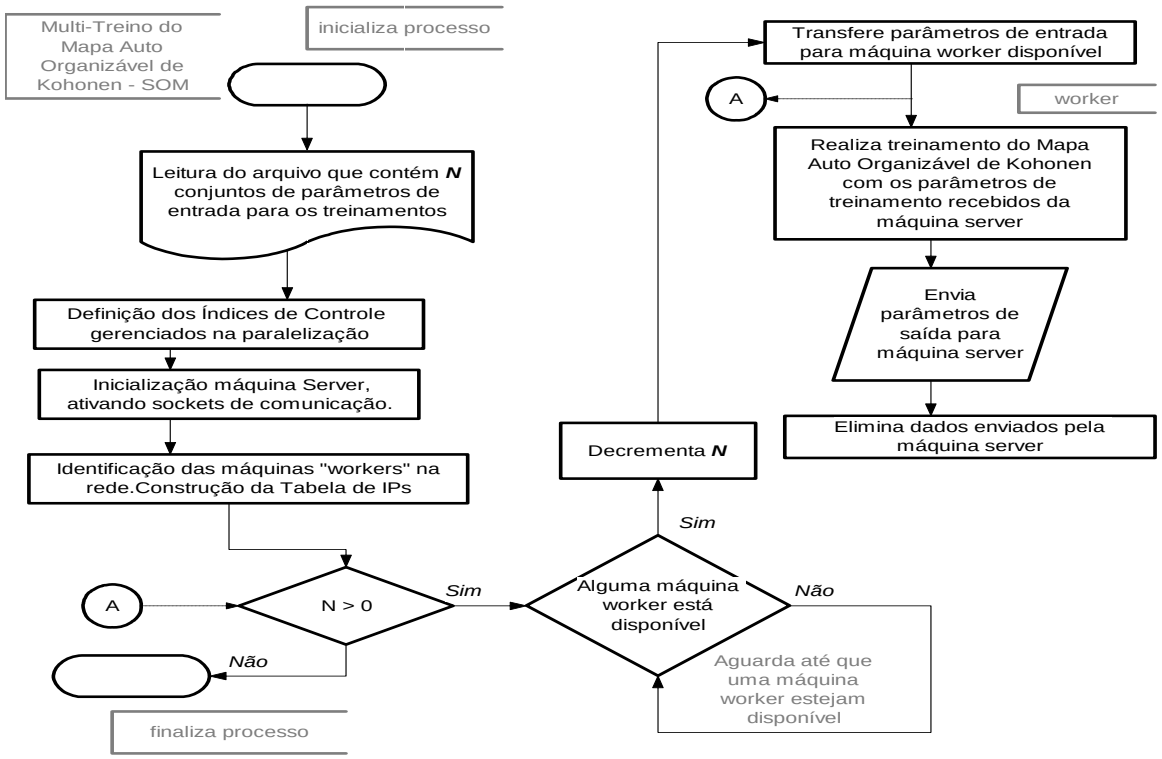

Figura 3 - Fluxograma de processamento distribuído

Adequação do Toolbox ParMatlab para o ambiente distribuído utilizado.

Algumas modificações foram necessárias para adaptar o funcionamento do "Toolbox" ParMatlab ambiente de processamento distribuído utilizado no IEAv/CTA:

A função *Getmyip.m* retorna o endereço IP da máquina. O código original está preparado para identificar a seguinte string: address ip. No caso de sistema operacional em português esta string deve ser modificada para endereço ip ou endere□o ip se não houver suporte para caracteres acentuados. O autor do "Toolbox" ParMatlab sugere que cada usuário modifique esta string de acordo com sua necessidade.

A função *worker.m* ativa a máquina escrava ou "worker", estabelecendo uma porta de comunicação diretamente com a máquina que gerencia a paralelização (máquina Master), recebe um nome de função com seus respectivos parâmetros, na seqüência, monta uma string de comando para essa função e executa-a em ambiente Matlab local (no "worker").No caso de transferência de arquivos entre a máquina Master e a máquina "worker" algumas modificações foram necessárias. No código modificado, se a função a ser executada for tcpip_getfile o código não executa a montagem original da string de comando. Neste caso, o código armazena o parâmetro de entrada (string contendo o nome do arquivo a ser transferido) e executa a função tcpip_getfile.

Resultados

A definição adotada para *speedup* neste artigo é: aumento de velocidade observado quando se executa um determinado processo em *p* processadores em relação à execução deste processo em 1 processador. Então, tem-se:

$$speedup = \frac{T_1}{T_p} \qquad (1)$$

Onde: T_1 = tempo de execução em 1 processador e T_p = tempo de execução em p processadores.

Outra medida importante é a eficiência, que trata da relação entre o *speedup* e o número de processadores.

$$eficiência = \frac{speedup}{p} \qquad (2)$$

No caso ideal (*speedup* = p), a eficiência seria máxima e teria valor 1 (100%).

O ambiente distribuído utilizado é composto de 4 máquinas: 1 Pentium IV 1.8GHz e 576MB de memória RAM, 2 Athlons 800MHz e 256MB de memória RAM, e 1 Pentium IV 3.0GHZ e 1GB de memória RAM.

As Figuras 4a e 4b apresentam os gráficos de desempenho observado durante os testes de execução do treinamento da rede de Kohonen em

ambiente de processamento distribuído semi-dedicado.

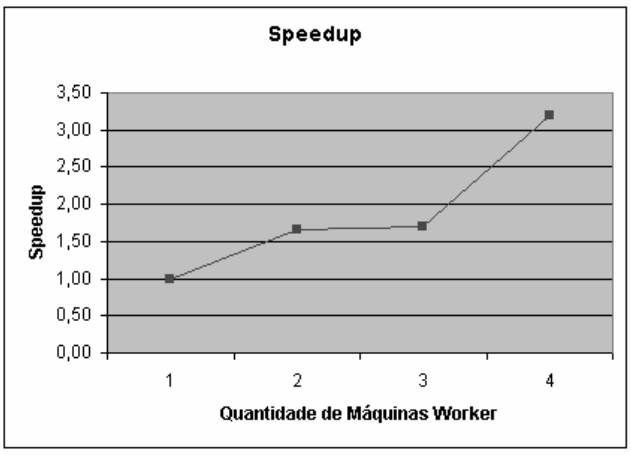

Figura 4a - Speedup - execução do treinamento do Mapa Auto-Organizável de Kohonen em ambiente de processamento distribuído.

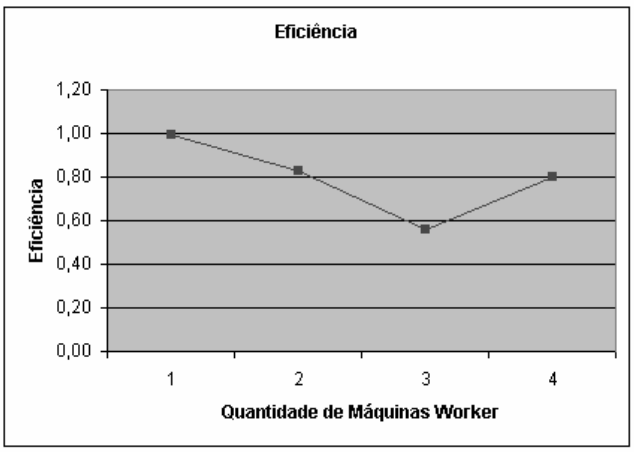

Figura 4b - Eficiência - treinamento do Mapa Auto-Organizável de Kohonen em ambiente de processamento distribuído.

Conclusões

A eficiência da implementação da distribuição do processamento da rede *de Kohonen* pôde ser observada durante alguns experimentos demonstrados na seção anterior.

Fica a hipótese de ter alcançado melhores resultados para *speedup* e *eficiência*, caso o ambiente de processamento distribuído fosse dedicado (sem tráfego de rede) e fosse um ambiente de processamento distribuído homogêneo (hardware e software).

Para um melhor desempenho do classificador neural, seria necessária a análise de desempenho e distribuição do processamento de todos os módulos que constituem o classificador em questão. E, neste caso, a análise poderia ter granularidades diferentes, e conseqüentemente, ganhos de desempenhos diferentes para cada módulo.

Balanceamento de carga seria outro fator importante para melhoria dos resultados. É possível notar nas Figuras 4a e 4b a deficiência deste item quando o número de nós é três. No caso dos testes realizados neste trabalho isso ocorreu devido a necessidade de blocos de treinamentos (4 blocos) contra a quantidade de máquinas "workers" disponíveis para realizar o processamento (3 máquinas).

Como um processo de melhoria, já está em andamento:

Implementação do Classificador Neural em outra linguagem, já desenvolvendo um sistema totalmente distribuído e independente de sistema operacional.

Análise de balanceamento de carga para um número elevado de treinamentos do Mapa Auto-Organizável de Kohonen, em relação ao número de máquinas "workers" disponíveis.

Referências Bibliográficas

[1] MARTINS, M. P. Classificador Neural de atributos de textura para busca e recuperação de imagens de Sensoriamento Remoto. Dissertação de Mestrado, Instituto Nacional de Pesquisas Espaciais, Computação Aplicada, 2003.

[2] HAYKIN, S. Redes Neurais - Princípios e Prática**,** 2ª ed., Bookman, 2001.

[3] PASSARO, A.; NETO, O. F. L.; JÚNIOR, A. M.; RIBEIRO, A. L.; FREITAS, L M.; ABE, N. M.; TANAKA, R. Y. Curso Montagem e Avaliação de Clusters de PC's voltados para aplicações científicas. Instituto de Estudos Avançados – IEAv. 2003.

[4] RANGEL, K. M.; GUIMARÃES, L. N. F. Análise de Desempenho de um Classificador Neural de Atributos de Textura em Ambiente Distribuído. Trabalho de Conclusão de Curso. Engenharia de Computação. Universidade Braz Cubas, 2004.

[5] KOHONEN, T. Self-Organization and Associative Memory, 3ª ed., Springer Series in Information Sciences. 1989.

[6] LEUNG, C. Image and Vision Computing 17 – Introduction. 1999, p. 463-464.

[7] NAVAUX, P. O. A. Introdução ao processamento paralelo, RBC- Revista Brasileira de Computação, v. 5, nº 2, pp.31-43, Outubro, 1989.

[8] MCBRYAN, O. A. An overview of message passing environments, Parallel Computing, v. 20, pp. 417-444, 1994.

[9] BALDOMERO, J. F.; Message Passing under MATLAB. Universidade de Granada, Espanha. 2003.