ALGORITMOS GENÉTICOS DE AGRUPAMENTO PARA EXTRAÇÃO DE REGRAS DE REDES NEURAIS

Eduardo Raul Hruschka

TESE SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE INTEGRANTE DOS REQUISITOS NECESSÁRIOS PARA OBTENÇÃO DO GRAU DE DOUTOR EM CIÊNCIAS EM ENGENHARIA CIVIL.

Aprovada por:

_____
Prof. Nelson Francisco Favilla Ebecken, D.Sc.

_____
Prof. André Carlos Ponce de Leon Ferreira de Carvalho, Ph.D.

_____
Prof.ª Beatriz de Souza Leite Pires de Lima, D.Sc.

_____
Prof. Djalma Mosqueira Falcão, Ph.D.

_____
Prof. José Luis Drummond Alves, D.Sc.

RIO DE JANEIRO, RJ - BRASIL
NOVEMBRO DE 2001

HRUSCHKA, EDUARDO RAUL

Algoritmos Genéticos de Agrupamento para Extração de Regras de Redes Neurais

[Rio de Janeiro] 2001

IX, 139 p., 29,7 cm (COPPE/UFRJ, D.Sc., Engenharia Civil, 2001)

Tese - Universidade Federal do Rio de Janeiro, COPPE

1. Extração de Regras de Redes Neurais
2. Algoritmos Genéticos
3. Agrupamento (*Clustering)*
4. D*ata mining*

I - COPPE/UFRJ II -Título(série)

DEDICATÓRIA

À Luciana, pela compreensão da importância desta tese para a minha realização pessoal e, principalmente, pelo apoio dedicado durante os momentos mais difíceis.

AGRADECIMENTOS

Ao professor Nelson, pela confiança depositada neste trabalho. Sua sabedoria, sua paciência e suas orientações sempre pertinentes foram determinantes para a realização desta tese. Sua amizade me manteve sempre motivado.

Aos meus pais, Deodato e Bernadete, pelo incentivo aos meus estudos.

À minha irmã, Daniela, e aos meus irmãos, Gustavo e Ricardo, pelo apoio recebido durante a realização deste trabalho.

À minha avó, Relinda e ao meu avô, Arnoldo, pelo exemplo que representam e pelo que têm me ensinado.

Ao meu amigo Estevam, pelo interesse demonstrado neste trabalho e por suas contribuições nas várias discussões técnicas que tivemos.

À minha sogra, Maria Oneide, pelo acolhimento dedicado e atencioso que recebi durante a fase final de elaboração desta tese.

A todos os amigos que me incentivaram na realização deste trabalho.

Aos professores da COPPE que muito contribuíram para a minha formação acadêmica: Álvaro L. G. A. Coutinho, Edison C. Prates de Lima, José A. F. Santiago, José C. F. Telles, Luis V. S. Sagrilo, Marcos C. Cavalcanti, Nelson F. F. Ebecken e Roberto F. de Oliveira.

Aos professores e professoras que dedicaram o seu precioso tempo para me ensinar. Agradeço especialmente à professora Maria da Graça Albino de Oliveira.

A todos os funcionários da COPPE que, de alguma forma, contribuíram para a realização deste trabalho. Agradeço especialmente à Estela, pela sua dedicação e pela alegria com que trabalha.

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Doutor em Ciências (D.Sc.).

ALGORITMOS GENÉTICOS DE AGRUPAMENTO PARA EXTRAÇÃO DE REGRAS DE REDES NEURAIS

Eduardo Raul Hruschka

Novembro/2001

Orientador: Nelson Francisco Favilla Ebecken

Programa: Engenharia Civil
Área Interdisciplinar de Computação de Alto Desempenho
Sub-área de Sistemas Computacionais

A principal limitação da utilização de redes neurais supervisionadas, em aplicações de mineração de dados (*data mining),* se deve à dificuldade do entendimento da representação do conhecimento incorporado pelos seus modelos. Para tentar superar esta limitação, faz-se uso de *Algoritmos de Extração de Regras de Redes Neurais.* Nesse sentido, o desenvolvimento de algoritmos eficazes para extração de regras de redes neurais, mais especificamente de perceptrons de múltiplas camadas, é uma tarefa importante e se constitui na principal contribuição do presente trabalho. Basicamente, descreve-se o desenvolvimento de uma metodologia, baseada em algoritmos genéticos de agrupamento, para extrair regras de perceptrons de múltiplas camadas treinados em problemas de classificação que envolvam atributos discretos e/ou contínuos.

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Doctor of Science (D.Sc.).

CLUSTERING GENETIC ALGORITHMS FOR RULE EXTRACTION FROM NEURAL NETWORKS

Eduardo Raul Hruschka

November/2001

Advisor: Nelson Francisco Favilla Ebecken

Department: Civil Engineering
High Performance Computation
Computer Systems

The main challenge to the use of supervised neural networks in data mining applications is to get explicit knowledge from these models. For this purpose, algorithms for rule extraction from neural networks are developed. This work presents two new algorithms that extract logical classification rules from supervised neural networks. Basically, a methodology based on clustering genetic algorithms is described and evaluated. The developed algorithms are designed to extract rules from multilayer perceptrons trained in classification problems that involve numeric and categorical attributes.

ÍNDICE

CAPÍTULO 1 - Introdução..... 1

1.1 Motivação..... 5

1.2 Revisão Bibliográfica..... 6

1.3 Objetivo.....21

1.4 Organização.....24

CAPÍTULO 2-Algoritmos Genéticos em Problemas de Agrupamento 25

2.1 Descrição e Complexidade do Problema 25

2.2 Aspectos da Aplicação de Algoritmos Genéticos 28

CAPÍTULO 3 - Algoritmo Genético de Agrupamento Tradicional. 32

3.1 Algoritmo Implementado - AGAT 33

3.1.1 Função Objetivo..... 35

3.1.2 Recombinação e Mutação..... 37

3.1.3 População Inicial..... 38

3.2 Estudo de Casos..... 39

3.2.1 Iris Plants Database …….....39

3.2.2 Pima Indians Diabetes Database..... 42

3.2.3 Australian Credit Approval Database 45

3.3 Comentários …… 49

CAPÍTULO 4 - Algoritmo Genético de Agrupamento Específico.....50

4.1 Codificação 50

4.2 Função Objetivo.....52

4.3 Seleção..... 53

4.4 Operadores 54

4.4.1 Recombinação.....54

4.4.2 Mutação.....56

4.5 População Inicial.....57

4.6 Vantagens do AGAE e Detalhes de Implementação 59

4.7 Exemplos Didáticos – Ruspini e Dados Aleatórios.....60
4.7.1 Eficácia do AGAE – grupos na base de dados completa.....61
4.7.1.1 Ruspini..... 61
4.7.1.2 Dados Aleatórios 62
4.7.2 Eficácia do AGAE em encontrar sub-grupos 63
4.7.2.1 Ruspini..... 63
a) Geração Inicial.....64
b) 200 gerações..... 65
4.7.2.2 Dados Aleatórios 67
a) Geração Inicial..... 67
b) 200 gerações..... 67
c) 1000 gerações..... 68
4.8 Estudo de Casos.....69
4.8.1 Iris Plants Database …….....71
4.8.1.1 Espaço Original dos atributos – áreas de sépalas e de pétalas 71
a) Base de Dados Completa 71
b) Classes Separadamente 73
4.8.1.2 Espaço das Ativações de uma Rede Neural..... 75
a) Base de Dados Completa 76
b) Classes Separadamente 78
4.8.2 Australian Credit Approval Database…….....….....79
4.8.2.1 Aplicando o AGAE no Espaço Original dos atributos..... 80
4.8.2.2 Aplicando o AGAE no Espaço das Ativações..... 83
4.8.3 Pima Indians Diabetes Database …….....86
4.8.3.1 Aplicando o AGAE na Base de Dados Original 87
a) Base de Dados Completa 87
b) Regras de Classificação.....88
4.8.3.2 Regras Obtidas por meio de uma Rede Neural 89
4.8.4 Wisconsin Breast Cancer Database…….....….....….....94
4.8.4.1 Aplicando o AGAE no Espaço Original dos atributos..... 94
4.8.4.2 Aplicando o AGAE no Espaço das Ativações..... 96

4.8.5 Dados Meteorológicos…….....…. 100
4.8.5.1 Aplicando o AGAE na Base Original de Treinamento..... 103

4.8.5.2 Extração de Regras na Base Original de Treinamento.....105
4.8.5.3 Treinamento de Redes Neurais Supervisionadas.....107
4.8.5.4 Aplicação do AGAE nas Ativações do Conjunto de Treinamento.109
4.8.5.5 Extração de Regras de Classificação no Espaço das Ativações 110
4.8.5.6 Codificando Sete Classes Diferentes 113

CAPÍTULO 5 - Conclusões.....118
5.1 Discussão dos Resultados Obtidos 120
5.2 Sugestões para Trabalhos Futuros 122

CAPÍTULO 6 – Referências Bibliográficas.....125

CAPÍTULO 1
Introdução

A tecnologia computacional tem evoluído de forma extraordinária. Nesse sentido, a velocidade de processamento e a capacidade de armazenamento têm aumentado de maneira insigne. Concomitantemente, ampliam-se as nossas capacidades de gerar, coletar e processar dados, propiciando a aplicação eficiente das tecnologias de informação emergentes.

A ampla utilização dos códigos de barras identificadores de produtos, a informatização das transações comerciais e os avanços nas ferramentas de coleta de dados têm proporcionado a utilização de grandes bancos de dados em muitas áreas da atividade humana. Paradoxalmente, esses avanços da tecnologia têm proporcionado um problema de super-abundância de dados, pois nossas capacidades de coletar e armazenar dados têm superado nossa habilidade de analisar e extrair conhecimento destes dados [1]. Nesse contexto, é necessária a aplicação de técnicas e ferramentas que transformem, de maneira *inteligente* e automática, os dados disponíveis em informações úteis, que representem *conhecimento*. Observa-se, então, que pesquisadores das mais variadas áreas têm encontrado um vasto campo de aplicação da tecnologia em uma área de pesquisa recente: *data mining.*

O termo *data mining* [2,3,4] surgiu do enorme interesse em se utilizar grandes bancos de dados de uma maneira inteligente, e sugere uma analogia interessante: explora-se uma mina de dados, purificando-se o minério para obter o ouro - *conhecimento*. Nesse sentido, o conhecimento se refere às informações e aos modelos, armazenados por uma pessoa ou máquina, utilizados para apropriadamente interpretar, prever e responder ao mundo exterior [5].

A idéia de descobrir conhecimento em grandes conjuntos de dados é muito interessante, e até intuitiva, mas tecnicamente é uma tarefa significativamente difícil e desafiante [6], requerendo ativa participação *de engenheiros de conhecimento*, analistas de sistemas, analistas de dados, especialistas do domínio, usuários do sistema, etc. É, portanto, uma tarefa interdisciplinar e multidisciplinar, centralizada na participação ativa do ser humano [6,7].

De um modo mais formal, pode-se dizer que *data mining* se refere a uma classe de métodos utilizados em alguns passos que abrangem o processo de descoberta de conhecimento em bancos de dados - *Knowledge Discovery in Databases - KDD.* Este termo, cunhado em 1989, se refere ao processo, interativo e iterativo, de descoberta de conhecimento em conjuntos de dados, incorporando conhecimento de domínio e interpretação de resultados, com ênfase na aplicação dos métodos de *data mining* [2]. O processo de descoberta de conhecimento em banco de dados é útil em vários domínios da ciência, bem como em várias aplicações práticas [8], dentre as quais se destacam aquelas que envolvem a tecnologia da informação.

Nesse contexto, os bancos de dados representam *depósitos de conhecimento* em potencial, que podem ser explorados para se descobrir relações, padrões e regras. Pode-se definir a descoberta de conhecimento em bancos de dados - KDD - como sendo o processo não trivial de identificação de padrões válidos, novos, potencialmente úteis e compreensíveis em dados [2].

Assim sendo, FAYYAD et al. [2] sugerem que o termo KDD se refere ao processo completo de descoberta de conhecimento em dados - incluindo conhecimento pré-adquirido e análise de resultados - enquanto que o termo d*ata mining* se refere somente à aplicação de algoritmos de reconhecimento de padrões em dados, não incluindo as atividades de pré-processamento e pós-processamento. Entretanto, alguns autores consideram que o termo *data mining* pode ser conceituado de outras maneiras.

BIGUS [3], por exemplo, define *data mining* como sendo a descoberta eficiente de informações valiosas e não óbvias de um grande conjunto de dados. A avaliação relativa ao tamanho do banco de dados é de caráter subjetivo. Genericamente, pode-se dizer que as aplicações de *data mining* se justificam quando a quantidade de dados oculta as valiosas relações presentes no conjunto de dados. O processo de *data mining* envolve basicamente três etapas [3]: a preparação dos dados - seleção, *purificação* e pré-processamento, a utilização de um algoritmo de processamento de dados e a análise das informações obtidas, objetivando-se a avaliação criteriosa do conhecimento obtido.

A maioria dos métodos de *data mining* é baseada em conceitos de aprendizado automático (*machine learning)*, reconhecimento de padrão e estatística, sendo utilizados, na maioria das aplicações, em tarefas de previsão e de descrição [2]. A previsão envolve a utilização de variáveis independentes conhecidas no prognóstico de variáveis dependentes. A descrição assume a descoberta de padrões interpretáveis que descrevem os dados. O aprendizado automático significa a realização de procedimentos

computacionais automáticos que aprendem uma determinada tarefa por meio de uma determinada série de exemplos [9].

Em se tratando de aplicações de *data mining*, constata-se [10] que o maior obstáculo à obtenção de conhecimento de alta qualidade se deve às limitações provenientes do conjunto de dados propriamente dito, pois os dados raramente são coletados especificamente para o processo de extração de conhecimento, mas normalmente como sendo um sub-produto de outras tarefas. Observa-se [11], ainda, que até mesmo nos casos em que se dispõe de um conjunto de dados adequado à aquisição de conhecimento, as tarefas de d*ata mining* apresentam dificuldades inerentes ao tamanho e à multidimensionalidade dos bancos de dados.

A viabilidade da aplicação das técnicas de *data mining* pode ser avaliada por meio de critérios práticos e técnicos [2]. O critério prático envolve, basicamente, a consideração do impacto de uma aplicação, ou seja, uma análise de custo / benefício, na qual o valor da informação obtida deve ser superior ao custo do processamento dos dados. Consideram-se, ainda, a disponibilidade de utilização de técnicas tradicionais, bem como de especialistas no domínio. Um aspecto que adquire importância fundamental diz respeito à segurança e privacidade das informações contidas no banco de dados. Em aplicações científicas, avalia-se fundamentalmente o potencial referente à originalidade e à qualidade do conhecimento adquirido. No tocante ao critério técnico, consideram-se a disponibilidade de dados em quantidade e qualidade necessárias e suficientes para realizar a aplicação. Cumpre observar, ainda, que a disponibilidade de conhecimento de domínio é, talvez, o fator mais importante para o sucesso de uma aplicação de *data mining* [2,6]. Devem ser consideradas, também, as implicações éticas nas aplicações que envolvem a utilização de informações pessoais.

Atualmente, pode-se constatar [2,3,4,7,12] muitas aplicações das técnicas de descoberta de conhecimento de banco de dados. Cumpre, entretanto, citar que as aplicações que realmente alcançam sucesso raramente estão à disposição do público, particularmente se o conhecimento adquirido pode ser utilizado como uma vantagem competitiva [7].

FAYYAD [2] descreve as técnicas mais populares de *data mining.* A abordagem utilizada neste trabalho trata da aquisição de conhecimento de bancos de dados por meio de técnicas *de Inteligência Computacional.* Mais especificamente, faz-se uso de redes neurais com aprendizado por retropropagação de erros (*backpropagation*) e de algoritmos genéticos de agrupamento. Conforme observa CARVALHO [12], o termo

rede neuronal deveria ser preferido ao termo *rede neural*. No entanto, este termo é mais usual e popular e, por este motivo, será utilizado no presente trabalho.

As redes neurais são baseadas na *metáfora cerebral*, sugerindo que a inteligência se manifesta por meio de um grande número de elementos de processamento (neurônios) inter-conectados em uma rede, na qual cada um destes elementos desempenha computações relativamente simples [13]. Nesse sentido, os modelos de redes neurais têm capacidade de aprender e de descobrir representações de conhecimento. HAYKIN [5] define a rede neural como sendo um processador, maciçamente paralelo e distribuído, que possui uma tendência natural para armazenar conhecimento, proveniente da experiência, tornando-o disponível para utilização. Os detalhes referentes ao processamento realizado por redes neurais podem ser encontrados em [5,13,14,15].

Basicamente, as redes neurais aprendem por meio da apresentação de exemplos, no sentido de que se utilizam de um conjunto de dados - muitas vezes complexos e ruidosos - para realizar suposições baseadas no aprendizado passado [3]. As aplicações de *data mining* podem ser realizadas quando se dispõe de *dados históricos*. Nesse contexto, as redes neurais surgem como uma tecnologia natural para estas aplicações, pois se constituem num modelo computacional baseado na habilidade de reconhecer padrões em dados. Além disso, as redes neurais solucionam problemas por meio de um aprendizado próprio e apresentam as seguintes vantagens computacionais [13] :

- A aquisição de conhecimento em condições de ruído e de incerteza - falta de informação adequada e correta para tomar decisões - é satisfatória, permitindo uma capacidade adequada de generalização e de extração das propriedades estatísticas dos dados;
- Possuem uma representação de conhecimento flexível, criando suas próprias representações por meio de uma organização própria;
- São tolerantes às falhas, na medida em que a representação distribuída do conhecimento diminui a possibilidade de erros;
- Proporcionam um processamento eficiente do conhecimento. Conquanto o treinamento consome uma parcela considerável de tempo, observa-se que a utilização de uma rede neural treinada é eficiente.

Redes neurais *não supervisionadas* são, ainda, as mais utilizadas nas aplicações de *data mining* [3,16]. O aprendizado não supervisionado é utilizado quando se deseja agrupar um conjunto de dados, de modo que dados similares pertençam ao mesmo grupo. Este paradigma apresenta menor complexidade computacional e menor precisão em relação ao paradigma supervisionado, e é muito prático nos casos em que a disponibilidade de informação em tempo real, bem como a velocidade de aprendizado são características importantes [13]. Redes neurais de *Kohonen* [3,5,13] são as mais conhecidas, sendo muito aplicadas em problemas que envolvam algum tipo de segmentação ou agrupamento. A metodologia de *Kohonen* é uma técnica de aprendizado competitivo [17], cuja idéia básica é tal que cada unidade de saída compete pela *posse* dos exemplos de treinamento [16], sendo semelhantes aos métodos estatísticos de agrupamento conhecidos por *k-means algorithms* [18,19].

O aprendizado supervisionado, por sua vez, também é muito usado em aplicações de *data mining*. Neste, assume-se a disponibilidade de um banco de dados que contenha variáveis independentes e dependentes. Nesse contexto, as redes neurais de retropropagação são muito usadas em problemas de classificação e na previsão de séries temporais. Genericamente, pode-se dizer que a aplicação de modelos de redes neurais supervisionadas apresenta duas desvantagens principais: grande tempo de processamento na fase de definição do modelo - treinamento da rede neural - e falta de compreensibilidade. O tempo de processamento pode ser diminuído por meio da utilização de processamento paralelo e distribuído. O conhecimento de domínio também pode permitir a diminuição da complexidade da rede neural que, conseqüentemente, diminuirá o tempo de treinamento. A falta de compreensibilidade se constitui, pois, no principal obstáculo à utilização de redes neurais em aplicações de *data mining*. Basicamente, o presente trabalho trata da aplicação de métodos que permitam compreender o conhecimento obtido pelos modelos de redes neurais supervisionadas – mais especificamente de perceptrons de múltiplas camadas – tornando-os mais atrativos para as aplicações de *data mining*.

1.1 Motivação

Redes neurais supervisionadas têm sido utilizadas em muitas aplicações. A principal limitação da utilização dessas redes em aplicações de *data mining* se deve à dificuldade de entendimento da representação do conhecimento incorporado pelos seus modelos.

Para tentar superar esta limitação, faz-se uso de *Algoritmos de Extração de Regras de Redes Neurais*. Nesse sentido, o desenvolvimento de algoritmos eficazes de extração de regras de redes neurais, mais especificamente de perceptrons de múltiplas camadas, é uma tarefa importante e se constitui na principal motivação do presente trabalho.

Perceptrons de Múltiplas Camadas (PMC) ajustam seus parâmetros internos por meio de mapeamentos de vetores de entrada em vetores de saída. Conquanto os PMC possam alcançar altas taxas de classificação corretas, o conhecimento adquirido por tais redes neurais é normalmente incompreensível aos humanos [20]. Este fato pode ser um obstáculo importante em tarefas de *data mining,* nas quais é importante obter regras simbólicas ou outras formas de *estruturas de conhecimento* [21]. Assim, muitos métodos tem sido desenvolvidos para obter conhecimento explícito destes modelos.

Fundamentalmente, o conhecimento adquirido por um modelo de rede neural é codificado nos pesos das conexões [13]. Estes, por sua vez, determinam os valores das funções de ativação. Desta forma, o processo de aquisição de conhecimento de redes neurais implica no uso de algoritmos baseados nos pesos das conexões ou nos valores das ativações das unidades ocultas, que são consideradas detectoras de características. Os algoritmos utilizados para obter conhecimento de redes neurais são normalmente denominados *algoritmos para extração de regras de redes neurais*. A seção 1.2 fornece um referencial teórico sobre estes algoritmos, enquanto que a seção 1.3 define o objetivo desse trabalho e exprime, em termos gerais, a metodologia sugerida para extrair regras de perceptrons de múltiplas camadas.

1.2 Revisão Bibliográfica

Vários trabalhos sobre extração de regras de redes neurais tem sido publicados nos últimos anos, evidenciando a crescente importância desse assunto nos meios científicos. Nesse contexto, a presente seção pretende citar, da maneira mais sucinta possível, as principais metodologias desenvolvidas para extração de regras de redes neurais, fornecendo um referencial teórico sobre este assunto. Sempre que possível, procurar-se-á respeitar a ordem cronológica em que os diversos trabalhos foram publicados. Além disso, a citação de trabalhos *derivados* será realizada, preferencialmente, de maneira a complementar as publicações *originais*. Uma descrição detalhada dos principais métodos para extração de regras de redes neurais desenvolvidos entre 1988 e 1996 pode ser encontrada em [22].

Em 1988, GALLANT [23] desenvolveu a primeira abordagem para entender o conhecimento armazenado em redes neurais. Para tal, configura-se a rede neural de modo que cada nó represente uma entidade conceitual. Por exemplo, as unidades de entrada poderiam representar sintomas, as unidades ocultas as doenças e as unidades de saída representariam o tratamento. A tarefa da rede neural se resume no aprendizado das relações entre estas entidades conceituais. A extração de regras é realizada através da interpretação dos pesos como sendo informações correlacionais entre as entidades conceituais. Esta abordagem propicia o entendimento da rede mas requer a disponibilidade de suficiente conhecimento do domínio [24]. BARON [25] observa que o método apresenta a inconveniente necessidade de se atribuir critérios subjetivos de escolha de regras, haja vista a possível ocorrência de regras redundantes. Além do trabalho pioneiro sobre extração de regras de redes neurais, GALLANT desenvolve os conceitos que relacionam o aprendizado de redes neurais com os sistemas especialistas em [26].

Em 1992, McMILLAN et al. [27] descreveram uma rede neural, chamada *RuleNet*, que aprende regras simbólicas em um domínio de manipulação de vetores. Os autores também mostraram como a arquitetura desenvolvida pode ser aplicada ao processamento de linguagem natural.

Em 1993, TOWELL e SHAVLIK [28] publicaram um trabalho que torna possível a utilização de redes neurais num sistema de refinamento de regras. Primeiramente, traduz-se um conjunto de regras em uma rede neural. Para refinar estas regras, treina-se a rede neural em determinado conjunto de dados até alcançar a convergência desejada, seguindo-se, então, a extração das regras refinadas.

A metodologia se baseia na análise dos pesos. Para tal, considera-se que o treinamento origina neurônios ativos (valor de ativação unitário) ou inativos (valor de ativação nulo). Assim, busca-se pelo conjunto de pesos que permite superar o valor do *bias* de cada neurônio, tornando-o, portanto, ativo. Todos os métodos de extração de regras conhecidos até 1992 se baseiam nesta metodologia [28]. Como as ativações das unidades pertencem ao conjunto $\{0,1\}$, a busca pelas regras é simplificada. Este método é intitulado de Algoritmo SUBSET [28] e, segundo os autores, representa o estado da arte até 1991. O maior problema do método reside na utilização de redes neurais de grande complexidade, quando o custo relativo à busca do conjunto de regras é elevado. Este fato restringe a sua utilização em problemas de baixa complexidade.

Os autores também desenvolveram outro algoritmo, intitulado de MofN [28], que é uma variante do algoritmo SUBSET, e observam que as regras individuais geradas pelo Algoritmo SUBSET são mais compreensíveis do que as geradas pelo MofN. Entretanto, o conjunto de regras gerado pelo MofN propicia melhor entendimento, pois o SUBSET gera regras repetitivas. Em termos de precisão, o MofN apresenta resultados próximos aos da rede neural, enquanto que o SUBSET apresenta resultados significativamente piores. O algoritmo MofN requer redes neurais baseadas em conhecimento, isto é, inicialmente compreensíveis, bem como a manutenção dos conceitos representados pelas unidades. Assim sendo, a metodologia está restrita aos domínios cujo conhecimento possa ser traduzido em regras adequadas à codificação de uma rede neural. Observa-se [28] que a distribuição da informação durante o treinamento compromete a qualidade das regras extraídas. OLIVEIRA et al. [29] desenvolveram um algoritmo para realizar a extração de regras de redes neurais baseado na mesma premissa básica utilizada em [28]. Os autores [29] observam que a abordagem sugerida, além de mais simples e direta, é mais eficiente, representando, pois, um avanço na aplicação do método desenvolvido em [28].

THRUN [30] descreve uma abordagem para a extração de regras de redes neurais baseada na análise do intervalo de validade -*Validity Interval Analysis* (VIA). VIA é uma ferramenta genérica utilizada para a aquisição de conhecimento de redes neurais de retropropagação. Esta técnica é realizada por meio da propagação de intervalos de ativações através da rede neural na direção da entrada para a saída e vice-versa. Na extração de regras esses intervalos são utilizados para aprovar ou não a correção das regras. O procedimento não requer arquiteturas específicas de redes neurais nem procedimentos específicos de treinamento.

LI MIN FU [31] apresentou uma metodologia para o desenvolvimento de Sistemas Inteligentes Híbridos, ou seja, sistemas que combinam inferências conexionistas e simbólicas. Segundo o autor, a principal linha de pesquisa em Sistemas Inteligentes Híbridos diz respeito ao estudo das redes neurais baseadas em conhecimento. A arquitetura inicial deste tipo de rede é determinada pelo prévio conhecimento do domínio. A integração da teoria do domínio (regras) e do aprendizado empírico (dados) é importante para a construção de sistemas de aprendizado, pois a teoria nem sempre é perfeita e os dados, por sua vez, nem sempre são confiáveis ou adequados.

Nesse contexto, o autor apresenta um modelo de rede neural baseado em conhecimento em que os atributos e os conceitos inerentes ao domínio são primeiramente identificados e relacionados de maneira consistente com o conhecimento inicial. Feito isto, as conexões da rede neural são ponderadas adequadamente de modo a se manter a semântica. Unidades ocultas e conexões adicionais podem ser incorporadas à rede neural quando assim se fizer necessário. Busca-se, então, a minimização do erro, por meio de um conjunto de dados, deste modelo de rede neural primitivo. Após esta etapa, pode-se traduzir o conhecimento, agora revisto, para a forma de regras explícitas. Resumidamente, pode-se revisar empiricamente (por meio de um conjunto de dados) uma determinada teoria (apresentada em forma de regras). Esta revisão é realizada por meio de uma rede neural, sendo, então, decodificada para a forma simbólica. Integra-se, assim, a teoria do domínio, que pode ser imperfeita ou aproximada à rede neural que, por meio de dados, refina o conhecimento empiricamente. A tradução da rede neural revista em regras, é realizada através do algoritmo denominado de *KT Algorithm* [32]. Observa-se que a metodologia proposta por FU [31] é útil às aplicações de d*ata mining* nos casos em que se dispõe de algum conhecimento prévio do domínio.

A metodologia para extração de regras de modelos de redes neurais desenvolvida por KANE e MILGRAM [33], em 1994, se baseia na definição de unidades numéricas e lógico-numéricas. Observa-se [22] que esta metodologia requer pouco processamento após o treinamento da rede neural, propiciando a obtenção de regras não redundantes, mas nem sempre claras.

CRAVEN e SHAVLIK [34] exploraram o fato de que redes neurais treinadas podem ser eficazmente *questionadas*, isto é, podem ser utilizadas para responder perguntas sobre as situações em que exemplos específicos são englobados por determinada classe. Desse modo, o processo de extração de regras pode ser visualizado como uma tarefa de aprendizado, cujo objetivo se resume em aprender a função computada pela rede neural após o treinamento. A metodologia propicia a extração de regras *conjuntivas* e MofN [30].

SETHI e YOO [35] propuseram um procedimento para converter os pesos dos neurônios em representações simbólicas, e demonstraram o uso do método desenvolvido no entendimento e na aproximação de redes neurais.

ALEXANDER e MOZER [36] também desenvolveram um método para extração de regras baseado nos valores dos pesos. Os autores consideram redes neurais utilizando

funções do tipo sigmóide, assumindo que estas exibam ativações com comportamentos aproximadamente Booleanos.

Em 1995, ANDREWS et al. [37] estudaram os principais trabalhos sobre os procedimentos e algoritmos utilizados para inserir conhecimento em redes neurais, extrair regras dos seus modelos e refinar bases de regras existentes por meio de redes neurais. Nesse sentido, os autores descrevem e analisam detalhadamente os principais trabalhos sobre extração de regras de redes neurais publicados até 1995. Além disso, os autores desenvolveram uma taxonomia para classificar os vários algoritmos de extração de regras de redes neurais. O método de classificação proposto considera a forma e a qualidade das regras extraídas, o emprego de algoritmos de treinamento específicos (que caracteriza a portabilidade de cada método de extração de regras), a complexidade do algoritmo de extração de regras e a translucidez (que revela a relação entre as regras extraídas e a arquitetura da rede neural). Neste aspecto, as técnicas de extração de regras são classificadas como *decomposicionais*, *pedagógicas* e *ecléticas*. A abordagem *decomposicional* envolve os métodos nos quais as unidades das camadas ocultas e de saída são mapeadas de forma binária, permitindo a análise dos pesos que superam o *bias* de determinado neurônio. Assim, cada neurônio das camadas ocultas e de saída pode ser interpretado por meio de uma regra. A abordagem *pedagógica* supõe a extração de regras que relacionem as entradas da rede neural diretamente com as suas saídas, considerando que este processo envolve uma tarefa de aprendizado. O método proposto no presente trabalho utiliza o conhecimento da arquitetura da rede neural (expressões de ativação dos neurônios da primeira camada oculta) para complementar os resultados de um algoritmo de *agrupamento* que é aplicado aos valores das ativações destes neurônios. Nesse sentido, o método proposto no presente trabalho pode ser denominado de *eclético* [37].

NARAZAKI et al. [24,38] propuseram um método para reorganizar o conhecimento distribuído na rede neural e extrair regras de classificação aproximadas, mais especificamente regras *fuzzy* [39,40]. A dificuldade relativa à interpretação e entendimento do conhecimento adquirido por meio de redes neurais provém da natureza distribuída de representação do conhecimento [38]. A metodologia desenvolvida pelos autores é baseada na análise da função incorporada pela rede neural após o treinamento, mais do que na interpretação direta dos pesos das conexões.

Em 1996, OMLIN e GILES [41] desenvolveram um método para inserir conhecimento prévio (regras) em redes neurais recorrentes, mostrando que estas são capazes de revisar as regras inseridas.

ANDREWS e GEVA [42,43] publicaram um trabalho em que descrevem algumas técnicas de extração e de refinamento de regras desenvolvidas especificamente para redes neurais com camadas ocultas formadas por neurônios que produzem respostas locais. O artigo também aborda as vantagens e desvantagens destas redes neurais.

CECHIN et al. [44] descrevem um algoritmo para extrair regras *fuzzy* de redes neurais. As principais vantagens do método são: a possibilidade de usar um sistema de classificação *fuzzy*, a seleção dos atributos mais importantes, o cálculo da influência das unidades de entrada nas unidades de saída e a descoberta de neurônios que possam ser podados.

DIEDERICH et al. [45] desenvolveram um método de inicialização e refinamento de regras em redes neurais conectadas de forma esparsa. Os autores examinam o método em uma aplicação de linguagem natural.

DECLOEDT et al. [46] apresentaram um sistema para explicitação do conhecimento adquirido por redes neurais. Para tal, propuseram um sistema de inteligência artificial híbrido (neuro-simbólico) que tenta representar somente a parte mais significativa do conhecimento adquirido pela rede neural, baseando-se na incrementalidade do método de aprendizado e na simplificação do modelo.

FRÄMLING [47] propôs um método para explicar os resultados obtidos por redes neurais sem *transformar* o conhecimento adquirido durante o treinamento em regras. O método se baseia na importância e na utilidade de uma determinada unidade de entrada da rede neural para as unidades de saída.

GOLEA [48] tratou sobre a complexidade inerente à tarefa de extração de regras de redes neurais e observou que esse problema é geralmente difícil. Nesse sentido, o autor observa que existem casos em que qualquer método de extração de regras falhará, de modo que o desafio será encontrar uma sub-classe de redes neurais ou uma sub-classe de regras em que o problema de extração possa ser resolvido eficientemente.

HOWES e CROOK [49] desenvolveram um algoritmo para extração de regras de redes neurais treinadas com valores de entrada contínuos. O método usado é baseado naquele desenvolvido em [23].

JOHNSON et al. [50] apresentaram um método para descobrir o conhecimento adquirido por uma rede neural através de *questionamentos* feitos à própria rede neural. O método usa informações obtidas do conjunto de treinamento e as *respostas* da rede neural para criar um conjunto de relações de relevância entre as variáveis de entrada e de saída da rede neural. As regras são obtidas a partir dessas relações de relevância.

SALOMON [51] propôs um novo procedimento de inicialização, que acelera o aprendizado, tornando-o quase insensível ao número de unidades ocultas (desde que o número de conexões seja menor do que o número de padrões de entrada) e melhora a generalização. O método desenvolvido interpreta as amostras de treinamento como sendo regras, i.e. supõe-se que a rede mapeie a parte condicional da regra na saída especificada. Mais do que propor um método para extração de regras, o artigo propõe a indução de regras anteriormente ao treinamento.

TICKLE et al. [52] desenvolveram um método para extração de regras denominado de DEDEC que basicamente identifica as dependências funcionais entre as variáveis de entrada e de saída. O método é baseado nos vetores dos pesos de uma rede neural previamente treinada.

TSUKIMOTO e MORITA [53] propuseram um método para extrair funções Booleanas de redes neurais previamente treinadas. O método pode ser aplicado a vários modelos de redes neurais (perceptrons de múltiplas camadas, redes neurais recorrentes, redes de Hopfield e máquinas de Boltzmann).

ULTSCH et al. [54] desenvolveram um algoritmo que extrai regras de mapas de Kohonen treinados para classificar dados multidimensionais. O algoritmo seleciona atributos significantes e constrói condições apropriadas para que estes caracterizem cada classe envolvida.

BOLOGNA [55] desenvolveu um modelo específico de perceptron de múltiplas camadas que permite a extração de regras simbólicas até mesmo nos casos em que os atributos de entrada são contínuos. Os resultados da aplicação do algoritmo em três problemas de classificação são comparados àqueles obtidos pelo algoritmo C4.5.

SETHI e YOO [56] desenvolveram uma metodologia para extração de regras de redes neurais baseada na análise dos valores dos pesos das conexões. Esta metodologia também se aplica às redes neurais cujas entradas não são binárias e não depende do conjunto de treinamento.

FILER et al. [57] realizaram um estudo comparativo de duas abordagens usadas para extrair regras de redes neurais treinadas em dados contínuos em duas bases de

dados. Maiores detalhes sobre os métodos de extração de regras, bem como sobre as bases de dados utilizadas, também podem ser encontrados em [56,58].

HAYWARD e DIEDERICH [59] mostraram que o método da generalização baseada na explicação pode ser aplicado com sucesso aos modelos conexionistas estruturados.

KRISHNAN [60] desenvolveu um método sistemático para gerar combinações de pesos que tornam determinado neurônio ativo, a fim de se extrair regras baseadas nestas combinações. O autor considera somente redes neurais com entradas Booleanas, mas explica como o algoritmo desenvolvido pode ser adaptado para extrair regras de redes neurais com entradas contínuas.

LeBLANC et al. [61] exploraram alguns trabalhos que tratam da combinação de sistemas especialistas com redes neurais - as chamadas redes especialistas. Os autores argumentam que estas proporcionam uma estrutura para a incorporação de vários tipos de raciocínio em um único sistema.

NEDJARI [62] desenvolveu um método de extração de regras de redes neurais baseado na parametrização de regiões do espaço dos pesos por meio do critério da Informação Mútua. O método não necessita de modelos de redes neurais especialmente treinados para a tarefa de extração de regras. Além disso, o método pode ser aprimorado para otimizar a arquitetura de redes neurais.

SAITO e NAKANO [63] propuseram uma abordagem conexionista para descobrir leis numéricas. Nesse sentido, os autores desenvolveram um método para descrever a função incorporada pelo modelo de rede neural.

WILES e BOLLAND [64] desenvolveram um estudo sobre o modo em que os espaços de estados de redes neurais recorrentes são visualizados, e seus inerentes problemas, no contexto das tarefas de extração de regras.

GILES et al. [65] consideraram o problema de se prever séries temporais em que há presença de ruído nos dados. Um modelo de rede neural híbrido que permite a extração de regras é desenvolvido e avaliado na tarefa de previsão de taxas de câmbio.

VISSER et al. [66] descrevem a aplicação de um conjunto de técnicas de extração de conhecimento [67,68,69], em forma de regras e incorporado por redes neurais em uma aplicação real, na qual se procura determinar os herbicidas a serem utilizados para a proteção de alguns cereais.

CRAVEN e SHAVLIK [70] desenvolveram um algoritmo genérico - que não requer arquiteturas nem algoritmos de aprendizado específicos - para extrair

representações simbólicas compreensíveis de redes neurais treinadas, intitulado de TREPAN. O algoritmo se utiliza de questionamentos feitos à rede neural, que assume a função do oráculo, para induzir uma árvore de decisão [71] que aproxima o conceito representado pela rede neural. Assim, o *oráculo* - rede neural - é utilizado para determinar as classes referentes aos exemplos de treinamento, para selecionar critérios de divisão - *splits* - para cada nó interno da árvore de classificação e para verificar a abrangência de cada nó, ou seja, as classes descritas por cada nó. O algoritmo se utiliza de expressões Booleanas para realizar as divisões. A seleção dos critérios de divisão é realizada por meio do critério da taxa de ganho que avalia as divisões candidatas. A principal vantagem apresentada pelo algoritmo TREPAN diz respeito à generalidade, não requerendo arquiteturas e algoritmos de treinamento específicos nem tampouco que o modelo seja uma rede neural [16].

SHAVLIK [72] analisou os trabalhos sobre redes neurais baseadas em conhecimento desenvolvidos na Universidade de Wisconsin entre 1989 e 1996.

HERRMANN [73] desenvolveu um algoritmo de treinamento de *redes neurais lógicas*, nas quais as unidades realizam as funções de uma fórmula lógica. Utiliza-se o algoritmo de retropropagação para treinar tais redes, bem como um algoritmo para podar as conexões redundantes. O autor argumenta que as regras obtidas podem ser diretamente utilizadas em um sistema especialista, de tal modo que as mesmas possam realizar a mesma tarefa desempenhada pela rede neural.

TAHA, I. e GHOSH, J. [74] desenvolveram um algoritmo de ordenamento e de avaliação de regras extraídas de modelos de redes neurais. No concernente à tarefa de extração de regras, os autores desenvolveram três algoritmos denominados BIO-RE (Binarized Input-Output Rule Extraction), Partial-RE (Parcial Rule Extraction) e Full-RE (Full Rule Extraction). A metodologia para avaliação de um conjunto de regras está adequadamente descrita em [75], enquanto que os algoritmos de extração de regras estão adequadamente descritos em [76].

Basicamente, o algoritmo BIO-RE extrai regras binárias e é adequado quando as variáveis de entrada e de saída são naturalmente binárias, ou quando a *binarização* não diminui sensivelmente o desempenho do treinamento. O algoritmo PARTIAL-RE é baseado nos valores dos pesos das conexões. Estes são classificados de maneira que se possa procurar pelas combinações que ocasionam a ativação de determinada unidade oculta ou de saída. Combinando estes agrupamentos de pesos, é possível extrair regras do modelo de rede neural. Cumpre citar, ainda, que esta metodologia proporciona a

extração de regras parciais, que representam o conhecimento mais importante codificado no modelo de rede neural. Os autores afirmam que esta metodologia pode ser utilizada eficientemente nos casos em que se objetiva, principalmente, estudar os principais parâmetros que influenciam as saídas da rede.

Em [77], os autores descrevem a utilização de um sistema híbrido, combinando redes neurais e sistemas especialistas, que faz uso de um algoritmo semelhante ao PARTIAL-RE para o controle de reservatórios de água. O algoritmo FULL-RE é similar ao PARTIAL-RE, mas proporciona a extração integral do conhecimento codificado na rede neural. Para os casos em que as entradas representam valores contínuos, a metodologia utiliza o *Algoritmo Chi2* [78], discretizando-se as entradas para a obtenção das regras.

LU et al. [79] desenvolveram uma metodologia para extração de regras de redes neurais supervisionadas baseada nos valores das ativações das unidades ocultas. As ativações se referem à soma, ponderada pelos pesos, das entradas da unidade [80]. Primeiramente, realiza-se o treinamento da rede neural, de modo que se obtenha a taxa de classificação correta desejada. Eliminam-se, então, as conexões redundantes, seguindo-se uma análise dos valores das ativações para a obtenção de regras baseadas nestes valores. Basicamente, pode-se resumir o algoritmo de extração de regras, denominado de **RX** [79], nos seguintes passos:

1) Processar os eventos do conjunto de treinamento através da rede computando-se os valores de ativação das unidades ocultas;
2) Aplicar um algoritmo de *agrupamento* a estes valores;
3) Enumerar os valores de ativação discretizados e computar os respectivos valores de saída da rede. Gerar regras que descrevam as saídas da rede em termos dos valores de ativação discretizados em 2);
4) Para cada unidade oculta, enumerar os valores de entrada que conduzem aos respectivos valores de ativação das unidades ocultas, gerando regras que descrevam os valores discretizados das unidades ocultas em termos das entradas da rede neural.
5) Combinar as regras obtidas em 3) e 4) para obter as regras que relacionam as variáveis dependentes às variáveis independentes.

O primeiro passo do algoritmo **RX** agrupa os valores de ativação das unidades ocultas sem o comprometimento da precisão da rede neural. Após a execução do algoritmo, obtém-se um conjunto de valores de ativação para cada unidade oculta. O algoritmo de agrupamento é aplicado em cada unidade oculta separadamente. Primeiramente, classificam-se os valores de ativação em ordem crescente. Segue-se, então, a procura pelos intervalos correspondentes às ativações obtidas para cada classe, de maneira que os agrupamentos não gerem informações redundantes, isto é, intervalos de ativações iguais gerando classes diferentes. Considerando-se esta metodologia, obtém-se regras da seguinte forma:

Se $(a_1\ \theta\ v_1)$ e $(a_2\ \theta\ v_2)$ e $(a_3\ \theta\ v_3)$ e ... e $(a_n\ \theta\ v_n)$ então **C**;

Neste modelo de regra, os a_i´s {i=1,...,n} representam os atributos – unidades ocultas da rede neural, θ pertence ao conjunto $\{=, \leq, \geq, <, >, \neq\}$ os v_i´s {i=1,...,n} representam constantes e **C** representa a classe envolvida.

Este algoritmo serviu de base para a implementação do *Algoritmo RX Modificado* [22,81,82,83] que representa o nosso primeiro trabalho sobre extração de regras de redes neurais. Basicamente, as modificações realizadas no Algoritmo RX são baseadas no fato de que cada classe possui um conjunto próprio de ativações, o que sugere uma extração de regras particular para cada classe. Definido o modelo de rede neural, computam-se as ativações para cada classe, por meio da separação dos exemplos de treinamento nas diferentes classes. Além disso, esta abordagem simplifica o algoritmo, eliminando os passos 3) e 4) do *Algoritmo RX*.

Em princípio, poder-se-ia supor que o *Algoritmo RX* somente propiciaria resultados adequados nos casos em que a informação se encontrasse representada localmente. Assim, optou-se pela utilização de um algoritmo de agrupamento diferenciado em relação ao exposto em [79], de modo que os agrupamentos obtidos fossem independentes do *ponto de partida*. Mais especificamente, desejava-se que os valores das ativações pertencentes aos maiores agrupamentos estivessem próximos ao valor de ativação médio da unidade. Nesses casos, poder-se-ia supor que os valores das ativações poderiam ser parametrizados estatisticamente, de modo que valores extremos corresponderiam a exemplos ruidosos. Maiores detalhes sobre o algoritmo

implementado, bem como alguns resultados experimentais, podem ser encontrados em [22,81,82,83].

Cumpre, ainda, citar que o Algoritmo RX Modificado se constituiu no *ponto de partida* para este trabalho, haja vista que a metodologia aqui desenvolvida é semelhante àquela usada pelo Algoritmo RX Modificado. Nesse sentido, os dois algoritmos desenvolvidos neste trabalho também se baseiam no processo de *agrupamento* para extrair regras de redes neurais mas diferem-se em dois pontos principais: (i) os algoritmos de *agrupamento* desenvolvidos são aplicados no espaço das unidades ocultas e não em cada uma delas separadamente; (ii) o processo de *agrupamento* é baseado em algoritmos genéticos e estes fornecem um meio de se otimizar os grupos tanto em relação à quantidade quanto em relação às suas estruturas.

SANTOS et al. [84], também desenvolveram um método para extrair regras de redes neurais baseado no Algoritmo RX [79], utilizando arquiteturas de redes neurais otimizadas por meio de algoritmos genéticos. Desta forma, os autores argumentam que é possível se obter um conjunto de regras com maior taxa de acerto e com menor complexidade.

Em 1997, SETIONO [85] descreveu uma metodologia que envolve o Algoritmo RX [79] e um algoritmo de eliminação de conexões redundantes. Neste aspecto, se o número de entradas conectadas a uma unidade oculta é suficientemente pequeno, então geram-se regras através do Algoritmo RX. Caso contrário, a unidade oculta é *dividida* e tratada como sendo uma unidade de saída. Insere-se, então, uma nova camada oculta, formando uma nova *sub-rede*, que, por sua vez, é treinada e podada. Este processo é repetido até que haja um número relativamente pequeno de unidades de entrada conectadas às unidades ocultas. Quando existem muitas entradas conectadas a uma determinada unidade oculta, as ativações desta unidade são discretizadas e utilizadas como sendo os valores alvo de saída dessas unidades de saída da sub-rede. O processo de poda proporciona a seleção dos atributos importantes à modelagem.

DUCH et al. [86] desenvolveram dois algoritmos para extração de regras de redes neurais. Estes algoritmos impõem restrições à estrutura da rede neural de retropropagação por meio da adição de termos de regularização à função de erro. Assim, redes com poucas conexões são obtidas, favorecendo a obtenção de poucas regras lógicas. Os autores comparam os resultados obtidos em duas bases de dados de classificação.

DAS e MOZER [87] desenvolveram um algoritmo de aprendizado que favorece a formação de uma representação discreta, mais adequada à extração de regras. Sob este ponto de vista, tem-se, como conseqüência de um treinamento adequado, valores de ativação formando diferentes *grupos* para cada classe. A metodologia é desenvolvida para aplicação em redes recorrentes, mas os autores argumentam que a mesma pode ser facilmente aplicada às redes não recorrentes.

BENÍTEZ et al. [88] desenvolveram uma metodologia para interpretar modelos de redes neurais. Os autores estabelecem uma relação de igualdade entre os *perceptrons de três camadas* e um sistema baseado em regras *fuzzy*. Desta maneira, torna-se possível interpretar o conhecimento adquirido por modelos de redes neurais por meio de regras *fuzzy*. Os autores afirmam que tanto o modelo quanto as regras proporcionam medidas de precisão idênticas.

Essa metodologia não proporciona simplesmente um conjunto de regras que descreve o modelo de rede neural obtido, nem tampouco explicita a função incorporada pela rede neural. O conhecimento adquirido é descrito através de um conjunto de regras e de uma função, que pondera as pertinências (força) de cada regra para um exemplo do conjunto de dados considerado. O número de regras é igual ao número de unidades ocultas e o número de premissas de cada regra é igual ao número de unidades de entrada. A função obtida depende dos *valores de pertinência* (força) de cada regra. Nesse contexto, admite-se somente uma unidade de saída para o modelo de rede neural, o que de certa forma restringe a sua aplicação, haja vista que alguns problemas exigem arquiteturas maiores. A metodologia não proporciona uma estrutura de conhecimento adequada às aplicações de *Data Mining*, pois não se obtém um conjunto de regras que descreve diretamente o conjunto de dados, sendo mais útil às aplicações de sistemas especialistas baseados em redes neurais.

SETIONO e LIU [89] apresentaram um sistema para extrair regras de decisão oblíquas de redes neurais treinadas em problemas de classificação. Nesse sentido, cada regra corresponde a um particionamento do espaço dos atributos que não é necessariamente paralelo ao eixo do referido atributo. Os resultados experimentais obtidos pelos autores mostram que o sistema é eficaz na extração de regras compactas e compreensíveis e, ainda, que estas regras apresentam elevada precisão. O trabalho é desenvolvido considerando-se somente redes neurais sem retro-alimentação, com uma única camada oculta.

Para se realizar a extração de regras, agrupam-se os valores das ativações das unidades ocultas por meio do algoritmo Chi2 [78]. O algoritmo não possui restrições quanto a magnitude dos pesos, bem como em relação aos valores das ativações e dos *bias*. Os autores argumentam que o *Neurolinear* [89] apresenta bons resultados em problemas cujos atributos não precisam ser discretizados. As regras geradas por este sistema preservam a precisão verificada no modelo de rede neural.

Em 1998, WEIJTERS et al. [90] desenvolveram um algoritmo - BP-SOM - de aprendizado para redes neurais supervisionadas que permite a descoberta de padrões de ativação para as unidades ocultas, restringindo suas ativações a conjuntos de valores discretos. Concomitantemente ao treinamento da rede neural supervisionada, treina-se uma rede de Kohonen, que agrupa as ativações das unidades ocultas. A metodologia proporciona a extração automática de regras, por meio de sua habilidade de organizar o conjunto de dados em subconjuntos de exemplos importantes à tarefa de classificação.

DUCH et al. [91] desenvolveram um método para extrair regras fuzzy de redes neurais de retropropagação em que os valores dos pesos pertencem ao conjunto {-1,0,+1}. Além disso, os autores propuseram um algoritmo de treinamento em que a maioria dos pesos é forçada a assumir o valor nulo, de maneira que um número mínimo de regras é obtido por meio da análise dos pesos. Os autores também descrevem os resultados da aplicação do algoritmo em três bases de dados.

BARROS et al. [92] desenvolveram uma abordagem para extrair e modelar o conhecimento adquirido por redes neurais artificiais Booleanas, partindo de experimentos práticos e de análises teóricas.

NOBRE et al. [93] compararam as regras obtidas de redes neurais artificiais usando o algoritmo TREPAN com algoritmo CN2. As comparações envolvem o número de regras geradas, o número de literais e a precisão das regras em uma base de dados do *jogo da velha*.

OSÓRIO e DECLOEDT [94] desenvolveram um método para extração de regras de redes neurais similares àquelas baseadas em conhecimento. Os autores utilizam um algoritmo de extração incremental de conhecimentos que é associado à construção das redes neurais.

LUDERMIR [95] descreveu um método para extrair regras de redes neurais Booleanas, argumentando que a extração de regras nestas redes é mais viável e natural do que em outros modelos de redes neurais. O método permite entender como as redes

neurais obtêm as suas soluções nos problemas em que são usadas. Além disso, a autora argumenta que o método de extração de regras pode ser útil em sistemas especialistas.

Em 1999, DUCH et al. [96] apresentaram uma nova metodologia para extração de regras lógicas. A abordagem híbrida utilizada considera um algoritmo em que as regras são geradas por meio da adição de neurônios a perceptrons de múltiplas camadas. As regras são, então, otimizadas por meio de técnicas de minimização global. O desempenho do algoritmo foi avaliado em várias bases de dados. Além disso, em [97] os autores desenvolveram um novo método de otimização de regras extraídas por meio de redes neurais baseado na conversão de regras tradicionais em regras *fuzzy*.

Em 2000, DUCH et al. [20] apresentaram uma metodologia para extrair regras tradicionais e *fuzzy* de redes neurais. A metodologia basicamente inclui a determinação e a otimização das variáveis lingüísticas envolvidas, a geração inicial de conjuntos de regras de diferentes complexidades, a otimização destas regras e a estimativa da confiabilidade de classificação. A metodologia proposta foi avaliada em várias bases de dados e os resultados obtidos foram excelentes.

KEEDWELL et al. [21] desenvolveram um sistema em que um algoritmo genético é usado para extrair regras de redes neurais treinadas com dados contínuos e discretos. O algoritmo busca por regras interessantes no espaço das entradas da rede neural, e é avaliado em bases de dados em que há presença de ruído.

O principal problema enfrentado pelos algoritmos de extração de regras ocorre quando o modelo de rede neural incorpora uma representação distribuída [16]. Nesta, os conceitos incorporados pelo modelo são representados por padrões de ativação distribuídos em várias unidades, contrariamente ao que ocorre na representação local. Um exemplo deste tipo de problema pode ser observado na aplicação do *Algoritmo RX Modificado* [22,81,82,83], que se baseia no agrupamento das ativações das unidades ocultas, na modelagem da base de dados denominada *Pima Indians Diabetes*. Nesse caso, a rede neural proporciona bons resultados, enquanto que as regras extraídas apresentam resultados insatisfatórios devido ao tipo de representação incorporado pela rede neural. Para evitar esse problema, alguns autores sugerem o uso de arquiteturas e de algoritmos de treinamento de redes neurais especialmente desenvolvidos para a extração de regras. O presente trabalho se propõe a fornecer um método de extração de regras mais genérico, i.e. aplicável a uma maior variedade de arquiteturas e de algoritmos utilizados no treinamento da rede neural. Assim, os problemas oriundos da representação distribuída podem acontecer e, para tentar melhorar os resultados das

regras extraídas nesses casos, propõe-se o uso de algoritmos genéticos que *otimizem* os agrupamentos obtidos. Nesse sentido, foram desenvolvidos, implementados e avaliados dois algoritmos genéticos de agrupamento (*clustering)* para extrair regras de perceptrons de múltiplas camadas. A próxima seção define melhor o escopo deste trabalho.

1.3 Objetivo

Este trabalho foi motivado pelos resultados obtidos por meio do Algoritmo RX Modificado e se constitui, pois, no aperfeiçoamento dessa metodologia. No concernente à aplicabilidade, o método proposto se destina a obter conhecimento, na forma de regras lógicas, de modelos de perceptrons de múltiplas camadas treinados em problemas de classificação que envolvem atributos discretos e/ou contínuos. Além disso, o método proposto não requer algoritmos de treinamento específicos, isto é, o processo de extração de regras depende apenas do conhecimento da arquitetura da rede neural. Cumpre observar, entretanto, que a utilização de funções lineares nas unidades de entrada permitem a extração de regras mais compreensíveis quando da aplicação da metodologia proposta.

Basicamente, pode-se dizer que o método proposto supõe uma rede neural treinada, através da qual são computados os valores das ativações dos neurônios da primeira camada oculta. Essas ativações formam um vetor para cada objeto da base de dados. Os vetores são separados em subconjuntos de acordo com a classe a qual pertencem, isto é, o processo de extração de regras é realizado separadamente para cada classe envolvida. Em termos gerais, pode-se descrever a metodologia empregada para extrair regras de perceptrons de múltiplas camadas em dois passos principais:

1) Aplicar um algoritmo de *agrupamento* para encontrar *grupos* de ativações das unidades ocultas para cada classe, isto é, o algoritmo de *agrupamento* é aplicado ao espaço dos valores das ativações das unidades ocultas. Assim, estas representam as variáveis do processo de *agrupamento*;
2) Relacionar as expressões de ativação, de cada unidade oculta, aos valores mínimos e máximos das ativações de cada *grupo*. Dessa forma, pode-se descrever as ativações das unidades ocultas em função das unidades de entrada que, por sua vez, estão relacionadas às variáveis do problema em questão.

Essa metodologia fornece um meio de se relacionar regiões do domínio das variáveis às classes, por meio do agrupamento dos valores das ativações. Assim, o método proposto proporciona a extração de um conjunto de regras da seguinte forma:

Se $\{ v^1_{min} \leq a_1 \leq v^1_{max}$ e ... e $v^n_{min} \leq a_n \leq v^n_{max} \}$
Então classe C_j ;

onde os a_is são as expressões de ativação das unidades ocultas da primeira camada (em função das unidades de entrada), os $v^i s$ são os valores mínimos e máximos obtidos para cada agrupamento de uma classe C_j e n representa o número de unidades ocultas.

Pode-se citar que o processo de extração de regras se dá através da obtenção de hiperplanos que dividem o espaço dos atributos transformado linearmente. Esta transformação será linear sempre que a transformação dos atributos originais também o for, pois as ativações das unidades ocultas da primeira camada de uma rede neural podem ser escritas pela expressão:

$$a_k = \sum_{j=1}^{j=m} w_{kj} . x_j$$

onde a_k representa a ativação da *k-ésima* unidade oculta, m é o número de atributos envolvidos, w_{kj} são os pesos sinápticos para a *k-ésima* unidade oculta e os x_j´*s* representam os valores dos atributos transformados segundo a função utilizada. Nos experimentos realizados nesse trabalho, somente dois tipos de transformações são aplicadas nas unidades de entrada da rede neural: a função linear [0,1] e a função linear [-1,+1], pois, desta forma, pode-se obter regras mais compreensíveis, haja vista que nestes casos as expressões de ativação também são funções lineares dos atributos envolvidos. Assim, embora a rede neural possa mapear funções não lineares, o processo de extração de regras se dá em um espaço em que os atributos foram transformados linearmente pela otimização dos pesos das conexões.

Assim, a aplicação de um algoritmo de *agrupamento* no espaço das ativações pode proporcionar bons resultados porque é mais simples extrair padrões em um espaço transformado pelas unidades ocultas, que são detectoras de características. Basicamente, cada unidade oculta na primeira camada de um perceptron de múltiplas camadas forma um hiperplano [13], pois a transformação usada na camada de entrada é linear. Ou ainda, pode-se dizer que as unidades ocultas são detectoras de características porque

elas tentam mapear classes que, no espaço original não são linearmente separáveis, para o espaço das ativações, em que as classes possuem uma chance maior de serem linearmente separáveis [5]. Sob o ponto de vista geométrico, as regras lógicas correspondem a uma divisão do espaço das características, que se dá por meio de hiperplanos perpendiculares aos eixos, em *áreas* com nomes simbólicos. Como se descreve em [20], se as classes no espaço das entradas das unidades ocultas podem ser separadas por tais hiperplanos, então a descrição lógica dos dados é possível e meritória.

Conforme se pode observar em alguns trabalhos citados na seção 1.2, alguns autores notaram a necessidade de se simplificar os modelos de redes neurais, sendo favoráveis à utilização de arquiteturas e de algoritmos de treinamento específicos para o processo de extração de regras. Entretanto, entende-se que o desenvolvimento de algoritmos *mais genéricos* também é importante, principalmente porque estes podem ser usados em modelos de redes neurais previamente treinados bem como em novos problemas em que redes neurais sejam potencialmente úteis. Entretanto, o método proposto é dependente da arquitetura da rede neural obtida, principalmente do número de unidades de entrada e do número de unidades ocultas. Assim, modelos de redes neurais muito complexos tendem a fornecer regras também muito complexas. Essa constatação é natural e inerente à própria natureza da modelagem, pois não é razoável supor que um fenômeno complexo possa ser modelado por meio de um conjunto de regras simples. Cumpre observar, pois, que um algoritmo eficaz para extração de regras poderá ser usado tanto em redes neurais previamente treinadas quanto em novas tarefas de *data mining*. Além disso, conquanto os modelos de redes neurais usados neste trabalho contenham somente uma camada oculta, acredita-se que o método é facilmente estendido aos modelos de redes neurais com mais de uma camada oculta.

Um outro aspecto importante a ser observado diz respeito ao número de regras lógicas obtidas. Alguns algoritmos produzem centenas de regras lógicas, fornecendo uma descrição *opaca* dos dados e, por este motivo, estas regras não são mais compreensíveis do que o próprio modelo de rede neural [20]. Nesse sentido, desenvolveu-se um algoritmo de *agrupamento* que procura obter a quantidade *ideal* de regras, pois este algoritmo procura otimizar os *grupos* envolvidos e, para cada *grupo* encontrado há uma regra que o descreve.

O processo de *agrupamento* envolve a identificação de um conjunto de categorias ou *grupos* que descrevam o conjunto de dados. Assim, deseja-se encontrar a solução ótima para o problema de *agrupamento* das ativações das unidades da primeira camada

oculta, para se obter os melhores resultados possíveis nas regras extraídas. Desta forma, deve-se enfrentar o problema que envolve a descoberta da solução ótima para o particionamento de *N* dados em *C* classes. Este problema é NP-completo [98] e pode ser resolvido por meio de algoritmos genéticos [99]. Assim sendo, o desenvolvimento de algoritmos genéticos de *agrupamento* pode proporcionar um modo adequado de se extrair regras de perceptrons de múltiplas camadas e se constitui, pois, na principal contribuição desse trabalho.

1.4 Organização

Basicamente, este trabalho descreve o desenvolvimento de uma metodologia, baseada em algoritmos genéticos de *agrupamento (clustering)*, para extrair regras de perceptrons de múltiplas camadas treinados em problemas de classificação que envolvam atributos discretos e/ou contínuos. Nesse contexto, o próximo capítulo define genericamente o método de *agrupamento* utilizado. O processo de otimização dos *grupos* por meio de algoritmos genéticos também é abordado e alguns trabalhos correlatos são citados. Nesse sentido, pode-se dizer que os conceitos abordados no Capítulo 2 motivaram o desenvolvimento, a implementação e a avaliação dos dois algoritmos genéticos de *agrupamento* descritos nos capítulos seguintes.

Inicialmente, elaborou-se um algoritmo genético de *agrupamento* baseado em operadores tradicionais, sendo por este motivo denominado de Algoritmo Genético de Agrupamento Tradicional (AGAT), que é objeto de estudo do Capítulo 3. Os resultados obtidos por este algoritmo em três bases de dados – *Iris Plants, Pima Indians Diabetes* e *Australian Credit Approval* - motivaram a implementação de um algoritmo mais adequado aos problemas de *agrupamento.* A descrição e a avaliação desse novo algoritmo, denominado de Algoritmo Genético de Agrupamento Específico (AGAE) - que se constitui na principal contribuição do presente trabalho - pode ser encontrada no Capítulo 4.

Finalmente, o Capítulo 5 discute o método proposto, à luz dos resultados obtidos, e sugere trabalhos futuros promissores, enquanto que o Capítulo 6 apresenta as referências bibliográficas.

CAPÍTULO 2

Algoritmos Genéticos em Problemas de Agrupamento

Como já foi observado anteriormente, a motivação principal do presente trabalho se constitui na descrição do conhecimento incorporado por perceptrons de múltiplas camadas em aplicações de *data mining* que envolvam problemas de classificação. Assim sendo, pretende-se obter soluções *ótimas* para o problema de *agrupamento* das ativações das unidades ocultas pois, desta forma, regras *ótimas* que descrevam o conhecimento representado pela rede neural serão conseqüentemente obtidas. Cumpre observar, ainda, que o processo de *agrupamento* será aplicado às ativações de cada classe separadamente, de modo que se objetiva encontrar sub-classes que possam ser traduzidas em regras adequadas. Este capítulo fornece os conceitos que motivaram o desenvolvimento de algoritmos genéticos [100,101] para enfrentar o problema de otimização dos agrupamentos das ativações das unidades ocultas.

2.1 Descrição e Complexidade do Problema

Basicamente, o processo de *agrupamento* envolve a identificação de um conjunto de categorias - *grupos* - que descrevam o conjunto de dados [2]. Neste sentido, o processo de *agrupamento*, que é uma técnica de aprendizado não supervisionado, objetiva maximizar a homogeneidade entre os objetos de um mesmo grupo e, concomitantemente, maximizar a heterogeneidade entre objetos de *grupos* distintos.

Segundo HARTIGAN [19], *clustering* é o agrupamento de objetos semelhantes. O conceito de similaridade normalmente envolve o conceito de distâncias entre pares de objetos. Neste aspecto, ANDERBERG [18], afirma que a proximidade dos indivíduos é expressa como sendo uma distância. Assim sendo, a maioria dos algoritmos de *agrupamento* supõe conhecidas as distâncias entre pares de objetos, agrupando em *clusters* os objetos cujas distâncias são suficientemente pequenas, ou seja, menores do que determinada tolerância. Neste contexto, é de fundamental relevância a definição *da função de distância* a ser utilizada, bem como dos valores de tolerância. Em muitas situações, contudo, torna-se difícil ao usuário o estabelecimento do valor da tolerância [102]. Além disso, ANDERBERG [18] afirma que se o procedimento é aceito como

válido, então não há necessidade de se questionar a validade dos *grupos*, que são os conjuntos de objetos agrupados pela similaridade ou proximidade que guardam entre si [8].

Existem várias técnicas utilizadas no processo de *agrupamento*. A literatura sobre a análise de *grupos* é muito ampla [103] e entende-se que a descrição dos variados métodos desenvolvidos foge do escopo deste capítulo. Assim, optar-se-á pela citação dos conceitos mais importantes ao método proposto, procurando dedicar uma atenção especial à formalização da metodologia empregada. O leitor interessado em obter um referencial teórico sobre *agrupamento* poderá consultar, por exemplo, o excelente trabalho de ARABIE et al. [103].

O presente trabalho trata da aplicação de uma abordagem de *agrupamento* que atribui cada objeto da base de dados a um único grupo. Assim, considera-se que o processo de *agrupamento* envolve a separação de um conjunto X de objetos em uma coleção de sub-conjuntos mutuamente disjuntos C_i de X. Formalmente, considere-se um conjunto de n objetos $X=\{x_1, x_2,...,x_n\}$ que deve ser agrupado, onde cada $x_i \in \Re^\rho$ é um vetor de atributos composto por "ρ" medições reais (contínuas). Os objetos devem ser agrupados em grupos não sobrepostos $C=\{C_1, C_2,..., C_k\}$ onde k é o número de *grupos*, tal que [104]:

$$C_1 \cup C_2 \cup ... \cup C_k = X, \quad C_i \neq \varnothing, \quad \text{e} \quad C_i \cap C_j = \varnothing \text{ para } i \neq j.$$

No concernente ao valor de k, cumpre observar que a maioria dos métodos assume que o mesmo é fornecido pelo usuário [105,106]. Dessa forma, estes algoritmos se concentram em obter k grupos de objetos semelhantes de acordo com algum critério pré-estabelecido. LIU [107] define que o número de possibilidades de se classificar n objetos em k grupos é dado por:

$$N(n,k) = \frac{1}{k!}\sum_{i=0}^{k}(-1)^i \binom{k}{i}(k-i)^n$$

Por exemplo, existem N(25,5) = 2.436.684.974.110.751 formas de classificar 25 objetos em 5 grupos. Dessa maneira, percebe-se a complexidade de se agrupar *corretamente* uma base de dados de 25 objetos em 5 grupos. É necessário observar, ainda, que normalmente esse problema é considerado simples quando comparado com aqueles encontrados em aplicações de *data mining*. Entretanto, ao considerar que o

valor de *k* é desconhecido, como por exemplo nos problemas abordados neste trabalho, existem $\sum_{i=1}^{n} N(n,k)$ maneiras de se agrupar *n* objetos em *k grupos*. Considerando 25 objetos, este número representa aproximadamente $4x10^{18}$ agrupamentos [104].

De uma maneira mais formal, o problema de se encontrar uma solução ótima para a separação de *n* objetos em *k* grupos é NP-completo [99] e, devido ao fato de que o número de separações possíveis desses *n* objetos em *k grupos* aumenta aproximadamente na razão $k^n/k!$, a tentativa de se encontrar uma solução ótima globalmente é usualmente inviável sob o ponto de vista computacional [108].

Por outro lado, a escolha do valor de *k* é uma tarefa complicada, haja vista que alguns desses valores não implicam em *agrupamentos naturais.* Nesse sentido, aconselha-se rodar o algoritmo várias vezes, variando-se o valor de *k*, para depois escolher a solução cujas características se parecem melhores ou, ainda, aquelas soluções que forneçam a interpretação mais *significativa* [105]. Esta estratégia freqüentemente assume um certo conhecimento do domínio em questão.

Uma outra alternativa se constitui em escolher a melhor solução – valor mais adequado de *k* - de acordo com algum critério numérico. A determinação do número ótimo de grupos em um conjunto de dados é um dos mais difíceis aspectos do processo de agrupamento [21]. Como já foi observado [105], a maioria das abordagens encontradas na literatura adota critérios que determinam o número de grupos baseando-se em agrupamentos obtidos para vários valores de *k*. Nesse sentido, algoritmos de *agrupamento* tradicionais procuram pela melhor solução em um subconjunto relativamente pequeno do espaço de busca e, conseqüentemente, suas possibilidades de sucesso são pequenas [104]. FASULO [109] faz uma análise de vários trabalhos publicados sobre algoritmos de *agrupamento*, e observa que a *mineração de dados* é uma aplicação promissora da tecnologia de *agrupamento*, mas que poucos trabalhos tem sido realizados para elegantemente eliminar a necessidade do uso de parâmetros fixos fornecidos pelo usuário dos algoritmos de *agrupamento*. Assim, a maioria das abordagens para *data mining* implicitamente requer que o usuário rode o algoritmo de *agrupamento*, modifique os parâmetros e escolha os resultados que pareçam ser os melhores.

No entanto, cumpre observar que alguns trabalhos recentes têm proporcionado desenvolvimentos significativos no tocante à determinação do valor *ideal* de *k*. O leitor interessado neste assunto pode consultar, por exemplo, os trabalhos descritos em

[98,102,104,105,110-119]. Outras referências podem ser encontradas em [106,120]. O presente trabalho também contribui para esta área de pesquisa.

A metodologia empregada neste trabalho supõe a *otimização* dos grupos para determinado valor de *k* bem como do próprio valor de *k*. Assim sendo, deve-se maximizar a homogeneidade interna dos grupos e a heterogeneidade entre os grupos distintos, considerando-se também a obtenção do *melhor* valor de *k*. Dessa forma, deseja-se procurar pela melhor solução em um subconjunto muito maior do espaço de busca. Algoritmos genéticos são eficazes em problemas de otimização de *agrupamento* e podem fornecer soluções sub-ótimas em um tempo razoável [98] e, assim, representam a alternativa escolhida para enfrentar esse problema. A próxima seção se propõe a discutir os conceitos gerais que envolvem a utilização de algoritmos genéticos em problemas de *agrupamento*.

2.2 Aspectos da Aplicação de Algoritmos Genéticos

Esta seção aborda alguns trabalhos que descrevem a aplicação de algoritmos genéticos em problemas de *agrupamento*. Existem diversas complicações que surgem na utilização de algoritmos genéticos *tradicionais* em problemas de *agrupamento*. Dentre estes, pode-se citar como principais a codificação redundante e a insensibilidade ao contexto.

COLE [104] estudou a aplicação de algoritmos genéticos em problemas de *agrupamento*. A autora revisou vários trabalhos publicados sobre o assunto e implementou vários algoritmos de *agrupamento*, comparando seus desempenhos em três bases de dados. No tocante aos métodos de *agrupamento* que supõem conhecido o valor de *k*, os experimentos realizados não representaram vantagens aos algoritmos genéticos quando comparados a alguns métodos de *agrupamento tradicionais*. A autora também aborda problemas em que o valor de *k* não é conhecido, a exemplo dos problemas abordados no presente trabalho.

No método implementado, a autora faz uso da mesma representação cromossômica usada no Algoritmo Genético de Agrupamento Específico (AGAE), discutido no Capítulo 4, e de populações iniciais geradas aleatoriamente. O problema de insensibilidade ao contexto é evitado por meio de um operador de recombinação específico, que é baseado na interseção dos grupos pertencentes aos genótipos-pais. Assim, os grupos comuns aos dois genótipos são mantidos e os demais são agrupados

de acordo com a maior semelhança possível. Este processo gera genótipos-filhos formados por um número de grupos que varia no intervalo [k_1, k_2] em que k_1 e k_2 representam o número de *grupos* dos pais 1 e 2 respectivamente, a exemplo do que acontece na aplicação do operador de recombinação usado no AGAE.

No concernente aos operadores de mutação, COLE [104] implementou três operadores. Desses, somente o operador denominado de *merge* não origina uma busca totalmente aleatória por uma melhor solução. Este operador elimina um *grupo* e re-aloca todos os seus objetos nos *grupos* de centróides mais próximos, a exemplo do *operador 1* do AGAE. Cumpre observar, entretanto, que a utilização desse operador *força* o algoritmo a procurar por genótipos formados por um menor número de grupos. Nesse sentido, o algoritmo pode, com maiores chances, ficar *preso* em regiões de mínimos locais, pois a união de grupos em genótipos formados por *k grupos* nem sempre origina bons genótipos formados por (k-1) *grupos.* A autora utiliza da função objetivo definida em [115] que se baseia na minimização das distâncias Euclidianas entre os objetos e os centróides de seus *grupos*.

COLE [104] conclui o seu trabalho sugerindo que o desenvolvimento de operadores mais adaptados ao problema de *agrupamento* e de populações melhor estruturadas podem melhorar a eficácia dos algoritmos genéticos em problemas de *agrupamento*. Nesse sentido, acredita-se que o AGAE contribui de maneira significativa para esta área de pesquisa. Além disso, pode-se dizer que o AGAE se constitui na adaptação e no desenvolvimento do paradigma de utilização de algoritmos genéticos em problemas de agrupamento proposto por FALKENAUER [99].

Basicamente, o autor mostra os problemas enfrentados no emprego de algoritmos genéticos em problemas de *agrupamento*, mas não define uma metodologia específica para solucionar tais problemas, dada a grande variedade dos problemas encontrados nessa área. Além disso, o autor descreve como a metodologia proposta pode ser empregada em alguns problemas *reais*.

Pode-se afirmar que a metodologia desenvolvida no presente trabalho tem forte influência do trabalho de FAlKENAUER [99], que é um excelente ponto de referência ao leitor interessado no assunto. Inicialmente, optou-se por implementar um algoritmo de *agrupamento* baseado em operadores tradicionais, cuja descrição pode ser encontrada no Capítulo 3. Este algoritmo serviu de motivação para o desenvolvimento de um outro algoritmo, projetado especificamente para resolver problemas de *agrupamento* que envolvam a separação de um conjunto de dados. Este algoritmo foi

denominado de Algoritmo Genético de Agrupamento Específico (AGAE) e sua descrição pode ser encontrada no Capítulo 4. Nesse sentido, pode-se dizer que o AGAE representa uma adaptação do paradigma proposto em [99]. Essa adaptação é adequada aos problemas enfrentados no presente trabalho e também resolve alguns dos problemas não abordados em [99].

Basicamente, o autor sugere que a codificação normalmente utilizada, i.e. genótipos orientados a objetos, é a principal responsável pelos principais problemas enfrentados pelos algoritmos genéticos de agrupamento, e propõe o uso de um novo esquema de codificação. Na codificação orientada a objetos, as soluções para o problema de *agrupamento* são codificadas por meio de um vetor de *n* posições, no qual cada coordenada se refere a um objeto e assume o valor do grupo correspondente. FALKENAUER [99] adota um esquema de codificação em que estes vetores são aumentados, contendo uma parte adicional que representa os grupos, resultando em um tipo de codificação em que os vetores possuem comprimentos variáveis. Além disso, os operadores genéticos operam com a segunda parte do vetor, i.e. com os grupos de objetos. A metodologia proposta [99] permite a otimização tanto dos *grupos* quanto do número de *grupos*.

No concernente aos operadores sugeridos, observa-se que Algoritmo Genético de Agrupamento Específico (AGAE) faz uso de operadores que, conceitualmente, representam o método proposto em [99]. Cumpre observar que não se faz uso de operadores de inversão. Estes poderiam ter sido utilizados no Algoritmo Genético de Agrupamento Tradicional (AGAT), que se constitui no ponto de partida para o desenvolvimento do AGAE, mas, conquanto a aplicação do operador de *inversão* também faça sentido na codificação sugerida pelo autor, a motivação de sua aplicação é questionável.

Conceitualmente, o papel do operador de inversão é mudar as posições de alguns genes para favorecer a transmissão dos bons esquemas às gerações seguintes quando da aplicação do operador de *recombinação*. No entanto, este operador é aplicado aleatoriamente em dois pontos, selecionando-se alguns grupos que são, então, transferidos à geração seguinte. Assim, pode-se concluir que a eficácia do operador fica condicionada à aleatoriedade envolvida, e este fato justifica a nossa opção pela não utilização do operador em questão. Em outras palavras, o operador de inversão tanto poderá *aproximar* grupos co-adaptados como poderá separá-los, e a freqüência com que cada situação ocorre depende da aleatoriedade inerente ao processo de seleção dos

grupos. Por outro lado, a tentativa de se favorecer a transmissão de bons esquemas acaba nos levando à resolução do próprio problema original, i.e. de encontrar os agrupamentos corretos. Dessa forma, o ponto em discussão é essencialmente a aplicação/demonstração do *teorema dos esquemas* em problemas de agrupamento. FALKENAUER [99] acredita que a demonstração desse teorema aos problemas de agrupamento enfrentados de acordo com o paradigma proposto é apenas uma questão de tempo. TANOMARU [121], por sua vez, observa que a demonstração formal do funcionamento dos algoritmos genéticos tem poucas chances de elucidar o poder de processamento dos mesmos.

Entende-se que o método proposto no AGAE representa uma evolução do paradigma proposto em [99], haja vista a utilização de genótipos de comprimento constante e do uso da codificação tradicional, notoriamente mais simples. Além disso, o presente trabalho soluciona os problemas inerentes aos operadores de recombinação e mutação propostos em [99] para as aplicações de *agrupamento* que envolvem a separação de um conjunto de dados. A descrição do método proposto (AGAE) pode ser encontrada no Capítulo 4. Os trabalhos de FALKENAUER [99] e de COLE [104] estão muito relacionados ao desenvolvimento do AGAE e, por este motivo, receberam uma atenção especial nesta seção. Entretanto, pode-se encontrar outros trabalhos que tratem do problema de otimizar agrupamentos por meio de algoritmos genéticos.

PARK e SONG [98], por exemplo, desenvolveram um algoritmo genético de *agrupamento* que assume ser desconhecido o número de partições ideais – valor de *k*, para separar grupos de palavras com sentidos coerentes. Os autores desenvolveram uma codificação e um operador de recombinação específicos para o problema a ser resolvido. Pode-se observar, ainda, que embora a codificação proposta seja *redundante,* os autores descrevem resultados melhores do que aqueles obtidos por meio de outros algoritmos de *agrupamento*. PARMEE [122] descreve um algoritmo genético de agrupamento que decompõe espaços complexos de projetos em regiões de alto desempenho. Basicamente, a estratégia proposta permite que engenheiros identifiquem *regiões de projeto*, por exemplo no projeto de turbinas, procurando em um espaço de soluções em que há restrições de tempo e de orçamento.

CAPÍTULO 3

Algoritmo Genético de Agrupamento Tradicional

Teoricamente, algoritmos genéticos podem ser muito úteis para resolver problemas de *agrupamento.* Entretanto, existem poucos trabalhos que tratam desse assunto. Assim sendo, cumpre avaliar o potencial da aplicação de algoritmos genéticos de *agrupamento* na extração de regras de redes neurais. Para tal, implementou-se um protótipo, denominado de Algoritmo Genético de *Agrupamento Tradicional (AGAT)*. Este algoritmo, denominado de *tradicional* devido aos operadores utilizados, foi avaliado em três bases de dados: *Iris Plants, Pima Indians Diabetes* e *Australian Credit Approval.* Os resultados obtidos pelo AGAT nestas três bases de dados motivaram o desenvolvimento do Algoritmo Genético de Agrupamento Específico, que é o objeto de estudo do Capítulo 4. O presente capítulo descreve o AGAT e os resultados de sua aplicação. Considerando-se que o AGAT se constituiu numa espécie de *ponto de partida* para a AGAE, que é a principal contribuição do presente trabalho, optou-se por descrever apenas os resultados mais importantes de sua aplicação. Além disso, o leitor interessado poderá encontrar uma descrição mais detalhada das aplicações realizadas por meio do AGAT em [123-126].

A aplicação de algoritmos genéticos em problemas de *agrupamento* implica em algumas complicações. A principal diz respeito à utilização de uma representação cromossômica adequada. Considerando-se uma base de dados formada por N objetos a serem agrupados, cada fenótipo é representado por um vetor de N elementos, cada um dos quais representando o número inteiro referente ao rótulo do *grupo*. Assim, levando-se em consideração que cada objeto é numerado de 1 a N, o i-ésimo elemento (i-ésimo gene) de um genótipo representa o i-ésimo objeto da base de dados. Esta representação ocasiona uma codificação redundante, bem como uma insensibilidade ao contexto [99]. Para evitar tais problemas, desenvolveu-se um algoritmo, denominado de Algoritmo de Consistência (AC), que é um algoritmo de re-numeração. A seção 3.1 descreve o algoritmo implementado, enquanto que a seção 3.2 descreve os resultados experimentais da sua aplicação. A seção 3.3 comenta os principais resultados obtidos pelo AGAT.

3.1. Algoritmo Implementado - AGAT

No concernente ao algoritmo implementado, optou-se por limitar o número de *grupos*, pois este influencia de maneira significativa na complexidade do conjunto de regras obtido. Cada *grupo* representa uma regra para uma classe. Ou seja, quanto maior o número de *grupos* , maior a complexidade do conjunto de regras. Assim, optou-se por utilizar no máximo cinco *grupos* por classe. Dessa forma, cada gene de um indivíduo assume um valor do alfabeto {1,2,3,4,5}. Conquanto o número máximo de *grupos* seja definido *a priori*, o número de grupos obtido é *otimizado* (dentro do limite previamente estabelecido) pelo AGAT. É necessário, entretanto, citar que situação ideal envolve a utilização de uma metodologia para definir a *quantidade ótima* de *grupos* para cada aplicação. Este problema é enfrentado pelo AGAE – capítulo 4.

A fim de se exemplificar os problemas relativos à codificação redundante e de insensibilidade ao contexto [23], considere-se um conjunto de dados composto por 20 objetos. Um dos genótipos possíveis seria:

Genótipo 1: 22345123453321454552

Este genótipo representa uma solução formada por cinco *grupos*, na qual o *grupo* cujo rótulo é igual a 2 é constituído por 5 objetos {1,2,7,13,20}. O *grupo* de rótulo 1 é composto pelos objetos {6,14}. Esta codificação é redundante, pois existem 120 genótipos diferentes representando o mesmo fenótipo tais como, por exemplo, os genótipos abaixo:

Genótipo 2: 33451234514432515113
Genótipo 3: 44512345125543121224
Genótipo 4: 55123451231154232335
Genótipo 5: 11234512342215343441

Assim, o AGAT procura pela melhor solução em um espaço de busca muito maior do que o espaço de busca original. Além disso, a solução ótima depende do agrupamento de objetos e não do rótulo do *grupo* associado. Um outro problema que surge na aplicação do esquema de codificação em questão se refere à aplicação do operador de recombinação (*crossover),* e acontece com a utilização de informações dependentes de contexto como se estas não dependessem do contexto. Este problema é denominado de "insensibilidade ao contexto" [99].

Considere-se, por exemplo, a recombinação do genótipo 2 com o genótipo 3. Neste caso, o genótipo resultante deveria ser idêntico aos seus pais, pois estes representam o mesmo fenótipo. Isto raramente acontece e, para facilitar a visualização deste fenômeno, considere-se a aplicação do operador de recombinação (em dois pontos) para os genótipos 6 e 7, resultando nos genótipos filhos 1 e 2:

Genótipo 6: 2 *2222* 111113333344444
Genótipo 7: 4 *4444* 333335555511111

Filho 1: 2 *4444* 111113333344444
Filho 2: 4 *2222* 333335555511111

Observa-se que os genótipos filhos obtidos são diferentes dos seus pais (genótipos 6 e 7), isto é, eles representam fenótipos diferentes. Na verdade, o *filho 2* representa 5 *grupos*, enquanto que os seus pais representam somente 4 *grupos*. Para evitar estes problemas, desenvolveu-se um algoritmo de consistência, que permite uma reorganização dos genótipos. Considere-se um genótipo *G* – onde *G(i)* representa o rótulo do grupo para o gene "*i* " – e um vetor auxiliar *S* de dimensão *N*. Assim sendo, o algoritmo de consistência realiza os seguintes passos:

```
1 – Fazer todos os N genes de S iguais a zero;
2 – Rótulo = 1
3 – Para i =1,...,N faça:
            Se  S[i] = 0  então:
                  Aux = G[i];
                  Para  j=1,...,N faça:
                        Se ( G[j] = Aux  e  S[j] = 0 ) então:
                              G[j] = Rótulo;
                              S[j] = 1;
                        Fim Se;
                  Fim Para;
                  Rótulo = Rótulo + 1;
            Fim Se;
   Fim Para.
```

Figura 1. Algoritmo de Consistência (AC)

Aplicando este algoritmo aos genótipos {1,2,3,4,5}, constata-se que todos eles codificariam a mesma solução, que seria representada pelo seguinte genótipo:

Genótipo* : 11234512342215343441

O algoritmo de consistência (AC) permite, então, que o AGAT procure pela melhor solução em um espaço em que existe somente um único genótipo representando cada fenótipo. Basicamente, o *Algoritmo Genético de AgrupamentoTradicional* segue os passos descritos na Figura 2.

1 – Inicializar a população de genótipos;
2 – Aplicar o AC à população atual;
3 – Avaliar cada genótipo da população e aplicar uma normalização linear;
4 – Selecionar indivíduos através do método da roleta;
5 – Aplicar os operadores de recombinação e mutação nesses genótipos;
6 – Eliminar os genótipos da população antiga e formar uma nova população com os genótipos obtidos no passo 5;
7 – Se a convergência foi alcançada, parar; Senão, voltar para o passo 2.

Figura 2. Algoritmo Genético de Agrupamento Tradicional (AGAT) – Descrição Geral.

3.1.1. Função Objetivo

A função objetivo, também conhecida por função de custo ou função de adequabilidade [121], deve refletir o modo segundo o qual cada genótipo está, comparativamente aos demais, mais próximo do ponto de ótimo. Em problemas de *agrupamento*, um grupo formado por um único objeto não representa uma solução adequada. Assim, a função objetivo deve, em princípio, proporcionar valores maiores na medida em que o tamanho do *grupo* aumente. Além disso, cumpre também evitar a solução trivial, que é representada pelo agrupamento de todos os objetos em um só grupo. A função objetivo utilizada no AGAT leva em consideração o número de objetos pertencentes a cada *grupo* e a distância Euclidiana entre estes objetos. Basicamente, a função objetivo pode ser escrita como:

$$F = E.dex - I.din + Fator \qquad (1)$$

onde:

- E e I são os fatores externo e interno respectivamente (definidos pelo usuário);
- dex é a distância externa média, i.e. a distância média entre os centróides dos *grupos* representados por um genótipo;
- din é a distância interna média, que é calculada como a distância média entre os centróides e os objetos pertencentes aos grupos codificados no genótipo;
- *Fator* é uma função do número de objetos presentes nos grupos.

Espera-se que todos os termos da equação (1) tenham aproximadamente a mesma influência no valor da função objetivo. A principal preocupação diz respeito ao termo *Fator*. Como este termo basicamente depende do número de objetos pertencentes a cada grupo, ele não deveria ser, na média, maior do que os outros termos. Nos casos em que isto ocorre, o algoritmo procuraria por soluções de *grupos* formados por um número elevado de elementos, independentemente da proximidade entre os elementos de cada *grupo.* Além disso, deseja-se obter *grupos* uniformes, isto é, em princípio a solução para o problema deveria representar cinco *grupos* com um número igual de elementos. Uma maneira de se dispor de um certo controle sobre este fenômeno se dá pela imposição de que *Fator* possua valores que, na média, sejam aproximadamente iguais aos da distância média entre todos os objetos. Além disso, deseja-se que, em princípio, os *grupos* possuam o mesmo número de objetos. Desta forma, considera-se, em princípio, que a melhor solução possui cinco *grupos* de mesmo tamanho. Para se ter *grupos* uniformes, pode-se fazer com que o termo *Fator* seja máximo quando o genótipo representar *grupos* uniformes. Uma maneira de se impor esta condição se dá através de:

$$Fator = (n_t - n_{máx}) \qquad (2)$$

Onde n_t representa o número total de objetos a serem agrupados e $n_{máx}$ representa o número de objetos do maior *grupo* (considerando-se os cinco possíveis). No entanto, deseja-se ponderar *Fator* de maneira que o mesmo não represente valores desproporcionais em relação aos outros termos. Assim sendo, pode-se multiplicar este termo por um multiplicador de modo que:

$$Fator = (n_t - n_{máx}) \bullet \text{multiplicador} \quad (3)$$

Considerando-se, então, que a situação ideal envolve *grupos* uniformes tem-se que $n_{máx} = n_t/5$. Deseja-se que *Fator* seja, na média, aproximadamente igual à distância média (d_m) entre todos os objetos, de modo que *Fator* = d_m , donde se conclui que multiplicador = $5d_m / 4n_t$ e que, portanto:

$$Fator = \frac{5\,(n_t - n_{máx}) \bullet d_m}{4n_t} \qquad (4)$$

A função objetivo utilizada proporciona valores maiores quando as distâncias externas são elevadas e quando as distâncias internas são pequenas. Além disso, os

grupos devem ser uniformes. Mais especificamente, *dex* e *din* são calculadas de acordo com as Figuras 3 e 4 respectivamente:

1 – Calcular os centróides de cada um dos *grupos* representados pelo genótipo;
2 – Calcular as distâncias Euclidianas entre estes centróides;
3 – Somar as distâncias calculadas no passo 2);
4 – Dividir o valor obtido em 3) pelo número de combinações de *grupos*, i.e. por c! / [2!(c-2)!], onde *c* é o número de grupos.

Figura 3. Cálculo da distância externa (*dex).*

1 – Calcular os centróides de cada grupo representado pelo genótipo;
2 – Para cada grupo, calcular as distâncias Euclidianas entre o seus centróides e os seus objetos. Somar estas distâncias e dividir este resultado pelo número de objetos de cada um dos grupos em consideração, obtendo-se a distância média entre os centróides e os objetos de cada grupo;
3 – Somar as distâncias calculadas no passo 2), que são as distâncias médias para cada grupo e dividir este valor pelo número de grupos.

Figura 4. Cálculo da distância interna (*din).*

Para proporcionar uma comparação mútua entre os genótipos, utiliza-se da normalização linear [101]. Primeiramente, os valores das funções objetivo de cada genótipo são classificados e então re-calculados por meio da expressão:

$$F' = \frac{R(i) \bullet [\, F'(max) - F'(min) \,]}{\text{Tamanho da População}} ; \qquad (5)$$

onde:

- *R(i)* é a posição do genótipo *i* na classificação – utiliza-se de uma classificação em que os melhores genótipos possuam maiores valores; *F'(max)* é o maior valor da função objetivo para a população em questão;
- *F'(min)* é o menor valor da função objetivo para a população;

3.1.2. Recombinação e Mutação

Os indivíduos que fazem parte de cada geração são selecionados de acordo com a estratégia de seleção da *roleta*. Além disso, utiliza-se de uma técnica elitista: 20% dos genótipos de cada geração representam os melhores da geração anterior. Assim, 80% dos indivíduos são selecionados de acordo com o "método da roleta". Todos os indivíduos selecionados são re-combinados: o primeiro indivíduo é re-combinado com o

último selecionado, o segundo com o penúltimo e assim por diante. Utiliza-se um processo de recombinação de dois pontos (*two-point crossover*) pois este é mais adequado em problemas de agrupamento [99].

No concernente à mutação, cada indivíduo pode ter todos os seus genes alterados. Para tal, sorteia-se uma semente – $s \in [0,1]$ - para cada gene. Caso esta seja menor ou igual do que a probabilidade de mutação – p_m - então o gene poderá ser mutado. Isto acontece porque sempre que $s \leq p_m$ uma nova semente é sorteada para definir o valor do alelo do gene. Considerando-se que se dispõe de 5 alelos possíveis, então o gene selecionado tem 80% de chance de ser mutado a partir do momento em que a semente sorteada atenda ao critério da p_m. Considerando-se que a mutação é muito destrutiva quando o AGAT está próximo de convergir [99], então p_m é igual a um valor tipicamente baixo, i.e. 0,004.

3.1.3. População Inicial

Em problemas de *agrupamento*, a geração aleatória da população inicial freqüentemente origina genótipos cujos valores da função objetivo são negativos e, portanto, impróprios para emprego quando da utilização do método de seleção da roleta. Dessa forma, utilizou-se de um método bastante simples para evitar esse problema e, ainda, para tentar acelerar a convergência do algoritmo.

Primeiramente, os valores das ativações das unidades ocultas são padronizados, de modo que todas as unidades ocultas possuam média igual a zero e desvio padrão unitário. Para tal, subtrai-se de cada valor a média e divide-se este resultado pelo desvio padrão. Este procedimento permite eliminar tendências que possam existir devido às diferenças de escala das variáveis [127]. A aplicação desta metodologia, por sua vez, também permite a utilização de um método muito simples para gerar a população inicial.

Para formar a população inicial, calculam-se as distâncias entre os padrões (vetores) de ativação e o centróide de cada classe. Dado que o centróide de cada classe é um vetor de coordenadas nulas, a distância entre um padrão e o centróide é igual ao próprio módulo do vetor de ativação. O maior valor obtido é chamado de *max*. Considere-se, então, que a distância Euclidiana entre um padrão *i* e o centróide é denotada por *d[i]*, e que *N* é o número de padrões de ativação (exemplos). Estabelecem-se, então, quatro soluções iniciais, de modo que todos os outros genótipos são formados

por meio de mutações – considerando-se que cada gene tem 30% de chance de ser alterado - dos quatro genótipos estabelecidos de acordo com a Figura 5.

Indivíduo1: Para i=1,...,N faça: Se d[i] < 0.1max então grupo 1. Se 0.1max ≤ d[i] < 0.2max então grupo 2; Se 0.2max ≤ d[i] < 0.3max então grupo 3; Se 0.3max ≤ d[i] < 0.4max então grupo 4; Se d[i] ≥ 0.4max então grupo 5; Fim Para.	**Indivíduo 2:** Para i=1,...,N faça: Se d[i] < 0.2max então grupo 1. Se 0.2max ≤ d[i] < 0.3max então grupo 2; Se 0.3max ≤ d[i] < 0.4max então grupo 3; Se 0.4max ≤ d[i] < 0.5max então grupo 4; Se d[i] ≥ 0.5max então grupo 5; Fim Para.
Indivíduo 3: Para i=1,...,N faça: Se d[i] < 0.2max então grupo 1. Se 0.2max ≤ d[i] < 0.4max então grupo 2; Se 0.4max ≤ d[i] < 0.6max então grupo 3; Se 0.6max ≤ d[i] < 0.8max então grupo 4; Se d[i] ≥ 0.8max então grupo 5; Fim Para.	**Indivíduo 4:** Para i=1,...,N faça: Se d[i] < 0.3max então grupo 1. Se 0.3max ≤ d[i] < 0.4max então grupo 2; Se 0.4max ≤ d[i] < 0.5max então grupo 3; Se 0.5max ≤ d[i] < 0.6max então grupo 4; Se d[i] ≥ 0.6max então grupo 5; Fim Para.

Figura 5. Geração da População Inicial.

3.2. Estudo de Casos

Esta seção tem por objetivo descrever os resultados obtidos pelo Algoritmo Genético de Agrupamento Tradicional (AGAT), que foi avaliado em três bases de dados distintas: *Iris Plants*, *Pima Indians Diabetes* e *Australian Credit Approval.* Estes experimentos tinham por objetivo avaliar o potencial que um algoritmo genético de agrupamento teria para a extração de regras de redes neurais. Os resultados descritos nesta seção foram publicados em [123-126], onde se pode encontrar uma descrição mais detalhada dos experimentos realizados.

3.2.1 Iris Plants Database

Utilizou-se, como primeiro exemplo, o banco de dados conhecido como *Iris Plants Database,* de autoria de R.A. Fisher. Este problema é clássico na literatura de *Reconhecimento de Padrão,* sendo também referenciado em muitos trabalhos que envolvem técnicas de *data mining*.

Os dados contém três classes de plantas intituladas de Setosa, Versicolour e Virginica. Cada classe contém 50 exemplos. A primeira classe é linearmente separável das demais, enquanto que as classes Versicolour e Virginica não são linearmente

separáveis entre si. Os quatro atributos envolvidos são: comprimento e largura de sépalas e de pétalas.

Considerando-se que o conjunto de dados representa a população para o domínio em questão, optou-se pelo treinamento da rede em todo o conjunto de dados que constitui, pois, no conjunto de treinamento. Assim sendo, não se utilizou de conjuntos de teste e de validação.

Como já foi anteriormente observado [22] as áreas de pétalas e de sépalas usualmente proporcionam melhores resultados do que a utilização dos quatro atributos originais envolvidos. Os resultados obtidos para esta base de dados são comparados àqueles obtidos pelo Algoritmo RX Modificado [22,81,82,83], que também é um algoritmo para extrair regras de redes neurais.

Com o objetivo de encontrar a melhor precisão possível e com a menor arquitetura possível, várias redes neurais foram treinadas. A melhor delas possui duas unidades de entrada, utilizando uma transformação linear [-1,+1], duas unidades ocultas com a função tangente hiperbólica, e duas unidades de saída com função logística. Uma descrição completa deste experimento pode ser encontrada em [22]. Após 15.000 épocas, esta rede neural obteve um erro médio quadrado igual a 0,0003. Calculam-se, então, os valores das ativações para cada exemplo do conjunto de treinamento de acordo com as seguintes expressões:

$$a_1 = 0.22A_s - 2.16A_p + 0.84 \quad (6)$$
$$a_2 = 3.12A_s - 11.91A_p + 41.99 \quad (7)$$

Os valores das ativações para cada exemplo são padronizados e então o AGAT é aplicado para extrair as regras de classificação. Utilizou-se uma população formada por 200 genótipos e a função de adequabilidade é dada pela expressão (1) considerando-se que *E* e *I* são iguais a 1 e também que *Fator* é dado pela expressão (4). Considerando a classe *setosa*, o *AGAT* converge após 463 gerações e o valor da função objetivo associada ao genótipo solução é igual a 4,88. O genótipo obtido representa 5 *grupos*, constituídos por 16, 8, 16, 9 e 1 objetos respectivamente, e que por sua vez representam as regras *R1, R2, R3, R4* e *R5* da Figura 6. Considerando a classe *versicolor* , o *AGAT* converge após 421 gerações e o valor da função objetivo associada ao genótipo solução é igual a 10,58. O genótipo obtido representa 5 *grupos*, constituídos por 18, 18, 12, 1 e 1 objetos respectivamente, e que por sua vez representam as regras *R6, R7, R8, R9* e *R10* da Figura 6. Considerando a classe *virgínica* , o *AGAT* converge após 401 gerações e o

valor da função objetivo associada ao genótipo solução é igual a 10,68. O genótipo obtido representa 5 *grupos*, constituídos por 16, 15, 17, 1 e 1 objetos respectivamente, e que por sua vez representam as regras *R11, R12, R13, R14* e *R15* da Figura 6. O tempo médio de CPU para extrair as regras de cada classe foi igual a 45 segundos em um Pentium 233 MHz, 32RAM. O conjunto de regras obtido proporciona uma taxa de classificação correta igual a 97%. Observa-se, ainda, que as regras R5, R9, R10, R14 e R15 não possuem poder de generalização e que a regra R3 representa todos os exemplos modelados pela regra R1.

R1: Se $\{3.6 \leq a_1 \leq 4.2$ e $85.87 \leq a_2 \leq 95.3\}$
então SETOSA.
R2: Se $\{2.3 \leq a_1 \leq 3.5$ e $69.7 \leq a_2 \leq 85.2\}$
então SETOSA.
*R3:*Se $\{3.2 \leq a_1 \leq 4.5$ e $82.86 \leq a_2 \leq 103.5\}$
então SETOSA.
*R4:*Se$\{4.0 \leq a_1 \leq 5.4$ e $95.2 \leq a_2 \leq 113.1\}$
então SETOSA.
*R5:*Se $\{a_1 = 5.5$ e $a_2 = 111.6\}$
então SETOSA.

R6: Se $\{-9.2 \leq a_1 \leq -6.2$ e $19.5 \leq a_2 \leq 33.8\}$
então VERSICOLOR.
R7: Se $\{-13.7 \leq a_1 \leq -7.9$ e $-2 \leq a_2 \leq 22.8\}$
então VERSICOLOR.
R8: Se $\{-6.3 \leq a_1 \leq -3.5$ e $31.5 \leq a_2 \leq 42.5\}$
então VERSICOLOR.
R9: Se $\{a_1 = -3.4$ e $a_2 = 46.5\}$
então VERSICOLOR.
R10: Se $\{a_1 = -13.2$ e $a_2 = -4.7\}$
então VERSICOLOR.

*R11:*Se $\{-27 \leq a_1 \leq -19.7$ e $-71.8 \leq a_2 \leq -31\}$
então VIRGINICA.
*R12:*Se $\{-17.4 \leq a_1 \leq -12.5$ e $-25 \leq a_2 \leq -1.1\}$
então VIRGINICA.
*R13:*Se$\{-26.4 \leq a_1 \leq -16$ e $-58.6 \leq a_2 \leq -14.6\}$
então VIRGINICA.
*R14:*Se $\{a_1 = -29.1$ e $a_2 = -84.6\}$
então VIRGINICA.
*R15:*Se $\{a_1 = -11.8$ e $a_2 = 5.9\}$
então VIRGINICA.

Figura 6. Regras para o *Iris Plants Database*

O AGAT foi avaliado em uma base de dados muito utilizada para testar algoritmos de *data mining*. Assim, pode-se comparar os resultados obtidos com alguns relatados na literatura. Um dos melhores resultados obtidos para este problema é encontrado em [128], em que se descreve uma árvore de classificação cuja precisão é igual a 97%. Além disso, o Algoritmo RX Modificado [22,81,82,83] também fornece esta mesma taxa de classificação correta média. Observa-se, então, que para este problema, o AGAT proporcionou um nível de precisão muito bom. Cumpre, ainda, citar que a aplicação do AGAT no espaço original das áreas fornece regras que classificam 93% dos exemplos de treinamento corretamente. Estes resultados foram publicados em [126].

3.2.2 Pima Indians Diabetes Database

Este exemplo, clássico na literatura de *data mining*, representa um complexo problema de classificação. O conjunto original de dados pertence ao *National Institute of Diabetes and Digestive and Kidney Diseases* e pode ser obtido em [129]. O conjunto de dados utilizado na literatura foi extraído deste conjunto original, considerando-se todos os indivíduos do sexo feminino cuja idade é maior ou igual a 21 anos.

O conjunto de dados contém 768 exemplos. Destes, 500 representam condições negativas para diabetes - valor de classe igual a zero - e 268 representam condições positivas - valor de classe igual a um. Cada exemplo é constituído de 8 atributos e do valor representativo da classe. Os atributos envolvidos são: A - número de vezes em que ficou grávida; B - concentração de glicose; C - pressão sangüínea; D - espessura da pele no tríceps; E - nível de insulina; F - índice de massa muscular; G - função de diabetes e H – idade.

Considerou-se o conjunto de dados desprovido de ruído. Para o treinamento da rede neural, separou-se o conjunto de dados em dois subconjuntos - treinamento e teste. Para tal, considerou-se 576 exemplos - 75% do total - para treinamento e 192 exemplos - 25% do total para teste. Os exemplos pertencentes a cada conjunto foram selecionados de maneira a se manter a proporção verificada no conjunto total - 65% de exemplos pertencentes à *classe zero* e 35% de exemplos pertencentes à *classe um*.

Desta forma, o conjunto de treinamento, formado randomicamente, é constituído por 576 exemplos (374 pertencentes à classe *zero* e 202 pertencentes à classe *um*) e o conjunto de teste é constituído por 192 exemplos (126 pertencentes à classe *zero* e 66 pertencentes à classe *um*). Observa-se, ainda, que os valores representativos da média

para cada variável são semelhantes nos dois conjuntos, bem como em relação ao conjunto original.

A melhor rede neural obtida em nossos experimentos possui quatro unidades de entrada com transformação linear [-1,+1], oito unidades ocultas com função de ativação *tangente hiperbólica* e uma unidade de saída com função de ativação *logística*. As variáveis usadas no melhor modelo obtido são: A, B, D e H. A descrição completa dos experimentos que envolvem o treinamento de redes neurais pode ser obtida em [22,81].

Para o conjunto de treinamento foi obtida uma taxa de classificação correta igual a 76,4% em 180 épocas enquanto que para o conjunto de teste a referida taxa é 82,85%. Para o conjunto completo, obtivemos uma taxa de 77,90%. Observa-se que o melhor resultado obtido para este problema de classificação se dá pela utilização do algoritmo *Logdisc*, cuja taxa de classificação correta, para o conjunto de teste é de 77,7% [35].

Após o treinamento da rede neural, os valores das ativações para cada unidade oculta foram calculados de acordo com as expressões listadas na Figura 7. Estes valores foram então padronizados e utilizados para se extrair regras do modelo de rede neural empregado. Utilizou-se de uma população formada por 200 genótipos. Além disso, foram utilizadas três tipos de funções objetivo, fazendo-se variar os valores de *E* e *I* da expressão (1). O *AGAT* é interrompido após 500 gerações, e então escolhe-se (em função da taxa de classificação correta média na base de dados) a melhor função objetivo. A Tabela 1 mostra os resultados obtidos para cada um destes exemplos (classe diabetes).

$a_1 = 51.31 - 0.91A - 0.35B + 0.07D - 0.56H$
$a_2 = 38.20 + 0.22A - 0.05B - 0.42D - 0.35H$
$a_3 = -0.17 + 0.02A + 0.01B - 0.01D - 0.02H$
$a_4 = -0.08 - 0.004A - 0.001D + 0.004H$
$a_5 = -7.88 + 0.76A + 0.24B - 0.46D - 0.26H$
$a_6 = 2.13 + 0.06A + 0.02B - 0.04D - 0.07H$
$a_7 = 40.15 + 0.18A + 0.11B + 0.15D - 0.91H$
$a_8 = 0.13 + 0.04A - 0.005D - 0.003H$

Figura 7. Expressões das Ativações - *Pima Indians Diabetes.*

O melhor resultado é obtido quando *E* e *I* possuem a mesma importância na função objetivo. O tempo de CPU foi igual a 16 minutos para a *classe um* e 36 minutos para a *classe zero* em um Pentium 233Mhz, 32RAM. Permitiu-se, então, ao AGAT procurar pela melhor solução, considerando-se que E=I=1, durante 1.000 gerações. Desta forma, a convergência se dá após 751 gerações para a *classe um* e após 982 gerações para a *classe zero.* A solução encontrada, no entanto, pouco acrescenta à qualidade da estrutura de conhecimento obtida, pois a taxa de classificação correta

média passa de 59% para 61%. Considerando-se a *classe zero* (saudável), o genótipo obtido representa 5 *grupos*, constituídos por 119, 118, 115, 117 e 31 objetos respectivamente. Considerando-se a *classe um* (diabetes), o genótipo obtido representa 5 *grupos*, constituídos por 54, 54, 52, 54 e 54 objetos respectivamente. Para comparar os resultados obtidos pelo *AGAT* com aqueles obtidos por meio do *Algoritmo RX Modificado* [22,81,82,83], considerar-se-á que a *classe zero* é a classe *default* [130]. Desta forma, os cinco *grupos* obtidos para a *classe um* representam as regras *R1, R2, R3, R4* e *R5* da Figura 8. A regra *R6* representa o valor *default*.

Tabela 1. Taxa de Classificação Correta Média para valores de *E* e *I (diabetes)*.

Tamanho da População	Convergência	E	I	Valor da Função objetivo	Taxa de Classificação Média
200	492	3	1	59.63	43%
200	489	2	1	16.09	45%
200	490	1	1	5.23	59%

R1: Se {(-56.47≤ a_1 ≤ -15.37) e (-8.58 ≤ a_2 ≤18.02) e (-0.32 ≤ a_3 ≤ 0.86)
e (-0.03 ≤ a_4 ≤ 0.14) e (-0.54 ≤ a_5 ≤ 27.62) e (-0.31 ≤ a_6 ≤ 2.26)
e (-4.83 ≤ a_7 ≤ 27.60) e (-0.03 ≤ a_8 ≤ 0.61)} então Diabetes;
R2: Se {(-46.55 ≤ a_1 ≤ -17.64) e (-0.23 ≤ a_2 ≤ 21.66) e (0.74 ≤ a_3 ≤ 1.49)
e (-0.07 ≤ a_4 ≤ 0.03) e (8.85 ≤ a_5 ≤ 35.12) e (1.69 ≤ a_6 ≤ 4.24)
e (15.31 ≤ a_7 ≤ 48.7) e (0.01 ≤ a_8 ≤ 0.57)} então Diabetes;
R3: Se {(-32.24 ≤ a_1 ≤ 7.01) e (-6.01 ≤ a_2 ≤ 15.13) e (0.28 ≤ a_3 ≤ 1.26)
e (-0.06 ≤ a_4 ≤ 0.00) e (-5.95 ≤ a_5 ≤ 12.49) e (0.76 ≤ a_6 ≤ 2.78)
e (26.86 ≤ a_7 ≤ 46.58) e (-0.13 ≤ a_8 ≤ 0.35)} então Diabetes;
R4: Se {(-45.84 ≤ a_1 ≤ 28.41) e (-34.43 ≤ a_2 ≤ 18.77) e (-1.17 ≤ a_3 ≤ 0.53)
e (-0.03 ≤ a_4 ≤ 0.08) e (-32.99 ≤ a_5 ≤ 14.48) e (-1.98 ≤ a_6 ≤ 1.88)
e (5.45 ≤ a_7 ≤ 33.12) e (-0.30 ≤ a_8 ≤ 0.53)} então Diabetes;
R5: Se {(-39.60 ≤ a_1 ≤ -0.21) e (13.07 ≤ a_2 ≤ 26.28) e (0.30 ≤ a_3 ≤ 1.05)
e (-0.05 ≤ a_4 ≤ 0.03) e (7.96 ≤ a_5 ≤ 28.17) e (1.82 ≤ a_6 ≤ 3.68)
e (15.10 ≤ a_7 ≤ 36.51) e (0.12 ≤ a_8 ≤ 0.64)} então Diabetes;
R6: Senão, saudável.

Figura 8. Regras Obtidas para o *Pima Indians Diabetes Database*.

Estas regras classificam 61% dos exemplos corretamente. Esta baixa taxa de classificação correta ocorre devido à representação distribuída. A Figura 9 mostra um exemplo deste fenômeno para a unidade oculta 2. Nesta, os primeiros 500 valores representam a classe zero enquanto que os últimos 268 representam a classe um. Todas as demais unidades ocultas apresentam um comportamento muito semelhante [22], isto

é, os valores das ativações para as duas classes estão distribuídos uniformemente no mesmo intervalo.

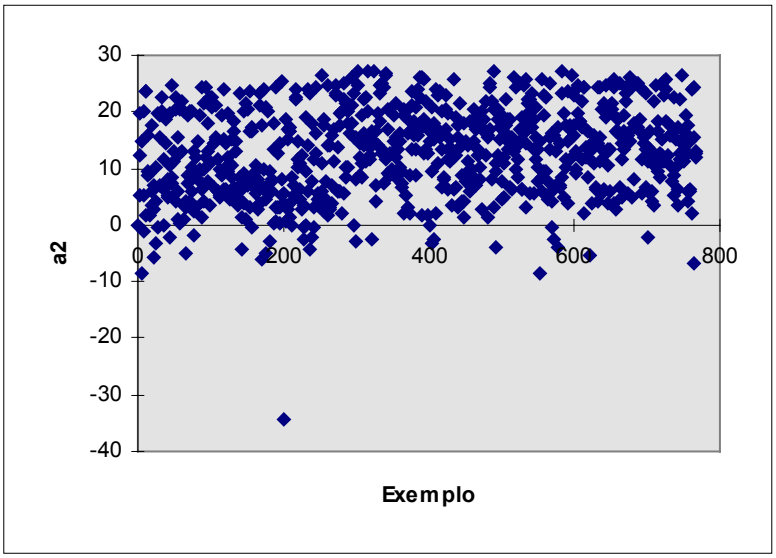

Figura 9. Ativações para a unidade oculta 2.

Os resultados obtidos para esse problema de classificação não são satisfatórios, principalmente considerando-se o fato de que na utilização de um conjunto de regras que descreva as duas classes se tem uma precisão igual a 31%. Entretanto, os resultados obtidos são superiores àqueles obtidos pelo *Algoritmo RX Modificado* [22,81,82,83], que proporciona 41% de precisão para um conjunto de regras, quando este descreve somente a classe *diabetes*. Conquanto a utilização de mais de cinco *grupos* ocasionaria melhores valores de precisão, ter-se-ia também um aumento na medida de complexidade. Além disso, a aplicação do *AGAT* no espaço original dos atributos proporciona um conjunto de regras - descrevendo somente a classe *diabetes* - que classifica corretamente 58,60% dos exemplos. Assim sendo, pode-se considerar que, para esta base de dados, o modelo de rede neural não proporciona um aumento significativo na qualidade da estrutura de conhecimento, comparativamente à aplicação do *AGAT* no espaço original dos atributos. Os resultados obtidos pelo AGAT descritos nessa seção foram publicados em [126].

3.2.3 Australian Credit Approval Database

Esta base de dados, que permite a simulação de uma aplicação de *data mining* na administração de cartões de crédito, contém 690 exemplos – 307 de pessoas que receberam crédito e 383 exemplos de pessoas que não receberam crédito e pode ser

encontrada em *http://www.ncc.up.pt/liacc/ML/statlog/datasets.html*. A base de dados contém 14 atributos – seis numéricos e 8 discretos - e o rótulo da classe. A *classe um* representa as pessoas que obtiveram crédito, enquanto que a *classe zero* representa as pessoas que não obtiveram crédito. Todos os nomes dos atributos estão convertidos em símbolos para proteger o sigilo sobre esses dados.

Diversas redes neurais supervisionadas foram treinadas. O conjunto de dados completo foi usado para treinamento. Permitiu-se um número máximo de 14 unidades ocultas. Este limite foi estabelecido visando-se limitar a complexidade das regras e considerando-se que o número de atributos é igual a 14. O modelo de rede neural que proporcionou o menor erro quadrático foi escolhido para o processo de extração de regras.

A rede neural selecionada contém 14 unidades de entrada utilizando uma transformação linear [-1,+1], 11 unidades ocultas usando a função logística e uma unidade de saída também utilizando a função logística. Utilizou-se uma taxa de aprendizado igual a 0,10 e um termo de momento igual a 0,10. Após 2.912 épocas (13 minutos em um Pentium 233, 32 RAM) o erro quadrático médio foi igual a 0,01292. As expressões de ativação das unidades ocultas estão listadas na Figura 10, onde as letras maiúsculas representam os atributos. Alguns deles não aparecem nestas expressões por possuírem coeficientes associados muito baixos.

Os valores das ativações das unidades ocultas para cada exemplo de treinamento foram calculados de acordo com as expressões descritas na Figura 10. Assim, obtém-se uma base de dados composta por 690 exemplos e 11 variáveis, correspondentes aos valores das ativações, e o rótulo da classe. Estes padrões de ativação foram, então, separados em dois conjuntos, cada um dos quais representando uma das classes. Em seguida, estes dois conjuntos são padronizados e o AGAT é aplicado para extrair as regras de classificação para cada uma das classes.

a_1 = -37.3 + 11.7A + 0.2B - 0.2C + 5.0D + 1.6E - 0.1F + 0.7G + 8.9H - 6.6I – 0.1J - 2.5K + 0.5L
a_2 = 23.4 + 0.7A + 1.1D - 0.7E - 0.2F + 0.1G - 7.8H - 6.1I - 0.1J - 3.1K - 1.8L
a_3 = -16.1 + 7.2A + 0.1B + 0.2C + 0.8D + E - F + 0.5G + 8.1H - 3.7I - 5.8K - 0.2L
a_4 = -26.5 + 11.3A + 0.1B - 0.4C - 3.8D + 0.6E + 0.2F + 0.2G + 16.4H - 0.8I + 0.1J + 10K - 2.9L
a_5 = -23.2 + 0.2A – 0.1B + 0.2C – 0.6D + 0.5E + 0.4F + 0.3G + 10.5H - 10.5I + 0.1J + 2.3K + 4L
a_6 = -14.5 + 2A + 0.2B + 0.1C - 5D + 0.7E – 0.5F + 5.2H – 1.4I + 0.1J + 8K – 0.9L
a_7 = 12.4 - 2.5A - 0.3C – 0.5D - 0.1E – 0.4F - 0.1G - 8.2H – 11.4I + 0.8K + 5.0L
a_8 = -13.6 - 3.9A - 0.1B + 0.1C + 1.9D + 1.1E + 0.3G + 8.2H - 4.8I + 0.1J - 4.5K - 1.3L
a_9 = 33.2 - 2.4A - 0.4C – 5.8D + 1.3E – 0.8F - 0.5G - 14.5H - 9I - 5.5K - 1.7L
a_{10} = -43.8 + 13.7A + 0.2B - 0.2C + 1.7D + 0.6E + 0.1F + 0.2G + 13H - 1.8I + 0.1J + 14.8K - 0.4L
a_{11} = -51.0 - 3.4A + 0.7C + 1.1D +1.5E - 0.6F + 0.1G + 20.1H - 3.5I + 0.1J - 3.8K + 13.4L

Figura 10. Expressões de Ativação para o *Credit Approval Database*.

Para extrair as regras do modelo de rede neural, utilizou-se de uma população constituída por 200 indivíduos e a função objetivo usada é dada pela expressão (1), considerando-se que E=I=1. O AGAT foi interrompido após 2.000 gerações. Considerando-se a classe um, o algoritmo converge após 1.481 gerações e o valor da função objetivo é igual a 9,5. O genótipo obtido representa cinco *grupos* - *C1, C2, C3, C4,* e *C5* – cujos números de objetos são iguais a 78, 76, 75, 77 e 1 respectivamente. Considerando-se a classe zero, o algoritmo converge após 1.948 gerações e o valor da função objetivo é igual a 22,4. O genótipo obtido representa cinco *grupos – C6, C7, C8, C9,* e *C10* – cujos números de objetos são iguais a 153, 154, 74, 1 e 1 respectivamente. O tempo de CPU foi igual a 36 minutos para a *classe um* e uma hora e 57 minutos para a *classe zero* (Pentium 233, 32 RAM).

Os *grupos* formados somente por um objeto – *C5, C9* e *C10* – foram considerados inadequados para o processo de descoberta de conhecimento, pois estes podem representar ruído nos dados. Entretanto, na prática esta estratégia merece uma atenção especial, ou seja, a validação por um especialista de domínio. Além disso, os demais *grupos* foram avaliados através de dois aspectos. Primeiramente, a regra descrita pelo *grupo* deve representar o maior número possível de exemplos da sua classe. Além disso, o *grupo* ideal não representa nenhum exemplo da outra classe, i.e. da classe que ele não representa. Desta maneira, utilizou-se da seguinte estratégia para classificar os *grupos*: considerando-se um *grupo* que representa a classe *C*, conta-se o número de exemplos da classe *C* representados pelo *grupo* e, a este valor, soma-se o número de exemplos não representados por *C* nas outras classes.

A Tabela 2 mostra os resultados da aplicação desta metodologia para todos os *grupos*, enquanto que a Tabela 3 mostra a taxa de classificação correta para diversas estruturas de conhecimento. Observa-se que o número de exemplos que cada classe modela – Tabela 2 – é diferente daquele previamente descrito. Isto acontece devido ao fato de que esta tabela considera que todos os objetos cujos atributos pertencem aos domínios do *grupo* pertencem ao próprio *grupo,* enquanto que o algoritmo apenas considera aqueles objetos realmente agrupados no processo de *clustering*. A Tabela 3 mostra que o melhor conjunto de regras que modela as duas classes é representado pelos três melhores *grupos* da *classe um* e os dois melhores *grupos* da *classe zero*. Obviamente que a consideração das duas regras que modelam somente a *classe zero* proporciona uma taxa de classificação correta maior (86,67%), mas esta metodologia

implica na utilização de uma classe *default (classe um)* o que de fato impede o conhecimento das características dos elementos pertencentes a esta classe.

Tabela 2. Classificação dos *Grupos.*

Grupo	Classe um	Classe zero	Total	Classificação
C1	109	313	422	3 (Classe1)
C2	110	361	471	2 (Classe1)
C3	122	356	478	1 (Classe1)
C4	123	281	404	4 (Classe1)
C6	192	293	485	2 (Classe2)
C7	221	300	521	1 (Classe2)
C8	88	118	206	3 (Classe2)

Tabela 3. Taxa de Classificação Média Correta.

Classe um	Classe zero	Classificação
4 melhores grupos	3 melhores grupos	44.78 %
3 melhores grupos	2 melhores grupos	72.61 %
2 melhores grupos	Melhor grupo	60.00 %
Melhor grupo	Melhor grupo	49.71 %
3 melhores grupos	----------	78.12 %
----------	2 melhores grupos	86.67 %
2 melhores grupos	----------	77.39 %
----------	Melhor grupo	75.51 %
Melhor grupo	----------	69.28 %

Considerando-se a importância de se utilizar regras que descrevam as duas classes, observa-se que a melhor solução para este problema de classificação se dá pelo emprego dos três melhores *grupos* da *classe um* e dos dois melhores *grupos* da *classe zero.* A estrutura de conhecimento obtida da utilização destes *grupos*, ilustrada por meio da Figura 11 proporciona uma taxa de classificação correta igual a 72,61%. O *AGAT* também foi aplicado no espaço original dos atributos e a taxa de classificação correta média obtida no conjunto de treinamento foi igual a 61% quando se considera um conjunto de regras que modela as duas classes. Ao se utilizar uma regra *default*, o AGAT proporciona, no espaço original dos atributos, uma taxa de classificação correta média igual a 81%. Os resultados obtidos no espaço original dos atributos foram publicados em [124], enquanto que os resultados obtidos no espaço das ativações foram publicados em [125]. Uma descrição mais detalhada dos resultados obtidos no presente exemplo pode ser encontrada em [123].

R1: Se {-20.9≤ a_1 ≤ 9.4 e -8.7 ≤ a_2 ≤ 15.6 e -16.6 ≤ a_3 ≤ 6.7 e –32.9≤ a_4 ≤ -2.1 e –19.2 ≤a_5≤ 6.4
e -18.6 ≤ a_6 ≤ -2.4 e -9.3 ≤ a_7 ≤ 23.6 e –13.9 ≤ a_8 ≤ 9.3 e -19.2 ≤ a_9 ≤ 36.0
e -31.5 ≤ a_{10} ≤ -2.9 e -18.0 ≤ a_{11} ≤ 22.7} então ***classe um;***

R2: Se {-4.5 ≤ a_1 ≤ 23.0 e -5.7 ≤ a_2 ≤ 13.3 e -5.9 ≤ a_3 ≤ 16.7 e -21.8 ≤ a_4 ≤ 6.3 e -14.8 ≤ a_5 ≤ 2.2
e -17.9 ≤ a_6 ≤ -0.4 e -15.2 ≤ a_7 ≤ 21.1 e -14.4 ≤ a_8 ≤ 3.4 e -30.2 ≤ a_9 ≤ 22.6
e -18.0 ≤ a_{10} ≤ 11.4 e -11.0 ≤ a_{11} ≤ 18.0} então ***classe um;***

R3: Se {1.4 ≤ a_1 ≤ 30.6 e -15.2 ≤ a_2 ≤ 10.9 e -5.1 ≤ a_3 ≤ 26.5 e -8.5 ≤ a_4 ≤ 20.1 e -7.2 ≤ a_5 ≤ 8.6
e -11.8 ≤ a_6 ≤ 7.6 e -16.7 ≤ a_7 ≤ 9.2 e -13.0 ≤ a_8 ≤ 7.8 e -22.8 ≤ a_9 ≤ 17.8
e -14.1 ≤ a_{10} ≤ 19.0 e -8.1 ≤ a_{11} ≤ 22.6} então ***classe um ;***

R4: Se {-25.4≤ a_1 ≤ 15.0 e 4.8 ≤ a_2 ≤ 21.0 e –17.0 ≤ a_3 ≤ 6.5 e –41.4 ≤ a_4 ≤ -0.6 e -20.8 ≤ a_5 ≤ -1.4
e -24.2 ≤ a_6 ≤ 1.8 e 3.9 ≤ a_7 ≤ 21.1 e –20.1 ≤ a_8 ≤ -0.1 e 4.0≤ a_9 ≤ 40.1 e -39.7 ≤ a_{10} ≤ 2.6
e -31.5 ≤ a_{11} ≤ -0.2} então ***classe zero;***

R5: Se {-34.6 ≤ a_1 ≤ 9.4 e 0.2 ≤ a_2 ≤ 21.7 e -23.6 ≤ a_3 ≤ -1.9 e –37.4 ≤ a_4 ≤ -1.3 e -27.2 ≤ a_5 ≤ -4.0
e -22.7 ≤ a_6 ≤ 0.1 e -0.4 ≤ a_7 ≤ 21.0 e –28.3 ≤ a_8 ≤ -9.8 e 1.1 ≤ a_9 ≤ 32.0 e -37.7 ≤ a_{10} ≤ 3.8
and -44.9 ≤ a_{11} ≤ -5.3} então ***classe zero***.

Figura 11. Conjunto de Regras para Avaliação de Crédito.

3.3 Comentários

Esta seção objetiva descrever, resumidamente, os principais resultados obtidos na aplicação do AGAT [123-126]. Assim sendo, observa-se que os resultados obtidos pelo AGAT no espaço das ativações, considerando-se as taxas de classificação corretas médias obtidas, foram melhores do que os resultados obtidos no espaço original dos atributos. A Tabela 4 descreve os principais resultados obtidos pelas regras extraídas nas três bases de dados utilizadas nos experimentos. Nesta tabela, pode-se constatar que o desempenho do AGAT também foi superior ao do Algoritmo RX Modificado [22,81,82,83]. Dessa forma, os resultados descritos neste capítulo motivaram o desenvolvimento de um algoritmo genético específico para enfrentar os problemas que envolvem *agrupamento*. Para tal, desenvolveu-se o AGAE – Algoritmo Genético de Agrupamento Específico – descrito no próximo capítulo.

Tabela 4. Taxas de Classificação Corretas Médias

Base de Dados	AGAT - Ativações	AGAT - Original	RX Modificado
Iris Plants	97%	93%	97%
Pima Indians Diabetes	61%	58,6%	41%
Australian Credit Approval	72,61%	61%	Não avaliado

CAPÍTULO 4

Algoritmo Genético de Agrupamento Específico

O capítulo anterior mostrou os resultados da aplicação do Algoritmo Genético de Agrupamento Tradicional (AGAT) em algumas bases de dados que normalmente são utilizadas em aplicações de *data mining*. Esses resultados foram satisfatórios, principalmente considerando-se o emprego de operadores tradicionais e de uma população inicial que objetivava somente formar genótipos de aptidão positiva. Além disso, o desempenho do AGAT foi superior ao desempenho do Algoritmo RX Modificado e esta constatação levou à consideração de que algoritmos genéticos de *agrupamento* podem ser realmente úteis para extrair regras de redes neurais supervisionadas. Assim sendo, desenvolveu-se um novo algoritmo genético de *agrupamento*, específico aos problemas de *clustering*, sendo este denominado de Algoritmo Genético de Agrupamento Específico (AGAE). O presente capítulo descreve este algoritmo e os resultados de sua aplicação em quatro bases de dados muito utilizadas para avaliar algoritmos de data mining. Além disso, o AGAE também foi avaliado em uma aplicação prática que envolve a modelagem de dados meteorológicos.

4.1 Codificação

Considerando-se que há N objetos para se agrupar, um fenótipo é representado por um vetor de (N+1) posições. Cada objeto pode ser numerado de 1 até N, então o i-ésimo gene representa o i-ésimo objeto, enquanto que o último gene representa o número de grupos do genótipo. Assim, cada gene de um indivíduo pertence ao alfabeto {1,2,3,...k} onde *k* é o número máximo de grupos. Por exemplo, considerando-se um conjunto de dados formado por 20 objetos, pode-se obter o seguinte agrupamento:

Genótipo1: 22345123453321454552 5

Isto significa que o grupo de rótulo "2" é formado por 5 objetos, correspondendo ao conjunto de objetos {1,2,7,13,20}, e o grupo de rótulo "1" tem 2 objetos (6 e 14). O último gene representa o número de grupos codificados pela solução, nesse caso igual a 5. Conforme já tinha sido anteriormente observado (Capítulo 3), este tipo de codificação

é altamente redundante [99], pois para cinco grupos existem 5! formas diferentes de se codificar a mesma solução para o problema de agrupamento, isto é, a codificação é do tipo "1 para n". Por exemplo, pode-se encontrar outros genótipos que também representam a mesma solução tal como:

Genótipo 2: 33451234514432515113 5
Genótipo 3: 44512345125543121224 5
Genótipo 4: 55123451231154232335 5
Genótipo 5: 11234512342215343441 5

Por isso, o tamanho do espaço que o AGAE precisa *varrer* é muito maior do que o espaço original das soluções. De fato, a função objetivo de um problema de agrupamento depende exclusivamente dos grupos de objetos, isto é, o que importa não é o rótulo do grupo mas sim os objetos que o mesmo contém.

Além disso, existe um outro problema ocasionado pela utilização deste esquema de representação quando aplicado com os operadores tradicionais de recombinação e mutação. Trata-se do efeito indesejável de considerar a informação dependente do contexto como se o contexto não existisse [99], que é o problema de *insensibilidade ao contexto*. Por exemplo, se o genótipo 2 fosse re-combinado com o genótipo 3, o genótipo resultante deveria ser igual aos seus pais, que representam o mesmo fenótipo. Mas isto raramente acontece. Para facilitar a visualização deste fenômeno, apliquemos o operador de recombinação em dois pontos aos seguintes genótipos:

Pai 1: 2 | 2222 | 111113333344444 4
Pai 2: 4 | 4444 | 333335555511111 4

Aplicando este operador nos pontos representados pelas barras verticais acima, os *genótipos pais* 1 e 2 proporcionariam os seguintes filhos:

Filho1: 2 *4444* 111113333344444 4
Filho 2: 4 *2222* 333335555511111 5

Estes filhos representam fenótipos diferentes dos seus pais. Por exemplo, o segundo filho representa cinco grupos, enquanto que os seus pais só representam quatro. Estes

problemas podem ser resolvidos aplicando-se o Algoritmo de Consistência, cuja descrição pode ser encontrada no Capítulo 3, e os resultados de sua aplicação em alguns problemas pode ser encontrados em [123-126]. No entanto, optou-se por desenvolver operadores genéticos adequados ao problema de agrupamento que não sofram dos problemas listados anteriormente. Basicamente, o AGAE consiste dos seguintes passos:

1) Formar uma população inicial de genótipos;
2) Avaliar cada genótipo da população;
3) Aplicar uma normalização linear;
4) Selecionar genótipos pelo método da roleta;
5) Aplicar a recombinação e a mutação;
6) Substituir os velhos genótipos pelos formados em 5);
7) Se a convergência foi alcançada, parar; senão, ir para o passo 2);

Figura 12. AGAE - Descrição Geral.

4.2 Função Objetivo

Determinar o número ótimo de *grupos* em um conjunto de dados é um dos mais difíceis aspectos do processo de agrupamento. A maioria das abordagens encontradas na literatura adota critérios que determinam o número de grupos baseando-se em agrupamentos obtidos. Entretanto, deseja-se que o AGAE otimize os *grupos* para determinado valor de *k* bem como o próprio valor de *k*. Assim sendo, deve-se maximizar a homogeneidade interna dos grupos e a heterogeneidade entre os grupos distintos, considerando-se que o AGAE também deve encontrar o *melhor* valor de *k*.

O AGAE emprega uma função objetivo baseada no comprimento médio de silhueta descrito em [105]. Em princípio, este conceito foi desenvolvido para determinar o número *correto* de grupos considerando-se que diversos agrupamentos diferentes tenham sido obtidos. Entretanto, entende-se que é possível empregar o mesmo conceito para estabelecer uma função objetivo adequada ao AGAE. Considerando-se que os algoritmos genéticos trabalham em paralelo e com diversas soluções distintas, então é razoável supor que o critério da silhueta média seja adequado para otimizar os agrupamentos de uma maneira mais dinâmica, isto é, durante o próprio processo de *agrupamento*.

Basicamente, considere-se um objeto "i" pertencente ao grupo A. Então a dissimilaridade média de "i" em relação a todos os outros objetos de A pode ser denotada por a(i). Considere-se um grupo diferente C e calculemos a dissimilaridade média de "i" em relação a todos os objetos de C, que será denotada por d(i,C). Após calcular d(i,C) para todos os grupos C≠A, seleciona-se a menor destas, que será aqui denotada por b(i), i.e.:

$$b(i) = \min d(i,C) \text{ , } C \neq A \text{ .} \qquad (8)$$

Este número representa a dissimilaridade de "i" em relação ao seu grupo vizinho. Assim define-se a silhueta s(i) como:

$$s(i) = \frac{b(i) - a(i)}{\max \{ a(i) , b(i) \}} . \qquad (9)$$

Quando o grupo A contém somente um objeto, considera-se que s(i) é igual a zero, que é a escolha mais neutra possível [105]. Além disso, pode-se observar que $(-1 \leq s(i) \leq 1)$. A função objetivo representa a média de s(i) para i=1,2,...,N. Neste caso, o melhor valor de *k* acontecerá quando o valor de s(i) médio for máximo. Além disso, para proporcionar uma comparação mútua entre os cromossomos, aplica-se uma normalização linear [100,101] aos valores da função objetivo. Primeiramente os valores de s(i) para cada genótipo são classificados em ordem crescente e então substituídos por:

$$F' = \frac{R(i) [F'(max) - F'(min)]}{\text{Tamanho da População}} \qquad (10)$$

Onde *R(i)* é a posição do genótipo "i" na classificação , *F'(max)* é o maior valor da função objetivo e *F'(min)* é o menor valor da função objetivo.

4.3 Seleção

Os indivíduos que fazem parte de cada geração são selecionados de acordo com o método da roleta. Como esta técnica não permite valores negativos para a função objetivo, uma constante igual a 1,0 é somada aos valores da função objetivo. Assim,

para o método de seleção usado temos que ($0 \leq s(i) \leq 2$). Além disso, utiliza-se de uma estratégia elitista, isto é, o melhor indivíduo de cada geração é sempre copiado para a geração seguinte. Terminado o processo de seleção, soma-se (-1) a todos os valores da função objetivo, de modo que os valores de s(i) mostrados pelo programa implementado pertencem ao intervalo [-1,+1].

4.4 Operadores

Operadores genéticos clássicos não são adequados para problemas de agrupamento [99] porque eles usualmente trabalham com conjuntos de genes não relacionados. Em problemas de *agrupamento*, os genes estão relacionados, isto é, genes de alelos iguais significam que os objetos por eles representados pertencem aos mesmos grupos e estes são os blocos de construção significativos de uma solução. Assim, é necessário usar operadores que trabalhem com grupos de genes, em vez de se trabalhar com os genes propriamente ditos. Estes operadores são referenciados como *operadores orientados a grupos* [99]. Surge, então, a necessidade de se definir novos operadores. Assim, desenvolveu-se um operador de recombinação e dois operadores de mutação.

4.4.1 Recombinação

O operador de recombinação combina partes de informação provenientes de genótipos diferentes. Basicamente, alguns grupos de um genótipo A são copiados em B e vice-versa. O operador funciona da seguinte maneira:

1) Selecionar dois genótipos (A e B);
2) Considerando que A representa $k1$ *grupos*, escolher aleatoriamente n ($1 \leq n \leq k1$) grupos para copiar em B. Os grupos não alterados de B são mantidos e aqueles alterados têm seus objetos re-posicionados ao grupo de centróide mais próximo. Desta maneira se obtém o filho C.
3) Para obter o filho D, nós copiamos neste todos os grupos de B que foram alterados no passo 2). Simula-se, então, a cópia destes grupos no genótipo A, alterando-se, por conseqüência, alguns grupos deste genótipo. Os grupos inalterados de A são copiados em D e os objetos que não pertencem a nenhum grupo são re-posicionados ao grupo de centróide mais próximo.

Para ilustrar este procedimento, vamos considerar os seguintes genótipos em que os últimos genes, correspondentes aos números de grupos da solução, não estão presentes:

A – 1123245125432533424
B – 1212332124423221321

Pode-se selecionar aleatoriamente, por exemplo, os grupos 2 e 3 (que estão em negrito) em A:

A – 11**232**451**2**54**32**5**33**4**2**4
B – 1212332124423221321

Estes grupos modificam os grupos 1, 2 e 3 em B, enquanto que o grupo 4 não é modificado:

A – 11**232**451**2**54**32**5**33**4**2**4
B – **12123321**244**23221321**

Os alelos dos genes sublinhados serão mudados, em C, porque eles também pertencem aos grupos modificados em B. Assim, após a cópia dos grupos selecionados de A {2,3} tem-se o seguinte genótipo:

C – 0023200020032033020

Os genes de C completados com alelos iguais a zero indicam posições com potencial para receber grupos não alterados de B ou representam genes de objetos que serão re-posicionados no grupo de centróide mais próximo. Por exemplo, o grupo 4 é mantido porque o mesmo não foi mudado em B. Desta forma, o processo de recombinação resulta no seguinte genótipo:

C - 0023200024432033020

Calculam-se, então, os centróides de cada um desses grupos {2,3,4} e os outros objetos – cujos alelos ainda estão zerados - são colocados no grupo de centróide mais próximo. Considerando o filho D, observa-se que este será formado pelos grupos 1,2 e 3 de B – que foram alterados pela cópia dos grupos de A em B - mais aqueles grupos inalterados de A quando ocorre a cópia dos grupos {1,2,3} de B em A, isto é, os grupos alterados em B quando da formação de C são também copiados em D :

D - 1212332120023221321

Desta maneira, observa-se que, neste exemplo, todos os grupos de A são mudados e assim os objetos restantes são posicionados no grupo de centróide mais próximo. Além disso, o operador de re-combinação é aplicado em 50% dos genótipos de cada geração. Mais especificamente, o primeiro genótipo selecionado pelo método da roleta é re-combinado com o genótipo *n/2*, em que *n* representa o número de genótipos selecionados, o segundo genótipo é re-combinado com o genótipo (*n/2-1*) e assim por diante. Além disso, observa-se que os genótipos filhos representam um número de *grupos* que varia no intervalo $[k_1,k_2]$, sendo k_1 e k_2 respectivamente o menor e o maior número de *grupos* representados pelos genótipos pais.

4.4.2 Mutação

Existem dois operadores para mutação. O operador 1 (que somente opera em agrupamentos formados por mais de dois grupos) elimina aleatoriamente um grupo e coloca todos os seus objetos no grupo de centróide mais próximo. O operador 2 divide um grupo selecionado aleatoriamente em dois novos grupos. O primeiro sub-grupo será formado pelos objetos mais próximos do centróide, enquanto que o segundo sub-grupo será formado pelos objetos mais próximos do objeto mais distante do centróide. Isto é, duas sementes são escolhidas: a mais próxima ao centróide e a mais distante do centróide. Feito isto, cada objeto do grupo é deslocado para o grupo da semente mais próxima.

Estes operadores (recombinação e mutação) são adequados para problemas de *agrupamento* porque eles somente mudam os genótipos na menor medida possível, isto é, dividindo, eliminando e reunindo grupos na esperança de encontrar melhores soluções para o problema em questão. Além disso, 25% são mutados pelo operador 1 e 25% pelo operador 2. Conforme observado na seção anterior, o operador de re-combinação é aplicado aos primeiros 50% dos n genótipos selecionados, isto é aos genótipos cuja *numeração de seleção* pertença ao intervalo [1,n/2]. O operador de mutação 1, por sua vez, é aplicado aos genótipos que pertencem ao intervalo [$n/2+1$, $3n/4$], enquanto que o operador 2 é aplicado aos genótipos que pertencem ao intervalo [$3n/4+1$, n].

4.5 População Inicial

A população inicial é gerada de acordo com algumas sementes (objetos representativos) previamente escolhidas, ou seja, todos os objetos são agrupados de acordo com as suas proximidades às sementes escolhidas. Assim, o primeiro genótipo contém dois grupos (formados por duas sementes), o segundo genótipo contém três grupos (três sementes) e assim por diante, de modo que o j-ésimo genótipo contém (j+1) grupos. Em cada genótipo, os objetos da base de dados são agrupados de acordo com as proximidades relativas às sementes. Assim, para escolher as sementes, adota-se a metodologia desenvolvida em [105]:

1) A primeira semente é o objeto cuja distância média em relação aos outros objetos da base de dados seja mínima, isto é, a primeira semente é o objeto mais *central* do

conjunto de dados. As demais sementes são selecionadas de acordo com os passos abaixo:

2) Consideremos um objeto "i" não selecionado:

2.1) Considere um objeto "j" não selecionado. Consideremos que a sua dissimilaridade com o objeto mais próximo previamente selecionado seja chamada de D_j, e que sua dissimilaridade com o objeto "i" seja chamada de d(j,i). Assim, podemos calcular a diferença [D_j – d(j,i)].

2.2) Se esta diferença é positiva, o objeto "j" irá contribuir para a decisão de selecionar o objeto "i". Assim, calcula-se a contribuição do objeto "j" para a seleção de "i" como semente da seguinte maneira: C_{ji} = máx $\{D_j - d(j,i),0\}$.

2.3) Pode-se, então, calcular o ganho total obtido em se escolher o objeto "i" para semente de um grupo por meio de:

$$G(i) = \sum_j C_{ji}$$

2.4) Escolhe-se o objeto "i"que maximize o valor de G(i).

Este processo é repetido até que se tenham *k* sementes. Na verdade, o usuário do AGAE define o número máximo desejado de grupos (*k*), mas o programa implementado somente considera as sementes para as quais o valor de G(i) é maior do que zero. Desta forma, pode acontecer que o número máximo de sementes, que também define o número de grupos, seja menor do aquele previamente definido pelo usuário. Observa-se, ainda, que o número de genótipos utilizados pelo AGAE é igual ao número de sementes selecionadas menos um. Isto é, o primeiro genótipo representa dois grupos, dividindo a base de dados de acordo com a proximidade às sementes selecionadas; o segundo genótipo representa três grupos e assim por diante até finalmente o genótipo (k-1), que representa *k* grupos.

A metodologia utilizada é dependente da ordem em que as sementes são selecionadas e, obviamente, pode-se questionar o método de escolha da primeira semente. Assim sendo, esta escolha é de certa forma *arbitrária*. Observa-se, ainda, que na metodologia descrita em [105] há uma segunda fase do algoritmo que determina a possibilidade de existirem sementes melhores do que as selecionadas pelo método acima descrito. Assim, KAUFMAN e ROUSSEEUW [105] desenvolveram o algoritmo SWAP, que considera todos os pares de objetos (i,h) em que "i" foi selecionado e "h"

não foi selecionado, para determinar se a substituição da semente “i” pelo objeto “h” é vantajosa. Para avaliar a qualidade do agrupamentos, calcula-se a soma das distâncias entre os objetos e as sementes mais próximas. Desta forma pode-se avaliar se a substituição da semente “i’pela “h”é proveitosa. A título de ilustração, seguem os passos computados por este algoritmo:

1) Considere um objeto “j” não selecionado e calcule a sua contribuição C_{jih} para a substituição de *i* por *h*:

   a) Se “j” está mais distante tanto de “i” quanto de “h” em relação a um dos outros objetos representativos (sementes), então C_{jih} é igual a zero; (“j” está mais perto de outra semente do que de “i” e de “h”)

   b) Se “j” não está mais distante de “i” do que de qualquer outra semente , ou seja $d(j,i)=D_j$, ou ainda, pode-se dizer que a distância de “j” até “i” é menor ou igual à distância de “j” a qualquer outra semente. Neste caso, duas situações são consideradas:

      b.1) “j” está mais perto de “h” do que da segunda semente mais próxima, isto é, $d(j,h)<Ej$, onde E_j é a dissimilaridade entre “j” e a segunda semente mais próxima – normalmente “i”. Neste caso a contribuição do objeto “j” para a mudança da semente é: $C_{jih} = d(j,h) – d(j,i)$;

      b.2) “j” está no mínimo tão distante de “h” quanto da segunda semente mais próxima, isto é, $d(j,h) \geq E_j$ - distância de “j” até “h” é maior ou igual à distância de “j” até a segunda semente mais próxima; Neste caso a contribuição do objeto “j” para a substituição é igual a $C_{jih} = E_j – D_j$.

   Observa-se que na situação b.1) a contribuição C_{jih} pode ser positiva ou negativa, dependendo da posição de “i, j, h”.Somente se o objeto “j” estiver mais próximo de “i” do que de “h” esta contribuição será positiva, indicando que a substituição não é vantajosa relativamente ao objeto “j”. Na situação b.2), a contribuição será sempre positiva porque não seria vantajoso substituir “i” pelo objeto “h” mais distante de “j” do que da segunda semente mais próxima.

c) "j" está mais distante de "i" do que pelo menos uma das outras sementes, mas mais perto de "h" do que de qualquer outra semente. Neste caso a contribuição de "j" para a substituição é $C_{jih}=d(j,h)-D_j$.

2) Calcular o resultado total da substituição T_{ih} somando os valores de C_{jih}, ou seja, $T_{ih}=\sum_j C_{jih}$.

3) Selecionar o par (i,h) que minimiza T_{ih}.

4) Se o menor valor de T_{ih} é negativo, a substituição de "i" por "h" é efetuada e o algoritmo volta ao passo 1). Se $T_{ih} \geq 0$, o algoritmo pára.

4.6 Vantagens do AGAE e Detalhes de Implementação

Levando-se em consideração que o AGAE se baseia na metodologia desenvolvida por KAUFMAN e ROUSSEEUW [105], surge a pergunta: por que não usar esta mesma metodologia? A resposta surge da complexidade computacional de se utilizar esta metodologia em bases de dados nas quais tanto o número de objetos quanto o número de grupos são elevados. Nesse sentido, o AGAE busca pela solução ótima numa região em que ela tem maiores chances de aparecer, não precisando avaliar todos os pares de substituições possíveis. Mais especificamente, o algoritmo original obteria várias soluções para cada valor de *k* (número de grupos) tendo de avaliar as melhores sementes para cada uma delas, enquanto que o AGAE procura otimizar o valor de *k* e a estrutura dos grupos concomitantemente. Utilizando-se o AGAE, espera-se que não seja necessário *varrer* todo o espaço de busca para se obter soluções razoáveis, mas somente aquelas regiões mais promissoras. Além disso, deseja-se obter regras que descrevam a base de dados, de modo que a determinação das sementes *ótimas* não é tão importante para a abordagem utilizada no presente trabalho. Conquanto as sementes *ótimas* automaticamente levem aos grupos *ótimos*, não é necessário conhecê-las desde que os grupos *ótimos* sejam conhecidos.

Em outras palavras, a metodologia descrita em [105] seleciona *k* sementes otimizadas através de uma busca exaustiva para cada valor de *k*, obtém os grupos para cada um destes valores e, por fim, escolhe o melhor agrupamento. O AGAE simplifica o processo, procurando varrer as regiões mais promissoras do espaço de busca, tanto em relação ao valor de *k*, quanto em relação aos melhores grupos para este valor.

No concernente à implementação, é necessário que se façam algumas observações:

(i) o AGAE foi implementado em linguagem C padrão ANSI;

(ii) Sempre que um genótipo possuir dois grupos, o operador de mutação 1 não é aplicado e o genótipo em questão não é alterado;

(iii) as distâncias entre os objetos são calculadas no início do programa e armazenadas em uma matriz que é usada durante toda a execução do programa;

(iv) os genótipos formados por um único grupo são substituídos pelo melhor genótipo da população inicial. Na eventualidade de existir apenas genótipos formados por um único grupo, o algoritmo é interrompido.

4.7 Exemplos Didáticos – Ruspini e Dados Aleatórios

Os exemplos desta seção têm por objetivo ilustrar a metodologia utilizada pelo AGAE. Neste sentido, serão utilizadas duas bases de dados: Ruspini Dataset e 200 Randomly Generated. A descrição de ambas pode ser encontrada em [105]. Utilizou-se da medida *Euclidiana* para calcular as distâncias entre os objetos. Realizou-se dois tipos básicos de experimentos: inicialmente, aplicou-se o AGAE nas bases de dados completas, utilizando-se de uma população inicial formada por 20 genótipos gerada aleatoriamente. Neste caso, os genótipos iniciais representavam 2 grupos aleatórios no primeiro cromossomo, 3 no segundo, ..., e 21 no último, sendo todos eles gerados aleatoriamente. Este experimento teve por objetivo mostrar a eficácia dos operadores implementados. Em seguida aplicou-se o AGAE nos exemplos de cada um dos grupos objetivando-se a obtenção de sub-grupos. Neste caso, a população inicial não foi gerada aleatoriamente, mas baseando-se na metodologia descrita em [105]. A metodologia para extrair regras de redes neurais supervisionadas se baseia na segunda abordagem e pode ser encontrada na seção 1.3.

4.7.1 Eficácia do AGAE - grupos na base de dados completa

Considerando-se que as bases de dados são relativamente simples e pequenas, o AGAE foi aplicado em dez simulações. Em cada uma das simulações foram utilizadas populações formadas por 20 genótipos, o que significa que as populações iniciais foram formadas por genótipos que codificam de 2 a 21 *grupos*. Além disso, estabeleceu-se um número máximo de gerações, isto é, interrompe-se o AGAE após 200 gerações. A

Tabela 5 ilustra o desempenho do AGAE em cada uma das simulações em termos do número de gerações necessárias para o AGAE encontrar os *agrupamentos* corretos, isto é, o número de gerações necessárias para o AGAE encontrar o maior valor possível para a função objetivo (correspondentes aos agrupamentos ótimos).

Tabela 5. Resultados obtidos para populações iniciais geradas aleatoriamente.

Simulação	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**Média**	**Desvio Padrão**
Ruspini	13	24	61	28	73	83	45	54	33	30	44	22.8
Aleatórios	14	11	12	16	14	13	4	12	8	16	12	3.7

4.7.1.1 Ruspini

Esta base de dados é formada por 75 objetos descritos por duas variáveis (x e y). O agrupamento correto é formado por 4 grupos, como pode ser observado na Figura 13. O AGAE encontrou o agrupamento correto na média após 44 gerações com a função objetivo assumindo o valor 0,74 , o que indica que uma forte estrutura foi encontrada [105].

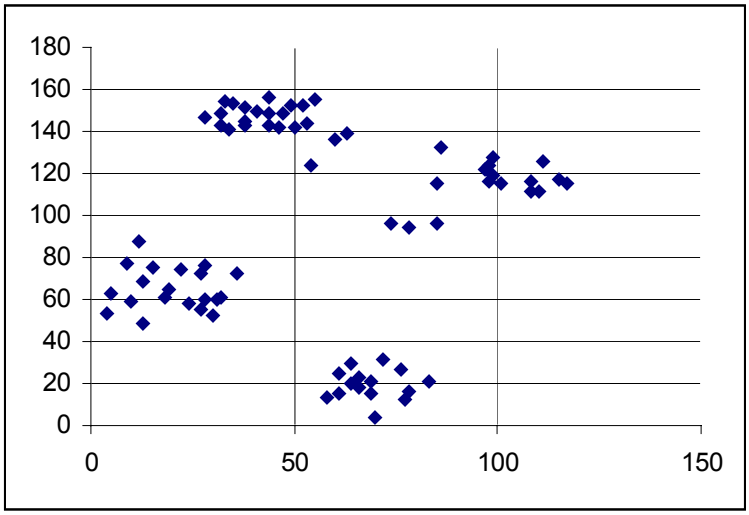

Figura 13. Ruspini Data.

4.7.1.2 Dados Aleatórios

Esta simulação envolve 200 objetos gerados aleatoriamente e descritos por meio de duas variáveis. Existem três grupos de objetos gerados por uma distribuição normal esférica de acordo com as características abaixo:

Grupo 1:	120 objetos	$\mu_x = 0$	$\mu_y = 10$	$\sigma = 1.7$
Grupo 2:	60 objetos	$\mu_x = 20$	$\mu_y = 12$	$\sigma = 0.7$
Grupo 3:	20 objetos	$\mu_x = 10$	$\mu_y = 20$	$\sigma = 1.0$

Os grupos para este problema estão ilustrados na Figura 14. O AGAE encontrou o agrupamento correto na média após 12 gerações com a função objetivo assumindo o valor 0,82 , o que também indica que uma forte estrutura foi encontrada [105].

No concernente à tarefa de extração de regras para estes dois exemplos, pode-se verificar facilmente que os grupos podem ser descritos por retângulos que fornecem uma taxa de classificação correta média igual a 100%, pois nestes casos não há sobreposição de domínios. Estes dois exemplos mostram a eficácia dos operadores desenvolvidos, haja vista que as soluções corretas foram encontradas a partir de soluções iniciais geradas aleatoriamente.

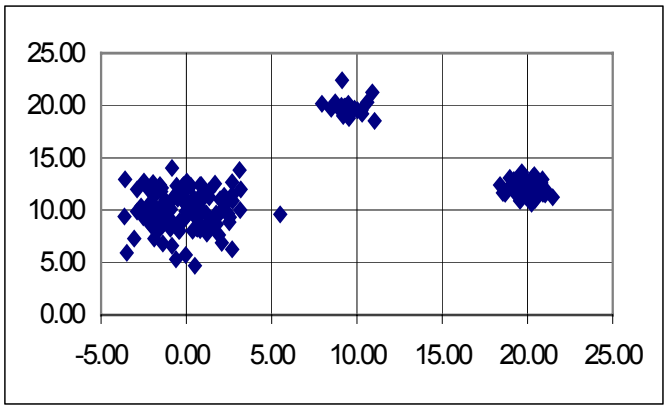

Figura 14. Dados Aleatórios.

4.7.2 Eficácia do AGAE em encontrar sub-grupos

Neste exemplo, o interesse recai em modelar cada classe separadamente, isto é, deseja-se obter sub-grupos que descrevam cada classe. Esta abordagem pode ser útil para a tarefa de extração de regras de redes neurais, dado que nestes casos é comum a ocorrência de classes cujos valores dos atributos estão superpostos. Para tal, utilizou-se de populações de tamanho variável, com o número máximo de sub-grupos sendo igual ao número de exemplos da classe em questão. Nesse sentido, tem-se que o número de genótipos da população é igual ao número de sub-grupos menos um. Cumpre observar que o número de sementes selecionadas pode ser menor do que o número máximo de sub-grupos definido a priori, i.e. quando $G(i) \leq 0$. Além disso, quatro experimentos foram

realizados: inicialmente, obtém-se a solução inicial obtida pelo AGAE, isto é, a solução obtida após a geração inicial. Em seguida, obtém-se as soluções após 200, 1.000 e 2.000 gerações.

4.7.2.1 Ruspini

Neste exemplo, aplicou-se o AGAE em cada classe separadamente, isto é, os grupos foram separados a priori e o AGAE foi então aplicado em cada grupo separadamente. Desta forma, aplicou-se o AGAE quatro vezes. Observa-se, entretanto, que o presente exemplo proporciona, comparativamente ao exemplo anterior, um conjunto de regras com maior complexidade e de mesma precisão. Neste caso, prefere-se usar as regras obtidas anteriormente, dado que as mesmas são mais simples e fornecem a mesma taxa de classificação correta média. Neste caso, no primeiro grupo (G1), composto por 20 objetos, o AGAE obteve 11 sementes, o que corresponde a 10 genótipos. Para o segundo grupo (G2) composto por 23 objetos, o AGAE também obteve 11 sementes, o que corresponde a 10 genótipos. Para o terceiro grupo (G3), composto por 17 objetos, e para o quarto grupo (G4), composto por 15 objetos, o AGAE obteve 8 sementes, formando populações de 7 genótipos.

a) Geração Inicial

Para o primeiro grupo (G1), foram obtidos os *clusters* c1, c2, c3, c4 e c5 (valor da função objetivo igual a 0,38). Para o segundo grupo (G2), foram obtidos os *clusters* c6 e c7 (valor da função objetivo igual a 0,39). Para o terceiro grupo (G3), foram obtidos os *clusters* c8 e c9 (valor da função objetivo igual a 0,60). Para o quarto grupo (G4), foram obtidos os *clusters* c10 e c11 (valor da função objetivo igual a 0,36). Estes *clusters* estão ilustrados na Figura 15. Assim sendo, seriam obtidas as seguintes regras:

R1: Se {4≤x≤13 e 49≤y≤63}	então G1;	(c1 – 4 objetos)
R2: Se {9≤x≤15 e 69≤y≤88}	então G1;	(c2 – 4 objetos)
R3: Se {18≤x≤19 e 61≤y≤65}	então G1;	(c3 - 2 objetos)
R4: Se {22≤x≤36 e 72≤y≤76}	então G1;	(c4 - 4 objetos)
R5: Se {24≤x≤32 e 52≤y≤61}	então G1;	(c5 - 6 objetos)
R6: Se {28≤x≤55 e 141≤y≤156}	então G2;	(c6 - 17 objetos)
R7: Se {46≤x≤63 e 124≤y≤144}	então G2;	(c7 - 6 objetos)

R8: Se {85≤x≤117 e 111≤y≤132} então G3; (c8 - 14 objetos)

R9: Se {74≤x≤85 e 94≤y≤96} então G3; (c9 - 3 objetos)

R10: Se {70≤x≤83 e 4≤y≤21} então G4; (c10 - 4 objetos)

R11: Se {58≤x≤76 e 13≤y≤31} então G4; (c11 - 11 objetos)

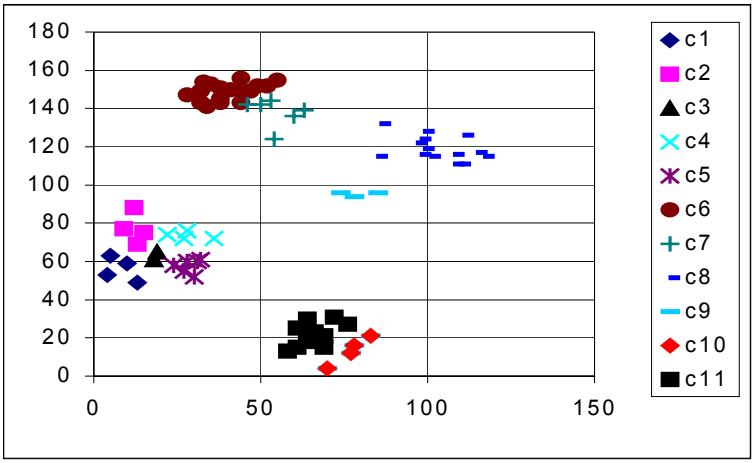

Figura 15. Geração Inicial.

b) 200 gerações

Para o primeiro grupo (G1), foram obtidos os *clusters* c1, c2, c3 e c4 (valor da função objetivo igual a 0,43) após 4 gerações. Para o segundo grupo (G2), foram obtidos os *clusters* c5 e c6 (valor da função objetivo igual a 0,49) após 2 gerações. Para o terceiro grupo (G3), foram obtidos os *clusters* c7 e c8 (valor da função objetivo igual a 0,60) na primeira geração. Para o quarto grupo (G4), foram obtidos os *clusters* c9 e c10 (valor da função objetivo igual a 0,36) após a primeira geração. Estes grupos estão ilustrados na Figura 16. Além disso, o AGAE não apresenta melhores resultados após 1.000 gerações e após 2.000 gerações. Assim sendo, seriam obtidas as seguintes regras:

R1: Se {4≤x≤13 e 49≤y≤63} então G1; (c1 – 4 objetos)

R2: Se {9≤x≤15 e 69≤y≤88} então G1; (c2 – 4 objetos)

R3: Se {18≤x≤32 e 52≤y≤65} então G1; (c3 - 8 objetos)

R4: Se {22≤x≤36 e 72≤y≤76} então G1; (c4 - 4 objetos)

R5: Se {28≤x≤55 e 141≤y≤156} então G2; (c5 - 20 objetos)

R6: Se {54≤x≤63 e 124≤y≤139} então G2; (c6 - 3 objetos)

R7: Se {85≤x≤117 e 111≤y≤132} então G3; (c7 - 14 objetos)

R8: Se {74≤x≤85 e 94≤y≤96} então G3; (c8 - 3 objetos)

R9: Se {70≤x≤83 e 4≤y≤21} então G4; (c9 - 4 objetos)

R10: Se {58≤x≤76 e 13≤y≤31} então G4; (c10 - 11 objetos)

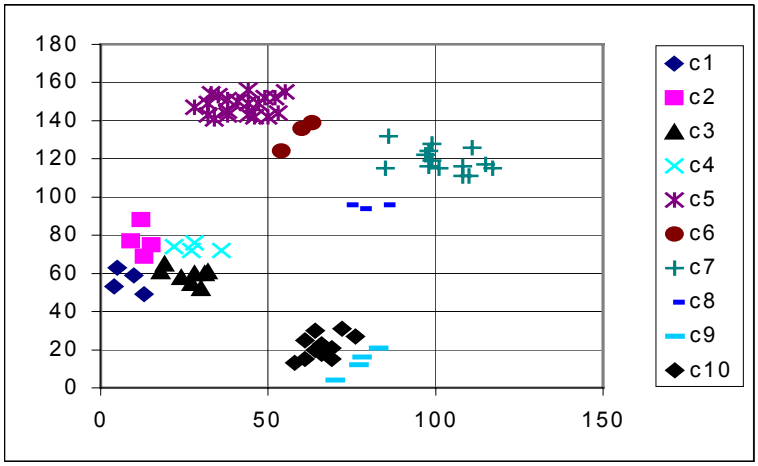

Figura 16. 200, 1.000 e 2.000 gerações.

Pode-se descrever o problema de extração de regras de associação como sendo a geração de todas as regras que possuem valores de suporte e de confiança maiores do que aqueles estabelecidos pelo usuário. O "suporte" diz respeito ao número de exemplos modelados pela regra dividido pelo número total de exemplos da base de dados. A "confiança" diz respeito ao quociente entre o número de exemplos pertencentes à classe em questão que respeitam as premissas e o número de exemplos que se enquadram nas premissas da regra [131,132,133].

Neste exemplo, a regra R6 modela três exemplos (suporte = 3/75 = 4% e confiança = 3/3 = 100%) do grupo G2 da mesma forma que a regra R8 modela três exemplos (suporte = 3/75 = 4% e confiança = 3/3 = 100%) do grupo G3. Considerando-se a metodologia para a obtenção das regras utilizada, percebe-se que a confiança de uma regra só será menor do que 100% nos casos em que houver sobreposição de domínio. A eliminação das regras de baixo coeficiente de suporte (por exemplo considerando-se um suporte mínimo igual a 5%) proporcionaria um conjunto de regras mais simples mas com uma precisão igual a 92%.

Cumpre enfatizar que, em aplicações de *data mining*, as regras normalmente são obtidas em um conjunto de treinamento e avaliadas em um conjunto de teste. Esta metodologia não foi aplicada a este exemplo devido ao tamanho reduzido da base de dados e pela suposição de que os dados não contém ruídos. Além disso, os grupos c6 e c8 podem representar justamente as regras interessantes e imprevisíveis procuradas.

Este problema também é conhecido como o problema dos itens raros [131]. Alternativamente, pode-se agrupar as regras R1,...,R10 obtendo-se o conjunto de regras abaixo:

Se {4≤x≤13 e 49≤y≤63} ou {9≤x≤15 e 69≤y≤88} ou {18≤x≤32 e 52≤y≤65}
ou {22≤x≤36 e 72≤y≤76} então G1;
Se {28≤x≤55 e 141≤y≤156} ou {54≤x≤63 e 124≤y≤139} então G2;
Se {85≤x≤117 e 111≤y≤132} ou {74≤x≤85 e 94≤y≤96} então G3;
Se {70≤x≤83 e 4≤y≤21} ou {58≤x≤76 e 13≤y≤31} então G4;

Observa-se, para o exemplo em questão, que a aplicação do AGAE na base de dados completa proporciona um conjunto de regras mais simples mas de mesma precisão do que as regras acima descritas.

4.7.2.2 Dados Aleatórios

Neste exemplo, o AGAE selecionou 62 sementes, resultando numa população formada por 61 genótipos para o grupo *um*, 30 sementes (29 genótipos) para o grupo *dois* e 11 sementes (originando 10 genótipos) para o grupo *três*.

a) Geração Inicial

Considerando-se o primeiro grupo (120 objetos), obteve-se 20 *clusters* e a função objetivo assumiu o valor 0,34 após a primeira geração. Para o grupo dois, constituído de 20 objetos, obteve-se dois *clusters* e o valor da função objetivo foi igual a 0,35. No grupo 3 (60 objetos), obteve-se dois *clusters* e o valor máximo da função objetivo foi igual a 0,41. Os grupos obtidos após a primeira geração podem ser visualizados por meio da Figura 17.

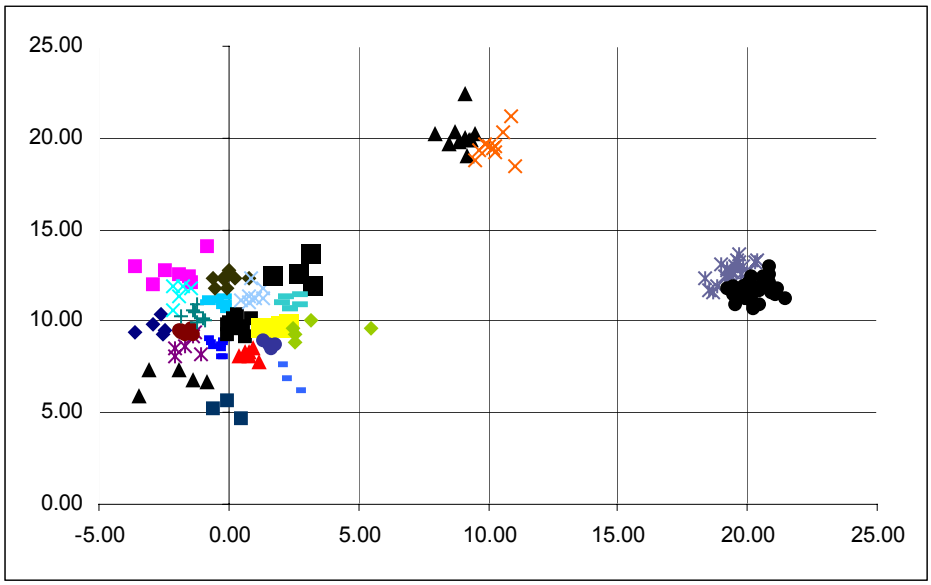

Figura 17. Dados Aleatórios - Solução Inicial.

b) 200 gerações

Considerando-se o primeiro grupo, obteve-se o melhor valor da função objetivo, igual a 0.39, após 193 gerações. Considerando-se o segundo grupo, obteve-se o melhor valor da função objetivo, igual a 0.51, após 6 gerações. Considerando-se o terceiro grupo, obteve-se o melhor valor da função objetivo, igual a 0.41 na geração inicial. O agrupamento para este exemplo pode ser visualizado na Figura 18.

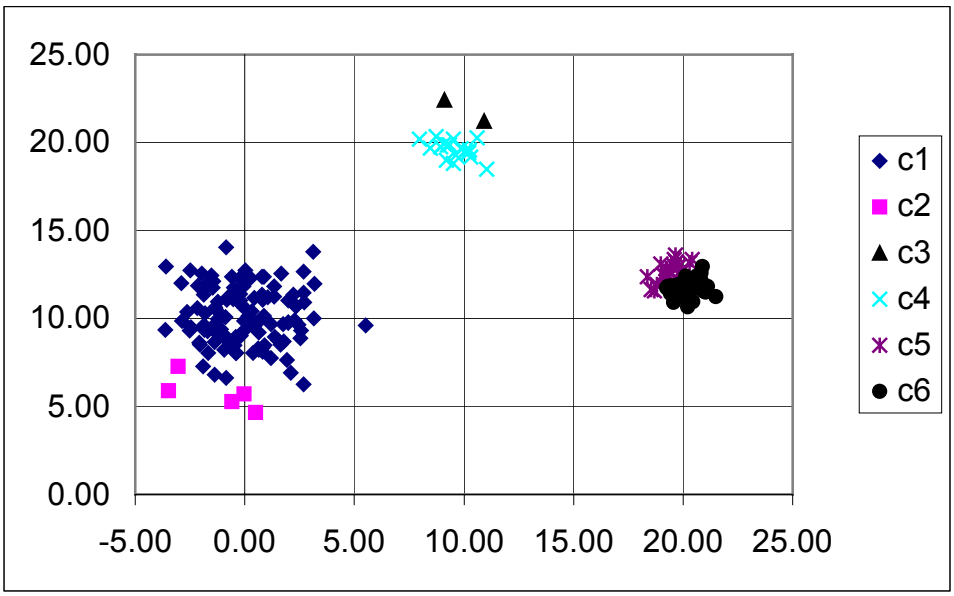

Figura 18. Dados Aleatórios – 200 Gerações.

c) 1.000 gerações

Considerando-se o primeiro grupo, obteve-se o melhor valor da função objetivo, igual a 0.40, após 954 gerações. Considerando-se o segundo grupo, obteve-se o melhor valor da função objetivo, igual a 0.51, após 6 gerações. Considerando-se o terceiro grupo, obteve-se o melhor valor da função objetivo, igual a 0.41 na geração inicial. O agrupamento para este exemplo pode ser visualizado na Figura 19. Cumpre, ainda, enfatizar que não se obtém uma solução melhor após 2.000 gerações. Assim sendo, seguem as regras obtidas:

R1: Se {-3,62≤x≤3,19 e 4,67≤y≤14,04} então G1; (c1 – 118 objetos)
R2: Se {2,67≤x≤5,52 e 6,25≤y≤9,59} então G1; (c2 – 2 objetos)
R3: Se {9,12≤x≤10,92 e 21,23≤y≤22,42} então G2; (c3 – 2 objetos)
R4: Se {7,96≤x≤11,04 e 18,50≤y≤20,33} então G2; (c4 – 18 objetos)
R5: Se {18,38≤x≤20,40 e 11,56≤y≤13,62} então G3; (c5 – 20 objetos)
R6: Se {19,24≤x≤21,49 e 10,65≤y≤12,96} então G3; (c6 – 40 objetos)

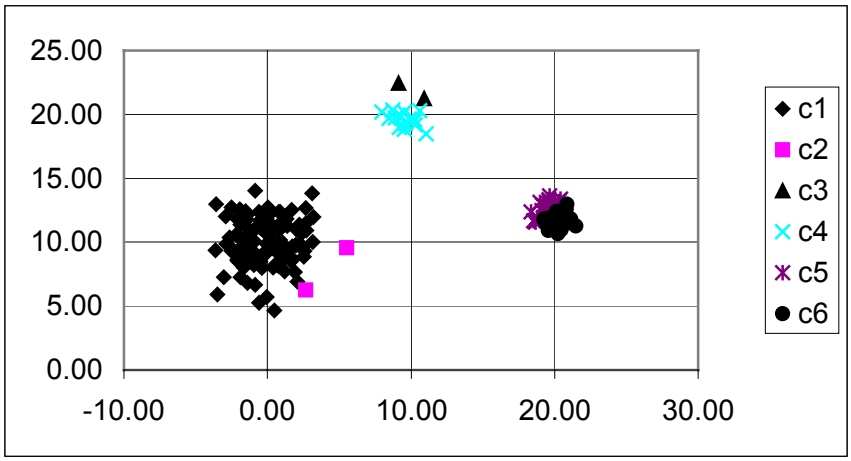

Figura 19. Dados Aleatórios - 1.000 e 2000 gerações.

4.8 Estudos de Casos

Esta seção descreve os resultados da aplicação do AGAE (Algoritmo Genético de Agrupamento Específico) em cinco aplicações de *data mining*. O primeiro estudo de caso se dá na base de dados intitulada *Iris Plants Database*, que é muito conhecida na comunidade científica que estuda *data mining*. Os experimentos realizados nesta base tem como principal objetivo esclarecer a metodologia empregada nos demais exemplos,

constituindo-se, pois, em uma espécie de *exemplo didático*, em que se optou por utilizar a base de dados completa, não usando conjuntos de treinamento, validação e de teste. Os exemplos descritos nas seções 4.8.2 a 4.8.5 refletem uma maior preocupação com aplicações de *data mining*, e, nestes, faz-se uso de conjuntos separados de treinamento, validação e de teste, aplicando o conceito de validação cruzada descrito em [3,5,13,15].

Uma questão importante a ser tratada diz respeito à proporção de exemplos nos conjuntos de treinamento, validação e de teste. Vários autores têm tratado deste assunto. SMITH [15] sugere um método de validação cruzada em que se usa 40% dos exemplos para treinamento, 30% para validação e 30% para teste. O autor afirma, ainda, que nas aplicações práticas que envolvem o uso de redes neurais, pode-se preterir do uso do conjunto de teste. BIGUS [3] sugere o uso de 80% dos exemplos disponíveis para treinamento da rede neural, 10% para teste e 10% para validação. Observa-se, ainda, que os autores adotam denominações diferentes para o conjunto de dados usado para interromper o treinamento de uma rede neural quando o super-ajuste começa a ocorrer. SMITH [15] chama este conjunto de *validação* enquanto que BIGUS [3] o chama de *teste*. HENERY [134] sugere a utilização de 70% a 80% dos dados para treinamento, sendo os demais utilizados para teste. Os autores sugerem, ainda, que de acordo com a necessidade pode-se utilizar três conjuntos de dados. Neste caso, sugerem que a amostra de treinamento deve conter de 55% a 65% dos dados, a amostra de validação deve conter de 15% a 25% dos dados e a de teste de conter 20% dos dados. HAYKIN [5] sugere um método de validação cruzada em que se usa 80% dos exemplos para estimação(treinamento) e 20% para validação.

Neste trabalho, optou-se por usar a média das proporções sugeridas por SMITH [15], BIGUS [3] e HENERY [134]. Desta forma, 60% dos exemplos serão empregados para treinamento, 20% para validação e 20% para teste. Cumpre observar que denomina-se *conjunto de validação* aquele utilizado para treinar as redes neurais, isto é, para avaliar o aprendizado que acontece no conjunto de treinamento, evitando o super-ajuste.

No tocante às redes neurais treinadas para classificação, prefere-se que os exemplos de cada classe possuam a mesma influência no treinamento da rede neural [15]. Isto não acontece em alguns dos experimentos realizados. Nestes casos, a amostra é dita não representativa [15]. Poder-se-ia, numa tentativa de se diminuir os efeitos inconvenientes deste fato, alterar a estrutura da amostra, incluindo exemplos repetidos pertencentes às classes com menor número de exemplos. Este procedimento não foi

utilizado, pois inviabilizaria a comparação dos resultados aqui obtidos com os resultados de outros métodos encontrados na literatura. Nos casos de utilização de amostras não representativas, observa-se que o ruído exerce mais influência no aprendizado das classes cujo número de exemplos é menor [15].

Na maior parte dos experimentos foram utilizadas populações de tamanho definidos pelo próprio AGAE, pois em cada experimento o número máximo de grupos foi considerado igual ao número de exemplos da base de dados modelada. Nestes casos, o AGAE define o número de genótipos por meio do número de objetos representativos, pois se $G(i) \leq 0$ então o objeto "i" não tem potencial para se constituir em uma semente inicial. Assim, escolhem-se novos objetos representativos enquanto $G(i) > 0$. Além disso, interrompeu-se o AGAE após 2.000 gerações. Alguns experimentos, interessantes mas que não se constituem no escopo principal desta tese, foram realizados considerando-se populações de tamanho constante, isto é, número de genótipos igual a 50 e um máximo de 1.000 gerações. Nestes casos, far-se-á menção aos parâmetros utilizados; caso contrário, considera-se implícito que o AGAE define o número de objetos representativos e que o critério de *convergência* adotado implica em interromper o algoritmo após 2.000 gerações. Além disso, todos os experimentos dessa seção foram realizados utilizando-se um PC Celerom 600Mhz, 64Mb RAM. Assim, todos os custos computacionais descritos levam em conta a utilização desta máquina. Além disso, todas as simulações de redes neurais do presente trabalho foram realizadas por meio do software *NeuroShell* [135].

4.8.1 Iris Plants Database

Este problema é clássico na literatura de *Reconhecimento de Padrão* sendo também referenciado em muitos trabalhos que envolvem técnicas de *data mining*. Os dados contém três classes de plantas intituladas Setosa, Versicolour e Virgínica. Cada classe contém 50 exemplos. A primeira classe é linearmente separável das demais, enquanto que as classes Versicolour e Virgínica não são linearmente separáveis entre si. Os quatro atributos envolvidos são: comprimento e largura de sépalas e de pétalas.

Considerando-se que o conjunto de dados representa a população para o domínio em questão, optou-se pelo treinamento da rede em todo o conjunto de dados que constitui, pois, no conjunto de treinamento. Assim sendo, não se utilizou de conjuntos de validação e de teste, nem tampouco se considerou os problemas de generalização.

Como já foi anteriormente observado [22] as áreas de pétalas e de sépalas usualmente proporcionam melhores resultados do que a utilização dos quatro atributos originais envolvidos. Assim sendo, aplicou-se o AGAE no espaço original dos atributos, isto é, no espaço formado pelas áreas de sépalas e de pétalas e também no espaço gerado pelas ativações de uma rede neural treinada para este problema.

4.8.1.1 Espaço original dos atributos – áreas de sépalas e de pétalas

Inicialmente, aplicou-se o AGAE na base de dados original, isto é, na base de dados formada pelas áreas de sépalas e de pétalas. O experimento foi dividido em dois: primeiramente aplicou-se o AGAE para verificar a sua eficácia em encontrar as três classes de plantas. Em seguida, aplicou-se o AGAE nos valores das áreas das classes no intuito de se encontrar sub-classes que descrevam cada uma das classes.

a) Base de dados completa

Os *grupos* corretos para este exemplo estão ilustrados na Figura 20. Considerando a medida de distância Euclidiana, o AGAE encontrou 76 objetos representativos que proporcionam uma contribuição G(i) positiva. Desta maneira, foram utilizados 75 genótipos representando de 2 a 76 grupos e a melhor solução foi obtida após 725 gerações e o valor da função objetivo (VFO) foi igual a 0,47. Esta solução representa um grupo de 80 objetos e um de 70 objetos - Figura 21. Conforme se pode observar, o AGAE não encontrou a solução correta, ilustrada na Figura 20. Este fato pode ser explicado pela função objetivo usada, que assume o valor 0,39 para os grupos corretos. Como este valor é inferior àquele encontrado, a solução correta não poderia ser encontrada. Além disso, ao separar os objetos da classe setosa das demais, a função objetivo assume o valor 0,43 que também é inferior ao encontrado pelo AGAE.

Estes resultados levam ao questionamento sobre a adequação da função objetivo utilizada, bem como da medida de distância empregada. Algumas respostas a este questionamento podem ser obtidas por meio da observação dos resultados obtidos na aplicação da medida de distância de Manhattan. Usando-se esta medida de dissimilaridade, obtém-se a melhor solução após 68 gerações (VFO=0,49) e os *grupos* obtidos estão ilustrados na Figura 22. Conquanto esta solução não represente a solução correta, 88% dos objetos são classificados corretamente e este fato comprova que o AGAE é sensível à medida de distância utilizada.

Cumpre observar, ainda, que a aplicação do AGAE - tanto com o uso da medida de distância Euclidiana quanto com o uso da medida de Manhattan - na base de dados original formada pelos quatro atributos, não proporciona a solução ideal para este problema, pois o AGAE somente separa a classe linearmente separável (setosa) das demais, fato que também é confirmado por outros autores [102,104].

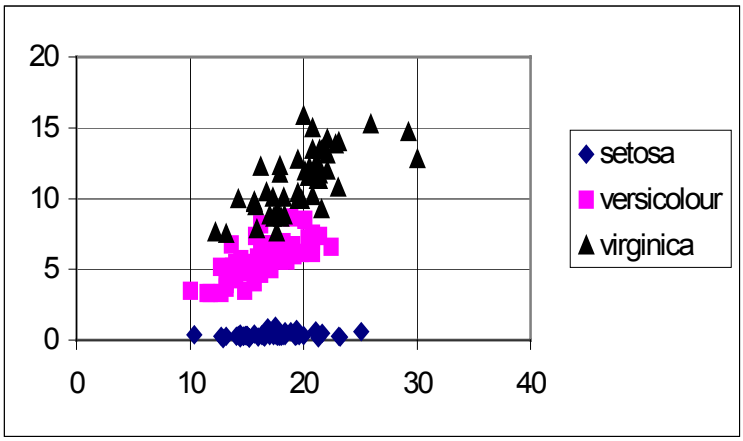

Figura 20. Grupos Corretos - Áreas.

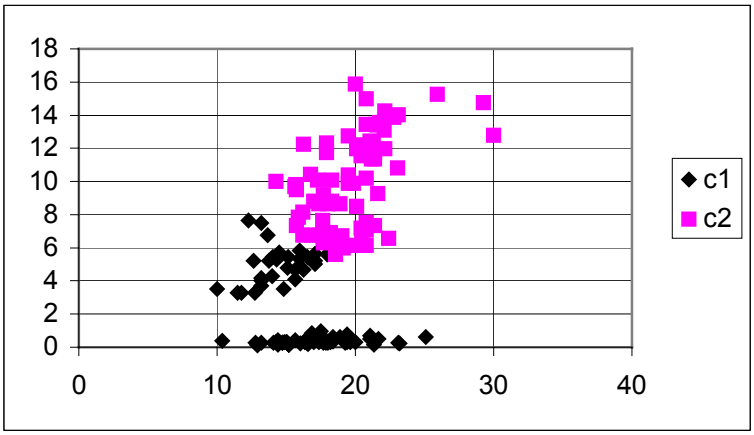

Figura 21. Grupos obtidos pelo AGAE - Euclidiana.

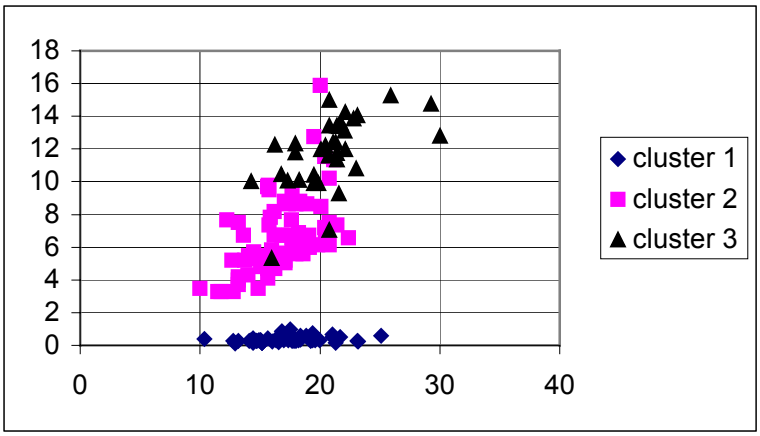

Figura 22. Grupos Obtidos AGAE – Manhattan.

b) Classes separadamente

Neste exemplo, o AGAE é aplicado em cada classe separadamente. Obviamente que, neste caso, as classes sempre são modeladas corretamente (i.e. os grupos de cada uma das classes são encontrados). Assim, deseja-se avaliar as regras obtidas através da

modelagem das sub-classes, dado que o problema de extração de regras de modelos de redes neurais por meio dos valores das ativações das unidades ocultas eventualmente se dá em um espaço em que há sobreposição de domínio.

Considerando-se a classe *setosa*, o AGAE encontrou 25 sementes significativas, ou seja, utilizou 24 genótipos. Após 356 gerações, obteve-se o melhor valor da função objetivo (VFO) igual a 0,58 e as regras obtidas são descritas pelos sete primeiros grupos ($c_{1,..,}c_7$).

Considerando-se a classe *versicolour*, o AGAE também encontrou 25 sementes significativas. Após 25 gerações, obteve-se a melhor solução (VFO=0,49) e as regras obtidas são descritas pelos grupos c_8 e c_9.

Considerando-se a classe *virgínica*, o AGAE encontrou 24 sementes significativas. Após 329 gerações, obteve-se a melhor solução (VFO=0,58) e as regras obtidas são descritas pelos grupos c_{10} e $c_{11.}$ Os *grupos* obtidos, suas regras e número de objetos correspondentes são:

Se ($17,34 \leq A_s \leq 18,36$ e $0,28 \leq A_p \leq 0,96$) então c_1 – 10 objetos;
Se ($14,08 \leq A_s \leq 15,64$ e $0,14 \leq A_p \leq 0,42$) então c_2 – 14 objetos;
Se ($21,06 \leq A_s \leq 21,66$ e $0,15 \leq A_p \leq 0,68$) então c_3 – 4 objetos;
Se ($16,00 \leq A_s \leq 17,00$ e $0,20 \leq A_p \leq 0,85$) então c_4 – 8 objetos;
Se ($10,35 \leq A_s \leq 13,20$ e $0,11 \leq A_p \leq 0,39$) então c_5 – 4 objetos;
Se ($18,87 \leq A_s \leq 19,98$ e $0,26 \leq A_p \leq 0,76$) então c_6 – 7 objetos;
Se ($23,10 \leq A_s \leq 25,08$ e $0,24 \leq A_p \leq 0,60$) então c_7 – 3 objetos;
Se ($15,75 \leq A_s \leq 22,40$ e $4,68 \leq A_p \leq 8,64$) então c_8 – 31 objetos;
Se ($10,00 \leq A_s \leq 15,66$ e $3,30 \leq A_p \leq 6,75$) então c_9 – 19 objetos;
Se ($12,25 \leq A_s \leq 25,92$ e $7,50 \leq A_p \leq 15,87$) então c_{10} – 48 objetos;
Se ($29,26 \leq A_s \leq 30,02$ e $12,80 \leq A_p \leq 14,74$) então c_{11} – 2 objetos;

As regras acima descrevem a base de dados com uma taxa de classificação correta média (TCCM) igual a 94,67%, ou seja, 8 exemplos são classificados incorretamente. A Figura 23 permite a visualização dos grupos e dos exemplos em que há sobreposição de domínio.

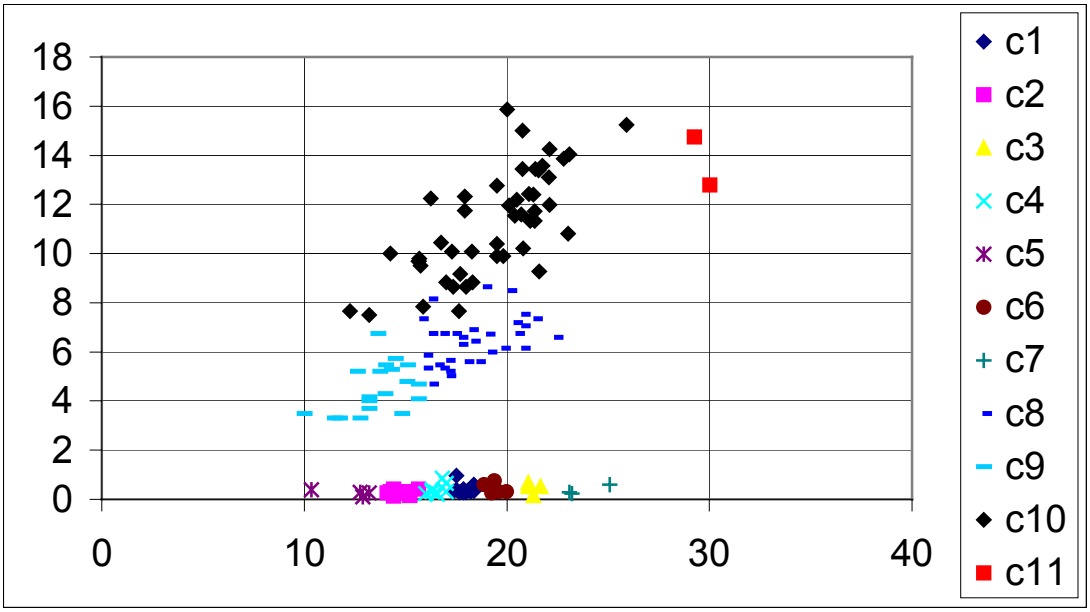

Figura 23. Grupos (sub-classes) obtidos para os 50 exemplos de cada classe.

4.8.1.2 Espaço das ativações de uma rede neural

Este exemplo pretende exemplificar, didaticamente, o processo de extração de regras de redes neurais supervisionadas por meio da aplicação do AGAE no espaço das ativações das unidades ocultas. Os atributos envolvidos são as áreas de sépalas e de pétalas. Com o objetivo de encontrar a melhor precisão possível e com a menor arquitetura possível, várias redes neurais foram treinadas. A melhor delas possui duas unidades de entrada, utilizando uma transformação linear [-1,+1], duas unidades ocultas com a função tangente hiperbólica, e duas unidades de saída com função logística. Uma descrição completa deste experimento pode ser encontrada em [22]. Após 15.000 épocas esta rede neural proporciona um erro quadrático médio igual a 0,0002975. Calculam-se, então, os valores das ativações para cada exemplo do conjunto de treinamento e então o AGAE é aplicado para extrair as regras de classificação. As ativações das unidades ocultas são dadas pelas expressões abaixo:

$$a_1 = 0.22A_s - 2.16A_p + 0.84$$
$$a_2 = 3.12A_s - 11.91A_p + 41.99$$

Através das expressões acima, pode-se calcular os valores das ativações das unidades ocultas. De posse destes valores, aplica-se o AGAE neste espaço transformado, tanto na base de dados completa quanto nos exemplos de cada classe separadamente.

a) Base de dados completa:

Neste exemplo, o AGAE é aplicado em toda a base de dados, isto é, de maneira "não supervisionada". Busca-se, pois, encontrar os diversos *grupos* que descrevem as três classes de plantas. A Figura 24 ilustra os grupos obtidos e, neste caso, obtém-se o agrupamento correto, ao contrário do que se verificou na aplicação do AGAE no espaço original.

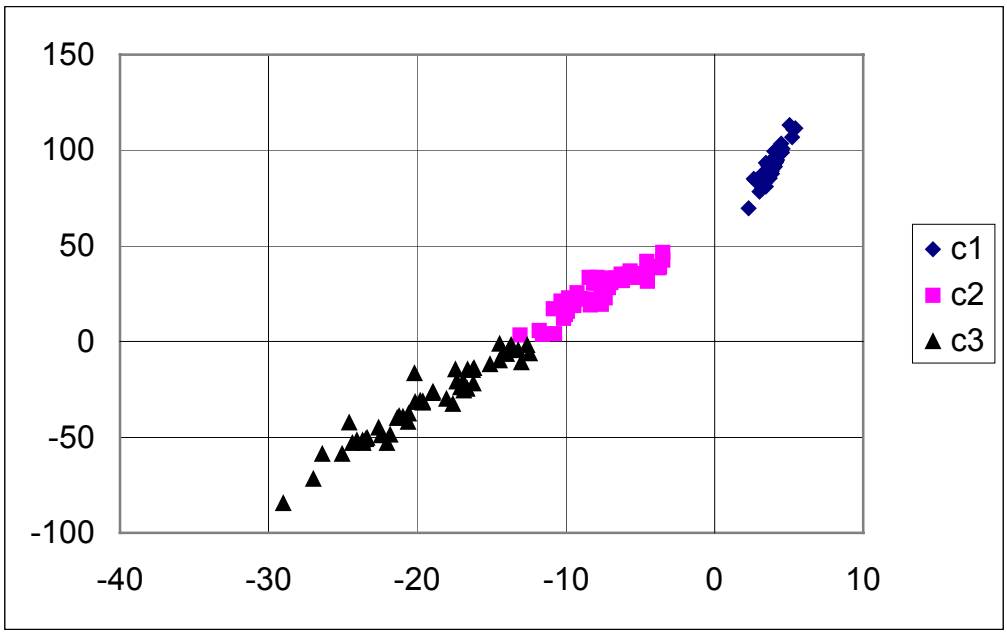

Figura 24. Ativações a_1 e a_2 representadas nos eixos x e y respectivamente.

Desta forma, observa-se que as ativações da RN permitiram a modelagem correta dos dados. Isto se deve ao fato de que as unidades ocultas são "detectoras de características" e reforça a idéia de que a tarefa de extração de regras em um espaço transformado pela RN pode ser mais simples do que no espaço original dos atributos. Neste caso, a RN permite a modelagem das classes e sugere que a metodologia proposta é útil em aplicações de *data mining*. Além disso, a solução de melhor adequabilidade, i.e. de maior valor da função objetivo (VFO) igual a 0,72 , foi obtida na geração inicial e as regras obtidas proporcionam uma taxa de classificação correta média (TCCM) igual a 96%, ou seja, 6 exemplos são classificados incorretamente, pois há sobreposição de domínio entre as classes versicolour e virgínica. As regras obtidas para cada classe são as seguintes:

Se ($2,2746 \leq a_1 \leq 5,4256$ e $69,6371 \leq a_2 \leq 113,0936$) então C_1 – 50 objetos - setosa;

Se ($-13,0980 \leq a_1 \leq -3,4596$ e $3,4670 \leq a_2 \leq 46,5434$) então C_2 – 49 objetos - versicolour;

Se ($-29,0348 \leq a_1 \leq -12,456$ e $-84,5593 \leq a_2 \leq -1,1428$) então C_3 – 51 objetos - virgínica;

> Pode-se citar que o processo de extração de regras se dá através da obtenção de hiperplanos (no caso de duas dimensões em retas) que dividem o espaço dos atributos transformado linearmente. Esta transformação será linear sempre que a transformação dos atributos originais também o for, pois as ativações das unidades ocultas da primeira camada de uma rede neural podem ser escritas pela expressão:

$$a_k = \sum_{j=1}^{j=m} w_{kj} . x_j$$

onde a_k representa a ativação da "k-ésima"unidade oculta, "m" é o número de atributos envolvidos, w_{kj} são os pesos sinápticos para a "k-ésima" unidade oculta e x_j representam os valores dos atributos transformados segundo a função escolhida. Neste trabalho, somente dois tipos de transformações são aplicadas nas unidades de entrada da rede neural, a saber: a função linear [0,1] e a função linear[-1,+1]. Assim, embora a RN possa mapear funções não lineares, o processo de extração de regras se dá em um espaço em que os atributos foram transformados linearmente pela otimização dos pesos das conexões.

Assim, a aplicação do AGAE no espaço das ativações proporcionou bons resultados. Isto acontece porque, neste caso, é mais simples extrair padrões em um espaço transformado pelas unidades ocultas, que são detectoras de características. Basicamente, cada unidade oculta na primeira camada de um perceptron de múltiplas camadas forma um hiperplano [13], pois a transformação usada na camada de entrada é linear. Ou ainda, pode-se dizer que as unidades ocultas são detectoras de características porque elas tentam mapear classes que, no espaço original não são linearmente separáveis, para o espaço das ativações, em que as classes possuem uma chance maior de serem linearmente separáveis [5]. Sob o ponto de vista geométrico, as regras lógicas correspondem a uma divisão do espaço das características, que se dá por meio de hiperplanos perpendiculares aos eixos, em *áreas* com nomes simbólicos. Como se descreve em [20], se as classes no espaço das entradas das unidades ocultas podem ser separadas por tais hiperplanos, então a descrição lógica dos dados é possível e meritória.

b) Classes separadamente

Neste caso, as classes são modeladas de maneira "supervisionada", pois o AGAE é aplicado separadamente aos valores das ativações de cada classe. Assim sendo, considerando-se a classe *setosa*, o AGAE encontrou 28 sementes significativas e, após 22 gerações, obteve-se a melhor solução (VFO=0,58). Para a classe versicolour, o AGAE encontrou 26 sementes significativas e, após 30 gerações, obteve-se a melhor solução (VFO= 0,65). Para a classe virgínica, o AGAE encontrou 25 sementes significativas e, após 254 gerações, obteve-se o melhor valor da função objetivo sendo igual a 0,61. Desta forma, as regras obtidas para este problema são as seguintes:

Se ($2{,}27 \leq a_1 \leq 4{,}59$ e $69{,}64 \leq a_2 \leq 103{,}50$)	então C_1 – setosa - 46 objetos;
Se ($5{,}06 \leq a_1 \leq 5{,}43$ e $106{,}72 \leq a_2 \leq 113{,}09$)	então C_2 – setosa - 4 objetos;
Se ($-10{,}83 \leq a_1 \leq -3{,}46$ e $12{,}14 \leq a_2 \leq 46{,}54$)	então C_3 – versicolour - 43 objetos;
Se ($-13{,}67 \leq a_1 \leq -10{,}74$ e $-10{,}9 \leq a_2 \leq 4{,}15$)	então C_4 – versicolour - 7 objetos;
Se ($-9{,}82 \leq a_1 \leq -9{,}04$ e $22{,}39 \leq a_2 \leq 22{,}83$)	então C_5 – virgínica - 2 objetos;
Se ($-29{,}03 \leq a_1 \leq -26{,}99$ e $-84{,}56 \leq a_2 \leq -71{,}80$)	então C_6 – virgínica - 2 objetos;
Se ($-19 \leq a_1 \leq -16{,}22$ e $-27{,}02 \leq a_2 \leq -21{,}03$)	então C_7 – virgínica - 9 objetos;
Se ($-24{,}56 \leq a_1 \leq -20{,}57$ e $-45{,}26 \leq a_2 \leq -37{,}74$)	então C_8 – virgínica - 7 objetos;
Se ($-26{,}4 \leq a_1 \leq -21{,}83$ e $-58{,}77 \leq a_2 \leq -48{,}83$)	então C_9 – virgínica - 11 objetos;
Se ($-20{,}15 \leq a_1 \leq -17{,}63$ e $-32{,}65 \leq a_2 \leq -30{,}09$)	então C_{10} – virgínica - 6 objetos;
Se ($-20{,}20 \leq a_1 \leq -14{,}47$ e $-16{,}8 \leq a_2 \leq -9{,}98$)	então C_{11} – virgínica - 7 objetos;
Se ($-14{,}45 \leq a_1 \leq -11{,}8$ e $-6{,}75 \leq a_2 \leq 5{,}92$)	então C_{12} – virgínica - 6 objetos;

As regras acima descrevem a base de dados com uma TCCM igual a 95,33%, ou seja, 7 exemplos são classificados incorretamente. A Figura 25 permite a visualização dos grupos e dos exemplos em que há sobreposição de domínio. Nesta figura, o exemplo mais nítido deste fenômeno ocorre com o grupo C5, pertencente à classe virgínica mas presente na mesma região do domínio da classe versicolour – grupos C3 e C4.

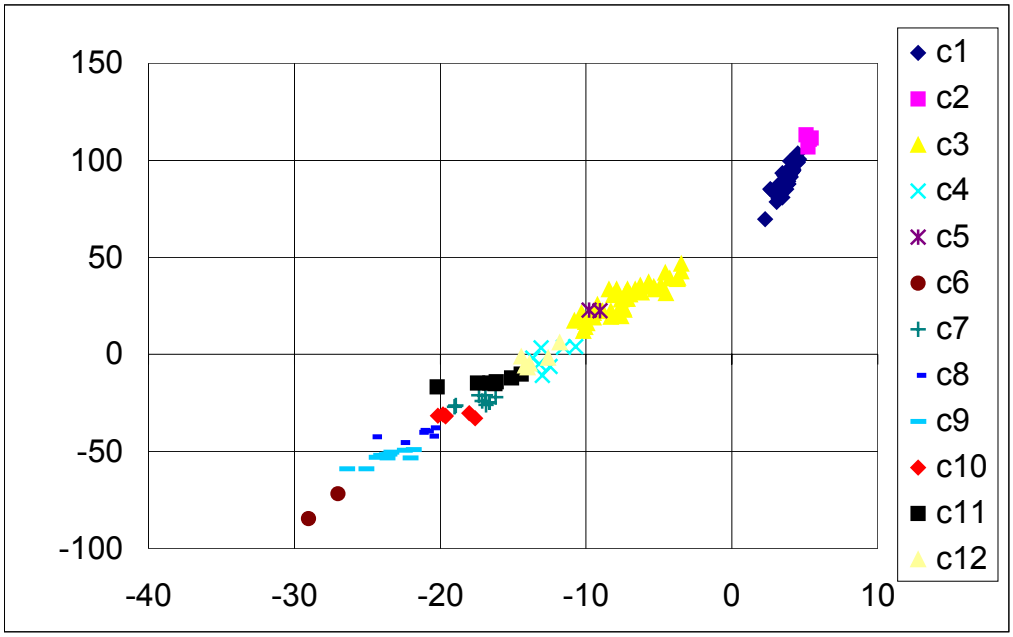

Figura 25. Sub-classes no espaço das ativações.

O AGAE foi avaliado em uma base de dados muito utilizada para testar algoritmos de *data mining*. Assim, pode-se comparar os resultados obtidos com aqueles relatados na literatura. DUCH et al. [20] descrevem um sumário dos resultados de alguns métodos de extração de regras para este exemplo, em que as taxas de classificação corretas médias variam de 95,70% a 100%. Observa-se, então que o desempenho do AGAE é satisfatório neste problema de classificação.

No tocante aos resultados obtidos no espaço original e no espaço das ativações, observa-se que o AGAE forneceu TCCM semelhantes – 94,67% no espaço original e 95.33% no espaço das ativações. Além disso, o AGAT [126] e o Algoritmo RX Modificado [22,81,82] forneceram resultados que, no espaço das ativações, foram um pouco melhores (TCCM=97%) do que os obtidos pelo AGAE.

A título de ilustração, o AGAE executou as 2.000 gerações dos experimentos com 150 exemplos em aproximadamente 8 minutos e os experimentos que envolviam a obtenção de sub-classes, com 50 exemplos, em aproximadamente 1 minuto.

4.8.2 Australian Credit Approval Database

Esta base de dados, formada por 690 exemplos e 15 variáveis, pode ser obtida em *www.ncc.up.pt/liacc/ML/statlog/datasets.html* e trata sobre uma aplicação de cartões de crédito. O banco de dados contém 307 exemplos de pessoas que receberam crédito

(classe *um*), e 383 de pessoas que não receberam crédito (classe *zero*). Os exemplos são descritos por meio de 14 atributos e o rótulo da classe. Todos os nomes dos atributos e os valores estão originalmente codificados para proteger a confidência dos dados. Assim, os atributos envolvidos serão chamados de A, B, C, D, E, F, G, H, I, J, K, L, M, N. O AGAE foi aplicado tanto no espaço original dos atributos como no espaço das ativações do melhor modelo de rede neural obtido.

4.8.2.1 Aplicando o AGAE no espaço original dos atributos

Inicialmente, o AGAE foi aplicado na base de dados completa com o objetivo de se verificar se os objetos de cada grupo seriam corretamente *classificados*. Cumpre observar, entretanto, que o presente trabalho objetiva, principalmente, mostrar a aplicação do AGAE para extrair regras, seja no espaço original dos atributos ou no espaço das ativações de uma rede neural. Por outro lado, entende-se ser importante descrever os resultados da aplicação do AGAE na base de dados original e de maneira não supervisionada, isto é, não com o objetivo de se extrair regras de classificação mas com o objetivo de se verificar se o AGAE separa os objetos de cada classe corretamente, pois assim pode-se ter uma noção do desempenho do método proposto bem como da complexidade das bases de dados utilizadas. Neste sentido, optou-se por limitar o tamanho das populações utilizadas nestes tipos de exemplo, utilizando 50 genótipos e estabelecendo o número máximo de gerações igual a 1.000. Estes parâmetros foram definidos *a* priori e refletem a preocupação com o custo computacional dos experimentos. O AGAE não encontrou os dois grandes grupos de objetos, separando apenas 3 objetos dos demais. Esta solução foi obtida após 3 gerações e o valor da função objetivo (VFO) foi igual a 0,98 , indicando que uma forte estrutura foi obtida. Este valor indica a possibilidade de que os três objetos separados poderiam se constituir em ruídos da base de dados. Este mesmo resultado foi obtido tanto para a medida de distância Euclidiana quanto para a medida de distância de Manhattan e cada um dos experimentos consumiu aproximadamente 1 hora. Optou-se, então, por extrair estes três objetos da base de dados e aplicar o AGAE novamente nesta nova base de dados, formada por 687 objetos e usando a medida de distância Euclidiana. Entretanto, o AGAE novamente encontrou um grupo formado por 6 objetos, e outro formado por 681 objetos. Seguindo a mesma metodologia, estes seis objetos foram extraídos e o AGAE foi aplicado novamente, resultando numa solução formada por dois grupos,

separando cinco objetos dos demais. Cumpre observar que estes pequenos *grupos* continham exemplos das duas classes envolvidas.

A segunda aplicação descrita nesta seção considera o método da validação cruzada. Este método considera a utilização de três conjuntos de dados: treinamento, teste e validação. Os três conjuntos foram obtidos por meio da extração aleatória de exemplos da base de dados original. O conjunto de treinamento, usado para a obtenção das regras, é formado por 60% dos exemplos da base de dados, ou seja, por 229 exemplos da classe *zero* e por 183 objetos da classe *um*. Os conjuntos de validação (usado posteriormente para avaliar os modelos de redes neurais durante o treinamento) e de teste contém o mesmo número de exemplos: 77 objetos da classe *zero* e 62 objetos da classe *um*. Para se obter o conjunto de regras de classificação, aplica-se o AGAE separadamente em cada subconjunto dos exemplos de treinamento, sendo um subconjunto formado pelos objetos pertencentes à classe *zero* e outro formado pelos objetos pertencentes à classe *um*.

Considerando a classe *zero*, o AGAE encontrou 107 sementes. Assim, a população usada contém 106 genótipos e a melhor solução foi encontrada após 21 gerações (VFO=0,91). Esta solução é constituída por dois grupos: c_1 e c_2 formados por 225 e 4 objetos respectivamente. Considerando a classe *um*, o AGAE encontrou 91 objetos representativos e a melhor solução foi obtida após 1.186 gerações (VFO=0,94), correspondendo aos grupos c_3 e c_4, os quais são formados por 181 e 2 objetos respectivamente. Estes grupos podem ser traduzidos por meio das regras descritas na Figura 26. Estas regras classificam corretamente 12,95% e 10,07% dos exemplos de validação e de teste respectivamente. Por um lado, considerando a classe *um* como *default* (grupos c_1 e c_2 para descrever as regras de classificação), obtém-se taxas corretas médias de classificação iguais a 66,91% e 63,31% nos conjuntos de validação e de teste respectivamente. Por outro lado, considerando a classe *zero* como *default* (grupos c_3 e c_4 para descrever as regras de classificação), obtém-se taxas corretas médias de classificação iguais a 45,32% e 42,45% nos conjuntos de validação e de teste respectivamente. Conquanto todos estes resultados foram obtidos pelo uso da distância Euclidiana, ao usar a medida de distância de Manhattan os mesmos resultados foram obtidos. No concernente ao custo computacional, o experimento envolvendo a classe zero - 106 genótipos e 2.000 gerações - consumiu 36 minutos e o experimento envolvendo a classe *um* – 90 genótipos e 2.000 gerações – consumiu 19 minutos.

Se { 0≤A≤1 e 15.17≤B≤80.25 e 0≤C≤26.33 e 1≤D≤2 e 1≤E≤14 e 1≤F≤8 e
0≤G≤13.88 e 0≤H≤1 e 0≤I≤1 e 0≤J≤11 e 0≤K≤1 e 1≤L≤3 e 0≤M≤2,000 e
1≤N≤2,198 } então c1 ;
Se { A=1 e 22.5≤B≤36.17 e 2.5≤ C ≤18.13 e 1≤D≤2 e 1≤E≤9 e 1≤F≤4 e
0.085≤G≤1.5 e H=0 e 0≤I≤1 e 0≤J≤2 e 0≤K≤1 e L=2 e M≤320 e
3,553≤N≤5,553 } então c2 ;
Se { 0≤A≤1 e 15.83≤ B ≤76.75 e 0≤C≤28 e 1≤D≤2 e 1≤E≤14 e 1≤F≤9 e
0≤G≤28.5 e 0≤H≤1 e 0≤I≤1 e 0≤J≤67 e 0≤K≤1 e 1≤L≤3 e 0≤M≤840 e
1≤N≤31,286} então c3;
Se { 0≤A≤1 e 34.17≤B≤47.42 e 1.54≤C≤8 e D=2 e 10≤E≤13 e 4≤F≤5 e
1.54≤G≤6.5 e H=1 e I=1 e 1≤J≤6 e 0≤K≤1 e L=2 e 375≤M≤520 e
5,001≤N≤51,101 }então c4 ;

Figura 26. Conjunto Completo de Regras (espaço original) – Crédito.

Os resultados obtidos promovem o interesse por um experimento em que os objetos dos menores grupos – c_2 e c_4 - são excluídos para que o AGAE seja novamente aplicado nos objetos remanescentes. Assim, considerou-se apenas os objetos do grupos c_1 e c_3 e aplicou-se o AGAE separadamente em cada um deles para obter novas regras de classificação, sendo que estas, por sua vez, foram novamente avaliadas nos conjuntos de validação e de teste. Para estes experimentos, considerou-se apenas o uso da medida de distância Euclidiana.

Considerando a classe *zero*, o AGAE encontrou 103 sementes. Assim, a população usada contém 102 genótipos e a melhor solução foi encontrada após 60 gerações (VFO=0,86). Esta solução é constituída por dois grupos formados por 222 e 3 objetos respectivamente. Considerando a classe *um*, o AGAE encontrou 89 objetos representativos e a melhor solução foi obtida após 37 gerações (VFO=0,93). Esta solução é formada por dois grupos, constituídos por 179 e 2 objetos respectivamente. As regras obtidas classificam corretamente 14,39% e 12,95% dos exemplos de validação e de teste respectivamente. Por um lado, considerando a classe *um* como *default*, obtém-se taxas corretas médias de classificação iguais a 67,62% e 65,47% nos conjuntos de validação e de teste respectivamente. Por outro lado, considerando a classe *zero* como *default*, obtém-se taxas corretas médias de classificação iguais a 45,32% e 42,45% nos conjuntos de validação e de teste respectivamente. Como se pode observar, não houve

melhorias significativas nas taxas de classificação quando se eliminaram os objetos dos menores grupos.

Resumidamente, as melhores regras obtidas nos experimentos realizados no espaço original dos atributos estão descritas na Figura 27 e classificam 66,91% e 63,31% dos exemplos dos conjuntos de validação e de teste respectivamente. A título de ilustração, as melhores regras obtidas por meio do AGAT proporcionaram melhores resultados em termos da TCCM, pois classificam corretamente 81% dos exemplos de treinamento. Cumpre observar, contudo, que nos experimentos envolvendo o AGAT não se utilizou da *validação cruzada*, o que inviabiliza uma comparação *mais justa* dos resultados obtidos pelos dois algoritmos, pois as regras obtidas pelo AGAE foram obtidas no conjunto de treinamento e avaliadas nos conjuntos de validação e de teste.

Se { 0≤A≤1 e 15.17≤B≤80.25 e 0≤C≤26.33 e 1≤D≤2 e 1≤E≤14 e 1≤F≤8 e 0≤G≤13.88 e 0≤H≤1 e 0≤I≤1 e 0≤J≤11 e 0≤K≤1 e 1≤L≤3 e 0≤M≤2,000 e 1≤N≤2,198 } então *classe zero* ;

Se { A=1 e 22.5≤B≤36.17 e 2.5≤ C ≤18.13 e 1≤D≤2 e 1≤E≤9 e 1≤F≤4 e 0.085≤G≤1.5 e H=0 e 0≤I≤1 e 0≤J≤2 e 0≤K≤1 e L=2 e M≤320 e 3,553≤N≤5,553 } então *classe zero* ;

Senão *classe um*.

Figura 27. Melhores Regras obtidas no espaço original – Crédito Australiano.

4.8.2.2 Aplicando o AGAE no espaço das ativações

Esta seção trata do processo de extração de regras de modelos de redes neurais. Para tal, inicialmente foram treinadas 40 redes neurais supervisionadas. Os modelos de redes neurais treinados e os valores dos erros médios quadrados no conjunto de validação estão listados na Tabela 6.

Os experimentos consideram apenas redes neurais com uma camada oculta, formada por 5 unidades no máximo. O algoritmo de aprendizado considera uma taxa de aprendizado igual a 0,10 e de um termo de momento também igual a 0,10. Os pesos foram iniciados aleatoriamente no intervalo [-0,3;+0,3]. Neste caso os exemplos foram apresentados à rede neural na ordem em que os mesmos aparecem no conjunto de treinamento e o erro médio quadrado foi verificado no conjunto de validação com freqüência de uma época.

Tabela 6. Redes Neurais Treinadas – Crédito Australiano.

Classificação	Entradas	Ocultas	Saídas	Erro Validação
1	14 [0,1]	1 logistic	2 logistic	0.1358
2	14 [-1,1]	1 logistic	2 logistic	0.1374
3	14 [-1,1]	1 tanh	2 logistic	0.1447
4	14 [-1,1]	2 tanh	2 logistic	0.1571
5	14 [-1,1]	3 tanh	2 logistic	0.1653
6	14 [-1,1]	4 tanh	2 logistic	0.1746
7	14 [-1,1]	2 logistic	2 logistic	0.1773
8	14 [0,1]	2 logistic	2 logistic	0.1934
9	14 [-1,1]	3 logistic	2 logistic	0.2097
10	14 [0,1]	3 logistic	2 logistic	0.2186
11	14 [-1,1]	4 logistic	2 logistic	0.2227
12	14 [0,1]	5 logistic	2 logistic	0.2258
13	14 [-1,1]	5 tanh	2 logistic	0.2304
14	14 [0,1]	4 logistic	2 logistic	0.2390
15	14 [-1,1]	5 logistic	2 logistic	0.2446
16	14 [-1,1]	4 logistic	2 tanh	0.8710
17	14 [-1,1]	5 logistic	2 tanh	0.8832
18	14 [0,1]	3 logistic	2 tanh	0.9627
19	14 [0,1]	5 logistic	2 tanh	1.0327
20	14 [0,1]	4 logistic	2 tanh	1.2143
21	14 [-1,1]	2 logistic	2 tanh	1.3912
22	14 [0,1]	1 logistic	2 tanh	1.3914
23	14 [-1,1]	1 logistic	2 tanh	1.8709
24	14 [0,1]	2 logistic	2 tanh	2.1262
25	14 [-1,1]	3 logistic	2 tanh	2.3139
26	14 [-1,1]	4 tanh	2 linear	2.9735
27	14 [-1,1]	2 tanh	2 linear	3.1840
28	14 [-1,1]	5 tanh	2 linear	3.1971
29	14 [-1,1]	3 logistic	2 linear	3.3108
30	14 [-1,1]	1 logistic	2 linear	3.3146
31	14 [0,1]	1 logistic	2 linear	3.3411
32	14 [-1,1]	5 logistic	2 linear	3.4981
33	14 [0,1]	3 logistic	2 linear	3.5000
34	14 [-1,1]	4 logistic	2 linear	3.5740
35	14 [0,1]	2 logistic	2 linear	3.6471
36	14 [0,1]	4 logistic	2 linear	3.6600
37	14 [0,1]	5 logistic	2 linear	3.8028
38	14 [-1,1]	3 tanh	2 linear	4.0081
39	14 [-1,1]	1 tanh	2 linear	4.2428
40	14 [-1,1]	2 logistic	2 linear	4.2486

O melhor modelo de rede neural - formado por 14 unidades de entrada usando a transformação linear [0,1], uma unidade oculta usando a função de ativação logística e duas unidades de saída usando também a função de ativação logística - foi usado para a

tarefa de extração de regras. Para extrair as regras, os valores das ativações da unidade oculta desta rede neural foram calculados pela seguinte expressão:

$$a_1 = -0.58A-0.02B+0.089C+1.23D+0.3E-0.086F+0.026G+7.02H+3.27I+ \\ + 0.034J-0,02K-0.065L-0.003M+0.00007N-8.88;$$

Aplicou-se, então o AGAE nos valores das ativações do conjunto de treinamento, considerando-se as ativações da classes separadamente. As regras obtidas foram avaliadas nos conjuntos de validação e de teste. Considerando a classe *zero*, o AGAE usou 113 genótipos e encontrou o melhor agrupamento na primeira geração (VFO=0,67). Esta solução corresponde aos grupos c_5 e c_6 formados por 170 e 59 objetos respectivamente. Considerando a classe *um*, o AGAE usou 86 genótipos e encontrou o melhor agrupamento após 26 gerações (VFO=0,75). Esta solução corresponde aos grupos c_7 e c_8 formados por 169 e 14 objetos respectivamente. Considerando o custo computacional, o experimento envolvendo a classe zero - 113 genótipos e 2.000 gerações - consumiu 33 minutos e o experimento envolvendo a classe *um* – 86 genótipos e 2.000 gerações – consumiu 15 minutos. Estes quatro *grupos* correspondem às regras descritas na Figura 28.

Se ($-9.86 \leq a_1 \leq -2.83$) então c_5 – classe *zero*;
Se ($-2.65 \leq a_1 \leq 7.24$) então c_6 – classe *zero*;
Se ($-2.23 \leq a_1 \leq 9.27$) então c_7 – classe *um*;
Se ($-9.87 \leq a_1 \leq -4.01$) então c_8 – classe *um*;

Figura 28. Conjunto de Regras (espaço das ativações)

Este conjunto de regras não fornece boas taxas de classificação corretas médias. Considerando o conjunto de validação (usado para salvar o melhor modelo de rede neural) observa-se que 7,19% dos exemplos são classificados corretamente, enquanto que no conjunto de teste este valor é igual a 3,60%. Pode-se explicar este fraco poder de classificação por meio da observação das próprias regras, haja vista que os valores de a_1 para os exemplos das duas classes pertencem a intervalos de variação muito semelhantes.

Contudo, ao considerar somente os maiores grupos de cada classe (c_5 e c_7) que representam 74% e 92% dos exemplos de cada classe respectivamente, obtém-se melhores resultados. Assim sendo, a taxa de classificação correta média para o conjunto de validação é igual a 86,33% e para o conjunto de teste é igual a 76,26%. Além disso,

ao considerar o grupo c_5 como *default*, observa-se que 87.05% e 76.97% dos exemplos de validação e de teste respectivamente são classificados corretamente. Considerando o *grupo* c_7 como *default*, 86,33% e 76,62% dos exemplos de validação e de teste respectivamente são classificados corretamente.

Desta forma, o melhor resultado obtido no espaço das ativações para este problema envolve as regras descritas na Figura 29, correspondentes aos *grupos* c_5 e c_7 pois, desta forma, pode-se entender as características das duas classes envolvidas. Estas regras descrevem 82% dos exemplos de treinamento e classificam corretamente 86,33% e 76,26% dos exemplos dos conjuntos de validação e de teste respectivamente. Estes resultados são melhores do que aqueles obtidos pelo AGAE no espaço original (descrição das duas classes), cujas regras classificam 12,95% e 10,07% dos exemplos de validação e de teste respectivamente. MICHIE et al [136] descrevem vários resultados obtidos por meio de diversos métodos para esta base de dados. Ainda que os resultados descritos em [136] foram obtidos em testes de validação cruzada usando 10 objetos, eles fornecem um parâmetro de comparação para os resultados descritos nesta tese. Os resultados descritos em [136] sugerem taxas de classificação corretas médias que variam entre 79,30% e 86,90% para os conjuntos de teste, de modo que os resultados obtidos pelo AGAE na espaço das ativações são satisfatórios e, além disso, muito melhores do que os resultados obtidos no espaço original dos atributos. A título de ilustração, o AGAT obteve, no espaço das ativações, regras que classificam 72,61% dos exemplos de treinamento quando ambas as classes são descritas [123,125]. Considerando-se uma classe *default*, o AGAT classifica corretamente 86,67% dos exemplos de treinamento.

Se ($-9.86 \leq a_1 \leq -2.83$) então *zero*;
Se ($-2.23 \leq a_1 \leq 9.27$) então *um*;

Figura 29. Melhores Regras Extraídas da Rede Neural.

4.8.3 Pima Indians Diabetes Database

Este exemplo, clássico na literatura de *Data Mining*, representa um complexo problema de classificação, como observam MICHIE et al. [136]. O conjunto original de dados pertence ao *National Institute of Diabetes and Digestive and Kidney Diseases*. O conjunto de dados utilizado na literatura foi extraído deste conjunto original,

considerando-se todos os indivíduos do sexo feminino cuja idade é maior ou igual a 21 anos. A base de dados pode ser obtida no UCI Machine Learning Database [129].

O conjunto de dados contém 768 exemplos. Destes, 500 representam condições negativas para diabetes - valor de classe igual a zero - e 268 representam condições positivas - valor de classe igual a um. Cada exemplo é constituído de 8 atributos e do valor representativo da classe. Os atributos envolvidos são: número de vezes em que ficou grávida (A), concentração de glicose(B), pressão sangüínea (C), espessura da pele no tríceps (D), nível de insulina (E), índice de massa muscular (F), função de diabetes (G) e idade (H).

4.8.3.1 Aplicando o AGAE na base de dados original

A descrição dos resultados deste experimento está dividida em duas seções: na seção "a" o AGAE é aplicado na base de dados completa, enquanto que na seção "b" a base de dados original é dividida em três conjuntos, i.e. treinamento, validação e teste. Seguindo-se a metodologia previamente descrita, 460 exemplos foram usados para treinamento (300 da classe *zero* e 160 da classe *um*) e os conjuntos de validação e de teste são formados por 154 exemplos (100 da classe *zero* e 54 da classe *um*). Assim, os exemplos pertencentes a cada conjunto foram selecionados de maneira a se manter a proporção verificada no conjunto total - 65% de exemplos pertencentes à classe 1 (valor de classe nulo) e 35% de exemplos pertencentes à classe 2 (valor de classe unitário).

a) Base de dados completa

Inicialmente, o AGAE foi aplicado na base de dados completa (768 exemplos) para verificar se o mesmo separa os objetos de cada uma das classes corretamente. Para tal, foram usados 50 genótipos (51 grupos no máximo) estabelecendo-se um máximo de 1.000 gerações.

Utilizando a distância Euclidiana como medida de dissimilaridade, o AGAE encontrou dois grupos, um formado por 760 objetos e outro formado por 8 objetos (3 da classe *zero* e 5 da classe *um*). A função objetivo apresentou um valor igual a 0,75 após 23 gerações, indicando que um *agrupamento* adequado foi obtido. Contudo, observa-se que o AGAE não encontrou os grupos corretos. Assim, extraíram-se estes oito objetos e o AGAE foi aplicado novamente na base de dados formada por 760 objetos, mas

novamente a solução correta não foi obtida, pois o AGAE separou novamente 11 objetos (4 da classe *zero* e 7 da classe *um*) dos demais. Neste caso, o valor da função objetivo foi igual a 0,68 após 16 gerações. Tentou-se o mesmo procedimento novamente, mas o AGAE somente comprovou seu fraco desempenho neste problema, separando 6 (3 de cada classe) objetos dos 743 demais, para um valor da função objetivo igual a 0,59 após 7 gerações. Cada experimento consumiu 1 hora e dez minutos.

Resultados semelhantes foram obtidos quando a medida de distância *Manhattan* foi utilizada. Inicialmente o AGAE separou dois objetos (um de cada classe) dos demais, obtendo uma adequabilidade igual a 0,74 após 550 gerações. Repetindo-se o procedimento descrito no parágrafo anterior, estes dois objetos foram extraídos da base de dados e o AGAE foi novamente aplicado nos dados remanescentes. Desta vez, o AGAE separou 7 objetos (dois da classe *zero* e 5 da classe *um*) dos 759 demais, fornecendo uma adequabilidade igual a 0,65 após 843 gerações. Considerando-se os 759 objetos restantes, o AGAE obteve uma adequabilidade igual a 0,57 após 120 gerações e separou 16 objetos (7 da classe *zero)* dos demais.

b) Regras de Classificação

Esta seção descreve os resultados obtidos na aplicação do AGAE para encontrar regras de classificação que descrevam a base de dados original. O AGAE foi aplicado no conjunto de treinamento, no qual se obtém as regras de classificação, e avaliado nos conjuntos de validação e de teste.

Para a classe *zero*, formada por 300 objetos no conjunto de treinamento, obteve-se a melhor solução (VFO=0,82) após 72 gerações utilizando-se a medida de distância Euclidiana, enquanto que para a classe *um* obteve-se a melhor solução (VFO=0,71) após 459 gerações. O AGAE encontrou 147 sementes (população de 146 genótipos) para a classe *zero* e 74 sementes para a classe *um*. A classe *zero* é descrita por dois grupos (c_1 e c_2) formados por 298 e 2 objetos respectivamente e a classe *um* também é descrita por dois grupos (c_3 e c_4) formados por 158 objetos e 2 objetos respectivamente. A precisão de classificação das regras descritas por estes grupos é igual a 3,90% no conjunto de validação e 5,19% no conjunto de teste. Isto se deve ao fato de que os objetos das duas classes são muito parecidos. As regras abaixo permitem verificar esta característica:

Se { 0≤A≤13 e 0≤B≤193 e 0≤C≤122 e 0≤D≤60 e 0≤E≤545 e 0≤F≤47,9 e
0,078≤G≤1,781 e 21≤H≤81 } então c_1 – classe *zero*;
Se { 0≤A≤4 e 165≤B≤197 e 70≤C≤90 e 33≤D≤39 e 680≤E≤744 e
36,7≤F≤52,29 e 0,427≤G≤2,329 e 23≤H≤31 } então c_2 – classe *zero*;
Se { 0≤A≤17 e 0≤B≤197 e 0≤C≤110 e 0≤D≤56 e 0≤E≤543 e 0≤F≤67,1 e
0,127≤G≤2,228 e 21≤H≤67 então c_3 – classe *um*;
Se { A=1 e 172≤B≤189 e 60≤C≤68 e 23≤D≤49 e 579≤E≤846 e
30,1≤F≤42,4 e 0,398≤G≤0,702 e 28≤H≤59 então c_4 – classe *um*;

Fazendo-se uso da medida de distância de Manhattan, obtém-se a mesma solução, mas observa-se que a convergência acontece após um número maior de gerações. Para a classe *zero*, obtém-se uma adequabilidade igual a 0,77 após 458 gerações e para a classe *um* obtém-se uma adequabilidade igual a 0,65 após 582 gerações.

Considerando-se somente as regras que descrevem a classe zero, obtém-se taxas de classificação corretas médias iguais a 68,18% e 66,88% nos conjuntos de validação e de teste respectivamente, enquanto que ao considerar somente as regras que descrevem a classe um, obtém-se taxas de classificação corretas médias iguais a 33,77% e 36,36% nos conjuntos de validação e de teste respectivamente. O melhor conjunto de regras obtido pelo AGAT fornece uma taxa de classificação correta média igual a 58,60% no conjunto de treinamento.

Cumpre citar que, retirando-se os *grupos* c_2 e c_4 não se obtém melhorias significativas nas taxas de classificação, pois as regras obtidas passam a classificar corretamente 6,49% dos exemplos do conjunto de validação e 5,84% dos exemplos do conjunto de teste. Os experimentos envolvendo a classe *zero* consumiram 1 hora e 13 minutos e os experimentos envolvendo a classe *um* consumiram 15 minutos.

4.8.3.2 Regras obtidas por meio de uma rede neural

A metodologia utilizada nesta seção segue os mesmos passos utilizados nos exemplos anteriores. Inicialmente, treinou-se 40 modelos de redes neurais supervisionadas e os resultados, em termos do erro médio quadrado no conjunto de validação, estão descritos na Tabela 7. Os experimentos consideram apenas redes neurais com uma camada oculta, formada por 5 unidades no máximo. O algoritmo de aprendizado emprega uma taxa de aprendizado igual a 0,10 e de um termo de momento também igual a 0,10. Os pesos

foram iniciados aleatoriamente no intervalo [-0,3;+0,3]. Neste exemplo os exemplos foram apresentados à rede neural na ordem em que os mesmos aparecem no conjunto de treinamento e o erro médio quadrado foi verificado no conjunto de validação com freqüência de uma época.

Tabela 7. Erro médio quadrado após 1.000 épocas.

Classificação	Entrada	Oculta	Saída	Erro Validação
1	8 [-1,1]	2 logistic	2 logistic	0.39
2	8 [-1,1]	3 tanh	2 logistic	0.40
3	8 [0,1]	1 logistic	2 logistic	0.41
4	8 [-1,1]	4 logistic	2 logistic	0.42
5	8 [-1,1]	3 logistic	2 logistic	0.43
6	8 [0,1]	2 logistic	2 logistic	0.43
7	8 [-1,1]	1 logistic	2 logistic	0.44
8	8 [-1,1]	2 tanh	2 logistic	0.45
9	8 [0,1]	3 logistic	2 logistic	0.46
10	8 [-1,1]	5 logistic	2 logistic	0.47
11	8 [-1,1]	4 tanh	2 logistic	0.50
12	8 [0,1]	4 logistic	2 logistic	0.52
13	8 [-1,1]	5 tanh	2 logistic	0.55
14	8 [-1,1]	1 tanh	2 logistic	0.59
15	8 [0,1]	5 logistic	2 logistic	0.60
16	8 [-1,1]	5 logistic	2 tanh	1.48
17	8 [-1,1]	4 logistic	2 tanh	2.65
18	8 [0,1]	3 logistic	2 tanh	2.91
19	8 [-1,1]	3 logistic	2 tanh	3.50
20	8 [0,1]	5 logistic	2 tanh	3.67
21	8 [-1,1]	2 logistic	2 tanh	3.75
22	8 [0,1]	2 logistic	2 tanh	3.86
23	8 [0,1]	1 logistic	2 tanh	4.00
24	8 [0,1]	4 logistic	2 tanh	4.12
25	8 [-1,1]	1 logistic	2 tanh	4.62
26	8 [0,1]	1 logistic	2 linear	4.64
27	8 [-1,1]	2 logistic	2 linear	4.68
28	8 [-1,1]	2 tanh	2 linear	4.77
29	8 [-1,1]	3 logistic	2 linear	4.82
30	8 [-1,1]	3 tanh	2 linear	4.84
31	8 [0,1]	2 logistic	2 linear	4.91
32	8 [-1,1]	4 tanh	2 linear	4.93
33	8 [0,1]	4 logistic	2 linear	4.93
34	8 [-1,1]	4 logistic	2 linear	4.96
35	8 [0,1]	5 logistic	2 linear	4.96
36	8 [0,1]	3 logistic	2 linear	4.96
37	8 [-1,1]	5 tanh	2 linear	4.98
38	8 [-1,1]	5 logistic	2 linear	5.01
39	8 [-1,1]	1 logistic	2 linear	5.17
40	8 [-1,1]	1 tanh	2 linear	5.18

Ao contrário do exemplo da seção 4.8.2.2, quando as redes neurais convergiram antes da última época estipulada, as melhores redes neurais deste exemplo convergiram sempre na última época. Além disso, os erros médios quadrados obtidos no conjunto de validação foram relativamente altos. Por isso, optou-se por treinar as cinco melhores redes neurais até 5.000 épocas. Desta maneira, obtém-se os resultados descritos na Tabela 8.

Tabela 8. Melhores Redes Neurais – 5.000 épocas.

Classificação	Entrada	Oculta	Saída	Erro validação
1	8 [-1,1]	2 logistic	2 logistic	0.35
2	8 [-1,1]	3 tanh	2 logistic	0.37
3	8 [-1,1]	4 logistic	2 logistic	0.38
4	8 [0,1]	1 logistic	2 logistic	0.41
5	8 [-1,1]	3 logistic	2 logistic	0.42

Observando-se a Tabela 8, constata-se que o melhor modelo de rede neural, após 5.000 épocas, é composto por duas unidades ocultas. Conquanto o segundo e o terceiro melhores modelos fornecem erros médios semelhantes, eles também são mais complexos e portanto menos interessantes ao processo de extração de regras de redes neurais. Optou-se, então por treinar o melhor modelo até a sua convergência, que acontece após 9.070 épocas, fornecendo um erro médio quadrado no conjunto de validação igual a 0.345. As ativações das unidades ocultas deste modelo de rede neural são descritas segundo as seguintes expressões:

$$a_1 = 44{,}61-0{,}43A-0{,}17B+0{,}04C+0{,}16D-0{,}03E-0{,}51F-7{,}60G+0{,}02H$$

$$a_2 = 47{,}34-0{,}15A-0{,}14B+0{,}01C-0{,}03D+0{,}04E-0{,}54F-2{,}59G-0{,}68H$$

A extração de regras no espaço das ativações foi realizada por meio da mesma metodologia empregada anteriormente. Para a classe *zero*, o AGAE encontrou 152 sementes e a solução final representa dois grupos c_5 e c_6 formados por 298 e 2 objetos respectivamente. Esta solução (VFO=0,62) foi obtida após 63 gerações. Para a classe *um*, o AGAE encontrou 82 objetos representativos e a solução *ótima* foi obtida após 51 gerações (VFO=0,44). Esta solução é formada por três grupos (c_7, c_8 e c_9) formados por 78, 75 e 7 objetos respectivamente. Seguem as regras obtidas:

Se { $-13{,}19 \leq a_1 \leq 36{,}09$ e $-41{,}09 \leq a_2 \leq 21{,}40$ } então c_5 – classe *zero*;

Se { $-39{,}07 \leq a_1 \leq -23{,}71$ e $2{,}08 \leq a_2 \leq 6{,}90$ } então c_6 – classe *zero*;

Se { $-15{,}65 \leq a_1 \leq 11{,}28$ e $-40{,}76 \leq a_2 \leq -11{,}06$ } então c_7 – classe *um*;

Se { $-4{,}21 \leq a_1 \leq 28{,}00$ e $-19{,}64 \leq a_2 \leq 5{,}02$ } então c_8 – classe *um*;

Se { $-24{,}20 \leq a_1 \leq -11{,}09$ e $-6{,}65 \leq a_2 \leq 6{,}89$ } então c_9 – classe *um*;

Estas regras fornecem uma precisão de classificação igual a 12,98% no conjunto de validação e 11,69% no conjunto de teste. Estes resultados, ainda que não são satisfatórios, são melhores do que aqueles obtidos no espaço original e podem ser entendidos por meio da Figura 30, que mostra os valores das ativações para os exemplos de treinamento das duas classes.

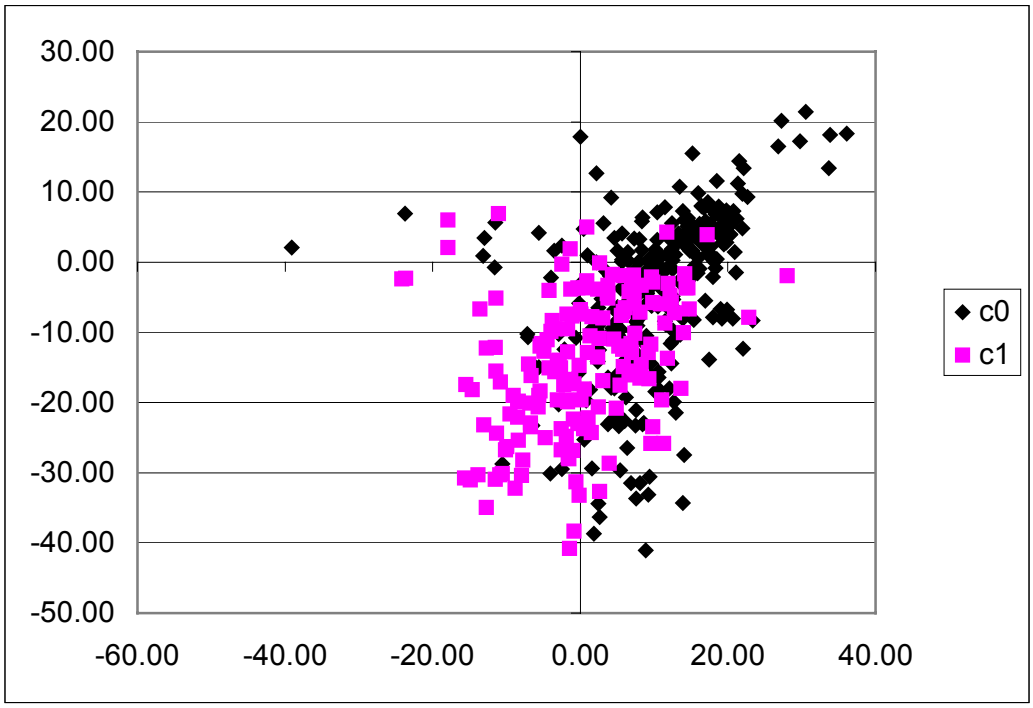

Figura 30. Valores das ativações da melhor rede neural obtida – Pima Indians.

Considerando-se somente as regras que descrevem a classe *zero*, e portanto tomando a classe *um* como *default*, obtém-se 66,88% de precisão no conjunto de validação e 66,23% no conjunto de teste. Considerando-se a classe *zero* como *default*, obtém-se 43,51% de precisão no conjunto de validação e 45,21% no conjunto de teste. Usando-se a distância Manhattan, verifica-se uma solução cuja precisão é igual à obtida quando do uso da distância Euclidiana. O AGAE encontra grupos idênticos para a classe

zero, mas converge após 45 gerações (VFO=0,59). Para a classe *um*, obtém-se três grupos formados por 77, 76, e 7 objetos após 326 gerações (VFO=0,44). Desta forma, a melhor estrutura de conhecimento obtida para este problema é aquela descrita na Figura 31.

Se $\{-13{,}19 \leq a_1 \leq 36{,}09$ e $-41{,}09 \leq a_2 \leq 21{,}40\}$ ou $\{-39{,}07 \leq a_1 \leq -23{,}71$ e $2{,}08 \leq a_2 \leq 6{,}90\}$ então negativo;

Senão positivo para diabetes;

Figura 31. Melhor Conjunto de Regras (Espaço das Ativações) – Pima Indians Diabetes.

Estas regras, cujas taxas de classificação corretas médias nos conjuntos de validação e de teste são iguais a 66,88% e 66,23% respectivamente, fornecem resultados melhores do que aqueles obtidos pelo Algoritmo RX Modificado (TCCM=41% na base de treinamento) e pelo Algoritmo Genético de Agrupamento Tradicional (TCCM=61% na base de treinamento). Entretanto, este desempenho é um pouco inferior quando comparado com os melhores resultados relatados pela literatura. MICHIE et al. [136] descrevem resultados obtidos por meio de vários métodos diferentes através do método da validação cruzada com 12 objetos. Nestes, observa-se que as taxas de classificação corretas médias nos conjuntos de teste variam de 67,60% até 77,70%. Além disso, considerando a Figura 30, observa-se que a exclusão dos menores grupos (c_6 e c_9) formados por 2 e 7 objetos respectivamente, muito provavelmente não aumentaria a taxa média de classificação correta das regras obtidas. No tocante ao custo computacional, o experimento envolvendo o AGC aplicado no espaço das ativações para obter as regras da classe *zero* consumiu 1 hora e 12 minutos, enquanto que o experimento envolvendo a classe *um* consumiu 13 minutos. A melhor rede neural treinada neste exemplo consumiu 6 minutos.

É possível melhorar os resultados obtidos nesta seção por meio de duas maneiras: (i) obter um melhor modelo de rede neural, i.e. com um erro médio quadrado no conjunto de validação menor, através de outros algoritmos de aprendizado e (ii) otimizar arquitetura da rede neural, principalmente no que diz respeito à seleção das variáveis mais importantes, diminuindo, assim, a complexidade das regras. Além disso, pode-se constatar que as melhores regras obtidas no espaço das ativações

proporcionaram resultados parecidos com aqueles obtidos no espaço original dos atributos.

4.8.4 Wisconsin Breast Cancer Database

Esta base de dados pode ser obtida em [129]. Cada objeto da base de dados é formado por 9 atributos e a classe associada {benigno,maligno}. Os atributos são: espessura da massa (A), uniformidade do tamanho da célula (B), uniformidade da superfície da célula (C), adesão marginal (D), tamanho da célula epitelial (E), núcleos abertos (F), cromatina (G), nucléolos normais (H) e mitoses (I). As duas classes não são linearmente separáveis. O número total de objetos é igual a 699, sendo 458 casos de câncer benigno e 241 de câncer maligno. Contudo, existem 16 exemplos com valores de atributos ausentes. Estes exemplos foram removidos e então o AGAE foi aplicado tanto no espaço original dos atributos como no espaço das ativações do melhor modelo de rede neural obtido.

4.8.4.1 Aplicando o AGAE no Espaço Original dos Atributos

Esta seção descreve a aplicação do AGAE no espaço original dos atributos. O primeiro experimento descreve a aplicação do AGAE para encontrar os grupos de objetos corretos, i.e. os grupos corretos que descrevem cada uma das classes na base completa de dados, formada pelos 683 objetos. Nesse caso, não se usou o método da validação cruzada, pois o experimento objetivava avaliar se o AGAE seria capaz de separar adequadamente os dois grupos de objetos. Neste caso, o experimento foi realizado utilizando a medida de distância Euclidiana, 50 genótipos e interrompendo o algoritmo após 1.000 gerações. O AGAE encontrou dois grupos após 8 gerações, com o valor da função objetivo (VFO) igual a 0,59. Este valor, compreendido no intervalo [0,51-0,70] indica que uma estrutura razoável foi encontrada [105]. O AGAE separou 94,30% dos exemplos corretamente e este resultado é comparável aos melhores descritos na literatura [20,102]. O experimento completo (1.000 gerações) consumiu 51 minutos.

Considerando novamente o fato de que o desempenho do AGAE será avaliado tanto no espaço original como no espaço das ativações, separamos a base de dados em três conjuntos: treinamento (60%), validação (20%) e teste (20%). O conjunto de treinamento usado é formado por 409 exemplos (266 da classe benigna e 143 da classe maligna). Os conjuntos de validação e de teste são formados por 137 exemplos (89 da classe benigna e 48 da classe maligna).

Para extrair as regras que modelam cada uma das classes envolvidas, é necessário separar os exemplos de cada uma delas no conjunto de treinamento. Assim, dois subconjuntos de treinamento são obtidos. Um, formado pelos exemplos da classe

benigna e outro formado pelos exemplos da classe maligna. Considerando a classe benigna, o AGAE encontrou 71 objetos representativos. Desta forma, a população deste experimento foi formada por 70 genótipos e a melhor solução foi obtida após 126 gerações (VFO=0,69). Esta solução representa dois grupos (c_1 e c_2) formados por 260 e 6 objetos respectivamente. Para a classe benigna, o AGAE encontrou 61 objetos representativos. Assim, a população deste experimento foi formada por 60 genótipos e a melhor solução foi obtida após 296 gerações (VFO=0,25). Esta solução representa dois grupos (c_3 e c_4) formados por 141 e 2 objetos respectivamente. Estes grupos podem ser traduzidos para o seguinte conjunto de regras:

Se {1≤A≤8 e 1≤B≤4 e 1≤C≤6 ∧ 1≤D≤6 e 1≤E≤10 e 1≤F≤10 e 1≤G≤7 e 1≤H≤8 e 1≤I≤7 } então c1;
Se {5≤A≤8 e 3≤B≤9 e 3≤C≤8 e 1≤D≤5 e 3≤E≤7 e 4≤F≤10 e 3≤G≤7 e 2≤H≤8 e 1≤I≤3 } então c2;
Se {1≤A≤10 e 1≤B≤10 e 1≤C≤10 e 1≤D≤10 e 1≤E≤10 e 1≤F≤10 e 1≤G≤10 e 1≤H≤10 e 1≤I≤10}então c3;
Se {A=10 e 8≤B≤10 e C=10 e D=10 e 6≤E≤10 e F=1 e 3≤G≤8 e 1≤H≤8 e 8≤I≤10 } então c4;

Este conjunto de regras classifica corretamente somente 30,65% dos exemplos do conjunto de validação e 32,11% dos exemplos do conjunto de teste. O problema com este conjunto de regras é que o terceiro grupo não possui poder de classificação, pois as premissas contemplam todos os exemplos da base de dados, porque todos as variáveis possuem valores que pertencem ao intervalo [1,10]. No entanto, ao considerar as regras formadas pelos grupos c_1 e c_2 obtém-se o seguinte conjunto de regras:

Se {1≤A≤8 e 1≤B≤4 e 1≤C≤6 e 1≤D≤6 e 1≤E≤10 e 1≤F≤10 e 1≤G≤7 e 1≤H≤8 e 1≤I≤7 } então benigno;
Se {5≤A≤8 e 3≤B≤9 e 3≤C≤8 e 1≤D≤5 e 3≤E≤7 e 4≤F≤10 e 3≤G≤7 e 2≤H≤8 e 1≤I≤3 } então benigno;
Senão maligno.

Figura 32 . Melhores regras obtidas no espaço original – Wisconsin Breast Cancer.

Estas regras classificam corretamente 94,89% dos exemplos do conjunto de validação e 97,81% dos exemplos do conjunto de teste. Observa-se, entretanto, que estas regras somente permitem saber quando um exemplo pertence à classe *maligno* por meio das características da classe *benigno*. Conquanto este tipo de conjunto de regras não corresponda à situação ideal, regras desta natureza são comuns em aplicações de data mining. O experimento envolvendo a classe *benigno* consumiu 71 minutos e o experimento envolvendo a classe *maligno* consumiu 7 minutos.

Um experimento interessante se constitui em excluir os objetos dos menores grupos obtidos (c_2 e c_4) e aplicar o AGAE novamente nos objetos remanescentes. Considerando-se a classe *benigna*, obtém-se uma solução formada por 2 grupos após

1.457 gerações (VFO=0,62), sendo um grupo formado por 258 objetos e o outro formado por 2 objetos. Considerando-se a classe *maligna*, obtém-se uma solução formada por 2 grupos após 307 gerações (VFO=0,23), sendo um grupo formado por 138 objetos e o outro formado por 3 objetos. O conjunto de regras obtido classifica corretamente somente 32,85% dos exemplos do conjunto de validação e 33,57% dos exemplos do conjunto de teste. Considerando somente as regras que classificam a classe *benigna*, as regras classificam corretamente 96,35% dos exemplos do conjunto de validação e 97,81% dos exemplos do conjunto de teste. De qualquer forma, observa-se que não se obtém melhorias significativas.

4.8.4.2 Aplicando o AGAE no Espaço das Ativações

Esta seção descreve a aplicação do AGAE no espaço das ativações de uma rede neural. Inicialmente, 40 redes neurais foram treinadas com o objetivo de se obter o menor erro possível no conjunto de validação. Os experimentos consideram redes neurais com uma camada oculta, formada por 5 unidades no máximo. O algoritmo de aprendizado emprega uma taxa de aprendizado igual a 0,10 e de um termo de momento também igual a 0,10. Os pesos foram iniciados aleatoriamente no intervalo [-0,3;+0,3]. Neste caso, os exemplos foram apresentados à rede neural na ordem em que os mesmos aparecem no conjunto de treinamento e o erro médio quadrado foi verificado no conjunto de validação com freqüência de uma época. A Tabela 9 descreve os resultados dessas simulações.

Como se pode observar, o melhor modelo de rede neural é formado por nove unidades de entrada com função de transformação linear [0,1] , duas unidades ocultas usando a função logística e duas unidades de saída com a função logística. Considerando este modelo, obtém-se as seguintes expressões para as ativações:

$a_1 = 8.34-0.51A-0.23B-0.40C-0.43D-0.04E-0.61F-0.12G-0.24H-0.27I;$

$a_2 = -5.52+0.36A-0.23B+0.15C+0.12D+0.10E+0.19F+0.23G+0.12H+0.32I;$

Utilizando estas expressões, calculam-se os valores das ativações para os exemplos de cada um dos conjuntos de dados (treinamento, validação e teste). Para extrair regras de classificação no espaço das ativações, dividiu-se a base de dados de treinamento em dois subconjuntos: um contendo os exemplos da classe *benigno* e outro contendo os exemplos da classe *maligno*. O AGAE foi, então, aplicado nestes

subconjuntos e as regras obtidas foram avaliadas nas ativações dos conjuntos de validação e de teste.

Tabela 9. Classificação das redes neurais – Wisconsin Breast Cancer Database.

Classificação	Entradas	Ocultas	Saída	Erro Validação
1	9 [0,1]	2 logistic	2 logistic	0.013
2	9 [0,1]	3 logistic	2 logistic	0.014
3	9 [-1,1]	2 logistic	2 logistic	0.014
4	9 [0,1]	1 logistic	2 logistic	0.014
5	9 [0,1]	5 logistic	2 logistic	0.015
6	9 [-1,1]	2 tanh	2 logistic	0.015
7	9 [0,1]	4 logistic	2 logistic	0.015
8	9 [-1,1]	1 tanh	2 logistic	0.015
9	9 [-1,1]	1 logistic	2 logistic	0.015
10	9 [-1,1]	4 logistic	2 logistic	0.017
11	9 [-1,1]	3 logistic	2 logistic	0.018
12	9 [-1,1]	3 tanh	2 logistic	0.018
13	9 [-1,1]	5 logistic	2 logistic	0.019
14	9 [-1,1]	4 tanh	2 logistic	0.020
15	9 [-1,1]	5 tanh	2 logistic	0.020
16	9 [0,1]	5 logistic	2 tanh	0.059
17	9 [0,1]	3 logistic	2 tanh	0.059
18	9 [0,1]	2 logistic	2 tanh	0.061
19	9 [0,1]	4 logistic	2 tanh	0.068
20	9 [0,1]	1 logistic	2 tanh	0.076
21	9 [-1,1]	1 logistic	2 tanh	0.084
22	9 [-1,1]	2 logistic	2 tanh	0.112
23	9 [-1,1]	4 logistic	2 tanh	0.115
24	9 [-1,1]	5 logistic	2 tanh	0.118
25	9 [-1,1]	3 logistic	2 tanh	0.119
26	9 [-1,1]	1 tanh	2 linear	0.145
27	9 [-1,1]	2 tanh	2 linear	0.749
28	9 [-1,1]	2 logistic	2 linear	2.219
29	9 [0,1]	2 logistic	2 linear	2.358
30	9 [0,1]	4 logistic	2 linear	2.683
31	9 [0,1]	3 logistic	2 linear	3.095
32	9 [0,1]	5 logistic	2 linear	3.270
33	9 [0,1]	1 logistic	2 linear	3.367
34	9 [-1,1]	1 logistic	2 linear	4.466
35	9 [-1,1]	3 tanh	2 linear	4.508
36	9 [-1,1]	4 tanh	2 linear	4.609
37	9 [-1,1]	5 logistic	2 linear	4.721
38	9 [-1,1]	3 logistic	2 linear	4.773
39	9 [-1,1]	4 logistic	2 linear	4.900
40	9 [-1,1]	5 tanh	2 linear	4.922

Considerando a classe *benigno*, o AGAE encontrou 81 objetos representativos (80 genótipos) e a melhor solução foi obtida após 128 gerações (VFO=0,81). Este valor indica que uma forte estrutura foi encontrada [105]. A melhor solução é formada por dois grupos: c_5 (formado por 260 objetos) e c_6 (formado por 6 objetos). Considerando a

classe *maligno*, o AGAE encontrou 70 objetos representativos (69 genótipos) e a melhor solução foi obtida após 66 gerações (VFO=0,52). A melhor solução é formada por dois grupos (c_7 e c_8) formados por 99 e 44 objetos respectivamente. Estes grupos, obtidos por meio da medida de distância Euclidiana, se traduzem nas regras descritas na Figura 33.

Se { $-3{,}43 \leq a_1 \leq 5{,}49$ e $-4{,}17 \leq a_2 \leq -0.02$ } então c5;	(benigno)
Se {$-8{,}14 \leq a_1 \leq -5{,}06$ e $-0{,}97 \leq a_2 \leq 2{,}58$ } então c6;	(benigno)
Se {$-19{,}44 \leq a_1 \leq -7{,}30$ e $0{,}21 \leq a_2 \leq 7{,}18$ }então c7;	(maligno)
Se {$-7{,}69 \leq a_1 \leq 0{,}54$ e $-1{,}88 \leq a_2 \leq 2{,}74$ } então c8;	(maligno)

Figura 33. Regras Obtidas no Espaço das Ativações – Wisconsin Breast Cancer

Estas regras classificam corretamente 89,78% dos exemplos de validação e 85,40% dos exemplos de teste. Assim, a aplicação do AGAE no espaço das ativações proporcionou bons resultados. Conforme observado anteriormente, é mais simples extrair padrões em um espaço transformado pelas unidades ocultas, que são detectoras de características. Para o presente caso, a transformação realizada pelas unidades ocultas pode ser visualizada na Figura 34.

No concernente ao custo computacional, o experimento envolvendo a classe benigno consumiu aproximadamente 1 hora, enquanto que o experimento envolvendo a classe *maligno* consumiu aproximadamente 8 minutos. Cumpre, ainda, tecer alguns comentários sobre o grupo c_6. Este representa 1,47% do número total de exemplos de treinamento. Estas proporções se referem ao conceito de *suporte de uma regra* [131], que é usado para avaliar o quão interessante uma regra pode ser. Neste contexto, algumas metodologias consideram que uma regra é interessante se o seu suporte é maior do que o suporte mínimo estabelecido pelo usuário [132]. Assim, pode-se supor que a regra descrita pelo referido grupo possui suporte inferior àquele desejável (por exemplo 5%). Desta forma, esta regra poderia ser desconsiderada. Fazendo isto, o conjunto de regras obtido seria formado apenas pelos grupos c_5, c_7 e c_8. Esse conjunto de regras está descrito na Figura 35.

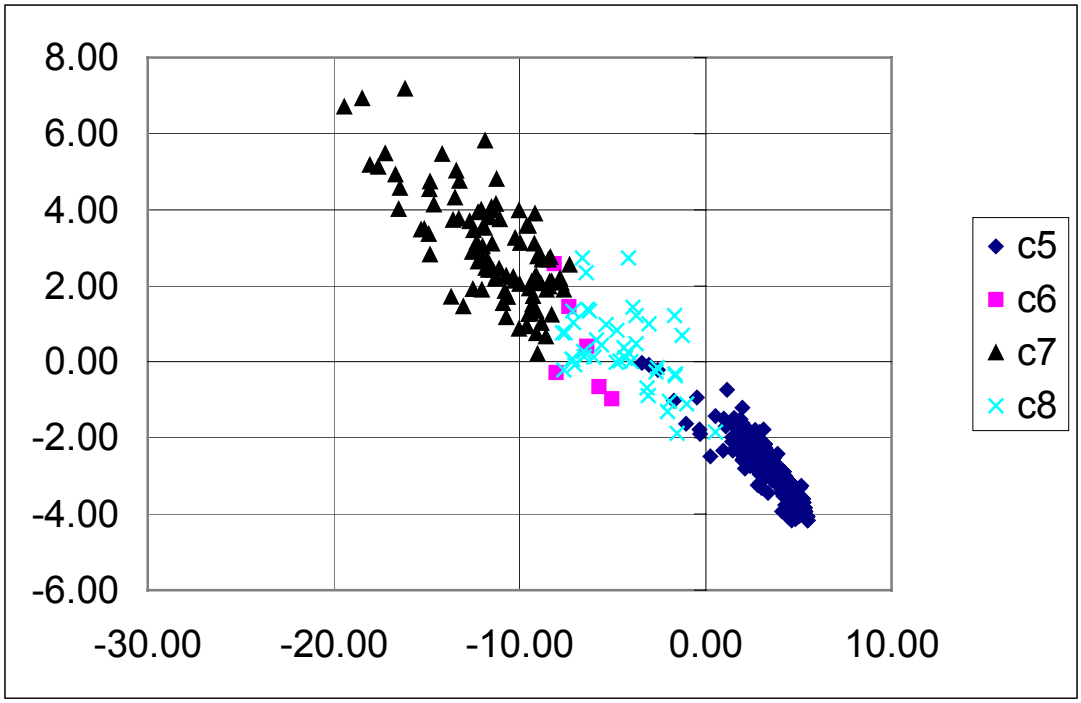

Figura 34. Grupos obtidos no espaço das ativações.

Se { $-3{,}43 \leq a_1 \leq 5{,}49$ e $-4{,}17 \leq a_2 \leq -0.02$ } então benigno;
Se {$-19{,}44 \leq a_1 \leq -7{,}30$ e $0{,}21 \leq a_2 \leq 7{,}18$ }então maligno;
Se {$-7{,}69 \leq a_1 \leq 0{,}54$ e $-1{,}88 \leq a_2 \leq 2{,}74$ } então maligno;

Figura 35. Melhor conjunto de regras obtido no espaço das ativações.

Estas regras classificam corretamente 97,81% dos exemplos de validação e 96,35% dos exemplos do conjunto de teste e este resultado é comparável aos melhores encontrados na literatura. Por exemplo, DUCH et al. [20] descrevem alguns dos melhores resultados obtidos nesta base de dados, haja vista que as taxas de classificação corretas relatadas variam entre 92,70% e 99,00%. Cumpre, todavia, observar que os resultados descritos em [20] consideram o método da validação cruzada para 10 exemplos, diferentemente daquele que utilizamos. Este método é muito eficaz para se avaliar precisamente a capacidade de generalização de um conjunto de regras. No entanto, os próprios autores argumentam que o teste de validação cruzada para 10 elementos é difícil de se realizar porque as regras tem de ser extraídas várias vezes. Além disso, não se usou este tipo de teste porque o mesmo fornece uma medida de precisão que está associada à média das avaliações para os diversos conjuntos de regras obtidos. Assim, é possível que nenhum dos conjuntos de regras obtidos tenha a precisão

obtida na média de todos os conjuntos de regras, o que dificulta a escolha do conjunto de regras representativo do modelo.

Ao considerar a modelagem do conjunto de dados por somente uma das classes, obtém-se resultados muito parecidos àqueles previamente descritos no que diz respeito às taxas de classificação corretas médias. Neste sentido, ao considerar a classe *maligno* como *default*, obtém-se regras que classificam 90,51% e 87,59% dos exemplos de validação e de teste respectivamente. Ao considerar a classe *benigno* como *default*, obtém-se regras que classificam 98,54% e 97,81% dos exemplos de validação e de teste respectivamente. Além disso, os resultados obtidos na modelagem de ambas as classes mostram que o desempenho das regras obtidas no espaço das ativações é muito superior ao das regras obtidas no espaço original.

No concernente aos resultados obtidos por meio da medida de distância de Manhattan, obtém-se resultados semelhantes àqueles obtidos por meio da medida de distância Euclidiana. Considerando a classe *benigno*, o AGAE encontrou 82 objetos representativos e a melhor solução foi obtida após 74 gerações (VFO=0,81). A melhor solução é formada por dois grupos, formados por 262 e 4 objetos respectivamente. Considerando a classe *maligno*, o AGAE encontrou 71 objetos representativos e a melhor solução foi obtida após 38 gerações (VFO=0,52). A melhor solução é formada por dois grupos, formados por 103 e 40 objetos respectivamente. Ao se aplicar o AGAE de maneira não supervisionada na base de dados de treinamento, o algoritmo separa corretamente 95,11% dos exemplos de treinamento.

Considerando as aplicações de *data mining*, as regras extraídas devem ser avaliadas quanto à precisão e quanto à complexidade. Neste sentido, cumpre enfatizar que redes neurais de arquiteturas otimizadas podem produzir melhores resultados, principalmente na aplicação de métodos para seleção das variáveis mais importantes.

4.8.5 Dados Meteorológicos

O conjunto de dados utilizado neste exemplo constitui-se de uma série de medições realizadas de hora em hora durante dez anos (1951 a 1960) no Aeroporto Internacional do Rio de Janeiro. O conjunto de dados, obtido por meio de um arquivo em cartões perfurados, contém 88.000 exemplos de observações meteorológicas, constituídas de 39 variáveis: código do cartão, número sinótico do bloco e da estação, ano, mês, data, hora, total de nuvens, direção (dezenas de graus) e velocidade (nós) do vento de

superfície, visibilidade horizontal (decâmetros), tempo presente (variável discreta), quantidade, tipo e altura (decâmetros) das nuvens para as camadas 1, 2, 3 e 4, ajuste do altímetro (milibares), temperatura do ponto de orvalho (graus celsius), velocidade de rajada (nós), pressão atmosférica ao nível do mar e da pista, direção das nuvens (código) nas camadas 1, 2, 3 e 4, variação da pressão em tendência e na diferença de 3 horas, temperatura (graus celsius) medida no bulbo úmido e no bulbo seco, precipitação (milímetros) e umidade relativa (em porcentagem).

Primeiramente, define-se o objeto de estudo, ou seja, a meta em termos de conhecimento. Cabe, aqui, a ressalva de que se dispõe apenas de um conhecimento bastante superficial do domínio em questão e que não se pretende elucidar definitivamente os fenômenos complexos da Meteorologia. Deseja-se, de maneira ilustrativa, mostrar a possibilidade de se obter algum conhecimento baseando-se somente no conjunto de dados. Obviamente que esta hipótese somente é válida em termos acadêmicos, e que, na prática, a presença de um especialista seria muito útil para orientar a aquisição de conhecimento e imprescindível para validá-lo. Sob estes aspectos, a variável *tempo presente* se mostrou muito atrativa para assumir o papel de variável dependente. Esta variável contém um código numérico, representado por números inteiros no intervalo [0,9] de maneira que cada código corresponde a uma condição de tempo de acordo com a Tabela 10.

Tabela 10. Classes envolvidas.

0	Nada a relatar
1	Precipitação à vista
2	Névoa seca / fumaça
3	Areia / poeira
4	Névoa úmida / nevoeiro
5	Chuvisco
6	Chuva
7	Neve
8	Trovoada / relâmpago
9	Granizo

Uma condição que se mostra bastante interessante diz respeito à presença de névoa úmida ou nevoeiro, pois este fenômeno é crítico para a operação de aeroportos. Neste aspecto, surgiu o interesse de se descobrir quais as variáveis importantes no processo e a relação entre as mesmas. Neste exemplo, utilizou-se de dois anos de observações:1959 e 1960. O ano de 1959 contém 8.709 observações, enquanto que o ano de 1960 contém 8.734 observações. Os atributos ano e os códigos dos cartões foram eliminados, pois

não influenciam na modelagem. Além disso, foram eliminados os seguintes atributos por conterem muitos valores ausentes: quantidade, tipo e altura de nuvens da terceira e da quarta camada de nuvens, rajada de vento e direção das nuvens nas camadas 1, 2, 3 e 4. Os atributos *código do cartão, número sinótico do bloco/estação* e *ano* foram eliminados porque não influenciam na modelagem. As classes 3 (areia/poeira), 7 (neve) e 9 (granizo) não contêm exemplos que as representem na base de dados. Além disso, optou-se por eliminar os exemplos que contém dados ausentes para as variáveis mantidas.

Obteve-se, então, uma base de dados formada por 1.226 exemplos. Levando-se em consideração o processo de treinamento de redes neurais, realizou-se a análise de correlação linear entre as variáveis envolvidas, haja vista que redes neurais aprendem melhor quando a correlação linear entre as variáveis independentes envolvidas é baixa [15]. Assim sendo, é vantajoso diminuir a dimensionalidade da base de dados através da eliminação de variáveis muito correlacionadas. A Tabela 11 mostra as variáveis mais correlacionadas (coeficiente de correlação linear maior ou igual a 0,9).

Tabela 11. Variáveis mais correlacionadas.

Variável "A"	Variável "B"	Correlação
Ajuste do Altímetro	Pressão ao nível do	1,0
Ajuste do Altímetro	Pressão ao nível da	1,0
Temperatura de	Temperatura do bulbo	0.9
Pressão ao nível do	Pressão ao nível da	1.0
Temperatura do bulbo	Temperatura do bulbo	0.9

Optou-se, então, por eliminar as seguintes variáveis: ajuste do altímetro, temperatura do bulbo seco e pressão ao nível do mar. Resumindo, teremos uma base dados formada por 1.226 exemplos e 20 variáveis: mês; data; hora; total de nuvens; direção do vento de superfície; velocidade do vento de superfície; visibilidade horizontal; tempo presente; quantidade, tipo e altura das nuvens (camadas 1 e 2 respectivamente); temperatura do ponto de orvalho; pressão atmosférica ao nível da pista; variação da pressão (tendência e na diferença de 3 horas respectivamente); temperatura medida no bulbo úmido; precipitação e umidade relativa.

Neste exemplo, deseja-se diferenciar a classe *névoa úmida* das demais, dado que desta forma pode-se encontrar as características que caracterizam um fenômeno importante para a operação de aeroportos. Para seguir a mesma metodologia aplicada anteriormente, é necessário dividir a base de dados em três conjuntos: treinamento

(60%), validação (20%) e teste (20%). O problema, então, recai em separar estas três bases de modo que os exemplos estejam adequadamente distribuídos.

Intuitivamente, a variável *mês* parece ser importante para modelar o fenômeno de nevoeiro, dada a sua relação direta com as estações do ano. Por outro lado, a variável *dia* parece ser menos importante, embora auxilie na caracterização das estações quando analisada em conjunto com a variável mês. Optou-se, então, por separar a base de dados por classes, obtendo-se 7 subconjuntos de dados, um para cada classe envolvida. Estes subconjuntos foram classificados de acordo com o atributo dia, pois assim é possível separar os exemplos de uma maneira tal que os valores do atributo mês estejam distribuídos mais uniformemente nas três amostras.

Os conjuntos de treinamento, validação e teste foram obtidos pela extração proporcional de exemplos destes subconjuntos de dados representativos de cada classe. Exemplificando: o subconjunto de exemplos da classe nevoeiro contém 512 exemplos. Os primeiros 60% se destinam ao conjunto de treinamento, os 20% seguintes ao conjunto de validação e os 20% últimos ao conjunto de teste. Resumidamente, os conjuntos de dados contém as quantidades de exemplos ilustradas na Tabela 12.

Todos os experimentos com o AGAE desta seção envolvem a utilização de populações formadas por 50 genótipos e o algoritmo foi interrompido após 1.000 gerações.

Tabela 12. Número de exemplos nos conjuntos de dados.

Cla	Exem	Treina	Valid	Test
0	302	180	61	61
1	34	20	7	7
2	125	75	25	25
4	512	306	103	103
5	67	39	14	14
6	152	90	31	31
8	34	20	7	7
Tot	1226	730	248	248

4.8.5.1 Aplicando o AGAE na base original de treinamento

Esta seção descreve a aplicação do AGAE na base original de treinamento, formada por 730 exemplos. O atributo tempo presente, que corresponde ao código da classe envolvida, não foi incluído no experimento, pois desejava-se verificar se o AGAE seria capaz de separar os objetos de acordo com as classes envolvidas. Desta forma, as variáveis envolvidas são:

x_1 - mês;

x_2 - hora;

x_3 - total de nuvens;

x_4 - direção do vento de superfície;

x_5 - velocidade do vento de superfície;

x_6 - visibilidade horizontal;

x_7 - quantidade de nuvens na camada 1;

x_8 - tipo de nuvens na camada 1;

x_9 - altura das nuvens para a camada 1;

x_{10} - quantidade de nuvens na camada 2;

x_{11} - tipo de nuvens na camada 2;

x_{12} - altura das nuvens para a camada 2;

x_{13} - temperatura do ponto de orvalho;

x_{14} - pressão atmosférica (pista);

x_{15} - variação da pressão em tendência;

x_{16} - variação da pressão – diferença de 3h;

x_{17} - temperatura do bulbo úmido;

x_{18} - precipitação;

x_{19} - umidade relativa;

y - tempo presente (não usada)

Usando-se a medida de distância Euclidiana, o AGAE encontrou dois grupos de objetos, um deles formado por 335 objetos e o outro por 395 objetos. Estes objetos pertencem às diversas classes, mas é interessante observar que os exemplos da classe *nada a relatar* foram agrupados todos num mesmo grupo. O valor da função objetivo (VFO) para este exemplo foi igual a 0,57, obtido após duas gerações. Usando-se a medida de distância de Manhattan, o AGAE encontrou três grupos de objetos formados por 333, 395 e 2 objetos respectivamente. Estes objetos pertencem às diversas classes, mas é interessante observar que os exemplos da classe *nada a relatar* foram novamente agrupados num mesmo grupo. O valor da função objetivo (VFO) para este exemplo foi igual a 0,48 e foi obtido após 53 gerações. Estes resultados mostram que o AGAE não encontrou uma solução razoável nesta aplicação, quer seja através da separação dos exemplos da classe *nevoeiro* dos demais exemplos ou até mesmo da separação das sete classes envolvidas. Cada um dos experimentos acima descritos consumiu 1 hora e 12 minutos.

4.8.5.2 Extração de regras na base original de treinamento

Com o intuito de extrair regras de classificação, separou-se a base de treinamento em dois subconjuntos: um formado pelos exemplos da classe *névoa úmida* e outro formado pelos exemplos das demais classes. O AGAE foi então aplicado em cada uma destas bases e as regras foram avaliadas nos conjuntos de teste e de validação. Cada experimento envolvendo a classe *névoa úmida* consumiu aproximadamente 15 minutos, e cada experimento envolvendo as demais classes consumiu aproximadamente 25 minutos.

Usando-se a medida de distância Euclidiana, o AGAE obteve dois grupos de objetos para a classe *nevoeiro* (c_1 e c_2) formados por 126 e 180 objetos respectivamente. Nesta classe, obteve-se um valor da função objetivo (VFO) igual a 0,54 após 5 gerações. Para as demais classes, o AGAE também obteve dois grupos de objetos (c_3 e c_4), formados por 268 e 156 objetos respectivamente (VFO=0,64) após 2 gerações. A taxa de classificação correta média nos conjuntos de validação e de teste, quando da utilização das regras fornecidas por esses quatro grupos, foi igual a 23,79% no conjunto de validação e 22,58% no conjunto de teste. No entanto, ao considerar apenas os grupos que modelam a classe *nevoeiro*, obtém-se um conjunto de regras cuja taxa de classificação correta média é igual a 62,50% no conjunto de validação e 59,27% no conjunto de teste.

Usando-se a medida de distância de Manhattan, o AGAE obteve dois grupos de objetos para a classe *nevoeiro* (c_5 e c_6) formados por 304 e 2 objetos respectivamente. Nesta classe, obteve-se um VFO igual a 0,51 após 28 gerações. Para as demais classes, o AGAE também obteve dois grupos de objetos (c_7 e c_8) formados por 269 e 155 objetos respectivamente (VFO=0,57) após 18 gerações. A taxa de classificação correta média nos conjuntos de validação e de teste, quando da utilização das regras descritas pelos grupos c_5, c_6, c_7 e c_8 foi igual a 21,77% no conjunto de validação e 22,18% no conjunto de teste. No entanto, ao considerar apenas as regras que modelam a classe *nevoeiro*, obtém-se uma taxa de classificação correta média igual a 60,48% no conjunto de validação e 59,68% no conjunto de teste. Cumpre observar, ainda, que quando o AGAE foi aplicado na base de dados de *nevoeiro* com os dois exemplos do grupo c_6 excluídos, obteve-se novamente uma solução formada por dois grupos, formados por 124 e 180 objetos. Considerando-se estes dois grupos descritivos da classe *nevoeiro*, obtém-se uma taxa de classificação correta média igual a 62,5% no conjunto de

validação e 59,27% no conjunto de teste. Pode-se observar que esta solução fornece taxas de classificação corretas médias iguais àquelas obtidas pela utilização da medida de distância Euclidiana.

Os resultados obtidos através do uso da medida de distância Euclidiana são muito parecidos com aqueles obtidos pelo uso da medida de Manhattan. Assim sendo, optou-se por descrever as regras obtidas para a classe nevoeiro por meio dos grupos c_1 e c_2 respectivamente:

Se $\{1 \leq x_1 \leq 12$ e $0 \leq x_2 \leq 21$ e $4 \leq x_3 \leq 8$ e $0 \leq x_4 \leq 36$ e $0 \leq x_5 \leq 20$ e $160 \leq x_6 \leq 1000$ e $1 \leq x_7 \leq 7$ e $3 \leq x_8 \leq 8$ e $10 \leq x_9 \leq 300$ e $3 \leq x_{10} \leq 8$ $1 \leq x_{11} \leq 6$ e $20 \leq x_{12} \leq 700$ e $15 \leq x_{13} \leq 25$ e $46 \leq x_{14} \leq 269$ e $0 \leq x_{15} \leq 8$ e $0 \leq x_{16} \leq 41$ e $155 \leq x_{17} \leq 268$ e $0 \leq x_{18} \leq 188$ $80 \leq x_{19} \leq 100$ $\}$ então nevoeiro;

Se $\{1 \leq x_1 \leq 12$ e $0 \leq x_2 \leq 21$ e $2 \leq x_3 \leq 8$ e $0 \leq x_4 \leq 36$ e $0 \leq x_5 \leq 17$ e $1200 \leq x_6 \leq 2000$ e $1 \leq x_7 \leq 7$ e $3 \leq x_8 \leq 8$ e $10 \leq x_9 \leq 350$ e $2 \leq x_{10} \leq 8$ $1 \leq x_{11} \leq 6$ e $45 \leq x_{12} \leq 700$ e $14 \leq x_{13} \leq 25$ e $25 \leq x_{14} \leq 268$ e $0 \leq x_{15} \leq 8$ e $0 \leq x_{16} \leq 44$ e $151 \leq x_{17} \leq 280$ e $0 \leq x_{18} \leq 280$ $76 \leq x_{19} \leq 100$ $\}$ então nevoeiro;

Além disso, nestas regras as variáveis x_1, x_2, x_4, x_7 e x_{15} não contribuem para esclarecer o fenômeno em questão, pois seus valores máximos e mínimos nas duas regras correspondem justamente aos valores máximos e mínimos observados nos exemplos de treinamento, validação e teste. Desta forma, pode-se eliminar estes atributos das regras sem o comprometimento da precisão de classificação, de maneira que as regras resultantes são:

Se $\{ 4 \leq x_3 \leq 8$ e $0 \leq x_5 \leq 20$ e $160 \leq x_6 \leq 1000$ e $3 \leq x_8 \leq 8$ e $10 \leq x_9 \leq 300$ e $3 \leq x_{10} \leq 8$ $1 \leq x_{11} \leq 6$ e $20 \leq x_{12} \leq 700$ e $15 \leq x_{13} \leq 25$ e $46 \leq x_{14} \leq 269$ e $0 \leq x_{16} \leq 41$ e $155 \leq x_{17} \leq 268$ e $0 \leq x_{18} \leq 188$ e $80 \leq x_{19} \leq 100$ $\}$ então *nevoeiro*;

Se $\{ 2 \leq x_3 \leq 8$ e $0 \leq x_5 \leq 17$ e $1200 \leq x_6 \leq 2000$ e $3 \leq x_8 \leq 8$ e $10 \leq x_9 \leq 350$ e $2 \leq x_{10} \leq 8$ $1 \leq x_{11} \leq 6$ e $45 \leq x_{12} \leq 700$ e $14 \leq x_{13} \leq 25$ e $25 \leq x_{14} \leq 268$ e $0 \leq x_{16} \leq 44$ e $151 \leq x_{17} \leq 280$ e $0 \leq x_{18} \leq 280$ e $76 \leq x_{19} \leq 100$ $\}$ então *nevoeiro*;

4.8.5.3 Treinamento de redes neurais supervisionadas

Esta seção trata da extração de regras de uma rede neural treinada para diferenciar as condições de nevoeiro de outras condições. As duas saídas da rede neural foram codificadas de tal modo que a classe nevoeiro é representada pelo vetor [0,1] e as outras classes são representadas pelo vetor [1,0]. Assim, utilizou-se da base de treinamento para ajustar os pesos das conexões. A base de validação foi usada para avaliar o desempenho das redes neurais, isto é, para salvar os pesos correspondentes ao menor erro quadrado médio. Inicialmente, alguns modelos de redes neurais foram treinados

para se avaliar aquele de melhor desempenho no presente problema de classificação. Para o treinamento, usou-se de uma taxa de aprendizado igual a 0.1 e de um termo de momento também igual a 0.1. Os pesos iniciais foram gerados aleatoriamente no intervalo [-0.3,0.3] e o treinamento foi interrompido após 3.000 épocas. Os erros quadrados médios são calculados após cada época. A Tabela 13 apresenta os resultados obtidos nos experimentos.

Tabela 13. Redes neurais treinadas até 3.000 épocas.

Classificação	Entradas	Ocultas	Saídas	Erro Validação
1	19 [-1,1]	4 logistic	2 logistic	0.3705
2	19 [-1,1]	3 logistic	2 logistic	0.4134
3	19 [-1,1]	5 logistic	2 logistic	0.4249
4	19 [0,1]	2 logistic	2 logistic	0.4414
5	19 [-1,1]	1 logistic	2 logistic	0.4538
6	19 [0,1]	1 logistic	2 logistic	0.4560
7	19 [0,1]	3 logistic	2 logistic	0.4561
8	19 [-1,1]	2 tanh	2 logistic	0.4609
9	19 [0,1]	5 logistic	2 logistic	0.4674
10	19 [0,1]	4 logistic	2 logistic	0.4697
11	19 [-1,1]	3 tanh	2 logistic	0.4702
12	19 [-1,1]	5 tanh	2 logistic	0.4777
13	19 [-1,1]	2 logistic	2 logistic	0.4803
14	19 [-1,1]	1 tanh	2 logistic	0.4847
15	19 [-1,1]	4 tanh	2 logistic	0.4923
16	19 [0,1]	5 logistic	2 tanh	2.3614
17	19 [-1,1]	5 logistic	2 tanh	2.4017
18	19 [0,1]	4 logistic	2 tanh	2.5444
19	19 [-1,1]	4 logistic	2 tanh	2.7501
20	19 [0,1]	3 logistic	2 tanh	2.7847
21	19 [-1,1]	3 logistic	2 tanh	3.0412
22	19 [0,1]	2 logistic	2 tanh	3.0657
23	19 [-1,1]	2 logistic	2 tanh	3.1090
24	19 [-1,1]	1 logistic	2 tanh	3.1400
25	19 [0,1]	1 logistic	2 tanh	3.1417

Para tentar diminuir o número de unidades de entrada do modelo de rede neural obtido, aplicou-se um método bastante simples, que se constitui em medir a contribuição de cada variável independente envolvida através da magnitude dos seus pesos associados. Este método, denominado de análise dos fatores de contribuição [135], basicamente se resume em somar os valores absolutos dos pesos para cada variável de entrada com o objetivo de estabelecer o grau de importância dessas para o modelo em questão. Assim, variáveis com elevados fatores de contribuição tem maior

importância para a modelagem considerada. A Tabela 14 lista os fatores de contribuição de cada uma das variáveis envolvidas no melhor modelo de rede neural obtido.

Tabela 14. Fatores de Contribuição.

Variável	Fator de Contribuição
x_1	3,51
x_2	2,61
x_3	6,57
x_4	3,91
x_5	5,52
x_6	19,92
x_7	2,06
x_8	6,26
x_9	5,50
x_{10}	2,94
x_{11}	6,41
x_{12}	7,46
x_{13}	5,46
x_{14}	15,06
x_{15}	3,42
x_{16}	4,46
x_{17}	12,85
x_{18}	13,66
x_{19}	16,91

Com base nos fatores de contribuição listados na tabela 14, tentou-se eliminar variáveis, no intuito de simplificar o modelo de rede neural, mas concomitantemente sem aumentar o erro médio quadrado no conjunto de validação. Desta forma, primeiramente eliminou-se a variável x_7, pois esta possui o menor fator de contribuição no modelo considerado. Treinou-se a rede neural novamente e o erro médio quadrado não aumentou. Assim, conclui-se que esta variável pode ser eliminada. Subseqüentemente, eliminou-se a variável x_2 e assim sucessivamente até concluir que é possível eliminar 6 variáveis (x_7, x_2, x_{10}, x_{15}, x_1 e x_4) sem que o erro médio quadrado no conjunto de validação aumente. Além disso, observa-se que o erro quadrado médio no conjunto de validação da rede neural podada foi igual a 0,15 após 405 épocas, isto é, obteve-se um modelo melhor do que o modelo formado por todas as variáveis.

Cumpre ressaltar, ainda, que as variáveis eliminadas são justamente aquelas desconsideradas na obtenção das regras de classificação obtidas pelo AGAE no espaço original dos atributos, a menos da variável x_{10}. A rede neural obtida proporciona um erro médio absoluto igual 0,20 no conjunto de treinamento e 0,22 no conjunto de

validação. Resumindo, a melhor rede neural obtida é formada por 13 unidades de entrada com transformação linear [-1,1], 4 unidades ocultas com função de ativação logística e 2 unidades de saída com função logística. As expressões abaixo descrevem as ativações de cada unidade oculta:

$a_1 = 25{,}03-0{,}21x_3+0{,}06181x_5+0{,}00176x_6-0{,}1x_8-0{,}00364x_9+0{,}0825x_{11}+0{,}0035x_{12}$
$-0{,}70857x_{13}-0{,}00245x_{14}-0{,}04346x_{16}+0{,}071414x_{17}-0{,}02081x_{18}-0{,}35636x_{19}.$

$a_2 = 4{,}0077+0{,}87333x_3-0{,}01091x_5-0{,}00281x_6-0{,}5033x_8+0{,}00139x_9-0{,}275x_{11}$
$-0{,}0045x_{12}+0{,}2186x_{13}-0{,}00447x_{14}+0{,}11962x_{16}+0{,}00465x_{17}-0{,}0084x_{18}-0{,}06364x_{19}.$

$a_3 = -3{,}06+0{,}3467x_3+0{,}1491x_5-0{,}00238x_6+0{,}26333x_8-0{,}0222x_9+0{,}7825x_{11}$
$-0{,}0049x_{12}+0{,}32286x_{13}-0{,}014x_{14}-0{,}0627x_{16}-0{,}0498x_{17}+0{,}03753x_{18}+0{,}06545x_{19}.$

$a_4=38{,}457+0{,}82667x_3+0{,}01455x_5-0{,}01211x_6-0{,}5467x_8-0{,}00467x_9-0{,}375x_{11}-0{,}00497x_{12}$
$-0{,}11x_{13}-0{,}00769x_{14}+0{,}00154x_{16}+0{,}01374x_{17}-0{,}01344x_{18}-0{,}17848x_{19}.$

Considerando-se as expressões de ativação acima descritas, pode-se calcular os valores das ativações para os exemplos de treinamento, validação e teste. A seção seguinte trata da aplicação do AGAE no espaço das ativações das unidades ocultas.

4.8.5.4 Aplicação do AGAE nas ativações do conjunto de treinamento

Esta seção descreve a aplicação do AGAE no espaço das ativações da base de dados usada para o treinamento da rede neural, com o objetivo de se verificar se os exemplos das diversas classes são separados adequadamente. Usando a medida de distância Euclidiana, o AGAE encontrou dois grupos, um formado por 728 objetos e outro formado por 2 objetos, após 634 gerações (VFO=0,50). Este resultado indica que os valores das ativações são semelhantes para os exemplos das várias classes envolvidas, o que acontece quando o modelo de rede neural incorpora uma representação distribuída [13,137]. Usando-se a distância de Manhattan, obtém-se a mesma solução, diferenciando-se somente no VFO, igual a 0,48 que foi obtido após 306 gerações. Cada experimento envolvendo o AGAE consumiu 1 hora e 10 minutos.

4.8.5.5 Extração de regras de classificação no espaço das ativações

Esta seção trata do processo de extração das regras da melhor rede neural treinada no presente problema de classificação. Para tal, aplicou-se o AGAE no espaço das ativações dos exemplos da classe *nevoeiro* e nos exemplos das demais classes

separadamente (procedimento idêntico àquele descrito na seção 4.8.5.2), avaliando as regras obtidas nos conjuntos de validação e de teste.

Considerando a medida de distância Euclidiana, o AGAE encontrou 2 grupos para os exemplos da classe *nevoeiro*, um deles formado por 303 objetos e o outro formado por 3 objetos. Esta solução foi obtida após 96 gerações (VFO=0,45). Para o conjunto de exemplos das demais classes, o AGAE também encontrou dois grupos, um formado por 293 objetos e o outro formado por 131 objetos e esta solução foi obtida na geração inicial (VFO=0,45). Considerando-se as regras que modelam ambas as classes (*nevoeiro* e *demais classes*) a taxa de classificação correta média no conjunto de validação (usado para obter o modelo) foi igual a 22,98% enquanto que no conjunto de teste este valor é igual a 24,19%. Considerando-se somente as regras que descrevem a classe *nevoeiro*, obtém-se uma taxa de classificação correta média igual a 61,29% no conjunto de validação e 64,92% no conjunto de teste. Considerando que um dos grupos da classe *nevoeiro* é formado por somente 3 objetos, optou-se por extrair estes e aplicar o AGAE nos 303 objetos remanescentes. Neste caso, o AGAE encontrou dois grupos após 44 gerações, um formado por 98 objetos e outro formado por 205 objetos. O valor da função objetivo foi igual a 0,37 mas as taxas de classificação corretas médias nos conjuntos de validação e de teste não mudaram significativamente (63,7% e 64,52% respectivamente).

Considerando a distância de Manhattan, obtém-se soluções muito parecidas. Para a classe *nevoeiro*, encontram-se dois grupos, um formado por 302 objetos e outro formado por 4 objetos após 519 gerações (VFO=0,45). Para os exemplos das demais classes, também são obtidos dois grupos, um formado por 292 objetos e outro formado por 132 objetos e esta solução foi obtida após 21 gerações (VFO=0,46). Considerando-se as regras que modelam ambas as classes (*nevoeiro* e *demais classes*) a taxa de classificação correta média no conjunto de validação foi igual a 22,58% enquanto que no conjunto de teste este valor é igual a 25,40%. Considerando-se somente as regras que descrevem a classe *nevoeiro*, obtém-se uma taxa de classificação correta média igual a 61,29%, idêntica àquela obtida pelo uso da distância Euclidiana, no conjunto de validação e 64,52% no conjunto de teste.

Desta forma, o melhor conjunto de regras obtido para este problema descreve a classe *nevoeiro* com o uso da medida de distância Euclidiana. Considerando as expressões de ativação previamente descritas, seguem as regras obtidas (a primeira regra representa o maior grupo):

Se {$-11,63 \leq a_1 \leq 2,42$ e $-3,99 \leq a_2 \leq 7,08$ e $-15,45 \leq a_3 \leq 4,97$ e $-5,45 \leq a_4 \leq 20,83$}
então nevoeiro;
Se {$-14,06 \leq a_1 \leq -10,77$ e $-0,37 \leq a_2 \leq 3,14$ e $8,07 \leq a_3 \leq 10,86$ e $3,53 \leq a_4 \leq 12,94$}
então nevoeiro;

Observa-se, então, que neste exemplo as regras obtidas no espaço das ativações não proporcionaram melhorias significativas em termos de precisão de classificação. Por outro lado, aplicando-se o AGAE no espaço original formado pelas variáveis selecionadas pelo modelo de rede neural, observa-se que as taxas médias de classificação corretas são mantidas, mas obtém-se regras mais simples. Mais especificamente, quando se aplica o AGAE nos exemplos de treinamento da classe *nevoeiro* (formados pelos 13 atributos usados no melhor modelo de rede neural) usando a medida de distância Euclidiana, obtém-se 2 grupos, um formado por 126 objetos e outro formado por 180 objetos. Estes grupos, que proporcionam um valor máximo da função objetivo igual a 0,54 e foram obtidos após 8 gerações, se traduzem em regras que classificam corretamente 62,50% e 59,27% dos exemplos de validação e de teste respectivamente. Usando a medida de distância de Manhattan, obtém-se um grupo formado por 304 objetos e outro formado por 2 objetos (VFO=0,52) após 10 gerações, e estes classificam corretamente 60,48% e 59,68% dos exemplos de validação e de teste respectivamente. Novamente, foram excluídos estes dois objetos e aplicou-se o AGAE novamente nos 304 objetos remanescentes. Desta forma, obtém-se a melhor solução (VFO=0,46) após 16 gerações e encontram-se 2 grupos, um formado por 124 objetos e outro formado por 180 objetos. Esta solução fornece taxas de classificação corretas iguais àquelas obtidas pelo uso da medida de distância Euclidiana (62,50% e 59,27% para os conjuntos de validação e de teste respectivamente). Desta forma, o melhor conjunto de regras obtido para este problema, pela sua menor complexidade, classificando corretamente 62,50% dos exemplos de validação e 59,27% dos exemplos de teste é:

Se { $4 \leq x_3 \leq 8$ e $0 \leq x_5 \leq 20$ e $160 \leq x_6 \leq 1000$ e $3 \leq x_8 \leq 8$ e $10 \leq x_9 \leq 300$ e $1 \leq x_{11} \leq 6$ e $20 \leq x_{12} \leq 700$ e $15 \leq x_{13} \leq 25$ e $46 \leq x_{14} \leq 269$ e $0 \leq x_{16} \leq 41$ e $155 \leq x_{17} \leq 268$ e $0 \leq x_{18} \leq 188$ e $80 \leq x_{19} \leq 100$ }
então *nevoeiro*;

Se { $2 \leq x_3 \leq 8$ e $0 \leq x_5 \leq 17$ e $1200 \leq x_6 \leq 2000$ e $3 \leq x_8 \leq 8$ e $10 \leq x_9 \leq 350$ e $1 \leq x_{11} \leq 6$ e $45 \leq x_{12} \leq 700$ e $14 \leq x_{13} \leq 25$ e $25 \leq x_{14} \leq 268$ e $0 \leq x_{16} \leq 44$ e $151 \leq x_{17} \leq 280$ e $0 \leq x_{18} \leq 280$ e $76 \leq x_{19} \leq 100$ }
então *nevoeiro*;

O leitor observará que este conjunto de regras é idêntico àquele obtido pela aplicação do AGAE no espaço original, após a eliminação das variáveis sem poder de classificação – seção 4.8.5.2 – a menos da variável x_{10}. Conquanto os resultados obtidos neste exemplo são bastante consistentes, entende-se que as taxas de classificação corretas poderiam ser melhores. Uma possível causa para estas baixas taxas de classificação corretas pode advir da maneira como os diferentes exemplos foram apresentados à rede neural. Tendo-se em mente que as saídas da rede neural foram modeladas em duas unidades, para as quais o vetor (0,1) significa *exemplos de nevoeiro* e (1,0) significa *outras condições*, pode-se levantar uma questão: será possível uma rede neural aprender que condições tão diferentes como por exemplo *chuva* e *névoa seca* são *semelhantes*, isto é, sob o ponto de vista do aprendizado, pertencentes à mesma classe? Uma possível resposta para esta questão é obtida dos resultados descritos na próxima seção. A título de ilustração, a melhor rede neural obtida foi treinada em 40 segundos (405 épocas). No tocante aos experimentos realizados com o AGAE, observou-se que os exemplos da classe *nevoeiro* consumiram aproximadamente 13 minutos e os experimentos envolvendo os exemplos das demais classes consumiram 21 minutos.

4.8.5.6 Codificando sete classes diferentes

Para tentar melhorar a precisão das regras obtidas, optou-se por treinar redes neurais em que as diversas condições de tempo presentes na base de dados são codificadas separadamente. Para tal, foram utilizadas 7 unidades de saída na rede neural, uma para cada condição de tempo. Por exemplo, a classe *nevoeiro* é codificada pelo vetor [0,0,0,0,0,0,1] e a classe *nada a relatar* é codificada pelo vetor [0,0,0,0,0,1,0]. No que diz respeito ao treinamento das redes neurais, optou-se pela utilização da melhor arquitetura – no que diz respeito às diversas transformações e funções de ativação usadas - obtida anteriormente para este mesmo problema – seção 4.8.5.3 – variando-se o número de unidades ocultas de 1 a 5. As redes foram treinadas considerando-se que a taxa de aprendizado e o termo de momento são iguais a 0,10. Os pesos foram novamente iniciados aleatoriamente no intervalo [-0,3,+0,3]. O erro no conjunto de validação foi continuamente verificado após a apresentação aleatória de 730 exemplos e o treinamento foi interrompido após 3.000 épocas. A Tabela 15 mostra os resultados obtidos.

Tabela 15. Redes neurais treinadas com sete classes.

Classificação	Entradas	Ocultas	Saídas	Erro Validação
1	13 [-1,1]	4 logistic	7 logistic	0,2109
2	13 [-1,1]	5 logistic	7 logistic	0,2112
3	13 [-1,1]	3 logistic	7 logistic	0,2123
4	13 [-1,1]	2 logistic	7 logistic	0,2718
5	13 [-1,1]	1 logistic	7 logistic	0,3190

A melhor rede foi então treinada até 10.000 épocas mas não produziu melhores resultados. O menor erro quadrado médio no conjunto de validação foi obtido após 849 épocas. Optou-se, então, por tentar diminuir o número de variáveis de entrada, seguindo a metodologia previamente descrita na seção 4.8.5.3. Eliminando-se as variáveis x_i {i = 3, 5, 8, 9, 11, 12, 16} e mantendo-se as variáveis x_j {j = 6, 13, 14, 17, 18, 19} obtém-se um erro médio quadrado igual a 0,2143 após 638 épocas. Neste caso, a perda de precisão, igual a 1,6% é compensada pela simplificação do modelo de rede neural, o que favorecerá a obtenção de regras menos complexas. Resumindo, a rede neural escolhida para a extração de regras é formada por seis unidades de entrada, 4 unidades ocultas e sete unidades de saída. As variáveis selecionadas são: x_6 (visibilidade), x_{13} (temperatura do ponto de orvalho), x_{14} (pressão ao nível da pista), x_{17} (temperatura do bulbo úmido), x_{18} (precipitação) e x_{19} (umidade relativa). As expressões das ativações da rede neural em questão são:

$a_1 = -22{,}0247+0{,}01253x_6+0{,}06714\,x_{13}+0{,}00264\,x_{14}-0{,}00677\,x_{17}-0{,}00535\,x_{18}-0{,}03\,x_{19}.$

$a_2 = -28{,}3366-0{,}0014x_6+0{,}45714\,x_{13}-0{,}0029\,x_{14}-0{,}059\,x_{17}+0{,}02827\,x_{18}+0{,}44636\,x_{19}.$

$a_3 = 17{,}08-0{,}00813x_6+0{,}07714\,x_{13}+0{,}00034\,x_{14}+0{,}02152\,x_{17}-0{,}00265\,x_{18}-0{,}15636\,x_{19}.$

$a_4 = 10{,}0222-0{,}00155x_6+0{,}12286\,x_{13}+0{,}00809\,x_{14}+0{,}00777\,x_{17}-0{,}02971\,x_{18}-0{,}15848\,x_{19}.$

Os valores das ativações para cada exemplo foram então calculados e o AGAE foi aplicado na base de treinamento e as regras obtidas foram avaliadas nos conjuntos de validação e de teste. O AGAE foi aplicado seguindo a mesma metodologia da seção 4.8.5.5. O conjunto de treinamento foi dividido em dois: um formado pelas ativações dos exemplos de *nevoeiro* e outro formado pelas ativações dos demais exemplos.

Para a medida de distância Euclidiana, foram obtidos dois grupos (formados por 127 e 179 objetos) nos exemplos da classe *nevoeiro*. O valor da função objetivo foi igual a 0,45 e foi obtido após 17 gerações. Para os demais exemplos, obteve-se também dois grupos (formados por 285 e 139 objetos) após 28 gerações (VFO=0,51). Considerando-se estes quatro grupos, obtém-se um conjunto de regras que classifica

corretamente 30,24% dos exemplos de validação e 27,02% dos exemplos de teste. Ao considerar somente os *grupos* da classe *nevoeiro,* o conjunto de regras classifica corretamente 71,37% dos exemplos de validação e 68,15% dos exemplos de teste. Como se pode observar, estes resultados são significativamente melhores do que aqueles descritos na seção 4.8.5.5.

Para a medida de distância de Manhattan, foram também obtidos dois grupos (formados por 104 e 202 objetos) nos exemplos da classe *nevoeiro*. O valor da função objetivo foi igual a 0,46 e foi obtido após 53 gerações. Para os demais exemplos, também foram obtidos dois grupos (formados por 282 e 142 objetos) após 29 gerações e o valor da função objetivo foi igual a 0,51. Considerando-se estes quatro *grupos*, obtém-se um conjunto de regras que classifica corretamente 31,05% dos exemplos de validação e 29,03% dos exemplos de teste. Considerando-se somente os grupos da classe *nevoeiro,* o conjunto de regras classifica corretamente 72,18% dos exemplos de validação e 71,37% dos exemplos de teste.

Para os resultados obtidos com o uso da medida de distância de Manhattan, observa-se uma melhoria média de 13,74% em relação à precisão de classificação das regras extraídas da rede neural descrita na seção 4.8.5.5. Além disso, entende-se que, considerando a complexidade da base de dados e a diminuição da complexidade das regras (somente 6 variáveis envolvidas), estes resultados são animadores. As regras abaixo representam a estrutura de conhecimento obtida através do uso da medida de distância de Manhattan:

Se$\{-22{,}78 \leq a_1 \leq -11{,}45$ e $0{,}26 \leq a_2 \leq 12{,}67$ e $0{,}31 \leq a_3 \leq 9{,}39$ e $-3{,}32 \leq a_4 \leq 2{,}52\}$ então nevoeiro;
Se$\{-13{,}02 \leq a_1 \leq 0{,}72$ e $-1{,}93 \leq a_2 \leq 17{,}09$ e $-8{,}23 \leq a_3 \leq 1{,}78$ e $-10{,}01 \leq a_4 \leq 1{,}32\}$ então nevoeiro;

Figura 36. Melhores regras obtidas no espaço das ativações.

A fim de tentar fornecer uma comparação *mais justa* com a aplicação do AGAE no espaço original dos atributos, desenvolveu-se uma nova simulação considerando-se somente os atributos selecionados pela melhor rede neural obtida, isto é: x_6, x_{13}, x_{14}, x_{17}, x_{18}, x_{19}.

Para a medida de distância Euclidiana, foram obtidos três grupos (formados por 73,106 e 127 objetos) nos exemplos da classe *nevoeiro*. O valor da função objetivo foi igual a 0,60 e foi obtido após 46 gerações. Para os demais exemplos, foram obtidos dois *grupos* (formados por 268 e 156 objetos) após 17 gerações e o valor da função objetivo

foi igual a 0,71. Considerando-se estes cinco grupos, obtém-se um conjunto de regras que classifica corretamente 26,21% dos exemplos de validação e 22,58% dos exemplos de teste. Se considerarmos somente os grupos da classe *nevoeiro,* o conjunto de regras classifica corretamente 62,90% dos exemplos de validação e 61,29% dos exemplos de teste. Como se pode observar, estes resultados são significativamente piores do que aqueles obtidos no espaço das ativações.

Para a medida de distância de Manhattan, foram obtidos oito grupos (formados por 45, 52, 85, 53, 5, 24, 37, e 5 objetos) nos exemplos da classe *nevoeiro*. O valor da função objetivo foi igual a 0,57 (após 86 gerações). Para os demais exemplos, foram obtidos três grupos (formados por 266, 156 e 2 objetos) após 4 gerações e o valor da função objetivo foi igual a 0,68. Considerando-se estes 11 grupos, obtém-se um conjunto de regras que classifica corretamente 37,90% dos exemplos de validação e 35,48% dos exemplos de teste. Considerando-se somente os grupos da classe *nevoeiro,* o conjunto de regras classifica corretamente 72,18% dos exemplos de validação e 73,79% dos exemplos de teste. Como se pode observar, a seleção das variáveis realizada através do treinamento da melhor rede neural também foi útil para se obter melhores resultados no espaço original dos atributos. Além disso, estes resultados são ligeiramente superiores àqueles obtidos no espaço das ativações e as regras obtidas são as seguintes:

Se { $400 \leq x_6 \leq 700$ e $15 \leq x_{13} \leq 24$ e $89 \leq x_{14} \leq 261$ e $170 \leq x_{17} \leq 268$ e $0 \leq x_{18} \leq 121$ $80 \leq x_{19} \leq 97$ } então *nevoeiro*;
Se { $900 \leq x_6 \leq 1000$ e $15 \leq x_{13} \leq 25$ e $46 \leq x_{14} \leq 269$ e $155 \leq x_{17} \leq 268$ e $0 \leq x_{18} \leq 188$ $80 \leq x_{19} \leq 100$ } então *nevoeiro*;
Se { $1500 \leq x_6 \leq 1800$ e $16 \leq x_{13} \leq 24$ e $48 \leq x_{14} \leq 266$ e $179 \leq x_{17} \leq 270$ e $0 \leq x_{18} \leq 60$ $76 \leq x_{19} \leq 96$ } então *nevoeiro*;
Se { $x_6 = 1200$ e $14 \leq x_{13} \leq 24$ e $83 \leq x_{14} \leq 268$ e $151 \leq x_{17} \leq 261$ e $0 \leq x_{18} \leq 280$ $80 \leq x_{19} \leq 100$ } então *nevoeiro*;
Se { $1800 \leq x_6 \leq 2000$ e $19 \leq x_{13} \leq 22$ e $95 \leq x_{14} \leq 177$ e $202 \leq x_{17} \leq 272$ e $0 \leq x_{18} \leq 4$ $82 \leq x_{19} \leq 96$ } então *nevoeiro*;
Se { $800 \leq x_6 \leq 900$ e $18 \leq x_{13} \leq 23$ e $92 \leq x_{14} \leq 239$ e $182 \leq x_{17} \leq 262$ e $0 \leq x_{18} \leq 76$ $80 \leq x_{19} \leq 100$ } então *nevoeiro*;
Se { $x_6 = 1400$ e $16 \leq x_{13} \leq 25$ e $25 \leq x_{14} \leq 244$ e $180 \leq x_{17} \leq 280$ e $0 \leq x_{18} \leq 255$ $80 \leq x_{19} \leq 96$ } então *nevoeiro*;
Se { $160 \leq x_6 \leq 300$ e $19 \leq x_{13} \leq 23$ e $79 \leq x_{14} \leq 256$ e $210 \leq x_{17} \leq 264$ e $0 \leq x_{18} \leq 48$ $81 \leq x_{19} \leq 96$ } então *nevoeiro*;

Figura 37. Melhores regras obtidas no espaço original dos atributos.

O AGAE também foi aplicado de maneira *não supervisionada* na base de dados de treinamento formada por seis atributos, mas os resultados não foram satisfatórios. Considerando medida de distância Euclidiana, obteve-se dois grupos após 5 gerações, sendo que um grupo formado por 335 objetos e o outro formado por 395 objetos (VFO=0,62). Para a medida de distância de Manhattan, obteve-se uma solução cujo valor da função objetivo foi igual a 0,59 após 2 gerações e obteve-se 8 grupos. O único ponto positivo destas simulações se deu na colocação de todos os objetos da classe *nada a relatar* no mesmo *grupo*.

Considerando apenas o processo de extração de regras (sem levar em conta as tarefas de preparação dos dados) o custo computacional para obter o conjunto de regras da Figura 36 foi igual a 14 minutos (1 minuto no treinamento da rede neural ótima e 13 minutos na aplicação do AGAE na base de dados formada pelas ativações da rede neural). O conjunto de regras descrito na Figura 37 também consumiu 14 minutos. Cumpre observar que o custo computacional da aplicação do AGAE pode variar quando se repete um mesmo experimento. Isto acontece porque o algoritmo depende da geração de números aleatórios que, por sua vez, podem produzir soluções diferentes em cada experimento. A variação no custo computacional advém do tratamento das soluções inválidas que podem ser geradas, como por exemplo: agrupamentos formados por um único grupo, eliminação de um grupo na aplicação do operador de *mutação* em genótipos formados por apenas dois grupos, etc. Além disso, também se deve considerar o custo computacional de treinamento das outras redes neurais que serviram para escolher as melhores variáveis, de modo que os valores descritos servem somente como um indicativo do custo computacional do método desenvolvido.

Observando-se os resultados obtidos, constata-se que as regras obtidas são muito interessantes, principalmente quando se leva em conta a complexidade dos fenômenos meteorológicos. Assim sendo, a precisão das regras obtidas sugere que o método pode ser útil na modelagem de dados meteorológicos. Além disso, a *descoberta* das variáveis mais importantes nesta modelagem também é muito importante, haja a vista a redução

CAPÍTULO 5

Conclusões

Redes neurais supervisionadas têm sido utilizadas em muitas aplicações de *data mining*. Cumpre observar, entretanto, que a utilização dessas redes apresenta dois obstáculos principais: (i) grande tempo de processamento na fase de definição do modelo (treinamento da rede neural) e (ii) falta de compreensibilidade do modelo obtido. O tempo de treinamento pode ser diminuído por meio da utilização de processamento paralelo e distribuído, bem como pela utilização de amostras de treinamento adequadas [138]. Para tentar superar a falta de compreensibilidade do modelo obtido, faz-se uso de *Algoritmos de Extração de Regras de Redes Neurais*. Nesse sentido, o desenvolvimento de algoritmos eficazes para extração de regras de redes neurais, mais especificamente de Perceptrons de Múltiplas Camadas (PMC), é uma tarefa importante e se constitui na principal contribuição do presente trabalho.

Conquanto os PMC possam alcançar altas taxas de classificação corretas, o conhecimento adquirido por tais redes neurais é normalmente incompreensível aos humanos [20]. Este fato pode ser um obstáculo importante em tarefas de *data mining,* nas quais é importante obter regras simbólicas ou outras formas de *estruturas de conhecimento* [21]. Assim, muitos métodos tem sido desenvolvidos para obter conhecimento explícito destes modelos e a descrição de alguns trabalhos sobre esse assunto pode ser encontrada na seção 1.2. Conforme se pode observar em alguns trabalhos citados nesta seção, alguns autores notaram a necessidade de se simplificar os modelos de redes neurais, sendo favoráveis à utilização de arquiteturas e de algoritmos de treinamento específicos para o processo de extração de regras. Todavia, entende-se que o desenvolvimento de algoritmos de extração de regras *mais genéricos* também é importante, principalmente porque estes podem ser usados em modelos de redes neurais previamente treinados bem como em novos problemas em que redes neurais sejam potencialmente úteis.

No concernente à aplicabilidade, o método proposto se destina a obter conhecimento, na forma de regras lógicas, de perceptrons de múltiplas camadas treinados em problemas de classificação que envolvam atributos discretos e/ou contínuos. Além disso, o método proposto não requer algoritmos de treinamento

específicos, isto é, o processo de extração de regras depende apenas do conhecimento da arquitetura da rede neural. Cumpre observar, entretanto, que a utilização de funções lineares nas unidades de entrada permite a extração de regras mais compreensíveis quando da aplicação da metodologia desenvolvida.

O método proposto basicamente se constitui no aperfeiçoamento da metodologia empregada pelo Algoritmo RX Modificado [22,81,82,83]. Assim, supõe-se a disponibilidade de uma rede neural treinada, através da qual são computados os valores das ativações dos neurônios da primeira camada oculta. Essas ativações formam um vetor para cada objeto da base de dados. Os vetores são separados, por meio de algoritmos de *agrupamento*, em subconjuntos de acordo com a classe a qual pertencem, isto é, o processo de extração de regras é realizado separadamente para cada classe envolvida.

Em termos gerais, pode-se dizer que a metodologia empregada para extrair regras de perceptrons de múltiplas camadas se resume a: (i) aplicar um algoritmo de *agrupamento* para encontrar *grupos* de ativações das unidades ocultas para cada classe e (ii) relacionar as expressões de ativação de cada unidade oculta aos valores mínimos e máximos das ativações dos *grupos* obtidos, que podem ser traduzidos em regras de classificação. Dessa forma, pode-se descrever as ativações das unidades ocultas em função das unidades de entrada que, por sua vez, estão relacionadas às variáveis do problema em questão. Essa metodologia fornece um meio de se relacionar regiões do domínio das variáveis às classes, por meio do agrupamento dos valores das ativações.

Além disso, pode-se citar que o processo de extração de regras se dá através da obtenção de hiperplanos que dividem o espaço dos atributos transformado linearmente. Esta transformação será linear quando a rede neural usar funções lineares nas unidades de entrada. Nos experimentos realizados nesse trabalho, somente dois tipos de transformações são aplicadas nas unidades de entrada da rede neural: a função linear [0,1] e a função linear [-1,+1], pois, desta forma, pode-se obter regras mais compreensíveis, haja vista que nestes casos as expressões de ativação também são funções lineares dos atributos envolvidos. Assim, embora a RN possa mapear funções não lineares, o processo de extração de regras se dá em um espaço em que os atributos foram transformados linearmente pela otimização dos pesos das conexões.

Um outro aspecto importante a ser observado diz respeito ao número de regras lógicas obtidas. Alguns algoritmos produzem centenas de regras lógicas, fornecendo uma descrição *opaca* dos dados, de modo que as regras obtidas são tão *compreensíveis*

quanto o próprio modelo de rede neural [20]. Nesse sentido, desenvolveu-se um algoritmo que procura *otimizar* o conjunto de regras por meio da *otimização* dos *grupos*, pois cada *grupo* corresponde a uma regra de classificação.

O problema de se otimizar *agrupamentos* é NP-completo [98] e foi enfrentado por meio de algoritmos genéticos. Inicialmente, optou-se por implementar um algoritmo de *agrupamento* baseado em operadores tradicionais sendo, por este motivo, denominado de Algoritmo Genético de Agrupamento Tradicional (AGAT), cujo estudo é objeto do Capítulo 3. Os resultados obtidos por meio do AGAT [123-126] motivaram o desenvolvimento de um outro algoritmo, projetado especificamente para resolver problemas de *agrupamento* que envolvem a separação de um conjunto de dados. Este algoritmo foi denominado de Algoritmo Genético de Agrupamento Específico (AGAE) e sua descrição pode ser encontrada no Capítulo 4.

5.1 Discussão dos resultados obtidos

A metodologia proposta foi avaliada em quatro bases de dados muito usadas para testar algoritmos de *data mining*. Nesse sentido, as regras de classificação obtidas pelo AGAE mostraram um desempenho satisfatório, haja vista que as taxas de classificação corretas médias são semelhantes às melhores descritas na literatura. DUCH et al. [20] descrevem os resultados obtidos por vários métodos de extração de regras de redes neurais, enquanto que MICHIE et al. [136] descrevem os resultados obtidos por outros métodos usados em *data mining*. A tabela 16 permite comparar os resultados descritos em [20,136] com os principais resultados obtidos nesse trabalho. Além disso, o método proposto foi também avaliado em uma aplicação prática de mineração de dados meteorológicos.

No concernente aos resultados obtidos, pode-se observar que o método de extração de regras proposto produziu bons resultados, principalmente quando se considera que é possível obter melhores arquiteturas de redes neurais para os problemas estudados. Além disso, os resultados obtidos na base de dados *Pima Indians Diabetes* sugerem que a metodologia desenvolvida é dependente da estrutura de conhecimento incorporada pela rede neural e, a exemplo da maioria dos métodos de extração de regras, não proporciona resultados satisfatórios quando a rede neural assume uma representação tipicamente distribuída [137]. Conquanto a representação distribuída seja

vantajosa em eficiência de memória e adaptabilidade, verificam-se desvantagens em relação à interpretação do conhecimento codificado no modelo de rede neural [13,38].

Tabela 16. Resumo dos melhores resultados obtidos pelo AGAT e pelo AGAE.

Base de Dados	AGAT Original	AGAT Ativações	AGAE Original	AGAE Ativações	Literatura
Iris Plants	93%*	97%*	94,67%*	95,33%*	95,7%-100%[20]
Australian Credit Approval	81%*	86,67%*	66,91%^{v} 63,31%^{t}	86,33%^{v} 76,26%^{t}	79,3%-86,9%[136]
Pima Indians Diabetes	58,6%*	61%*	68,18%^{v} 66,88%^{t}	66,88%^{v} 66,23%^{t}	71,9%-77,7%[20] 67,6%-77,7%[136]
Wisconsin Breast Cancer	NA	NA	94,89%^{v} 97,81%^{t}	97,81%^{v} 96,35%^{t}	92,7%-99%[20]
Meteorologia	NA	NA	72,18%^{v} 73,79%^{t}	72,18%^{v} 71,37%^{t}	NA

* Resultados obtidos sem usar validação cruzada – base completa (treinamento).
^{v} Resultados obtidos no conjunto de validação;
^{t} Resultados obtidos no conjunto de teste;
NA – Não Avaliado.

A comparação das regras obtidas no espaço original dos atributos com aquelas obtidas no espaço das ativações também fornece resultados interessantes. Para tal, são considerados somente os conjuntos de regras que modelam ambas as classes envolvidas, isto é, não serão considerados os conjuntos de regras que dispõem de uma classe *default* [130]. Nesse sentido, não serão levados em conta os exemplos referentes às bases de dados *Pima Indians Diabetes e Meteorologia*, haja vista que os melhores conjuntos de regras obtidos nestas bases de dados empregam uma regra *default.* A tabela 17 permite comparar os resultados obtidos pelo AGAE nas demais bases de dados estudadas, nas quais os melhores conjuntos de regras descrevem todas as classes envolvidas. É possível, então, observar a superioridade, considerando-se as taxas de classificação corretas médias, dos resultados alcançados no espaço das ativações.

A aplicação de um algoritmo de *agrupamento* no espaço das ativações proporcionou melhores resultados porque é mais simples extrair padrões em um espaço transformado pelas unidades ocultas, que são detectoras de características, assim chamadas porque elas tentam mapear classes que, no espaço original não são linearmente separáveis, para o espaço das ativações, em que as classes possuem uma chance maior de serem linearmente separáveis [5]. Sob o ponto de vista geométrico, as regras lógicas correspondem a uma divisão do espaço das características, que se dá por meio de hiperplanos perpendiculares aos eixos, em *áreas* com nomes simbólicos. Como se descreve em [20], se as classes no espaço das entradas das unidades ocultas podem

ser separadas por tais hiperplanos, então a descrição lógica dos dados é possível e meritória.

Tabela 17. Taxas de classificação corretas médias obtidas nos espaços originais dos atributos e nos espaços das ativações dos melhores modelos de redes neurais obtidos.

Base de Dados	AGAE – Original	AGAE – Ativações
Iris Plants	94,67%*	95,33%*
Australian Credit Approval	12,95%[v] 10,07%[t]	86,33%[v] 76,26%[t]
Wisconsin Breast Cancer	30,65%[v] 32,11%[t]	97,81%[v] 96,35%[t]

* Resultados obtidos sem usar validação cruzada – base completa (treinamento).
[v] Resultados obtidos no conjunto de validação;
[t] Resultados obtidos no conjunto de teste;

Cumpre observar, ainda, que os experimentos envolvendo a base de dados de meteorologia permitem mostrar um exemplo da *sinergia* que pode aparecer na utilização de redes neurais e de algoritmos genéticos de *agrupamento* em uma aplicação *prática* de mineração de dados. Neste contexto, observa-se que, embora o melhor conjunto de regras tenha sido obtido no espaço original dos atributos, o treinamento de redes neurais contribuiu de maneira significativa para a seleção dos melhores atributos, permitindo regras mais compreensíveis e com taxas de classificação corretas médias semelhantes.

As considerações expostas nessa seção mostraram algumas das vantagens e desvantagens encontradas na utilização da metodologia proposta para extrair regras de perceptrons de múltiplas camadas treinados em problemas de classificação. Neste contexto, inspiram-se algumas soluções que possam suprir as deficiências apresentadas pelo AGAE, e a próxima seção apresenta algumas sugestões para trabalhos futuros que possam aperfeiçoar a metodologia proposta.

5.2 Sugestões para trabalhos futuros

Como se pode observar, a metodologia desenvolvida para extrair regras de perceptrons de múltiplas camadas é muito dependente do algoritmo de *agrupamento* utilizado. Assim sendo, entende-se que um desenvolvimento importante diz respeito ao aperfeiçoamento do AGAE. Nesse sentido, afiguram-se alguns trabalhos interessantes:

a) Estudar e implementar outros tipos de função objetivo, permitindo que AGAE procure pela melhor solução segundo outros critérios de *agrupamento*. Dessa forma, parece que a implementação de um algoritmo que permita uma

otimização dos *agrupamentos*, empregando-se vários critérios diferentes concomitantemente, pode proporcionar resultados interessantes;

b) Avaliar o desempenho dos operadores de recombinação e mutação propostos. Nesse sentido, seria interessante aplicar esses operadores de maneira dinâmica, segundo seus desempenhos nas diversas gerações, fazendo-se variar suas taxas de utilização de acordo com a *adequabilidade* dos genótipos resultantes;
c) Implementar outras medidas de distância;
d) Desenvolver outros operadores e avaliar os parâmetros usados no algoritmo genético;

Além disso, cumpre avaliar a metodologia proposta em outros problemas de classificação que aparecem em *mineração de dados*. Nesse sentido, a investigação das regras extraídas de redes neurais *otimizadas*, isto é, com as melhores arquiteturas possíveis, se constitui numa pesquisa importante, bem como a avaliação do método proposto em perceptrons com mais de uma camada oculta.

O desenvolvimento de métodos de avaliação do conjunto de regras obtido, isto é, da complexidade da estrutura de conhecimento extraída do modelo de rede neural é uma tarefa interessante. Nesse sentido, a comparação da complexidade e da precisão das regras obtidas por diferentes métodos também se constitui num trabalho importante. Alguns trabalhos que tratam desse assunto podem ser encontrados em [131,132,139,140].

A implementação de sistemas formados por módulos que integrem simuladores de redes neurais e algoritmos de extração de regras se constitui numa tarefa interessante. Assim, a implementação de sistemas automatizados, englobando técnicas de visualização e preparação de dados, algoritmos de seleção de variáveis e de amostras, simuladores de redes neurais, validação de regras, etc., é uma abordagem promissora às aplicações de *data mining*. No tocante à seleção das amostras de treinamento, a metodologia desenvolvida em [138] parece ser muito interessante, pois permite aumento de eficiência computacional, na medida em que pode propiciar a determinação do número ideal de exemplos necessários ao treinamento de uma determinada rede neural.

A incorporação de conhecimento de domínio aos modelos de redes neurais pode proporcionar um meio de se aperfeiçoar e atualizar determinado conhecimento por meio de dados [31]. Nestes casos, a revisão do conhecimento depende da manutenção da representação local previamente estabelecida. Seguindo uma metodologia semelhante, a

utilização de regras *fuzzy* pode resultar em uma rede neural *fuzzy* [13], cujas unidades de entrada representam variáveis lingüísticas.

FU [31] observou empiricamente que utilização da função de ativação baseada no fator de certeza - *CF-Based Activation Function* - tende a uma manutenção maior da semântica relativamente à função *sigmóide*. Neste aspecto, a utilização destas funções de ativação pode favorecer a obtenção de representações locais. A implementação de um sistema integrado, formado por um simulador de redes neurais baseado na função de ativação *CF* e por algoritmos de extração de regras originaria um sistema interessante.

Finalmente, a aplicação do AGAE em conjuntos de dados com *exemplos ruidosos* também se constitui num trabalho interessante, pois permitiria avaliar o desempenho do AGAE na separação de exemplos *ruidosos*, que é uma aplicação muito útil às tarefas de mineração de dados.

CAPÍTULO 6
Referências Bibliográficas

[1] FAYYAD, U., SHAPIRO, G., SMYTH, P., UTHURUSAMY, R., "Preface". In: **Advances in Knowledge Discovery and Data Mining**, Fayyad, U.M., Piatetsky-Shapiro, G., Smyth, P., Uthurusamy, R., Editors, MIT Press, pp. xiii-xiv, 1996.

[2] FAYYAD, U. M., SHAPIRO, G. P., SMYTH, P. "From Data Mining to Knowledge Discovery : An Overview". In: **Advances in Knowledge Discovery and Data Mining**, Fayyad, U.M., Piatetsky-Shapiro, G., Smyth, P., Uthurusamy, R., Editors, MIT Press, pp. 1-37, 1996.

[3] BIGUS, J. P., **Data Mining with Neural Networks** , 1ª ed. USA, McGraw-Hill, 1996.

[4] CHEN, M., HAN, J., YU, P. S. "Data Mining: An Overview from a Database Perspective", **IEEE Transactions on Knowledge and Data Engineering**, vol. 8, n. 6, pp 866-883, Dezembro, 1996.

[5] HAYKIN, S., **Neural Networks - A Comprehensive Foundation**, 1ª ed., Macmillan College Publishing Company, New York, 1994.

[6] BRACHMAN, R., ANAND, T., "The Process of Knowledge Discovery in Databases". In: **Advances in Knowledge Discovery and Data Mining**, Fayyad, U.M., Piatetsky-Shapiro, G., Smyth, P., Uthurusamy, R., Editors, MIT Press, pp.37-57 , 1996.

[7] UTHURUSAMY, R. "From Data Mining to Knowledge Discovery". In: **Advances in Knowledge Discovery and Data Mining**, Fayyad, U.M., Piatetsky-Shapiro, G., Smyth, P., Uthurusamy, R., Editors, MIT Press, pp.561-573 , 1996.

[8] KLÖSGEN, W., ZYTKOW, J. M., "Knowledge Discovery in Databases Terminology". In: **Advances in Knowledge Discovery and Data Mining**, Fayyad, U.M., Piatetsky-Shapiro, G., Smyth, P., Uthurusamy, R., Editors, MIT Press, pp.573-593 , 1996.

[9] MICHIE, D., SPIEGELHALTER, D. J., TAYLOR, C. C. “Introduction”. In: **Machine Learning, Neural and Statistical Classification**, Ellis Horwood Series in Artificial intelligence, pp 1-5, 1994.

[10] WIEDERHOLD, G., “On the Barriers and Future of Knowledge Discovery ”. In: **Advances in Knowledge Discovery and Data Mining**, Fayyad, U.M., Piatetsky-Shapiro, G., Smyth, P., Uthurusamy, R., Editors, MIT Press, pp. vii-xi , 1996.

[11] MANNILA, H., "Methods and Problems in Data Mining", **Proceedings of International Conference on Database Theory**, Greece, Janeiro, 1997.

[12] CARVALHO, L.A.V., **Datamining**, 1ª edição, Editora Érica LTDA, São Paulo, 2001.

[13] FU, L. **Neural Networks in Computer Intelligence**, 1ª ed., USA, McGraw-Hill Inc., 1994.

[14] RUMELHART, D. E., HINTON, G. E., WILLIAMS, R. J., “Learning Internal Representations by Error Propagation”. In: **Parallel Distributed Processing**, 10 ed., pp 318-363, MIT Press, Bradford Books, USA, 1986.

[15] SMITH, M., **Neural Networks for Statistical Modeling,** 1ª ed. USA, International Thomson Computer Press, 1996.

[16] CRAVEN, M. W., SHAVLIK, J. W., "Using Neural Networks for Data Mining", **Future Generation Computer Systems**, accepted to appear, 1997.

[17] RUMELHART, D. E., ZIPSER, D. ”Feature Discovery by Competitive Learning”. In: **Parallel Distributed Processing**, 10 ed., pp 151-194, MIT Press, Bradford Books, USA, 1986.

[18] ANDERBERG, M. R., **Cluster Analysis for Applications,** 1ª ed., USA, Academic Press, Inc., 1973.

[19] HARTIGAN, J. A., **Clustering Algorithms,** 1ª ed. USA, John Wiley & Sons Inc., 1975.

[20] DUCH, W., ADAMCZAK, R., GRABCZEWSKI, K., “A new methodology of extraction, optimization and application of crisp and fuzzy logical rules”, **IEEE Transactions on Neural Networks,** vol.11, n.2, Março de 2000.

[21] KEEDWELL, E., NARAYANAN, A., SAVIC, D., “Creating Rules from Trained Neural Networks Using Genetic Algorithms”, **International Journal of Computers, Systems and Signals,** vol. 1, n.1, 2000.

[22] HRUSCHKA, E.R., **Um Estudo sobre a Extração de Regras de Redes Neurais em Aplicações de Data Mining**, Tese de M.Sc., COPPE/UFRJ, Rio de Janeiro, RJ, Brasil, Fevereiro de 1998.

[23] GALLANT, S. I. “Connectionist Expert Systems”, **Communications of the ACM,** v.31, n. 2, pp. 152-169, 1988.

[24] NARAZAKI, H., SHIGAKI, I., WATANABE, T. “A Method for Extracting Approximate Rules from Neural Network”, **Proceedings of the 4Th IEEE International Conference on Fuzzy Systems & 2nd International Fuzzy Engineering Symposium ,** Yokohama, Japan, March, 1995.

[25] BARON, R. **Knowledge Extraction from Neural Networks: A Survey**, In: Report n° 94-17, Laboratoire de l’Informatique du Parallélisme, Ecole Normale Supérieure de Lyon, 1994.

[26] GALLANT, S.E., **Neural Network Learning and Expert Systems,** MIT Press, Cambridge, Massachussets, 1994.

[27] McMILLAN, C., MOZER, M.C., SMOLENSKI, P., **“**Rule Induction through Integrated Symbolic and Subsymbolic Processing”, In: **Advances in Neural Processing Systems IV,** J.E. Moody, S.J. Hanson, R.P. Lippmann Eds., pp. 969-976, CA, Morgan Kaufmann Publishers, 1992.

[28] TOWELL, G. G., SHAVLIK, J., "Extracting Refined Rules from Knowlege-Based Neural Networks", **Machine Learning**, v.13, pág 71-101, 1993.

[29] OLIVEIRA, J. P., ULIANA, P. B., LIMA, W. C., "Algoritmos de Extração de Regras de Redes Neurais Artificiais", **Anais do III Congresso Brasileiro de Redes Neurais**, pág 173-177, Florianópolis, Julho de 1997.

[30] THRUN, S., **Extracting Provably Correct Rules from Artificial Neural Networks**, *Technical Report IAI-93-5,* Institut für Informatik III, Universität Bonn, 1993.

[31] FU, L., "Knowledge-Based Connectionism for Revising Domain Theories", **IEEE Transactions on Systems, Man. and Cybernetics**, v.23, n. 1, pag. 173-182, Feb.,1993.

[32] FU, L. " Rule Generation from Neural Networks ", **IEEE Transactions on Systems, Man., and Cybernetics**, v.24 , n.8 , pp.1114 - 1124 , Aug, 1994.

[33] KANE, R., MILGRAM, M. "Extraction of semantic rules from trained neural networks", **Proceedings of the ICNN**, San Francisco, CA, 1993.

[34] CRAVEN, M. W., SHAVLIK, J. W., "Using Sampling and Queries to Extract Rules from Trained Neural Networks", **Machine Learning: Proceedings of the Eleventh International Conference,** San Francisco, CA, 1994.

[35] SETHI, I.K.,YOO, J.H., "Symbolic approximation of feedforward neural networks", In: **Pattern Recognition in Practice IV,** E.S. Gelsema, L.N. Kanal Eds., pp. 313-324, North-Holland, 1994.

[36] ALEXANDER, J.A., MOZER, M.C., "Template-based algorithm for connectionist rule extraction", In: Tesauro, G., Touretzky, D., Leen, T. (editors), **Advances in Neural Information Processing Systems,** vol.7, MIT Press, 1995.

[37] ANDREWS, R., DIEDERICH, J., TICKLE, A.B., "A Survey and Critique of Techniques for Extracting Rules from Trained Artificial Neural Networks", **Knowledge-Based Systems,** n.8, pp. 373-389, 1995.

[38] NARAZAKI, H., WATANABE, T., YAMAMOTO, M.,"Reorganizing Knowledge in Neural Networks: An Explanatory Mechanism for Neural Networks in Data Classification Problems", **IEEE Transactions on Systems, Man. and Cybernetics,** v.26, n. 1, pág. 107-117, Feb., 1996.

[39] COX, E., **The Fuzzy Systems Handbook - A practitioner's guide to building, using, and maintaining fuzzy systems**, 1ª ed., USA, Academic Press Inc., 1994.

[40] BARDOSSY, D., DUCKSTEIN, L., **Fuzzy Rule Based Modelling with Applications to Geophysical, Biological and Engineering Systems**, CRC Press, 1995.

[41] OMLIN, C.W., GILES, C.L., “Rule Revision with Recurrent Neural Networks”, **IEEE Transactions of Knowledge and Data Engineering**, vol.8, n.1, Fevereiro, 1996.

[42] ANDREWS, R., GEVA, S., “Rules and Local Function Networks”, In: **Proceedings of the Rule Extraction from Trained Artificial Neural Networks Workshop**, pp. 1-15, University of Sussex, Brighton, UK, Abril, 1996.

[43] ANDREWS, R., GEVA, S., “Rule Refinement and Local Function Networks”, In: **Proceedings of the NIPS*96 Rule Extraction from Trained Artificial Neural Networks Workshop**, pp. 1-12, Snowmass, CO, Dezembro, 1996.

[44]CECHIN, A.L., EPPERLEIN, U., ROSENSTIEL, W., KOPPENHOEFER, B., “The Extraction of Sugeno Fuzzy Rules From Neural Networks”, In: **Proceedings of the Rule Extraction from Trained Artificial Neural Networks Workshop**, pp. 16-24, University of Sussex, Brighton, UK, Abril, 1996.

[45] DIEDERICH, J., HOGAN, J.M., GOLEA, M., MUTHAI, S., “Rule Initialisation by Neural Networks”, In: **Proceedings of the Rule Extraction from Trained Artificial Neural Networks Workshop**, pp. 25-33, University of Sussex, Brighton, UK, Abril, 1996.

[46] DECLOEDT, L., OSORIO, F., AMY, B., “RULE_OUT Method: A New Approach For Knowledge Explicitation From Trained ANN”, In: **Proceedings of the Rule Extraction from Trained Artificial Neural Networks Workshop**, pp. 34-42, University of Sussex, Brighton, UK, Abril, 1996.

[47] FRÄMLING, K., “Explaining Results of Neural Networks by Contextual Importance and Utility”, In: **Proceedings of the Rule Extraction from Trained Artificial Neural Networks Workshop**, pp. 43-50, University of Sussex, Brighton, UK, Abril, 1996.

[48] GOLEA, M., “On the Complexity of Rule-Extraction from Neural Networks and Network-Querying”, In: **Proceedings of the Rule Extraction from Trained Artificial Neural Networks Workshop**, pp. 51-59, University of Sussex, Brighton, UK, Abril, 1996.

[49] HOWES, P., CROOK, N., “Rule Extraction from Neural Networks”, In: **Proceedings of the Rule Extraction from Trained Artificial Neural Networks Workshop**, pp. 60-67, University of Sussex, Brighton, UK, Abril, 1996.

[50] JOHNSON, G., NEALON, L., LINDSAY, R.O., “Using relevance information in the acquisition of rules from a neural network”, In: **Proceedings of the Rule Extraction from Trained Artificial Neural Networks Workshop**, pp. 68-80, University of Sussex, Brighton, UK, Abril, 1996.

[51] SALOMON, R., “ Initialization by Rule Induction Prior to Learning”, In: **Proceedings of the Rule Extraction from Trained Artificial Neural Networks Workshop**, pp. 81-89, University of Sussex, Brighton, UK, Abril, 1996.

[52] TICKLE, A.B., ORLOWSKI, M., DIEDERICH, J., DEDEC: A Methodology For Extracting Rules From Trained Artificial Neural Networks”, In: **Proceedings of the Rule Extraction from Trained Artificial Neural Networks Workshop**, pp. 90-102, University of Sussex, Brighton, UK, Abril, 1996.

[53] TSUKIMOTO, H., MORITA, C., “An Algorithm for Extracting Propositions from Trained Neural Networks Using Multilinear Functions”, In: **Proceedings of the Rule Extraction from Trained Artificial Neural Networks Workshop**, pp. 103-114, University of Sussex, Brighton, UK, Abril, 1996.

[54] ULTSCH, A., KORUS, D., “Automatic Acquisition of Symbolic Knowledge from Subsymbolic Neural Networks”, In: **Proceedings of the Rule Extraction from Trained Artificial Neural Networks Workshop**, pp. 115-121, University of Sussex, Brighton, UK, Abril, 1996.

[55] BOLOGNA, G., “Rule Extraction from the IMLP Neural Network: a Comparative Study”, In: **Proceedings of the NIPS*96 Rule Extraction from Trained Artificial Neural Networks Workshop**, pp.13-19, Snowmass, CO, Dezembro, 1996.

[56] SETHI, I.K., YOO, J.H., “Multivalued Logic Representation of Neurons in Feedforward Networks”, **Engineering Intelligent Systems**, vol. 4, n. 4, pp. 243-253, 1996.

[57] FILER, R.J.H., SETHI, I.K., AUSTIN, J., “A Comparison between Two Rule Extraction Methods for Continuous Input Data”, In: **Proceedings of the NIPS*96 Rule Extraction from Trained Artificial Neural Networks Workshop**, pp.21-28, Snowmass, CO, Dezembro, 1996.

[58] FILER, R.J.H., AUSTIN, J., "A Neural Network Interrogation Tool for the Intensive Therapy Unit. In: **Proceedings of the NNESMED´96**, The University of Plymouth, 1996.

[59] HAYWARD, R., DIEDERICH, J., "Steps Towards Connectionist Explanation-Based Generalisation", In: **Proceedings of the NIPS*96 Rule Extraction from Trained Artificial Neural Networks Workshop**, pp.29-37, Snowmass, CO, Dezembro, 1996.

[60] KRISHNAN, R., "A Systematic Method for Decompositional Rule Extraction from Neural Networks", In: **Proceedings of the NIPS*96 Rule Extraction from Trained Artificial Neural Networks Workshop**, pp.38-45, Snowmass, CO, Dezembro, 1996.

[61] LeBLANC, C., LACHER, R.C., ADAIR, K., ELLING, J.W., HRUSKA, S.I., "Rule Refinement using Expert Networks", In: **Proceedings of the NIPS*96 Rule Extraction from Trained Artificial Neural Networks Workshop**, pp.46-53, Snowmass, CO, Dezembro, 1996.

[62] NEDJARI, T., "MITER: Mutual Information and Template for Extracting Rules", In: **Proceedings of the NIPS*96 Rule Extraction from Trained Artificial Neural Networks Workshop**, pp.54-61, Snowmass, CO, Dezembro, 1996.

[63] SAITO, K., NAKANO, R., "Law Discovery using Neural Networks", In: **Proceedings of the NIPS*96 Rule Extraction from Trained Artificial Neural Networks Workshop**, pp.62-69, Snowmass, CO, Dezembro, 1996.

[64] WILES, J., BOLLAND, S., "Beyond Finite State Machines: Steps towards representing and extracting context free languages from recurrent neural networks", In: **Proceedings of the NIPS*96 Rule Extraction from Trained Artificial Neural Networks Workshop**, pp.70-77, Snowmass, CO, Dezembro, 1996.

[65] GILES, C.L., LAWRENCE, S., TSOI, A.C., "Rule Inference for Financial Prediction using Recurrent Neural Networks", In: **Proceedings of the NIPS*96 Rule Extraction from Trained Artificial Neural Networks Workshop**, pp.78-84, Snowmass, CO, Dezembro, 1996.

[66] VISSER, U., TICKLE, A., HAYWARD, R., ANDREWS, R., "Rule-Extraction from trained neural networks: Different techniques for the determination of herbicides for the plant protection advisory system PRO_PLANT", In: **Proceedings of the Rule**

Extraction from Trained Artificial Neural Networks Workshop, pp. 122-130, University of Sussex, Brighton, UK, Abril, 1996.

[67] POP, E., HAYWARD, J., DIEDERICH, J., **RULENEG: extracting rules from a trained ANN by step-wise negation**, QUT NRC Technical report, Queensland University of Technology, Neurocomputing Research Center, Dezembro, 1994.

[68] ANDREWS, R., GEVA, S.,"Inserting and extracting knowledge from constrained error back-propagation networks", **Proceedings of the 6th Australian Conference on Neural Networks,** Sydney, NSW, 1995.

[69] HAYWARD, R., TICKLE, A., DIEDERICH, J., "Extracting rules for grammar recognition from Cascade-2 networks", In: **Connectionist, Statistical and Symbolic Approaches to Learning for Natural Language Processing**, Springer-Verlag, Berlin, Março, 1996.

[70] CRAVEN, M. W., SHAVLIK, J. W., "Extracting Tree-Structured Representations of Trained Networks", **Advances in Neural Information Processing Systems**, v.8, MIT Press, Cambridge, MA, 1996.

[71] BREIMAN, L., FRIEDMAN, J. H., OLSHEN, R. A., STONE, C. J., **Classification and Regression Trees**, 1a ed. Monterey, California, USA, Wadsworth & Brooks, 1984.

[72] SHAVLIK, J.W., "An Overview of Research at Wisconsin on Knowledge-Based Neural Networks", **Proceeedings of the International Conference on Neural Networks,** pp. 65-69, Washington, DC, 1996.

[73] HERMANN, C., THEIR, A., **Backpropagation for Neural DNF- and CNF-Networks**, Technical Report, FG Intellektik, TH Darmstadt, Germany, 1996.

[74] TAHA, I., GHOSH, J., "Characterization of the Wisconsin Breast Cancer Database Using a Hybrid Symbolic-Connectionist System", **Proceedings of ANNIE´96,** St. Louis, Novembro, 1996.

[75] TAHA, I., GHOSH, J., "Evaluation and Ordering of Rules Extracted from Feedforward Networks", **Proceedings of the IEEE International Conference on Neural Networks,** pp. 221-226, 1997.

[76] TAHA, I., GHOSH, J., **Symbolic Interpretation of Artificial Neural Networks**, Technical Report TR-97-01-106, The Computer and Vision Research Center, University of Texas, Austin, 1996.

[77] TAHA, I., GHOSH, J., "A Hybrid Intelligent Architecture and Its Application to Water Reservoir Control", **Journal of Smart Engineering Systems,** Outubro de 1996.

[78] LIU, H., SETIONO, R., "Chi2: Feature Selection and Discretization of Numeric Attributes", **Proceedings of the 7th IEEE International Conference on Tools with Artificial Intelligence**, pp. 388-391, 1995.

[79] LU, H., SETIONO, R., LIU, H. "Effective Data Mining Using Neural Networks", **IEEE Transactions on Knowledge and Data Engineering**, v. 8, n. 6, pp. 957-961, 1996.

[80] RUMELHART, D. E., HINTON, G. E., MCCLELLAND, J. L. "A General Framework for Parallel Distributed Processing". In: **Parallel Distributed Processing**, 10 ed., pp 45-77, MIT Press, Bradford Books, USA, 1986.

[81] HRUSCHKA, E.R. and EBECKEN, N.F.F. "Rule Extraction from Neural Networks in Data Mining Applications" In: **Data Mining**, N.F.F. Ebecken Editor, Computational Mechanics Publications, Southampton, UK, pp. 289-301, 1998.

[82] HRUSCHKA, E.R. and EBECKEN, N.F.F. "Rule Extraction from Neural Networks: Modified RX Algorithm", **Proceedings of the IEEE International Joint Conference on Neural Networks** (IJCNN'99), Washington DC, USA, July, 1999.

[83] HRUSCHKA, E.R. and EBECKEN, N.F.F. "Extração de Regras de Redes Neurais por meio do Algoritmo RX Modificado: Um Exemplo de Aplicação em Modelagem de Dados Meteorológicos", **Anais do IV Congresso Brasileiro de Redes Neurais**, ITA, São José dos Campos, Julho, 1999.

[84] SANTOS, R.T., NIEVOLA, J.C., FREITAS, A.A., LOPES, H.S., "Extração de Regras de Redes Neurais via Algoritmos Genéticos", **Anais do IV Congresso Brasileiro de Redes Neurais,** pp. 158-163, ITA, São José dos Campos, Julho, 1999.

[85] SETIONO, R., "Extracting rules from neural networks by pruning and hidden-unit splitting", **Neural Computation,** vol.9, n.1, pp. 205-225.

[86] DUCH, W., ADAMCZAK, R., GRABCZEWSKI, K., ISHIKAWA, M., UEDA, H., "Extraction of crisp logical rules using constrained backpropagation networks", **Proceedings of the European Symposium on Artificial Neural Networks,** Bruge, 1997.

[87] DAS, S., MOZER, M., "Dynamic on-line Clustering and state extraction: An approach to symbolic learning", **Neural Networks,** 11(1), pp. 53-64, 1998.

[88] BENÍTEZ, J.M., CASTRO, J.L., REQUENA, I., "Are Artificial Neural Networks Black Boxes?", **IEEE Transactions on Neural Networks**, vol. 8, n.5, Setembro, 1997.

[89] SETIONO, R., LIU, H., "NeuroLinear: From neural networks to oblique decision rules", **Neurocomputing**, vol. 16, n.3, pp. 225-235, Setembro, 1997.

[90] WEIJTERS, T., BOSH, A.V.D., HERIK, J.V.D, "Intelligible Neural Networks with BP_SOM", **Proceedings of the Ninth Dutch Conference on Artificial Intelligence,** pp. 27-36, 1997.

[91] DUCH, W., ADAMCZAK, R., GRABCZEWSKI, K., "Extraction of Logical rules from training data using backpropagation networks", **Neural Processing Letters**, n. 7, 1-9, 1998.

[92] BARROS, M.M., SOUZA G. JÚNIOR, G.M., LUDERMIR, T.B., "Extração de Regras em uma Rede Neural Booleana utilizando o modelo RAM", **Anais do V Simpósio Brasileiro de Redes Neurais,** pp. 121-125, Belo Horizonte, Dezembro, 1998.

[93] NOBRE, C.N., BRAGA, A.P., CARVALHO, A.C.P.L, LUDERMIR, T.B., BRAGA, J.L. "Extração de Conhecimento: uma comparação entre os métodos clássico e conexionista", **Anais do V Simpósio Brasileiro de Redes Neurais,** pp.126-131, Belo Horizonte, Dezembro, 1998.

[94] OSÓRIO, F.S., DECLOEDT, L., "Rule_Out: Um novo método de extração incremental de regras a partir de redes neurais construtivas do tipo KBANN", **Anais do V Simpósio Brasileiro de Redes Neurais,** pp.132-137, Belo Horizonte, Dezembro, 1998.

[95] LUDERMIR, T.B., "Extracting Rules from Feedforward Boolean Neural Networks", **Proceedings of the V Brazilian Symposium on Neural Networks**, pp. 61-65, Belo Horizonte, 1998.

[96] DUCH, W., ADAMCZAK, R., GRABCZEWSKI, K., ZAL, G. "Hybrid neural-global minimization method of logical rule extraction", **Journal of Advanced Computational Intelligence,** vol. 3, 1999.

[97] DUCH, W., ADAMCZAK, R., GRABCZEWSKI, K., "Optimization of Logical Rules Derived by Neural Procedures", **Proceedings of the International Joint Conference on Neural Networks,** Washington, DC, Junho, 1999.

[98] PARK, Y., SONG, M., "A Genetic Algorithm for Clustering Problems", **Proceedings of the Genetic Programming 1998 Conference,** pp. 568-575, University of Wisconsin in Madison, Julho, 1998.

[99] FALKENAUER, E., **Genetic Algorithms and Grouping Problems**, John Wiley and Sons Ltd., West Sussex, England, 1998.

[100] GOLDBERG, D.E., **Genetic Algorithms in Search, Optimization, and Machine Learning**, Addison Wesley Longman Inc., 1989.

[101] DAVIS, L., **Handbook of Genetic Algorithms,** 1ª ed., Boston, USA, International Thomson Computer Press, 1996.

[102] KOTHARI, R., PITTS, D., "On finding the number of clusters", **Pattern Recognition Letters,** n.20, pp. 405-416, Elsevier, 1999.

[103] ARABIE, P., HUBERT, L.J., DE SOETE, G., **Clustering and Classification**, World Scientific, Publishing Co. Pte. Ltd., Singapore, 1996.

[104] COLE, R.M., **Clustering With Genetic Algorithms**, Tese de M.Sc., Department of Computer Science, University of Western Australia, Janeiro,1998.

[105] KAUFMAN, L., ROUSSEEUW, P.J., **Finding Groups in Data – An Introduction to Cluster Analysis,** Wiley Series in Probability and Mathematical Statistics, John Wiley & Sons Inc., 1990.

[106] MILIGAN, G.W., "Clustering Validation: Results and Implications for Applied Analyses", In: ARABIE, P., HUBERT, L.J., DE SOETE, G. Editors, **Clustering and Classification**, pp. 341-373, World Scientific, Publishing Co. Pte. Ltd., Singapore, 1996.

[107] LIU, G.L., **Introduction to combinatorial mathematics**, McGraw Hill, 1968.

[108] ARABIE, P., HUBERT, L.J., "An Overview of Combinatorial Data Analysis", In: ARABIE, P., HUBERT, L.J., DE SOETE, G. Editors, **Clustering and Classification**, pp. 5-64, World Scientific, Publishing Co. Pte. Ltd., Singapore, 1996.

[109] FASULO, D., "An Analysis of Recent Work on Clustering Algorithms", http://www.cs.washington.edu/homes/dfasulo/clustering.ps, Abril, 1999.

[110] NAKAMURA, E., KEHTARNAVAZ, N., "Determining number of clusters and prototype locations via multi-scale clustering", **Pattern Recognition Letters,** n.19, pp. 1265-1283, Elsevier, 1998.

[111] BOCK, H.,"Probability Models and Hypotheses Testing in Partitioning Cluster Analysis", In: ARABIE, P., HUBERT, L.J., DE SOETE, G. Editors, **Clustering and Classification**, pp. 377-454, World Scientific, Publishing Co. Pte. Ltd., Singapore, 1996.

[112] MAITRA, R., "Clustering Massive Datasets", Department of Statistics, Carnegie Mellon University, Pittsburgh, PA, www.math.umbc.edu/~maitra/publications.html, in preparation, 1998.

[113] BOLEY, D., BORST, V., **Unsupervised Clustering: A Fast Scalable Method for Large Datasets**, Technical Report IIS-9811229, Department of Computer Science and Engineering, University of Minnesota.

[114] FRALEY, C., RAFTERY, A.E., **How Many Clusters? Which Clustering Method? Answers via Model-Based Cluster Analysis**, Technical Report No. 329, Department of Statistics, University of Washington, Seattle, USA, Fevereiro, 1998.

[115] CALINSKI, T., HARABASZ, J., "A dendrite method for cluster analysis", **Communications in statistics**, 3(1), pp. 1-27, 1974.

[116] DAVIES, D.L., BOULDIN, D.W., "A cluster separation measure", **IEEE Transactions on pattern analysis and machine intelligence,** PAMI-1(2), pp. 224-227, 1979.

[117] EVERITT, B.S., **Cluster Analysis,** Halsted Press, terceira edição, 1993.

[118] MILLIGAN, G.W., COOPER, M.C., "An examination of procedures for determining the number of clusters in a data set", **Psychometrika**, 50(2), pp. 159-179, 1985.

[119] SYMONS, M.J., “Clustering criteria and multivariate normal mixtures”, **Biometrics**, n.37, pp. 655-659, 1995.

[120] OKADA, A., “A Review of Cluster Analysis Research in Japan”, In: ARABIE, P., HUBERT, L.J., DE SOETE, G. Editors, **Clustering and Classification**, pp. 271-294, World Scientific, Publishing Co. Pte. Ltd., Singapore, 1996.

[121] TANOMARU, J. “Motivação, Fundamentos e Aplicações de Algoritmos Genéticos”. In: **II Congresso Brasileiro de Redes Neurais** - *Relatório de Conferência.*

[122] PARMEE, I.C., “Cluster-Oriented Genetic Algorithms (COGAS) for the Identification of High-Performance Regions of Design Spaces”, **Proceedings of the EvCA96 Conference**, Moscow, Junho, 1996.

[123] HRUSCHKA, E.R. and EBECKEN, N.F.F. "A Clustering Genetic Algorithm for Extracting Rules from Supervised Neural Network Models in Data Mining Tasks", **International Journal of Computers Systems and Signals, Special Issue: Knowledge Discovery from Structured and Unstructured Data**, AP Engelbrecht ed., v.1, n.1, pp. 17-29, December 2000.

[124] HRUSCHKA, E.R. and EBECKEN, N.F.F. "*Credit approval by a clustering genetic algorithm*", In: **Data Mining II**, N.F.F. Ebecken and C.A. Brebbia Editors, Computational Mechanics Publications, WIT Press, Southampton, UK, pp. 403-412, 2000.

[125] HRUSCHKA, E.R. and EBECKEN, N.F.F. "Applying a Clustering Genetic Algorithm for Extracting Rules from a Supervised Neural Network", **Proceedings of the IEEE–INNS-ENNS International Joint Conference on Neural Networks (IJCNN’2000)**, Como, Italy, July, 2000.

[126] HRUSCHKA, E.R. and EBECKEN, N.F.F. "Using a Clustering Genetic Algorithm for Rule Extraction from Artificial Neural Networks", **Proceedings of The First IEEE Symposium on Combinations of Evolutionary Computation and Neural Networks**, pp. 199-206, San Antonio, *TX, USA,* May, 2000.

[127] HAIR, J. F. J, ANDERSON, R. E., TATHAM, R. L., BLACK, W. C., **Multivariate Data Analysis**, Fifth Edition, Prentice Hall, USA, 1998.

[128] HENERY, R, J. “Classification”. In: **Machine Learning, Neural and Statistical Classification**, D. Michie, D.J.Spiegelhalter and C.C. Taylor editors, pp. 6-16, Ellis Horwood Series in Artificial Intelligence, Bookcraft, Midsomer Norton, 1994.

[129] MERZ, C.J., MURPHY, P.M., “UCI Repository of Machine Learning Databases”, [http://www.ics.uci.edu/~mlearn/MLRepository.html], Irvine, CA, University of California, Department of Computer Science, 1996.

[130] REITER, R. “Nonmonotonic Reasoning”, **Annual Reviews of Computer Science,** pp. 147-188, 1987.

[131] LIU, B., HSU, W., CHEN, S., MA, Y., “Analyzing the Subjective Interestingness of Association Rules”, **IEEE Transactions on Intelligent Systems**, pp. 47-55, September, 2000.

[132] LIU, B., HSU, W., MA, Y., “Prunning and Summarizing the Discovered Associations”, **Proceedings of the ACM-SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD´99),** San Diego, C.A., USA, August, 1999.

[133] AGRAWAL, R., MANNILA, H., SRIKANT, R., TOIVONEN, H., VERKAMO, A.I., “Fast Discovery of Association Rules”. In: **Advances in Knowledge Discovery and Data Mining**, Fayyad, U.M., Piatetsky-Shapiro, G., Smyth, P., Uthurusamy, R., Editors, MIT Press, pp. 307-328, 1996.

[134] HENERY, R, J. “Methods for Comparison”. In: **Machine Learning, Neural and Statistical Classification**, D. Michie, D.J.Spiegelhalter and C.C. Taylor editors, pp. 107-124, Ellis Horwood Series in Artificial Intelligence, Bookcraft, Midsomer Norton, 1994.

[135] Software NeuroShell 2, Documentação do Programa, Ward Systems Group, 1993.

[136] NAKHAEIZADEH, G., KRESSEL, U., KEH, S. et al. “Dataset Descriptions and Results”. In: **Machine Learning, Neural and Statistical Classification**, D. Michie, D.J.Spiegelhalter and C.C. Taylor editors, pp. 131-174, Ellis Horwood Series in Artificial Intelligence, Bookcraft, Midsomer Norton, 1994.

[137] HINTON, G.E., MCCLELLAND, J.L., RUMELHART, D.E., "Distributed Representations", In: **Parallel Distributed Processing**, vol. 1: Foundations, 10 ed., pp. 77-109, MIT Press, Bradford Books, USA, 1986.

[138] PLUTOWSKI, M., **Selecting Training Exemplars for Neural Network Learning,** Ph.D. Dissertation, University of California, San Diego, USA, 1994.

[139] GAINES, B.R., "Transforming Rules and Trees", In: **Advances in Knowledge Discovery and Data Mining**, Fayyad, U.M., Piatetsky-Shapiro, G., Smyth, P., Uthurusamy, R., Editors, MIT Press, pp. 205-228, 1996.

[140] NAGAI, W.A., **Avaliação do Conhecimento Extraído de Problemas de Regressão**, Dissertação de Mestrado, Instituto de Ciências Matemáticas e de Computação, USP, São Carlos, Agosto, 2000.