# Uma Estratégia de Poda de Vizinhos para Treinamento de Redes de Kohonen em Multi-Resolução

Agostinho de Medeiros Brito Júnior, Adrião Duarte Dória Neto e Jorge Dantas de Melo

***Resumo*—Apesar da impressionante capacidade de auto-organização dos mapas de Kohonen, uma de seus maiores problemas ainda consiste nos elevados custos computacionais decorrentes do treinamento da rede neural. Redes com elevado número de neurônios e de conexões geralmente devem manter uma quantidade considerável de vizinhos na etapa de cooperação, de modo a não comprometer a convergência do algoritmo. Muitos desses vizinhos, entretanto, podem não influenciar significativamente no resultado final, servindo apenas para retardar o treinamento. Um exemplo disso se dá quando o treinamento é realizado em multi-resolução. Visando atenuar este problema, este trabalho apresenta uma estratégia simples e eficiente de poda de vizinhos, para aplicação em treinamento em multi-resolução. O método proposto permite remover automaticamente os vizinhos topológicos distantes quando estes não mais contribuem para a modificação do resultado final do treinamento. A eficácia do método é demonstrada através de uma aplicação em reconstrução tridimensional de superfícies.**

***Index Terms*—Redes de Kohonen, mapas auto-organizáveis, treinamento em multi-resolução, poda de vizinhos, redução de custos computacionais.**

## I. Introdução

As redes de Kohonen há muito têm sido utilizadas como valiosas ferramentas para uso em sistemas industriais, aplicações médicas, telecomunicações, robótica, processamento de voz, etc [1]. O mapa auto-organizável provê uma representação compacta de um conjunto de dados presente no espaço n-dimensional, na forma de estruturas geometricamente simples, tais como grafos uni ou bi-dimensionais[2]. O algoritmo de aprendizado permite descobrir a geometria do espaço de entradas, que não é conhecida *a priori*.

Um dos principais inconvenientes do algoritmo é o tempo de processamento necessário para treinar a rede neural. De modo a garantir que esta última tenha convergência adequada, é necessário utilizar um grande número de passos de treinamento. A cada passo, não só o peso de um neurônio vencedor deve ser atualizado, mas também os pesos de uma boa quantidade de vizinhos, visando assegurar uma ordenação topológica correta.

Se o reticulado da rede é muito grande, enormes quantidades de vizinhos deverão ser atualizados, surgindo a demanda por mecanismos de treinamento mais eficientes, tais como SOM hierárquico [3], em multi-camadas. Adicionalmente, estruturas de dados para busca otimizada podem ser agregadas para acelerar a etapa de competição [4].

Departamento de Engenharia da Computação e Automação, Universidade Federal do Rio Grande do Norte, Natal, Brasil {ambj, adriao, jdmelo}@dca.ufrn.br

Muitas vezes, o treinamento pode ser realizado para produzir resultados em múltiplas resoluções. Esta alternativa é especialmente atrativa quando ocorre a necessidade de recuperar apenas informações de baixa freqüência, presentes em níveis de resolução mais grosseiros. Esta necessidade emerge para espaços de entrada muito grandes, quando então pode ser conveniente treinar a rede neural para gerar grafos em resoluções mais grosseiras (freqüências baixas) e, em seguida, retreiná-las para aprender detalhes.

A reconstrução tridimensional de superfícies é uma aplicação típica onde o treinamento em multi-resolução demonstra sua utilidade [5]. Em aplicações desta natureza, quanto menor a quantidade de elementos necessária para representar as características de um objeto e seus detalhes, mais rápida se dá a realimentação visual no terminal gráfico que irá exibi-lo para o usuário [6]. Se um objeto é representado por uma quantidade muito grande de elementos, a realimentação visual rápida pode ser feita em níveis de resolução baixos, sendo os detalhes apresentados quando a necessidade de realimentação diminui [7]. A rotação de um objeto na tela, por exemplo, é um exemplo de operação que requer realimentação visual rápida.

Começando de um grafo inicial bastante grosseiro, a reconstrução em multi-resolução com uma rede neural opera adicionando neurônios aos níveis de resolução grosseiros, permitindo assim o aprendizado de detalhes que estarão presentes nos níveis de resolução mais finos. Com isso, terminado o treinamento, o usuário tem à disposição reconstruções em diferentes níveis de detalhamento, podendo selecionar o nível de detalhes que deseja exibir de acordo com a disponibilidade de recursos do seu *hardware* gráfico. Quando a necessidade de realimentação rápida diminuir, as reconstruções obtidas em níveis de resolução mais refinados podem ser então exibidas.

No treinamento em multi-resolução, é desejável que os grafos de resolução mais grosseira gerados pela rede neural possam refletir os dados presentes na amostra de entrada como um todo. Por outro lado, nas resoluções mais finas, é desejável que cada região do grafo se especialize apenas na representação dos detalhes da região correspondente no espaço de entradas. Sendo assim, é justo estreitar as vizinhanças dos neurônios nos níveis de resolução mais finos de modo a direcionar o treinamento de uma abrangência global, nas resoluções grosseiras, para uma abrangência local, nas resoluções finas.

Neste contexto, a modificação do algoritmo SOM para aplicações em multi-resoluções pode acelerar o processo de treinamento, sem prejuízos práticos na qualidade dos re-

sultados obtidos. Este trabalho introduz regras simples para acelerar o treinamento da rede neural, que levam em conta a qualidade da aproximação obtida pelo grafo da rede neural, nos termos de uma métrica Euclidiana. A eficiência das regras propostas é testada com uma aplicação em reconstrução tridimensional da superfície de um mapa de elevação de terreno. A quantidade de elementos representativos da superfície utilizada neste trabalho quadruplica do nível de resolução mais grosseiro para o nível mais fino imediatamente posterior. Quando os detalhes são acrescentados nos níveis de resolução mais finos, o erro obtido com a reconstrução é calculado e comparado com o da resolução mais grosseira. Na medida em que este erro diminui, menos vizinhos são levados em consideração na atualização dos pesos, restringindo a abrangência do treinamento.

## II. Mapas Auto-organizados: O algoritmo de Kohonen

Teuvo Kohonen[2] propôs em seu modelo de rede neural um estrutura estatística auto-organizada que possui a capacidade de aproximar o espaço de entradas utilizado para treinamento através de um grafo, ou mapa auto-organizado (*Self-organizing Map* - SOM). O objetivo do treinamento é transformar padrões de entrada de uma dimensão arbitrária em um mapa discreto, de forma adaptativa e topologicamente ordenada [8]. O mapa é normalmente um reticulado bidimensional (ver Fig. 1) de nós cujos pesos armazenam uma aproximação para o espaço de entradas.

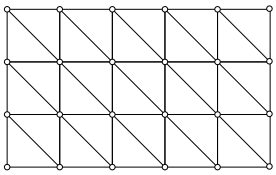

Figura 1. Mapa auto-organizado de Kohonen com reticulado triangular.

Os padrões de entradas consistem em um conjunto $\mathscr{X}$ de vetores da forma:

$$\mathbf{x} = [\ x_1 \quad x_2 \quad \dots \quad x_m\ ]^T \tag{1}$$

e os pesos da rede são vetores denotados por:

$$\mathbf{w}_j = [\ w_{j1} \quad w_{j2} \quad \dots \quad w_{jm}\ ]^T, \quad j = 1,2,\dots,l, \tag{2}$$

onde $m$ é a dimensão do espaço de entradas e $l$ é o número de neurônios no mapa.

O algoritmo de treinamento é composto de três passos principais para cada padrão de entrada apresentado ao mapa:

1. **Competição**, onde o valor de uma função discriminante é calculada para cada neurônio. A distância Euclidiana é normalmente a escolha natural para esta métrica. O neurônio $i$ com menor valor para a função discriminante

$$i(\mathbf{x}) = \arg\min_i \|\mathbf{x} - \mathbf{w_j}\| \tag{3}$$

é considerado vencedor;
2. **Cooperação**, onde cada neurônio estabelece uma vizinhança topológica de nós para serem afetados. Os nós afetados serão ativados de acordo com uma função de vizinhança, geralmente uma gaussiana:

$$h_{j,i(\mathbf{x})} = \exp\left(-\frac{d_{j,i}^2}{2\sigma^2}\right) \tag{4}$$

$$\sigma(n) = \sigma_0 \exp\left(-\frac{n}{\tau_1}\right) \tag{5}$$

onde $d_{j,i}$ é a distância topológica entre os nós $j$ e $i$, $\sigma_0$ é o espalhamento da função de vizinhança no início do treinamento, $\tau_1 = niter/\log(\sigma_0)$ é uma constante de tempo, e $n = 0,1,\dots,niter$ é o passo de execução do algoritmo, com $niter$ iterações utilizadas para treinamento;
3. **Adaptação**, onde os pesos de cada neurônios são modificados para melhorar a resposta aos padrões de entrada similares. Tais modificações podem ser *sequenciais*, se os os pesos forem modificados na medida em que os padrões de entrada são apresentados ao mapa, ou *em lote*, utilizando a equação

$$\mathbf{w}_j(n+1) = \frac{\sum_{k=0}^{m} h_{j,i(\mathbf{x}(k))}\mathbf{x}(k)}{\sum_{k=0}^{m} h_{j,i(\mathbf{x})}}, \tag{6}$$

para atualização dos pesos dos neurônios. Observe que os pesos são modificados apenas uma vez, somente após todos os padrões de entrada terem sido apresentados à rede neural.

Neste trabalho, é utilizada a versão em lote do SOM. Uma vez que a atualização dos pesos não depende da ordem de apresentação dos padrões de entrada à rede neural, a comparação entre duas execuções diferentes do algoritmo não levará em conta o processo de sorteio utilizado pela versão seqüencial. Com isso, a comparação dos tempos de execução da rede neural evidenciará apenas as diferenças entre a versão do SOM original e a que é proposta neste trabalho.

## III. Reconstrução 3D com redes de Kohonen

Os dados de entrada utilizados para a rede consistem em conjuntos de pontos não estruturados no espaço 3D, ou seja, sem informação de vizinhança presente. Dispõe-se, portanto, apenas das posições da cada um dos pontos, desconhecendo-se qualquer informação acerca da geometria da superfície a ser reconstruída.

Uma malha triangular é utilizada para representar a superfície durante o processo de reconstrução. Cada superfície é uma variedade-2 com bordas [9], representada por uma malha poligonal $M$, topologicamente equivalente a uma folha de papel. Esta malha é composta de um conjunto de vértices $V$, um conjunto de arestas $E$ interconectando pares de vértices e um conjunto de faces triangulares $F$, como ilustra a Figura 2.

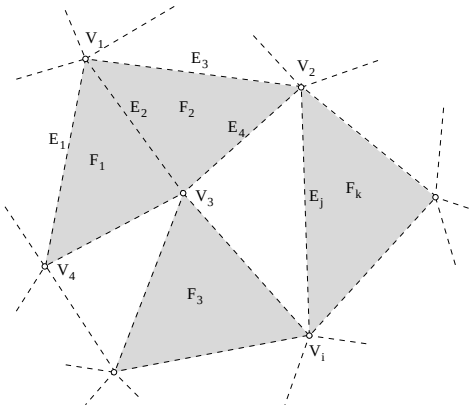


Figura 2. Representação de malha triangular

O treinamento da rede neural utiliza como espaço de entradas uma nuvem de pontos amostrados da superfície a ser reconstruída. Cada ponto dessa nuvem é uma tripla

$$\overline{x} = \left[ \begin{array}{ccc} x_1 & x_2 & x_3 \end{array} \right]^T \tag{7}$$

representando as suas coordenadas $x$, $y$ e $z$, respectivamente, no espaço cartesiano.

O processo completo de reconstrução é realizado em multi-resolução e dividido em duas etapas: treinamento de uma rede neural, para criação de uma malha de superfície a partir de um conjunto de pontos de entrada, e refinamento da superfície, utilizando técnicas de subdivisão de elementos.

O processo de reconstrução inicia com a criação do reticulado da rede neural. Partindo de uma malha inicial pré-definida pelo utilizador do método, os vértices dessa malha e suas respectivas posições espaciais determinarão a quantidade de neurônios presentes na rede neural e os pesos atribuídos a cada um. Sendo assim, os pesos dos neurônios serão da forma

$$w_i = \left[ \begin{array}{ccc} w_{i1} & w_{i2} & w_{i3} \end{array} \right]^T, i = 1, \ldots, n_v, \tag{8}$$

onde $n_v$ é o número de vértices da malha inicial ou neurônios na rede.

As conexões existentes entre os vértices, representadas pelas arestas da malha, irão determinar às ligações que deverão existir entre os neurônios, definindo as relações de adjacência na rede. Em suma, a topologia e a geometria da rede neural deverão espelhar a topologia e a geometria da malha inicial. Cada vértice da malha irá corresponder a um neurônio da rede neural, ambos marcados com um mesmo número identificador. Neste caso, o $i$-ésimo neurônio da rede neural irá corresponder ao $i$-ésimo vértice da malha. Esta restrição permite fixar a topologia da malha durante todas as etapas da reconstrução em multi-resolução.

O treinamento irá movimentar os vértices da malha de modo que esta possa se aproximar cada vez mais dos pontos da nuvem, ou seja, realizar a reconstrução propriamente dita.

Terminado o treinamento da rede neural, cada triângulo da malha gerada é subdivido em outros quatro, conforme ilustrado na Figura 3. Os novos neurônios são inseridos na posição central da aresta presente entre cada dois vértices. Este procedimento de subdivisão assegura a manutenção a mesma topologia da malha inicial, e, de forma bastante simples, a correspondência entre uma face de uma resolução mais grosseira e uma face de uma resolução mais fina.

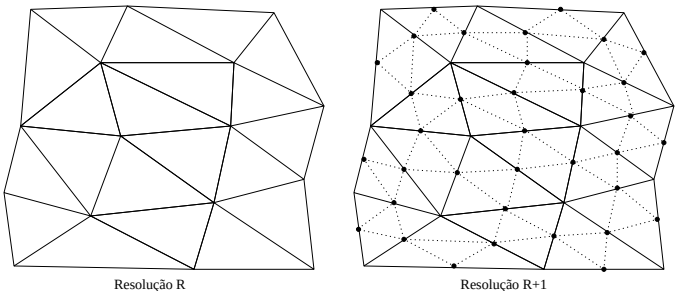


Figura 3. Subdivisão dos triângulos da malha

Terminada a subdivisão de triângulos, a rede neural, agora com mais elementos, é novamente treinada, utilizando como pesos iniciais dos neurônios as posições dos vértices da malha subdividida.

O erro de reconstrução é calculado com base na medida das distâncias entre os triângulos e a amostra de pontos. Para quantizar o erro (ou instabilidade) de um triângulo, a medida da distância é realizada entre seu baricentro $\overline{B}$ e o espaço de entradas $\mathscr{X}$, sendo representada como $dT = d(\overline{B}, \mathscr{X})$. A distância real entre o triângulo e a amostra não caracteriza, por si só, instabilidade pois, se todos os vértices dos triângulos estiverem próximos da amostra, mas não o seu baricentro, o triângulo ainda será considerado estável. Sendo assim, as distâncias Euclidianas mínimas dos baricentros $B$ dos triângulos para o conjunto de treinamento são obtidas pela equação

$$dT = d(\overline{B}, \mathscr{X}) = \min_{\overline{x} \in \mathscr{X}} \|\overline{x} - \overline{B}\|. \tag{9}$$

O valor médio de $dT$, denotado por $\overline{dT}$, irá representar o erro existente entre a amostra e a superfície reconstruída. Valores elevados de $\overline{dT}$ indicam superfícies distantes da amostra de pontos, e vice-versa. Fatores como planaridade, curvatura, continuidade, conformidade etc [10] não foram considerados neste trabalho, posto que geralmente não são preponderantes nas malhas regulares aqui utilizadas [10].

Este erro pode ser definido também como um percentual da diagonal caixa limitante (*Bounding Box*)[1] da superfície, numa forma mais fácil de abstrair o valor absoluto do erro em relação ao tamanho do objeto gerado. Sendo assim, o erro absoluto percentual será

$$\overline{dT}\% = \frac{\overline{dT}}{\sqrt{L^2 + H^2 + C^2}} \times 100\%, \tag{10}$$

onde $L$, $H$ e $C$ são a largura, altura e comprimento, respectivamente, da caixa limitante da amostra de pontos.

São utilizados $N$ passos de treinamento e outros $N$ passos para uma fase de convergência adicional. Nesta última,

[1] Menor paralelepípedo que abriga uma figura geométrica.

apenas os neurônios mais próximos do neurônio vencedor são mantidos na vizinhança. Estes passo adicionais são utilizados para compensar alguma capacidade adaptativa que é perdida na versão em lote do SOM [2].

## IV. Modificações no algoritmo de Kohonen para treinamento em multi-resolução

Na medida em que os níveis de resolução obtidos pela rede neural se tornam mais finos, a quantidade de elementos na rede aumenta. Por conseguinte, o tamanho da vizinhança descrito pela equação (4) também irá aumentar. O parâmetro $\sigma_0$ apresentado nesta mesma equação exerce papel fundamental na determinação da zona de influência do neurônio vencedor. Quanto maior o valor de $\sigma_0$, maior a ponderação $h_{j,i(\mathbf{x})}$ dos vizinhos pelo neurônio vencedor.

Tradicionalmente, $\sigma_0$ é escolhido como sendo igual a metade do diâmetro da rede neural. Para o caso de um treinamento convencional, a escolha é acertada. Contudo, a opção pela mesma regra no caso do treinamento em multi-resolução irá surtir um efeito indesejado: o encolhimento exagerado da rede neural. Se $\sigma_0$ iniciar, nos níveis de resolução mais finos, com um valor muito alto a malha subdividida será demasiadamente encolhida, ao invés de continuar a se expandir do ponto onde parou. Em termos práticos, o valor elevado de $\sigma_0$ impedirá a rede neural de aprender novos detalhes e fará com que aprenda novamente detalhes grosseiros que já aprendeu em treinamento anterior.

À medida em que a quantidade de elementos aumenta, melhores reconstruções são obtidas no próximo nível de resolução. Neste caso, a zona de influência de cada neurônio pode ser reduzida mais e mais nos treinamentos subseqüentes, para obter uma reconstrução que reflita as características locais do conjunto de treinamento.

Esta redução é feita pela **mudança do tamanho do parâmetro de vizinhança** $\sigma_0$ na equação (4). Foi estabelecido um método para determinar os valores de $\sigma_0$ no terceiro nível de resolução e seguintes, utilizando o valor médio de $dT$ obtido após a primeiro treinamento e os valores médios subseqüentes para $dT$. A notação $\overline{dT}(R)$ será utilizada para identificar o valor médio de $dT$ no $R$-ésimo nível de resolução. Assumindo para $\sigma_0(R)$, o valor de $\sigma_0$ na $R$-ésima resolução, ter-se-á portanto:

$$\sigma_0(R) = \frac{diamRede}{2}\frac{\overline{dT}(R-1)}{\overline{dT}(0)}, \qquad (11)$$

o treinamento da rede passa a contar com uma importante informação sobre a qualidade da reconstrução realizada, ou seja, sobre o nível de detalhes aprendido. Quanto mais a superfície se aproxima do conjunto de pontos, mais estreita se torna a vizinhança inicial do neurônio, ou seja, menos vizinhos serão levados em consideração no somatório do numerador na equação (6). Com isso, o treinamento torna o esquema de reconstrução de uma abrangência global para uma abrangência local, especializando cada vez mais os neurônios da rede nos níveis de resolução mais fina. Para este trabalho, os vizinhos com valores iniciais para o valor da função de vizinhança $h_{j,i(\mathbf{x})}$ menores que $10^{-15}$ são descartados, ou seja, não terão seus pesos atualizados pela equação (6).

Um exemplo da influência da poda de vizinhos é mostrada na Figura 4. Nela está representada uma malha de superfície reconstruída no sexto nível de resolução. O neurônio vencedor é indicado pela seta, ficando situado no centro da vizinhança topológica. As regiões marcadas pelos triângulos hachuriados representam os vértices que serão abrangidos pela equação (6). Observe que sem a poda de vizinhança a quantidade de vizinhos que serão atualizados por esta equação engloba a totalidade da malha, ao passo que, quando a poda de vizinhos é considerada, a atualização condensa-se apenas numa região estreita ao redor do neurônio vencedor.

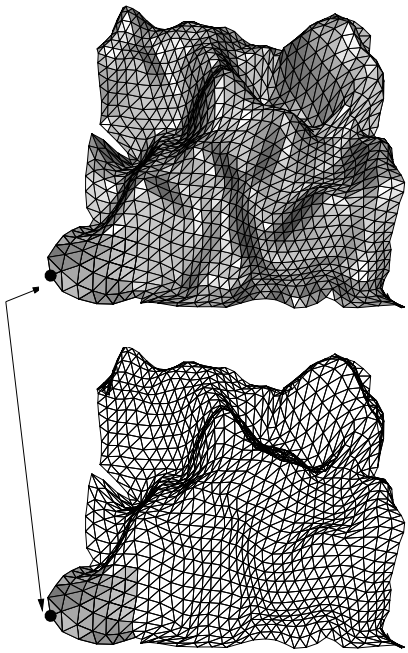

Figura 4. Vizinhos topológicos sem poda (acima) e com poda de vizinhança (abaixo).

Sem a restrição de poda, além do aumento do tempo de treinamento da rede, a capacidade de especialização de cada nó é reduzida nos nível mais finos. Sem a restrição da região de influência de cada neurônio, as regiões da malha passarão a receber influências cada vez mais fortes de neurônios remotos, cuja importância para o comportamento local deveria ser mínima. No caso particular da reconstrução tridimensional, a não realização da poda de vizinhos pode, inclusive, prejudicar os resultados, levando à geração de malhas incorretas e retorcidas.

## V. Experimentos e resultados

As modificações no algoritmo de Kohonen foram implementadas em C++ e testadas em um PC dotado de um processador Pentium III 800MHz rodando Linux. A malha inicial utilizada é formada por apenas dois triângulos. Os pesos iniciais da rede neural, no treinamento para o

primeiro nível de resolução, são dados pelas posições dos vértices da malha inicial. O algoritmo SOM foi treinado com 100 iterações em cada nível de resolução. Foram testados números de iterações maiores, mas os resultados não diferiram muito.

O método foi testado com um conjunto de dados sintético, representando o mapa de elevação de um terreno, composto de 10.000 pontos. As redes neurais foram treinadas de modo a gerar superfícies até o sétimo nível de resolução, equivalendo a uma malha triangular de 4225 vértices e 8192 triângulos.

As superfícies resultantes do treinamento da rede neural no sétimo nível de resolução são mostrados na Figura 5.

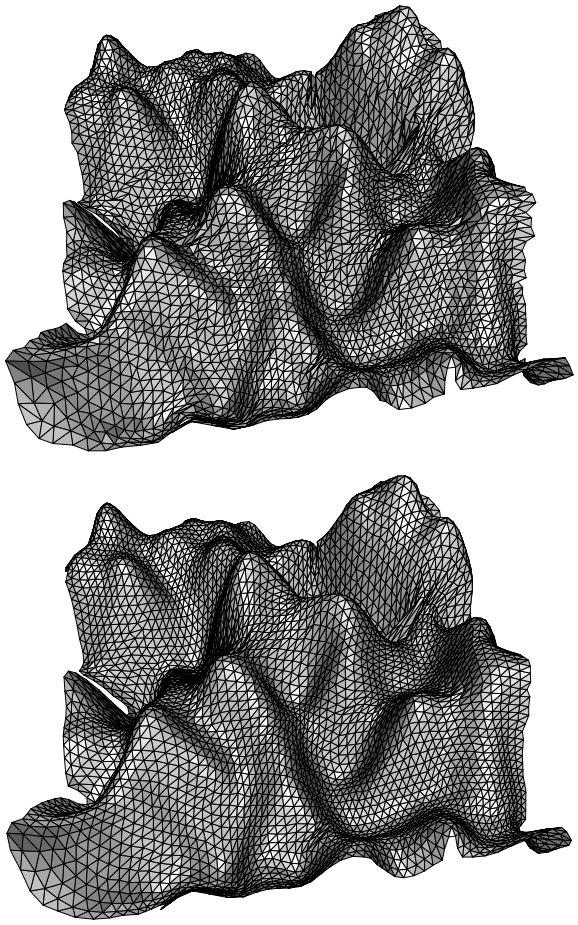

Figura 5. Reconstrução de superfícies sem poda (acima) e com poda de vizinhos (abaixo).

Os tempos de reconstrução para cada nível de resolução, para os caso sem poda e com poda de vizinhos, são mostrados na Figura 6. É possível observar que o tempo necessário para reconstruir a superfície vai decrescendo quando a poda é realizada e o número de neurônios na vizinhança é reduzido. No sétimo nível de resolução, por exemplo, o tempo de treinamento cai para aproximadamente metade do tempo necessário para realizar o treinamento convencional. Os erros $\overline{dT}\%$ obtidos para as reconstruções no sétimo nível de resolução foram 0.5395% e 0.4943%, respectivamente, para os casos com poda de vizinhos e sem poda de vizinhos.

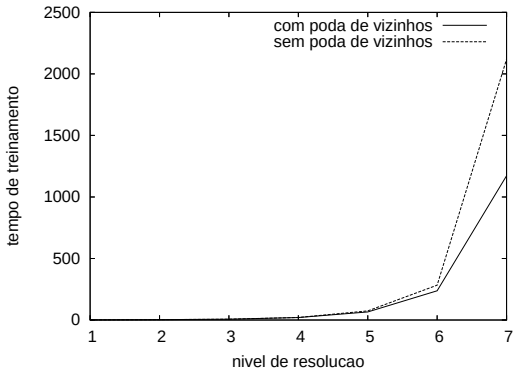


Figura 6. Tempos de treinamento para diferentes níveis de resolução, com e sem poda de vizinhos.

É possível notar que, mesmo com a poda de vizinhos, ainda foi possível obter uma reconstrução com erro semelhante ao obtido no caso sem poda. Para fins práticos, a pequena diferença entre ambos não é significativa. Entretanto, o decréscimo no tempo de treinamento para as resoluções mais finas é evidente no caso da poda quando comparado ao caso sem poda de neurônios, mostrando a utilidade da modificação proposta no parâmetro $\sigma_0$.

Embora o método tenha sido aplicado em reconstrução tridimensional, é facilmente estendido para outras aplicações de treinamento em multi-resoluções. Dispondo de uma maneira de avaliar a qualidade do treinamento da rede neural, ou seja, uma métrica de erro entre o espaço de entradas e o grafo gerado, é possível aplicar a poda de vizinhos sem fortes prejuízos nos resultados obtidos. Como principal benefício vem a redução drástica do tempo de treinamento da rede.

## VI. Conclusões

Foi apresentada uma proposta para acelerar o tempo de treinamento do algoritmo SOM para aplicações que geram resultados em múltiplas resoluções. Nestas aplicações o resultado do treinamento da rede obtido em um nível de resolução grosseiro é utilizado como estado inicial da rede em um nível de resolução mais refinado. A modificação básica foi realizada no parâmetro de espalhamento da vizinhança $\sigma_0$, colocando este como função de uma métrica de qualidade do treinamento. Com isto, o valor de $\sigma_0$ nos níveis de resolução mais finos tornam-se funções das métricas de qualidade obtidas para os níveis de resolução mais grosseiros. O resultado prático da modificação na escolha deste parâmetro é a evidente redução do tempo de treinamento nos níveis de resolução mais finos sem grandes prejuízos da qualidade do treinamento. O método foi aplicado em um problema de reconstrução tridimensional de superfícies, mas pode ser facilmente adaptado para outras aplicações.

## Referências

[1] S. Kaski, J. Kangast, and T. Kohonen, "Bibliograpy of self-organizing map (som) papers: 1981-1997," *Neural Computing Surveys*, vol. 1, 1998.

[2] T. Kohonen, *Self-organization and associative memory*. New York: Springer-Verlag, 2001.
[3] S. P. Luttrell, "Hierarchical self-organising networks," in *Proceedings of 1st International Conference on Artifical Neural Networks*, 1989.
[4] S. Arya, D. M. Mount, N. S. Netanyahu, R. Silverman, and A. Y. Wu, "An optimal algorithm for approximate nearest neighbor searching," *Journal of the ACM*, no. 45, pp. 891–923, 1998.
[5] A. de Medeiros Brito Júnior, A. D. D. Neto, and J. D. de Melo, "Surface reconstruction using neural networks and adaptive geometry meshes," in *Proceedings of the International Joint Conference on Neural Networks 2004*, (Budapest), 2004.
[6] H. Hoppe, *Surface reconstruction from unorganized points*. PhD thesis, University of Washington, 1994.
[7] E. Shaffer and M. Garland, "A multiresolution representation for massive meshes," *IEEE Transactions on Visualization and Computer Graphics*, vol. 11, no. 2, pp. 139–148, 2005.
[8] S. Haykin, *Neural Networks: a comprehensive foundation*. New Jersey: Prentice-Hall, 2 ed., 1999.
[9] E. W. Weisstein, ed., *CRC Concise Encyclopedia of Mathematics*. CRC Press, 2 ed., 2002.
[10] P. J. Frey and H. Boroucraki, "Surface mesh quality evaluation," *International Journal for Numerical Methods in Engineering*, vol. 45, pp. 101–118, 1999.