ESCOLA POLITÉCNICA DE
PERNAMBUCO

Trabalho de Conclusão de Curso

Engenharia de Computação

Seleção de Variáveis: Um Sistema Híbrido Baseado em Colônia de Formigas e Rede Neural MLP

Autor: Arthur Fernandes Minduca de Sousa
Orientador: Mêuser Jorge Silva Valença

ESCOLA POLITÉCNICA DE PERNAMBUCO

Arthur Fernandes Minduca de Sousa

Seleção de Variáveis: Um Sistema Híbrido Baseado em Colônia de Formigas e Rede Neural MLP

Monografia apresentada como requisito parcial para obtenção do diploma de Bacharel em Engenharia da Computação pela Escola Politécnica de Pernambuco – Universidade de Pernambuco.

Recife, junho de 2010.

Dedico este trabalho aos meus pais, José Tadeu Fernandes de Sousa e Severina Maria Minduca, e toda minha família.

Agradecimentos

Agradeço aos meus pais, José Tadeu Fernandes de Sousa e Severina Maria Minduca pela educação e apoio.

Agradeço aos meus amigos Caio Medeiros, João Fausto Lorenzato, Diego Siqueira, Gilliard Alan, Nathalia Temudo e Rafael Galvão pelo conhecimento construído e compartilhado ao longo do curso.

Ao meu orientador Mêuser Valença, pela orientação, paciência e apoio, proporcionando os significativos resultados apresentados neste trabalho.

Resumo

As redes neurais artificiais desempenham um papel de fundamental importância em problemas de classificação e previsão na área da inteligência computacional resolvendo problemas extremamente complexos através de um paradigma de processamento inspirado no modo que o cérebro processa informações. Seu processamento começa com a leitura de uma base de dados, que é basicamente a fonte de conhecimento a ser adquirido pela rede durante a etapa de aprendizagem, que por sua vez é responsável pelo processo de aquisição do conhecimento contido na base de dados através do ajuste dos pesos sinápticos. O tamanho dessa base de dados, a quantidade de variáveis e a quantidade de exemplos afetam diretamente o tempo e a qualidade do treinamento. Além disso, uma base de dados eventualmente pode conter uma ou mais variáveis de entrada que não possuem um impacto relevante na qualidade do treinamento da rede, ou pior que isso, uma variável poderia estar diminuindo a qualidade do treinamento de uma rede neural. Neste trabalho, aplicaremos técnicas relacionadas a a*nt colony optimization* (ACO) para realizar a etapa de pré-processamento da base de dados e selecionar as melhores variáveis de entrada, de forma a conseguir o melhor balanceamento entre a quantidade de variáveis de entrada e a taxa de acerto da rede neural.

Palavras-chave: Pré-processamento de Redes Neurais, Ant Colony Optimization, Seleção de variáveis, Aprendizado de Máquina.

Abstract

Artificial neural networks play a vital role in classification and forecasting problems in the area of computational intelligence solving problems extremely complex through an inspired processing paradigm so that the brain processes information. The processing starts with the reading of a database, which is basically the source of knowledge to be acquired by the network during the stage of learning, which in turn is responsible for the acquisition of knowledge contained in the base data by adjusting the synaptic weights. The size of this database, the number of variables and number of examples directly affect the length and quality of training. In addition, a database eventually may contain one or more input variables that do not have a relevant impact on the quality of training of the network, or worse, a variable could be lowering the quality of training a neural network. In this paper we apply techniques related to ant colony optimization (ACO) to perform stage of preprocessing of preprocessing of a database and select the best input variables in order to achieve the best balance between the amount of input variables and the neural network.

Keywords: Pre-Processing Neural Networks, Ant Colony Optimization, Variable selection, Learning Machine.

Sumário

Índice de Figuras

Índice de Tabelas

Tabela de Símbolos e Siglas

ACO – Ant Colony Optimization

CFS – Correlation-based Feature Selection

PCA – Principal Component Analysis

AS – Ant System

EAS – Elitist Ant System

ACS – Ant Colony System

RNA– Rede Neural Artificial

MLP – Multilayer Perceptron

Capítulo 1
Introdução

A seleção de variáveis tem recebido atenção especial em aplicações que usam conjuntos de dados com muitos atributos, tais como: processamento de texto, recuperação de informação em banco de imagens, bioinformática, processamento de séries temporais, etc. O objetivo principal da seleção de variáveis consiste em melhorar o desempenho dos algoritmos de previsão e classificação. Por meio deste processo de seleção, eliminação de variáveis redundantes ou irrelevantes, o que se busca é simplificar os modelos de previsão e classificação de forma a reduzir o custo computacional para processar esses modelos.

Com o objetivo de encontrar este subconjunto de variáveis diversas técnicas têm sido propostas na literatura, tais como: ganho da informação (GI), a seleção baseada em correlação (CFS) [9] e a análise de componentes principais (PCA) [8], algoritmos genéticos [6], entre outras.

Neste trabalho se propõe utilizar a meta-heurística construtiva ACO [1] em conjunto com uma rede neural para redução do número das variáveis da base de dados.

1.1. Motivação e Problema

Em uma rede neural, o tempo de processamento está diretamente relacionado a quantidade atributos de entradas na base de dados, dentre outros fatores [11]. Um atributo de entrada pode ser visto como uma propriedade ou característica que agrega valor a informação processada pela rede. Entretanto, algumas dessas características podem não estar contribuindo para melhorar o desempenho durante o treinamento da rede neural, pois seus valores podem não ter um relacionamento que contribua de forma significativa com os exemplos de saída, podendo inclusive resultar em um processamento com

menores taxas de acerto. Outros fatores, como a quantidade de exemplos utilizados, qualidade dos exemplos e a função de ativação utilizada, também estão relacionados ao tempo de processamento da rede neural.

Um atributo de entrada desnecessário pode ser mais perturbador do que aparenta. Além de não classificar a informação adequadamente e aumentar a quantidade de informação na base de dados, aumentando o tempo de processamento durante o treinamento, a eliminação deste atributo poderia resultar em um treinamento mais rápido e até com desempenho significativamente melhor. O problema é que na maioria das vezes isso não é facilmente perceptível para olhos humanos, pois a informação pode não estar representada de forma intuitiva.

A solução para este problema poderia ser, por exemplo, avaliar o treinamento com todas as combinações de atributos de entrada possíveis, ou seja, realizar uma busca exaustiva. Entretanto, à medida que o número de atributos cresce, a quantidade de testes necessários iria se tornar cada vez menos plausível, pois a quantidade de permutações se tornaria bastante elevada.

Desta forma o uso de um sistema híbrido baseado em um algoritmo de busca um algoritmo de classificação/previsão é uma solução bastante interessante para este problema. Com este objetivo se utiliza neste trabalho o algoritmo de busca ACO integrado a uma rede neural MLP.

1.2. Objetivos

1.2.1. Objetivos Gerais

Aplicar algoritmos conhecidos de colônia de formigas, como o *Ant System* (AS), *Elitist Ant System* (EAS) e o *Ant Colony System* (ACS) [3] [4] para realizar o pré-processamento de uma base de dados de uma rede neural e extrair da base de dados características que não agregam valor a taxa de acerto.

Nesta metáfora do algoritmo para resolução de problemas de pré-processamento, as formigas em busca de alimento irão representar seleção de

subconjuntos de atributos da base de dados, de tal forma que no final a saída seja um subconjunto de dados que representa uma porção menor da base original, sem diferenças significativas na qualidade da aprendizagem.

1.2.2. Objetivos Específicos

- Implementar um módulo para treinamento de RNAs, incluindo pelo menos o Perceptron, Adaline e MLP.
- Implementar um módulo para pré-processamento de bases de dados com colônia de formigas, incluindo configuração de arquiteturas AS, EAS e ACS.
- Comparar resultados de pré-processamentos entre as arquiteturas de colônia de formigas.
- Comparar os resultados obtidos com os resultados encontrados em outros trabalhos.

1.3. Estrutura da Monografia

Este trabalho está dividido em 5 capítulos. O primeiro capítulo aborda em alto nível o problema a ser estudado e a necessidade de haver técnicas de seleção de atributos de entrada.

O capítulo 2 aborda o conteúdo teórico necessário para a compreensão do trabalho proposto abrangendo conceitos de computação inteligente, assim como técnicas de RNAs bastante conhecidas e utilizadas na literatura. Também será visto conceitos de algoritmos de colônias de formigas e detalhamento das técnicas mais difundidas na comunidade científica.

O capítulo 3 descreve a metodologia utilizada para o estudo do problema proposto. Nele é descrito como se processa o treinamento da rede neural e como o algoritmo de colônia de formigas seleciona as variáveis de entrada que interferem nos resultados encontrados pela rede.

No capítulo 4 são apresentados resultados dos processamentos de redes neurais realizados sobre diferentes bases de dados, variando na

arquitetura da colônia de formigas, tornando possível realizar comparações entre diferentes arquiteturas de ACO.

O capítulo 5 apresenta uma visão geral do conteúdo visto e propostas de trabalhos futuros.

Capítulo 2
Revisão Bibliográfica

Este capítulo abrange as principais técnicas de redes neurais e a*nt colony optimization* utilizadas pela comunidade científica.

2.1. Redes Neurais Artificiais (RNAs)

Redes neurais artificiais é uma técnica de inteligência computacional que realiza o processamento de dados se inspirando na forma como o cérebro humano trabalha. Nesse sentido, um cérebro humano pode ser visto como um computador bastante complexo, capaz de realizar processamentos com velocidade maior do que qualquer computador já construído.

2.1.1. Visão geral de uma RNA

As redes neurais artificiais desempenham um papel de extrema importância pra computação, resolvendo problemas extremamente complexos através de um paradigma de processamento inspirado no modo que o cérebro processa informações.

Um Rede neural, assim como o cérebro humano, toma suas decisões se baseando no conteúdo armazenado durante a etapa de aprendizagem. Dessa forma, fica claro que a estrutura de uma rede neural é, portanto, basicamente composta por neurônios.

O processamento de uma rede neural começa com a leitura de uma base de dados, que é basicamente a fonte de conhecimento a ser adquirido pela rede durante a etapa de aprendizagem, que por sua vez é responsável pelo processo de aquisição do conhecimento contido na base de dados através do ajuste dos pesos sinápticos.

2.1.2. Neurônio biológico

Um neurônio biológico (Figura 1) pode ser dividido basicamente em três partes distintas: corpo celular, dendritos e axiônio. Os impulsos celulares vindos de outros neurônios são recebidos pelos dendritos. Em seguida, o impulso é transmitido para o corpo celular, onde a informação é processada e um novo impulso é gerado. O novo impulso é transmitido dos axônios para os dendritos dos outros neurônios, num processo denominado sinapse.

Os neurônios biológicos seguem os padrões da lei do tudo ou nada. Partindo desse princípio, um neurônio possui um limiar de ativação responsável por definir se um estímulo é ou não transmitido adiante.

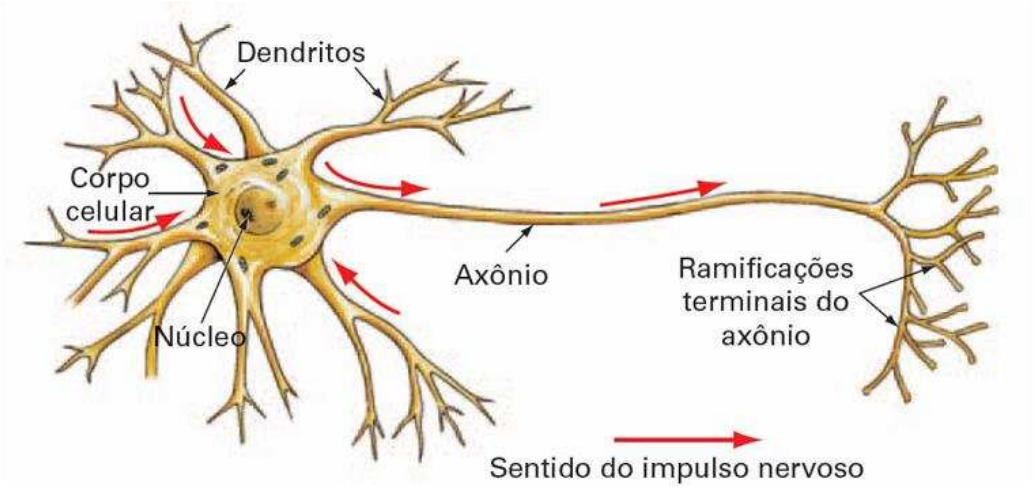

Figura 1. Neurônio biológico

2.1.3. Neurônio artificial

O primeiro modelo matemático do neurônio foi definido por McCulloch e Pitts em 1943 (Figura 2) [10], e tinha como objetivo simular o comportamento do neurônio do cérebro humano. Um modelo bastante simples, *benchmark* para os mais variados modelos subseqüentes.

No neurônio de McCulloch e Pitts, as entradas representam os dendritos, que apresentam pesos nos relacionamentos com o corpo celular. Os

estímulos são processados na *função de soma* e em seguida passam por uma função de transferência, responsável por representar a lei do tudo-ou-nada, interferindo na saída para o próximo neurônio.

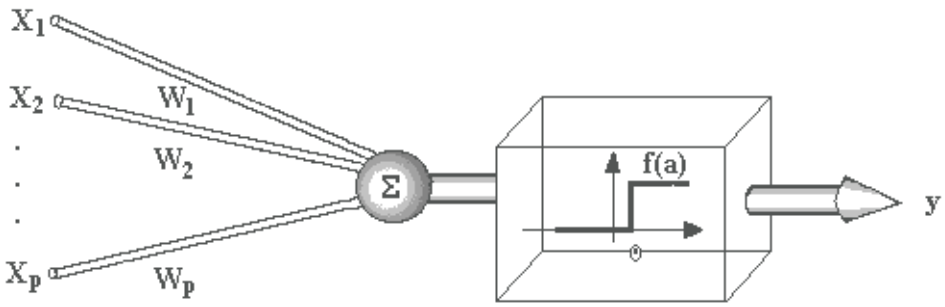

Figura 2. Modelo matemático do neurônio de McCulloch e Pitts

2.1.4. Aprendizado

A qualidade do aprendizado de uma rede neural define o quão eficaz será sua resposta diante do problema ao qual a rede foi configurada para resolver. Assim como os humanos, uma rede neural tira conclusões baseada na experiência que teve com o tipo de informação apresentado. Dessa forma, o processo de aprendizagem está diretamente relacionado com a qualidade da inteligência da rede, permitindo inclusive aperfeiçoamento no desempenho da rede através desse processo.

Existem hoje várias metodologias de processamento e aprendizagem de redes neurais [7], as quais podem ser basicamente agrupadas em aprendizado supervisionado e não supervisionado.

Aprendizado Supervisionado

O método consiste em um processo de aprendizagem em que as entradas e saídas são conhecidas. Dessa forma, é como se um professor acompanhasse o processo de aprendizagem da rede, verificando se a saída da rede está de

acordo com a saída desejada, e assim a rede vai se ajustando até que possa responder com sucesso uma considerável parte. A cada entrada na rede, a rede comparar a saída calculada com a saída esperada. Se o resultado não foi o esperado, os pesos da rede são recalculados para se ajustar ao resultado esperado.

Aprendizado Não Supervisionado

Esta metodologia se difere da anterior no fato de que não existe um “professor” que avalie a qualidade da saída processada pela rede neural. O aprendizado não supervisionado consiste na rede ter a capacidade de avaliar características das entradas e agrupá-las ou criar novas classes. Dessa forma, a rede neural desenvolve uma capacidade de criar representações internas a partir de padrões estatísticos.

2.1.5. Perceptron de uma camada

A rede perceptron é a mais antiga rede neural. A rede possui somente as camadas de entrada e saída (sem camadas intermediárias), e devido ao fato da rede perceptron utilizar a função degrau como função de ativação, as saídas da são somente dois valores, geralmente zero ou um. Se a saída da rede neural antes de entrar na função de ativação for menor que um determinado limiar (geralmente zero), a saída é 1, senão, 0.

A rede perceptron contendo apenas uma camada apenas é capaz de solucionar problemas cujas classes são linearmente separáveis.

2.1.6. Perceptron multicamada (MLP – *Multilayer perceptron*)

As redes perceptron multicamadas contém além dos nós de entrada e saída, uma ou mais camadas intermediárias de neurônios. Diferentemente da rede

perceptron de uma única camada, a rede MLP geralmente aplica a função sigmóide ou tangente hiperbólica como função de ativação.

Existe uma enorme variedade de técnicas de aprendizado para redes perceptron multicamadas, sendo o *back-propagation* a técnica mais difundida. No *back-propagation*, o erro gerado na saída é processado por uma função específica para processamento do erro e o resultado é então propagado para as demais camadas.

2.1.7. Aplicações

Redes neurais é uma técnica largamente difundida, e soluciona problemas complexos dos mais variados gêneros. Dentre as aplicações mais conhecidas, destacam-se sistemas voltados para o mercado financeiro, como suporte a decisões no mercado de ações, por exemplo. Redes neurais também são encontradas em sistemas de reconhecimento de caracteres (OCR), reconhecimento de voz, auxílio em diagnóstico médico, robótica, reconhecimento de padrões, avaliação de crédito, etc.

2.2. Otimização por colônia de formigas (ACO – *Ant colony optimization*)

Otimização por colônia de formigas é uma meta-heurística construtiva baseada no comportamento real das formigas, utilizado para resolução de problemas de otimização, incluindo seleção de subconjunto de fatores.

Como Blum e Langley descrevem em [2], a maioria dos algoritmos de seleção de subconjunto de fatores consistem de 4 componentes básicos: a escolha do ponto de partida no espaço de subconjunto de fatores, a função de busca, uma função de avaliação e critérios de parada.

O algoritmo foi primeiramente proposto por Marco Dorigo [3][4], e foi desenvolvido com o intuito de encontrar soluções eficazes para problemas de otimização através de probabilidade e deposição de feromônio.

2.2.1. Visão geral do ACO

Muitas espécies de formigas são praticamente cegas. Dessa forma, as formigas produzem uma substância denominada feromônio e a utilizam para se comunicar.

Na busca por alimentos, as formigas iniciam seus percursos de forma aleatória (Figura 3), até que alguma formiga encontre alimento e retorne para a colônia depositando feromônio no percurso. A partir desse ponto, a aleatoriedade no percurso das formigas começa a diminuir, pois elas tendem a seguir caminhos que possuem mais feromônio (Figura 4).

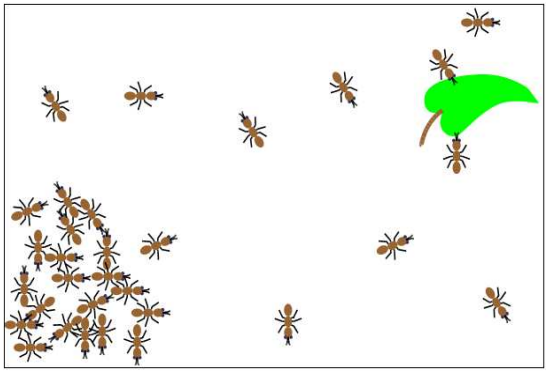

Figura 3. Formigas explorando o ambiente em busca de alimento

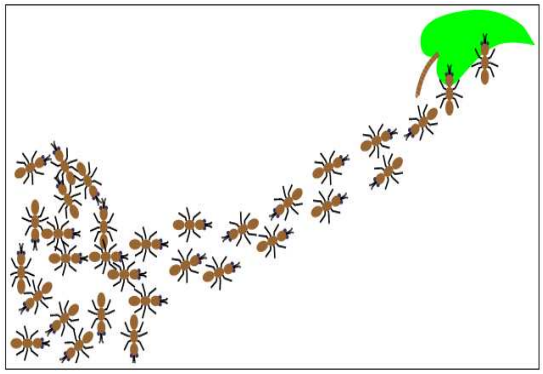

Figura 4. Formigas intensificando o melhor caminho encontrado

À medida que o tempo transcorre, o feromônio antigo evapora, e novas deposições são feitas, mantendo então trilhas às vezes muito bem definidas por terem altos índices de feromônio. O processo de evaporação é importante, pois evita que o feromônio cresça indefinidamente e permite que soluções ruins sejam esquecidas pelas formigas. As formigas tendem a seguir o caminho mais curto, intensificando a quantidade de feromônio naquela trilha, fazendo com que mais formigas trilhem aquele percurso e diminuindo a probabilidade de exploração de novas vias.

A probabilidade $p_k(r,s)$ da formiga k que se encontra sobre um atributo s escolher um próximo atributo factível r (ainda não visitado) é dada pela Equação 2.1.

$$p_k(r,s) = \begin{cases} \frac{[\tau(r,s)]\cdot[n(r,s)]^{\beta}}{\sum_{u \in J_k(r)}[\tau(r,s)]\cdot[n(r,s)]^{\beta}}, & \text{se } s \in J_k(r) \\ 0, & \text{senão} \end{cases} \quad (2.1)$$

Na fórmula acima, $\tau(r,s)$ representa a quantidade de feromônio existente na trilha que liga a fonte de alimento r a s. Para o problema proposto, cada fonte de alimento pode ser visto como um atributo da base de dados. $n(r,s)$ é a informação heurística associada ao problema. No escopo do problema estudado, a informação heurística é o inverso do *fitness*. No problema do caixeiro viajante, essa informação é a distância entre as duas fontes de alimento [5]. β é um parâmetro que determina a influência da

heurística na escolha do caminho, realizando uma ponderação no valor associado. $J_k(r)$ Representa os caminhos factíveis (trilhas ainda não visitadas pela formiga k na iteração corrente). Percebe-se então que uma formiga nunca visita uma mesma fonte de alimento duas vezes em uma mesma iteração.

Na maioria dos algoritmos relacionados à ACO, a convergência do algoritmo ocorre no momento em que todas as formigas estiverem trilhando o mesmo caminho (Figura 5). Isso significa que nesse ponto não haverá mais exploração de novos caminhos e que as formigas se estagnaram em um determinado percurso.

Os algoritmos de otimização por colônia de formigas têm demonstrado bastante eficácia na resolução do problema do caixeiro viajante, encontrando soluções quase ótimas para o problema proposto [5].

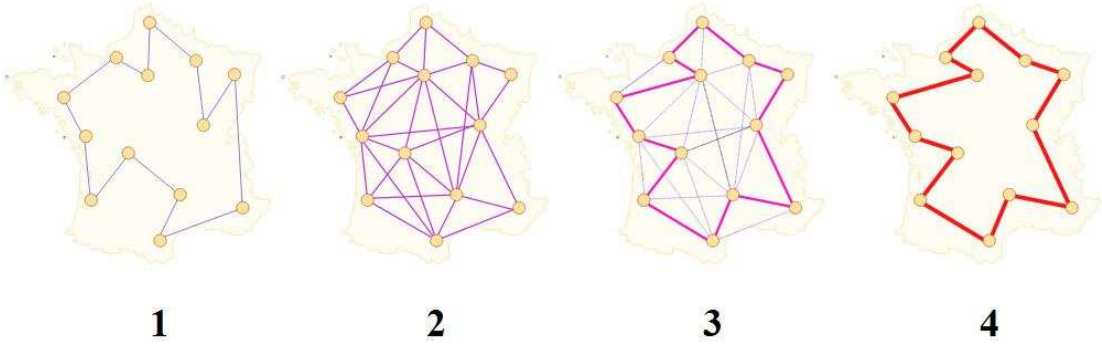

Figura 5. Processo de evaporação de feromônio nas trilhas ruins e intensificação do feromônio no caminho mais curto encontrado

2.2.2. *Ant System* (AS)

O *Ant System* [4] foi o primeiro algoritmo desenvolvido com inspiração em colônia de formigas. As formigas são inicialmente colocadas aleatoriamente sobre os atributos, e em seguida devem percorrer um caminho pelos atributos com probabilidade de escolha do próximo caminho dada pela Equação 2.1.

No *Ant System*, o processo de atualização de feromônio ocorre quando todas as formigas constroem seus percursos. A quantidade $\Delta\tau_k(r,s)$ de feromônio depositado pela formiga k no percurso de r até s é calculada dividindo uma constante Q definida no início da simulação pela informação heurística L_k encontrada no final do percurso da formiga, que representa a distância percorrida pela formiga k na construção da solução S_k, como mostra a Equação 2.2:

$$\Delta\tau_k(r,s) = \begin{cases} \frac{Q}{L_k}, & \text{se } (r,s) \in S_k \\ 0, & senão \end{cases} \quad (2.2)$$

No final de uma iteração em que todas as formigas constroem suas soluções, a atualização do feromônio na aresta $\tau(r,s)$ é calculada a partir da evaporação do feromônio naquela área acrescida ao somatório de todas as deposições realizada pelas formigas que trilharam o percurso citado, como é mostrada na Equação 2.3:

$$\tau(r,s) = (1 - \rho)\tau(r,s) + \sum_{k=1}^{m} \Delta\tau_k(r,s) \quad (2.3)$$

2.2.3. *Ant Colony System* (ACS)

O algoritmo *Ant Colony System* foi desenvolvido por Dorigo e Gambardella [5] e é bastante utilizado na comunidade científica devido à qualidade das soluções que são construídas pelas formigas.

O ACS, diferentemente do AS, utiliza o conceito de elitismo durante a atualização de feromônio; somente a formiga *best-so-far* (formiga que construiu a melhor solução desde o início da simulação) realiza a atualização global de feromônio de acordo com a Equação 2.5, depositando ao final de uma iteração. Todas as formigas realizam uma atualização local de feromônio de acordo com a Equação 2.4, aplicado após a travessia de uma aresta, removendo parte do feromônio, aumentando a diversificação da construção das soluções.

$$\tau(r,s) = (1 - \varepsilon)\tau(r,s) + \tau_0 \quad (2.4)$$

ε ($0 < \varepsilon < 1$) é um coeficiente de evaporação local e τ_0 é o nível mínimo de feromônio.

$$\tau(r,s) = (1 - \rho)\tau(r,s) + \rho\Delta\tau_{bs}(r,s), \quad \forall (r,s) \in T_{bs} \qquad (2.5)$$

τ_{bs} representa uma trilha que seja parte do conjunto de trilhas que compõem a melhor solução T_{bs}.

Além da regra de atualização de feromônio, o ACS possui mais uma particularidade que o diferencia do AS: uma regra proporcional pseudo-aleatória responsável por intensificar um caminho, ou utilizar uma diversificação tendenciosa. Com uma probabilidade q_0, a escolha do próximo atributo de uma formiga será a trilha com maior quantidade de feromônio, intensificando a quantidade de feromônio naquela região, e com probabilidade $(1 - q_0)$, a formiga irá utilizar a Equação 2.1 para escolha do próximo atributo, da mesma forma que no *Ant System*, realizando uma diversificação tendenciosa.

2.2.4. Aplicações

Alguns algoritmos de ACO produzem soluções quase ótimas para o problema do caixeiro viajante [5]. A colônia de formigas tem grande capacidade de adaptação, podendo ser aplicados em problemas de tempo real e em grafos que mudam dinamicamente, diferentemente do algoritmo genético e do *simulated annealing*. Devido a grande capacidade adaptativa, ACO é bastante utilizado para encontrar soluções para problemas de roteamento de redes, assim como em transporte urbano.

2.3. Sistemas Neurais Híbridos (SNHs)

Um sistema híbrido (SH) é composto por pelo menos duas técnicas de computação inteligente. Combinar técnicas de computação inteligente pode ser bastante poderoso, pois abre a possibilidade de resolver problemas que uma técnica por si só não seria suficiente. Nesse contexto, um sistema híbrido busca eliminar ou diminuir significativamente limitações que uma técnica individualmente possui, e dessa forma aumentar a qualidade das soluções geradas por esses sistemas.

Um sistema neural híbrido (SNH) é um sistema híbrido composto por uma rede neural e uma ou mais técnicas de computação inteligente. Utilizar um algoritmo de colônia de formigas para realizar o pré-processamento da base de dados pode ser considerado um sistema híbrido.

Dessa forma, SNHs possibilitam o desenvolvimento de técnicas para, por exemplo, realizar o ajuste de parâmetros da rede, como função de ativação, número de neurônios, taxa de aprendizado, ou ajuste dos pesos iniciais da rede [7].

Capítulo 3

Metodologia

Este capítulo tem como foco demonstrar na prática o desenvolvimento de aplicações relacionadas à solução do problema proposto por este trabalho, com base nas teorias descritas no capítulo anterior.

A seção 3.1 descreve o que é o alimento das formigas e como ele influencia no resultado do processamento. A seção 3.2 descreve como que o algoritmo de colônia de formigas realiza o pré-processamento dos dados da rede neural.

3.1 Seleção de subconjunto de fatores das formigas

Os algoritmos de colônias de formigas trabalham com o conceito de atributo ou *feature*, que são dados capazes de ser processados e que possuem um valor heurístico que de alguma forma influencia no processo de otimização. Em analogia com a inspiração biológica, um atributo representa uma fonte de alimento a ser buscada pelas formigas artificiais.

Inicialmente, as formigas são dispostas aleatoriamente sobre as fontes de alimento. À medida que forem construindo suas soluções, irão percorrer outros atributos da base de dados e ao final haverá um subconjunto de atributos da base para cada formiga.

O procedimento de busca resultará em um processo de otimização que consistirá na visita dos atributos da base de dados, levando em consideração uma função probabilística que considera, além do fator aleatoriedade, a quantidade de feromônio na trilha. Dessa forma, será possível encontrar uma solução muito boa para o problema, apesar de não necessariamente ser a solução ótima.

O alimento das formigas são os atributos da base de dados. A cada iteração, as formigas escolhem aleatoriamente uma quantidade de atributos a serem percorridos que é menor ou igual à quantidade de atributos da base de dados original. A partir deste ponto, as formigas “caminham” pelos atributos da base de dados, escolhendo os atributos de acordo com a Equação 2.1, O melhor subconjunto de atributos deveria ser encontrado testando todas as combinações possíveis, mas isso se torna inviável à medida que o número de atributos aumenta. Portanto, o ACO encontra não necessariamente a melhor solução possível, mas é garantida uma boa solução para o problema.

No final de cada iteração, cada formiga terá criado um conjunto de atributos que corresponde a um subconjunto da base de dados original. No final do processamento, a formiga *best-so-far* irá conter a base de dados que resulta no melhor treinamento. Se o treinamento da rede neural com essa nova base resultar em um resultado tão bom quanto a base de dados original, significa que a base original contém atributos que podem ser descartados.

É possível que o resultado do treinamento resulte em uma taxa de acerto maior do que a da base original. Isso significa que existiam atributos na base de dados que estavam de alguma forma prejudicando o treinamento da rede, comprometendo a qualidade da solução.

Modelos que fornecem um bom desempenho com a menor quantidade possível de parâmetros são os melhores a serem estudados devido à boa capacidade de generalização. Uma vez que é utilizado uma rede neural MLP como classificador/previsor, a qualidade da solução construída pela formiga é avaliada a partir do cálculo do fitness, estabelecendo uma relação que envolve o erro médio quadrático da função objetivo e a quantidade de atributos utilizados para o treinamento da mesma.

Logo, para atender este objetivo, a quantidade de atributos selecionados com relação ao total deve ser considerada no cálculo do *fitness* [1]. A heurística utilizada neste trabalho para avaliar a qualidade do processamento de uma formiga considera, portanto, mais variáveis do que apenas o erro. A Equação 3.1 mostra como é calculado o *fitness*, informação heurística, que indica a qualidade do treinamento resultante do processamento de uma rede neural para uma dada formiga.

$$Fitness = \frac{1}{((k_1 * EMQ) + (k_2 * \frac{N_s}{N_t}))} \quad (3.1)$$

k_1 e k_2 são constantes de ponderação; k_1 pondera a importância do EMQ no cálculo do *fitness*, enquanto k_2 pondera o quanto o número de atributos de entrada influencia no *fitness*. O EMQ indica o erro médio quadrático encontrado, enquanto N_s representa o número de variáveis selecionadas e N_t representa o número de variáveis da base inicial antes do pré-processamento.

3.2 Pré-processamento da base de dados com ACO

O sistema desenvolvido permite a configuração de várias redes que processam em paralelo. Inicialmente, uma ou mais redes devem ser pré-configuradas, definindo a arquitetura utilizada (e.g. perceptron, adaline, MLP), valores de alfas pra cada rede, valores de beta e quantidade de nós na(s) camada(s) escondida(s) (se aplicável), como mostra a Figura 6. Em seguida, é realizado o treinamento da rede, com a base de dados contendo todos os atributos de entrada. Até o momento, não houve nenhuma influência de algoritmo de colônia de formigas sobre a base de dados.

Após o treinamento de todas as redes, um sistema especialista verifica os resultados e escolhe a rede que apresentou os melhores resultados e armazena em memória ou em arquivo XML. Este procedimento é importante, pois os processamentos da rede neural utilizando somente subconjuntos da base original serão feitos utilizando a rede que for considerada a melhor configurada.

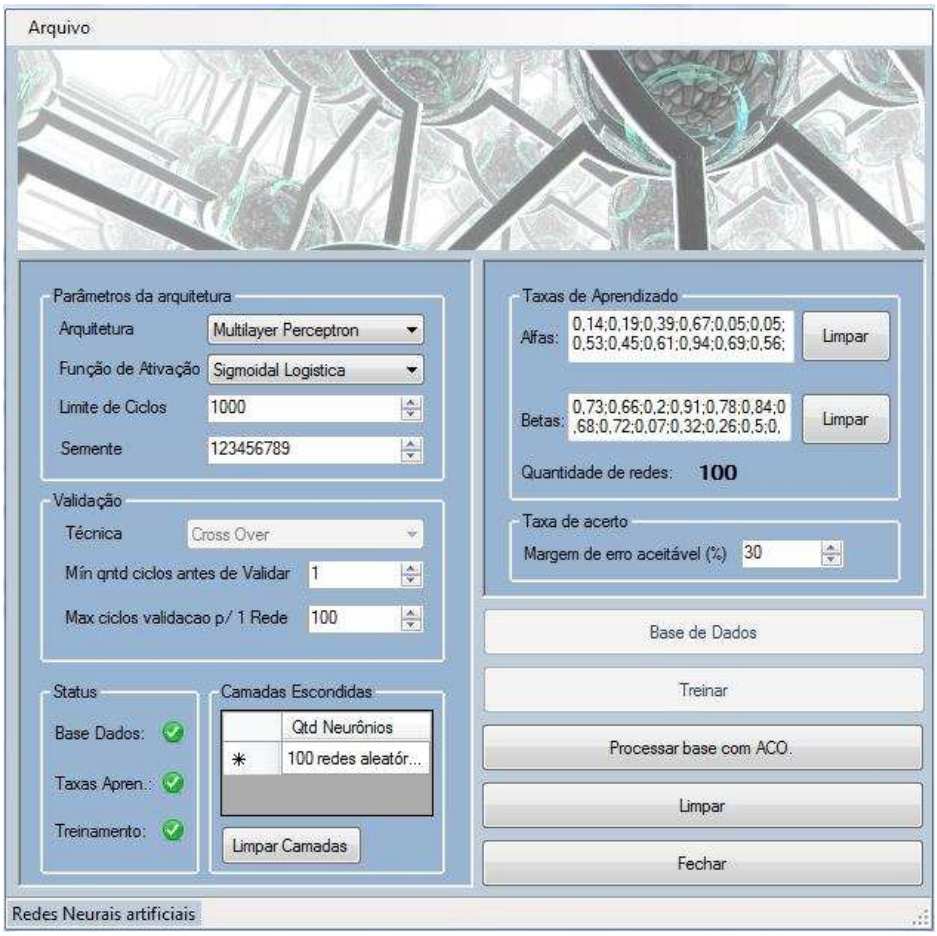

Figura 6. Tela de configuração da rede neural

Terminado o processamento das redes neurais, as formigas podem começar o trabalho de otimização. Para tal, elas utilizam a configuração da melhor rede encontrada. O sistema permite a configuração de uma colônia com a arquitetura Ant System, Elitist Ant System ou Ant Colony System.

A Figura 7 define o fluxograma do algoritmo desenvolvido. No início da simulação são criados as trilhas que interligam todos os atributos da base de dados, lembrando que cada atributo da base corresponde a uma fonte de alimento. Essas trilhas criadas serão percorridas pelas formigas para se moverem de uma fonte de alimento à outra.

Depois que as trilhas estão criadas, ocorre uma deposição inicial de feromônio em todas as trilhas. Essa deposição é uniforme e ocorre para evitar que todas as trilhas fiquem sem feromônio, evitando uma divisão por zero na Equação 2.1.

Em cada iteração, enquanto a solução não convergir, armazena-se dados da simulação (e.g. tempo de processamento, iteração corrente). Em seguida, As formigas são dispostas aleatoriamente sobre os atributos da base de dados, definindo assim o primeiro atributo do subconjunto de fatores a ser construído por cada formiga.

Cada formiga então escolhe um número aleatório que varia entre 1 e o número de atributos da base de dados original. Esse número determina o tamanho do subconjunto de atributos que cada formiga deve obter no final da iteração. Em seguida, cada formiga percorre seu caminho pelos atributos factíveis, não podendo passar pelo mesmo atributo duas vezes em uma mesma iteração, levando em consideração a quantidade de feromônio já existente, como foi apresentado na Equação 2.1.

O parâmetro $[n(r,s)]^{\beta}$ (Equação 2.1), como já foi mencionado, representa a influência da heurística no processo de exploração. Na otimização do problema do caixeiro viajante, por exemplo, este parâmetro representa a distância entre as cidades r e s. No caso do pré-processamento da rede neural, o β é igual a 0 (zero), pois todo processo probabilístico relacionado a escolha do próximo caminho factível está relacionado somente a quantidade de feromônio existente nas trilhas. A heurística (o *fitness*) está implícita dentro da fórmula de deposição de feromônio, como mostra a Equação 3.2.

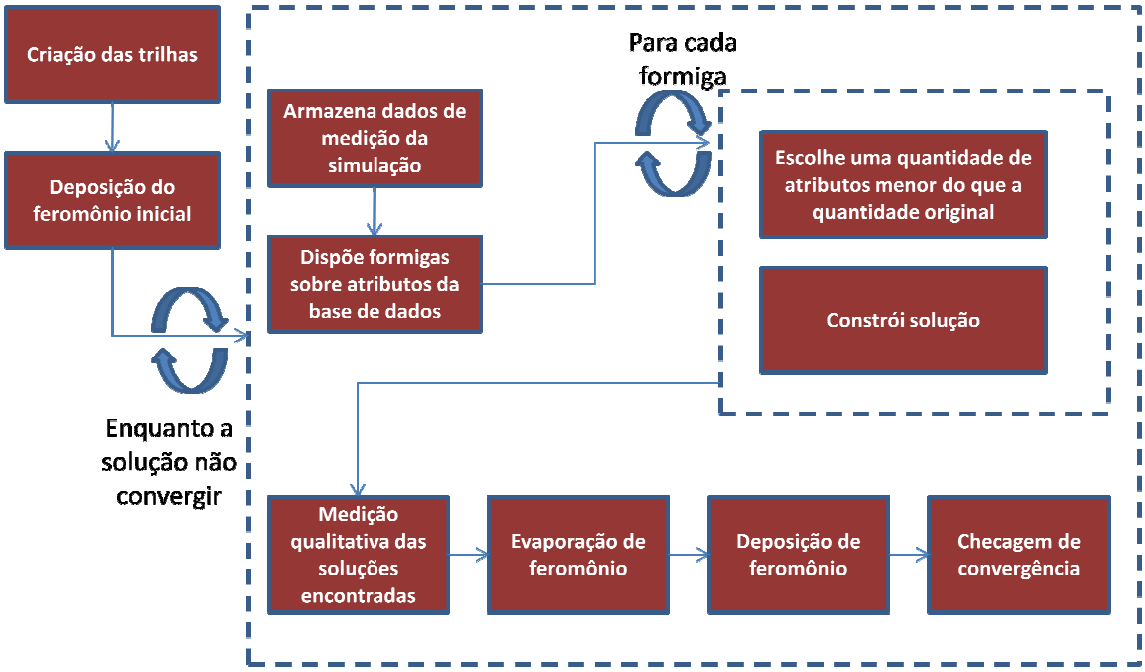

Figura 7. Fluxo do algoritmo desenvolvido para realização do pré-processamento da base de dados

No final da iteração, todas as formigas já contêm seus atributos e neste ponto suas soluções passam por um processo de medição qualitativa que corresponde a um treinamento da rede neural para cada subconjunto criado pelas formigas. Os parâmetros dessa rede neural são os mesmos parâmetros da melhor rede configurada na etapa anterior. O que muda na rede neural de uma formiga para a outra é somente a base de dados utilizada, já que cada rede será treinada utilizando uma base que contém o subconjunto de dados escolhido pela formiga. A deposição de feromônio está diretamente relacionada ao resultado do treinamento dessa rede. Quanto maior o *fitness* calculado segundo a Equação 3.1, mais feromônio será depositado. Dessa forma, a informação heurística L_k que representa a distância percorrida pela k-ésima formiga será representada pelo inverso do Fitness.

$$L_k = \frac{1}{Fitness}$$

Partindo da Equação 2.2, temos que

$$\Delta\tau_k(r,s) = \begin{cases} \frac{Q}{\left(\frac{1}{Fitness}\right)}, & \text{se } (r,s) \in S_k \\ 0, & senão \end{cases}$$

A fórmula para deposição de feromônio passa a ser então:

$$\Delta\tau_k(r,s) = \begin{cases} Q.Fitness\,, & \text{se } (r,s) \in S_k \\ 0, & senão \end{cases} \qquad (3.2)$$

Dessa forma, após o cálculo do *fitness* ocorre o processo de evaporação de feromônio, seguido pelo processo de deposição do novo feromônio. Em seguida, checa-se a convergência para saber se a simulação já pode ser interrompida. Em ACO, a convergência ocorre quando todas as formigas estão trilhando o mesmo caminho. Para o problema proposto, como as formigas são inicialmente dispostas aleatoriamente sobre a base de dados, é praticamente impossível que todas criem o mesmo subconjunto, já que naturalmente uma formiga é disposta sobre um atributo que não faz parte da solução ideal. Dessa forma, foi criado um limite de iterações empírico como condição de parada.

No final de todas as iterações, a resposta é uma comparação do melhor processamento da rede neural com uma base inteira, e o resultado do melhor processamento da rede contendo um subconjunto de dados, como mostra a Figura 8.

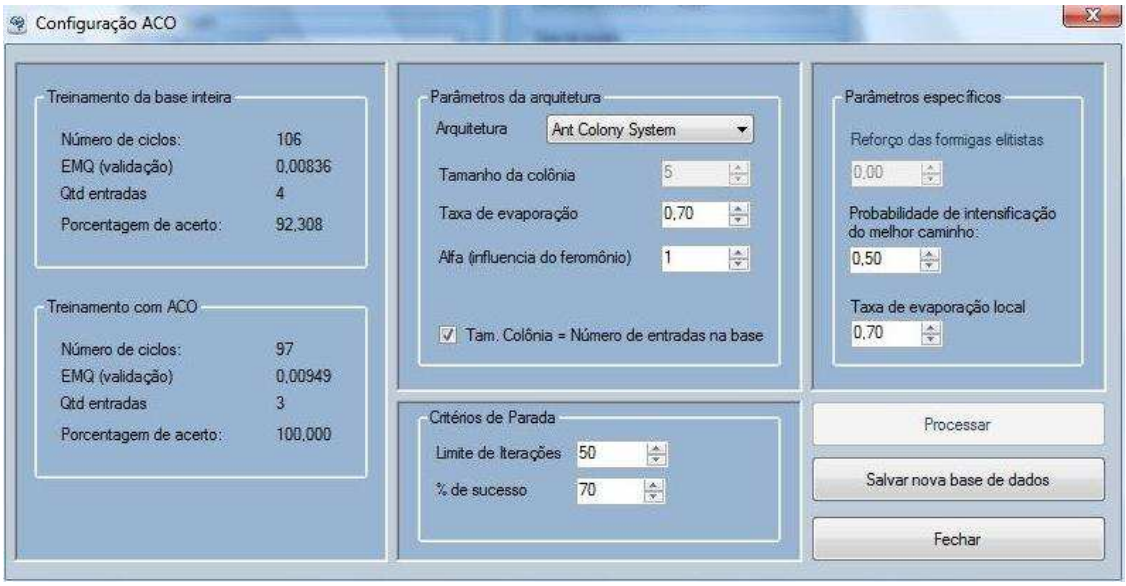

Figura 8. Resultado do processamento da colônia de formigas para uma base de dados de classificação de planta íris: formigas descartaram um atributo da base e conseguiram um acerto de praticamente 100%

Capítulo 4

Resultados

Neste capítulo será descrito como os experimentos foram conduzidos nas bases de dados, juntamente com a análise dos resultados obtidos. As seções abordam os resultados encontrados como experimentos sobre determinada base de dados. Foram utilizadas bases de dados conhecidas na literatura.

Vale ressaltar que remover um atributo que seja da base original não significa dizer que ele não tem importância no mundo real e deve ser completamente desconsiderado, e sim que o atributo de entrada utilizado para o processamento não contém valores significativos capazes de fazer uma separação tal que contribua na qualidade da classificação do algoritmo utilizado.

Neste trabalho, todas as bases de dados utilizadas são parte da UCI *Machine Learning Repository*. Todas as bases de dados tiveram seus dados nominais tratados e convertidos para dados numéricos. Todos os dados numéricos foram colocados na faixa de valores entre 0 e 1, normalizados segundo intervalo da função sigmoidal.

No cálculo do *fitness* (Equação 3.1), foi atribuído a k_1 = 3 e k_2 = 1. Esses valores foram atribuídos empiricamente. Esses valores ponderam o fitness, sendo k_1 a variável que regula a relevância do EMQ, enquanto k_2 regula a relevância da quantidade de atributos.

A seção 4.1 descreve os resultados encontrados para uma base de dados de planta íris. A seção 4.2 descreve os resultados obtidos para uma base de dados de vinhos.

4.1. ACO aplicado a base da Planta Íris

A base de dados da Planta Íris utilizada possui 150 instâncias, sendo 50 instâncias de cada uma das 3 classes possíveis. A base possui 4 atributos

numéricos, e saídas nominais, as quais foram pré-processadas para serem convertidas em valores numéricos.

Os atributos de saídas eram classificações nominais, as quais foram substituídas por 3 atributos numéricos indicando a classificação, sendo combinações de zeros e uns. A Tabela 1 apresenta o mapeamento criado para traduzir os dados nominais. A Tabela 2 indica a semântica dos atributos da base de dados.

Entre as configurações de parâmetros utilizadas, a que apresentou o melhor resultado de processamento foi uma rede MLP. Os parâmetros utilizados nesta rede neural estão descritos pela Tabela 3. As instâncias escolhidas para treinamento, validação e testes foram selecionadas aleatoriamente da base de dados.

Tabela 1. Conversão das saídas nominais para valores numéricos distribuídos em três atributos

Dado Nominal	Valor Representativo da Classe
íris-setosa	1-0-0
íris-versicolor	0-1-0
íris-virginica	0-0-1

Tabela 2. Atributos de entrada da base da planta íris e sua respectiva ordem

Ordem	Atributo
1	Comprimento da sépala (cm)
2	Largura da sépala (cm)
3	Comprimento da pétala (cm)
4	Largura da pétala (cm)

Tabela 3. Parâmetros da RNA MLP utilizada para os processamentos da base de dados da planta íris

Parâmetros	Valores
Alfa	0,25
Beta	0,4
Função ativação	Sigmoidal Logística
Quantidade camadas escondidas	1
Quantidade de neurônios nas camadas escondidas	3
Técnica de validação	Cross Over
Quantidade de Instâncias para Treinamento	76
Quantidade de Instâncias para Validação	37
Quantidade de Instâncias para Testes	37

Ao final do treinamento, a taxa de acerto encontrada foi de 95.595%. A Figura 9 apresenta o gráfico do erro médio quadrático relacionado aos ciclos processados.

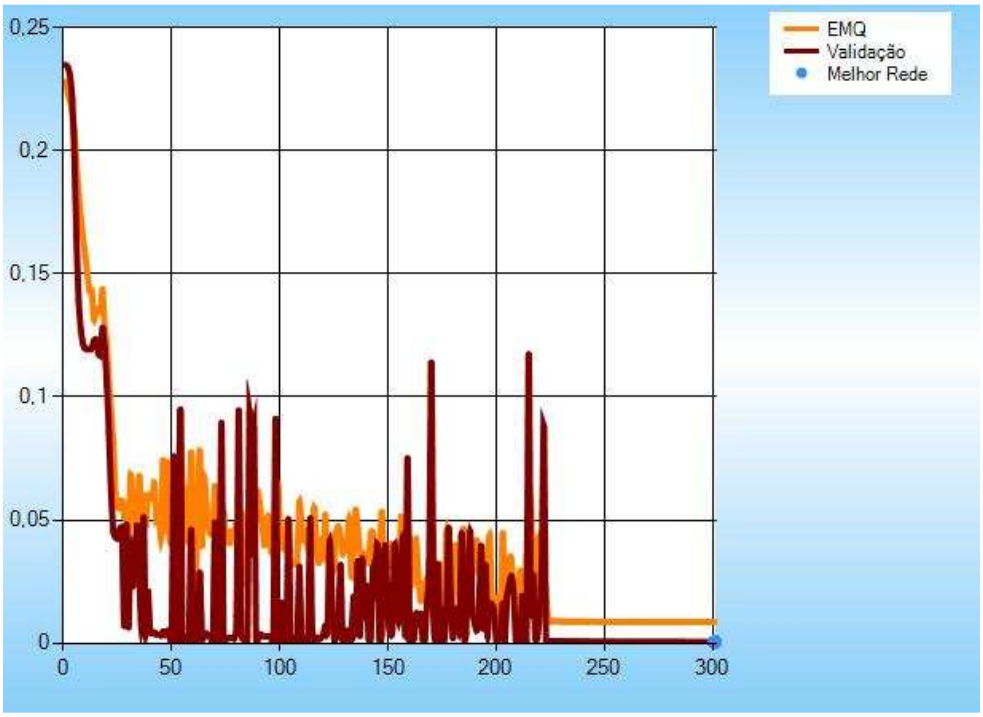

Figura 9. Ciclo x EMQ relacionado o processamento da base de dados da planta Iris com uma RNA MLP

Utilizando a mesma base de dados e configurações de parâmetros, foi realizado um pré-processamento da base utilizando os algoritmos de colônia de formigas conhecidos na literatura e definidos no capítulo anterior. A Tabela 4 apresenta os parâmetros fixados para todas as simulações realizadas sobre este experimento. Como critério de parada, foi definido um máximo de 50 iterações ou porcentagem de acerto relativamente estagnada acima de 70%.

Tabela 4. Parâmetros comuns a todas as simulações realizadas no experimento da planta íris

Parâmetro	**Valores**
Qtd Formigas	5
Taxa Evaporação Global	0,7
Alfa	1
Beta	0

Os algoritmos *Ant System* e *Elitist Ant System* não geraram bons resultados para o problema proposto. O melhor resultado encontrado pelo AS foi uma taxa de acerto de 24.323% para 3 atributos. O EAS, com uma taxa de reforço de 30% sobre o melhor caminho, conseguiu uma taxa de acerto de 45.485% com 3 atributos. A Tabela 6, assim como a Figura 11, apresenta os resultados encontrados por cada algoritmo. A Figura 10 dá uma noção na qualidade do processamento para os algoritmos de colônia de formigas.

No algoritmo *Ant Colony System*, os parâmetros utilizados estão descritos na Tabela 5. Após o processamento, a base resultante foi uma base com um atributo a menos. As formigas apresentaram uma configuração da base de dados contendo 3 dos 4 atributos de entrada, alcançando uma taxa de acerto de 93.248%. Segundo a saída do algoritmo, o atributo que indica o comprimento da sépala não precisa existir nessa base, já que a ausência do mesmo resultou em uma diferença muito pouco significativa na taxa de acerto. Isso não significa que o atributo deva ser considerado no mundo real. Significa apenas que para esta base de dados, seus valores não separam bem as classes.

Tabela 5. Parâmetros utilizados pelo ACS para simulações com base de dados da planta íris

Parâmetro	Valores
Taxa Evaporação Local	0,6
Prob. Intensificação melhor caminho	0,6

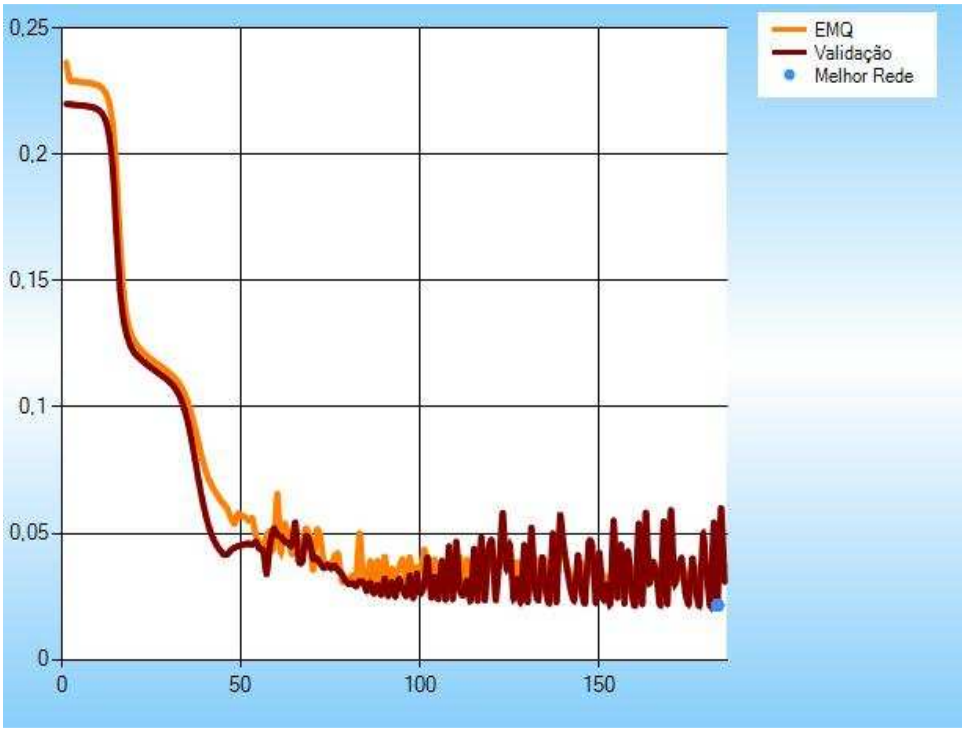

Figura 10. Ciclo x EMQ relacionado o pré-processamento da base de dados da planta Iris de uma RNA MLP com ACS

Tabela 6. Relação entre os resultados obtidos, quantidade de atributos e algoritmo de colônia de formigas utilizado

Algoritmo	Qtd Atributos	Taxa de Acerto
AS	3	24,323%
EAS	3	45,485%
ACS	3	93,248%
Nenhum	4	95,595%

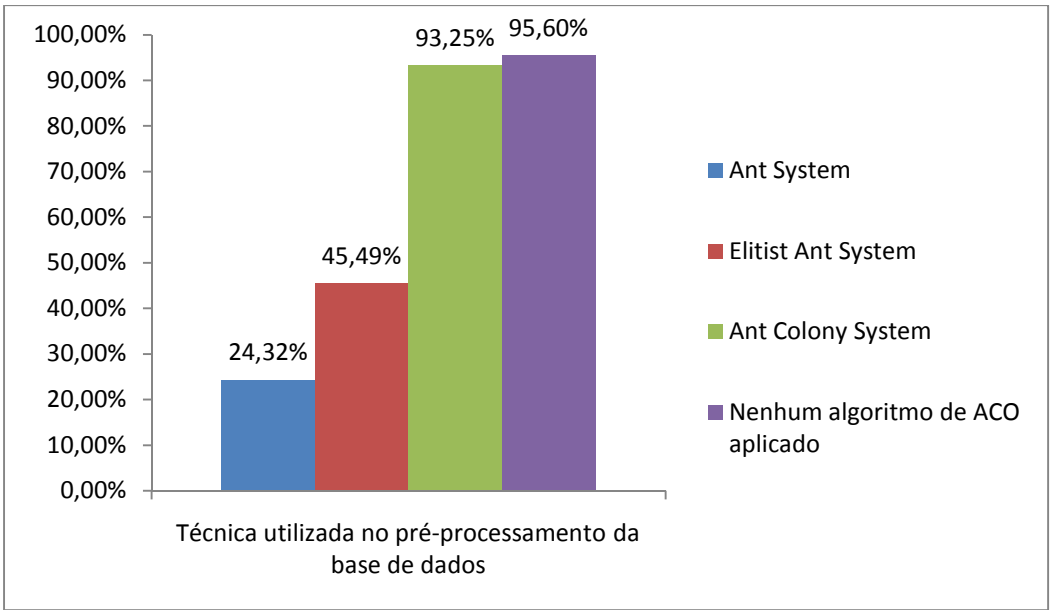

Figura 11. Gráfico comparativo entre as taxas de acerto encontradas a partir do treinamento da rede neural com a base de dados da planta íris pré-processada por algoritmos de ACO

4.2. ACO aplicado a uma base de dados de Vinhos

A base de dados de vinhos utilizada possui 178 instâncias e na saída apresenta 3 possíveis classificações, que indicam o lugar onde foi cultivado o vinho dentro de uma determinada região da Itália. A Tabela 7 mostra a distribuição de instância de acordo com as classificações possíveis.

Tabela 7. Divisão de instância entre as classificações possíveis na base de dados de vinhos

Classificação	Quantidade de Instâncias
1	59
2	71
3	48

A base de dados de vinhos possui 13 atributos de entrada, dispostos de acordo com a Tabela 8.

Tabela 8. Atributos da base de vinhos e sua respectiva ordem

Ordem	Atributo
1	Álcool
2	Ácido málico
3	Cinzas
4	Alcalinidade das cinzas
5	Magnésio
6	Total de fenols
7	Flavonóides
8	Fenols não-flavonóides
9	Proantocianidina
10	Intensidade da cor
11	Matiz
12	OD280/OD315 de vinhos diluídos
13	Prolina

A rede MLP apresentou o melhor resultado no processamento. Os parâmetros utilizados podem ser visualizados na Tabela 9. As instâncias da base foram selecionadas aleatoriamente para formarem as bases de treinamento, validação e testes.

Tabela 9. Parâmetros da RNA MLP utilizada para os processamentos da base de dados de vinhos

Parâmetros	Valores
Alfa	0,4
Beta	0,32
Função ativação	Sigmoidal Logística
Quantidade camadas escondidas	1
Quantidade de neurônios nas camadas escondidas	7
Técnica de validação	Cross Over
Quantidade de Instâncias para Treinamento	89
Quantidade de Instâncias para Validação	45
Quantidade de Instâncias para Testes	44

A taxa de acerto encontrada no final do treinamento foi de 96.238%. A Figura 12 apresenta o gráfico do erro médio quadrático relacionado aos ciclos processados.

Figura 12. Ciclo x EMQ relacionado o processamento da base de dados de vinhos com uma RNA MLP

O pré-processamento da base de dados realizado pelos algoritmos de colônia de formigas descritos no capítulo anterior foi realizado sobre a mesma base de dados, utilizando inclusive mesma configuração e parâmetros. A Tabela 10 apresenta os parâmetros compartilhados entre as simulações. Foi definido um máximo de 50 iterações ou porcentagem de acerto relativamente estagnada acima de 70% como critério de parada.

Tabela 10. Parâmetros comuns a todas as simulações realizadas no experimento da base de vinhos

Parâmetro	Valores
Qtd Formigas	13
Taxa Evaporação Global	0,7
Alfa	1
Beta	0

Os algoritmos *Ant System* e *Elitist Ant System* não geraram bons resultados. Simulando com AS, foi obtida uma taxa de acerto de 68.813% para 7 atributos. Simulando com EAS, com uma taxa de reforço de 30% sobre o melhor caminho, foi obtida uma taxa de acerto de 79.731% com 5 atributos. A Tabela 12 apresenta os resultados encontrados, filtrados por técnica utilizada. A Figura 13 apresenta o gráfico do erro médio quadrático relacionado aos ciclos processados. A Figura 14 apresenta um comparativo entre os resultados encontrados por cada algoritmo simulado.

O ACS mais uma vez apresentou resultados bastante significativos. Os parâmetros utilizados para o processamento estão descritos na Tabela 11. A base foi reduzida para 15% do seu tamanho original, contendo apenas 2 dos 13 atributos de entrada iniciais, com uma taxa de acerto de 95.455%. Os resultados mostram que, para esta base de dados, conhecendo apenas a taxa de flavonóides e a OD280/OD315 de vinhos diluídos, é possível conseguir uma taxa de acerto bastante próxima a taxa de acerto do treinamento da base inteira. Vale ressaltar que isso não significa que os demais atributos não tenham importância no mundo real. Os resultados mostram que a base de dados contém informações que poderiam ser desconsideradas para obter resultados satisfatórios.

Tabela 11. Parâmetros utilizados pelo ACS para simulações com base de dados de vinhos

Parâmetro	Valores
Taxa Evaporação Local	0,6
Prob. Intensificação melhor caminho	0,6

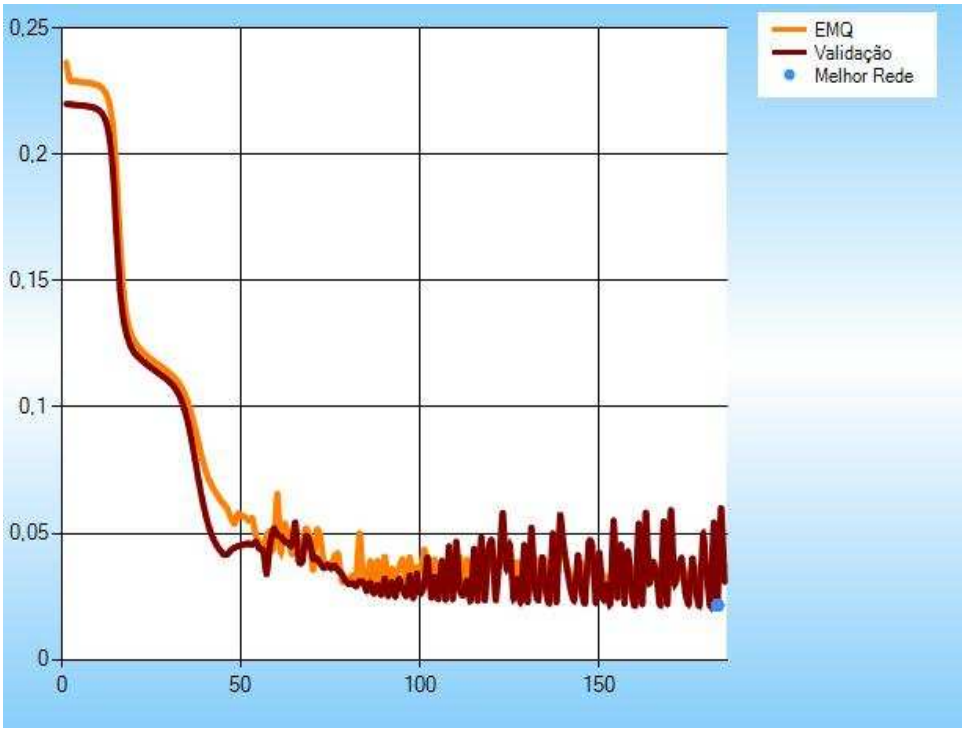

Figura 13. Ciclo x EMQ relacionado o pré-processamento da base de dados de vinhos de uma RNA MLP com ACS

Tabela 12. Relação entre os resultados obtidos, quantidade de atributos e algoritmo de colônia de formigas utilizado no pré-processamento da base de vinhos

Algoritmo	Qtd Atributos	Taxa de Acerto
AS	7	68,813%
EAS	5	79,731%
ACS	2	95,455%
Nenhum	13	96,238%

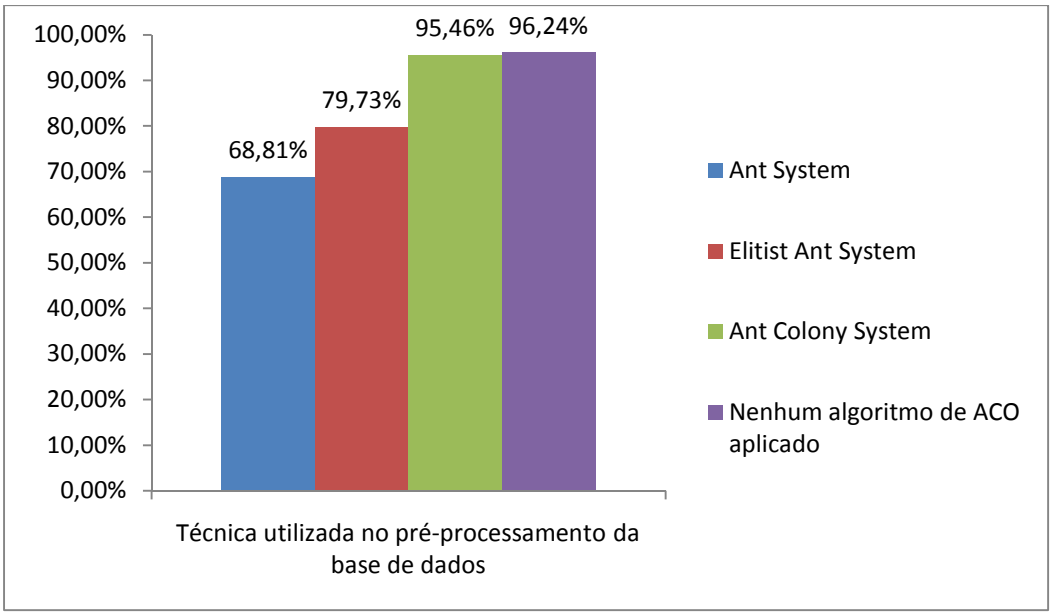

Figura 14. Gráfico comparativo entre as taxas de acerto encontradas a partir do treinamento da rede neural com a base de dados de vinhos pré-processada por algoritmos de ACO

4.3. Comentários Finais

Nesta seção, realizamos experimentos utilizando o algoritmo ACO para realização do pré-processamento da base de dados. Nas duas bases utilizadas, o desempenho do ACO foi bastante satisfatório, reduzindo significativamente o tamanho da base de dados sem perder qualidade da taxa de acerto.

Em todos os testes realizados, o Ant Colony System apresentou resultados melhores do que os algoritmos de colônia de formigas Elitist Ant System e o Ant System, apresentando taxas de acerto melhores e base de dados ainda menores do que os demais algoritmos.

Capítulo 5

Conclusão

O principal objetivo deste trabalho foi melhorar a qualidade do processamento de uma rede neural, reduzindo o tempo de processamento significativamente sem perder a qualidade da taxa de acerto, podendo até mesmo aumentá-la durante o processo. Foi visto que com ACO podemos realizar o pré-processamento da base de dados através da seleção de um subconjunto de fatores e assim reduzir significativamente o tamanho da base de dados sem reduzir a taxa de acerto.

Neste projeto foram abordadas as técnicas de redes neurais mais tradicionais e difundidas na literatura, assim como os conceitos dos algoritmos de otimização por colônia de formigas mais conhecidos na literatura e a importância desses algoritmos em problemas de otimização e seleção de variáveis.

No capítulo sobre a metodologia, foram abordados os fundamentos matemáticos básicos que regem os algoritmos estudados, e as diferenças entre eles.

Os experimentos foram realizados sobre bases de dados conhecidas na literatura. Utilizamos primeiramente a base de dados da planta íris por ser bastante conhecida, e vimos que apesar da base ser difundida por apresentar um resultado de treinamento satisfatório, há pelo menos um atributo que pode ser removido dela sem prejudicar a qualidade do treinamento da rede. Em seguida realizamos experimentos com uma base de dados de vinhos. Os resultados mostraram que mesmo com a base de dados sendo reduzida a 15% do seu tamanho original, a taxa de acerto permaneceu tão alta quanto a taxa resultante do processamento com a base inteira. Em todos os testes, o algoritmo ACS apresentou os melhores resultados.

Portanto, concluímos que os algoritmos de otimização de colônia de formigas, a partir dos resultados obtidos nos experimentos, é adequado ao

problema de pré-processamento da base de dados para eliminar atributos que não contribuem ou até mesmo atrapalham o processamento da rede neural.

5.1. Trabalhos Futuros

Como trabalhos futuros, este trabalho pode ser estendido da seguinte forma:

- Aplicar outras técnicas de IA para pré-processamento de base de dados e comparar os resultados com os dos algoritmos de formigas utilizados.
- Realizar experimentos em problemas de previsão.
- Utilizar o ACO para otimizar também a arquitetura da Rede Neural, ou seja, utilizar o ACO para otimizar o número de neurônios na camada escondida.
- Otimizar os parâmetros do algoritmo *backpropagation* (taxa de aprendizagem e momento) com ACO.
- Aplicar está técnica de ACO com outras arquiteturas de redes neurais, tais como, redes *neurofuzzy*.
- Realizar estudo do esforço computacional da aplicação de ACO para a seleção de atributos da base de dados.
- Realizar um comparativo entre técnicas de ACO e PCA para problemas de seleção de variáveis

Bibliografia

[1] Al-Ani, A, **Ant Colony Optimization for Feature Subset Selection**, World Academy of Science, Engineering and Technology, 2005

[2] Blum, A. L.; Langley, P., **Selection of relevant features and examples in machine learning**. Artificial Inteligence, 97.245-271, 1997.

[3] Dorigo, M.; Colorni, A.; Maniezzo, V., **Distributed optimization by ant colonies**, Proceedings of ECAL'91, European Conference on Artificial Life, Elsevier Publishing, Amsterdam, 1991.

[4] Dorigo, M.; Colorni, A.; Maniezzo, V., **The ant system: an autocatalytic optimizing process**, Technical Report TR91-016, Politecnico di Milano (1991).

[5] Dorigo, M.; Gambardella, L.M., **Ant Colony System : A Cooperative Learning Approach to the Traveling Salesman Problem**, IEEE Transactions on Evolutionary Computation, volume 1, número 1, pages 53-66, 1997.

[6] Goldberg, David E., **Genetic Algorithms in Search, Optimization, and Machine Learning**. Addison-Wesley, 1989.

[7] Haykin, S. **Neural Networks: A Comprehensive Foundation**. IEEE Press, 1999.

[8] Jolliffe, I. T., **Principal Component Analysis**. Springer-Verlag. pp. 487, 1986;

[9] Mark A. Hall., **Correlation-based Feature Selection for Machine Learning**. University of Waikato. 1999.

[10] MCCULLOCH, W. S.; PITTS, W, **A logical calulus of the ideas immanent in nervous activity**. Bulletin os Mathematical Biophisics, p.115-133, 1943.

[11] Mehrotra K.; Mohan C.; Ranka S., **Elements of Artificial Neural Networks**. Cambridge, MA: MIT Press; 1997.