UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA

ROGER NOBRE DE GOUVEIA

Aplicação Híbrida de Inteligência Computacional voltada à Predição de Séries Temporais

PORTO ALEGRE, 2012

ROGER NOBRE DE GOUVEIA

Aplicação Híbrida de Inteligência Computacional voltada à Predição de Séries Temporais

Monografia por Roger Nobre de Gouveia como trabalho de conclusão de curso na Universidade Federal do Rio Grande do Sul para a obtenção do título de bacharel em Ciência da Computação.
Orientador: Prof. Dr. Dante Barone

Porto Alegre, 2012

Roger Nobre de Gouveia

Aplicação Híbrida de Inteligência Computacional voltada à Predição de Séries Temporais

Trabalho de Conclusão apresentado ao curso de Ciência da Computação da Universidade Federal do Rio Grande do Sul como requisito parcial para obtenção do título de Bacharel em Ciência da Computação.

Comissão Examinadora

Prof. Dante Barone, Dr.
Profª. Renata Galante, Drª.
Profª. Rosa Maria Vicari, Drª.

Porto Alegre, dezembro de 2012

RESUMO

Redes Neurais Artificiais (RNAs) possuem capacidade de aprender padrões, e depois reconhecê-los mesmo quando inseridos em ambientes ruidosos. Séries temporais (medições ordenadas ao longo do tempo) comumente possuem uma grande quantidade de ruído aleatório, o que dificulta a sua previsão. A utilização de RNAs para previsão de séries temporais parece adequada. O problema principal são a quantidade de decisões de projeto para a construção de uma RNA. No caso de previsão de séries temporais, a distribuição neuronal (DN) causa um dos maiores impactos. Algoritmos genéticos (AGs) são uma heurística poderosa adequados para resolver problemas de natureza combinatorial. É bastante lógico combinar AGs e RNAs para previsão de séries temporais. O AG cria RNAs com diferentes DNs. As diversas RNAs fornecem diferentes previsões, o que resulta em competição. Ao final do processo evolutivo, espera-se que a melhor RNA fosse capaz de realizar a melhor previsão. Apesar de seu comportamento aparentemente “inteligente”, os resultados ficaram abaixo do esperado. Os resultados foram comparáveis ao consolidado método *Autoregressive Integrated Moving Average* (ARIMA), mas foram ligeiramente inferiores.

Palavras-chave: Redes Neurais, Algoritmo Genético, Previsão de Série Temporal

ABSTRACT

Artificial Neural Networks (ANN) are capable of learning patterns, and then recognizing them even when inserted in noisy environments. Time series (measurements ordered in time) normally have great quantities of random noise, what hampers their forecasting. The utilization of ANNs to time series forecasting seems adequated. The main problem is the amount of project decisions to build an ANN. In the case of time series forecasting, the neural distribution (ND) causes one of the biggest impacts. Genetic algorithms (AGs) are a powerful heuristics able to solve problems of combinatorial nature. It is logical combinate GAs and ANNs for time series forecasting. The GA creates ANNs with different forecastings, what leads to competition. In the end of the evolutionary process, it is expected that the best ANN be capable of doing the best forecasting. Although its behaviour seems “intelligent”, the results were under the expected. The results were comparable to the consolidated method Autoregressive Integrated Moving Average (ARIMA), yet slightly worse.

Keywords: Artificial Neural Networks, Genetic Algorithm, Time Series Forecasting

SUMÁRIO

1 INTRODUÇÃO

O desenvolvimento tecnológico da humanidade está crescendo de forma acelerada. A quantidade de informação gerada no mundo a cada dia está aumentando, assim como a necessidade de seu bom processamento. Nesse contexto, o desenvolvimento das Tecnologias de Informação e Comunicação (TICs) causa um grande impacto na vida das pessoas e empresas. Dentro das TICs, a área de Inteligência Artificial tem assumido um papel cada vez mais protagônico.

O campo de pesquisa em Inteligência Artificial nasceu em 1956, em Hanover, EUA [1], e, apesar de todas as dificuldades não previstas, tem desde então progredido consideravelmente, infiltrando-se progressivamente no cotidiano. Ele é interdisciplinar e frequentemente se inspira nos sistemas mais complexos que conhecemos - os sistemas biológicos.

Uma das diversas sub-áreas da Inteligência Artificial é a de Redes Neurais Artificiais (RNA). Elas são sistemas computacionais que tentam modelar a funcionalidade das redes neurais biológicas afim de adquirir características que lhes permitam ser mais adequadas à resolução de determinados problemas do que sistemas computacionais tradicionais. Uma dessas características mais distinguíveis é a sua capacidade de reconhecimento de padrões, o que as torna uma ótima ferramenta para o processamento de certos tipos de informação.

A escolha da metodologia para o correto processamento de informação é contextual. Dependendo do objetivo, a melhor forma de organizar e utilizar a informação varia. Quando não se possui dados suficientes para a descrição de um sistema (ou caso ele seja muito complexo) cujo comportamento varia ao longo do tempo, uma forma interessante de organizar a informação é através de uma série temporal.

Uma série temporal é um conjunto de medições ordenadas e equidistantes ao longo do tempo. Ela é uma boa forma de descrever externamente o comportamento de sistemas de níveis variados de complexidade. Tendo-se pelo menos a quantidade mínima de dados, e análisando-os corretamente, é possível que se encontre um

padrão oculto, tornando o comportamento desse sistema previsível dentro de uma margem de erro.

1.1 Objetivos

Esse trabalho tem por objetivo testar a utilização de um algoritmo genético como possível heurística para a escolha da topologia de uma rede neural artificial para a previsão de uma dada série temporal.

1.2 Justificativa

Ainda não existem regras claras para projetar uma rede neural artificial. Além do fato que o seu desempenho é contextual, existem inúmeras possibilidades de topologia, algoritmos de treinamento e funções de ativação. O espaço de possibilidades é simplesmente muito grande. Inicialmente as escolhas de projeto eram realizadas puramente através de tentativa e erro. Obviamente, além da ineficiência, um bom desempenho não é garantido. Surge então uma necessidade de criação de métodos para a seleção dessas possibilidades. Um algoritmo genético é um ótimo método para realizar análises efetivas e encontrar boas soluções nesse grande espaço de busca, automatizando, pelo menos em parte, as decisões de projeto de uma rede neural.

A classe de problemas abordada (previsão de séries temporais) é interessante por si só: Modelagem de um sistema, utilizando apenas informações empíricas de como ele interagiu no ambiente no passado, seguida da extrapolação de seu comportamento futuro.

1.3 Organização do trabalho

Essa monografia está organizada da seguinte forma:

O primeiro capítulo situa o leitor de forma geral na área de redes neurais artificiais, e a sua possível aplicação no problema de previsão de séries temporais.

O segundo capítulo aborda de forma geral os princípios do campo de redes neurais, enunciando seus pontos fortes, alguns de seus desafios, e mostrando uma visão geral de seu funcionamento.

No terceiro capítulo fundamentam-se os conceitos básicos dos algoritmos genéticos, oferencendo uma explicação sucinta sobre o seu funcionamento, suas qualidades e alguns de seus desafios.

O quarto capítulo descreve o algoritmo da implementação construído para esse trabalho, assim como a sua interface e funcionalidade.

O quinto capítulo mostra a metodologia utilizada para obtenção dos resultados apresentados.

No sexto capítulo é apresentada a validação do trabalho através da comparação dos resultados obtidos através da utilização da implementação criada para esse trabalho com outras metodologias.

O sétimo capítulo discute as conclusões que se pode tirar das experiências efetuadas nessa monografia.

2 SÉRIES TEMPORAIS

2.1 Visão geral

Uma série temporal é um conjunto de medidas ordenadas ao longo do tempo, normalmente em espaços regulares. Como exemplos podem ser citados a média das temperaturas diárias em Nova York ou o número de mortes mensais no trânsito de Porto Alegre durante o ano de 2012.

A maioria das técnicas para análise de séries temporais são técnicas estatísticas. Essas técnicas são divididas em paramétricas e não-paramétricas. As técnicas paramétricas assumem que existe uma estrutura estável sob o processo estocástico, o que permitiria uma descrição mais apurada. As técnicas não-paramétricas não assumem que a série temporal possua nenhuma estrutura estável.

Uma das áreas da Análise de Séries Temporais é a de Predição de Séries Temporais. Predição de Séries Temporais refere-se à utilização de um modelo que utiliza valores conhecidos de uma série temporal (valores passados) para prever valores futuros. Existem diversos modelos estatísticos para simulação estocástica para tentar prever o comportamento de uma série temporal. Um dos métodos utilizados para Predição de Séries Temporais, que não é estatístico, mas que vem demonstrando ótimo desempenho são as Redes Neurais Artificiais.

3 REDES NEURAIS ARTIFICIAIS

3.1 Visão geral

Uma rede neural artificial (RNA) é um sistema computacional análogo à uma rede neural biológica. Ao modelar os constituintes dessa última, a RNA adquire diversas características: emergência [2], auto-organização [3], capacidade de aprendizado, e robustez.

RNAs são usadas para modelar complexas relações entre os dados de entrada e saída ou para encontrar padrões nos dados, - mesmo em dados ruidosos, incompletos ou imprecisos - e quando bem construídas oferecem boas soluções (não necessariamente a melhor solução) para o problema dado.

3.2 Aprendizado

RNAs “aprendem” uma classe de problemas, e para cada instância dessa classe, oferecem uma solução. Esse aprendizado é, na verdade, um treinamento no qual a rede neural artificial é exposta a um conjunto de dados correspondentes a uma classe de problemas. A cada iteração de treino, a RNA se modifica para ser capaz de oferecer uma melhor solução para o conjunto de dados do problema. Escolher o número de iterações de treino é um problema por si só: Poucas iterações - RNA não aprende a classe de problemas; Muitas iterações - RNA não generaliza, aprendendo as instâncias da classe de problemas e não a classe (*overfitting*).

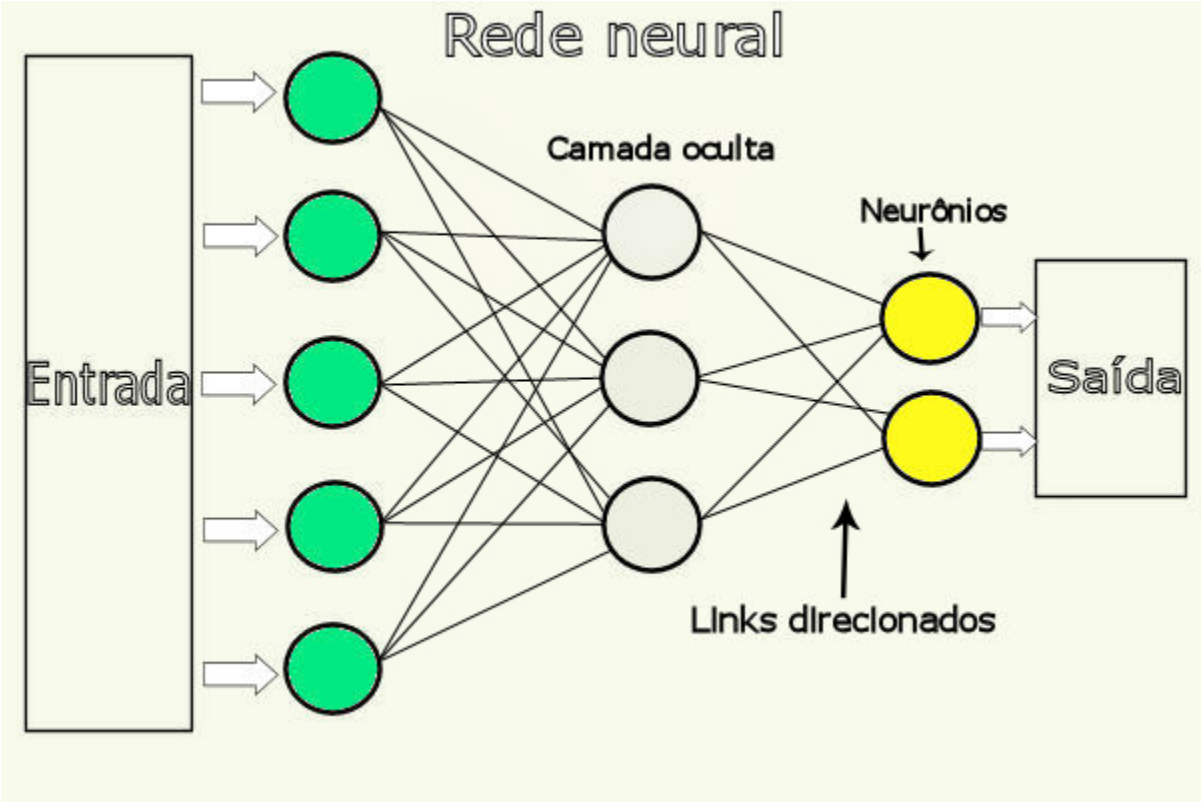
Essa capacidade de aprendizado diminui a necessidade de um conhecimento aprofundado sobre o problema ou sua solução. Uma RNA não precisa saber a origem de seus dados de entrada. Ela cria um modelo interno, que emerge do treinamento, e que é capaz de perceber os padrões dos dados, e consequentemente, fornecer uma boa resposta.

Um dos maiores desafios de projetar uma rede neural artificial é escolher uma topologia adequada ao problema.

3.2.1 Topologia da rede

Não existem ainda regras bem definidas sobre como deve ser a topologia de uma RNA, apenas diretivas. Uma topologia bastante usada (figura 1) consiste em 3 camadas de neurônios, cada qual totalmente conectada com a adjacente, através de ligações direcionadas para o sentido da camada de saída. Esse fluxo de informação em sentido único - da camada de entrada para a camada de saída - caracteriza uma RNA *feedforward*.

Figura 1 - Topologia de uma rede neural artificial tipo *feedforward*



verde - camada de entrada ; amarelo - camada de saída - Fonte:

*(*http://www.dsc.ufcg.edu.br/~pet/jornal/setembro2011/materias/informatica.html, s.d.)

Toda camada que não seja a camada que recebe os dados (camada de entrada) ou a camada que fornece a saída (camada de saída) é conhecida como camada oculta, escondida ou intermediária. O número de camadas ocultas interfere drasticamente na capacidade da rede: Sem camada oculta, a RNA consegue apenas classificar padrões linearmente separáveis; Com uma camada oculta, a RNA é capaz de aproximar qualquer função contínua; Com duas camadas, a RNA é capaz de aproximar qualquer função descontínua. O número de neurônios da camada oculta também interfere fortemente na rede: Com poucos neurônios na camada oculta, a RNA possui baixa capacidade de generalização (*underfitting*); Com muitos neurônios na camada oculta, a RNA pode ter um treinamento muito longo, e sofrer *overffiting*.

3.2.2 Treinamento

Treinamento é o processo pelo qual se dá a aprendizagem. Existem três paradigmas de aprendizado mais notórios: Treino por reforço; Treino não-supervisionado; E o utilizado neste presente trabalho, treino supervisionado [4,5].

No treino supervisionado, é apresentado à RNA um conjunto de dados de entrada, assim como a saída resultante ideal. Dessa forma o algoritmo de treinamento é capaz de inferir de que forma deve modificar os pesos dos neurônios para diminuir o erro entre a resposta atual da RNA e a resposta perfeita para o conjunto de treino dado. Normalmente é utilizado o método do gradiente descendente nessa diferença entre saída resultante e saída ideal, afim de que esse erro seja minimizado.

Existem inúmeros algoritmos de treinamento, cada qual com suas características. Nesse trabalho foi utilizado o Propagação Resiliente.

3.2.2.1 Propagação Resiliente

Propagação Resiliente [6,7] é uma heurística de aprendizado criada por Martin Riedmiller e Heinrich Braun. Além de ser um dos algoritmos de treino propagante mais

rápidos, outra vantagem é que ele não necessariamente necessita de parâmetros para ser utilizado, facilitando muito a sua aplicação.

No Propagação Resiliente, a magnitude do gradiente não é utilizada, apenas o seu sinal. É criado um *valor de atualização* - inicialmente bem pequeno - para cada um dos pesos da RNA. Para cada peso, se o sinal do seu gradiente mudou desde a última iteração, o seu *valor de atualização* é multiplicado por η^-, sendo $0 < \eta^- < 1$. Se o sinal do seu gradiente permaneceu igual, então seu *valor de atualização* é multiplicado por η^+, sendo $\eta^+ > 1$. A seguir, os pesos são modificados pelos seus valores de atualização na direção oposta à do gradiente, afim de minimizar o erro (gradiente descendente). Normalmente η^+ é empíricamente definido em 1.2, e η^- é definido em 0.5.

3 ALGORITMOS GENÉTICOS

3.1 Princípios e funcionamento

Um algoritmo genético (AG) é uma heurística utilizada em problemas de otimização e de busca para encontrar soluções em um espaço de possibilidades. O seu funcionamento é inspirado na Teoria da Evolução de Darwin.

Inicialmente o AG cria uma população aleatória de indivíduos. Cada indivíduo é um vetor de números ou caracteres (representando um genoma), e representa uma possível solução para o problema em questão. A cada geração (iteração), os indivíduos são avaliados, selecionados, e cruzados.

A avaliação se dá através de uma função de avaliação. Ela indica o quão bem adaptados (quão boa solução os indivíduos representam) os indivíduos são ao ambiente (problema). A função de avaliação é um dos pontos centrais de um AG, e é o que o adapta a cada problema. Os valores de adaptação também são utilizados como critério para a seleção de indivíduos.

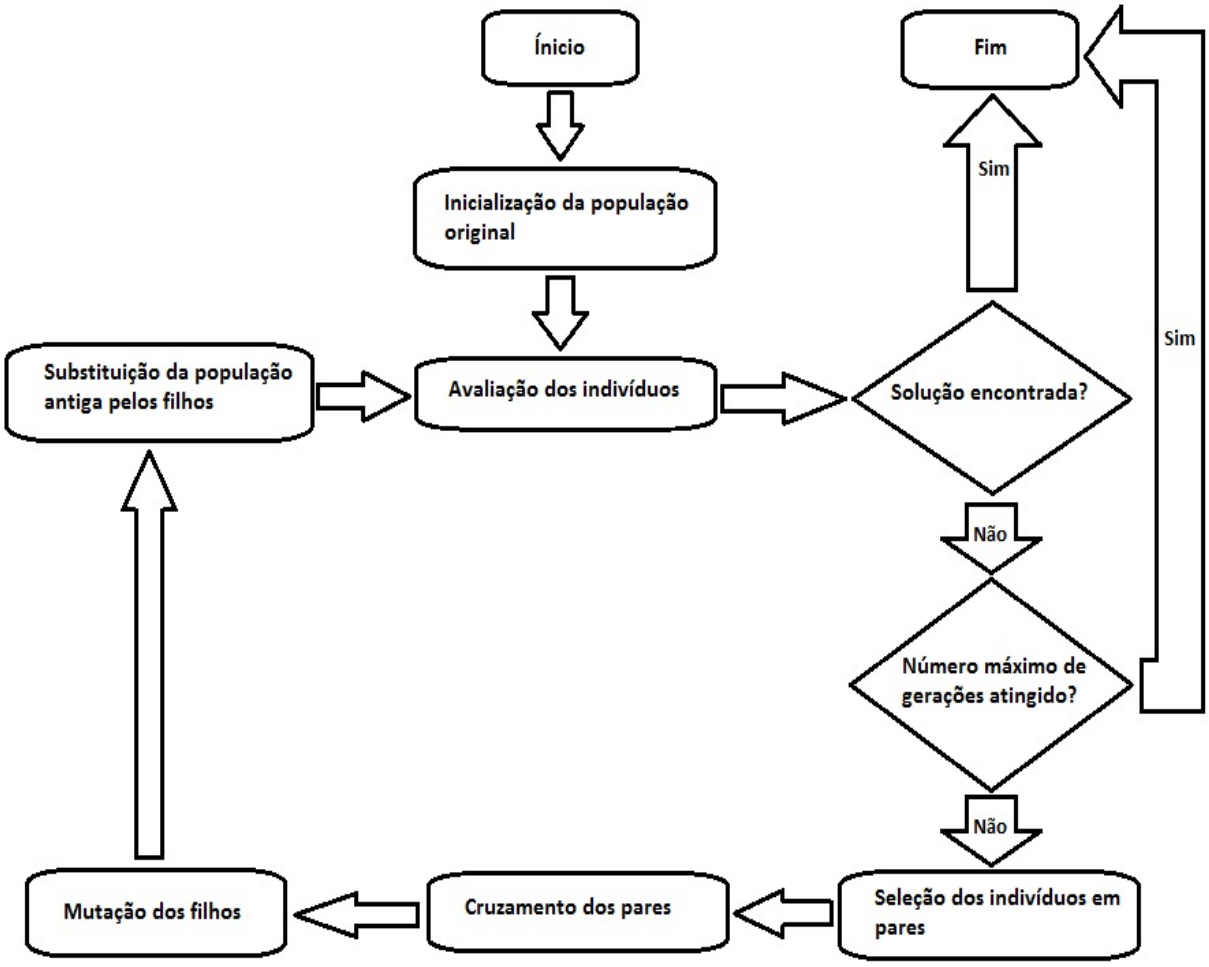
Na próxima etapa, um algoritmo de seleção é utilizado para agrupar alguns indivíduos em pares. Existem diversos algoritmos para esse fim, sendo o Método da Roleta um dos mais utilizados. Nesse método, os indivíduos são selecionados de forma aleatória, com a probabilidade de serem escolhidos sendo proporcional ao seu nível de adaptação.

Na etapa de cruzamento, um algoritmo de cruzamento é utilizado para misturar os vetores dos pares de indivíduos (casais), gerando novos indivíduos (filhos) que irão possuir nos seus vetores padrões (características) dos dois pais. Esses padrões podem torná-los mais adaptados ou menos. Durante sua criação, esses filhos estão sujeitos a mutações, que são possíveis (determinadas por uma probabilidade) mudanças aleatórias nos seus vetores. As mutações são capazes de inserir novas características em uma população, aumentando o espaço de busca do algoritmo.

Esses novos indivíduos então substituem a população antiga e uma nova geração começa.

Uma das dificuldades de utilizar um algoritmo genético é a quantidade de parâmetros. Como escolher o número de gerações? O tamanho da população? A probabilidade de mutação? De forma similar às redes neurais, não existem regras para se efetuar boas escolhas.

Figura 2 - Fluxograma de um Algoritmo Genético



Fonte:(Autor, 2012)

4 IMPLEMENTAÇÃO

4.1 Visão Geral

A funcionalidade da implementação consiste em uma aplicação na qual se carrega uma série temporal e, como resultado, retorna:

- Uma predição dos próximos valores da série temporal;

- A rede neural que possui a melhor topologia encontrada para reconhecer a série;

- O *Root-Mean-Square Error* (RMSE) entre as previsões da melhor RNA e os valores futuros reais (ver capítulo 6).

Para tanto, a implementação utiliza um algoritmo genético que gera uma população de redes neurais de diferentes topologias, e as treina na série temporal escolhida.

A interação com o programa ocorre através de uma interface gráfica, que foi criada para facilitar a execução das experiências e verificação dos resultados.

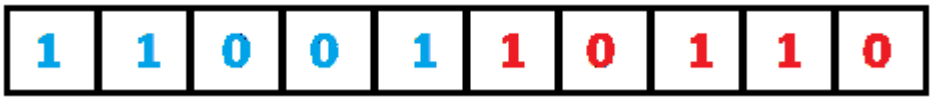
4.2 Detalhamento

A implementação é na verdade um algoritmo genético (AG), cujos indivíduos são vetores binários de 10 posições. O AG lê seus parâmetros de um arquivo de texto (arquivo de propriedades), que é parcialmente alterado pela interface a cada execução (parâmetros variáveis), enquanto o resto do arquivo é constante. Os parâmetros constantes representam escolhas arbitrárias que definem o AG da seguinte forma: O cruzamento utilizado é o de 1 ponto; O método de seleção é Roleta; O número de retentativas de geração de indivíduos aleatórios na população original para impedir duplicatas é 1000.

Após iniciada a execução o AG funciona normalmente, com o único ponto de destaque sendo o cálculo do *fitness* dos indivíduos. Para esse cálculo, o algoritmo interpreta as 5 primeiras posições do genoma do indivíduo como um número binário

que é a diferença entre a quantidade mínima e máxima de nodos na camada de entrada de uma rede neural, enquanto as outras 5 posições seriam a diferença entre o número mínimo e máximo de nodos na camada oculta (figura 3).

Figura 3 - Genoma de um indivíduo



5 primeiras posições (em azul) representam o número 25; 5 outras posições (em vermelho) representam o número 22 - Fonte:(Autor, 2012)

Esses valores, assim como outros parâmetros escolhidos na interface, são utilizados para gerar e treinar uma RNA, e assim obter o RMSE entre o modelo interno da série temporal na RNA e a série temporal real.

O RMSE (também conhecido como RMSD - *Root-Mean-Square Deviation)* [8] é um dos muitos métodos estatísticos para avaliação da diferença entre valores estimados e os valores reais correspondentes. Dados dois vetores $\boldsymbol{\theta}_1$ (valores previstos) e $\boldsymbol{\theta}_2$ (valores observados) ele calcula o seu erro da seguinte forma:

$$\theta_1 = \begin{bmatrix} x_{1,1} \\ x_{1,2} \\ \vdots \\ x_{1,n} \end{bmatrix} \qquad \text{and} \qquad \theta_2 = \begin{bmatrix} x_{2,1} \\ x_{2,2} \\ \vdots \\ x_{2,n} \end{bmatrix}.$$

$$\mathrm{RMSD}(\theta_1, \theta_2) = \sqrt{\mathrm{MSE}(\theta_1, \theta_2)} = \sqrt{\mathrm{E}((\theta_1 - \theta_2)^2)} = \sqrt{\frac{\sum_{i=1}^{n} (x_{1,i} - x_{2,i})^2}{n}}.$$

A partir do cálculo do RMSE, obtém-se o *fitness* da seguinte forma:

$$fitness = \frac{1}{RMSE}$$

O tamanho do genoma do indivíduo e a forma como ele é subdividido foram escolhas arbitrárias baseadas apenas no julgamento do autor do presente trabalho de qual seria uma margem razoável entre o menor e o maior valor. O número mínimo de nodos em cada camada é definido pelo usuário através da interface, o que indiretamente define o número máximo de nodos.

Para prever valores futuros, o algoritmo prevê apenas o próximo valor, e então o utiliza (junto com os valores anteriores) para prever o próximo, deslizando a janela de previsão apenas uma posição. Esse procedimento se repete até que tenham sido previstos o número de valores desejados. É interessante salientar que o RMSE utilizado para avaliar o erro no treinamento da rede é diferente do RMSE utilizado na validação. O RMSE utilizado na validação calcula o erro entre a predição da rede dos valores futuros e os reais valores futuros.

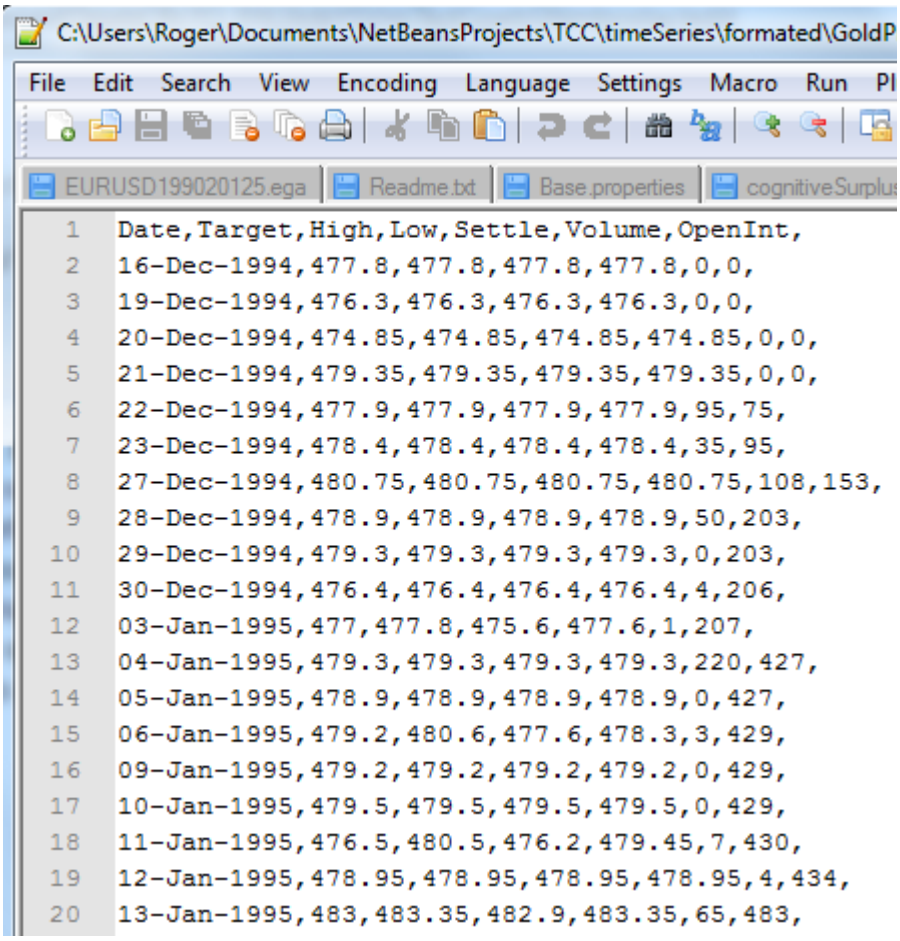
As séries temporais são carregadas através de arquivos no formato *Comma-separated values* (CSV). Afim de lidar com os arquivos de forma organizada, um sistema de pastas foi criado. As séries temporais ficam na pasta ./timeSeries/formated/, e os seus valores futuros ficam em ./timeSeries/forecasts/.

4.3 CSV

CSV [9] é um formato de arquivo de texto bastante utilizado para a representação de séries temporais. Existem inúmeros sites que fornecem grandes quantidades de séries temporais em formato CSV na internet.

Um arquivo CSV é na verdade uma tabela, que pode possuir ou não rótulos para as suas linhas e colunas, e cujos valores de cada coluna são separados por alguma pontuação (normalmente vírgula).

Figura 4 - Arquivo CSV



Exemplo de uma série temporal em um arquivo CSV. Fonte: (Autor, 2012)

Na figura 4, a primeira linha corresponde aos rótulos de cada coluna, enquanto cada uma das outras linhas são valores (correspondentes às colunas) separados por vírgulas. Os pontos são pontos decimais. Para que a implementação funcione é necessário que as colunas tenham rótulos, e caso as linhas tenham rótulos, a coluna dos rótulos de linhas deve ter um rótulo também.

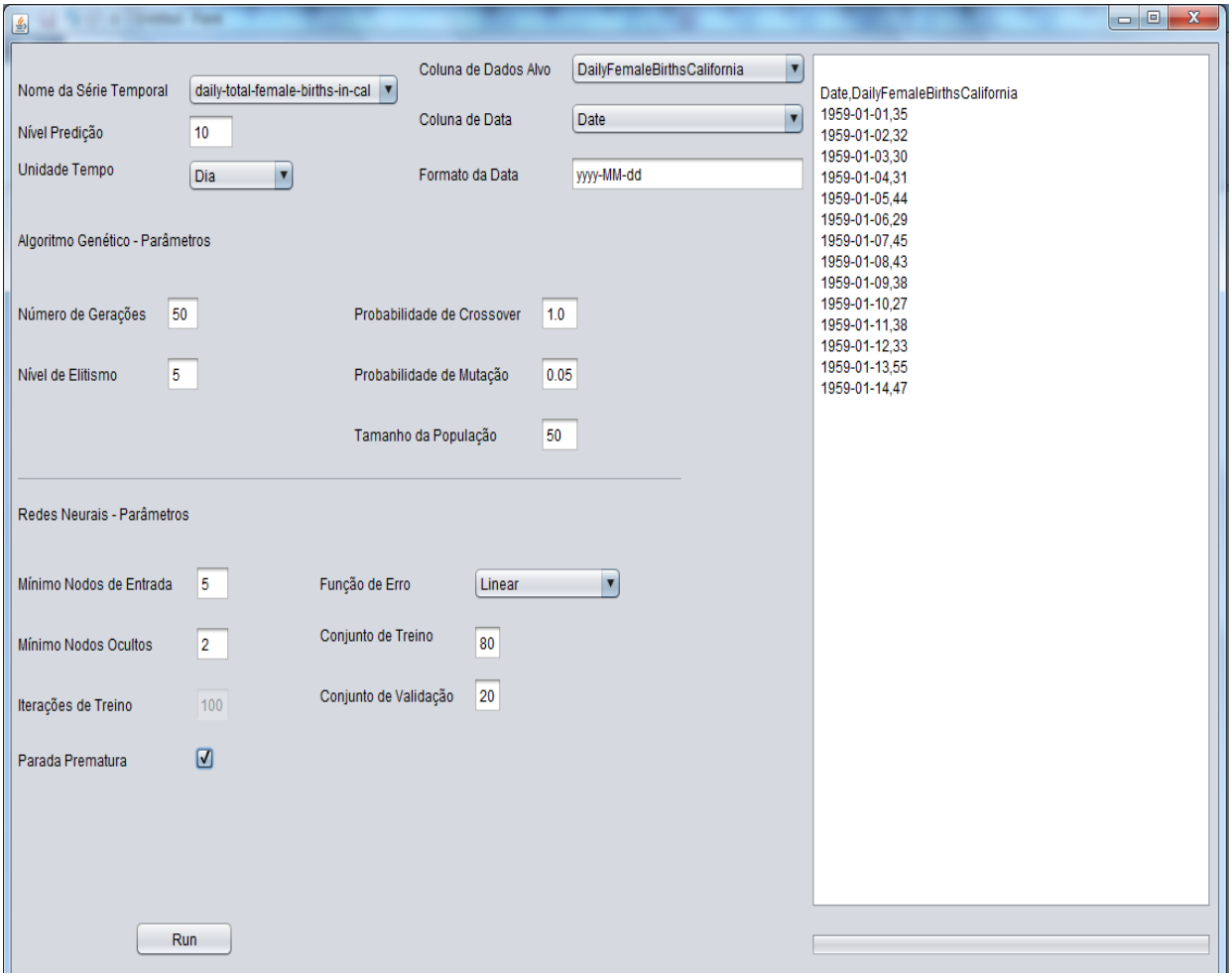
4.4 Interface

A implementação possui uma simples interface construída no Swing [10], que é uma API para design de interfaces gráficas em Java. O objetivo é simplesmente facilitar

a interação com o usuário, melhorar a visualização dos resultados e aumentar o nível de robustez.

4.4.1 Entrada

Figura 5 - Interface gráfica



Fonte:(Autor, 2012)

A seguir segue uma descrição de cada campo dos parâmetros de entrada:

Nome da série temporal

É aqui que se escolhe com qual série temporal se vai trabalhar. O nome é o mesmo do arquivo CVS da série temporal. A escolha se dá através de uma caixa de seleção.

Nível Predição

O número de valores futuros da série temporal que o algoritmo tentará prever. Deve ser um inteiro maior ou igual a zero.

Unidade Tempo

Unidade de tempo utilizado entre as medidas. A escolha se dá através de uma caixa de seleção.

Coluna de Dados Alvo

É a escolha de qual das colunas do arquivo CVS será usada pela RNA para modelagem do problema. A escolha se dá através de uma caixa de seleção.

Coluna de Data

Aqui se escolhe qual das colunas do arquivo CVS é a que contém a data de cada medição da série temporal. A escolha se dá através de uma caixa de seleção.

Formato da Data

Aqui se informa em qual formato estão os dados de data na série temporal. A semântica do formato é a mesma utilizada na classe Java SimpleDateFormat (http://docs.oracle.com/javase/1.5.0/docs/api/java/text/SimpleDateFormat.html) e é passada através de uma linha de texto.

Número de Gerações

O número de gerações que o algoritmo genético irá iterar. Deve ser um inteiro maior que 0.

Nível de Elitismo

A quantidade dos melhores indivíduos que irá passar para a próxima geração. Deve ser um inteiro entre 0 e o tamanho da população.

Probabilidade de Crossover

A probabilidade de que o casal escolhido para efetuar um cruzamento irá cruzar. Deve ser um número real entre 0 e 1.

Probabilidade de Mutação

A probabilidade de mutação para cada um dos genes a cada cruzamento. Deve ser um número real entre 0 e 1.

Tamanho da população

Tamanho da população. Interfere fortemente na velocidade de execução do algoritmo. Deve ser um inteiro maior que 1 e maior que o nível de elitismo.

Mínimo Nodos de Entrada

Quantidade mínima de neurônios na camada de entrada das RNAs criadas. Deve ser um inteiro maior que 1.

Mínimo Nodos Ocultos

Quantidade mínima de neurônios na camada oculta das RNAs criadas. Deve ser um inteiro maior que 2.

Iterações de Treino

Número de iterações em que as RNAs serão treinadas. Interfere drasticamente no desempenho do algoritmo. Deve ser um inteiro maior que 1.

Parada Prematura

Se o algoritmo deve ou não usar parada prematura para escolha do momento em que o treino deve ser encerrado. A escolha se dá através de uma caixa de marcação.

Função de Erro

Escolha da função de erro a ser utilizada. A escolha se através de uma caixa de seleção.

Conjunto de Treino

A porcentagem dos valores mais antigos da série temporal que será usada como conjunto de treinamento. Deve ser um inteiro entre 0 e 100.

Conjunto de Validação

A porcentagem dos valores mais recentes da série temporal que será usada como conjunto de validação. Deve ser um inteiro entre 0 e 100.

Caixa de texto de saída

Antes da execução, ela mostra o trecho inicial do arquivo CSV da série temporal escolhida; Após a execução, ela mostra alguns outros dados (capítulo 4.4.2).

4.4.2 Saída

A saída é apresentada através da Caixa de Texto de Saída e de dois gráficos. Os dados interessantes na Caixa de Texto de Saída são: O genoma do melhor indivíduo da geração n; O *fitness* do melhor indivíduo da geração n; Os RMSEs entre a predição da melhor e da pior rede neural e os valores futuros reais; Quanto tempo demorou a execução do algoritmo.

Figura 6 - Caixa de texto de saída

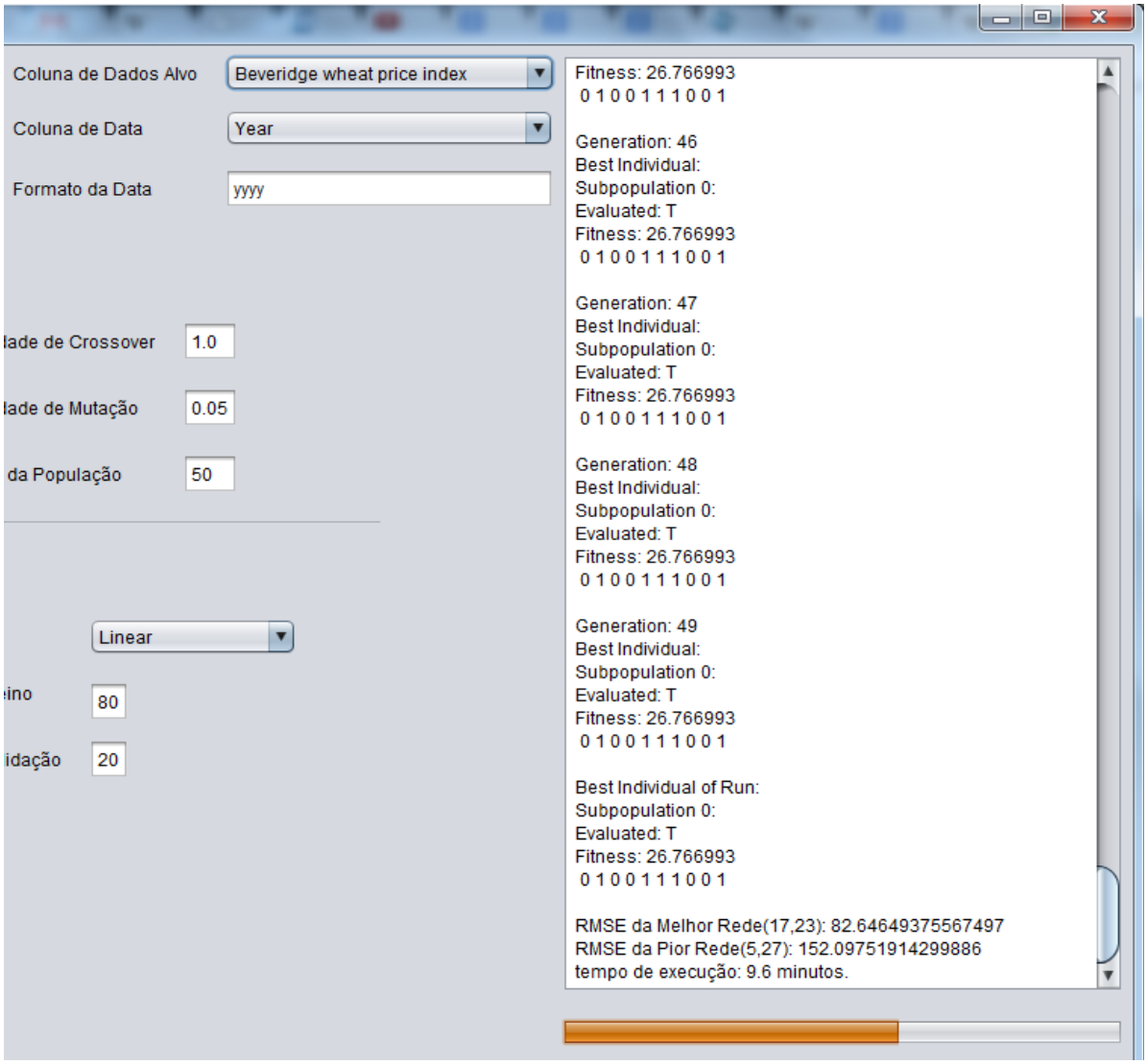
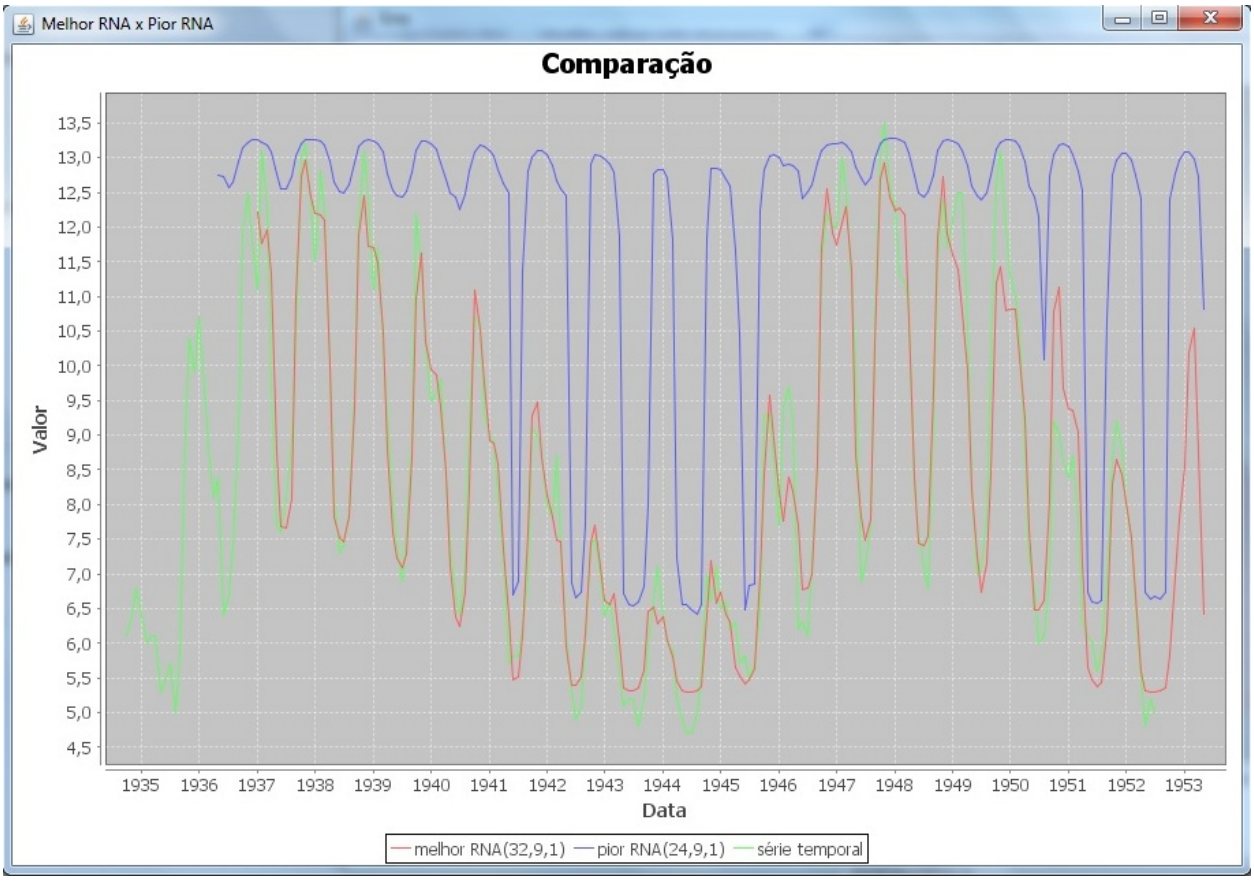


Imagem da caixa de texto de saída após a execução - Fonte:(Autor, 2012)

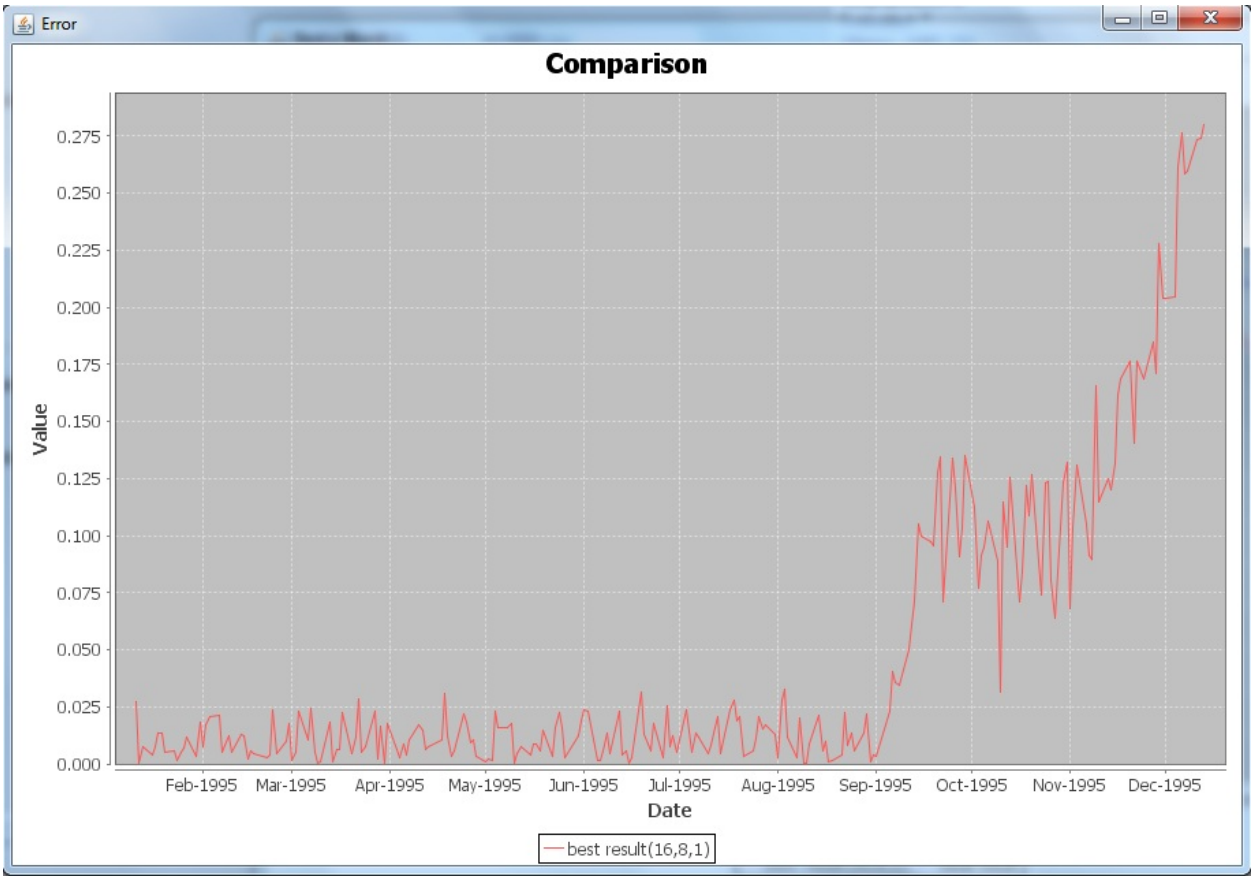
Gráfico 7 - Comparação



Comparação entre os resultados melhor, pior e ideal - Fonte:(Autor, 2012)

O gráfico da figura 7, que é gerado após a execução, mostra a comparação entre o desempenho da melhor RNA encontrada (vermelho), pior RNA encontrada (azul) e o resultado ideal (verde), além de indicar a quantidade de nós em cada camada. As RNAs vermelha e azul começam depois da série temporal (verde) pois elas precisam de um determinado número de medições passadas para começar a prever as próximas (nesse caso, 32 e 24 medições respectivamente). De forma semalhante, a série temporal (verde) acaba antes, pois as RNAs verde e vermelha estão prevendo os próximos 10 valores.

Gráfico 8 - Erro



Erro da previsão da RNA - Fonte:(Autor, 2012)

O gráfico da figura 8 simplesmente mostra o erro da melhor RNA encontrada em relação à série temporal. Como ele é gerado a partir de dados normalizados, o erro máximo é 1.

4.5 Frameworks

4.5.1 ECJ

O ECJ [11,12] é um framework gratuito e open-source em Java para computação evolucionária criado por Sean Luke, um professor de ciência da computação da universidade George Mason. O ECJ oferece suporte para algoritmos

genéticos, programação genética, estratégias de evolução, coevolução, otimização de enxame de partículas, e evolução diferencial.

4.5.2 Encog

O Encog [6] é um framework para aprendizado de máquina. Ele é um dos dois melhores frameworks gratuitos em Java para criação de redes neurais, além de possuir a sua própria wiki e comunidade para oferecer suporte. Outra característica positiva é que ele faz um ótimo uso de paralelismo, o que o faz ter um ótimo desempenho.

5 EXPERIMENTOS

As séries temporais utilizadas para a validação do trabalho foram extraídas de http://datamarket.com/data/list/?q=provider:tsdl%20cat_id:18655 [13]. Uma pequena reformatação de seus arquivos csv foi necessária em alguns casos. Os últimos 20 valores de cada série temporal foram separados em outro arquivo e foram tratados como dados futuros, para avaliação da eficiência da melhor RNA encontrada. Os exemplos utilizados para análise foram: “Monthly critical radio frequencies in Washington, D.C., May 1934 – April 1954. These frequencies reflect the highest radio frequency that can be used for broadcasting” e “Monthly U.S. Male (16-19 years) unemployment figures (thousands) 1948-1981”. Esses arquivos estão disponíveis em versão original na pasta ./timeSeries/backupCSV/. Versões modificadas (sem os últimos 20 valores) são utilizados no algoritmo.

Em todos os testes os parâmetros utilizados (escolhidos arbitrariamente) foram:

- Previsão dos próximos 10 valores da série temporal;
- O algoritmo genético funcionou por 50 gerações;
- A população do algoritmo genético era de 50 indivíduos;
- A probabilidade de crossover era de 100%;
- A probabilidade de mutação era de 5% - cada gene em cada cruzamento;
- O nível de elitismo era 5;
- O número mínimo de nodos na camada de entrada era 5. Ou seja, nos testes as RNAs possuíam entre 5 e 37 nodos nessa camada;
- O número mínimo de nodos na camada oculta era 2. Ou seja, nos testes as RNAs possuíam entre 2 e 34 nodos nessa camada;
- O fim do treinamento ocorria por parada prematura;
- A função de erro das redes neurais era linear;
- O conjunto de treino era 80% da série temporal, enquanto o conjunto de validação era 20%.

A testagem foi dividida em duas partes: A primeira parte tem por fim demonstrar o impacto que o número de neurônios em cada camada tem no desempenho das

RNAs; A segunda parte pretende validar o desempenho deste trabalho comparando-o com outros métodos.

Na primeira parte da testagem, os pesos das RNAs foram inicializados com a mesma *random seed*, eliminando o seu aspecto aleatório. Isso fez com que a distribuição dos neurônios fosse o único fator variável que determinasse o desempenho das inúmeras RNAs geradas pelo algoritmo. Ou seja, eram idênticas todas as RNAs que possuíam o mesmo número de neurônios em cada uma das camadas. Nessa testagem, são apresentadas diversas comparações entre a melhor e a pior RNA encontrada em cada execução. O fator aleatório da geração da população inicial do algoritmo genético foi mantido, pois assim pôde-se demonstrar a convergência da população para valores semelhantes (tanto de melhor quanto de pior indivíduo). Isso asserta a eficiência do algoritmo genético como método de escolha da distribuição neuronal das RNAs.

Nas tabelas a seguir, são comparados os RMSEs da melhor e pior RNA encontradas, suas respectivas DNs <neurônios na camada de entrada – neurônios na camada oculta>, e o tempo de execução do teste.

Tabela 1 - Testes da série temporal dos preços do trigo em Beveridge 1500-1869 com horizonte de previsão 1

Número do teste	RMSE Melhor RNA	RMSE Pior RNA	DN Melhor RNA	DN Pior RNA	Tempo (minutos)
1	54,0778645189	84,245204919	17 - 31	9 - 32	3,3
2	58,6835124119	85,0075300346	17 - 23	9 - 31	6,3
3	56,3388041986	85,0075300346	19 - 17	9 - 31	7,1
4	58,6835124119	84,245204919	17 - 23	9 - 32	6,2
5	58,6835124119	85,0075300346	17 - 23	9 - 31	5,3
6	65,4422156622	83,8162548432	19 - 25	10 - 30	8,1
7	58,6835124119	85,0075300346	17 - 23	9 - 31	3,2
8	56,3388041986	82,7569651627	19 - 17	10 - 29	9,6
9	65,4422156622	85,0075300346	19 - 25	9 - 31	9,5
10	58,6835124119	84,245204919	17 - 23	9 - 32	10,4

Tabela 2 - Testes da série temporal dos preços do trigo em Beveridge horizonte de previsão 10

Número do teste	RMSE Melhor RNA	RMSE Pior RNA	DN Melhor RNA	DN Pior RNA	Tempo (minutos)
1	68,8708631213	148,7381211106	17 - 23	10 - 30	7,4
2	64,2604937669	149,4323245582	22 - 31	10 - 28	7,3
3	67,0198671833	149,4323245582	20 - 21	10 - 28	8,5
4	69,7280535192	149,389059125	19 - 25	9 - 31	7,3
5	68,8708631213	149,389059125	17 - 23	9 - 31	7,3
6	68,8708631213	149,389059125	17 - 23	9 - 31	8,2
7	68,1872226659	148,7381211106	19 - 17	10 - 30	6,7
8	69,7280535192	144,768369237	19 - 25	9 - 32	7,8
9	68,5766517589	144,768369237	17 - 24	9 - 32	10,8
10	68,0161848264	143,4725500081	17 - 12	5 - 27	5,5

Tabela 3 - Testes da série temporal dos preços do trigo em Beveridge com horizonte de previsão 20

Número do teste	RMSE Melhor RNA	RMSE Pior RNA	DN Melhor RNA	DN Pior RNA	Tempo (minutos)
1	66,4928135302	140,9680435739	17 - 23	9 - 31	8,9
2	66,4928135302	140,6579643052	17 - 23	10 - 28	7,7
3	66,4928135302	136,3199536623	17 - 23	9 - 32	9,1
4	67,0508363505	136,3199536623	17 - 24	9 - 32	7,8
5	65,4358905328	139,0399374707	19 - 17	10 - 30	7
6	67,6893553678	136,5874958303	18 - 10	11 - 27	6,9
7	65,4358905328	138,5228294996	19 - 17	5 - 27	7,1
8	65,4358905328	136,5874958303	19 - 17	11 - 27	3,1
9	65,4358905328	140,9680435739	19 - 17	9 - 31	8,7
10	66,4928135302	140,9680435739	17 - 23	9 - 31	8,4

Como pode ser observado nas tabelas, a escolha errada da distribuição neuronal pode pode mais do que dobrar o valor de RMSE.

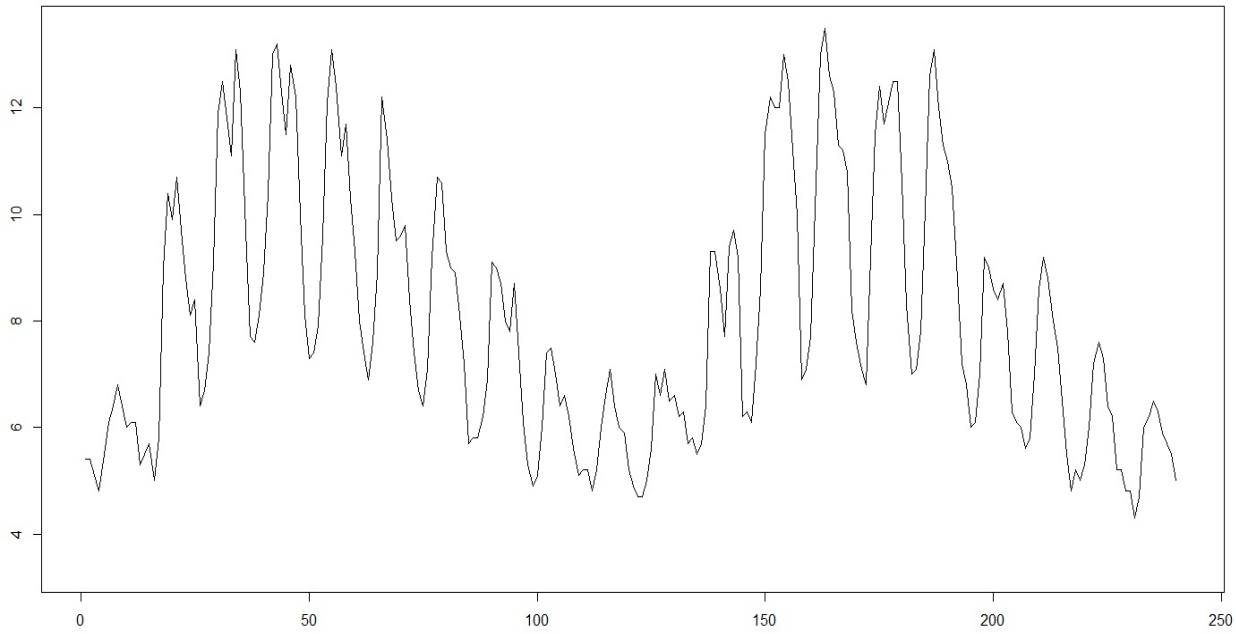
Para a validação dos resultados deste trabalho, eles foram comparados com o *Autoregressive Integrated Moving Average* (ARIMA) [14,15], que é um método estatístico comumente considerado o melhor método para predição de séries temporais existente. Para a sua aplicação, foi utilizada uma ferramenta para computação estatística - o R [16]. O R já possui todas as funções necessárias dos testes, bastando alguns comandos para obter os resultados. Nesse conjunto de testes os pesos das

RNAs são inicializados de forma aleatória, ou seja, RNAs com a mesma distribuição de neurônios provavelmente serão diferentes. Isso diminui a possibilidade que o algoritmo fique preso em mínimos locais.

A métrica usada na comparação foi o RMSE, que é um método frequentemente utilizado para medir a diferença entre valores previstos e valores observados (vide capítulo 4, seção 2).

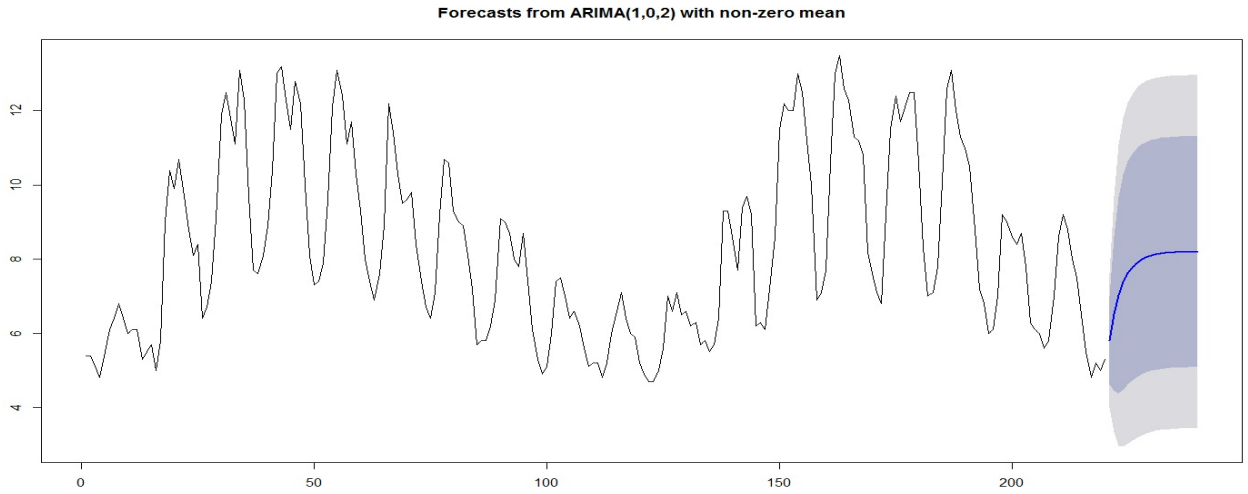
Primeiramente, foi usada uma série temporal que mostra as frequências de rádio críticas em Washington, EUA. A representação gráfica dela pode ser vista na figura 9, enquanto que a tentativa de previsão do ARIMA e da presente implementação podem ser vistas nas figuras 10 e 11 respectivamente.

Figura 9 - Série temporal das frequências críticas em Washington 1934-1954



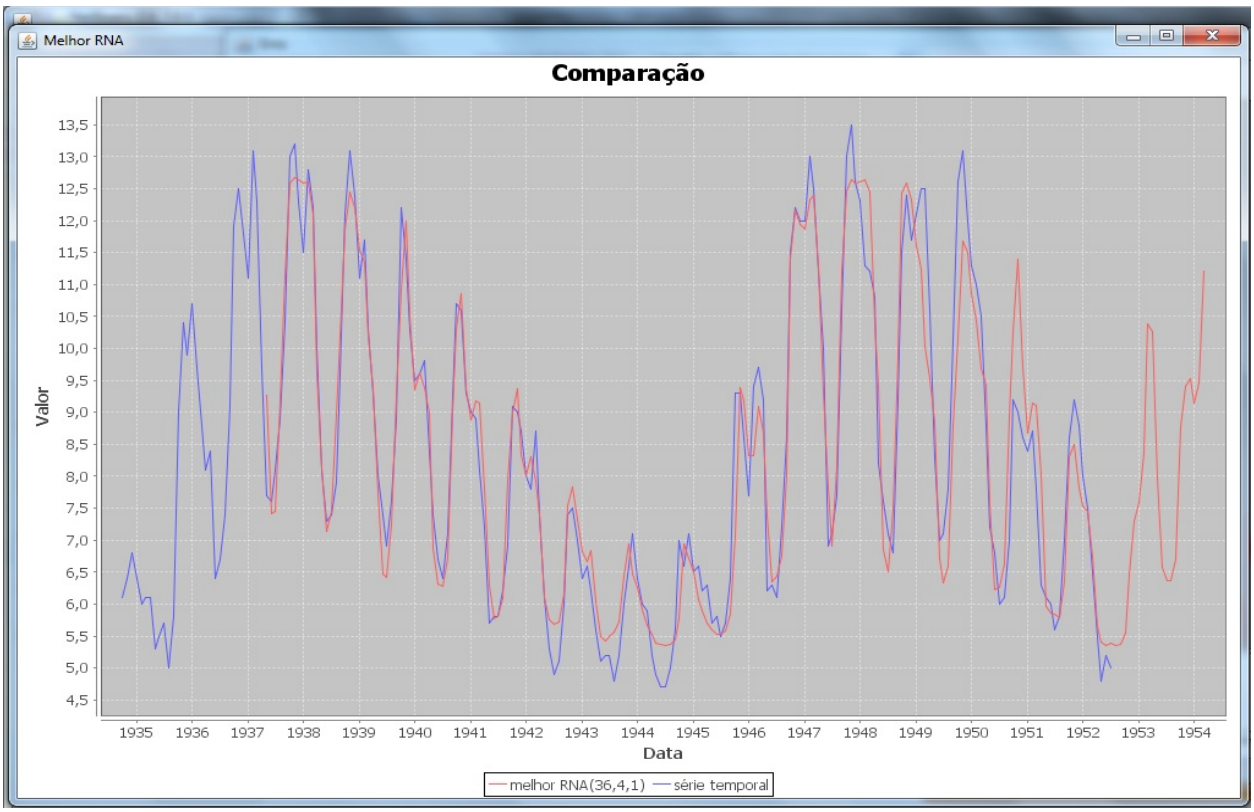
Série temporal completa. Fonte:(Autor, 2012)

Figura 10 - Previsão pelo ARIMA da série temporal das frequências de rádio críticas



Série temporal parcial, com tentativa de previsão dos últimos 20 valores pelo ARIMA. Fonte:(Autor, 2012)

Figura 11 - Previsão pelo algoritmo da série temporal das frequências de rádio críticas



Série temporal parcial (azul) e a previsão feita pela RNA (vermelho) dos últimos 20 valores. Fonte:(Autor, 2012)

Em todos os testes, a previsão da RNA possuia uma forma semelhante, que como pode se observar nos gráficos é bem mais “orgânica”. A constatação empírica é de que a RNA possui uma capacidade maior em reconhecer padrões e extrapolá-los. Infelizmente isso não foi suficiente, pois nesse exemplo, apesar dos resultados terem sido comparáveis, o RMSE da previsão do ARIMA foi melhor.

Tabela 4 – Testes da série temporal das frequências críticas de rádio com horizonte de previsão 1

Número do teste	RMSE Melhor RNA	RMSE Pior RNA	DN Melhor RNA	DN Pior RNA	Tempo (minutos)
1	0,7752078617	7,233179628	36 - 5	8 - 8	1,8
2	1,0185008806	6,3444164776	34 - 29	9 - 10	2,7
3	0,9110088403	6,7973904654	34 - 7	9 - 25	3,3
4	0,9537826884	7,5	36 - 9	7 - 17	3,4
5	1,1101706879	0,1814427194	23 - 4	7 - 29	2,2
6	0,804947315	0,3612756252	23 - 8	7 - 6	3,9
7	1,0052197573	1,6452358442	23 - 7	32 - 13	3
8	0,9707952429	7,5	36 - 27	5 - 18	2,9
9	0,9881275348	5,8162428287	36 - 15	5 - 19	4,3
10	0,821153862	0,8805558024	36 - 13	6 - 17	2,6

Tabela 5 – Testes da série temporal das frequências críticas de rádio com horizonte de previsão 5

Número do teste	RMSE Melhor RNA	RMSE Pior RNA	DN Melhor RNA	DN Pior RNA	Tempo (minutos)
1	1,6601744068	3,6593216995	36 - 7	24 - 31	1,9
2	1,458920999	6,6272166072	35 - 12	8 - 5	3,3
3	1,4833583338	6,5768507084	23 - 9	7 - 17	2,2
4	1,7259826687	4,6643620481	35 - 12	23 - 9	4
5	1,5557248925	6,4581552311	36 - 14	27 - 11	3,2
6	1,5038491742	3,6890213568	32 - 4	33 - 7	2,6
7	1,5072356215	4,4861873523	36 - 14	23 - 24	2,6
8	1,5950287012	6,6057158514	23 - 6	22 - 28	1,8
9	1,4010369723	6,4876588431	35 - 15	8 - 9	1,4
10	1,5343736647	4,8467755314	35 - 24	7 - 9	3,1

Tabela 6 – Testes da série temporal das frequências críticas - horizonte de previsão 10

Número do teste	RMSE Melhor RNA	RMSE Pior RNA	DN Melhor RNA	DN Pior RNA	Tempo (minutos)
1	2,874029145	3,6437655545	23 - 5	26 - 28	2
2	2,8383631536	6,8547091417	30 - 16	7 - 18	2,8
3	2,697364943	4,9852451107	32 - 3	7 - 15	2,8
4	3,2838426615	6,5082785355	35 - 32	32 - 9	3,1
5	3,2680572445	6,4994095519	36 - 33	5 - 21	2,6
6	3,8536416581	6,5513812404	36 - 17	11 - 4	2,5
7	2,6346532911	5,456106035	22 - 9	6 - 18	1,7
8	3,2935232837	5,4948663304	36 - 14	6 - 24	2,1
9	3,4100241538	4,4316959216	36 - 5	7 - 32	1,9
10	3,4980181544	7,495166139	36 - 11	8 - 21	2,8

Tabela 7 – Testes da série temporal das frequências críticas - horizonte de previsão 20

Número do teste	RMSE Melhor RNA	RMSE Pior RNA	DN Melhor RNA	DN Pior RNA	Tempo (minutos)
1	3,1226124975	5,6470495103	22 - 16	27 - 12	2,6
2	2,4364008208	5,2379966081	23 - 6	24 - 4	2,9
3	2,4172090798	4,9504001464	23 - 5	24 - 9	5,2
4	3,489140181	6,4053758067	33 - 4	6 - 4	3,1
5	3,6053436805	5,1969293649	35 - 6	26 - 3	3,4
6	3,5306205329	6,8163909502	36 - 16	6 - 5	3,1
7	2,8656539389	7,1778548239	31 - 4	7 - 15	2
8	2,8011347123	4,9800945288	32 - 6	24 - 8	2,1
9	3,477055582	7,7066116517	35 - 28	8 - 13	4
10	3,5192266883	7,4026112299	33 - 32	29 - 10	3,8

Tabela 8 – RMSE entre a previsão do ARIMA e a série temporal das frequências críticas

Horizonte de previsão	RMSE ARIMA
1	0,19172
5	0,6814559
10	2,040376
20	2,369644

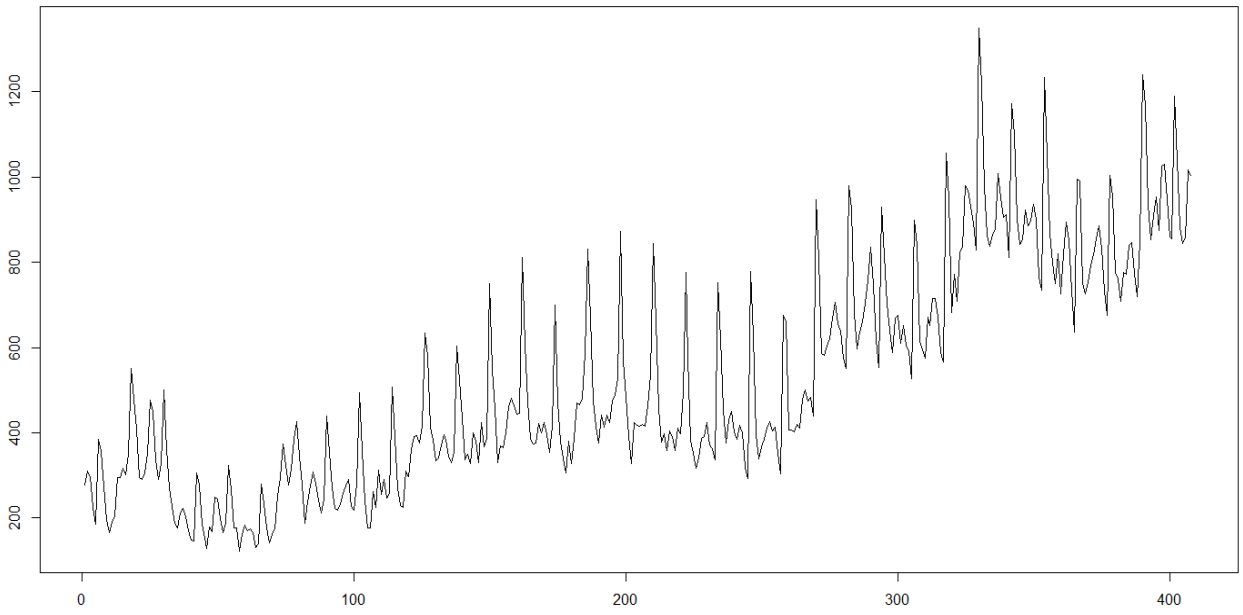
Como pode ser visto nas tabelas, a implementação não obteve melhor desempenho em nenhum dos casos dessa série de testes, apesar de ter se aproximado várias vezes do resultado apresentado pelo ARIMA.

É interessante ressaltar que em alguns casos pode ocorrer uma anomalia – A pior RNA encontrada possuir um RMSE menor que a melhor RNA encontrada. Dois exemplos são os testes 5 e 6 da tabela 4. Isso pode ocorrer quando a melhor RNA não

consegue aprender adequadamente o padrão oculto na série temporal. Isso pode ocorrer por excesso de ruído aleatório na série temporal, o que é impossível de ser previsto pela RNA, ou caso a janela temporal (número de passos que a RNA olha no passado para prever o futuro) seja muito pequena. Nesses casos pode ocorrer da pior RNA eventualmente oferecer uma previsão com menor erro do que a melhor RNA. Quanto maior o horizonte de previsão, menor a chance de que isso ocorra, já que um maior número de palpites da pior RNA teriam que ser mais próximos do ideal do que os da melhor RNA, o que é improvável.

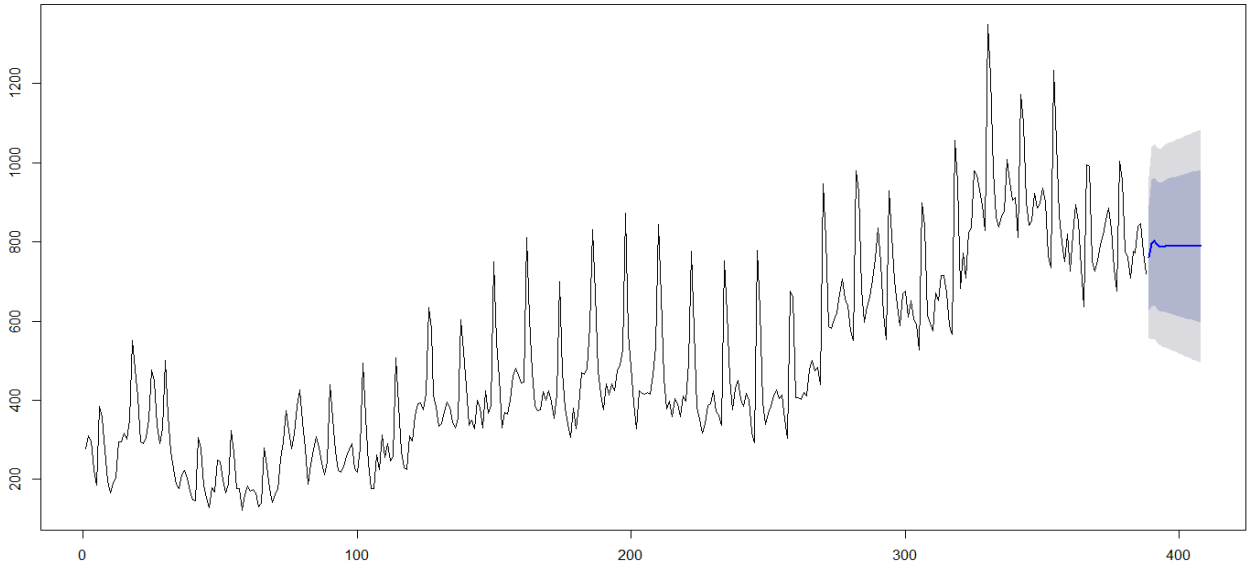
Na segunda série de testes foi utilizada uma série temporal que mostra o número de homens desempregados entre 16 e 19 anos nos EUA.

Figura 12 – Série temporal de desempregados entre 16 e 19 anos nos EUA



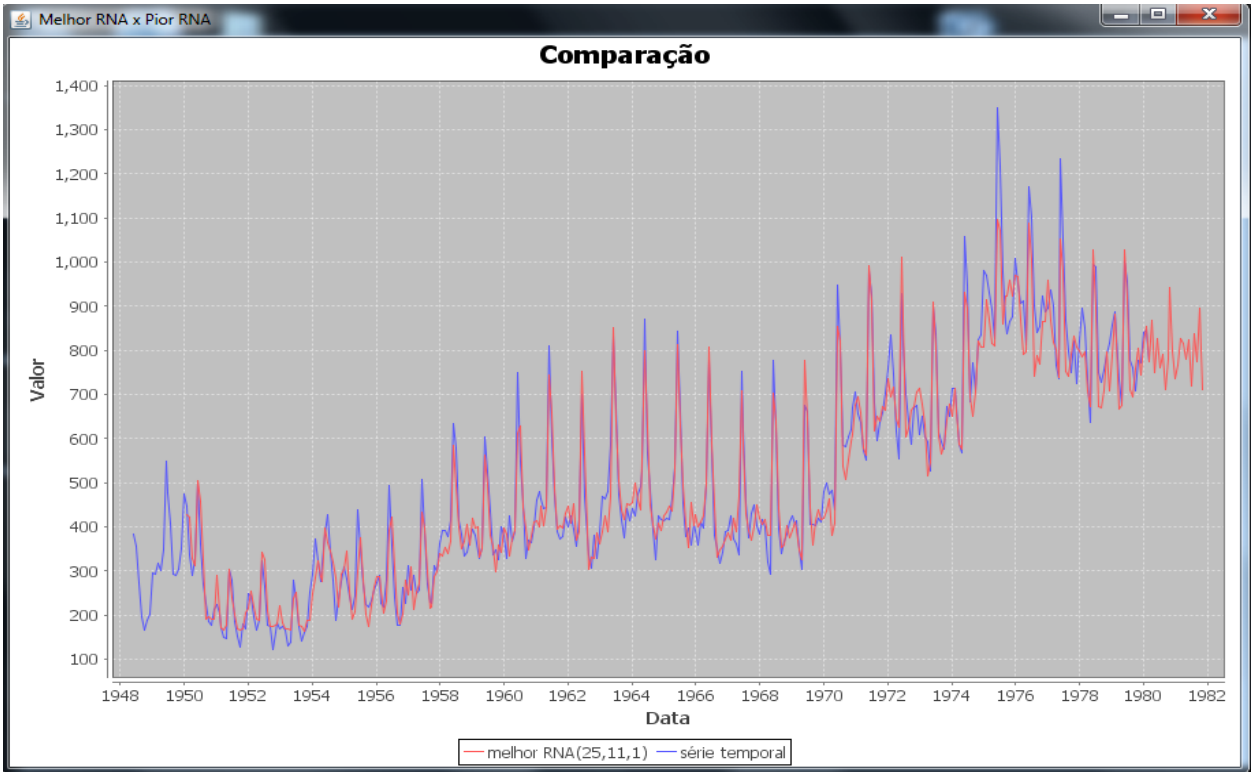
Fonte(Autor, 2012)

Figura 13 – Previsão do ARIMA da série temporal de desempregados nos EUA



Série temporal parcial, com tentativa de previsão dos últimos 20 valores pelo ARIMA (traço azul). Fonte: (Autor, 2012)

Figura 14 – Previsão pelo algoritmo da série temporal de desempregados nos EUA



Série temporal parcial (azul) e a previsão da RNA (vermelho) dos últimos 20 valores. Fonte:(Autor, 2012)

Como pode ser visto nas figuras 12 e 13, novamente a previsão do ARIMA é mais suave, pois ele calcula uma área com uma distribuição normal de probabilidades, e oferece como previsão a série de valores mais provável (mais no centro da distribuição). Comparando-se as figuras 12 e 14, pode-se perceber que a RNA tenta realmente simular o comportamento da série temporal, baseado no que foi aprendido do seu passado. Essa característica faz com que sua previsão seja mais oscilante, e mais semelhante com a série original.

A seguir as tabelas 9 a 14 mostram os RMSEs das previsões da presente implementação e do ARIMA:

Tabela 9 – Testes da série temporal dos desempregados - horizonte de previsão 1

Número do teste	RMSE Melhor RNA	RMSE Pior RNA	DN Melhor RNA	DN Pior RNA	Tempo (minutos)
1	3.8297002703659473	501.99999999107695	25 - 31	36 - 10	3.6
2	0.33894609842479895	495.3019758136661	34 - 18	18 - 5	4.2
3	30.512318836967438	501.9729104996302	32 - 11	17 - 18	4.6
4	27.23065504509748	488.7353496546848	34 - 18	30 - 11	15.6
5	13.27213146266729	480.53703068672166	35 - 31	32 - 16	18.5
6	26.2010553699472	502.0	26 - 28	20 - 11	4.4
7	7.3429335868675025	501.9999999934198	24 - 30	31 - 16	10
8	28.75865317609032	501.9988614673953	25 - 9	18 - 6	14
9	22.233844548831144	501.3077409947666	26 - 12	24 - 20	4
10	2.7915378527563917	494.21168308322376	29 - 13	19 - 16	8.5

Tabela 10 – Testes da série temporal dos desempregados - horizonte de previsão 5

Número do teste	RMSE Melhor RNA	RMSE Pior RNA	DN Melhor RNA	DN Pior RNA	Tempo (minutos)
1	254.94056174819707	378.3207629894725	31 - 13	30 - 15	8.3
2	252.31606469861285	378.32076336918885	27 - 6	35 - 14	16
3	288.20849071566505	378.3205622918873	34 - 4	25 - 21	7.7
4	291.35451041907186	378.32074286304675	36 - 16	31 - 5	10
5	264.42553834372626	378.3207633730648	19 - 22	26 - 17	9.4
6	278.019286026865	303.1341652628642	29 - 10	36 - 8	7.8
7	295.0752715661398	376.5067302214459	34 - 13	27 - 14	17.6
8	283.3692285366917	377.958554657987	34 - 10	31 - 18	8.3
9	285.9933449227156	378.32076337415054	26 - 7	35 - 28	8.6
10	288.5597532865014	376.62525927444744	27 - 23	27 - 7	7.9

Tabela 11 – Testes da série temporal dos desempregados - horizonte de previsão 10

Número do teste	RMSE Melhor RNA	RMSE Pior RNA	DN Melhor RNA	DN Pior RNA	Tempo (minutos)
1	222.85053516881473	380.0534624756699	31 - 6	30 - 10	21.2
2	257.5185872821465	382.4273618921126	34 - 13	29 - 21	4
3	241.35304824828242	384.95421029719034	26 - 8	33 - 8	7.4
4	275.90106215368576	341.80429896366724	32 - 26	24 - 29	4.9
5	252.31315163616796	381.236468166541	32 - 12	19 - 12	8
6	237.20643745696194	387.39971964863224	33 - 22	31 - 15	12.4
7	230.79992508206544	386.6207148320064	25 - 20	19 - 6	10.8
8	241.14887662364976	386.8400003476762	36 - 13	31 - 11	8.7
9	261.22670600894077	374.26093066377587	34 - 14	14 - 5	5.3
10	225.04646598085318	384.33212752363886	36 - 10	18 - 4	7.4

Tabela 12 – Testes da série temporal dos desempregados - horizonte de previsão 15

Número do teste	RMSE Melhor RNA	RMSE Pior RNA	DN Melhor RNA	DN Pior RNA	Tempo (minutos)
1	237.5806371364265	369.34913636013016	28 - 32	32 - 9	4.8
2	230.56918523274987	389.1091972639062	31 - 19	35 - 8	3.8
3	233.1343834784373	389.1083576961964	24 - 28	25 - 14	6.8
4	248.9206191493983	389.1068406467025	32 - 13	30 - 27	7.3
5	269.22179717462694	362.528377537844	27 - 12	32 - 16	6.4
6	242.22896809975785	389.106802185642	20 - 10	28 - 15	16.2
7	258.8193692838988	389.1091815051352	35 - 12	25 - 30	19.9
8	241.67077351297928	374.4715338917621	36 - 19	13 - 12	9.7
9	237.83334876148774	389.0340708712547	25 - 11	24 - 24	4.8
10	249.1935868791566	709.8545808066372	34 - 19	7 - 23	3.8

Tabela 13 – Testes da série temporal dos desempregados - horizonte de previsão 20

Número do teste	RMSE Melhor RNA	RMSE Pior RNA	DN Melhor RNA	DN Pior RNA	Tempo (minutos)
1	228.8177131186185	399.2095700673625	34 - 16	17 - 7	6.3
2	212.28793828480005	401.32542903998484	26 - 9	31 - 4	6
3	218.63918738796036	401.3254284435291	36 - 23	32 - 8	16.3
4	232.89649017412262	401.32542894179085	36 - 17	14 - 5	5.3
5	232.67180785037226	401.3254288610227	36 - 33	22 - 3	5
6	198.82813331185494	387.2427633692825	25 - 11	11 - 8	9
7	234.19440415513762	401.3254071207018	27 - 15	17 - 15	13.7
8	224.90882130624024	395.8046643749194	32 - 19	16 - 3	13.2
9	234.17326361154898	401.32510138972407	20 - 14	29 - 30	4.9
10	228.81942565393544	401.32533904086364	36 - 15	19 - 16	8.2

Tabela 14 – RMSE entre a previsão do ARIMA e a série temporal de desempregados

Horizonte de Previsão	RMSE ARIMA
1	87.88532
5	269.6324
10	229.3630
15	228.3443
20	211.5603

Ao contrário da última série de testes apresentada, em diversos casos a implementação possui um melhor desempenho. No caso da tabela 9, a RNA apresentou resultados bastante superiores apenas por um acaso, causado por um horizonte de previsão muito pequeno. No restante dos testes ela demonstra possuir um desempenho comparável ao do ARIMA, mas ligeiramente inferior.

7 CONCLUSÕES

Em todos os testes realizados, as RNAs apresentaram o que empiricamente pareceu satisfatório. Elas aprenderam determinados padrões dos dados apresentados e eram capazes de identificá-los e extrapolá-los, simulando o comportamento da série temporal, mas infelizmente não com a eficiência desejada. Os motivos para o ocorrido possivelmente estejam na escolha dos parâmetros. Apesar da escolha de um deles ter sido automatizada (a distribuição neuronal), o espaço de busca para o resto ainda é muito grande. Sem uma metodologia de desempenho comprovado, é improvável que tenha sido feita a escolha ótima, logo deve haver espaço para melhora. Talvez o problema esteja na escolha das séries temporais. Possivelmente se elas tivessem sido construídas artificialmente, de posse de um maior poder sobre suas características, fosse possível uma melhor análise do desempenho do algoritmo, assim como de seus pontos fortes e fracos.

Os testes mostram que os resultados dos dois métodos são comparáveis, mas que o ARIMA é preferível. O seu tempo de execução é quase instantâneo e o seu desempenho é garantido, enquanto que o algoritmo apresentado possui um tempo de execução na ordem dos minutos e desempenho variável, podendo oferecer um resultado melhor ou pior.

Para trabalhos futuros seria interessante utilizar um algoritmo genético que além de escolher a distribuição neuronal, também escolhesse outros parâmetros das RNAs, como a organização das sinapses por exemplo, aumentando a sua flexibilidade.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] Enciclopédia Online Wikipedia. Disponível em: <http://en.wikipedia.org/wiki/Artificial_intelligence>. Acesso em: 2012.

[2] Enciclopédia Online Wikipedia. Disponível em: <http://en.wikipedia.org/wiki/Emergence>. Acesso em: 2012.

[3] Enciclopédia Online Wikipedia. Disponível em: <http://en.wikipedia.org/wiki/Self-organization>. Acesso em: 2012.

[4] Enciclopédia Online Wikipedia. Disponível em: <http://en.wikipedia.org/wiki/Artificial_neural_network>. Acesso em: 2012.

[5] The Computation and Neural Network Laboratoty. Department of Electrical & Computer Engineering Wayne State University. Website. Disponível em: <http://neuron.eng.wayne.edu/tarek/MITbook/chap4/chapt4.html>. Acesso em: 2012.

[6] Enciclopédia Online Wikipedia. Disponível em: <http://en.wikipedia.org/wiki/Resilient_propagation>. Acesso em: 2012.

[7] HEATON, Jeff. Programming Neural Networks with Encog3 in Java. Livro eletrônico. 2011.

[8] Enciclopédia Online Wikipedia. Disponível em: <http://en.wikipedia.org/wiki/Root_mean_square_deviation>. Acesso em: 2012.

[9] Enciclopédia Online Wikipedia. Disponível em: <http://en.wikipedia.org/wiki/Comma-separated_values>. Acesso em: 2012.

[10] Enciclopédia Online Wikipedia. Disponível em: <http://en.wikipedia.org/wiki/Swing_(Java)>. Acesso em: 2012.

[11] Enciclopédia Online Wikipedia. Disponível em: <http://en.wikipedia.org/wiki/Java_Evolutionary_Computation_Toolkit>. Acesso em: 2012.

[12] LUKE, Sean. The ECJ Owner's Manual. Livro eletrônico. 2010.

[14] Enciclopédia Online Wikipedia. Disponível em: <http://en.wikipedia.org/wiki/Autoregressive_integrated_moving_average>. Acesso em: 2012.

[15] Página de membro da comunidade da Fuqua School of Business at Duke University. Disponível em: <http://people.duke.edu/~rnau/411arim.htm>. Acesso em: 2012.

[16] The R Project for Statistical Computing. Website. Disponível em: http://www.r-project.org/. Acesso em: 2012.

[17] NeuroAI. Website. Disponível em: <http://www.learnartificialneuralnetworks.com/>. Acesso em: 2012.

[18] Enciclopédia Online Wikipedia. Disponível em: <http://en.wikipedia.org/wiki/Genetic_algorithm>. Acesso em: 2012.

[19] Fundamentals of Genetic Algorithm: AI Course. Material de curso online. Disponível em <http://www.myreaders.info/09_Genetic_Algorithms.pdf>. Acesso em: 2012

[20] CORTEZ, Paulo; MACHADO, José; NEVES, José. An Evolutionary Artificial Neural Network Time Series Forecasting System. Departamento de Informática, Universidade do Minho. Artigo Científico. Largo do Paço, Portugal.

[21] KAASTRA, Iebeling; BOYD, Milton. Designing a Neural Network for Forecasting Financial and Economic Time Series. Artigo Científico.1995.

[22] FARAWAY, Julian; CHATFIELD, Chris. Time Series Forecasting with Neural Networks: a Comparative Study Using the Airline Data. Artigo científico. 1997.

[23] PERALTA, Juan; XIAODONG, Li; GUTIERREZ, German; SANCHIS, Araceli. Time Series Forecasting by Evolving Artificial Neural Networks Using Genetic Algorithms and Differential Evolution. Artigo científico apresentado em WCCI 2010 IEEE World Congress on Conputational Intelligence. Barcelona, Espanha. Julho, 2010.

[24] PLUMMER, Eric. Time Series Forecasting with Feed-Forward Neural Networks: Guidelines and Limitations. Tese de Mestrado apresentada ao Department of Computer Science and The Graduate School fo The University of Wyoming. Julho, 2000.

[25] Data Market. Website. Disponível em <http://datamarket.com/data/list/?q=provider:tsdl%20cat_id:18655>. Acesso em: 2012.