PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS
Programa de Pós-Graduação em Informática

Uso da Análise dos Fatores de Sensibilidade para encontrar a Quantidade Ideal Mínima de Neurônios na Camada Escondida de uma RNA Perceptron Multicamadas através dos Algoritmos Genéticos

Fabrício Roulin Bittencout

Belo Horizonte
2007

Fabrício Roulin Bittencout

Uso da Análise dos Fatores de Sensibilidade para encontrar a Quantidade Ideal Mínima de Neurônios na Camada Escondida de uma RNA Perceptron Multicamadas através dos Algoritmos Genéticos

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Informática da Pontifícia Universidade Católica de Minas Gerais, como requisito parcial para a obtenção do grau de Mestre em Informática.

Orientador: Luis Enrique Zárate

Belo Horizonte
2007

FICHA CATALOGRÁFICA

Elaborada pela Biblioteca da Pontifícia Universidade Católica de Minas Gerais

Bittencout, Fabrício Roulin

B624u Uso da análise dos fatores de sensibilidade para encontrar a quantidade ideal mínima de neurônios na camada escondida de uma RNA perceptron multicamadas através dos algoritmos genéticos / Fabrício Roulin Bittencout. – Belo Horizonte, 2007.

102 f. : il.

Orientador: Prof. Dr. Luis Enrique Zárate.

Dissertação (mestrado) – Pontifícia Universidade Católica de Minas Gerais, Programa de Pós-graduação em Informática.

Bibliografia.

1. Redes neurais (Computação). 2. Algoritmos genéticos. 3. Inteligência artificial I. Zárate, Luis Enrique. II. Pontifícia Universidade Católica de Minas Gerais. III. Título

CDU: 681.3.055

Bibliotecário: Fernando A. Dias – CRB6/1084

PUC Minas
Programa de Pós-graduação em Informática

FOLHA DE APROVAÇÃO

"Uso da Análise dos Fatores de Sensibilidade para encontrar a Quantidade Ideal Mínima de Neurônios na Camada Escondida de uma RNA *Perceptron Multicamadas* através dos Algoritmos Genéticos"

Fabricio Roulin Bittencout

Dissertação defendida e aprovada pela seguinte banca examinadora:

Prof. Luis Enrique Zárate Galvez - Orientador (PUC Minas)
Doutor em Engenharia Metalurgica e de Minas (UFMG)

Prof. João Antônio de Vasconcelos (UFMG)
Doutor em Engenharia Elétrica (França)

Prof. Mark Alan Junho Song (PUC Minas)
Doutor em Ciência da Computação (UFMG)

Prof. Ricardo Poley Martins Ferreira (PUC Minas)
Doutor em Ciência da Computação (UFMG)

Belo Horizonte, 18 de dezembro de 2007.

Anel Rodoviário – KM 23,5, Rua Walter Ianni, 255 – São Gabriel / Belo Horizonte – MG / CEP: 31980-110
Fone: (31) 3439-5204 – Fax: (31) 3439-5204 – e-mail: mestradoinfsec@pucminas.br

À Deus, por tudo.

À minha querida avó Esther, pelo amor incondicional e dedicação, sempre me dando força para seguir em frente.

Aos meus queridos pais por acreditarem em meu sonho e fazerem tudo para que este se tornasse realidade, mesmo quando nada mais poderia ser feito.

Às minhas irmãs que sempre acreditaram em mim, mais do que eu mesmo.

À minha esposa, por todo amor e apoio fundamental nesta fase tão importante da minha vida.

Agradecimentos

Meus sinceros agradecimentos à Fundação Comunitária de Ensino Superior de Itabira, pela ajuda e incentivo.

À Yana, pela amizade e força em todos os momentos em que precisei.

Ao meu orientador, Prof. Luis E. Zárate, que sempre foi muito mais que meu professor e meu orientador, sempre acreditou e confiou em meu trabalho; o responsável por chegar até aqui.

Não posso esquecer aquele que me colocou no caminho da pesquisa científica, Prof. Mark Alan J. Song. Serei eternamente grato por ter me dado a oportunidade para seguir na área acadêmica.

A todos os meus amigos, que, estando perto ou longe, sempre os levo comigo.

Ao Prof. Ricardo Poley, pelas importantes contribuições para o fechamento deste trabalho.

A todas as outras pessoas que, direta ou indiretamente, contribuíram para a realização deste trabalho.

A todos, muito obrigado.

"O grande segredo para a plenitude é muito simples: compartilhar."
Sócrates

"Quando você tem uma meta, o que era um obstáculo passa a ser uma etapa de um de seus planos."
Gerhard Erich Boehme

Resumo

Existe uma dificuldade para determinar a quantidade mais adequada de neurônios na camada escondida de uma RNA. Como solução, o projetista usa sua experiência, uma fórmula empírica, um mecanismo heurístico ou tentativa e erro. O objetivo deste trabalho consiste em encontrar a quantidade ideal mínima de neurônios na camada escondida de uma RNA Perceptron Multicamadas avaliando o aprendizado das RNA's, pelo conhecimento do especialista no domínio do problema, através da Análise de Sensibilidade, incorporado ao mecanismo evolutivo dos Algoritmos Genéticos. Para isso, foi proposta uma nova estrutura evolucionária que consegue ajudar, de forma rápida, na definição da quantidade ideal mínima de neurônios na camada escondida de uma RNA perceptron multicamadas, sem que as redes sejam treinadas várias e exaustivas vezes, gerando uma estrutura otimizada sem que haja perda na capacidade de generalização.

Abstract

There is a difficulty in determining the most appropriate quantity of neurons in the hidden layer of a RNA. As a solution, the designer uses his experience, an empirical formula, a heuristic mechanism or trial and error. The objective of this work is to find the ideal minimum number of neurons in the hidden layer of a RNA Multilayer Perceptron assessing the learning of the RNAs, the knowledge of the specialist in the field of the problem, through the Sensitivity Analysis embedded in the evolutionary mechanism of Genetic Algorithms. Therefore, it was proposed that a new evolutionary structure can help, quickly, in the definition of the ideal minimum number of neurons in the hidden layer of a RNA Multilayer Perceptron, without which the networks are extensively and several times trained, creating an optimized structure without losing the generalization ability.

Lista de Figuras

Lista de Tabelas

Sumário

1 Introdução

Este capítulo tem por objetivo servir de escopo ao trabalho desenvolvido. Apresentando a motivação para o desenvolvimento desta pesquisa, juntamente com os objetivos a serem alcançados, e a organização da dissertação.

A capacidade de aprendizado é a principal característica de uma Rede Neural Artificial (RNA). Aprender por meio de exemplos que lhe são apresentados na fase de treinamento e, depois disto generalizar o conhecimento adquirido, é a principal motivação em usar uma RNA para solução de problemas do mundo real. A generalização, situação em que a RNA é capaz de aprender através de um pequeno conjunto de exemplos e em seguida apresentar respostas satisfatórias para valores não vistos durante o treinamento, é uma revelação de que a capacidade das RNA's ultrapassa o simples mapeamento de relações de entrada e saída. Esta e outras habilidades tornam as RNA's ferramentas atraentes e bastante poderosas computacionalmente para resolução de problemas complexos (BRAGA; CARVALHO; LUDEMIR, 2000).

Sabe-se que uma RNA possui alto desempenho quando é colocada em operação, mas para que esta rede aprenda a relação dos conjuntos a ela fornecida, é necessário um grande esforço computacional que dependerá da complexidade do processo, do tamanho do conjunto de treinamento, da natureza dos dados de treinamento, etc (HAYKIN, 2001).

Filho e Carvalho (1997) chamam a atenção para sistemas computacionais que utilizam RNA's do tipo Perceptron Multicamadas (MLP), pois estão condicionados tanto à estrutura destas redes: tamanho, estrutura, conexões; quanto aos seus parâmetros: taxa de aprendizado e termo momentum. Ou seja, a forma como a arquitetura da rede é definida afeta significativamente seu desempenho, que pode ser classificado em: velocidade de aprendizado, capacidade de generalização, tolerância à falhas e precisão no aprendizado.

Outra observação importante apresentada por Filho e Carvalho (1997) é a dificuldade em projetar uma RNA eficiente, mesmo existindo algumas técnicas que utilizam conhecimentos empíricos, os quais ora ajudam ora não. Isso se dá devido a peculiaridades inerentes aos processos em que as redes são aplicadas. Não há um modelo bem definido onde podemos confiar o projeto da estrutura da rede neural a ser utilizada.

Um teorema que é muito usado é o apresentado por Hecht-Nielsen (1989) e Hecht-Nielsen (1990) citado por Kovács (1996):

> *Teorema Kolmogorov-Nielsen: Dada uma função contínua arbitrária* $f:[0,1]^n \rightarrow \Re^m, f(x) = y$, *existe sempre para f, uma implementação exata com uma rede neural de três camadas, sendo a camada de entrada um vetor de dimensão n, a camada oculta composta por (2n+1) neurônios, e a camada de saída com m neurônios representando as m componentes do vetor y (KOVÁCS, 1996, p. 95).*

Poggio e Girosi (1990) afirmam que uma RNA com apenas uma camada escondida consegue aproximar (aprender), com uma precisão arbitrária, qualquer função contínua. A fundamentação desta afirmativa por Poggio e Girosi, dá-se pelo fato de uma função contínua qualquer, limitada sobre um intervalo, pode ser como uma superposição linear de sinos, os quais podem ser mapeados por pares de neurônios da camada escondida, desde que a função de ativação dos neurônios seja a função sigmóide.

Todavia, sabe-se a quantidade de neurônios da camada escondida de uma RNA está relacionada com a complexidade do processo a ser representado. Em problemas complexos, mesmo usando o critério apresentado por Hecht-Nielsen (1990) para estimar a quantidade de neurônios na camada escondida, não há garantia de que se encontrou a quantidade ideal. Não encontrando a quantidade ideal, não se sabe se é necessário aumentar ou diminuir a quantidade de neurônios. Neste caso, serão necessárias várias tentativas até encontrar a quantidade ideal.

Contudo, a definição da estrutura de uma RNA's é uma arte que exige experiência do projetista. Pois, ao aumentar a quantidade de neurônios na camada escondida, aumenta-se a capacidade de associação não-linear da RNA. No entanto, aumentar demais pode levar a rede neural a decorar os conjuntos de treinamento, inclusive algum ruído caso exista, prejudicando a generalização do aprendizado. Além disto, neurônios a mais exigem maior esforço computacional. Por outro lado, uma RNA com poucos neurônios na camada escondida pode não conseguir mapear todo o universo desejado (underfitting) (HAYKIN, 2001).

Em Filho e Carvalho (1997) é dito que na prática, uma abordagem largamente utilizada como ponto de partida é a construção de redes com arquiteturas semelhantes a outras já usadas em sistemas similares. A partir daí, a rede será testada para o domínio desejado e os resultados dos testes influenciarão na alteração da estrutura e de seus parâmetros. Este processo de adaptação tem fim quando se encontra uma arquitetura razoavelmente apropriada para a aplicação desejada. Embora esta estratégia seja muito utilizada na prática, ela demanda muito tempo, e não garante resultados muito confiáveis, além de ser altamente dependente da

experiência do projetista. Outro ponto negativo é o fato de não garantir otimização, pois o critério para otimizar é composto por uma combinação complexa de fatores. Como conseqüência disto, vários estudos vem sendo realizados para que seja possível otimizar estruturas de RNA's para classes particulares de problemas. Alguns destes estudos utilizam abordagens evolutivas como heurística de busca de estruturas aproximadamente ótimas.

Uma abordagem evolutiva proposta em Yao (1993a) e Yao (1993b) que vem sendo usada para solução deste problema são os Algoritmos Genéticos (AG). Estes são ferramentas poderosas de busca e otimização baseadas na Teoria da Evolução de Darwin. As variáveis do problema são representadas através de genes em um cromossomo, também denominados indivíduos. Cada ponto no espaço de busca e otimização representa um indivíduo e as coordenadas do ponto representam os genes deste cromossomo. Assim, através de um conjunto de pontos, constituí-se uma população que serão as soluções candidatas do problema (GOLDBERG, 1989).

Os AG's vem sendo empregados em RNA's, na maioria dos trabalhos no ajustes dos pesos, no projeto otimizado da arquitetura, na seleção de variáveis de entrada, na seleção dos pesos iniciais, dentre outras (YAO, 1995).

Quando o AG é usado para otimizar a arquitetura de uma RNA, todas as redes que compõe a população de soluções candidatas são treinadas, pois não há como avaliar uma RNA sem que esta passe por um processo de treinamento. Após o treinamento, são avaliadas na maioria das vezes pelo erro apresentado ao final deste ou pelo erro da simulação, situação em que a rede é submetida a um conjunto de testes (validação). Em outros casos são avaliadas pela quantidade de épocas gastas com treinamento, ou pela capacidade de generalização, ou ambos (YAO, 1995). Isto se torna um problema, pois as RNA's são treinadas várias e exaustivas vezes em busca de uma estrutura ideal. Desta forma, este mecanismo difere-se da definição arbitrária por parte do projetista, que ajusta a estrutura com base nas análises que faz do desempenho da rede, apenas por ser um mecanismo de busca automático.

Na fase de treinamento, a RNA passa por um processo iterativo de ajustes dos seus pesos sinápticos e bias, até atingir um erro pré-estipulado. A cada iteração a RNA aumenta seu aprendizado. O ponto de partida no treinamento de uma RNA é representado pelos seus pesos iniciais, pois serão os primeiros valores a serem ajustados. Iniciando os pesos iniciais aleatoriamente, podem-se ter valores próximos dos ideais, que com poucos ajustes o treinamento se completa, como se podem ter valores distantes dos ideais, os quais necessitarão de muitos ajustes, precisando de mais iterações e consequentemente mais tempo

computacional (HAYKIN, 2001). Isto é outro problema na hora de avaliar as RNA's candidatas a solução no processo evolutivo do AG.

O ideal seria que a estrutura de otimização encontrasse a quantidade ideal mínima de neurônios sem necessidade do completo treinamento (ou quase completo) da RNA. A estrutura de busca deve ter a capacidade de direcionar o próximo passo com poucas iterações de treinamento de uma RNA. Não ter esta capacidade implica ser mais interessante utilizar um critério empírico, reconhecido pela literatura, e treinar a RNA até o objetivo final.

Dentro deste contexto, torna-se necessário desenvolver uma forma de busca e otimização que encontre a quantidade ideal mínima de neurônios na camada escondida com o menor tempo de treinamento das RNA's possível. Para isso acontecer, é preciso que as redes sejam avaliadas qualitativamente, ou seja, pelo quanto elas estão representando e aprendendo do problema, pois do contrário, terão que ser treinadas até um erro global pré-estabelecido para serem avaliadas.

Em Zárate (1998) é apresentada uma proposta de uso da Análise de Sensibilidade para avaliar qualitativamente a capacidade de aprendizado de uma RNA com base nos fatores de sensibilidade obtidos por meio da diferenciação dos pesos de uma RNA sendo treinada. Partindo do mesmo princípio, analisar os fatores de sensibilidade de uma RNA com treinamento incompleto é uma forma de avaliar qualitativamente o quanto a rede aprendeu do processo até o momento do teste.

Trata-se como quantidade ideal de neurônios na camada escondida, um número qualquer de neurônios que seja suficiente para que a RNA aprenda a relação entre os conjuntos de entrada e saída, sem perder em generalização. Por outro lado, a quantidade ideal mínima de neurônios na camada escondida, é o menor número de neurônios possível que a camada escondida pode possuir e ainda conseguir aprender a relação entre os conjuntos de entrada e saída, sem perder em generalização.

Neste trabalho, a análise dos fatores de sensibilidade, com base no conhecimento do especialista no domínio do problema, será usada para encontrar a quantidade ideal mínima de neurônios na camada escondida de uma RNA Perceptron Multicamadas através dos AG's. Para que isso aconteça, é necessário incorporar ao mecanismo evolutivo do AG regras simbólicas que representam o conhecimento do especialista. O conhecimento do especialista será usado na fase de avaliação das RNA's, avaliando o quanto as redes representam do comportamento qualitativo do processo. Os Fatores de Sensibilidade serão obtidos com a diferenciação dos pesos da RNA sendo treinada.

1.1 Definição do Escopo do Problema

Existe uma dificuldade para definir a quantidade mais adequada de neurônios na camada escondida de uma RNA. Ora o projetista usa sua experiência, ora usa alguma fórmula empírica ou um mecanismo heurístico. Na literatura, os trabalhos que usam os AG's avaliam RNA's pelo erro apresentado ao final do treinamento, ou pelo erro após apresentar um conjunto de teste, ou pela quantidade de épocas gastas com treinamento, ou pela capacidade de generalização, etc, ou todos estes juntos (SCHAFFER; WHITLEY; ESHELMAN, 1992), (BALAKRISHNAN; HONAVAR, 1995), (YAO, 1995), (FILHO; CARVALHO, 1997), (CANTU-PAZ; KAMATH, 2005). Para que as RNA's sejam avaliadas elas são treinadas várias e exaustivas vezes até se encontrar uma estrutura ideal. Entretanto, sabe-se que é possível monitorar o aprendizado não somente pelo erro, mas independente da forma de monitoramento do aprendizado, é importante que seja feito no menor tempo possível.

Buscando melhorar este processo e propor uma nova estrutura evolucionária mais adequada e padronizada, neste trabalho, busca-se a quantidade ideal mínima de neurônios na camada escondida de uma RNA Perceptron Multicamadas avaliando o aprendizado das RNA's pelo conhecimento do especialista através Análise dos Fatores de Sensibilidade usando AG. A exploração do problema de pesquisa será apresentada no Capítulo 3.

1.2 Hipótese e Premissas

Hipótese (H_1): Usar o conhecimento do especialista expresso através dos fatores de sensibilidade, como um critério de avaliação das RNA's dentro do mecanismo evolutivo dos AG's, pode permitir encontrar mais rapidamente a quantidade ideal mínima de neurônios na camada escondida de uma RNA Perceptron Multicamadas, do que o procedimento tradicional de tentativa e erro.

Todos os experimentos deste trabalho foram realizados com uma RNA Perceptron Multicamadas aplicada ao Processo de Laminação a Frio apresentada no Anexo 01. Todos os treinamentos que foram realizados respeitam as seguintes premissas:

Premissa I: Deve ser estabelecido um erro médio tolerável para o processo de treinamento. Neste trabalho foi pré-estabelecido um erro médio de 10^{-4}, sobre os valores normalizados, o que equivale a aproximadamente 2% sobre o valor real.

Premissa II: Deve-se garantir que as condições iniciais das redes sejam as mesmas. Para isso os pesos iniciais devem iniciar todos com zero.

Premissa III: Uso de um algoritmo de treinamento que possui uma capacidade de convergência alta (rápida), pois iniciando os pesos com 0 (zero) é necessário um algoritmo que consiga ajustar os pesos com eficiência. O algoritmo utilizado é uma variação do Backpropagation tradicional: Levenberg-Marquardt.

1.3 Objetivo Geral

O objetivo geral deste trabalho consiste em encontrar a quantidade ideal mínima de neurônios na camada escondida de uma Rede Neural Artificial Perceptron Multicamadas avaliando o aprendizado das RNA's, pelo conhecimento do especialista no domínio do problema, através da Análise dos Fatores de Sensibilidade, incorporado ao mecanismo evolutivo dos Algoritmos Genéticos.

1.4 Objetivos Específicos

Para alcançar o objetivo geral deste trabalho, alguns objetivos específicos devem ser atingidos:

- Mapear o espaço de busca para diferentes quantidades de neurônios na camada escondida de uma RNA Perceptron Multicamadas.
- Apresentar uma representação do conhecimento do especialista no domínio do problema baseada em regras simbólicas mapeadas através dos fatores de sensibilidade.
- Propor uma estrutura baseada em Algoritmos Genéticos e fatores de sensibilidade para uso no processo de busca da quantidade ideal mínima de neurônios na camada escondida da RNA Perceptron Multicamadas.

- Validação da nova estrutura para encontrar a quantidade ideal mínima de neurônios na camada escondida de uma RNA Perceptron Multicamadas aplicada ao Processo de Laminação a frio.

1.5 Plano de Desenvolvimento da Pesquisa

Os passos para conclusão desta pesquisa são apresentados adiante:

- Levantamento bibliográfico para estudo das Redes Neurais Artificiais, Análise de Sensibilidade e Algoritmos Genéticos.
- Implementação de uma RNA Perceptron Multimadas para ser usada como objeto de pesquisa nas simulações. O algoritmo de treinamento da RNA implementada é uma variação do algoritmo Backpropagation tradicional: Backpropagation Levenberg-Marquardt.
- Treinamento de RNA's com diferentes quantidades de neurônios na camada escondida até atingirem um erro pré-determinado de 10^{-4}.
- Realização das simulações das RNA's usando conjuntos que não fizeram parte do treinamento e avaliação do comportamento.
- Implementação do algoritmo para cálculo dos fatores de sensibilidade.
- Utilização dos fatores de sensibilidade na criação das regras simbólicas para representação do conhecimento do especialista.
- Geração do Mapa de sensibilidade composto por várias simulações de RNA's avaliadas pelo quanto cada uma representa o comportamento do processo real por meio das regras simbólicas.
- Implementação do Algoritmo Genético para ser usado como mecanismo de busca e otimização.
- Uso do AG para estimar a quantidade ideal mínima de neurônios na camada escondida de uma RNA usando os critérios encontrados na literatura.
- Incorporação das regras simbólicas que representam o conhecimento do especialista no processo evolutivo do AG e uso para encontrar a quantidade ideal mínima de neurônios na camada escondida de uma RNA.
- Aplicação em uma RNA usada no Processo de Laminação a frio.
- Apresentação e análise descritiva dos resultados encontrados.

1.6 Organização da Dissertação

Para leitores experientes em AG's e RNA's sigam diretamente a seção 2.3.9 e o Anexo 01, respectivamente, para rápido entendimento do AG e da RNA usadas neste trabalho. Para os demais leitores sigam a leitura conforme a organização do trabalho apresentada a seguir.

Esta dissertação está organizada na forma que se segue: neste capítulo é apresentada a introdução da pesquisa, assim como seu escopo. No Capítulo 2, é descrito o Referencial Teórico com a fundamentação de Redes Neurais Artificiais, Algoritmos Genéticos e Fatores de Sensibilidade, juntamente com os trabalhos relacionados a esta pesquisa. No Capítulo 3 são apresentadas as avaliações experimentais do problema a ser tratado. No Capítulo 4 a estrutura evolucionária proposta neste trabalho é descrita. No Capítulo 5 é mostrada a aplicação da estrutura proposta ao Processo de Laminação a Frio e os resultados das simulações. Por fim, as conclusões são apresentadas.

2 Referencial Teórico

2.1 Trabalhos Relacionados

Nos últimos anos, foram propostos vários modelos combinando Algoritmos Genéticos e Redes Neurais Artificiais. Tanto os AG's quanto as RNA's representam duas tecnologias que possuem um crescente interesse nas pesquisas de diversas áreas como engenharia, computação, economia, dentre outras. Este interesse é motivado pelos recentes avanços de ambos, pois possuem um comportamento dinâmico, além de poderem ser usados para resolver problemas complexos. (SCHAFFER; WHITLEY; ESHELMAN, 1992).

Dentre as combinações entre AG's e RNA's, pode-se considerar duas abordagens importantes. A primeira, trata-se do uso do AG para treinar ou para ajudar no treinamento de uma RNA. Nesta abordagem, o AG é usado para encontrar os pesos ideais da rede neural, ou para reduzir a quantidade de entradas da rede neural, no qual selecionará (identificará) as entradas que possuem as características mais relevantes para o processo em questão. O segundo, trata-se do uso do AG para projetar uma estrutura ideal da rede neural. Visto que, uma forma largamente usada para determinação do número e do tamanho das camadas escondidas é a base de tentativa e erro. Nesta segunda abordagem, o AG's são usados para buscar os parâmetros ideais da rede neural, bem como suas conexões (CANTU-PAZ; KAMATH, 2005).

O treinamento de uma RNA é um processo de otimização, onde o objetivo é encontrar um conjunto de pesos que minimiza o erro apresentado pela saída da rede neural, dado um conjunto de entrada. Neste processo de otimização, o espaço de busca é grande e, dependendo do erro tolerável e dos conjuntos de treinamento, podem existir vários ótimos locais.

Tratando-se de algoritmos como o backpropagation (BP), os quais utilizam algum tipo de gradiente de pesquisa, estes podem ficar presos em regiões ótimas locais. Possuindo grande dificuldade em conseguir partir em direção a uma região ótima global. Contrária a esta limitação, os AG's não trabalham com técnicas de gradiente, e são susceptíveis a não ficarem presos em uma região ótima local. Isso porque trabalham simultaneamente com várias regiões do espaço de busca. Logo, a combinação entre algoritmos genéticos e redes neurais é uma forma interessante para encontrar os pesos da rede neural, para que seu comportamento seja o desejado (XUSHENG LU; BOURBAKIS, 1998), (POUR; ATLASBAF; HAKKAK, 2006). Segundo Cantu-Paz e Kamath (2005), neste caso, a arquitetura da rede é fixada pelo

projetista, além disso, Schaffer; Whitley e Eshelman (1992) recomendam que cada cromossomo genético represente um vetor com todos os pesos da rede.

Em Prados (1992) é apresentado um método eficiente baseado em AG's para ajustar os pesos de uma RNA multicamadas chamado *GenLearn*, onde sua principal característica é conseguir evitar o congelamento do treinamento quando se encontra um ponto local ótimo, comportamento característico de algoritmos que trabalham com regra delta.

Além de ajustar os pesos de uma RNA, os AG´s podem selecionar (identificar) as principais características dos conjuntos de entrada da RNA. Os conjuntos de treinamento podem possuir características que são irrelevantes ou redundantes. A princípio, o projetista geralmente desconhece quais são as características relevantes e quais são as irrelevantes. Contudo, necessita-se de uma maneira para descartar características irrelevantes e redundantes, não apenas para evitar uma rede neural superdimensionada, mas também para aumentar a precisão do aprendizado.

Segundo Yao (1995), a evolução das características de entrada é usada para reduzir e/ou combinar todas as possibilidades existentes de entrada de uma rede neural até encontrar um subconjunto reduzido e eficiente. Esse tipo de solução evolutiva é necessária quando se possui um conjunto muito grande de características (variáveis) do problema, necessitando de um mecanismo automático de busca.

Cantu-Paz e Kamath (2005) sugerem como modelagem genética para seleção de um subconjunto de entrada relevante, a utilização de uma codificação binária, onde o cromossomo possui um bit para cada atributo e a quantidade de bits representa a quantidade de características usadas na classificação. A partir de então, cada cromossomo é avaliadado pelo resultado do treinamento de uma rede neural, com estrutura pré-determinada, usando o subconjunto de características representado pelo seu cromossomo. Em Ozdemir et al. (2001), foi usada outra abordagem; neste, cada gene do cromossomo representa todas as características usadas na classificação, e o tamanho do cromossomo é definido pela quantidade de genes que ele representa. Em Guo e Uhrig (1992), foi usada a mesma modelagem genética apresentada por Cantu-Paz e Kamath (2005), onde os algoritmos genéticos foram utilizados encontrar uma combinação ótima de variáveis para as redes neurais para atingir os critérios: poucas entradas, rápido treinamento e aprendizado preciso.

A segunda abordagem, em combinações entre AG's e RNA's, é a definição da estrutura da rede neural, ponto importantíssimo, pois esta afeta intensamente o desempenho final da rede. Caso possua neurônios e conexões a menos, não irá aprender todo o conhecimento necessário. Caso contrário, com neurônios e conexões a mais, os conjuntos de treinamento

serão decorados e será perdida a capacidade de generalização (WHITLEY; KARUNANITHI, 1991).

Em Koza e Rice (1991) os AG's são usados para encontrar os pesos e a arquitetura de uma RNA, buscando o número de camadas, o número de neurônios por camada, e a conectividade entre os neurônios.

Em Schaffer, Whitley e Eshelman (1992), é apresentada uma estrutura de uso dos AG's na definição da estrutura da RNA onde, deve-se definir a representação genética da RNA (genótipo) dentro do processo evolutivo e um decodificador para mapear a representação genética em uma estrutura real (fenótipo). Em seguida, deve existir um mecanismo que, a partir da representação fenotípica, crie e treine a RNA. Para que em seguida, estas redes sejam avaliadas, recebendo uma nota (fitness) de acordo com seu desempenho. E para que a RNA evolua, usa-se o mecanismo evolutivo do AG para geração de novas estruturas, a partir das estruturas antigas.

Balakrishnan e Honavar (1995) apresentam graficamente um modelo evolutivo da estrutura de uma RNA, veja Figura 1. Nela, cada genótipo representa uma solução candidata do problema, neste caso a estrutura de uma RNA. Os genótipos, dentro de um processo de decodificação, são transformados em fenótipos (RNA's reais). Criadas as RNA's, estas devem ser treinadas por um algoritmo de aprendizado, como por exemplo, o algoritmo backpropagation, até uma quantidade de épocas pré-determinadas. Completada a fase de treinamento, o erro médio quadrático é usado como valor de fitness para cada genótipo. A partir de então, os genótipos com melhor fitness possuem mais probabilidade de serem selecionados para passarem pelos operadores genéticos: crossover e mutação.

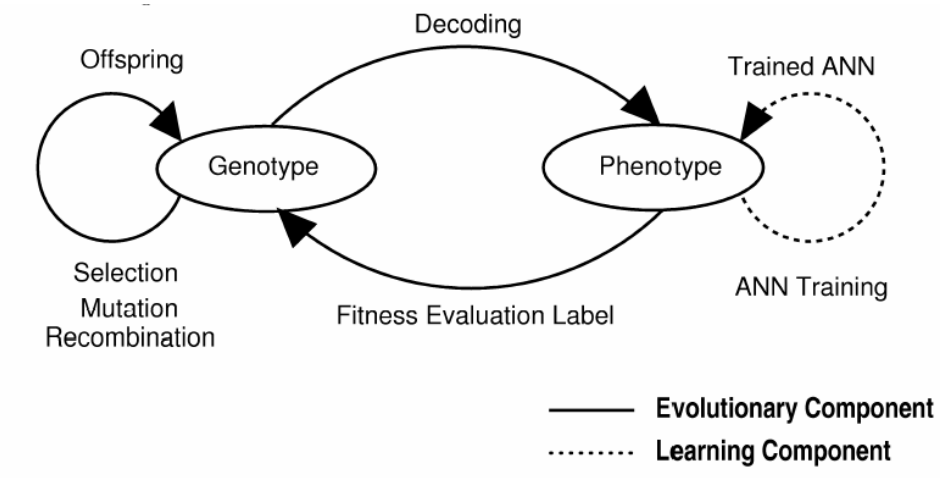

Figura 1 – Modelo evolutivo da estrutura de uma RNA

Fonte: (BALAKRISHNAN; HONAVAR, 1995)

Segundo Yao (1995), o processo evolutivo de estruturas de RNA's é um processo de adaptação automático para diferentes tarefas, sem que exista a intervenção do projetista. Em Yao (1995) é apresentado um ciclo típico da evolução de estruturas de RNA's onde, em cada geração os indivíduos devem ser decodificados para concepção da RNA que ele representa. Em seguida, todas as RNA's devem ser treinadas usando um algoritmo de aprendizado pré-determinado, o qual iniciará com valores aleatórios os pesos os demais parâmetros de aprendizado (taxa de aprendizado e momentum). Completo o processo anterior, as RNA's recebem uma nota (fitness), de acordo com o resultado do treinamento. O fitness de cada RNA é usado como critério de seleção das estruturas mais adequadas para reproduzirem e gerarem descendentes após aplicação dos operadores genéticos. Feito isto, inicia-se uma nova geração.

Vários trabalhos utilizam abordagens semelhantes às apresentadas anteriormente. A diferença entre elas reside na aplicação da abordagem evolutiva para projeto de uma estrutura otimizada para problemas específicos e, diversas outras abordagens cujo propósito é encontrar um mecanismo evolutivo eficiente (EBERHART; DOBBINS, 1991), (MANIEZZO, 1994), (BRANKE, 1995), (LIU; YAO, 1996), (YAO; LIU, 1997), (HÜSKEN; JIN; SENDHOFF, 2002), (HAYWARD, 2004).

Por outro lado, a análise de sensibilidade tem sido associada a mecanismos de otimização, como por exemplo, em Avila et al. (2006), onde são resolvidos alguns problemas de otimização de dispositivos eletromagnéticos. Para isso, foi usada a análise de sensibilidade por meio de uma avaliação a partir dos dados fornecidos por um algoritmo genético multiobjetivo, permitindo a identificação de quais parâmetros (de solução) são mais sensíveis a desvios.

Segundo Xiaoqin Zeng e Yeung (2001) estudos de análise de sensibilidade em RNA's Perceptrons Multicamadas possuem duas abordagens. A abordagem analítica, na qual a análise de sensibilidade é definida como a derivada parcial da saída pela entrada da rede neural. E a abordagem feita estatisticamente; nesta, a sensibilidade é definida como a taxa entre o desvio-padrão dos erros de saída pelo desvio-padrão do peso ou erros dada uma entrada, onde o erro tende a zero. As duas abordagens são úteis quando se precisa analisar o erro de saída com relação a um determinado padrão de entrada.

2.2 Redes Neurais Artificiais

2.2.1 Introdução

As Redes Neurais Artificiais (RNA's) são modelos matemáticos inspirados nas estruturas neurais biológicas. A implementação computacional de uma RNA pode aprender informações que lhe são apresentadas e generalizar seu aprendizado (HAYKIN, 2001).

O aprendizado de uma RNA normalmente é associado a sua capacidade de adaptar seus parâmetros como conseqüência de sua interação com o ambiente externo (conjuntos de treinamento). No processo de aprendizado a RNA aumenta seu desempenho gradativamente, dentro de um ciclo iterativo, à medida que interage com o ambiente externo. O desempenho de uma RNA, durante o processo de aprendizado, é usado tanto para determinar a qualidade do modelo neural computacional quanto como ponto de término do processo de treinamento. Em ambos, o erro quadrático médio das respostas da RNA para um determinado conjunto de dados é largamente usado.

Para que o processo de aprendizagem aconteça, utiliza-se um procedimento chamado algoritmo de aprendizagem, o qual modifica os pesos sinápticos da rede ordenadamente visando atingir um objetivo estipulado na fase de concepção da RNA (HAYKIN, 2001).

A generalização de uma RNA representa a capacidade da rede em apresentar respostas coerentes para conjuntos de dados não apresentados a ela durante o processo de treinamento. Um bom processo de treinamento trata a generalização juntamente com o treinamento da RNA e não como conseqüência dele.

No geral, uma RNA é uma estrutura projetada para modelar a forma como o cérebro humano realiza tarefas individuais ou em conjunto, podendo ser implementada computacionalmente através de softwares simuladores ou, utilizando-se de componentes eletrônicos. Independente da implementação as RNA's empregam uma interligação completa de suas células computacionais simples, chamadas neurônios ou unidades de processamento. Estas células simples são responsáveis pelo processamento das informações, o qual pode ser paralelo e distribuído (HAYKIN, 2001).

O neurônio artificial é o elemento de processamento da RNA e em McCulloc e Pits (1943) foi apresentado o modelo de um neurônio artificial. As entradas do neurônio correspondem ao vetor de entrada $X = [x_1, x_2, ..., x_n]^T$ de dimensão n. Para cada uma das

entradas x_i, há um peso correspondente w_i. A soma das entradas x_i multiplicadas por seus respectivos pesos w_i representa a saída linear u do neurônio, onde $u = \sum_{i=1}^{n} w_i x_i$. A saída y do neurônio, conhecida como saída de ativação, é obtida com a aplicação de uma função $f(.)$ diferenciável à saída linear u, representada por $y = f(u)$, veja Figura 2.

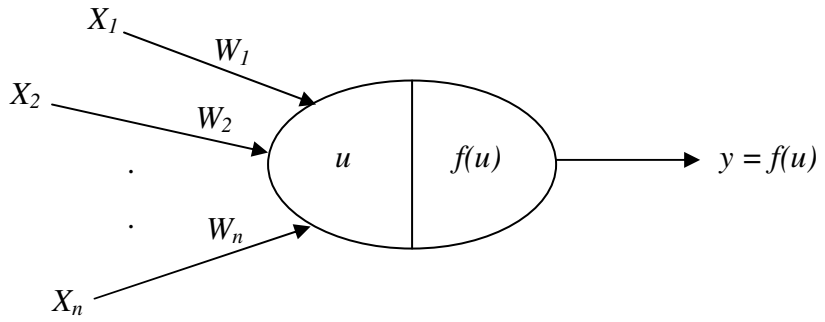

Figura 2 - Modelo do Neurônio McCulloch e Pitts

Logo, uma RNA é formada por neurônios artificiais simples, onde cada neurônio executa uma atividade simples, mas a RNA completa possui capacidade computacional para resolver problemas complexos. Uma RNA feed-forward totalmente conectada como na Figura 3, possui vários neurônios, onde sua estrutura é formada por cinco entradas ($[x_1, x_2, x_3, x_4, x_5]$), duas saídas ($[y_1, y_2]$) e quatro neurônios na camada intermediária. Esta estrutura, conforme dito por Braga, Carvalho e Ludemir (2000) é capaz de solucionar problemas de classificação, predição ou regressão, neste caso, no espaço R^5.

Uma particularidade importante das RNA's é delas serem aproximadores universais de funções multivariáveis contínuas (GIROSI; POGGIO, 1990). Isso quer dizer que uma RNA pode resolver qualquer problema de aproximação de funções contínuas, independentemente da quantidade de variáveis. Neste caso, para a RNA, deve-se definir o número de entradas e saídas, que estão diretamente relacionados à dimensão dos dados. Além disto, deve-se estipular a quantidade de neurônios na camada escondida, o que é um ponto crucial para o bom desempenho da RNA, visto que o número de neurônios nas camadas intermediárias depende da complexidade do problema. Ou seja, quanto maior for a complexidade das funções a serem mapeadas, maior deverá ser a quantidade de neurônios na camada intermediária. Por outro lado, quanto menor for a complexidade, menos neurônios serão necessários.

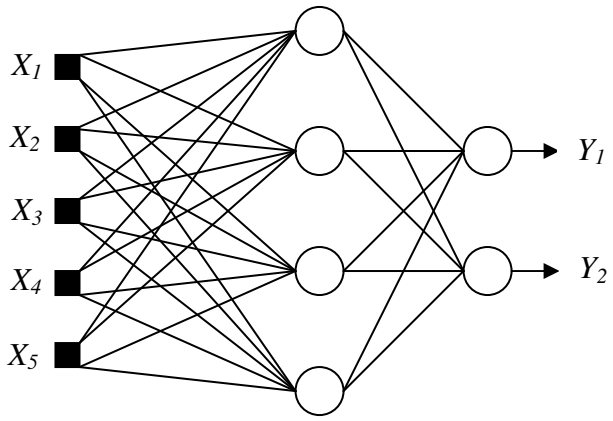

Figura 3 - RNA feed-forward totalmente conectada

Um problema que os projetistas enfrentam é a definição da quantidade ideal de neurônios na camada intermediária, pois uma quantidade excessiva de neurônios nesta camada pode levar a resultados indesejáveis, como overfitting. O overfitting ocorre quando se possui uma quantidade elevada de neurônios na camada intermediária, mais que o suficiente, o que leva a RNA decorar os padrões ao invés de extrair suas principais características, prejudicando fortemente a generalização do seu aprendizado.

2.2.2 RNA Perceptron Multicamadas

Uma RNA perceptron multicamadas, também conhecida como *Multilayer Perceptron*, possuem poder computacional muito maior que uma RNA perceptron com apenas uma camada de neurônios apresentada por Rosenblatt (1958). Isso devido ao fato de uma perceptron multicamadas (MLP) ter a capacidade de tratar dados que não são linearmente separáveis. Girosi e Poggio (1990) dizem que uma MLP com duas camadas intermediárias pode implementar qualquer função, linearmente separável ou não.

As redes MLP têm sido usadas com sucesso na solução de diversos problemas de difícil solução, utilizando em seu treinamento supervisionado um algoritmo conhecido como algoritmo de retropropagação do erro (error backpropagation). Tipicamente uma rede MLP é formada por um conjunto de neurônios compondo a camada de entrada, outro conjunto de neurônios distribuídos em uma ou mais camadas intermediárias (ocultas), e por fim um conjunto de neurônios que representa a camada de saída da rede.

Haykin (2001) apresenta três características distintivas que uma RNA perceptron multicamadas possui:

1. O modelo de cada neurônio da RNA contém uma função de ativação não-linear. A função de ativação não-linear mais usada por possuir uma não-linearidade suave (diferenciável em qualquer ponto) é a função log-sigmóide, também conhecida como função logística:

$$y_j = \frac{1}{1 + e^{-vj}} \tag{01}$$

   A utilização de uma função não-linear é importante, pois caso contrário, a relação entrada-saída da rede pode ser reduzida a de um perceptron com apenas uma camada. Além disso, a função logística possui toda uma motivação biológica devido a sua capacidade de representar a fase refratária dos neurônios reais.

2. A rede possui uma ou mais camadas de neurônios intermediários (ocultos), as quais não são partes das camadas de entrada e nem da saída. As camadas intermediárias permitem que a rede aprenda situações complexas, extraindo gradativamente as principais características dos conjuntos de entrada.

3. A rede possui um alto grau de conectividade, formando as sinapses da rede.

Em uma rede MLP o processamento realizado por cada neurônio é definido pela combinação dos processamentos realizados pelos neurônios da camada anterior, aos quais estão conectados. Quando seguimos da primeira camada intermediária rumo a camada de saída, as funções implementadas pela rede tornam-se cada vez mais complexas, e são elas que definem a divisão do espaço de decisão mapeado. Por exemplo, como dito em Braga, Carvalho e Ludemir (2000) uma rede com duas camadas intermediárias, pode-se dizer que o processamento em cada uma das camadas ocorre da seguinte maneira:

- **Primeira camada intermediária**: cada um dos neurônios traça retas no espaço de padrões de treinamento.
- **Segunda camada intermediária:** cada neurônio combina as retas traçadas pelos neurônios da camada anterior conectados a ele, formando regiões convexas, onde o número de lados é definido pelo número de neurônios conectados a ele.

- **Camada de saída:** cada neurônio forma regiões que são combinações das regiões convexas formadas pelos neurônios da camada anterior conectados a ele. Neste caso, estes nodos formam regiões com formatos abstratos.

Embora o poder de representação de uma RNA aumentar quando aumentamos o número de camadas intermediárias, não é recomendada a utilização de um grande número destas camadas. Pois muitas camadas dificultam o treinamento da rede, exigem maior esforço computacional, e a cada vez que o erro calculado no treinamento é propagado para a camada anterior ele se torna menos útil e preciso. No processo de retroprogação do erro apenas a camada de saída possui uma noção precisa do erro, já a última camada intermediária possui uma estimativa do erro, em seguida a penúltima camada intermediária possui a estimativa da estimativa do erro, e assim por diante (BRAGA; CARVALHO; LUDEMIR, 2000).

Outro ponto relevante é a quantidade de neurônios na camada intermediária, sendo definido geralmente de forma empírica. A quantidade de neurônios na camada intermediária depende fortemente da distribuição dos padrões de treinamento e de fatores como:

- Quantidade de padrões de treinamento;
- Quantidade de ruído presente nos padrões de treinamento;
- Complexidade da função a ser aprendida;
- Distribuição estatística dos padrões de treinamento;

Enfim, existem problemas que são resolvidos com apenas a camada de entrada e a camada de saída, já outros exigem várias camadas intermediárias. Deve-se analisar bem o problema e verificar qual a melhor quantidade de camadas e neurônios por camada devem ser definidos no projeto da RNA.

Neste trabalho, pretende-se criar um mecanismo para que se possa definir, qualitativamente, a quantidade ideal mínima de neurônios na camada escondida de uma RNA perceptron multicamadas com uma camada escondida, ajudando ao projetista a definir a estrutura ideal para seu problema.

2.2.3 Algoritmo Backpropagation

O algoritmo mais conhecido para o treinamento de redes multicamadas é o backpropagation. Ele é um algoritmo de aprendizado supervisionado que trabalha com pares

(entrada, saída desejada) que são usados para ajustar os pesos da rede, por meio de um mecanismo de correção de erros. O treinamento ocorre em duas fases (fase forward e fase backward), na fase forward a saída da rede para um dado conjunto de treinamento é gerada, já na fase backward a saída desejada é comparada a saída da rede para que seja encontrado o erro e este usado para atualizar os pesos das conexões da rede (HAYKIN, 2001).

O algoritmo backpropagation busca minimizar o erro obtido pela rede, ajustando os pesos e bias (limiares) para que correspondam às coordenadas mais baixas da superfície de erro. Para que isto ocorra, este algoritmo utiliza o método gradiente descendente (BRAGA; CARVALHO; LUDEMIR, 2000).

A seguir será apresentado o algoritmo backpropagation para uma RNA perceptron multicamadas com uma camada escondida (LIPPMANN, 1988).

Seja um conjunto de treinamento: $(X_{1,}Y_1),...,(X_p,Y_p)$, para cada conjunto de treinamento:

1. Calcular a saída linear da camada escondida:

$$net_j^h = \sum_{i=0}^{N} W_{ji}^h X_i \quad j = 1,...,L \tag{02}$$

2. Calcular a saída da camada escondida:

$$I_j = f_j^h(net_j^h) \quad j = 0,...,L \tag{03}$$

3. Calcular a saída linear da camada de saída:

$$net_j^o = \sum_{i=0}^{L} W_{ji}^o I_i \quad j = 1,...,M \tag{04}$$

4. Calcular a saída da rede:

$$Y_k = f_j^o(net_j^o) \quad j = 1,...,M \tag{05}$$

5. Calcular o erro da camada de saída:

$$\Gamma_k^o = (\psi_k - Y_k) f_k^{o'}(net_k^o) \quad k = 1,...,M \tag{06}$$

6. Calcular o erro da camada escondida:

$$\Gamma_k^h = f_j^{h'}(net_j^h) \sum_{k=1}^{M} \Gamma_k^o W_k^o \qquad j = 1,...,L \tag{07}$$

7. Atualizar os pesos da camada de saída:

$$W_{kj}^o(t+1) = W_{kj}^o(t) + \mu \Gamma_k^o I_j \qquad k = 1,...,M \qquad j = 0,...,L \tag{08}$$

8. Atualizar os pesos da camada escondida:

$$W_{ji}^h(t+1) = W_{ji}^h(t) + \mu \Gamma_j^h X_j \qquad j = 1,...,L \qquad i = 0,...,N \tag{09}$$

9. Verificar a tolerância:

$$E_p = \frac{1}{2}\sum_{k=1}^{M} \delta_{pk}^2 < tolerância \tag{10}$$

2.2.4 Algoritmo Levenberg-Marquardt

Este algoritmo é uma variação do backpropagation tradicional, diferenciando-se por ser uma extensão do método de Newton. O Levenberg-Marquardt é conhecido por possuir um ajuste de pesos rápido e eficiente (HAGAN e MENHAJ, 1994). Neste algoritmo, os pesos são ajustados considerando a razão entre as primeiras e segundas derivadas ou a razão entre o vetor gradiente e a matriz hessiana.

$$\Delta w_{jk} = -\frac{\nabla \varepsilon\big|_{w_n}}{H_{\varepsilon}\big|_{w_n}} \tag{11}$$

Quando se tem uma função objetivo quadrática, pode-se demonstrar que:

$$\nabla \varepsilon\big|_{w_n} = J^T(n)e(n) \tag{12}$$

$$H_{\varepsilon}\big|_{w_n} = J^T(n)J(n) \tag{13}$$

Onde *J(n)* é a matriz das primeiras derivadas da função do erro em relação aos pesos e bias da rede, ou matriz Jacobiana. Quando substituímos Eq.12 e Eq.13 em Eq.11, tem-se a seguinte equação:

$$\Delta w_{jk} = -[J^T(n)J(n)]^{-1} J^T(n)e(n) \tag{14}$$

Na equação acima, o termo $[J^T(n)J(n)]^{-1}$ é a maneira simplificada para se obter a matriz hessiana a partir das primeiras derivadas. Mas, para que o mecanismo de otimização funcione realmente, deve-se garantir que a matriz hessiana seja definida positiva para todo *n*. Para que isso aconteça, basta adicionar a matriz diagonal μI ao termo $[J^T(n)J(n)]^{-1}$ na Eq.14. Com isso, o método de otimização Levenberg-Marquardt é obtido (HAGAN e MENHAJ, 1994):

$$\Delta w_{jk} = -[J^T(n)J(n) + \mu I]^{-1} J^T(n)e(n) \quad (15)$$

Segundo Ngia e Sjoberg (2000) o algoritmo de treinamento Levenberg-Marquardt é superior aos demais algoritmos de treinamento off-line, pois além de possuir rápida convergência, esta acontece com boa generalização.

Para Hagan e Menhaj (1994) o algoritmo Levenberg-Marquardt é muito eficaz quando usado em redes que têm até algumas centenas de pesos. Embora os requisitos computacionais sejam muito mais elevados para cada iteração do algoritmo, este custo é proporcional a sua grande eficiência quando colocado em execução. Isto pode ser verificado quando a precisão de convergência exigida é alta.

Neste trabalho, foi usado o algoritmo Levenberg-Marquardt por atender as 03 (três) premissas definidas no Capítulo 01. Em especial, a terceira premissa, a qual determina o uso de um algoritmo de treinamento que possui uma alta capacidade de convergência, pois o ajuste de pesos deve ser feito com eficiência, visto que todos os pesos iniciam com valor 0 (zero).

2.3 Algoritmos Genéticos

2.3.1 Introdução

Os Algoritmos Genéticos (AG) pertencem a um grupo de técnicas computacionais, conhecido como Computação Evolutiva ou Evolucionária, tais técnicas fundamentam-se no desenvolvimento de modelos computacionais que apresentam uma representação genética e simulam a evolução natural dos seres vivos. Isto é possível através de uma representação matemática e algorítmica de alguns conceitos apresentados por Charles Darwin e de genética. Os AG's são, dentre os ramos da Computação Evolucionária, os mais conhecidos, e tiveram origem no trabalho de John Holland ainda na década de 60 e, depois com a publicação de seu livro Adaptation in Natural and Artificial Systems em 1975 (SRINIVAS; PATNAIK, 1994).

Futuramente, com a publicação do livro Genetic Algorithms in Search, Optimization and Machine Learning de David Goldberb a idéia de otimização ocupou lugar central nas pesquisas e teoria dos AG's (BAYER; LUI WANG, 1991).

Os AG's vem se aprimorando, e cada vez mais, sendo aplicados a problemas de busca e otimização em diversas áreas, apresentando bons resultados em áreas como engenharia, robótica, finanças empresariais, logística, gerenciamento de redes, otimização multicritério, ciências biológicas, telecomunicações, etc (BAYER; LUI WANG, 1991).

As técnicas de busca e otimização normalmente apresentam duas características importantes. A primeira é o universo de busca, onde serão encontradas as possíveis soluções do problema. A segunda é a função objetivo (função de aptidão), a qual é usada para avaliar todas as soluções encontradas, atribuindo uma nota a cada uma delas.

A técnica de busca e otimização implementada por um AG trabalha sobre uma população de candidatos em paralelo, realizando buscas em diferentes regiões do universo de soluções. Dessa forma, são altas as chances de se encontrar regiões mais promissoras (KOZA, 1995).

De acordo com Lacerda e Carvalho (1999) o AG é diferenciado de um método de busca e otimização tradicional principalmente por quatro aspectos.

- Trabalham com codificação dos parâmetros do problema e não com esses parâmetros diretamente;
- Trabalham com candidatos em paralelo (população) ao invés de um único candidato;

- Utilizam funções de aptidão ao invés de derivadas ou outras informações do processo de busca;
- Utilizam transições probabilísticas ao invés de determinísticas.

Embora o AG seja uma ferramenta que apresente bons resultados a sua utilização possui dois pontos de alta relevância que devem ser levados em consideração (KOZA, 1995):

- As soluções para o problema devem permitir sua codificação;
- Para a medição da solução quanto à sua eficiência é necessário uma forma de avaliação de cada solução apresentada.

Um AG possui etapas distintas capazes realizar as operações necessárias para o processo de busca e otimização e cada uma dessas etapas pode sofrer variações conforme o problema a ser solucionado a fim de melhorar o desempenho do AG.

2.3.2 Processo iterativo de um Algoritmo Genético

O funcionamento básico de um algoritmo genético é apresentado na Figura 4. Um processo iterativo onde são analisadas, em cada etapa, um grupo de possíveis soluções do problema em questão.

Inicialmente uma população aleatória é gerada, em seguida os indivíduos serão avaliados através de uma função de avaliação e logo após, selecionados os melhores. Os que não foram selecionados são descartados. Dentre os selecionados, eles irão reproduzir gerando descendentes. Todos os indivíduos da nova população poderão sofrer mutação, alterando assim suas características e diversificando a população (GOLDBERG, 1989).

Os indivíduos que compõem a nova população serão avaliados e o processo se repete até que uma solução ótima, ou próxima de ótima, seja encontrada entre os membros da população.

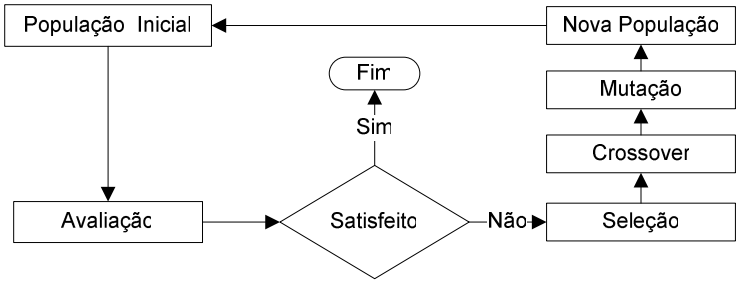

Figura 4 - Funcionamento básico do AG

Cada iteração desse processo é chamada de geração, assim a cada geração, os indivíduos mais aptos são selecionados para gerarem uma nova população. Dentro do processo, podem-se utilizar técnicas para garantir que as melhores soluções já encontradas passem inalteradas para nova população, garantindo assim, que o resultado já encontrado não seja perdido em uma próxima geração (LACERDA; CARVALHO, 1999).

Novas gerações ocorrem até que um critério de parada seja satisfeito, ou quando a solução ótima é encontrada. Cada evento do processo será mais detalhado nas próximas seções.

2.3.3 População

Em um AG, o conjunto de soluções possíveis para um problema recebe o nome de população. Nela, cada indivíduo é identificado como um cromossomo, o qual representa uma solução candidata para o problema. Um cromossomo pode possuir um ou mais genes, que representarão as diversas características do indivíduo (GOLDBERG, 1989).

A população inicial deve ser criada após a definição da quantidade de indivíduos que irão compor a população. A geração da população inicial pode ser aleatória ou usando conhecimento prévio do projetista, isso quando o universo de busca é conhecido (BAYER; LUI WANG, 1991). Quando uma população é gerada aleatoriamente, alguma área do universo de busca pode não ser representada; uma técnica utilizada para minimizar esse problema, quando se possui uma população de cromossomos binários, é gerar a primeira metade da população aleatoriamente e a segunda metade partindo da primeira, invertendo os bits do gene, o bit “0” é trocado para o bit “1” e vice versa. Dessa forma, pode-se garantir que cada gene do cromossomo ou posição da cadeia de bits possua os dois valores representados.

O número de indivíduos que compõem a população é uma característica importante na implementação de um AG. Esse número varia de acordo com o problema, e normalmente é definido empiricamente com base em testes iniciais, visto que não há uma regra que defina o tamanho ideal da população. O tamanho da população está diretamente ligado ao desempenho de um AG, quando se possui uma população pequena o desempenho do AG é baixo, pois algumas regiões do universo de busca não serão representadas. Porém, trabalhando com uma população grande, aumenta-se a diversidade de soluções, o que consequentemente irá requerer mais tempo de avaliação, necessitando de maior esforço computacional, podendo elevar o tempo de processamento. Diante disso, deve-se definir o tamanho da população com cuidado para que seja garantida a diversidade da população, sem comprometer o desempenho do AG na busca pela solução do problema (COSTA; SIMÕES, 2004).

O número de indivíduos da população é mantido durante o processamento do AG, sendo que a cada geração a população pode ser parcialmente ou completamente renovada.

2.3.4 Representação do Cromossomo

O primeiro ponto a ser definido para o uso de um AG é a representação do cromossomo, pois será definida a forma na qual o problema será representado para ser trabalhado pelo AG. A importância desta definição para todo o processo reside no fato do funcionamento dos operadores genéticos dependerem da escolha da representação do cromossomo (KOZA, 1995).

Como visto anteriormente o AG trabalha com um conjunto de indivíduos (cromossomos), no qual cada um corresponde a uma solução candidata para um determinado problema. Lacerda e Carvalho (1999) afirmam que normalmente o tamanho do cromossomo representa a quantidade de parâmetros utilizados na função objetivo (função de aptidão), e o total de configurações que um cromossomo pode assumir representam o seu universo de busca. Em um cromossomo com "*n*" parâmetros de uma função, tem-se um universo de busca de "*n*" dimensões.

A representação de um cromossomo pode ser feita de diversas maneiras: através de números reais, números inteiros, caracteres ou pela representação binária, etc. A escolha da representação do cromossomo deve ser feita para facilitar a representação das características do indivíduo e a manipulação pelos operadores genéticos. A Tabela 1exemplifica esses modos de representação.

Tabela 1 - Exemplos de representação do cromossomo

Representação	Exemplo
Números binários	Cromossomo = [0 1 0 1 0 1]
Números Inteiros	Cromossomo = [-22 ; 35 ; 7]
Números reais	Cromossomo = [5,3 ; -9,8 ; 4,2]
Caracteres	Cromossomo = [A B C D E]
Outros	Diferentes representações

A representação tradicional e também mais comum para os cromossomos é a binária. Nesta, cada cromossomo é representado por uma seqüência de bits 1 e 0. Em um cromossomo binário cada célula é um gene e os alelos são os valores assumidos por cada gene, veja Figura 5. Normalmente, a representação binária é mais fácil de ser manipulada e interpretada.

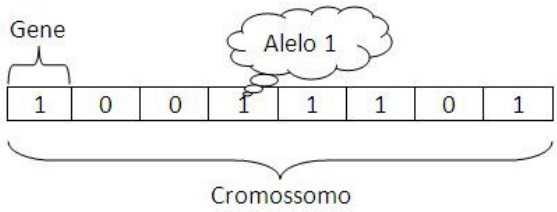

Figura 5 - Representação binária de um cromossomo

2.3.5 Função Objetivo (fitness)

A função objetivo tem a finalidade de mensurar quanto, determinado indivíduo, está próximo da solução ótima do problema, ou seja, mensurar seu grau de aptidão (fitness). A função recebe como entrada valores de acordo com os genes do cromossomo e devolve o fitness do indivíduo. Como a função objetivo está ligada diretamente ao problema, necessita-se de uma função específica para cada problema.

O grande desafio na implementação de um AG está ligado a situações onde a função objetivo do problema não é conhecida, nesse caso é necessário a utilização de uma heurística eficaz para avaliar a solução candidata. A aptidão (fitness) de um indivíduo é uma característica importante para o processo de seleção dos cromossomos que irão para a fase de reprodução (LACERDA; CARVALHO, 1999).

2.3.6 Métodos de Seleção

O processo de seleção é responsável por selecionar dentre a população de indivíduos os mais aptos à solução do problema em cada geração. Os indivíduos selecionados irão se reproduzir e passarão suas características para a próxima geração. A probabilidade do indivíduo ser selecionado dentre os outros indivíduos da população é diretamente proporcional ao seu grau de aptidão (fitness), porém isso não é uma regra e dependerá do método de seleção adotado. Alguns métodos de seleção não permitem exclusividade dos mais aptos a fim de permitir a diversidade da população (KOZA, 1995) (THIERENS; GOLDBERG, 1994).

Existem vários métodos de seleção, neste trabalho serão apresentados 03 (três) dos mais utilizados: método de ranking linear, método da roleta e método do torneio.

2.3.6.1 Método de Ranking Linear

Considerado o método de seleção mais simples, onde os indivíduos são ordenados em um rank por sua aptidão absoluta. A partir de então, os indivíduos mais aptos são selecionados para passarem pelos operadores genéticos gerando descendentes que irão formar a população intermediária. Este método é utilizado para garantir que os melhores indivíduos de uma determinada geração não deixem de gerar descendentes, levando suas características para as próximas gerações (BACK, 1994). Apesar de parecer o mais correto a se fazer, a utilização do ranking linear pode acarretar na diminuição da diversidade da população e propiciar uma convergência prematura.

2.3.6.2 Método da Roleta

O método de seleção mais utilizado é o método da roleta. Este método é probabilístico, baseado em uma roleta do jogo de azar, onde cada indivíduo é representado em um disco ocupando uma fatia proporcional a sua aptidão relativa. Consequentemente, os indivíduos com maior aptidão relativa representarão as maiores fatias no disco da roleta e terão maior probabilidade de serem selecionados. A Figura 6 representa hipoteticamente a ocupação de oito indivíduos no disco da roleta.

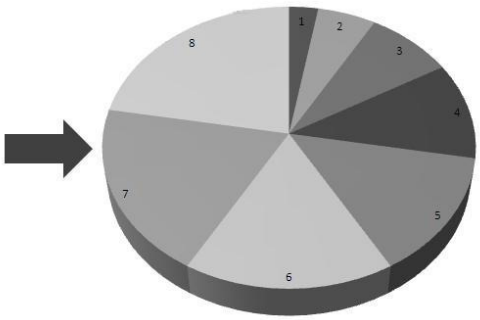

Figura 6 - Exemplo de seleção do método da roleta

Os indivíduos são selecionados pelo giro da roleta, o indivíduo apontado pela seta demonstrada na Figura 6 é encaminhado para a população intermediária. No exemplo, o indivíduo 07 (sete) foi selecionado para compor a população intermediária. O giro da roleta para seleção acontece quantas vezes for necessário até preencher a população intermediária completamente (BACK, 1994).

Uma variante do método da roleta seria a implementação do elitismo em um número pequeno dos indivíduos, garantindo assim a diversidade da população proposta pelo método da roleta e a preservação das melhores soluções pelo elitismo (GOLDBERG, 1989).

2.3.6.3 Método do Torneio

O método do torneio propicia uma maior diversidade da população do que o método roleta, apesar de menos praticado por ser relativamente mais aleatório (BACK, 1994). Nesse método, escolhe-se aleatoriamente um número de indivíduos. Dentre esses indivíduos, seleciona-se o de maior fitness para compor a população intermediária. Usualmente são escolhidos 03 (três) indivíduos aleatoriamente e retirado o de maior fitness. O processo é reiniciado até que a população intermediária esteja preenchida. Caso não haja um trabalho de elitismo para proteger as melhores soluções, o desempenho do AG pode não ser o esperado (THIERENS; GOLDBERG, 1994).

2.3.7 Operadores de Reprodução

Dada uma população, para que seja possível gerar populações sucessivas é necessário usar operadores de reprodução, visando melhorar a aptidão dos indivíduos ao longo do tempo (gerações). Estes operadores são: cruzamento (crossover) e mutação. Logo, a função principal dos operadores de reprodução é transformar a população por meio de sucessivas gerações, desdobrando a busca até se encontrar uma solução satisfatória.

Os indivíduos da população intermediária são submetidos ao operador de reprodução (crossover), o qual tem a função de gerar novos indivíduos. Para a aplicação do crossover, necessita-se de um par de cromossomos denominados cromossomos pais; para cada par de pais são gerados dois novos indivíduos denominados de cromossomos filhos, os quais carregam as características dos cromossomos pais selecionados para a operação de crossover (BACK; HAMMEL; SCHWEFEL, 1997).

As recombinações genéticas dos cromossomos devem respeitar a probabilidade predefinida de aplicação do operador de crossover conhecida como "taxa de crossover". Conforme apresentado por Bayer e Lui Wang (1991), essa taxa pode variar entre 50% e 100%. Caso o crossover não seja aplicado, os cromossomos filhos são exatamente iguais aqueles selecionados como cromossomos pais (LACERDA; CARVALHO, 1999). Uma taxa de crossover baixa demanda mais tempo para gerar novos indivíduos, o que pode acarretar uma estagnação, ou um tempo maior que o normal, para encontrar uma solução satisfatória. Por outro lado, quando se usa uma taxa de crossover alta, o AG tende a eliminar os indivíduos com boa aptidão rapidamente, pois novos indivíduos são introduzidos na população ligeiramente. Sendo assim, a taxa de crossover deve ser estudada para cada problema específico e, apesar de incomum, pode variar durante a execução do AG, a fim de melhorar o seu desempenho (GOLDBERG, 1989).

Os métodos mais conhecidos são crossover de um ponto, crossover multipontos e crossover uniforme. Estes serão apresentados a seguir.

2.3.7.1 Crossover de um Ponto

O crossover de um ponto é o método de recombinação que utiliza apenas um corte no cromossomo. Depois de selecionados os cromossomos pais, esses são cortados em um ponto gerado aleatoriamente, sendo o mesmo ponto de corte para ambos. Em seguida, cada pai troca

a segunda parte de seu cromossomo com o outro, resultando em dois descendentes que possuem parte do cromossomo de um pai e parte de outro. Os dois filhos consequentemente são diferentes dos pais, mas carregam parte das características dos progenitores em seu cromossomo (COSTA; SIMÕES, 2004). A Figura 7 demonstra o crossover de um ponto.

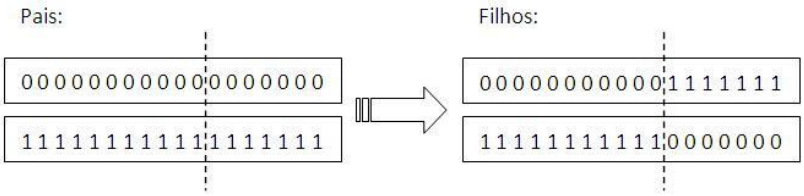

Figura 7 - Funcionamento do crossover de um ponto

2.3.7.2 Crossover Multipontos

O crossover multipontos ocorre de maneira semelhante ao crossover de um ponto, porém dois ou mais pontos de corte são gerados aleatoriamente e os cromossomos pais selecionados trocam os genes entre si, como ocorre no crossover de um ponto. A utilização de mais de dois pontos de corte não é comum, mas caso o aumento do número de cortes proporcione ao AG um desempenho melhor, esse pode ser utilizado. Deve-se tomar cuidado com o aumento excessivo do número de cortes, o que pode levar à perda de desempenho do AG, uma vez que pode destruir a cadeia genética do cromossomo (COSTA; SIMÕES, 2004).

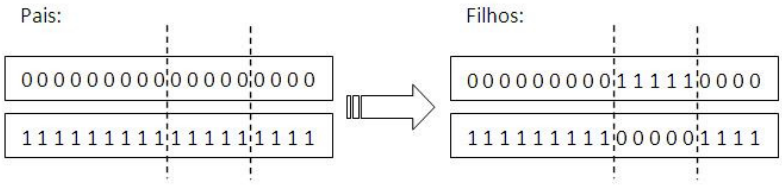

Figura 8 - Funcionamento do crossover de dois pontos

O local dos pontos de corte dos pais devem ser os mesmos, assim como ocorre no crossover de um ponto, no exemplo da Figura 8 foram gerados 02 (dois) pontos aleatórios de corte e o material genético entre esses dois pontos são trocados para a geração de dois cromossomos filhos. A recombinação do primeiro filho acontece utilizando a parte do cromossomo anterior ao primeiro corte do primeiro pai com a parte do cromossomo que está entre o primeiro e o segundo corte do segundo pai e, completando-se o cromossomo com a

parte após o segundo corte do primeiro pai. O segundo filho é gerado de maneira semelhante ao primeiro.

2.3.7.3 Crossover Uniforme

Um terceiro método de recombinação é chamado de crossover uniforme, neste os paradigmas de pontos de corte utilizados no crossover de um ponto e multipontos são abandonados e, utiliza-se um novo método capaz de determinar qual cromossomo pai fornecerá suas características para o cromossomo filho. Essa característica, como pode ser observada na Figura 9 difere os métodos de recombinação anteriores (um ponto e multipontos) do crossover uniforme. Esse método utiliza uma máscara de bits gerados aleatoriamente para determinar como vai ocorrer a recombinação. A máscara gerada possui o mesmo tamanho do cromossomo, cada posição da máscara representa um gene do cromossomo original que determina qual dos pais fornecerá ao filho o seu gene (COSTA; SIMÕES, 2004).

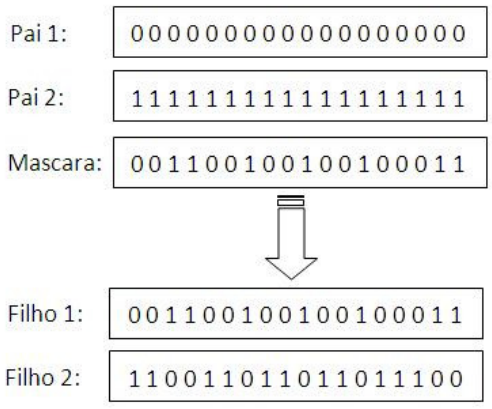

Figura 9 - Funcionamento do crossover uniforme

No exemplo da Figura 9 a geração do primeiro filho respeita um procedimento onde o bit 0 da máscara indica a utilização do gene respectivo do primeiro pai e o bit 1 indica a utilização do gene do segundo pai, a geração do segundo filho utiliza um procedimento inverso onde o bit 0 indica o gene do segundo pai e o bit 1 indica o gene do primeiro pai.

2.3.7.4 Operador de Mutação

Apesar do crossover ser o principal operador de reprodução, responsável por inserir novos indivíduos na população, o AG dispõe de um segundo mecanismo diversificador chamado mutação. Este possui a capacidade de levar o AG a explorar locais do universo de busca menos prováveis de serem alcançados. A mutação ocorre com menos freqüência do que o crossover, visto que possui uma taxa de ocorrência que normalmente varia de 0.1% a 5%. Uma taxa baixa para esse operador impede que o efeito da mutação destrua indivíduos de boa qualidade para a solução do problema, sendo o suficiente para garantir a diversidade da população. Através da mutação, pode-se provocar o surgimento de novas características pontuais em alguns indivíduos diferentes das herdadas dos indivíduos pais (BACK; HAMMEL; SCHWEFEL, 1997) (COSTA; SIMÕES, 2004).

O operador de mutação é aplicado a cada cromossomo gerado pelo operador de crossover dentro de uma probabilidade predeterminada pela taxa de mutação. Seu funcionamento se baseia na varredura dos genes do cromossomo e através de um número gerado aleatório e comparado com a taxa de mutação determina se a troca do valor do gene ocorre ou não. Quando a representação cromossômica utilizada é a binária, a troca do valor do gene acontece na mudança do valor 0 para 1 ou vice versa. Em outros casos, quando o gene possui valores não-binários (como na representação com números reais) a troca acontece gerando um valor aleatório na faixa de valores admitidos pela representação utilizada. A Figura 10 ilustra um exemplo de aplicação do operador de mutação em um cromossomo com representação binária.

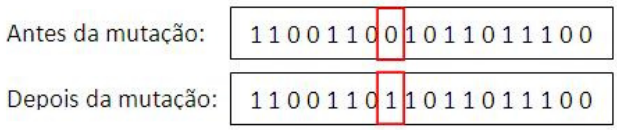

Figura 10 - Exemplo de aplicação do operador de mutação

2.3.8 Critérios de Parada

Um AG tem como objetivo a melhoria contínua dos indivíduos da população através de gerações sucessivas que levam em direção à um ponto ótimo. Isso pode ser evidenciado pelo

aumento da aptidão do melhor indivíduo e da aptidão média da população (RIBEIRO FILHO; TRELEAVEN; ALIPPI, 1994). Contudo, a utilização de um AG em problemas complexos pode levar à um custo computacional inviável, nestes casos os critérios de parada do AG devem ser definidos a fim de evitar esforço computacional elevado e/ou desnecessário.

Segundo Lacerda e Carvalho (1999) os critérios de parada mais comuns levam em consideração o tempo de processamento, o valor da aptidão do melhor indivíduo, o número de gerações ou a convergência a um ponto ótimo. Podendo ser definidos da seguinte maneira:

- O tempo de processamento de um algoritmo tem que ser monitorado em situações que se precisa da solução em pouco tempo. Neste caso, essa característica pode ser utilizada como critério de parada, porém deve-se observar que um critério de parada por tempo de processamento não garante que o AG tenha alcançado um resultado satisfatório;
- Quando o valor ótimo da função de aptidão é conhecido ele pode ser utilizado como critério de parada do AG, ou seja, quando a aptidão de um indivíduo atingir o valor ótimo o processamento pode ser interrompido. Podendo admitir valores próximos ao valor ótimo como valores satisfatórios;
- Outro critério seria estipular um número máximo de gerações, assim mesmo que o ponto ótimo não tiver sido alcançado o AG interrompe o processamento e evita um custo computacional elevado.

A convergência é uma característica interessante para o AG avaliar em meio a execução. Se após sucessivas gerações o valor de aptidão médio da população ou a aptidão dos melhores indivíduos não sofrer alteração ou variar dentro de um intervalo de valores muito próximos, por várias gerações consecutivas, pode-se concluir que foi encontrado um valor próximo do ótimo e o processo de busca pode ser interrompido (RIBEIRO FILHO; TRELEAVEN; ALIPPI, 1994).

2.3.9 Estrutura Genética usada neste trabalho

Os elementos de um AG mencionados anteriormente possuem grande influência no desempenho do algoritmo e devem ser considerados para obtenção de bons resultados.

Neste trabalho foi usada a representação binária para codificação do cromossomo; a população inicial é composta por 12 (doze) indivíduos que representam valores aleatórios distribuídos uniformemente dentro do espaço de busca; para compor uma nova população, os indivíduos são selecionados usando o método de seleção ranking linear, preservando os dois melhores indivíduos (elitismo); na fase de reprodução são aplicados o crossover de um ponto e mutação, respeitando as probabilidades de 100% e 0.5% respectivamente; como critério de parada foi usado à estabilidade dos índices de aptidão.

2.4 Fatores de Sensibilidade

Frequentemente não é possível obter uma relação direta entre um *parâmetro A* (causa) e uma *função B*(efeito). No entanto, pode-se às vezes correlacionar experimentalmente o *parâmetro A* com valores das derivadas da *função B* (efeito). Em Zárate (1998) é mostrado que a relação causa-efeito entre parâmetros de um processo pode ser expressa por fatores de sensibilidade $\partial y / \partial x$ via RNA. Onde cada fator de sensibilidade que representa um parâmetro A (causa) com um parâmetro B (efeito) serve para identificar qual das entradas (causa) apresenta maior influência na saída (efeito) do processo. Estes fatores podem servir para extrair o conhecimento e ajudar na avaliação do comportamento do processo de treinamento de uma RNA (ZÁRATE; BITTENCOUT, 2007a).

Antes de apresentar os fatores de sensibilidade, alguns termos, usados nas expressões serão mostrados:

U_i, $i = 0,...,N$ são as entradas da rede e $U_0 = 1$ é a entrada de polarização.

$f_i^a(.)$ $i = 0,...,N$ são as funções de normalização das entradas e $f_0^a(.) = 1$

X_i, $i = 0,...,N$ são as entradas normalizadas e $X_0 = U_0$

W_{ij}^h $i = 1,...,L$ e $j = 0,...,N$ contém o peso correspondente do neurônio j para cada entrada i.

$net_j^h = \sum_{i=0}^{N} W_{ji}^h X_i \quad j = 1,...,L$ produto de pesos pelas entradas.

$f_j^h(net_j^h) \quad j = 0,...,L$ com $f_0^h(net_0^h) = 1$ é a função sigmóide usada na camada escondida.

I_j, $j = 0,...,L$ são os valores resultantes da função sigmóide e $I_0 = 1$

W_{ij}^o $i = 1,...,M$ e $j = 0,...,L$ contém o peso do neurônio i e entrada j, para a camada de saída.

$net_j^o = \sum_{i=0}^{L} W_{ji}^o I_i \quad j = 1,...,M$ produto dos pesos pelas entradas, para a camada de saída.

$f_j^o(net_j^h) \quad j = 1,...,M$ é a função sigmóide usada na camada de saída.

Y_j, $j = 1,...,M$ são as saídas normalizadas da rede, obtidas da função sigmóide.

$f_i^b(.)$ $i = 1,...,M$ são as funções desnormalizadoras das saídas.

Z_i, $i = 1,...,M$ valores de saída da rede.

$e\max_k, e\min_k \; k = 1,..,N$ valores máximos e mínimos das entradas.

$s\max_k, s\min_k \; k = 1,..,M$ valores máximos e mínimos das saídas.

2.4.1 Equações de Sensibilidade

Neste trabalho, foi usada uma RNA perceptron multicamadas com uma camada escondida. A RNA possui N entradas, M saídas e L neurônios na camada escondida. A diferenciação da rede neural é genérica e depende somente de N, M, L e dos pesos das camadas escondida e saída, obtidos durante o processo de treinamento. O procedimento para obtenção das expressões de sensibilidade da rede foi deduzido em Zárate (1998), e será resumidamente apresentado a seguir:

$$\begin{aligned} Z_1 &= f_1^b(Y_1) \\ Z_2 &= f_2^b(Y_2) \\ &\vdots \\ Z_{\mathrm{M}} &= f_{\mathrm{M}}^{\mathrm{b}}(Y_{\mathrm{M}}) \end{aligned} \tag{16}$$

Trabalhando adequadamente as variáveis, obtém-se a Eq. (17) que relaciona as entradas com as saídas não normalizadas da RNA:

$$\begin{aligned} Z_1 &= f_1^b(f_1^o(\sum_{j=0}^{\mathrm{L}} W_{1j}^o f_j^h(\sum_{i=0}^{\mathrm{N}} W_{ji}^h f_i^a(U_i)))) \\ Z_2 &= f_2^b(f_2^o(\sum_{j=0}^{\mathrm{L}} W_{2j}^o f_j^h(\sum_{i=0}^{\mathrm{N}} W_{ji}^h f_i^a(U_i)))) \\ &\vdots \\ Z_{\mathrm{M}} &= f_{\mathrm{M}}^b(f_{\mathrm{M}}^o(\sum_{j=0}^{\mathrm{L}} W_{\mathrm{M}j}^o f_j^h(\sum_{i=0}^{\mathrm{N}} W_{ji}^h f_i^a(U_i)))) \end{aligned} \tag{17}$$

Substituindo as expressões para as funções $f^a(.), f^b(.), f^o(.), f^h(.)$, obtém-se a Eq. (18):

$$\begin{aligned} Z_1 &= \frac{1}{1+\exp^{-V_1}}[\mathrm{smax}_1 - \mathrm{smin}_1] + \mathrm{smin}_1 \\ Z_2 &= \frac{1}{1+\exp^{-V_2}}[\mathrm{smax}_2 - \mathrm{smin}_2] + \mathrm{smin}_2 \\ &\vdots \\ Z_{\mathrm{M}} &= \frac{1}{1+\exp^{-V_{\mathrm{M}}}}[\mathrm{smax}_{\mathrm{M}} - \mathrm{smin}_{\mathrm{M}}] + \mathrm{smin}_{\mathrm{M}} \end{aligned} \tag{18}$$

onde:

$$V_k = \sum_{j=0}^{\mathrm{L}} W_{kj}^{o} f_j^{h} \left(\sum_{i=0}^{\mathrm{N}} W_{ji}^{h} f_i^{a} (U_i) \right) \quad \text{para } k = 1,..., \mathrm{M} \tag{19}$$

Generalizando a Eq. (18), tem-se:

$$Z_k = \frac{1}{1+\exp^{-V_k}} \left[\mathrm{smax}_k - \mathrm{smin}_k \right] + \mathrm{smin}_k \text{ para } \mathrm{k} = 1,.., \mathrm{M} \tag{20}$$

Os fatores de sensibilidade podem ser calculados pela Eq. (21):

$$\frac{\partial Z}{\partial U} = \begin{bmatrix} \frac{\partial Z_1}{\partial U_1} & \frac{\partial Z_1}{\partial U_2} & \cdots\cdots & \frac{\partial Z_1}{\partial U_{\mathrm{N}}} \\ \cdots\cdots & \cdots\cdots & \cdots\cdots & \cdots\cdots \\ \frac{\partial Z_{\mathrm{M}}}{\partial U_1} & \frac{\partial Z_{\mathrm{M}}}{\partial U_2} & \cdots\cdots & \frac{\partial Z_{\mathrm{M}}}{\partial U_{\mathrm{N}}} \end{bmatrix} \tag{21}$$

onde cada termo da matriz de sensibilidade pode ser calculado pela Eq. (22):

$$\frac{\partial Z_k}{\partial U_i} = [\mathrm{smax}_k - \mathrm{smin}_k] \frac{\exp^{-V_k}}{(1+\exp^{-V_k})^2} \frac{\partial V_k}{\partial U_i} \tag{22}$$

Trabalhando com termo derivativo da Eq. (22) e considerando a Eq. (19), a seguinte expressão é obtida:

$$\frac{\partial V_k}{\partial U_i} = \frac{\partial}{\partial U_i} \left(W_{k0}^{o} + \sum_{j=1}^{\mathrm{L}} W_{kj}^{o} f_j^{h} \left(\sum_{i=0}^{\mathrm{N}} W_{ji}^{h} f_i^{a} (U_i) \right) \right) \tag{23}$$

Diferenciando a Eq. (23) e introduzindo o resultado na Eq. (22), obtém-se a Eq. (24), a qual permite calcular os fatores de sensibilidade a partir de uma RNA previamente treinada:

$$\frac{\partial Z_k}{\partial U_i} = \begin{bmatrix} R_1 W_{11}^o & R_1 W_{12}^o & \cdots & R_1 W_{1L}^o \\ R_2 W_{21}^o & R_2 W_{22}^o & \cdots & R_2 W_{2L}^o \\ \vdots & \vdots & \vdots & \vdots \\ R_M W_{M1}^o & R_M W_{M2}^o & \cdots & R_M W_{ML}^o \end{bmatrix}$$

$$\begin{bmatrix} \frac{Q_1 W_{11}^h}{\text{emax}_1 - \text{emin}_1} & \frac{Q_1 W_{12}^h}{\text{emax}_2 - \text{emin}_2} & \cdots & \frac{Q_1 W_{1N}^h}{\text{emax}_N - \text{emin}_N} \\ \frac{Q_2 W_{21}^h}{\text{emax}_1 - \text{emin}_1} & \frac{Q_2 W_{22}^h}{\text{emax}_2 - \text{emin}_2} & \cdots & \frac{Q_2 W_{2N}^h}{\text{emax}_N - \text{emin}_N} \\ \vdots & \vdots & \cdots & \vdots \\ \frac{Q_L W_{L1}^h}{\text{emax}_1 - \text{emin}_1} & \frac{Q_L W_{L2}^h}{\text{emax}_2 - \text{emin}_2} & \cdots & \frac{Q_L W_{LN}^h}{\text{emax}_N - \text{emin}_N} \end{bmatrix} \quad (24)$$

com:

$$Q_k = \frac{\exp^{-\left(\sum_{i=0}^{N} W_{ki}^h X_i\right)}}{\left(1 + \exp^{-\left(\sum_{i=0}^{N} W_{ki}^h X_i\right)}\right)^2} \qquad k = 1,..., L$$

$$R_k = (\text{smax}_k - \text{smin}_k) \frac{\exp^{-V_k}}{(1 + \exp^{-V_k})^2}$$

para $k = 1,.., M$

e:

$$W^o = \begin{bmatrix} W_{11}^o & W_{12}^o & \cdots & W_{1L}^o \\ W_{21}^o & W_{22}^o & \cdots & W_{2L}^o \\ \vdots & \vdots & \vdots & \vdots \\ W_{M1}^o & W_{M2}^o & \cdots & W_{ML}^o \end{bmatrix}$$

3 Avaliação Experimental do Problema de Pesquisa sendo Tratado

3.1 Avaliação dos Treinamentos pelo Erro

Um mecanismo muito usado para analisar o aprendizado da RNA é avaliação do erro apresentado na saída da rede ao final de uma época de treinamento. Nesta seção, várias RNA's serão treinadas exaustivamente, até atingirem um erro tolerável estipulado previamente.

O objetivo dos treinamentos exaustivos foi analisar o número de épocas necessárias para um completo treinamento das RNA's ao variar o número de neurônios na camada escondida. Para estes experimentos foi considerado o processo de laminação a frio (veja Anexo 01).

Inicialmente, 127 RNA's foram treinadas com diferentes quantidades de neurônios na camada escondida (veja Anexo 02). A primeira RNA com 01 neurônio na camada escondida e a última com 127 neurônios. Esta quantidade máxima de neurônios na camada escondida foi definida empiricamente com objetivo de representar um universo com mais de uma centena de valores para análise. No entanto, aumentar demais a quantidade de neurônios na camada escondida pode acarretar o overfitting, ou seja, a rede neural irá decorar os conjuntos de treinamento, inclusive algum ruído caso exista, prejudicando a generalização do aprendizado. Por outro lado, uma RNA com poucos neurônios na camada escondida pode não conseguir mapear todo o universo desejado (underfitting). Como já dito por Filho e Carvalho (1997) é difícil estimar uma quantidade de neurônios na camada escondida de uma RNA; até mesmo para testes, por tão evidente que é este problema.

Na Figura 11 não foi apresentada à quantidade de épocas para o treinamento com 01 neurônio na camada escondida, pois foram gastas 2.695 épocas (veja Anexo 02), a apresentação deste ponto elevado no gráfico condensaria graficamente os demais resultados, por isso o eixo "*y*" foi redimensionado para o intervalo de 0 a 120 épocas.

Uma primeira observação após os treinamentos com variação da quantidade neurônios na camada escondida da RNA, como pode ser visto na Figura 11, foi o reflexo variado na quantidade de épocas necessárias para o aprendizado da rede. Não houve uma relação direta ou proporcional referente ao aumento de neurônios na camada escondida. Observam-se duas regiões distintas na Figura 1, a primeira região dentro dos intervalos com [02 – 41] e [75 -

127] neurônios na camada escondida, variando entre 14 e 53 épocas de treinamento; a segunda região dentro do intervalo com [42 – 74] neurônios, variando entre 70 e 101 épocas de treinamento.

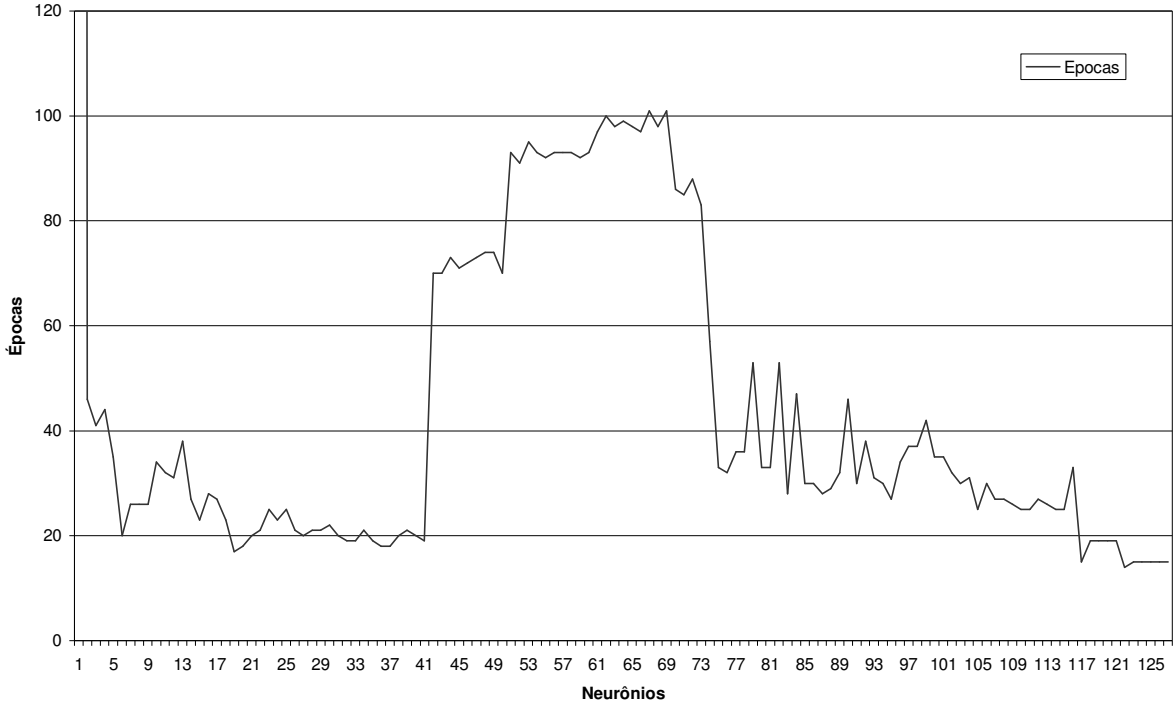

Figura 11 – Épocas de Treinamento das RNA's com 01 até 127 neurônios na camada escondida

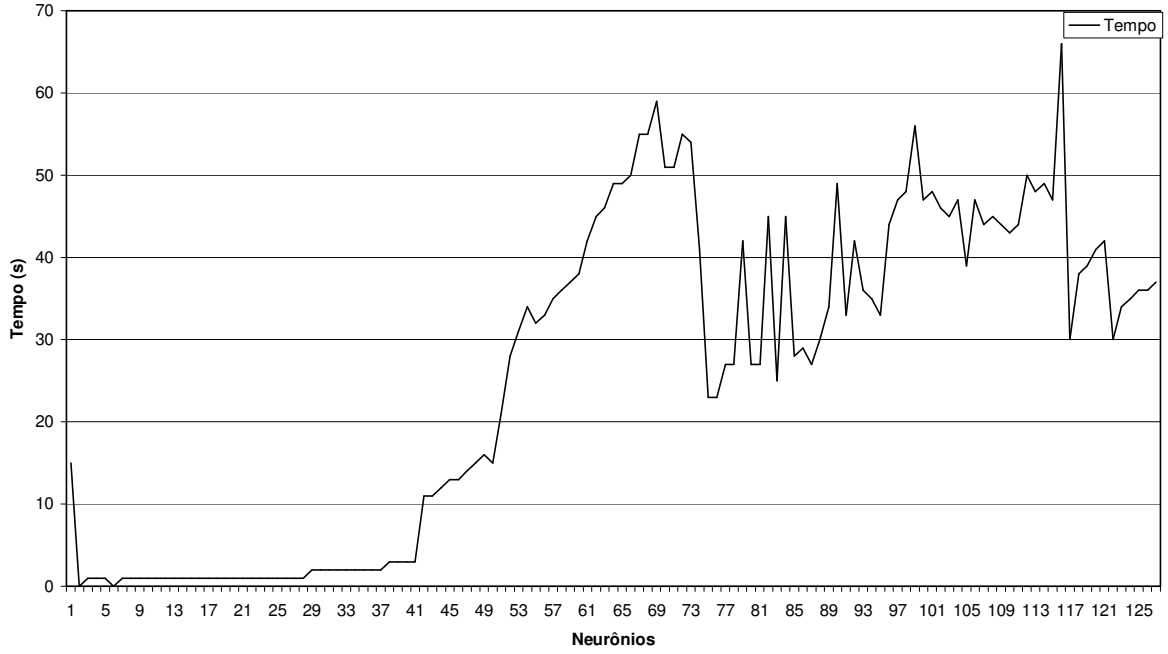

Figura 12 - Tempo de Treinamento das RNA's com 01 até 127 neurônios na camada escondida

Já com o tempo gasto no treinamento houve uma tendência proporcional de crescimento quando se aumenta a quantidade de neurônios na camada escondida, mas apenas uma tendência, pois a partir da rede com 69 neurônios o tempo começa a cair e volta a crescer com a rede que possui 79 neurônios, veja Figura 12.

Após os treinamentos realizados no Anexo 02, as RNA's foram testadas com 50 conjuntos desconhecidos, ou seja, que não fizeram parte do processo de treinamento. Tanto no processo de treinamento quanto no de validação, as redes são avaliadas pelo erro, o qual representa a diferença entre o valor esperado pelo valor resultante da RNA.

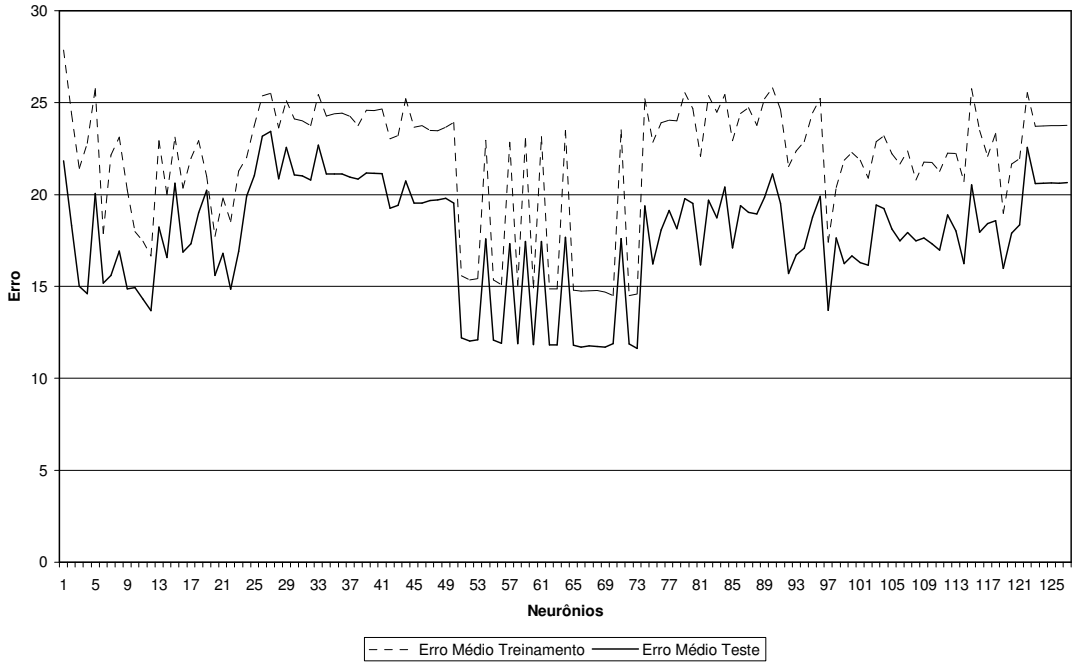

Figura 13 - Erro Médio dos Treinamentos e Teste (50 conjuntos) das RNA's com 01 até 127 neurônios na camada escondida

Na Figura 13 são apresentados os erros médios das 127 redes ao final do treinamento e ao final do teste com 50 conjuntos que não fizeram parte do treinamento. Pode-se ver o desempenho de todas as redes a medida que se aumenta a quantidade de neurônios na camada escondida. Nitidamente, vê-se que o desempenho das redes quando colocadas em operação foi melhor que ao final do treinamento.

Considerando o menor erro médio após o processo de treinamento como parâmetro para indicar a RNA que melhor aprendeu o comportamento do processo, encontrar-se-ia a RNA com 70 neurônios na camada escondida. Por outro lado, se for buscado o menor erro médio apresentado no teste com 50 conjuntos desconhecidos, encontrar-se-ia a RNA com 73 neurônios na camada escondida, veja Tabela 2.

Tabela 2 - Resumo dos Treinamentos e Teste com 50 conjuntos desconhecidos

Avaliação	Valor Real (Kgf/mm)	Neurônios
Melhor erro médio ao final dos treinamentos	14,50	70
Pior erro médio ao final dos treinamentos	27,87	01
Melhor erro médio ao final das simulações	11,62	73
Pior erro médio ao final das simulações	23,43	27
Tempo total das 127 simulações (min.)	54,94	

Ainda observando a Tabela 2, vê-se que após o treinamento até atingir um erro médio global de 10^{-4}, a RNA que possuía apenas 01(um) neurônio na camada escondida apresentou o pior erro médio ao final do treinamento. Mas, embora tenha apresentado o pior valor e tenha gastado mais épocas para treinar (veja Anexo 02), quando colocada em operação submetida a um teste em que lhe foram apresentados 50 conjuntos, que não foram usados no treinamento, esta apresentou um erro médio menor que o maior erro médio de todas as redes. A RNA com 27 neurônios na camada escondida foi quem apresentou o pior erro médio no teste. Isso mostra que com um neurônio na camada escondida a RNA não conseguiu aprender exatamente todos os conjuntos de treinamento, mas foi o suficiente para que pudesse generalizar o que aprendeu.

Este resultado é interessante, pois mostra que com apenas 01 neurônio na camada escondida, a rede apresentou desempenho razoável quando colocada em operação.

Em um outro treinamento, até atingir um erro médio global de 10^{-4}, as redes foram testadas com 15.625 conjuntos que não fizeram parte do processo de treinamento (veja Anexo 02). Logo, a RNA que possuía 70 neurônios na camada escondida apresentou o menor erro médio tanto ao final do treinamento quanto ao final do teste, veja Tabela 3.

Na Figura 14 são apresentados os erros médios das 127 redes ao final do processo de treinamento e ao final do teste com 15.625 conjuntos que não fizeram parte deste processo.

O resultado é bem próximo do apresentado na Tabela 2 com teste de 50 conjuntos, mantendo a quantidade de neurônios na camada escondida para o melhor erro médio ao final do treinamento e caindo de 73 para 70 a quantidade de neurônios para o melhor erro médio ao final das simulações. É importante observar que neste segundo teste a quantidade de conjuntos é muito grande, expondo a rede treinada a um universo bem amplo de possibilidades que ela não viu durante seu treinamento.

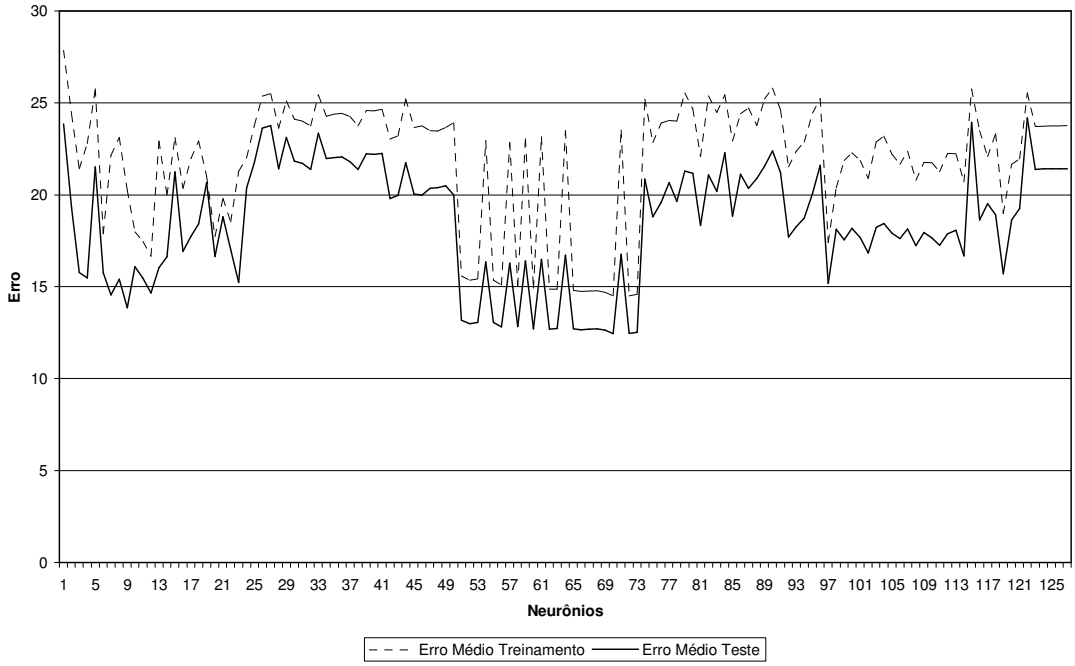

Figura 14 - Erro Médio dos Treinamentos e Teste (15.625 conjuntos) das RNA's com 01 até 127 neurônios na camada escondida

Além disto, assim como na simulação anterior com 50 conjuntos de teste, a RNA com apenas 01 neurônio na camada escondida apresentou o pior erro médio ao final do treinamento e gastou a maior quantidade de épocas para treinar. Mas também apresentou um erro médio menor que o maior erro médio de todas as redes. A RNA com 122 neurônios na camada escondida foi a quem teve o pior erro médio no teste.

Tabela 3 - Resumo dos Treinamentos e Teste com 15.625 conjuntos desconhecidos

Avaliação	Valor Real (Kgf/mm)	Neurônios
Melhor erro médio ao final dos treinamentos	14,50	70
Pior erro médio ao final dos treinamentos	27,87	01
Melhor erro médio ao final das simulações	12,45	70
Pior erro médio ao final das simulações	24,19	122
Tempo total das 127 simulações (min.)	58,44	

Observa-se que com 01 neurônio na camada escondida, a RNA tem dificuldade em refinar o ajuste dos pesos e reduzir o erro, mas o ajuste alcançado para um erro tolerável de 10^{-4}, é o suficiente para que possa generalizar seu aprendizado quando colocada em operação.

Pode-se dizer que embora a RNA com 70 neurônios na camada escondida apresente o melhor valor não se pode ignorar que RNA's com menos neurônios na camada escondida também aprenderam e respondem bem quando testadas, veja Figura 13 e Figura 14.

Embora a rede com 01 neurônio tenha gasto mais épocas para treinar, levou menos tempo (15 segundos) para completar o treinamento. Já a rede com 70 neurônios gastou menos épocas para treinar, mas levou mais tempo (52 segundos) para completar o treinamento. Isso devido ao esforço computacional necessário para manipular tantos neurônios.

Enfim, se as redes que foram treinadas forem avaliadas pela menor quantidade de épocas gastas no treinamento, escolher-se-ia a RNA com 122 neurônios na camada escondida. Se forem avaliadas pelo menor tempo de treinamento, escolher-se-ia a RNA com 02 ou com 06 neurônios. Mas se forem avaliadas pelo menor erro ao final do treinamento, escolher-se-ia a RNA com 70 neurônios. Por último, avaliando pelo erro no teste perante conjuntos que não fizeram parte do treinamento, escolher-se-ia a RNA com 70 ou 73 neurônios na camada escondida. Diante disto, é possível observar a existência de uma grande diversidade de possibilidades.

3.2 Avaliação dos Treinamentos pela Análise dos Fatores de Sensibilidade

O entendimento de um processo físico, químico, econômico, etc; pode ser conseguido através do estudo das relações das variáveis envolvidas no processo. Esta relação pode ser expressa por equações analíticas utilizando os princípios da física, da química, da economia, etc.

Um modelo é uma representação dos aspectos essenciais de um processo, que proporciona conhecimento e informação utilizável. O modelo analítico (dominante) permite analisar a causa-efeito de variações nas variáveis do processo. Em muitos casos, os modelos dominantes podem ser complexos, quando levado em consideração todos os aspectos da física que os governam. Em situações onde é necessária informação de um modelo dominante para melhorar a resposta de um sistema de inferência ou de tomada de decisão, a incorporação desses modelos podem não ser as mais adequadas. Sendo assim é interessante obter novas estratégias para introduzir o comportamento quantitativo ou qualitativo do processo que venha a reforçar a ação de um sistema para a tomada de decisão.

A incorporação desse conhecimento pode ser feita através dos fatores de sensibilidade (que neste trabalho serão obtidos através das redes neurais sendo treinadas), veja Tabela 4.

Tabela 4 - Fatores de sensibilidade

Entrada / Saída	x_1	x_2	.....	x_n
y_1	$\partial y_1 / \partial x_1$	$\partial y_1 / \partial x_2$	.....	$\partial y_1 / \partial x_n$
.....	.....	.....	.....	.....
y_m	$\partial y_m / \partial x_1$	$\partial y_2 / \partial x_2$	.....	$\partial y_m / \partial x_n$

Os fatores de sensibilidade usados neste trabalho permitiram a criação de um modelo qualitativo do processo de laminação a frio. Isso foi possível com a abstração das equações diferenciais criando regras simbólicas que representam o comportamento do processo. As regras criadas representam simbolicamente o conhecimento do especialista no domínio do problema, e serão usadas para avaliar o aprendizado qualitativo das RNA's.

Com o modelo, pode se representar e avaliar o comportamento qualitativo das RNA's. Cada regra simbólica define qual a sensibilidade da variável de saída da rede em relação a cada uma das variáveis de entrada (ZÁRATE, 1998).

Tabela 5 – Regras Simbólicas do Comportamento da Carga (P) com o aumento do valor de cada variável de entrada da RNA

Entrada	Saída
$\uparrow h_i$	$\uparrow P$
$\uparrow h_o$	$\downarrow P$
$\uparrow \mu$	$\uparrow P$
$\uparrow t_f$	$\downarrow P$
$\uparrow t_b$	$\downarrow P$
$\uparrow y$	$\uparrow P$

Tabela 6 - Regras Simbólicas do Comportamento da Carga (P) com a queda do valor de cada variável de entrada da RNA

Entrada	Saída
$\downarrow h_i$	$\downarrow P$
$\downarrow h_o$	$\uparrow P$
$\downarrow \mu$	$\downarrow P$
$\downarrow t_f$	$\uparrow P$
$\downarrow t_b$	$\uparrow P$
$\downarrow y$	$\downarrow P$

Tabela 7 – Regras Simbólicas da Classificação dos fatores de sensibilidade por ordem de grandeza numérica absoluta com sinal indicador de comportamento

1°	2°	3°	4°	5°	6°
$+\frac{\partial P}{\partial \mu}$	$-\frac{\partial P}{\partial h_o}$	$+\frac{\partial P}{\partial h_i}$	$+\frac{\partial P}{\partial y}$	$-\frac{\partial P}{\partial t_b}$	$-\frac{\partial P}{\partial t_f}$

Na Tabela 5 são apresentadas às regras simbólicas para avaliação do comportamento da P(carga) quando há aumento em qualquer uma das variáveis de entrada. Na Tabela 6, é apresentado o contrário, o comportamento da P(carga) quando há queda em qualquer uma das variáveis de entrada. Observa-se na Tabela 5 que a P(carga) aumenta, caso aumente a espessura de entrada (h_i) ou o atrito(μ) ou o escoamento da tira (y). E diminui, caso aumente a espessura de saída (h_o) ou a tensão a ré (t_b) ou a tensão a frente (t_f). O contrário da Tabela 5 é observado na Tabela 6.

Na Tabela 7, os fatores de sensibilidade são classificados em ordem de grandeza numérica absoluta, ou seja, a P(carga) é mais sensível ao atrito (μ), em segundo a espessura saída (h_o), e sucessivamente a espessura de entrada (h_i), escoamento da tira (y), tensão a ré (t_b), e por último tensão a frente (t_f). Um alto nível de sensibilidade significa que basta uma pequena variação em determinada variável de entrada para grande variação na variável de saída. E consequentemente, um baixo nível de sensibilidade significa que existindo uma variação em determinada variável de entrada pouco refletirá na variação da variável de saída.

Saber que a carga (P) é mais sensível ao atrito (μ) do que a espessura saída (h_o) significa que o valor absoluto do fator de sensibilidade do atrito (μ) é maior que o da espessura de saída (h_o). O sinal de cada fator de sensibilidade representa o comportamento da carga (P) ao aumentar o valor de sua respectiva variável de entrada. Ou seja, aumentando o atrito (μ) conforme o sinal "+" do fator ($+\frac{\partial P}{\partial \mu}$) é necessário aumentar a carga (P).

Criado o modelo qualitativo com todas as regras simbólicas representado o comportamento do sistema físico, os fatores de sensibilidade poderão ser calculados e usados para avaliar as RNA's.

O resultado das avaliações das RNA's pelos fatores de sensibilidade usando as regras simbólicas pode ser visto pela Figura 15 e Figura 16, as quais representam graficamente os mapas de sensibilidade do Anexo 03. No Anexo 03 os mapas possuem uma maior riqueza de detalhes, os quais serão explicados a seguir. Devido os mapas serem muito grandes foi gerado uma imagem horizontal de cada um dos mapas para que eles possam ser vistos em um todo.

Diferente da avaliação feita na Seção 3.1 que foi ao final do treinamento das RNA's, agora as redes são avaliadas durante o processo de treinamento. A cada época de treinamento, todas as redes são avaliadas pelas regras simbólicas verificando o quanto cada uma aprendeu qualitativamente o comportamento do processo.

Foram realizadas quinze avaliações em cada uma das 127 RNA's. Cada avaliação foi feita com um número de épocas de treinamento diferente, partindo de 01 até 15 épocas. Uma época representa a apresentação dos conjuntos de treinamento (entrada e saída) para a rede e o cálculo dos novos pesos e bias. Durante o treinamento os conjuntos podem ser apresentados um de cada vez (incremental) ou todos juntos (batch). Como foi usado neste trabalho o algoritmo de treinamento Levenberg-Marquardt, em cada época, todos os conjuntos de treinamento são apresentados e ao final, os pesos e bias são ajustados.

A primeira avaliação com apenas uma época de treinamento não foi considerada, pois não houve ajuste significativo nos pesos, todos os pesos ainda possuíam valores bem próximos de zero. Isso ocorreu devido ao fato de todos os pesos iniciarem com zero.

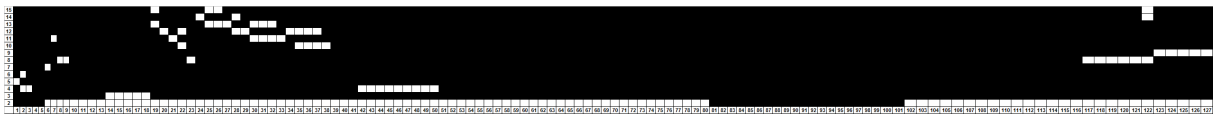

Figura 15 - Mapa de Sensibilidade: RNA's Avaliadas pelo Sinal dos Fatores de Sensibilidade

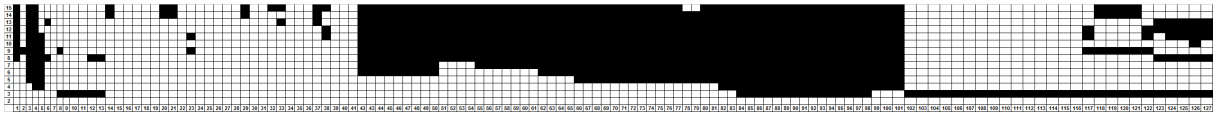

Figura 16 - Mapa de Sensibilidade: RNA's Avaliadas pelo Sinal e pela Ordem de Grandeza Numérica Absoluta dos Fatores de Sensibilidade

Tanto na Figura 15 quanto na Figura 16, o eixo horizontal "*x*" referencia a quantidade de neurônios na camada escondida da rede, iniciando na parte inferior esquerda (01 neurônio) até a parte inferior direita (127 neurônios). O eixo vertical "*y*" referencia a quantidade de épocas de treinamento, iniciando na parte inferior esquerda (02 épocas) até a parte superior esquerda (15 épocas). Os pontos "escuros" das figuras representam que para aquela quantidade "*x*" de neurônios na camada escondida e determinada quantidade "*y*" de épocas de treinamento foram acertadas todas as regras simbólicas pela RNA. Nos casos onde ocorreram erros, os pontos estão em "branco". Dessa forma, as regiões escuras representam os locais em que o comportamento qualitativo foi aprendido completamente pelas redes ao longo das épocas.

Na Figura 15 as redes são avaliadas apenas pelas regras da Tabela 5 e Tabela 6. Nestas regras avalia-se o sinal dos fatores de sensibilidade verificando se a RNA aprendeu o comportamento da carga (P) ao aumentar ou diminuir o valor de alguma variável de entrada. Observa-se que duas épocas de treinamento são suficientes para que as redes com 01 (um) até 05 (cinco) neurônios na camada escondida aprendam as regras que representam o comportamento do processo, como também as redes com 81 (oitenta e um) até 101(cento e um) neurônios na camada escondida. A partir de três épocas de treinamento, a grande maioria das redes aprendeu o comportamento. Em alguns pontos do mapa de sensibilidade da Figura 15, pode-se observar que algumas redes com determinada quantidade de épocas de treinamento aprenderam todas as regras de comportamento do processo, e estas mesmas redes quando avaliadas com uma quantidade maior de épocas de treinamento apresentam erros em algumas das regras. Isto se dá devido ao ajuste de pesos a cada época de treinamento, criando a situação em que alguns conjuntos de treinamento são aprendidos melhor pelo ajuste feito na época atual, mas para isto acabou comprometendo o ajuste feito em épocas anteriores para outros conjuntos. Este comportamento é normal, e a medida que o treinamento continua todos os conjuntos são aprendidos.

Nota-se no mapa de sensibilidade apresentado pela Figura 15 que praticamente todas as redes conseguem aprender o comportamento do processo com poucas épocas de treinamento. Embora esta avaliação seja significativa, outras regras devem ser consideradas visto que se possui mais conhecimento do processo para avaliar o aprendizado das redes.

Na Figura 16 as redes são avaliadas pelas regras da Tabela 5, Tabela 6 e Tabela 7. Nestas regras, avalia-se o sinal dos fatores de sensibilidade verificando se a RNA aprendeu o comportamento da carga (P) ao aumentar ou diminuir o valor de alguma variável de entrada, como também a ordem de grandeza numérica dos fatores de sensibilidade, examinando o grau de sensibilidade da carga (P) por cada uma das variáveis de entrada. Observa-se que diferentemente do que foi visto no mapa de sensibilidade da Figura 15, onde um conjunto de redes aprendeu as regras com duas épocas de treinamento, no mapa de sensibilidade da Figura 16, as redes começam a aprender o comportamento a partir de três épocas de treinamento. Isto, devido a um número maior de regras serem usadas para avaliar as redes, ou seja, mais conhecimento do processo foi incorporado na avaliação.

Várias regiões de representação do comportamento do processo são formadas a medida que se aumenta o número de épocas de treinamento. Podendo destacar a região formada pelas redes com 03 a 05 neurônios na camada escondida a partir de quatro épocas de treinamento, assim como a região formada pelas redes com 42 a 101 neurônios na camada escondida a

partir de três épocas de treinamento. Além de outras regiões isoladas com determinadas quantidades de neurônios na camada escondida em determinada época ou em poucas épocas sucessivas.

O objetivo destas avaliações foi mapear todo o universo de comportamento das 127 redes com diferentes quantidades de neurônios na camada escondida, treinadas de 01 até 15 épocas. Desta forma, a medida que o treinamento evolui, pode-se observar que as redes começam a aprender o comportamento do processo, formando regiões cujas quantidades de neurônios na camada escondida são ideais para aprendizado do comportamento qualitativo do processo.

Estes experimentos mostraram um "raio-x" do aprendizado qualitativo das RNA's dentro de um intervalo de 127 neurônios na camada escondida e 15 épocas de treinamento para cada RNA. Viu-se que, por exemplo, com cinco épocas de treinamento tanto uma rede com 03 neurônios na camada escondida quanto uma rede com 101, aprenderam da mesma forma o comportamento qualitativo do processo. Diante deste resultado, pode-se optar pela RNA com 03 neurônios na camada escondida para controle do processo, sem a necessidade de avaliar a estrutura pelo erro tolerável de operação, pré-estabelecido, para resultados gerados pela RNA. Isso porque, sabe-se que ela aprenderá o comportamento qualitativo do processo.

4 Estrutura Evolucionária Proposta

Diante dos resultados encontrados nos experimentos do capítulo anterior, enxerga-se a necessidade de um mecanismo de busca automático que ajude a encontrar a quantidade ideal mínima de neurônios na camada escondida de uma RNA. Pois treinar diversas redes até atingirem ao erro final pré-estabelecido para avaliar qual a quantidade ideal mínima de neurônios na camada escondida é completamente sem sentido, treinada uma rede completamente, por que não usá-la? Por outro lado, avaliar o comportamento qualitativo de diversas redes por meio de regras simbólicas através de fatores de sensibilidade durante o treinamento (época por época), demanda muito tempo e grande esforço computacional, o que torna esta prática dispendiosa.

Neste capítulo, é apresentada uma nova estrutura evolucionária, onde a avaliação do comportamento qualitativo da RNA é incorporada ao processo evolutivo dos algoritmos genéticos para que se possa encontrar a quantidade ideal mínima de neurônios na camada escondida de uma RNA Perceptron Multicamadas.

4.1 Descrição da Estrutura

A estrutura proposta neste trabalho é apresentada pela Figura 17. Nesta estrutura, o processo de busca pela quantidade ideal mínima de neurônios na camada escondida de uma RNA é feito por um algoritmo genético que avalia o conjunto de soluções candidatas (população) pelo quanto cada rede aprendeu do comportamento qualitativo do processo.

Nesta estrutura, é adotada a representação binária para codificar cada indivíduo da população, visto que esta modelagem pode representar facilmente qualquer quantidade inteira de neurônios que serão usados na camada escondida da RNA.

Cada etapa da estrutura proposta neste trabalho é descrita nas seções seguintes.

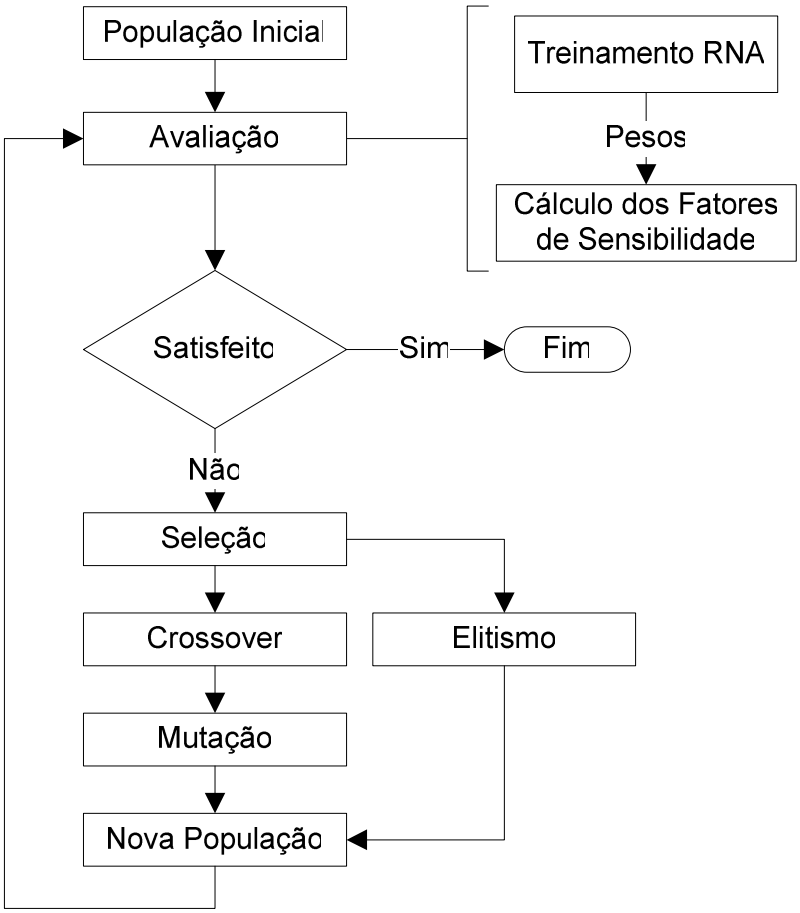

Figura 17 - Estrutura Evolucionária Proposta

4.1.1 População Inicial

A definição da população inicial é uma fase muito importante no uso do AG. O tamanho da população e a forma como esta é gerada são aspectos que devem ser observados. Quando se define uma população inicial pequena e aleatória, têm-se algumas regiões do espaço de busca que não serão representadas (GOLDBERG, 1989). Para minimizar este problema, a população inicial usada nesta estrutura deve ser composta por valores aleatórios distribuídos uniformemente dentro de todo o espaço de busca. Além disso, cada indivíduo da população deve possuir a mesma quantidade arbitrária de bits (0 e 1), pois quando convertidos para a base decimal representam o número de neurônios da camada escondida de uma RNA Perceptron Multicamadas.

4.1.2 Avaliação

Cada indivíduo da população é avaliado e recebe uma nota (fitness). A avaliação de cada indivíduo acontece após o treinamento de uma RNA com a quantidade de neurônios na camada escondida que o indivíduo representa. O treinamento é realizado até uma quantidade pré-determinada de épocas. Após o treinamento os fatores de sensibilidade são calculados sobre um ponto de operação e usados para avaliar o quanto as redes aprenderam do comportamento qualitativo do processo usando as regras simbólicas que representam o conhecimento do especialista. No processo de avaliação as notas das redes são referentes a quantidade de erros sobre as regras simbólicas.

Deve-se definir um critério de parada para o processo evolutivo. Nesta estrutura, sugere-se que seja usado como critério de parada a convergência, visto que não se sabe a quantidade ideal. Logo, se após sucessivas gerações o valor de aptidão médio da população ou a aptidão dos melhores indivíduos não sofrer alteração ou variar dentro de um intervalo de valores muito próximos, por várias gerações consecutivas, o processo de busca pode ser interrompido.

4.1.2.1 Treinamento da RNA

Primeiro, todas as RNA's devem ser treinadas até atingir um número de épocas pré-estipulado. O número de épocas de treinamento deve ser pequeno, pois as redes devem ser avaliadas em seu desempenho inicial, não sendo necessário o treinamento por completo da rede para que esta seja avaliada, até mesmo porque, se a rede é treinada completamente não há razão em continuar o processo evolutivo e não usá-la de imediato. No anexo 04 é sugerido um mecanismo para que a quantidade de épocas de treinamento seja estimada. Segundo, deve existir uma padronização dos pesos inicias das redes, ou seja, os pesos devem iniciar com valor zero. Isto é necessário para evitar que o processo de seleção seja tendencioso, optando por uma estrutura que iniciou com os pesos bem próximos dos ideais. Terceiro, é necessário usar um algoritmo de treinamento que possua uma capacidade de convergência alta (rápida), para que as mudanças de geração ocorram mais rapidamente. Isso garante que, com poucas épocas, os pesos sofrerão grandes ajustes e as RNA's podem ser avaliadas qualitativamente. O algoritmo proposto para ser usado é uma variação do Backpropagation tradicional, Levenberg-Marquardt.

4.1.2.2 Cálculo dos Fatores de Sensibilidade

Após o treinamento das redes, são calculados os fatores de sensibilidade sobre um ponto de operação que não fez parte do treinamento. Os fatores de sensibilidade são obtidos através da diferenciação dos pesos da RNA sendo treinada. Com base nos fatores de sensibilidade, pode-se avaliar o quanto cada rede aprendeu o comportamento qualitativo do processo, e atribuir sua nota (fitness). Neste ponto, é incorporada a representação simbólica do conhecimento do especialista no domínio do problema, que será usado para avaliar o aprendizado qualitativo das redes. Será o especialista no domínio do problema quem informará o quanto cada variável de saída da rede é sensível a cada uma das variáveis de entrada. Dessa forma, todos os fatores de sensibilidade serão avaliados individualmente e, em seguida, em conjunto para que a RNA receba sua nota. Individualmente, pois devem representar se uma variável de saída irá aumentar ou diminuir quando uma variável de entrada aumenta ou diminui. Em conjunto, quando são distribuídos em ordem de grandeza numérica absoluta partindo da variável de entrada que mais afeta para a que menos afeta. Enfim, as redes que melhor aprenderem estas regras são consideradas as melhores e, consequentemente, possuem maiores chances de serem selecionadas para gerarem descendentes, os quais formarão a próxima geração do processo evolutivo.

4.1.3 Seleção e Elitismo

Na fase de seleção, os indivíduos já foram avaliados e podem ser classificados em um ranking. As melhores estruturas ficarão no topo do ranking, seguidas das demais conforme seu fitness. Em situações de empate, mesmo valor de fitness, a melhor posição do ranking fica com o indivíduo que possui a menor quantidade de neurônios. A partir de então, os mais aptos são selecionados para a próxima geração e os menos aptos serão descartados (Darwinismo). Tratando-se de um processo de seleção do AG, pode-se usar qualquer critério de seleção dos AG's. Além disso, visando melhorar a convergência do algoritmo genético, propõe-se uma estratégia denominada "Elitismo", onde o melhor indivíduo de uma geração é transferido, sem alterações, para a próxima geração. Isto é feito para que não se perca a melhor solução encontrada até o momento.

4.1.4 Operadores Genéticos e Nova População

Selecionados os indivíduos mais aptos, são aplicados os operadores genéticos sobre eles (GOLDBERG, 1989). Segundo Lacerda e Carvalho (1999) os operadores de crossover e mutação são os principais mecanismos dos AG's para explorar regiões desconhecidas em um espaço de busca em que se deseja encontrar uma solução. Caso o crossover não seja aplicado, os cromossomos filhos são exatamente iguais aqueles selecionados como cromossomos pais. Em Bayer e Lui Wang (1991) sugere-se o crossover de 1 ou 2 pontos com probabilidade entre 50% a 100% de ocorrer, até mesmo porque o crossover de 1 ponto é um caso particular do crossover de 2 pontos. Além disso, possuir vários pontos de corte no crossover não representará em resultados significativos, pois destruirá com facilidade os blocos de construção. Realizado o crossover, aplica-se a mutação, operação em que ocorre a inversão dos valores dos bits, mudando o valor de um bit cujo valor é 1 para 0, ou vice-versa. A mutação assegura a diversidade da população, mas faz isto destruindo a informação contida no cromossomo. Logo, a taxa de mutação deve ser o suficiente para garantir a diversidade da população, em Lacerda e Carvalho (1999) é sugerido taxa entre 0,1% e 5%.

Por fim, a nova população é composta com os descendentes dos pais selecionados para reprodução, juntamente com o indivíduo da geração anterior selecionado pelo critério de elitismo.

5 Aplicação da Estrutura Proposta ao Processo de Laminação a Frio

5.1 Estrutura da RNA aplicada a Laminação a Frio

A RNA considerada neste trabalho é uma rede multicamadas de uma camada escondida. A estrutura da Rede Neural Artificial que foi aplicada nesta pesquisa é a apresentada pela Eq. (25):

$$(h_i, h_o, \mu, t_b, t_f, \bar{y}) \xrightarrow{\text{Rede Neural}} (P) \tag{25}$$

Sendo as variáveis de entrada da rede neural: h_i (espessura de entrada), h_o (espessura de saída), μ (atrito), t_b (tensão a ré), t_f (tensão a frente), y (escoamento da tira); e a variável de saída: P(carga por unidade de largura) (ZÁRATE; BITTENCOUT, 2007), (BITTENCOUT; ZARATE, 2002).

No Anexo 01 é apresentado o processo de laminação a frio e toda a modelagem desenvolvida para uso de uma RNA Perceptron Multicamadas para controle do processo.

5.2 Simulações e Resultados

A estrutura evolucionária proposta neste trabalho foi usada para encontrar a quantidade ideal mínima de neurônios na camada escondida da RNA Perceptron Multicamadas aplicada no processo de laminação a frio. Para isto, algumas definições iniciais foram necessários e são apresentados a seguir.

Cada indivíduo da população foi composto por 07 bits (0 e 1) que convertidos para a base decimal representam o número de neurônios da camada escondida de uma RNA Perceptron Multicamadas dentro do intervalo numérico de 01 à 127. A população inicial foi composta por 12 (doze) valores aleatórios distribuídos uniformemente dentro do espaço de busca, foi definida esta quantidade para possuir representantes na população dentro de todo o espaço de busca. As redes foram treinadas até 5 épocas conforme calculado pela estratégia apresentada

no Anexo 04. O conhecimento do especialista foi representado por regras simbólicas, veja Tabela 5, Tabela 6 e Tabela 7. Em seguida, as regras simbólicas foram incorporadas à estrutura evolucionária proposta para uso na avaliação das RNA's candidatas. Para compor uma nova população, os indivíduos são selecionados usando o método de seleção ranking linear, preservando os dois melhores indivíduos (elitismo). Na fase de reprodução são aplicados o crossover de um ponto e mutação, respeitando as probabilidades de 100% e 0.5% respectivamente; como critério de parada foi usado a estabilidade dos índices de aptidão.

Neste trabalho foram usados três critérios de avaliação de RNA's na estrutura evolucionária proposta. Dois tradicionalmente utilizados (BALAKRISHNAN; HONAVAR, 1995), (YAO, 1995), (FILHO; CARVALHO, 1997), onde as RNA's são avaliadas e selecionadas pelo: (1) erro apresentado ao final do treinamento; (2) erro apresentado ao final da simulação usando o conjunto de validação com dados que não fizeram parte do treinamento. E o terceiro critério, apresentado neste trabalho: (3) avaliação do quanto a rede aprendeu do comportamento qualitativo do processo.

Tratando-se de uma heurística, as simulações foram realizadas cinco vezes para cada critério de avaliação. Dessa forma não foi analisado um único resultado e sim um conjunto de resultados por cada critério usado.

Os resultados das simulações para cada um dos três critérios, respectivamente, são apresentados na Tabela 8, Tabela 9 e Tabela 10.

Na Tabela 8, Tabela 9 e Tabela 10 o número de épocas de cada simulação foi calculada multiplicando o número de épocas do processo de treinamento (cinco) pela quantidade de indivíduos da população (doze), e o resultado multiplicado pelo número de gerações necessárias para encontrar a quantidade ideal de neurônios na camada escondida.

Tabela 8 - Resultados da Simulação usando o erro do treinamento com critério de avaliação das RNA's

Simulação	Neurônios	Gerações	Épocas	Tempo[1](min.)
1	56	9	540	7,0
2	52	8	480	5,9
3	52	18	1080	11,6
4	51	10	600	8,0
5	52	9	540	6,6

[1] Todas as simulações foram realizadas no MATLAB 7.0 em um PC Pentium 4 com 512 MB de RAM.

Tabela 9 - Resultados da Simulação usando o erro da simulação (teste) como critério de avaliação das RNA's

Simulação	Neurônios	Gerações	Épocas	Tempo(min.)
1	67	14	840	10,4
2	69	10	600	7,0
3	67	10	600	7,0
4	67	14	840	11,1
5	69	10	600	6,9

Tabela 10 - Resultados da Simulação usando os fatores de sensibilidade como critério de avaliação das RNA's

Simulação	Neurônios	Gerações	Épocas	Tempo(min.)
1	3	5	300	3,1
2	4	7	420	4,0
3	3	9	540	6,3
4	3	7	420	4,8
5	4	6	360	3,6

Pode-se observar, comparando a Tabela 8, Tabela 9 e Tabela 10 que o critério de avaliação utilizando a análise dos fatores de sensibilidade apresentou bons resultados em comparação aos resultados dos critérios tradicionais. Enquanto os outros critérios apresentam como quantidade ideal de neurônios na camada escondida valores elevados, o critério apresentado neste trabalho apresenta uma quantidade de neurônios ideal mínima na camada escondida, veja a Figura 18. Pode se dizer também que não houve perda na qualidade de aprendizagem, pois as redes foram avaliadas qualitativamente, ou seja, em quanto aprenderam qualitativamente o comportamento do processo.

Além de apresentar resultados relevantes na definição da quantidade ideal mínima de neurônios na camada escondida, também encontrou, na maioria das vezes, este resultado gastando poucas gerações, o que implica em um menor número de épocas de treinamentos necessários para avaliação das redes em cada geração, conforme a Figura 19 e Figura 20.

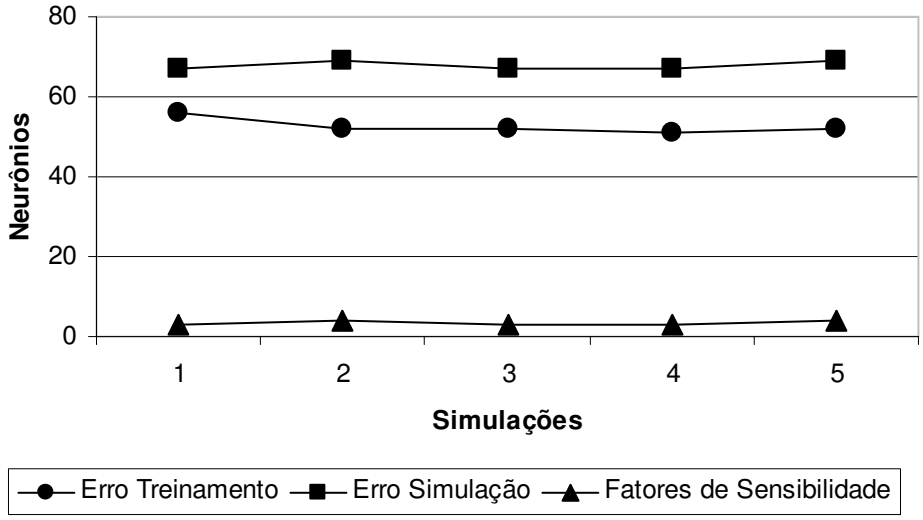

Figura 18 - Quantidade de neurônios encontrada por cada um dos três critérios de avaliação, em cada simulação.

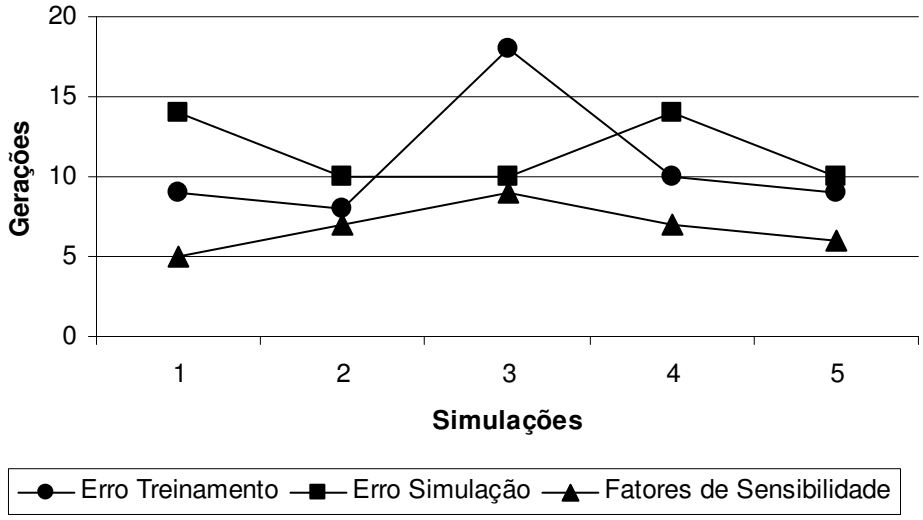

Figura 19 - Quantidade de gerações necessárias para encontrar a solução, realizada por cada um dos três critérios de avaliação, em cada simulação.

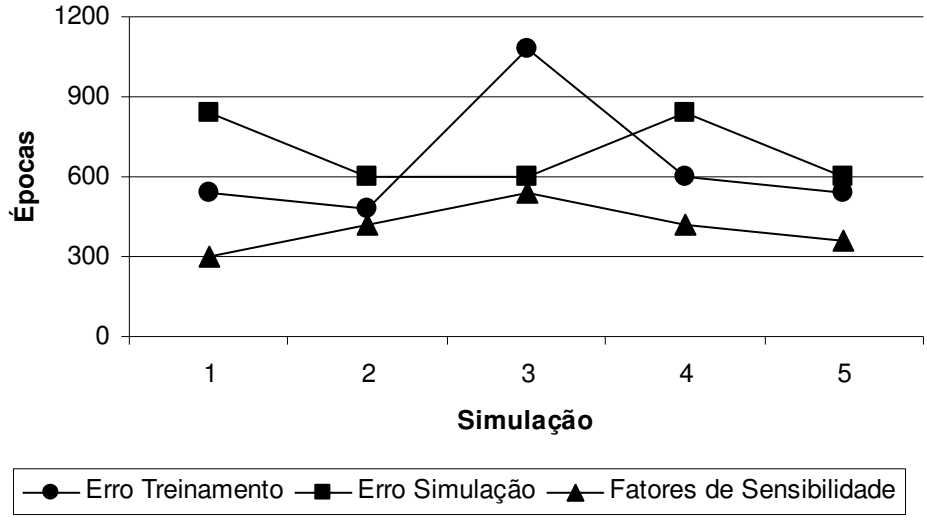

Figura 20 - Quantidade de épocas para encontrar a solução, gastos por cada um dos três critérios de avaliação, em cada simulação.

Para verificação, RNA's foram treinadas com os valores encontrados para a quantidade ideal de neurônios na camada escondida pelos três critérios. Todas deveriam atingir um erro global de 10^{-4}. O treinamento foi finalizado por todas as redes atingindo o erro estabelecido, cada uma consumindo uma quantidade de épocas para treinar, veja Tabela 11, Tabela 12 e Tabela 13. Pode-se observar que a solução encontrada com a estrutura de evolucionária apresentada neste trabalho, que incorporou o conhecimento do processo pelo especialista para análise dos fatores de sensibilidade como critério de avaliação precisou de menos épocas para treinar. Além de apresentar uma boa capacidade de generalização quando testada. Isso prova que a solução encontrada pela estrutura proposta é válida.

Para uma RNA, mais importante que aprender completamente os conjuntos que lhe são apresentados no processo de treinamento, é possuir uma boa capacidade de generalização. Isso, pois quando colocada em operação deverá responder satisfatoriamente a valores que não fizeram parte do treinamento.

Tabela 11 - Resultados dos treinamentos das RNA's com a quantidade de neurônios encontrada usando o erro final do treinamento como critério de avaliação

Neurônios	Épocas	Erro Médio Treinamento (Kgf/mm)	Erro Médio Validação (Kgf/mm)
51	93	15.58	12.21
52	91	15.37	12.02
56	93	15.10	11.90

Tabela 12 - Resultados dos treinamentos das RNA's com a quantidade de neurônios encontrada usando o erro final do teste como critério de avaliação

Neurônios	Épocas	Erro Médio Treinamento (Kgf/mm)	Erro Médio Validação (Kgf/mm)
67	101	14.75	11.77
69	101	14.70	11.70

Tabela 13 - Resultados dos treinamentos das RNA's com a quantidade de neurônios encontrada usando a análise dos fatores de sensibilidade como critério de avaliação

Neurônios	Épocas	Erro Médio Treinamento (Kgf/mm)	Erro Médio Validação (Kgf/mm)
3	41	21.42	15.01
4	44	22.80	14.59

Para que a capacidade de generalização das RNA's fossem avaliadas, todas elas foram testadas com um conjunto de 50 valores que não fizeram parte de seu treinamento. Os valores médios de erro podem ser observados na Tabela 11, Tabela 12 e Tabela 13. Graficamente, pode-se notar que o valor real (esperado) foi atendido com o valor de saída gerado pela rede.

Por fim, observa-se que todos os valores encontrados pelos critérios usados na estrutura evolucionária atendem e apresentam resultados muito próximos. O que difere a proposta deste trabalho das demais é que, buscou-se a quantidade ideal mínima de neurônios na camada escondida para representar o processo em questão, atendendo quantitativa e qualitativamente, e esta foi encontrada.

A Figura 21, Figura 22 e Figura 23 mostram graficamente as simulações com as RNA's com 03, 51 e 67 neurônios na camada escondida diante de 50 conjuntos que não fizeram parte do treinamento.

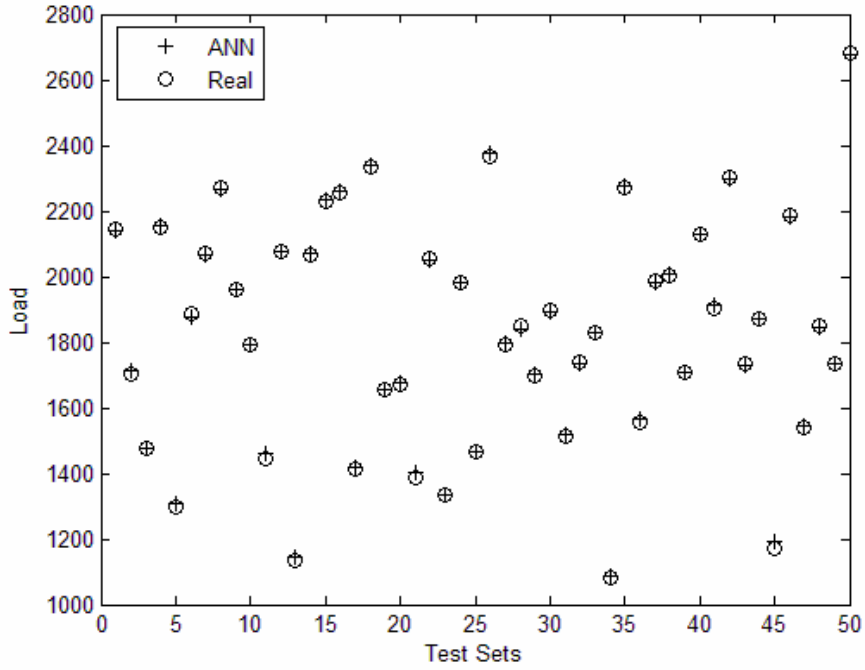

Figura 21 - Saída da RNA (ANN), treinada com 03 neurônios na camada escondida, para 50 conjuntos de teste (validação)

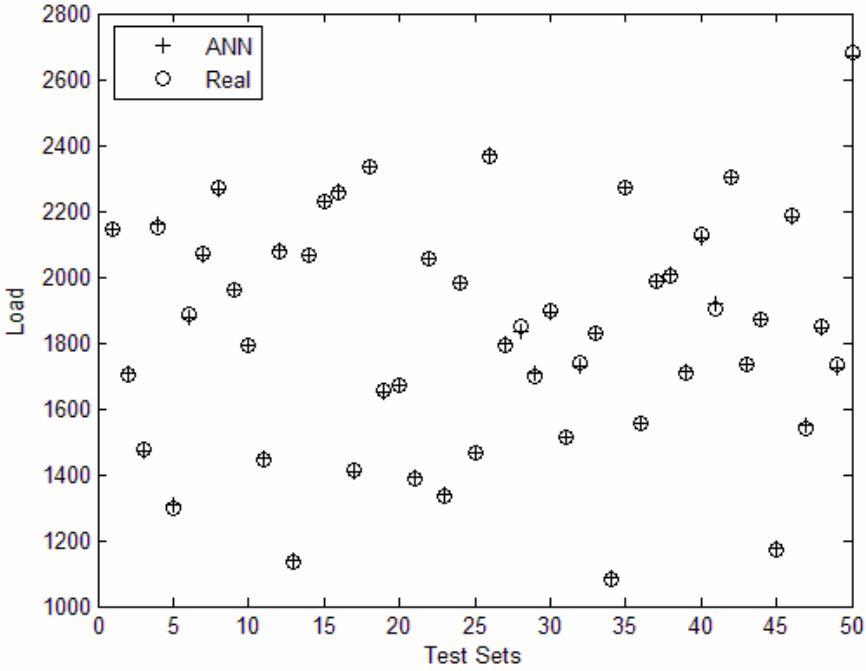

Figura 22 - Saída da RNA (ANN), treinada com 51 neurônios na camada escondida, para 50 conjuntos de teste (validação)

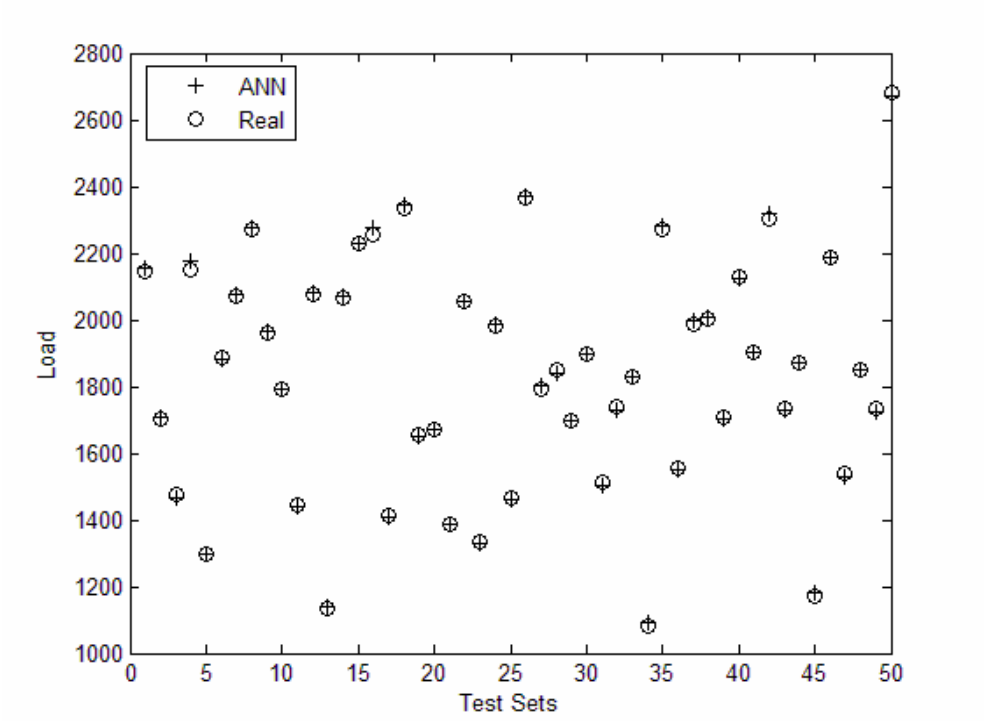

Figura 23 - Saída da RNA (ANN), treinada com 67 neurônios na camada escondida, para 50 conjuntos de teste (validação)

6 Conclusões

Neste trabalho, foi evidenciada a dificuldade em se determinar a quantidade ideal de neurônios na camada escondida de uma RNA perceptron multicamadas, tanto que a forma mais usada é a tentativa e erro. Outras maneiras seriam usar: o conhecimento prévio do projetista, uma fórmula empírica ou algum mecanismo heurístico.

Pode-se concluir que foi possível encontrar a quantidade ideal de neurônios na camada escondida de uma RNA perceptron multicamadas aplicada ao processo de laminação a frio, usando o mecanismo evolutivo proposto neste trabalho, onde as redes neurais foram avaliadas pelo quanto elas aprenderam do comportamento do processo.

Concluiu-se que o uso da análise de sensibilidade na avaliação do comportamento qualitativo das redes neurais dentro da estrutura evolucionária proposta, apresenta bons resultados caso possua um conhecimento prévio do comportamento do processo por parte do especialista.

Logo, a hipótese levantada neste trabalho foi confirmada, sendo possível encontrar mais rapidamente (como pode ser comparado o tempo das Tabelas 02 e 03 com o da Tabela 10), do que o procedimento tradicional de tentativa e erro, a quantidade ideal mínima de neurônios na camada escondida de uma RNA Perceptron Multicamadas, usando o conhecimento do especialista expresso através dos fatores de sensibilidade, como um critério de avaliação das RNA's dentro do mecanismo evolutivo dos AG's.

Isso pode ser visto, comparando os resultados das simulações sem uso da estrutura evolutiva proposta com os resultados das simulações usando a estrutura evolutiva proposta. Nas primeiras simulações (sem usar a estrutura proposta), foram realizados treinamentos avaliando as redes pelo erro. Considerando todas as 127 redes, foram necessários entre 54 e 58 minutos (veja Tabela 2 e Tabela 3) para que todas as 127 redes fossem treinadas exaustivamente, até um erro pré-estabelecido, e avaliadas, para que em seguida fosse possível indicar a melhor quantidade de neurônios na camada escondida. Em outras simulações, ainda sem usar a estrutura proposta, foram realizados treinamentos das 127 redes e, a cada época, estas eram avaliadas pela análise de sensibilidade, gastando no total aproximadamente 3,5 horas para que fosse possível indicar a melhor quantidade de neurônios na camada escondida (veja Anexo 03). Por outro lado, nas simulações usando a estrutura proposta, o tempo gasto em média é de 05 minutos (veja Tabela 10).

Foi encontrada uma RNA otimizada com a quantidade ideal de neurônios necessários na camada escondida para seu treinamento e aplicação no controle do processo de laminação a frio. A união de Algoritmos Genéticos, uma poderosa ferramenta de busca e otimização, com a Análise de Sensibilidade, mostrou-se interessante para solução de problemas onde existe a possibilidade de avaliar o comportamento qualitativo de uma solução candidata.

A principal contribuição deste trabalho foi propor uma nova estrutura evolucionária que consegue ajudar, de forma rápida, na definição da quantidade ideal mínima de neurônios na camada escondida de uma RNA perceptron multicamadas, sem que a rede seja treinada várias e exaustivas vezes, propiciando uma estrutura otimizada, impactando na redução do esforço computacional durante o processo de treinamento. Além disso, usando um critério coerente com o comportamento do processo, o qual é usado para avaliar qualitativamente o aprendizado da RNA, garantindo a generalização.

Como trabalho futuro, pretende-se aplicar a estrutura evolutiva proposta em diversos processos em que se possui o conhecimento do comportamento qualitativo por parte do especialista no domínio do problema, para que as regras simbólicas sejam criadas, permitindo projetar uma RNA com a quantidade ideal de neurônios na camada escondida.

Bibliografia

AVILA, S.L.; LISBOA, A.C.; KRAHENBUHL, L.; CARPES, W.P., JR.; VASCONCELOS, J.A.; SALDANHA, R.R.; TAKAHASHI, R.H.C. Sensitivity analysis applied to decision making in multiobjective evolutionary optimization. *Magnetics, IEEE Transactions on* , vol.42, no.4, pp. 1103-1106, April 2006

BACK, T. Selective pressure in evolutionary algorithms: a characterization of selection mechanisms. *Evolutionary Computation, 1994. IEEE World Congress on Computational Intelligence., Proceedings of the First IEEE Conference on* , vol., no., pp.57-62 vol.1, 27-29 Jun 1994

BACK, T.; HAMMEL, U.; SCHWEFEL, H.-P. Evolutionary computation: comments on the history and current state. *Evolutionary Computation, IEEE Transactions on* , vol.1, no.1, pp.3-17, Apr 1997

BALAKRISHNAN, K.; HONAVAR, V. Evolutionary design of neural architectures - a preliminary taxonomy and guide to literature. Technical report, A.I. Research Group, Iowa State University, 1995.

BAYER, S.E.; LUI WANG. A genetic algorithm programming environment: Splicer. *Tools for Artificial Intelligence, 1991. TAI '91., Third International Conference on* , vol., no., pp.138-144, 10-13 Nov 1991

BITTENCOUT, F. R.; ZARATE, L. E. Controle da Laminação a Frio Baseado em Redes Neurais, com Capacidade de Generalização, e Lógica Nebulosa via Fatores de Sensibilidade. In: XIV Congresso Brasileiro de Automática - CBA. 2002

BRAGA, A. P.; CARVALHO, A. C. P. L. F.; LUDEMIR, T. B. Redes neurais artificiais: teoria e aplicações. Rio de Janeiro: LTC, 2000

BRANKE, J. Evolutionary Algorithms for Neural Networks Design and Training. Technical Report, Institut für Angewandte Informatik und Formale Beschreibungsverfahren, University of Karlsruhe, Germany, 1995.

CANTU-PAZ, E.; KAMATH, C. An empirical comparison of combinations of evolutionary algorithms and neural networks for classification problems. *Systems, Man and Cybernetics, Part B, IEEE Transactions on* , vol.35, no.5, pp. 915-927, Oct. 2005

COSTA, E.; SIMOES, A. Inteligência Artificial Fundamentos e Aplicações. Rio de Janeiro: FCA, 2004.

EBERHART, R.C.; DOBBINS, R.W. Designing neural network explanation facilities using genetic algorithms. *Neural Networks, 1991. 1991 IEEE International Joint Conference on* , vol., no., pp.1758-1763 vol.2, 18-21 Nov 1991

FILHO, E.F.M.; CARVALHO, A. Evolutionary design of MLP neural network architectures. *Neural Networks, 1997. Proceedings., IVth Brazilian Symposium on* , vol., no., pp.58-65, 3-5 Dec 1997

GOLDBERG, D. E. Genetic Algorithms in Search, Optimization, and Machine Learning. Reading: Addison-Wesley Publishing Company, Inc., 1989.

GUO, Z.; UHRIG, R.E. Using genetic algorithms to select inputs for neural networks. *Combinations of Genetic Algorithms and Neural Networks, 1992. COGANN-92. International Workshop on* , vol., no., pp.223-234, 6 Jun 1992

HAGAN, M.T.; MENHAJ, M.B. Training feedforward networks with the Marquardt algorithm. *Neural Networks, IEEE Transactions on* , vol.5, no.6, pp.989-993, Nov 1994

HAYKIN, S. Redes Neurais: Princípios e Prática. 2ª ed. Porto Alegre: Bookman, 2001

HAYWARD, S. Evolutionary artificial neural network optimisation in financial engineering. *Hybrid Intelligent Systems, 2004. HIS '04. Fourth International Conference on* , vol., no., pp. 210-215, 5-8 Dec. 2004

HECHT-NIELSEN, R. Neurocomputing. New York: Addison Wesley Publ. Co. 1990

HECHT-NIELSEN, R. Theory of the backpropagation neural network. *Neural Networks, 1989. IJCNN., International Joint Conference on* , vol., no., pp.593-605 vol.1, 18-22 Jun 1989

HÜSKEN, M.; JIN, Y; SENDHOFF, B. Structure Optimization of Neural Networks for Evolutionary Design Optimization. Proceedings of the Genetic and Evolutionary Computation Conference – Workshop, pp. 13-16, New York, 2002

KOVACS, Z. L. Redes Neurais Artificiais: Fundamentos e Aplicações. 1. ed. São Paulo: Collegium Cognitio, v. 1500. 158 p. 1996

KOZA, J. R. Survey of genetic algorithms and genetic programming. *WESCON/'95. Conference record. 'Microelectronics Communications Technology Producing Quality Products Mobile and Portable Power Emerging Technologies'* , vol., no., pp.589-, 7-9 Nov 1995

KOZA, J.R.; RICE, J.P. Genetic generation of both the weights and architecture for a neural network. *Neural Networks, 1991., IJCNN-91-Seattle International Joint Conference on* , vol.ii, no., pp.397-404 vol.2, 8-14 Jul 1991

LACERDA, E. G. M.; CARVALHO, A. C. P. L. F. Anais do XIX Congresso Nacional da Sociedade Brasileira de Computação. Rio de Janeiro, 1999.

LIPPMANN, R.P. Neutral nets for computing. *Acoustics, Speech, and Signal Processing, 1988. ICASSP-88., 1988 International Conference on* , vol., no., pp.1-6 vol.1, 11-14 Apr 1988

LIU, Y.; YAO, X., "Evolutionary design of artificial neural networks with different nodes," *Evolutionary Computation, 1996., Proceedings of IEEE International Conference on* , vol., no., pp.670-675, 20-22 May 1996

MANIEZZO, V. Genetic evolution of the topology and weight distribution of neural networks. *Neural Networks, IEEE Transactions on* , vol.5, no.1, pp.39-53, Jan 1994

MCCULLOCH, W. S.; PITTS, W. H. A logical calculus of the ideas immanent in nervous activity. Bulletin of Mathematical Biophysics, n. 5, p.115-133, 1943

NGIA, L.S.H.; SJOBERG, J. Efficient training of neural nets for nonlinear adaptive filtering using a recursive Levenberg-Marquardt algorithm. *Signal Processing, IEEE Transactions on [see also Acoustics, Speech, and Signal Processing, IEEE Transactions on]* , vol.48, no.7, pp.1915-1927, Jul 2000

OZDEMIR, M.; EMBRECHTS, M.J.; ARCINIEGAS, F.; BRENEMAN, C.M.; LOCKWOOD, L.; BENNETT, K.P. Feature selection for in-silico drug design using genetic algorithms and neural networks. *Soft Computing in Industrial Applications, 2001. SMCia/01. Proceedings of the 2001 IEEE Mountain Workshop on* , vol., no., pp.53-57, 2001

POGGIO, T.; GIROSI, F. Networks for approximation and learning. *Proceedings of the IEEE*, vol.78, no.9, pp.1481-1497, Sep 1990

POUR, HAMED MOVAHEDI; ATLASBAF, ZAHRA; HAKKAK, MOHAMMAD. Performance of Neural Network Trained with Genetic Algorithm for Direction of Arrival Estimation. *Mobile Computing and Wireless Communication International Conference, 2006. MCWC 2007. Proceedings of the First* , vol., no., pp.197-202, 17-20 Sept. 2006

PRADOS, D.L. New learning algorithm for training multilayered neural networks that uses genetic-algorithm techniques. *Electronics Letters* , vol.28, no.16, pp.1560-1561, 30 Jul 1992

RIBEIRO FILHO, J.L.; TRELEAVEN, P.C.; ALIPPI, C. Genetic-algorithm programming environments. *Computer* , vol.27, no.6, pp.28-43, Jun 1994

ROSENBLATT, F. The perceptron: a probabilistic model for information storage and retrieval in the brain. Psychological Review, v. 65, p. 386-408, 1958

SCHAFFER, J.D.; WHITLEY, D.; ESHELMAN, L.J. Combinations of genetic algorithms and neural networks: a survey of the state of the art. *Combinations of Genetic Algorithms and Neural Networks, 1992. COGANN-92. International Workshop on* , vol., no., pp.1-37, 6 Jun

1992

SRINIVAS, M.; PATNAIK, L.M. Genetic algorithms: a survey. *Computer*, vol.27, no.6, pp.17-26, Jun 1994

THIERENS, D.; GOLDBERG, D. Elitist recombination: an integrated selection recombination GA. *Evolutionary Computation, 1994. IEEE World Congress on Computational Intelligence., Proceedings of the First IEEE Conference on* , vol., no., pp.508-512 vol.1, 27-29 Jun 1994

WHITLEY, D.; KARUNANITHI, N. Generalization in feed forward neural networks. *Neural Networks, 1991., IJCNN-91-Seattle International Joint Conference on* , vol.ii, no., pp.77-82 vol.2, 8-14 Jul 1991

XIAOQIN ZENG; YEUNG, D.S. Sensitivity analysis of multilayer perceptron to input and weight perturbations. *Neural Networks, IEEE Transactions on* , vol.12, no.6, pp.1358-1366, Nov 2001

XUSHENG LU; BOURBAKIS, N. Neural network training using genetic algorithms in ATM traffic control. *Intelligence and Systems, 1998. Proceedings., IEEE International Joint Symposia on* , vol., no., pp.396-401, 21-23 May 1998

YAO, X. A review of evolutionary artificial neural networks. *International Journal of Intelligent Systems*, 8:539--567, 1993b

YAO, X. Evolutionary artificial neural networks. *Encyclopedia of Computer Science and Technology*, ed. A. Kent et al., Vol. 33, pp.137-170, Marcel Dekker Inc., New York, NY 10016, 1995

YAO, X. Evolutionary artificial neural networks. *International Journal of Neural Systems*, 4(3):203—222, 1993a

YAO, X.; LIU, Y. A new evolutionary system for evolving artificial neural networks. *Neural Networks, IEEE Transactions on* , vol.8, no.3, pp.694-713, May 1997

ZÁRATE, L. E. Um Método de Análise para Laminadores Tandem a Frio. Tese de Doutorado, Universidade Federal de Minas Gerais, Belo Horizonte, MG, Brasil, 1998

ZÁRATE, L.E.; BITTENCOUT, F. R. Neural Networks Applied to Adjustment and Combination of the ControlActions for the Cold Rolling Process. In: IEEE IJCNN 2007, 2007, Orlando. Proc. 17th International Joint Conference on Neural Networks (To appear), 2007b.

ZÁRATE, L.E.; BITTENCOUT, F. R. Representation and Control of the Cold Rolling Process through Artificial Neural Networks via Sensitivity Factors. Journal of Materials Processing Technology, v. 1, p. 1-19, 2007a.

Anexo 01: RNA Aplicada ao Processo de Laminação a Frio

Um laminador a frio (Tandem) é formado por "n" cadeiras (normalmente 5) unidas pela tira que passa através delas. A tira é fornecida por uma bobina e quando laminada alimenta uma outra. A espessura da tira é reduzida em passes sequênciais através de uma alta pressão sobre uma pequena àrea de contato de 6,0 a 25,4 mm de comprimento. Qualquer mudança na espessura de entrada, nas tensões à frente, a ré, na tensão de escoamento e/ou no atrito, causarão mudanças na carga e torque de laminação e, correspondentemente, na espessura de saída (ZÁRATE, 1998).

Quando uma cadeira de laminação é perturbada por qualquer variação nos parâmetros, seu efeito será sentido tanto nas cadeiras anteriores e quanto posteriores. Em Zárate (1998) foram obtidas regras de comportamento qualitativas, deduzidas a partir das equações constitutivas do processo. Essas regras descrevem de forma genérica o estado estacionário do processo.

Estrutura da RNA aplicada a Laminação a Frio

A RNA considerada neste trabalho é uma rede multicamadas de uma camada escondida. A função de ativação do neurônio escolhida foi a função sigmóide, Eq. (26):

$$f = \frac{1}{1 + e^{-\sum Input \ \mathrm{x} \ Weights}} \tag{26}$$

Geralmente, o maior esforço para o treinamento de uma rede neural encontra-se na coleta de dados e no pre-processamento que a elas deve ser aplicado. O pre-processamento consiste na normalização dos dados de entrada e de saída. Para a rede em questão, o único requisito é que os valores das entradas e das saídas se encontrem no intervalo de 0 a 1. Como os valores de 0 e 1 são valores infinitos para a função sigmóide, é recomendável diminuir este intervalo para valores de 0.2 e 0.8 respectivamente, com o objetivo de facilitar a convergência durante o treinamento da rede. Os dados foram normalizados e desnormalizados através das seguintes expressões:

$$f^{a}(Lo) = Ln = (Lo - L\min)/(L\max - L\min) \tag{27a}$$

$$f^{b}(Ln) = Lo = Ln * L\max + (1 - Ln) * L\min \quad (27b)$$

$$L\min = (4 * LimiteInf - LimiteSup)/3 \quad (28a)$$

$$L\max = (LimiteInf - 0.8 * L\min)/0.2 \quad (28b)$$

Onde: *Ln* é o valor normalizado; *Lo* o valor a normalizar; *Lmín* e *Lmax* são valores mínimos e máximos dentre os valores das variáveis, *LimiteInf* e *LimiteSup* são os valores mínimos e máximos respectivamente dos conjuntos de dados originais.

A estrutura da Rede Neural Artificial que foi aplicada nesta pesquisa é a apresentada pela Eq. (29):

$$(h_i, h_o, \mu, t_b, t_f, \overline{y}) \xrightarrow{\text{Neural Network}} (P) \quad (29)$$

Sendo as variáveis de entrada da rede neural:

h_i = espessura de entrada,

h_o = espessura de saída,

μ = atrito,

t_b = tensão a ré,

t_f = tensão a frente,

y = escoamento da tira;

e a variável de saída:

P = carga por unidade de largura.

Todos os treinamentos realizados neste trabalho foram feitos com a estrutura da RNA apresentada anteriormente. O conjunto de treinamento foi composto por 215 valores que foram escolhidos usando os critérios propostos por Bittencout e Zárate (2002).

Anexo 02: Tabelas com Erros de Treinamentos e Simulações das RNA's

Tabela 14 - Tabela com 127 RNA's treinadas com diferentes neurônios na camada escondida e simulada com 50 conjuntos desconhecidos após o treinamento.

			Erros (Valor Real)					
Neurônios	**Épocas de Treinamento**	**Tempo Total (seg.)**	**MaxTreina**	**MinTreina**	**MedTreina**	**MaxTeste**	**MinTeste**	**MedTeste**
1	2695	15	137,61	0,71	27,87	66,25	0,67	21,84
2	46	0	101,68	0,04	24,38	48,08	0,22	18,32
3	41	1	95,54	0,01	21,42	46,79	0,07	15,01
4	44	1	102,74	0,35	22,80	54,41	0,91	14,59
5	35	1	114,84	0,02	25,78	52,35	0,97	20,06
6	20	0	105,61	0,20	17,88	38,44	0,02	15,17
7	26	1	107,84	0,06	22,16	51,39	0,28	15,60
8	26	1	102,04	0,09	23,12	50,61	0,86	16,91
9	26	1	101,04	0,28	20,26	41,92	0,43	14,85
10	34	1	109,78	0,11	17,97	38,41	0,29	14,95
11	32	1	108,90	0,01	17,46	37,53	0,18	14,31
12	31	1	107,26	0,00	16,67	35,51	0,05	13,68
13	38	1	95,68	0,12	22,94	52,27	0,84	18,24
14	27	1	110,35	0,30	20,00	55,23	0,33	16,57
15	23	1	133,53	0,63	23,13	61,41	0,24	20,61
16	28	1	118,65	0,05	20,36	60,32	0,28	16,87
17	27	1	103,52	0,03	21,96	57,39	0,78	17,33
18	23	1	129,69	0,19	22,92	59,45	0,76	19,02
19	17	1	129,18	0,06	20,99	60,48	1,58	20,24
20	18	1	108,97	0,33	17,71	43,78	0,20	15,60
21	20	1	132,99	0,12	19,81	62,79	0,24	16,80
22	21	1	127,22	0,13	18,52	55,53	0,42	14,85
23	25	1	111,59	0,27	21,28	51,00	0,67	16,89
24	23	1	117,44	0,29	22,03	60,89	0,05	19,92
25	25	1	128,26	0,00	23,78	54,52	0,58	21,04
26	21	1	128,12	0,20	25,37	60,95	0,39	23,16
27	20	1	135,69	0,09	25,50	58,94	0,63	23,43
28	21	1	129,77	0,22	23,63	54,32	1,56	20,86
29	21	2	129,70	0,09	25,10	58,45	0,13	22,58
30	22	2	132,34	0,00	24,12	54,03	0,81	21,08
31	20	2	131,99	0,27	24,00	53,06	1,06	21,01
32	19	2	131,80	0,23	23,74	52,19	1,75	20,79
33	19	2	130,67	0,41	25,44	55,70	1,11	22,68
34	21	2	133,20	0,40	24,28	55,53	0,64	21,12
35	19	2	133,88	0,12	24,38	55,94	0,58	21,12
36	18	2	134,33	0,31	24,43	55,98	0,68	21,12
37	18	2	134,61	0,18	24,25	54,61	1,09	20,96
38	20	3	131,49	0,09	23,75	52,46	0,76	20,83

39	21	3	134,04	0,19	24,59	57,74	0,36	21,18
40	20	3	134,09	0,40	24,57	57,64	0,43	21,15
41	19	3	134,46	0,20	24,64	58,03	0,36	21,14
42	70	11	108,00	0,02	23,02	45,43	3,06	19,26
43	70	11	107,94	0,36	23,22	45,07	3,73	19,41
44	73	12	114,97	0,33	25,21	48,72	0,19	20,74
45	71	13	110,20	0,02	23,67	46,46	3,57	19,54
46	72	13	111,45	0,15	23,76	47,42	3,09	19,53
47	73	14	108,36	0,17	23,49	44,93	3,19	19,68
48	74	15	108,48	0,44	23,47	44,99	3,16	19,71
49	74	16	108,67	0,02	23,68	44,94	2,63	19,80
50	70	15	113,74	0,14	23,94	49,12	2,35	19,54
51	93	21	104,34	0,23	15,58	29,48	0,19	12,21
52	91	28	103,45	0,05	15,37	29,32	0,36	12,02
53	95	31	103,90	0,30	15,42	29,43	0,29	12,10
54	93	34	111,77	0,05	22,95	55,06	0,66	17,57
55	92	32	103,89	0,15	15,37	29,45	0,54	12,08
56	93	33	102,15	0,09	15,10	29,35	0,80	11,90
57	93	35	110,91	0,26	22,85	54,32	0,52	17,32
58	93	36	102,13	0,07	15,07	29,40	0,81	11,89
59	92	37	111,95	0,23	23,04	54,73	0,52	17,44
60	93	38	101,28	0,06	14,92	29,42	1,67	11,84
61	97	42	112,62	0,16	23,14	54,81	0,39	17,44
62	100	45	101,14	0,04	14,85	29,51	1,57	11,82
63	98	46	101,29	0,20	14,86	29,55	1,75	11,82
64	99	49	114,80	0,01	23,48	55,62	0,10	17,69
65	98	49	100,46	0,07	14,80	29,72	0,78	11,80
66	97	50	99,99	0,30	14,73	29,80	0,14	11,71
67	101	55	100,64	0,00	14,75	29,72	0,65	11,77
68	98	55	100,82	0,02	14,77	29,52	0,74	11,73
69	101	59	100,28	0,04	14,70	29,54	0,02	11,70
70	86	51	97,39	0,20	14,50	30,19	0,92	11,88
71	85	51	113,45	0,16	23,52	54,84	0,05	17,59
72	88	55	97,75	0,32	14,50	30,02	1,01	11,87
73	83	54	99,56	0,03	14,57	29,43	0,14	11,62
74	57	41	121,97	0,07	25,21	56,62	0,98	19,37
75	33	23	130,86	0,56	22,84	51,61	0,03	16,22
76	32	23	111,46	0,11	23,92	53,47	0,34	18,07
77	36	27	104,94	0,09	24,05	55,68	0,29	19,13
78	36	27	108,19	0,05	24,01	53,06	0,84	18,14
79	53	42	119,14	0,11	25,55	54,95	0,49	19,77
80	33	27	105,18	0,06	24,67	56,67	0,37	19,52
81	33	27	126,78	0,35	22,06	47,23	0,80	16,19
82	53	45	120,55	0,37	25,39	57,20	1,15	19,70
83	28	25	107,11	0,42	24,48	51,46	0,48	18,73
84	47	45	118,37	0,06	25,45	56,01	1,08	20,41
85	30	28	120,64	0,01	22,93	50,64	0,88	17,09
86	30	29	106,03	0,09	24,40	55,37	0,70	19,39

87	28	27	105,49	0,20	24,73	54,20	0,50	19,03
88	29	30	109,08	0,20	23,78	51,58	0,16	18,94
89	32	34	108,43	0,24	25,21	58,25	0,26	19,86
90	46	49	118,86	0,16	25,80	50,97	0,08	21,10
91	30	33	105,17	0,17	24,64	56,27	0,48	19,52
92	38	42	126,08	0,07	21,51	45,70	0,24	15,69
93	31	36	119,60	0,05	22,38	49,18	0,91	16,71
94	30	35	118,73	0,04	22,87	50,74	0,25	17,09
95	27	33	106,81	0,22	24,45	52,28	1,01	18,78
96	34	44	109,36	0,59	25,23	58,26	0,08	19,89
97	37	47	116,05	0,22	17,41	32,47	0,39	13,69
98	37	48	118,42	0,01	20,39	55,51	0,70	17,63
99	42	56	118,57	0,12	21,88	48,55	0,08	16,24
100	35	47	121,13	0,06	22,29	48,72	0,82	16,65
101	35	48	119,73	0,05	21,88	47,53	0,26	16,30
102	32	46	109,25	0,37	20,90	56,10	0,33	16,16
103	30	45	115,20	0,06	22,88	60,23	1,82	19,43
104	31	47	125,38	0,01	23,19	61,22	1,62	19,24
105	25	39	120,06	0,15	22,21	50,46	0,32	18,11
106	30	47	98,11	0,14	21,64	45,22	0,49	17,48
107	27	44	105,12	0,01	22,36	48,14	0,17	17,93
108	27	45	99,69	0,15	20,78	45,40	0,75	17,47
109	26	44	106,21	0,05	21,77	53,09	0,14	17,64
110	25	43	107,59	0,02	21,75	50,87	0,05	17,31
111	25	44	106,01	0,04	21,24	49,80	0,23	16,98
112	27	50	110,41	0,00	22,24	54,87	1,44	18,90
113	26	48	109,53	0,22	22,23	51,38	0,82	18,02
114	25	49	109,35	0,14	20,69	46,42	0,12	16,24
115	25	47	100,61	0,05	25,74	57,67	1,68	20,53
116	33	66	116,47	0,13	23,49	54,54	0,30	17,96
117	15	30	123,70	0,34	22,08	58,09	0,49	18,42
118	19	38	124,84	0,01	23,33	62,76	0,05	18,58
119	19	39	112,35	0,07	18,97	53,15	0,24	15,98
120	19	41	126,88	0,22	21,66	59,68	0,90	17,90
121	19	42	125,92	0,11	21,94	58,47	0,42	18,35
122	14	30	127,83	0,26	25,54	58,28	0,29	22,57
123	15	34	132,89	0,23	23,70	56,50	0,30	20,60
124	15	35	132,68	0,10	23,74	56,69	0,32	20,62
125	15	36	132,79	0,07	23,74	56,76	0,33	20,63
126	15	36	132,71	0,14	23,74	56,79	0,30	20,62
127	15	37	133,01	0,00	23,76	56,88	0,35	20,65

Tabela 15 - Tabela com 127 RNA's treinadas com diferentes neurônios na camada escondida e simulada com 15.625 conjuntos desconhecidos após o treinamento.

			Erros (Valor Real)					
Neurônios	**Épocas de Treinamento**	**Tempo Total (seg.)**	**MaxTreina**	**MinTreina**	**MedTreina**	**MaxTeste**	**MinTeste**	**MedTeste**
1	2695	15	137,61	0,71	27,87	138,90	0,01	23,84
2	46	4	101,68	0,04	24,38	103,94	0,00	19,29
3	41	1	95,54	0,01	21,42	95,54	0,00	15,78
4	44	1	102,74	0,35	22,80	102,74	0,01	15,47
5	35	1	114,84	0,02	25,78	116,40	0,00	21,51
6	20	1	105,61	0,20	17,88	105,61	0,00	15,76
7	26	1	107,84	0,06	22,16	107,84	0,00	14,55
8	26	1	102,04	0,09	23,12	102,04	0,00	15,39
9	26	1	101,04	0,28	20,26	101,04	0,00	13,86
10	34	2	109,78	0,11	17,97	109,78	0,00	16,09
11	32	1	108,90	0,01	17,46	108,90	0,00	15,46
12	31	1	107,26	0,00	16,67	107,26	0,00	14,65
13	38	2	95,68	0,12	22,94	95,68	0,00	16,05
14	27	1	110,35	0,30	20,00	110,35	0,00	16,62
15	23	1	133,53	0,63	23,13	133,53	0,00	21,23
16	28	1	118,65	0,05	20,36	118,65	0,00	16,92
17	27	2	103,52	0,03	21,96	103,52	0,00	17,73
18	23	1	129,69	0,19	22,92	129,69	0,00	18,42
19	17	1	129,18	0,06	20,99	129,18	0,01	20,67
20	18	1	108,97	0,33	17,71	108,97	0,00	16,64
21	20	2	132,99	0,12	19,81	132,99	0,00	18,79
22	21	2	127,22	0,13	18,52	127,22	0,00	17,02
23	25	2	111,59	0,27	21,28	111,59	0,01	15,25
24	23	2	117,44	0,29	22,03	119,20	0,00	20,40
25	25	2	128,26	0,00	23,78	129,30	0,00	21,75
26	21	2	128,12	0,20	25,37	129,66	0,01	23,64
27	20	2	135,69	0,09	25,50	137,76	0,00	23,76
28	21	2	129,77	0,22	23,63	130,96	0,00	21,41
29	21	2	129,70	0,09	25,10	131,48	0,00	23,13
30	22	3	132,34	0,00	24,12	133,19	0,00	21,84
31	20	2	131,99	0,27	24,00	133,07	0,00	21,72
32	19	3	131,80	0,23	23,74	133,26	0,00	21,37
33	19	3	130,67	0,41	25,44	132,59	0,00	23,35
34	21	3	133,20	0,40	24,28	134,07	0,00	21,96
35	19	3	133,88	0,12	24,38	134,90	0,00	22,03
36	18	3	134,33	0,31	24,43	135,49	0,01	22,05
37	18	3	134,61	0,18	24,25	136,01	0,00	21,79
38	20	3	131,49	0,09	23,75	133,13	0,00	21,37
39	21	4	134,04	0,19	24,59	134,90	0,00	22,23
40	20	4	134,09	0,40	24,57	135,05	0,00	22,20
41	19	4	134,46	0,20	24,64	135,49	0,00	22,25
42	70	12	108,00	0,02	23,02	109,59	0,00	19,79
43	70	13	107,94	0,36	23,22	109,49	0,00	19,97
44	73	15	114,97	0,33	25,21	116,22	0,00	21,75

45	71	14	110,20	0,02	23,67	111,52	0,01	20,04
46	72	15	111,45	0,15	23,76	112,71	0,00	20,00
47	73	15	108,36	0,17	23,49	109,87	0,01	20,34
48	74	18	108,48	0,44	23,47	110,01	0,00	20,40
49	74	22	108,67	0,02	23,68	110,13	0,00	20,50
50	70	22	113,74	0,14	23,94	114,91	0,00	20,00
51	93	30	104,34	0,23	15,58	104,34	0,00	13,18
52	91	30	103,45	0,05	15,37	103,45	0,00	12,99
53	95	33	103,90	0,30	15,42	103,90	0,00	13,07
54	93	33	111,77	0,05	22,95	111,77	0,00	16,35
55	92	34	103,89	0,15	15,37	103,89	0,00	13,06
56	93	35	102,15	0,09	15,10	102,15	0,00	12,81
57	93	37	110,91	0,26	22,85	110,91	0,00	16,28
58	93	38	102,13	0,07	15,07	102,13	0,00	12,81
59	92	39	111,95	0,23	23,04	111,95	0,00	16,40
60	93	41	101,28	0,06	14,92	101,28	0,00	12,70
61	97	45	112,62	0,16	23,14	112,62	0,00	16,47
62	100	48	101,14	0,04	14,85	101,14	0,00	12,69
63	98	48	101,29	0,20	14,86	101,29	0,00	12,72
64	99	51	114,80	0,01	23,48	114,80	0,00	16,71
65	98	51	100,46	0,07	14,80	100,46	0,00	12,70
66	97	51	99,99	0,30	14,73	99,99	0,00	12,66
67	101	57	100,64	0,00	14,75	100,64	0,00	12,69
68	98	56	100,82	0,02	14,77	100,82	0,00	12,71
69	101	61	100,28	0,04	14,70	100,28	0,00	12,64
70	86	52	97,39	0,20	14,50	97,39	0,00	12,45
71	85	54	113,45	0,16	23,52	113,45	0,00	16,76
72	88	58	97,75	0,32	14,50	97,75	0,00	12,46
73	83	57	99,56	0,03	14,57	99,56	0,00	12,53
74	57	40	121,97	0,07	25,21	121,99	0,00	20,86
75	33	24	130,86	0,56	22,84	130,86	0,00	18,79
76	32	24	111,46	0,11	23,92	111,46	0,00	19,60
77	36	28	104,94	0,09	24,05	104,94	0,00	20,67
78	36	29	108,19	0,05	24,01	108,19	0,00	19,64
79	53	43	119,14	0,11	25,55	121,43	0,00	21,30
80	33	28	105,18	0,06	24,67	105,18	0,00	21,18
81	33	29	126,78	0,35	22,06	126,78	0,00	18,33
82	53	47	120,55	0,37	25,39	122,72	0,01	21,09
83	28	26	107,11	0,42	24,48	107,11	0,00	20,18
84	47	47	118,37	0,06	25,45	119,42	0,00	22,29
85	30	30	120,64	0,01	22,93	120,64	0,00	18,83
86	30	30	106,03	0,09	24,40	106,03	0,01	21,11
87	28	29	105,49	0,20	24,73	105,49	0,00	20,35
88	29	31	109,08	0,20	23,78	109,08	0,01	20,89
89	32	35	108,43	0,24	25,21	109,54	0,00	21,56
90	46	51	118,86	0,16	25,80	120,20	0,01	22,39
91	30	35	105,17	0,17	24,64	105,17	0,00	21,21
92	38	44	126,08	0,07	21,51	126,08	0,00	17,69

93	31	38	119,60	0,05	22,38	119,60	0,00	18,25
94	30	37	118,73	0,04	22,87	118,73	0,00	18,73
95	27	35	106,81	0,22	24,45	106,81	0,00	20,08
96	34	46	109,36	0,59	25,23	110,52	0,00	21,62
97	37	49	116,05	0,22	17,41	116,05	0,00	15,18
98	37	50	118,42	0,01	20,39	118,42	0,00	18,11
99	42	59	118,57	0,12	21,88	118,57	0,00	17,56
100	35	49	121,13	0,06	22,29	121,13	0,01	18,18
101	35	51	119,73	0,05	21,88	119,73	0,00	17,67
102	32	48	109,25	0,37	20,90	121,80	0,00	16,84
103	30	47	115,20	0,06	22,88	120,78	0,00	18,23
104	31	49	125,38	0,01	23,19	134,80	0,00	18,43
105	25	41	120,06	0,15	22,21	120,06	0,00	17,91
106	30	49	98,11	0,14	21,64	109,10	0,00	17,62
107	27	46	105,12	0,01	22,36	113,12	0,01	18,13
108	27	47	99,69	0,15	20,78	110,86	0,00	17,24
109	26	47	106,21	0,05	21,77	120,23	0,00	17,95
110	25	45	107,59	0,02	21,75	116,55	0,01	17,67
111	25	47	106,01	0,04	21,24	114,39	0,00	17,26
112	27	52	110,41	0,00	22,24	110,41	0,01	17,87
113	26	51	109,53	0,22	22,23	118,47	0,01	18,07
114	25	50	109,35	0,14	20,69	109,35	0,00	16,67
115	25	50	100,61	0,05	25,74	106,51	0,01	23,94
116	33	70	116,47	0,13	23,49	120,38	0,00	18,63
117	15	32	123,70	0,34	22,08	124,86	0,00	19,51
118	19	40	124,84	0,01	23,33	127,43	0,00	18,91
119	19	41	112,35	0,07	18,97	113,86	0,00	15,70
120	19	42	126,88	0,22	21,66	128,01	0,00	18,64
121	19	44	125,92	0,11	21,94	127,08	0,00	19,25
122	14	32	127,83	0,26	25,54	128,34	0,00	24,19
123	15	36	132,89	0,23	23,70	134,12	0,00	21,38
124	15	37	132,68	0,10	23,74	133,92	0,01	21,41
125	15	38	132,79	0,07	23,74	134,04	0,00	21,41
126	15	41	132,71	0,14	23,74	133,97	0,00	21,40
127	15	39	133,01	0,00	23,76	134,26	0,00	21,42

Anexo 03: Mapas de Sensibilidade

Tabela 16 - Mapa de Sensibilidade avaliando o sinal de cada fator de sensibilidade. Contém 127 RNA's com diferentes quantidades de neurônios na camada escondida, todas treinadas partindo de 01 até 15 épocas.

Épocas	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Tempo(s) / Neurônios	120	177	333	427	539	623	712	798	891	992	1059	1197	1244	1387	1477
1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
2	0	0	0	4	0	1	0	0	0	0	0	0	0	0	0
3	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0
7	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0
8	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0
9	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0
10	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
15	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
16	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
17	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
18	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
19	0	1	0	0	0	0	0	0	0	0	0	0	1	0	1
20	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0
21	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0
22	0	1	0	0	0	0	0	0	0	1	0	1	0	0	0
23	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0
24	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0
25	0	1	0	0	0	0	0	0	0	0	0	0	1	0	1
26	0	1	0	0	0	0	0	0	0	0	0	0	1	0	1
27	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0
28	0	1	0	0	0	0	0	0	0	0	0	1	0	1	0
29	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0
30	0	1	0	0	0	0	0	0	0	0	1	0	1	0	0
31	0	1	0	0	0	0	0	0	0	0	1	0	1	0	0
32	0	1	0	0	0	0	0	0	0	0	1	0	1	0	0
33	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0
34	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0
35	0	1	0	0	0	0	0	0	0	1	0	1	0	0	0
36	0	1	0	0	0	0	0	0	0	1	0	1	0	0	0
37	0	1	0	0	0	0	0	0	0	1	0	1	0	0	0
38	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0
39	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0

40	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
41	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
42	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0
43	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0
44	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0
45	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0
46	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0
47	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0
48	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0
49	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0
50	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0
51	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
52	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
53	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
54	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
55	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
56	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
57	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
58	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
59	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
60	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
61	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
62	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
63	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
64	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
65	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
66	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
67	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
68	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
69	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
70	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
71	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
72	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
73	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
74	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
75	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
76	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
77	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
78	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
79	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
80	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
81	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
82	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
83	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
84	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
85	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
86	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
87	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

88	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
89	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
90	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
91	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
92	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
93	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
94	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
95	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
96	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
97	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
98	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
99	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
101	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
102	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
103	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
104	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
105	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
106	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
107	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
108	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
109	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
110	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
111	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
112	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
113	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
114	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
115	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
116	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
117	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0
118	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0
119	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0
120	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0
121	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0
122	0	1	0	0	0	0	0	1	0	0	0	0	0	1	1
123	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0
124	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0
125	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0
126	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0
127	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0

Tabela 17 - Mapa de Sensibilidade avaliando o sinal de cada fator de sensibilidade e sua ordem de grandeza no processo físico. Contém 127 RNA's com diferentes quantidades de neurônios na camada escondida, todas treinadas partindo de 01 até 15 épocas.

Épocas	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Tempo(s) / Neurônios	120	177	295	433	532	618	709	796	892	990	1084	1167	1244	1349	1449
1	**0**	1	1	1	2	1	1	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
2	**0**	1	1	5	1	2	1	1	**0**	1	1	1	1	1	1
3	**0**	1	1	2	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
4	**0**	1	1	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
5	**0**	1	1	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	1	1	1	1
6	**0**	2	1	1	1	1	2	**0**	1	1	1	1	**0**	1	1
7	**0**	2	1	1	1	1	1	1	1	1	2	1	1	1	1
8	**0**	2	**0**	1	1	1	1	2	**0**	1	1	1	1	1	1
9	**0**	2	**0**	1	1	1	1	3	1	1	1	1	1	1	1
10	**0**	2	**0**	1	1	1	1	2	1	1	1	1	1	1	1
11	**0**	2	**0**	1	1	1	1	1	1	1	1	1	1	1	1
12	**0**	2	**0**	1	1	1	1	**0**	1	1	1	1	1	1	1
13	**0**	2	**0**	1	1	1	1	**0**	1	1	1	1	1	1	1
14	**0**	2	1	1	2	2	2	2	2	2	2	1	1	**0**	**0**
15	**0**	2	2	2	2	2	2	2	2	2	2	2	1	1	1
16	**0**	2	3	2	2	2	2	2	2	2	2	1	1	1	1
17	**0**	2	3	2	2	2	2	2	2	1	1	1	1	1	1
18	**0**	2	3	2	2	2	2	2	2	2	2	2	1	1	1
19	**0**	2	2	2	2	2	2	2	2	2	1	1	3	1	2
20	**0**	2	2	2	2	2	2	2	2	1	1	2	1	**0**	**0**
21	**0**	2	2	2	2	2	2	2	1	1	2	1	1	**0**	**0**
22	**0**	2	2	2	2	2	2	2	1	2	2	2	1	1	1
23	**0**	2	2	2	2	1	1	2	**0**	1	**0**	1	1	1	1
24	**0**	2	2	2	2	2	2	2	1	1	1	1	1	2	2
25	**0**	2	2	2	2	2	2	2	1	1	1	1	1	2	2
26	**0**	2	2	2	2	2	2	2	1	1	1	1	2	2	1
27	**0**	2	2	2	2	2	2	1	1	1	1	1	2	2	1
28	**0**	2	2	2	2	2	2	1	1	1	1	1	2	1	1
29	**0**	2	2	2	2	2	2	1	1	1	1	1	2	**0**	**0**
30	**0**	2	2	2	2	2	2	1	1	1	2	1	2	1	1
31	**0**	2	2	2	2	2	1	1	1	1	1	1	1	1	1
32	**0**	2	2	2	2	2	1	1	1	1	1	2	1	1	**0**
33	**0**	2	2	2	2	2	1	1	1	1	1	2	**0**	1	**0**
34	**0**	2	2	2	2	2	1	1	1	1	1	2	1	1	1
35	**0**	2	2	2	2	2	1	1	1	2	1	2	1	1	1
36	**0**	2	2	2	2	2	1	1	1	2	1	1	1	1	1
37	**0**	2	2	2	2	2	1	1	1	2	1	1	**0**	**0**	**0**
38	**0**	2	2	2	2	1	1	1	1	2	**0**	**0**	1	1	**0**
39	**0**	2	2	2	2	1	1	1	1	1	1	1	1	1	1
40	**0**	2	2	2	2	1	1	1	1	1	1	1	1	1	1
41	**0**	2	2	2	2	1	1	1	1	1	1	1	1	1	1
42	**0**	2	1	2	1	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
43	**0**	2	1	2	1	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

44	0	2	1	2	1	0	0	0	0	0	0	0	0	0	0
45	0	2	1	2	1	0	0	0	0	0	0	0	0	0	0
46	0	2	1	2	1	0	0	0	0	0	0	0	0	0	0
47	0	2	1	2	1	0	0	0	0	0	0	0	0	0	0
48	0	2	1	2	1	0	0	0	0	0	0	0	0	0	0
49	0	2	1	2	1	0	0	0	0	0	0	0	0	0	0
50	0	2	1	2	1	0	0	0	0	0	0	0	0	0	0
51	0	2	1	1	1	1	1	0	0	0	0	0	0	0	0
52	0	2	1	1	1	1	1	0	0	0	0	0	0	0	0
53	0	2	1	1	1	1	1	0	0	0	0	0	0	0	0
54	0	2	1	1	1	1	1	0	0	0	0	0	0	0	0
55	0	2	1	1	1	1	0	0	0	0	0	0	0	0	0
56	0	2	1	1	1	1	0	0	0	0	0	0	0	0	0
57	0	2	1	1	1	1	0	0	0	0	0	0	0	0	0
58	0	2	1	1	1	1	0	0	0	0	0	0	0	0	0
59	0	2	1	1	1	1	0	0	0	0	0	0	0	0	0
60	0	2	1	1	1	1	0	0	0	0	0	0	0	0	0
61	0	2	1	1	1	1	0	0	0	0	0	0	0	0	0
62	0	2	1	1	1	0	0	0	0	0	0	0	0	0	0
63	0	2	1	1	1	0	0	0	0	0	0	0	0	0	0
64	0	2	1	1	1	0	0	0	0	0	0	0	0	0	0
65	0	2	1	1	1	0	0	0	0	0	0	0	0	0	0
66	0	2	1	1	0	0	0	0	0	0	0	0	0	0	0
67	0	2	1	1	0	0	0	0	0	0	0	0	0	0	0
68	0	2	1	1	0	0	0	0	0	0	0	0	0	0	0
69	0	2	1	1	0	0	0	0	0	0	0	0	0	0	0
70	0	2	1	1	0	0	0	0	0	0	0	0	0	0	0
71	0	2	1	1	0	0	0	0	0	0	0	0	0	0	0
72	0	2	1	1	0	0	0	0	0	0	0	0	0	0	0
73	0	2	1	1	0	0	0	0	0	0	0	0	0	0	0
74	0	2	1	1	0	0	0	0	0	0	0	0	0	0	0
75	0	2	1	1	0	0	0	0	0	0	0	0	0	0	0
76	0	2	1	1	0	0	0	0	0	0	0	0	0	0	0
77	0	2	1	1	0	0	0	0	0	0	0	0	0	0	0
78	0	2	1	1	0	0	0	0	0	0	0	0	0	0	1
79	0	2	1	1	0	0	0	0	0	0	0	0	0	0	1
80	0	2	1	1	0	0	0	0	0	0	0	0	0	0	0
81	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0
82	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
83	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
84	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
85	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
86	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
87	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
88	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
89	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
90	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
91	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0

92	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
93	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
94	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
95	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
96	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
97	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
98	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
99	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
100	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
101	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
102	0	2	0	1	1	1	1	1	1	1	1	1	1	1	1
103	0	2	0	1	1	1	1	1	1	1	1	1	1	1	1
104	0	2	0	1	1	1	1	1	1	1	1	1	1	1	1
105	0	2	0	1	1	1	1	1	1	1	1	1	1	1	1
106	0	2	0	1	1	1	1	1	1	1	1	1	1	1	1
107	0	2	0	1	1	1	1	1	1	1	1	1	1	1	1
108	0	2	0	1	1	1	1	1	1	1	1	1	1	1	1
109	0	2	0	1	1	1	1	1	1	1	1	1	1	1	1
110	0	2	0	1	1	1	1	1	1	1	1	1	1	1	1
111	0	2	0	1	1	1	1	1	1	1	1	1	1	1	1
112	0	2	0	1	1	1	1	1	1	1	1	1	1	1	1
113	0	2	0	1	1	1	1	1	1	1	1	1	1	1	1
114	0	2	0	1	1	1	1	1	1	1	1	1	1	1	1
115	0	2	0	1	1	1	1	1	1	1	1	1	1	1	1
116	0	2	0	1	1	1	1	1	1	1	1	1	1	1	1
117	0	2	0	1	1	1	1	2	0	1	0	0	1	1	1
118	0	2	0	1	1	1	1	2	0	1	1	1	1	0	0
119	0	2	0	1	1	1	1	2	0	1	1	1	1	0	0
120	0	2	0	1	1	1	1	2	0	1	1	1	1	0	0
121	0	2	0	1	1	1	1	2	0	1	1	1	1	0	0
122	0	2	0	1	1	1	1	2	0	1	0	0	1	2	2
123	0	2	0	1	1	1	1	0	2	1	1	0	0	1	1
124	0	2	0	1	1	1	1	0	2	1	0	0	0	1	1
125	0	2	0	1	1	1	1	0	2	1	0	0	0	1	1
126	0	2	0	1	1	1	1	0	2	0	0	0	0	1	1
127	0	2	0	1	1	1	1	0	2	1	0	0	0	1	1

Anexo 04: Constante de Tempo do Processo de Aprendizagem

Uma estratégia para estimar a quantidade de épocas necessárias para avaliação do treinamento de uma RNA é tratar a curva de aprendizado como uma curva exponencial *f(x)*.

Seja a seguinte função $f(x) = e^{-x}$, veja Figura 24:

Para $x = 0$, tem-se $f(x) = 1$.

Para $x \to \infty$, tem-se $f(x) = 1/e^{\infty} = 1/\infty = 0$.

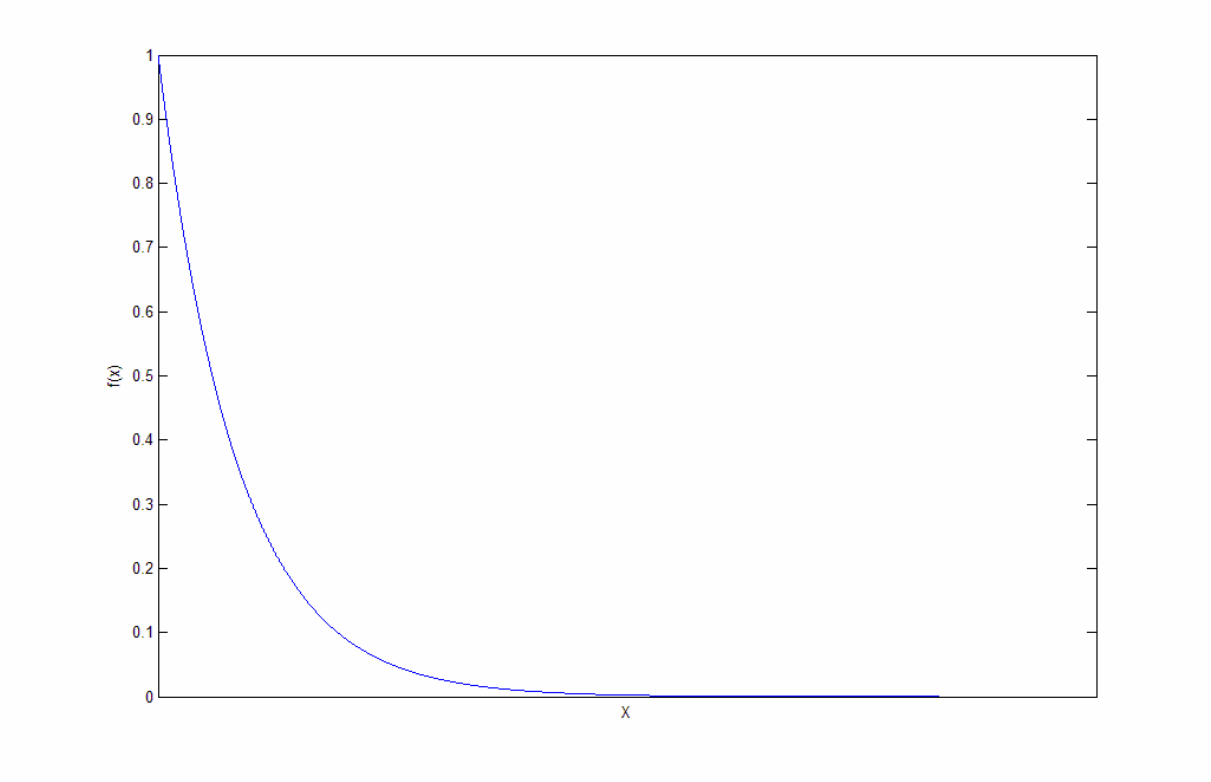

Figura 24 - Curva exponencial *f(x)*

Considerando $f(x) = K\, e^{-x}$, tem-se:

Para $x = 0$, tem-se $f(x) = K$, veja Figura 25.

Para $x \to \infty$, tem-se $f(x) = 0$.

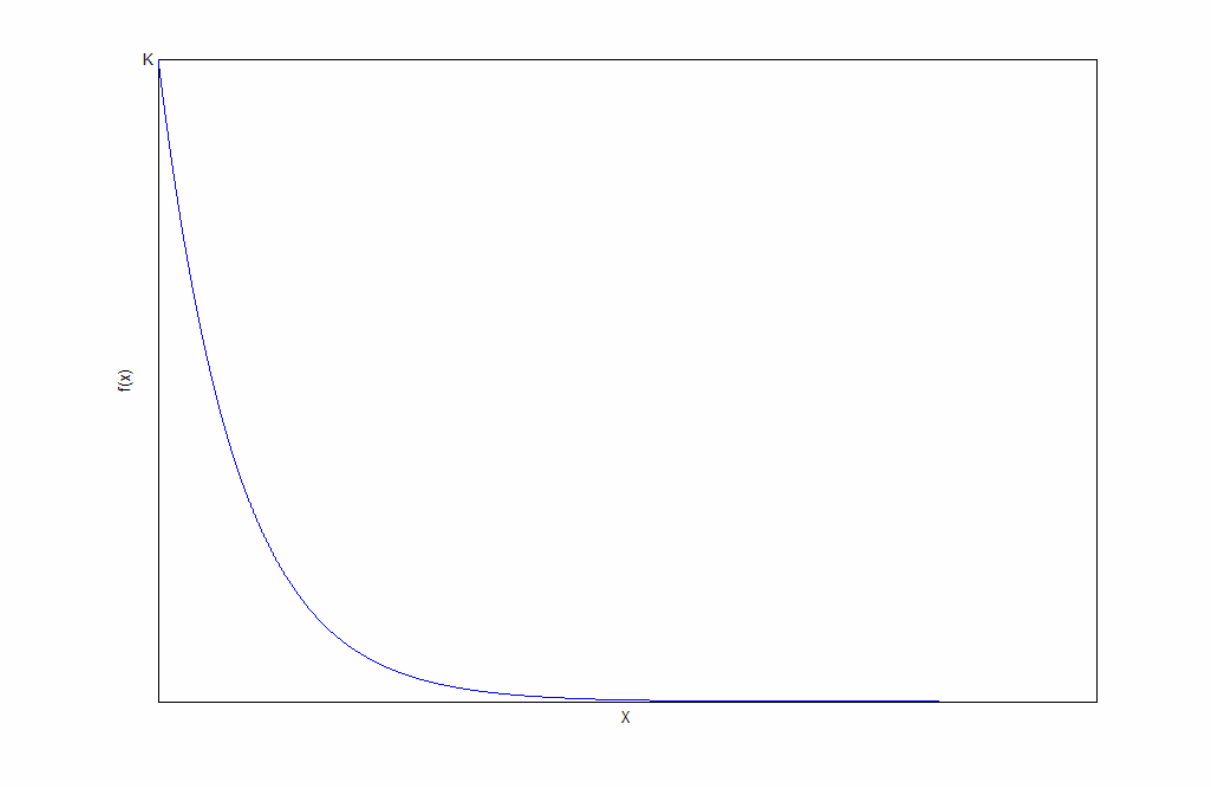

Figura 25 - Curva exponencial *f(x) = k* quando *x = 0*

Se $f(x) = K\, e^{-x/\tau}$, quando $\tau_1 > \tau_2 > \tau_3$, tem-se:

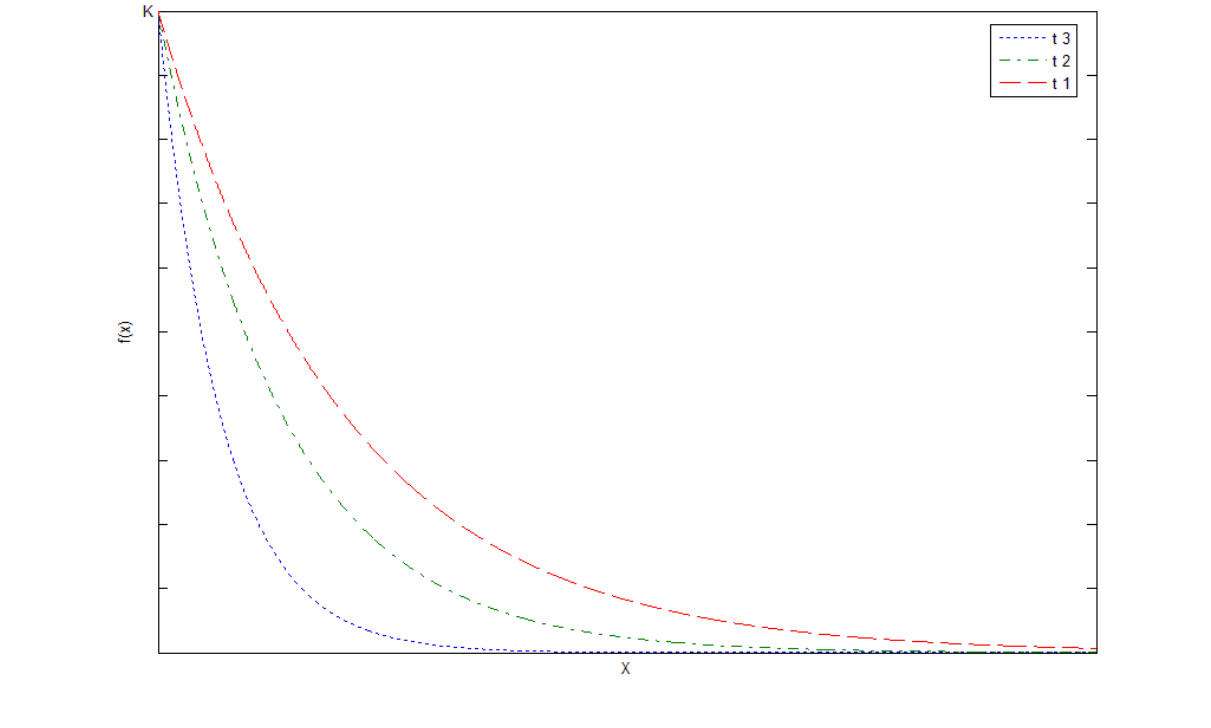

Figura 26 - Curva exponencial para diferentes constantes

Para estimar a curva de aprendizagem da RNA, é necessário estipular a constante de tempo τ.

Logo se $f(x) = K\, e^{-x/\tau}$, então temos:

$$f(x) = erro_{inicial}\, e^{-x/\tau} \tag{30}$$

onde, $x \geq 0$ e $\tau > 0$

Considerando os dois primeiros valores da curva de erro correspondentes as 02(duas) primeiras épocas de treinamento de uma RNA, com a quantidade de neurônios na camada escondida calculada pelo teorema apresentado por Hecht-Nielsen (1989) e com a matriz de pesos inicializada com zero, consegue-se estimar a constante de tempo τ, por aproximação entre a curva de treinamento e a curva exponencial, veja Figura 27.

A constante de tempo τ representa a quantidade de épocas necessárias para treinamento de uma RNA para que, a partir de então, seu aprendizado possa ser avaliado em aproximadamente 98,2%.

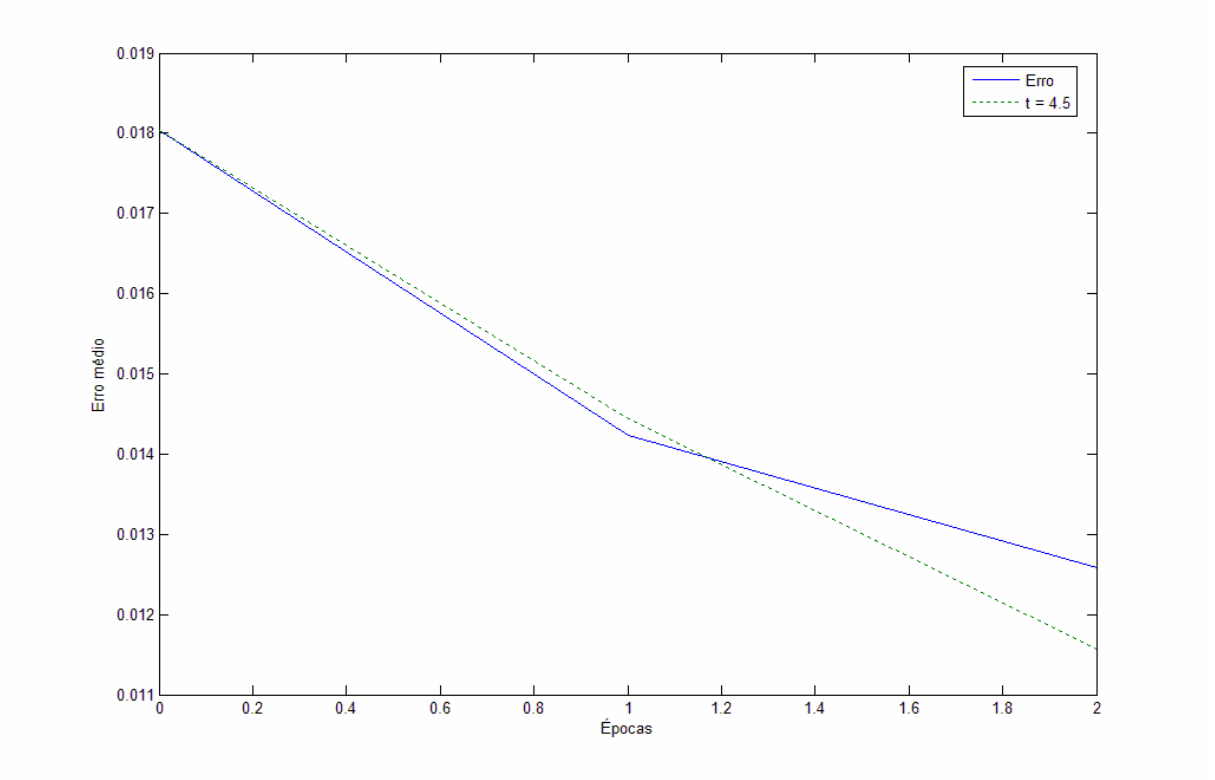

Figura 27 - Aproximação entre curva de aprendizado e constante

Pode-se observar na Figura 27, um exemplo de aproximação, onde a RNA com 13 neurônios na camada escondida, com a matriz de pesos inicializada com zeros, partiu do erro 0,018042, convergindo na primeira época para o erro 0,014233, em seguida, convergindo na segunda época para o erro 0,012587. Por aproximação, a curva do treinamento é relacionada a curva exponencial com constante de tempo $\tau = 4{,}5$. Logo, estima-se que com 04 constantes de tempo, a curva de aprendizado da RNA convergirá aproximadamente a 98,2% do aprendizado. Cabe ressaltar que a constante de tempo foi determinada para uma estrutura neural formada por "*2n+1*" neurônios na camada escondida, segundo o teorema apresentado por Hecht-Nielsen (1989). Observe que a constante de tempo τ pode ser um parâmetro de ajuste experimental.

É apresentada abaixo a curva de aprendizado da RNA com 13 neurônios na camada escondida até atingir ao objetivo final, veja Figura 28.

Em seguida, apresenta-se a aproximação entre a curva de aprendizado e a curva exponencial cuja constante de tempo é $\tau = 4{,}5$, veja Figura 29.

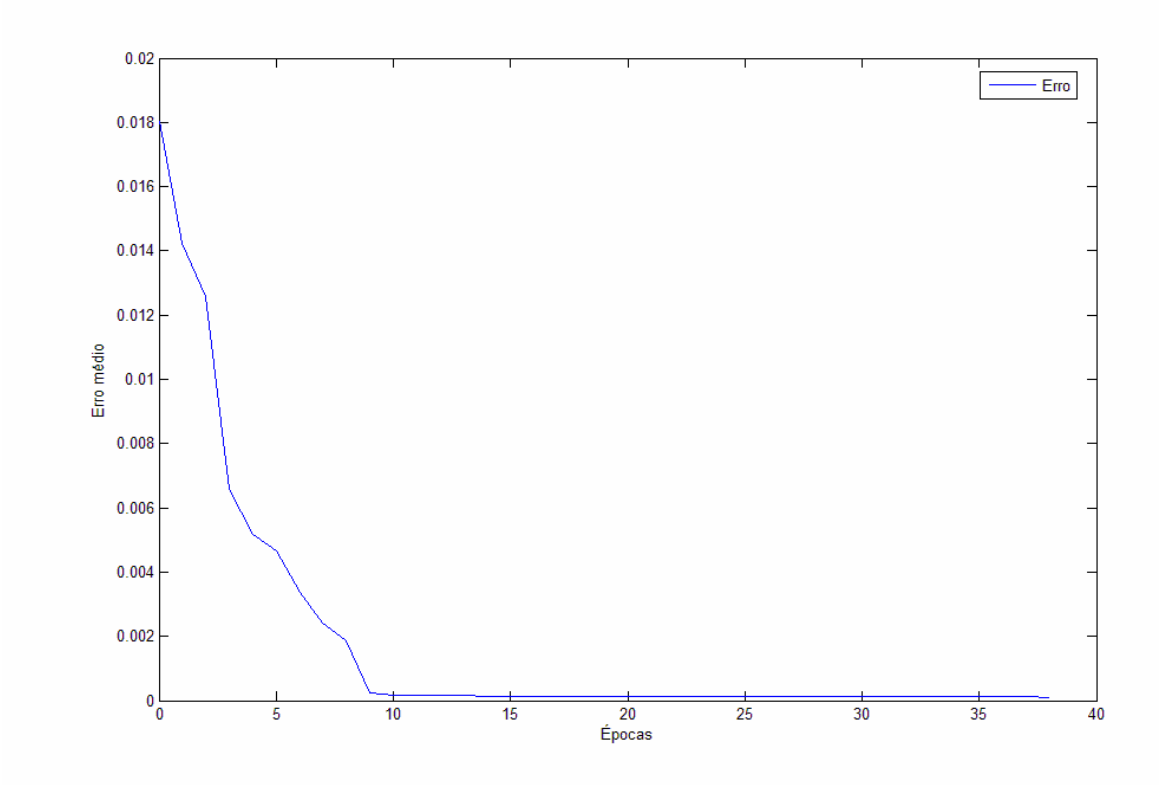

Figura 28 - Curva do erro de aprendizagem da RNA

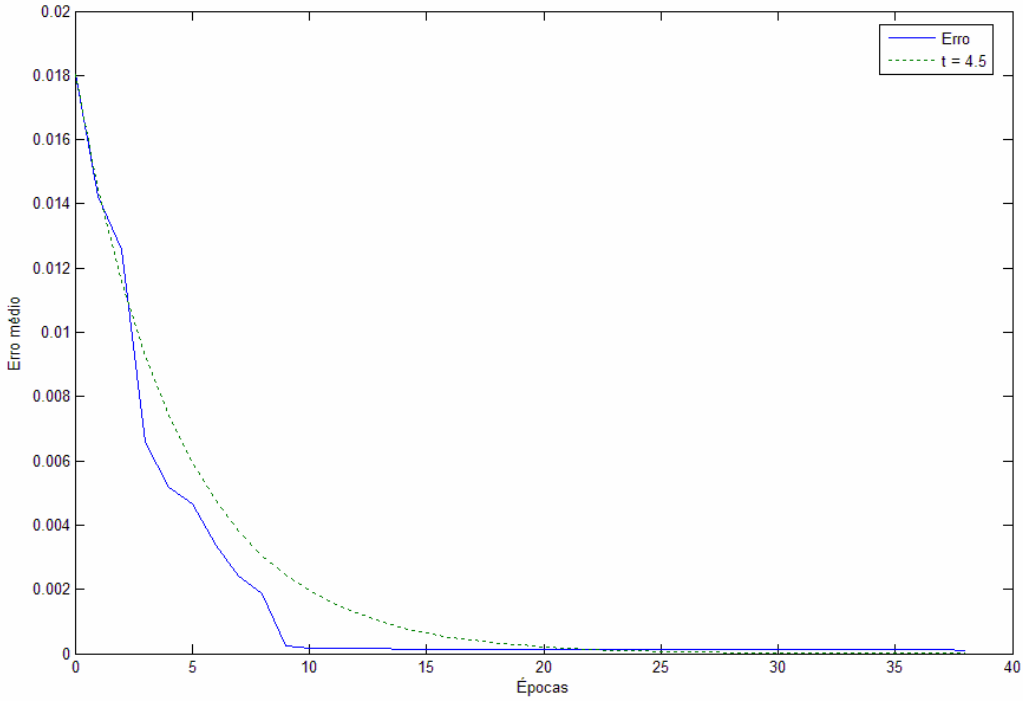

Figura 29 - Aproximação entre a curva do erro com a curva exponencial