sid.inpe.br/mtc-m19/2012/02.29.20.00.49-TDI

ANÁLISE DE CENÁRIOS PARA CLASSIFICAÇÃO DE DADOS DE SENSORIAMENTO REMOTO USANDO OTIMIZAÇÃO MULTIOBJETIVO E HIERARQUIA DE CLASSES

Eliana Pantaleão

Tese de Doutorado do Curso de Pós-Graduação em Computação Aplicada, orientada pelos Drs. Luciano Vieira Dutra, e Sandra Sandri, aprovada em 29 de fevereiro de 2012.

URL do documento original:
<http://urlib.net/8JMKD3MGP7W/3BELSUF>

INPE
São José dos Campos
2012

PUBLICADO POR:

Instituto Nacional de Pesquisas Espaciais - INPE
Gabinete do Diretor (GB)
Serviço de Informação e Documentação (SID)
Caixa Postal 515 - CEP 12.245-970
São José dos Campos - SP - Brasil
Tel.:(012) 3208-6923/6921
Fax: (012) 3208-6919
E-mail: pubtc@sid.inpe.br

sid.inpe.br/mtc-m19/2012/02.29.20.00.49-TDI

ANÁLISE DE CENÁRIOS PARA CLASSIFICAÇÃO DE DADOS DE SENSORIAMENTO REMOTO USANDO OTIMIZAÇÃO MULTIOBJETIVO E HIERARQUIA DE CLASSES

Eliana Pantaleão

Tese de Doutorado do Curso de Pós-Graduação em Computação Aplicada, orientada pelos Drs. Luciano Vieira Dutra, e Sandra Sandri, aprovada em 29 de fevereiro de 2012.

URL do documento original:
<http://urlib.net/8JMKD3MGP7W/3BELSUF>

INPE
São José dos Campos
2012

Dados Internacionais de Catalogação na Publicação (CIP)

Pantaleão, Eliana.
P195a Análise de cenários para classificação de dados de sensoriamento remoto usando otimização multiobjetivo e hierarquia de classes / Eliana Pantaleão. – São José dos Campos : INPE, 2012.
xvii + 87 p. ; (sid.inpe.br/mtc-m19/2012/02.29.20.00.49-TDI)

Tese (Doutorado em Computação Aplicada) – Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2012.
Orientadores : Drs. Luciano Vieira Dutra, e Sandra Sandri.

1. reconhecimento de padrões. 2. sensoriamento remoto. 3. otimização multiobjetivo. . I.Título.

CDU 004.93´1

Aprovado (a) pela Banca Examinadora em cumprimento ao requisito exigido para obtenção do Título de Doutor(a) em

Computação Aplicada

Dra. Corina da Costa Freitas

Presidente / INPE / SJCampos - SP

Dra. Sandra Aparecida Sandri

Orientador(a) / INPE / SJCampos - SP

Dr. Luciano Vieira Dutra

Orientador(a) / INPE / SJCampos - SP

Dr. Rafael Duarte Coelho dos Santos

Membro da Banca / INPE / SJCampos - SP

Dra. Leila Maria Garcia Fonseca

Membro da Banca / INPE / SJCampos - SP

Dr. Nelson Delfino D'Ávila Mascarenhas

Convidado(a) / UFSCAR / São Carlos - SP

Dr. Raul Queiroz Feitosa

Convidado(a) / PUC- RIO / Rio de Janeiro - RJ

Dr. Alejandro Cesar Frery Orgambide

Convidado(a) / UFAL / Maceió - AL

Este trabalho foi aprovado por:

( ) maioria simples

(X) unanimidade

Aluno (a): Eliana Pantaleão

São José dos Campos, 29 de fevereiro de 2012

"If I had eight hours to chop down a tree, I would spend six sharpening my axe."

Abraham Lincoln

AGRADECIMENTOS

Agradeço a todos que contribuíram de alguma forma para que esse trabalho pudesse ser concluído, em especial:

À minha mãe, Vanda Brinck Pantaleão, pelo grande incentivo, apoio incondicional, suporte logístico, afetivo e espiritual, além da grande torcida.

Ao meu pai, Vicente Nepomuceno Pantaleão, que, apesar da já longa ausência, continua inspirando todos que o conheceram a acreditar que tudo é possível e que a maior alegria da vida é ajudar alguém.

Ao meu orientador, Dr. Luciano Vieira Dutra, pela sabedoria, paciência, conhecimentos transmitidos e oportunidades de realizar atividades diversas.

À minha orientadora, Dra. Sandra Sandri, pela contribuição pessoal e intelectual e pelo enorme apoio à conclusão dessa tese.

Ao professor Guaraci Erthal por conseguir tornar mais simples e interessantes conhecimentos áridos e complexos, pela paciência de ler o trabalho em suas versões mais rudimentares e pela atenção infinita.

À amiga Flávia de Toledo Martins, que não me deixou desistir nos momentos difíceis, e seus pais, Clemente e Cândida, pelo apoio.

Às amigas Aline Soterroni, Graziela Balda Scofield e Marinalva Soares pela amizade, incentivo e ajuda.

Aos amigos de Senzala, companheiros de suplícios e vitórias, em especial: Rogério Galanti Negri, Alessandra Gomes, Michelle Parreira, Frederico Volotão, Carlos Pires e Luciana Pereira.

Aos amigos Marize Simões e Sidnei Santana pelo bom humor e presteza em ajudar.

A Rogério Stacciarini pelo exemplo de amor à ciência e à vida.

À Dra. Corina Costa Freitas por visualizar o potencial do trabalho em uma fase crítica.

Aos professores do curso de Computação Aplicada por compartilhar seus conhecimentos e experiências em suas aulas.

Aos membros da banca pelas críticas construtivas, contribuições à forma final do documento e sugestões de projetos futuros.

À CAPES pelo auxílio financeiro.

Aos funcionários da CAP e da DPI, em especial às secretárias Cláudia, Carol, Luciana e Ellen, pelo apoio e competência.

Aos funcionários da biblioteca, em especial à Yolanda, pela forma atenciosa e eficiente de trabalhar.

Aos funcionários da segurança, pela paciência dispensada nos fins de semana.

Muito obrigada a todos!

RESUMO

Este trabalho propõe um método para análise automática de cenários para a classificação de imagens em sensoriamento remoto. Em uma tarefa típica de classificação de imagens, o analista decide inicialmente vários parâmetros a serem utilizados, como o tipo do classificador, o conjunto de classes de interesse e o conjunto de atributos que melhor caracteriza suas classes. Em um processo de tentativa e erro, muitas vezes é necessário que se encontre a melhor combinação desses parâmetros, aquela que dará um resultado mais satisfatório na classificação. O método proposto modela a busca por esses parâmetros como um problema de otimização multiobjetivo, em que os diversos anseios do usuário podem ser considerados. Além de melhor desempenho na classificação, o uso do sistema exige menos conhecimento específico do usuário sobre o comportamento dos atributos e das classes de interesse. O problema da otimização é resolvido por um algoritmo evolutivo que busca a fronteira de Pareto do problema. O resultado são as soluções de compromisso, ou seja, aquelas que atendem o conjunto de objetivos como um todo. Ao final do processo, uma heurística é utilizada para reduzir o conjunto de Pareto e sugerir opções ao usuário. Foram realizados dois estudos de caso e os resultados coincidiram com aqueles obtidos por estudos não automatizados.

CLASSIFICATION SCENARIO ANALYSIS FOR REMOTE SENSING DATA USING MULTIOBJECTIVE OPTIMIZATION AND CLASS HIERARCHY

ABSTRACT

This work proposes an automatic scenario analysis system for the image classification task in remote sensing applications. In a typical image classification task, the analyst decides beforehand which parameters to use, such as the number of classes, the type of the classifier and which image channels to use. If there is a need to modify the classes or data channels, it is necessary to start over, and to repeat this process until an appropriate combination of these parameters is achieved. This thesis proposes a scenario analysis tool for the task of image classification as a way of automating this search. Each scenario represents the parameters that will be used in a complete supervised classification task, including training and classification. The proposed method uses an evolutive multi-objective optimization algorithm to search for the ompromising solutions, regarding the user objectives. In addition to better classification results, the system helps users with less knowledge about the features and classes behaviors. In the end of the process, an heuristic method reduces the Pareto set of compromising solutions, giving fewer options to the user. The system was applied to two data sets and the results coincided with those obtained through non automated experiments.

LISTA DE FIGURAS

Pág.

1.1 Passos para a construção de um sistema de classificação. 2

2.1 Classificador Maxver. 6
2.2 Classes não linearmente separáveis. 8
2.3 Árvores de decisão (a) univariada e (b) oblíqua. 11
2.4 Solução ideal utilizada no método de Zeleny (1973). 20
2.5 Critério de menor perda. 21

3.1 Abordagem filtro para seleção de atributos. 26

4.1 Exemplo de árvore de classes. 29
4.2 Diagrama do reticulado com $A = \{1, 2, 3, 6\}$, $(x \leq y) \leftrightarrow (x \text{ divide } y)$. . . 30
4.3 Diagrama do reticulado com $A = \mathscr{P}(\{1, 2, 3\})$, $(x \leq y) \leftrightarrow (x \subseteq y)$. . . . 31
4.4 Dois caminhos possíveis entre $\emptyset$ e $\{1, 2, 3\}$. 31
4.5 Mesma árvore da Figura 4.1 após o uso da função renomeadora. 32
4.6 Reticulado R_T para $L_T = \{1, 2, 3, 4\}$. 34
4.7 Caminho sobre o reticulado, representando uma árvore. 35
4.8 Árvore equivalente ao caminho destacado na Figura 4.7. 35
4.9 Caminho sobre o reticulado usando um arco implícito. 36
4.10 Árvore equivalente ao caminho destacado na Figura 4.9. 36
4.11 Formas possíveis de árvores com quatro folhas. 36
4.12 Árvores possíveis a partir da mesma forma. 37
4.13 Reticulado podado R_O para a ontologia da Figura 4.8. 38
4.14 Exemplos de árvores que podem ser geradas a partir da ontologia da Figura 4.8. 39
4.15 Reticulado podado R_O para a ontologia da Figura 4.10. 39

5.1 Método proposto para as etapas da tarefa de classificação de imagens em sensoriamento remoto. 41
5.2 Elementos principais de um cenário. 43
5.3 Classificação a partir de um cenário. 43
5.4 Elementos principais de um cenário para classificação hierárquica. 44
5.5 Regiões de rejeição e aceitação da hipótese nula no teste Z. 50

6.1 Atributos e métodos da classe *Cenario*. 52
6.2 Atributos e métodos da classe *Classe*. 54
6.3 Tela inicial do Analisador de Cenários. 56

6.4 Opções do ENVI para gravar amostras em formato ASCII. 56
6.5 Trechos do arquivo de amostras gravado pelo ENVI. 57
6.6 Exemplo de arquivo de árvore de classes, que gera a árvore mostrada na Figura 6.3. 58
6.7 Tela de opções do Analisador de Cenários. 59
6.8 Tela que mostra a fronteira de Pareto encontrada pelo sistema. 60
6.9 Tela que permite a redução da fronteira de Pareto. 60
6.10 Vetor de bits que representa uma população. 62

7.1 Imagem sintética. 65
7.2 Três canais de dados da imagem sintética. 66
7.3 Espaço de atributos da imagem sintética. 66
7.4 Projeção do espaço de atributos da imagem sintética nos dois primeiros canais. 67
7.5 Hierarquias de classe utilizadas nos testes com a imagem sintética. 67
7.6 Funções objetivo de todos os cenários da imagem sintética. 68
7.7 Localização da área de estudo e composição 543 (RGB) da imagem TM. 72
7.8 Hierarquia de classes utilizada nos experimentos. 72
7.9 Imagens classificadas para (Ω,F): (a) (Ω_A,$\{B1, B2, B3, B4, B5, B7\}$); (b) ($\Omega_F$,$\{B1, B2, B3, B4, B5, B7\}$); (c) ($\Omega_G$,$\{B1, B4\}$) 77
7.10 Detalhes das classificações da Figura 7.9. 77
7.11 Imagens classificadas para (Ω,F): (a) (Ω_H,$\{B1, B2, B3, B4, B5, B7\}$); (b) ($\Omega_H$,$\{B1, B3, B4, B7\}$); (c) ($\Omega_G$,$\{B1\}$) 78
7.12 Detalhes das classificações da Figura 7.11. 78

LISTA DE TABELAS

Pág.

2.1 Matriz de Confusão 12
2.2 Categorias de algoritmos de seleção de atributos 15

7.1 Valores dos canais da imagem sintética. 65
7.2 Valores das funções objetivo para os cenários da imagem sintética assinalados na Figura 7.6. 69
7.3 Tamanho das amostras utilizadas. 73
7.4 Conjuntos de classes gerados pelo sistema. 74
7.5 Maior fronteira de Pareto encontrada. 74
7.6 Redução da fronteira de Pareto usando o tamanho do conjunto de classes como índice de ordenação. 76
7.7 Redução da fronteira de Pareto usando o tamanho do conjunto de atributos como índice de ordenação. 76

LISTA DE ABREVIATURAS E SIGLAS

AHI	–	Airborne Hyperspectral Imager
AG	–	Algoritmo Genético
AVIRIS	–	Airborne Visible/Infrared Imaging Spectometer
DT	–	Decision Tree
FDP	–	Função Densidade de Probabilidade
GEO	–	Generalized Extremal Optimization
IDL	–	Interactive Data Language
M-GEO	–	Multiobjective Generalized Extremal Optimization
Maxver	–	Máxima Verossimilhança
OBCT	–	Ordinary Binary Classification Tree
ODT	–	Oblique Decision Tree
REM	–	Radiação Eletromagnética
RBF	–	Radial Basis Function
SVM	–	Support Vector Machine
TM	–	Thematic Mapper

LISTA DE SÍMBOLOS

$f_1, f_2, \cdots$	–	funções objetivo
f_{Ic}	–	função objetivo escolhida para ser otimizada
P_k	–	probabilidade de alterar o bit k
$\lhd$	–	dominância
τ	–	parâmetro de aleatoriedade do algoritmo
L	–	tamanho da cadeia de bits (ou quantidade de espécies da população)
N_{FO}	–	número de funções objetivo
N_A	–	número de amostras
N_C	–	número de classes
Ω	–	conjunto de classes de um cenário
ω	–	uma classe particular de Ω
ag	–	acurácia global
κ	–	índice de concordância kappa
au_i	–	acurácia do usuário para a classe ω_i
κu_i	–	kappa condicional do usuário
agp	–	acurácia ponderada pela prioridade da classe
Γ	–	conjunto de atributos disponíveis
Ψ	–	conjunto de algoritmos de classificação
ψ	–	um algoritmo de classificação em particular
F	–	um conjunto particular de atributos
O	–	ontologia
T	–	árvore: $T = (N, E)$
N	–	conjunto de nós de uma árvore
E	–	conjunto de arestas de uma árvore
L_T	–	conjunto de folhas da árvore T
$r(T)$	–	raiz da árvore T
$h(T)$	–	altura da árvore T
$l(T)$	–	quantidade de níveis da árvore T
$\mathscr{P}(N)$	–	conjunto potência do conjunto N
$\emptyset$	–	conjunto vazio
$\|A\|$	–	número de elementos do conjunto A
$renom(T)$	–	função renomeadora de árvores
T'	–	árvore renomeada
$P(T')$	–	conjunto de partições que descrevem T'
$\alpha(a)$	–	conjunto de descendentes do nó a
π	–	uma partição de um conjunto
π_i	–	um elemento do conjunto $P(T')$ de partições de $L(T')$
$\prec$	–	relação de refinamento entre duas partições
R	–	reticulado
R_T	–	reticulado induzido por L_T e $\prec$

$\perp_R$	–	primeiro elemento do reticulado R
$\top_R$	–	último elemento do reticulado R
G_R	–	grafo associado ao reticulado R
N_R	–	conjunto de nós do grafo G_R
E_R	–	conjunto de arestas do grafo G_R
R_O	–	reticulado podado pela ontologia O
N_{RO}	–	conjunto de nós do grafo associado ao reticulado R_O
E_{RO}	–	conjunto de arestas do grafo associado ao reticulado R_O

SUMÁRIO

Pág.

1 INTRODUÇÃO 1

2 BASE TEÓRICA 5

2.1 Classificação de Dados de Sensoriamento Remoto 5

2.2 Avaliação da Acurácia da Classificação 11

2.3 Seleção de Atributos 13

2.4 Otimização Multiobjetivo 16

3 TRABALHOS RELACIONADOS 23

4 UM FORMALISMO PARA A CRIAÇÃO DE CENÁRIOS . . . 27

4.1 Conceitos básicos 28

4.1.1 Árvores 28

4.1.2 Reticulados e Partições 29

4.2 Árvores como sequências de partições 32

4.3 Árvores como caminhos em um reticulado 34

4.4 Compatibilidade entre árvores sobre um reticulado 37

5 MÉTODO PROPOSTO 41

5.1 Cenários de classificação 42

5.1.1 Cenário básico de classificação 42

5.1.2 Cenário básico para classificação sequencial 44

5.1.3 Outros cenários de classificação 45

5.2 Otimização Multiobjetivo 45

5.3 Redução da fronteira de Pareto 49

6 IMPLEMENTAÇÃO 51

6.1 IDL 51

6.2 Estruturas de dados 52

6.3 Interface 55

6.4 Algoritmos 61

7 EXPERIMENTOS E RESULTADOS 65

7.1 Conjunto controlado 65

7.1.1 Dados do problema 65
7.1.2 Solução 68
7.1.3 Resultados obtidos pelo Analisador de Cenários 70
7.2 Conjunto real 71

8 CONCLUSÕES E TRABALHOS FUTUROS 79
8.1 Conclusões 79
8.2 Trabalhos Futuros 80

REFERÊNCIAS BIBLIOGRÁFICAS 83

1 INTRODUÇÃO

Este trabalho propõe um sistema para análise automática de cenários de classificação de imagens em aplicações de sensoriamento remoto. A análise de cenários consiste na avaliação de um espectro de possibilidades com o objetivo de melhorar o processo de tomada de decisão, permitindo uma avaliação mais completa de resultados possíveis e suas consequências (HEIJDEN et al., 2002). Essa abordagem pressupõe que não há uma única melhor resposta para todas as decisões estratégicas e que é preciso analisar as vantagens e desvantagens de cada possível solução antes de se optar por uma delas (FAHEY; RANDALL, 1998).

Sensoriamento remoto é o processo de coletar dados sobre objetos ou feições da paisagem sem contato físico direto. Sua utilização consiste em analisar e interpretar as medidas de radiação eletromagnética (REM) que são refletidas ou emitidas por um alvo e observadas ou registradas por um observador ou instrumento que não está em contato com o alvo (MATHER, 2004). Em geral, o resultado dessas medidas são matrizes de valores digitais, às quais normalmente chamamos imagens.

A geração de mapas temáticos a partir da interpretação de dados existentes nessas imagens é um dos principais interesses do sensoriamento remoto. O processamento digital de imagens auxilia nesse objetivo por permitir ao cientista manipular e analisar os dados das imagens registradas por estes sensores de modo a revelar informações que podem não ser imediatamente perceptíveis na forma original.

A classificação de dados de sensoriamento remoto é uma tarefa frequente em diversas aplicações, como controle do desmatamento, previsão de safra na agricultura de precisão e a prevenção de desastres naturais. Nesse tipo de aplicação, um uso típico da classificação de imagens é produzir um mapa de uso ou cobertura do solo. O quão específico esse mapa pode ser, considerando os dados disponíveis, é uma questão frequentemente enfrentada pelos analistas humanos.

Em muitos casos, os dados disponíveis não são suficientes para discriminar todas as classes sugeridas no modelo criado pelo usuário, resultando em classificações com baixa acurácia. Quando se escolhe, por exemplo, discriminar apenas *agricultura* ao invés de *milho* e *cana-de-açúcar*, o processo de classificação normalmente necessita menos atributos, e, em geral, resulta em um mapa de maior acurácia, com menos confusão entre as classes. A identificação de classes mais específicas usualmente exige mais atributos e reduz a acurácia do mapa. Entretanto, se esta redução é pouco significativa, este pode ser considerado um mapa melhor para determinados

usuários, por ser mais específico. Além disso, ainda é preciso considerar os atributos disponíveis e o classificador que se pretende utilizar.

Muitas vezes, a quantidade de atributos disponíveis para a classificação de um conjunto de dados é bastante superior àquela necessária para realizar satisfatoriamente a classificação. Um erro bastante comum é tentar adicionar o máximo de atributos possível, com a suposição de que, quanto mais informações o classificador possuir, melhor poderá discriminar as classes. Sabe-se, entretanto, que isso nem sempre ocorre. Dependendo do número de amostras disponíveis para a construção do modelo pelo classificador, a adição de novos atributos, em geral, faz com que o espaço de busca fique mais esparso, dificultando a identificação do modelo subjacente dos dados.

Quando se tenta minimizar o número de atributos, ao mesmo tempo buscando a maior acurácia e classes mais específicas, estamos diante de um problema de otimização multiobjetivo. Esses objetivos estão em conflito pois otimizar um deles significa degradar pelo menos um dos outros, na maior parte das vezes.

O modelo usual do processo para gerar uma imagem classificada pressupõe conhecimento prévio do analista sobre as classes a serem identificadas (vide Figura 1.1). A cena terrestre é capturada através de um sensor, cuja saída são os dados brutos, ou originais. Existem diversas formas de gerar novos atributos a partir desses dados, no processo denominado extração de atributos.

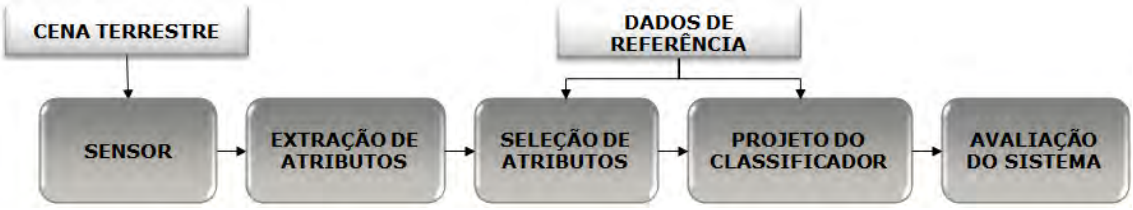

Figura 1.1 - Passos para a construção de um sistema de classificação.
Fonte: adaptado de Theodoridis e Koutroumbas (2006)

A seguir, os dados de referência são utilizados para selecionar os atributos mais importantes para discriminar as classes desejadas. Podem ser utilizados como dados de referência, amostras rotuladas obtidas através de verificação em campo ou análise visual por um especialista, ou ainda mapas obtidos a partir de dados ou procedimentos diferentes. O projeto do classificador varia conforme o seu tipo; ele

pode consistir em estimar um modelo estatístico, criar uma estrutura de árvore ou encontrar superfícies de separação entre as classes, entre outros. Por fim, é realizada a avaliação do processo. Na forma usual, o pesquisador analisa o resultado final, juntamente com métricas de acurácia, e avalia se é ou não satisfatório. Manualmente, usando sua experiência e intuição, modifica alguns dos parâmetros iniciais, como os atributos utilizados ou o tipo do classificador, em busca de um resultado melhor.

Neste trabalho propõe-se o conceito de *cenário de classificação*, que consiste nos elementos que devem ser escolhidos pelo analista antes de iniciar a tarefa de classificação, tais como o conjunto de atributos, o tipo do classificador e o conjunto das classes a serem discriminadas. O objetivo principal deste trabalho é propor um método para a análise automática dos diferentes cenários de classificação, considerando diferentes objetivos selecionados pelo usuário, conforme sua aplicação.

Na metodologia implementada, os cenários são produzidos a partir de uma proposta inicial do usuário, que fornece, além das amostras rotuladas, uma estrutura hierárquica das classes. É possível variar as classes discriminadas, o conjunto de atributos utilizado e o tipo do classificador. Os conjuntos de classes são gerados a partir da hierarquia fornecida, o que evita conjuntos semanticamente pouco significativos. Em cada cenário, os objetivos são avaliados simultaneamente, resultando nas melhores soluções de compromisso. Para isso, é usado um algoritmo evolutivo de otimização multiobjetivo, o M-GEO (GALSKI, 2006), cujo resultado é o conjunto de Pareto (DEB et al., 2002). Os objetivos considerados na análise automática são: a acurácia da classificação, a especificidade das classes discriminadas e a quantidade de atributos utilizados. O método também pode ser aplicado para outros objetivos, conforme necessidade da aplicação.

Também é proposta uma heurística para redução do conjunto de soluções com base em uma ordenação de prioridade dos objetivos, utilizando um teste estatístico para avaliar resultados semelhantes. A interação com o usuário ocorre após a obtenção da fronteira de Pareto (DEB et al., 2002), para que o mesmo possa avaliar os resultados obtidos.

O método foi aplicado a dois conjuntos de dados. O primeiro é um conjunto controlado, gerado a partir de uma imagem sintética com três canais e quatro classes bem delimitadas. O segundo consiste de uma imagem TM Landsat-5 e uma árvore construída com base em classes observadas em um trabalho de campo. Em todos os testes foi utilizado o índice de concordância kappa como medida de acurácia e foram permitidos nos cenários os classificadores Máxima Verossimilhança (Maxver),

Máquina de Vetores Suporte (SVM) e Árvore de Decisão Oblíqua (ODT). Os testes mostraram que o sistema foi capaz de encontrar satisfatoriamente a fonteira de Pareto quando seus parâmetros foram bem ajustados.

Esta tese está organizada do seguinte modo:

- Capítulo 2 - Base Teórica: faz uma revisão sucinta de alguns conceitos utilizados no desenvolvimento deste trabalho.
- Capítulo 3 - Trabalhos Relacionados: analisa alguns trabalhos com objetivos e metodologias semelhantes.
- Capítulo 4 - Um Formalismo para a Criação de Cenários: apresenta a formalização dos conceitos propostos nessa tese.
- Capítulo 5 - Método proposto: descreve o modelo do sistema proposto.
- Capítulo 6 - Implementação: explica como foi realizada a implementação do sistema.
- Capítulo 7 - Experimentos e Resultados: descreve testes realizados com o sistema e discute seus resultados.
- Capítulo 8 - Conclusões e Trabalhos Futuros: apresenta conclusões e sugere possíveis continuações para este trabalho.

2 BASE TEÓRICA

Este capítulo faz uma revisão rápida de alguns dos conceitos que serviram de base para a elaboração deste trabalho.

2.1 Classificação de Dados de Sensoriamento Remoto

Uma tarefa de reconhecimento de padrões procura associar categorias discretas ou classes a objetos representados por um conjunto de medidas, denominado vetor de atributos ou padrão. Na tarefa de classificação de imagens de sensoriamento remoto, uma abordagem possível é considerar como atributos a imagem em si, ou seja, cada padrão é representado pelos valores das medidas captadas pelos sensores para cada píxel da imagem. Atributos derivados, como parâmetros de textura, operações entre bandas e análise de componentes principais, também podem ser utilizados. Esse tipo de classificação é denominado classificação ponto a ponto. Outra possível abordagem é realizar antes uma segmentação da imagem, ou seja, dividi-la em regiões que apresentam uniformidade em relação a algum critério. Nesse caso, denominado classificação por região, será gerado um único vetor de atributos para o segmento, que receberá uma única classe atribuída pelo classificador.

A tarefa de associar classes aos vetores de atributos pode ser supervisionada, semi-supervisionada ou não supervisionada. No aprendizado supervisionado, o usuário fornece ao classificador um conjunto de treinamento com amostras rotuladas para cada classe, que podem ser utilizadas para estimar o modelo das classes ou as fronteiras entre elas. Após criado o modelo, é possível associar cada píxel ou região de rótulo desconhecido à classe apropriada. No caso em que os rótulos das classes estão associados a uma hierarquia de classes, é possível realizar a classificação hierárquica ou sequencial, em que, após o treinamento, o padrão desconhecido é classificado sucessivamente, em cada um dos níveis da hierarquia, a partir da raiz até atingir uma folha.

No aprendizado não supervisionado ou *clustering*, apenas as amostras não rotuladas são fornecidas ao classificador, que procura identificar agrupamentos destas amostras. Ao final do processo, o usuário pode dar nome aos grupos encontrados. O aprendizado semi-supervisionado usa tanto amostras não rotuladas quanto amostras rotuladas, procurando posicionar a fronteira de decisão em regiões de baixa densidade de pontos.

Existem diversos métodos para classificar uma imagem de forma supervisionada.

Neste trabalho são utilizados três classificadores: Maxver (Máxima Verossimilhança), SVM (*Support Vector Machine* ou Máquina de Vetores Suporte) e árvore de decisão (DT - *Decision Tree*).

O classificador Maxver com distribuição Gaussiana é um dos classificadores estatísticos mais utilizados na área de sensoriamento remoto. Isso se deve principalmente ao baixo custo computacional do método e à coerência no ajuste da distribuição estatística dos dados neste tipo de aplicação, em especial imagens óticas. O método utiliza uma estimação dos parâmetros da FDP (Função Densidade de Probabilidade) das N_A amostras rotuladas fornecidas. Para tanto, deve-se supor um tipo de distribuição, sendo a mais utilizada a distribuição Gaussiana. A FDP representa a probabilidade da ocorrência do padrão $\vec{x}$ em uma das N_C classes ω_i $(i = 1, \ldots, N_C)$ e é simbolizada por $p(\vec{x} \mid \omega_i)$ ou ainda $p(\vec{x} \mid \omega_i, \vec{\theta_i})$ para enfatizar a dependência no vetor $\vec{\theta}$ de parâmetros da distribuição suposta, que é estimado para cada classe ω_i. O classificador associa então a cada padrão $\vec{x}$ a classe com maior verossimilhança.

A Figura 2.1 ilustra esse método para um atributo (2.1a) e para dois atributos (2.1b). Para cada conjunto de pontos de mesmo rótulo, são estimados os parâmetros da função densidade de probabilidade. Os limites entre as classes são dados pela interseção entre as funções de densidade estimadas. No caso unidimensional, o limite é representado pelo ponto x_0, que divide o eixo do atributo x em duas regiões, R_1 e R_2. No caso bidimensional, a figura mostra os limites como linhas divisórias no plano dos atributos. Neste último caso, as elipses mostram os pontos compreendidos dentro do limite de um desvio padrão na distribuição estimada.

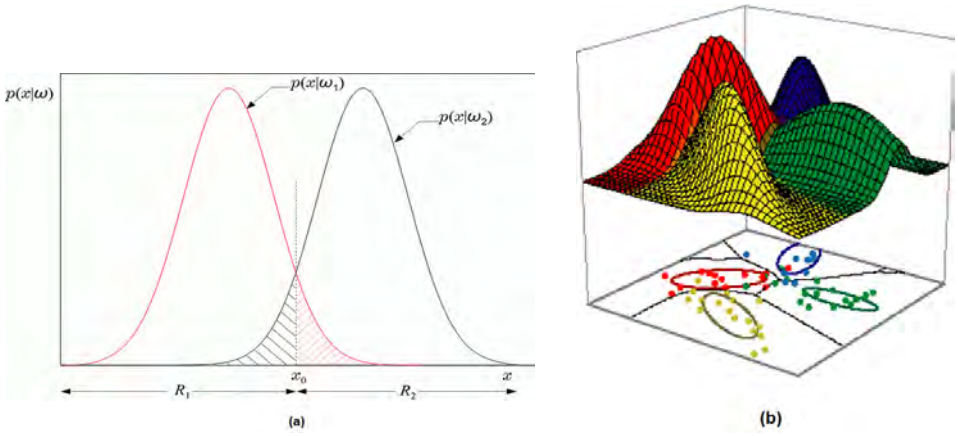

Figura 2.1 - Classificador Maxver.
Fonte: Duda et al. (2001)

Um dos problemas associados a este método é que ele exige uma grande quantidade de amostras para produzir bons resultados. O aumento da dimensionalidade agrava este problema, pois acarreta o aumento na quantidade de parâmetros a serem estimados. Além disso, embora a suposição da distribuição Gaussiana seja verdadeira para muitos dados de sensoriamento remoto, em alguns casos isso pode não ocorrer, como por exemplo, em imagens de radar. O método supõe que as amostras das classes de interesse são independentes e possuem a mesma probabilidade a priori. As amostras utilizadas na estimação dos parâmetros também devem ser independentes.

Uma Máquina de Vetores Suporte, ou SVM, é um método de aprendizagem universal cuja superfície de decisão é parametrizada por um conjunto de vetores suporte, e por um conjunto de pesos. Um classificador SVM procura encontrar a superfície de separação ótima $g(\vec{x})$ entre duas classes, de forma a deixar a maior margem livre entre as classes. Os vetores suporte são padrões, selecionados entre as amostras de cada classe, que definirão a superfície de decisão do classificador. A separação ótima entre os vetores suporte é equivalente à separação ótima entre as classes no conjunto completo de dados de treinamento (VAPNIK et al., 1996).

Na forma linear, a superfície de separação ótima é um hiperplano $g(\vec{x}) = \vec{w}^T\vec{x} + w_0$, cujos parâmetros $\vec{w}$ e w_0 podem ser encontrados pela maximização da função Lagrangeana (Equação 2.1) em que N_A é a quantidade de padrões de treinamento, y_i representa o rótulo da classe (-1 ou 1) e α_i são os multiplicadores de Lagrange, sendo $i = 1, \cdots, N_A$. Na prática, utiliza-se sua forma dual (Equação 2.2).

$$L_p = \frac{1}{2}\vec{w}^T\vec{w} - \sum_{i=1}^{N_A} \alpha_i(y_i(\vec{w}^T\vec{x_i} + w_0) - 1) \tag{2.1}$$

$$L_D = \sum_{i=1}^{N_A} \alpha_i - \frac{1}{2}\sum_{i=1}^{N_A}\sum_{j=1}^{N_A} \alpha_i\alpha_j y_i y_j \vec{x_i}^T\vec{x_j} \tag{2.2}$$

De acordo com as condições Karush-Kuhn-Tucker, os multiplicadores de Lagrange indicam a sensibilidade do valor ótimo de L_p à restrição correspondente. Consequentemente, seu valor será zero para qualquer vetor de atributos que não faça parte do conjunto de vetores suporte, indicando que a posição exata desses vetores não interfere no posicionamento do hiperplano separador ótimo (THEODORIDIS; KOUTROUMBAS, 2006).

Como ressalta Webb (2002), os vetores de treinamento são utilizados aos pares, na forma de produtos escalares, e o custo computacional não depende da dimensionalidade do espaço de atributos. Embora o hiperplano ótimo seja único, não há garantias que o mesmo ocorra com os multiplicadores de Lagrange.

Segundo Theodoridis e Koutroumbas (2006), em muitos casos reais não há um limite linear separando as classes, logo, não faz sentido procurar um hiperplano separador ótimo. A transformação dos dados em um espaço de maior dimensionalidade, no qual as classes são linearmente separáveis pode levar à especialização (*overfitting*) e, portanto, à perda de capacidade de generalização. Uma solução é incluir um valor de penalidade em caso de classificação incorreta, C. Neste caso diz-se que o hiperplano separador possui uma folga, pois permite-se que vetores de treinamento de uma classe ocorram no espaço correspondente à outra classe, desde que estejam a uma distância máxima, denominada margem. Quanto maior o valor de C, menos valores com classificação incorreta são permitidos, e o caso linearmente separável corresponde a $C \to \infty$.

Os padrões correspondentes aos multiplicadores de Lagrange menores que C localizam-se exatamente à distância de $\|w\|$ do hiperplano. A Figura 2.2 mostra duas classes não linearmente separáveis e o hiperplano separador com folga encontrado. A linha contínua representa o hiperplano separador e as linhas tracejadas delimitam a margem.

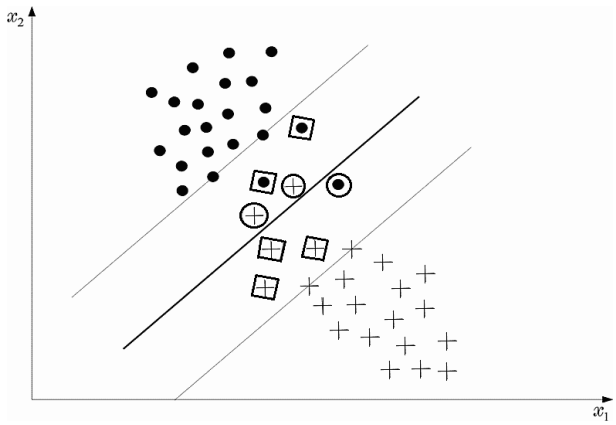

Figura 2.2 - Classes não linearmente separáveis.
Fonte: Theodoridis e Koutroumbas (2006)

De acordo com Webb (2002), quando é utilizado um hiperplano com folga, os vetores

de treinamento podem estar em uma destas situações:

- vetores que caem fora da margem (espaço entre a linha contínua e as tracejadas na Figura 2.2) e são corretamente classificados;
- vetores que são corretamente classificados, porém caem dentro da margem separadora (marcados com quadrados na Figura 2.2);
- vetores que são incorretamente classificados (marcados na Figura 2.2 com círculos).

O caso não linear também pode ser abordado através do uso de mapeamentos em um espaço de atributos de maior dimensionalidade, no qual as classes são linearmente separáveis. Em geral, são utilizadas, para este fim, funções simétricas denominadas funções de kernel (THEODORIDIS; KOUTROUMBAS, 2006). Essas funções são utilizadas para substituir o produto escalar presente no caso linear. Exemplos típicos de tais funções são: funções polinomiais, funções Gaussianas ou de base radial (RBF - *Radial Basis Function*) e sigmóides.

As máquinas de vetores suporte foram projetadas para separar apenas duas classes. Para um número maior de classes, é necessário utilizar uma estratégia de integração das superfícies geradas. Duas estratégias bastante simples são conhecidas como estratégia um-contra-todos (*one-against-all*) e estratégia um-contra-um (*one-against-one*). Na estratégia um-contra-todos, para discriminar N_C classes, são construídos N_C classificadores binários. Cada um dos classificadores é treinado para diferenciar uma classe de todas as demais.

Já na estratégia um-contra-um, são construídos $\frac{N_C(N_C-1)}{2}$ classificadores para diferenciar as N_C classes, duas a duas. Um padrão $\vec{x}$ é classificado através da execução de todos os classificadores e uma votação, por maioria absoluta, decide sua classe final. Isto pode levar a ambiguidades e indecisões em relação à classificação, em alguns casos. Outras formas de lidar com o problema multiclasse estão descritas em (THEODORIDIS; KOUTROUMBAS, 2006), tais como o uso de funções discriminantes, um procedimento para correção de erros, o critério de Fisher e o erro quadrático mínimo.

A vantagem na utilização do SVM é que o método não faz qualquer suposição inicial sobre a distribuição dos dados. Além disso, não é necessário um número tão grande de dados de treinamento para obter seu modelo. A desvantagem é o alto custo

computacional da solução do problema de otimização para encontrar a superfície ótima de separação. Outros problemas são a escolha apropriada da função de *kernel* e o ajuste do valor para a penalidade C.

Uma árvore de decisão para classificação de padrões utiliza a estratégia de particionar sucessivamente o espaço de busca em espaços mais homogêneos em relação às classes que se deseja discriminar, formando a estrutura de árvore. As árvores de decisão mais comuns são conhecidas como OBCT (*ordinary binary classification tree*) e dividem o espaço de busca em hiper-retângulos. São formadas por três elementos distintos:

- nodos internos, que representam os atributos;
- arcos, provenientes dos nodos internos e que recebem os valores possíveis para os atributos;
- nodos folha, que representam as diferentes classes de um conjunto de treinamento.

Já as árvores oblíquas (ODT - *oblique decision tree*), particionam o espaço de atributos com hiperplanos que são oblíquos aos eixos dos atributos. Seus nodos internos correspondem a uma comparação do padrão a ser classificado em relação a um desses hiperplanos. Ou seja, nas árvores oblíquas, diversos atributos são considerados simultaneamente em um único ponto de decisão, ao contrário das univariadas, em que é considerado um atributo de cada vez. A Figura 2.3 mostra um exemplo de árvore binária univariada (2.3a) e outro de árvore oblíqua (2.3b).

O modelo construído pelo classificador consiste em uma estrutura da árvore, utilizada posteriormente para classificar os padrões desconhecidos. Uma sequência de decisões a partir da raiz leva o padrão $\vec{x}$ até a classe ω_i.

Assim como o SVM, este método também não parte de suposições sobre a distribuição estatística dos dados, apresentando resultados satisfatórios na classificação de dados com distribuições não gaussianas ou multimodais. Outra vantagem é a possibilidade de utilizar atributos categóricos. Uma desvantagem é a necessidade da introdução de um critério para a poda da árvore que, caso contrário, poderá se tornar muito grande e especializada nos exemplos de treinamento. No caso extremo, cada nó folha contém apenas uma amostra.

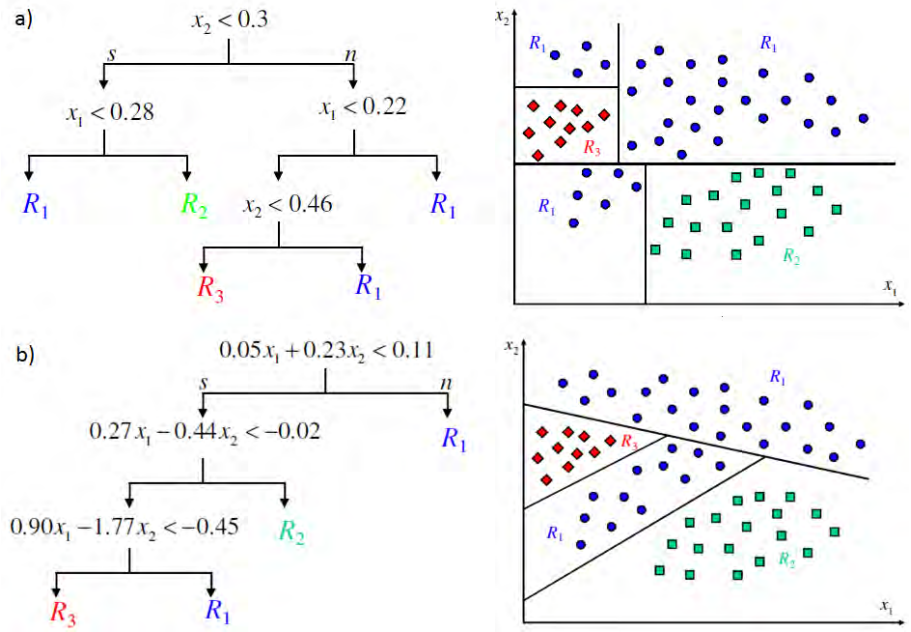

Figura 2.3 - Árvores de decisão (a) univariada e (b) oblíqua.
Fonte: Medeiros et al. (2011)

2.2 Avaliação da Acurácia da Classificação

A geração de mapas temáticos beneficia-se enormemente com o uso de sensoriamento remoto, especialmente no caso dos mapas que mostram a cobertura ou uso do solo, pois a classificação de imagens fornece uma representação semelhante a um mapa, que é espacialmente contínua e bastante consistente (FOODY, 2002). Além disso, em geral essas imagens estão disponíveis em diversas escalas espaciais e temporais. Entretanto, o usuário desses mapas deve estar ciente de que o processo de classificação automática não é perfeito. Ele precisa conhecer a qualidade do produto que está utilizando. Portanto, determinar a acurácia de mapas de cobertura do solo é de grande importância para usuários e pesquisadores do sensoriamento remoto.

Existem diversas formas de medir a acurácia da classificação, e diversos critérios para escolher qual medida se deve usar em determinada situação. Várias medidas foram desenvolvidas e testadas para diversos conjuntos de dados. A maioria dos métodos quantitativos utiliza como ponto de partida a *matriz de confusão* (STORY; CONGALTON, 1986) gerada a partir da comparação entre os dados classificados e os dados de referência, ou seja, um conjunto de amostras rotuladas reservado especialmente para este fim. A estrutura da matriz de confusão é mostrada na Tabela 2.1. Seus valores podem representar a quantidade exata de pontos ou sua porcentagem,

ou seja, sua divisão pelo total de pontos.

Tabela 2.1 - Matriz de Confusão

	Referência				
Classificação	1	2	$\cdots$	N_C	Total
1	p_{11}	p_{12}	$\cdots$	p_{1N_C}	p_{1+}
2	p_{21}	p_{22}	$\cdots$	p_{2N_C}	p_{2+}
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
N_C	p_{N_C1}	p_{N_C2}	$\cdots$	$p_{N_CN_C}$	p_{N_C+}
Total	p_{+1}	p_{+2}	$\cdots$	p_{+N_C}	

A Tabela 2.1 considera a existência de N_C classes. Nessa matriz, p_{ij} é a proporção de píxeis que foram associados com a classe ω_i pelo classificador, mas correspondem na realidade (de acordo com os dados de referência) à classe ω_j. O valor p_{+i} é a soma dos valores da coluna i, ou seja, a porcentagem total de pontos de referência pertencentes à classe ω_i. Do mesmo modo, p_{i+} é a soma dos valores da linha i, representando a porcentagem de pontos que foram atribuídos à classe ω_i pelo classificador.

Uma das medidas de acurácia mais simples é a acurácia global (ag), mostrada na Equação 2.3. A acurácia global é a razão entre a soma total de pontos corretamente classificados (diagonal principal da matriz de confusão) e o total de pontos considerados. No caso em que o valor da matriz já representa uma porcentagem, basta somar os valores da diagonal principal.

$$ag = \sum_{i=1}^{N_C} p_{ii} \tag{2.3}$$

O coeficiente kappa (κ), apresentado por Cohen (1960), é uma medida largamente utilizada em aplicações de sensoriamento remoto, como Lee et al. (2001) e Lim et al. (2007). Esta medida é bastante popular porque elimina da contagem de acertos o acerto por acaso, ou seja, a possibilidade de que o classificador efetuou a classificação correta por motivo aleatório (ver Equação 2.4). No entanto, Foody (1992) afirma que este coeficiente superestima o acerto por acaso, dessa forma subestimando a medida da acurácia. A sua popularidade está também associada ao fato de possuir uma distribuição conhecida, o que possibilita realizar testes de hipóteses.

$$\kappa = \frac{ag - \sum_{i=1}^{N_C} p_{i+}p_{+i}}{1 - \sum_{i=1}^{N_C} p_{i+}p_{+i}} \tag{2.4}$$

Também podem ser usadas medidas individuais por classe. A acurácia do usuário (STORY; CONGALTON, 1986) é mostrada na Equação 2.5 e o kappa condicional do usuário (ROSENFIELD; FITZPATRICK-LINS, 1986), na Equação 2.6. Ambas as medidas são calculadas para cada classe, ou seja, $i \in \{1, 2, \cdots, N_C\}$.

$$au_i = \frac{p_{ii}}{p_{i+}} \tag{2.5}$$

$$\kappa u_i = \frac{au_i - p_{+i}}{1 - p_{+i}} \tag{2.6}$$

Liu et al. (2007) compararam diversas medidas de acurácia através da avaliação da correlação entre elas. Os autores descreveram as principais características e relações entre as medidas, que foram divididas em dois grupos: global e por classe. Com base nessa metodologia, Pantaleão e Scofield (2009) compararam seis medidas por classe e dez medidas globais e verificaram que elas quase sempre concordam na ordenação dos resultados de classificação. A correlação encontrada entre as medidas globais foi acima de $0,89$ em todos os casos. Esse resultado sugere que, para um caso de comparação entre classificadores, a medida a ser usada não é muito relevante, desde que haja coerência em sua utilização, ou seja, que sejam comparados valores de uma mesma medida.

2.3 Seleção de Atributos

Em alguns casos, o número de atributos disponíveis é bastante superior à quantidade necessária para realizar a classificação. Por exemplo, o satélite hiperespectral Hyperion (USGS, 2008) e alguns sistemas aerotransportados como AHI (HIGP, 2001) e AVIRIS (NASA, 2007) possuem mais de duzentos canais de dados, consistindo de bandas bastante estreitas. Além disso, novos atributos podem ser derivados a partir dos originais, como índices de vegetação, razão entre bandas, coeficientes de textura e rotações no espaço de atributos, como componentes principais. Porém, nem sempre mais atributos significa mais informação. Por exemplo, os atributos podem estar altamente correlacionados, podem ser irrelevantes para discriminar as classes de interesse e podem estar em número superior ao permitido pela quantidade de

amostras.

Quando dois atributos estão altamente correlacionados, ainda que, individualmente, forneçam grande quantidade de informação, seu uso conjunto não aumenta significativamente a capacidade de discriminação entre as classes. Um atributo é considerado irrelevante quando não é útil para discriminar entre as classes de interesse. Em muitos casos, o aumento do número de atributos implica no aumento da quantidade de parâmetros a estimar, tais como os pesos sinápticos em uma rede neural ou pesos em um classificador linear (THEODORIDIS; KOUTROUMBAS, 2006). Adicionar atributos irrelevantes pode comprometer a eficiência sem nenhum ganho de acurácia. De acordo com Dash e Liu (1997), reduzir o número de atributos irrelevantes e redundantes pode reduzir drasticamente o tempo de execução de um algoritmo de aprendizado e ainda levar a um maior poder de generalização.

Quando a quantidade de amostras é pequena, usar um grande conjunto de atributos pode levar à superespecialização, fazendo o classificador bastante preciso ao identificar as amostras de treinamento, mas com baixa acurácia nos vetores desconhecidos. Portanto, além de aumentar o tempo gasto na tarefa de classificação, os atributos adicionais podem deteriorar o resultado da classificação. Esse problema é denominado *maldição da dimensionalidade*. De acordo com Jain et al. (2000), o desempenho de um classificador está associado com a relação entre o tamanho das amostras, o número de atributos e a complexidade do classificador. Theodoridis e Koutroumbas (2006) observam que, dependendo do classificador escolhido, uma grande quantidade de amostras pode ser necessária para um desempenho aceitável.

Considerando-se um número fixo de amostras, a adição de novos atributos pode degradar o desempenho do classificador, aumentando a taxa de erro. Isso ocorre principalmente com classificadores paramétricos que estimam os parâmetros desconhecidos a partir das amostras, e tomam estes parâmetros como sendo os parâmetros reais nas densidades condicionais das classes. Para um tamanho fixo de amostras, à medida em que o número de atributos aumenta, acompanhado pelo aumento correspondente no número de parâmetros desconhecidos, a confiabilidade das estimativas dos parâmetros diminui (JAIN et al., 2000). Além disso, Theodoridis e Koutroumbas (2006) notam que, no projeto do classificador, também se deseja estimar a probabilidade de erro do classificador projetado, e esta estimativa é melhor com o aumento da razão entre o tamanho do conjunto de amostras e a dimensionalidade.

A forma que se usa para reduzir a quantidade de atributos também é importante. Quando atributos com baixo poder de discriminação entre as classes são selecionados,

o classificador não funcionará satisfatoriamente. Por outro lado, se atributos com alto poder de discriminação são selecionados, o classificador obtido terá um desempenho superior, com menor complexidade de projeto. Identificar os atributos relevantes também pode, em alguns casos, economizar tempo e dinheiro com futuras medições.

Como notado por Jain et al. (2000), o ideal seria escolher (ou gerar) "atributos que façam com que padrões pertencentes a diferentes categorias ocupem regiões compactas e disjuntas no espaço de atributos". Essa afirmação sugere que a eficiência de um conjunto de atributos para discriminar um determinado conjunto de classes pode ser avaliada pela medida da separação entre padrões de diferentes classes.

Um algoritmo para seleção de atributos deve escolher automaticamente um subconjunto do conjunto inicial de todos os atributos disponíveis, com o menor tamanho possível. Para isso, uma alternativa seria simplesmente testar todos os subconjuntos possíveis e escolher o que tem maior potencial para discriminar as classes, por exemplo, o que possui a melhor medida de separabilidade entre as classes. Para evitar o procedimento computacionalmente caro da busca exaustiva, outros métodos baseados em heurísticas ou busca aleatória foram desenvolvidos. Há diversas formas de classificá-los, algumas delas são mostradas na Tabela 2.2 e explicadas a seguir.

Tabela 2.2 - Categorias de algoritmos de seleção de atributos

característica	**alternativas**	**categoria**
avaliação dos atributos	um de cada vez	seleção escalar
	um subconjunto por vez	seleção vetorial
interação com o classificador	nenhuma	filtro
	usa acurácia como objetivo	*wrapper*
	seleção é parte do aprendizado	embutido
cobertura do espaço de busca	busca em todo o espaço	completo
	busca em uma direção	heurístico
	busca em direções aleatórias	estocástico
conjunto selecionado	melhor solução possível	ótimo
	boa solução	sub-ótimo
uso de amostras rotuladas	usa amostras rotuladas	supervisionado
	não usa amostras rotuladas	não supervisionado

Algoritmos de seleção escalar avaliam a capacidade de discriminação de cada atributo individualmente, selecionando os melhores a partir de uma ordenação entre eles. Essa abordagem possui baixo custo computacional, mas o resultado não é sempre satisfatório. O conjunto com os melhores atributos nem sempre é o melhor conjunto de atributos. A avaliação do conjunto como um todo em geral encontra melhores resultados, pois evita a inclusão de atributos correlacionados (DASH; LIU, 1997).

Métodos de filtro realizam a seleção utilizando propriedades intrínsecas dos dados, sem vínculo com nenhuma aplicação. Possuem baixo custo computacional, mas não costumam ser ótimos por não considerarem o modelo utilizado pelo classificador. São exemplos desse método o uso de medidas de separabilidade entre as classes no espaço de atributos, como as distâncias de Bhattacharyya e Jeffreys-Matusita (THEODORIDIS; KOUTROUMBAS, 2006). Métodos de *wrapper* tentam minimizar o erro de generalização através do uso da acurácia do classificador como função de avaliação. Esse objetivo é o ideal, mas a busca é computacionalmente cara. Além disso, é dependente do classificador, de modo que o resultado de uma seleção pode ter um desempenho ruim se usado em diferentes classificadores. Nos métodos embutidos, a seleção faz parte do processo de aprendizado, como em uma árvore de decisão.

O uso de um algoritmo genético (AG) é uma opção bastante comum para resolver o problema da seleção de atributos. A natureza aleatória da busca oferece uma boa cobertura do espaço viável sem utilizar a busca exaustiva. Por exemplo, Kavzoglu e Mather (2002) usam AG para determinar as bandas de entrada de uma rede neural artificial. Em Santos (2007), são extraídos diversos atributos de forma, tais como densidade e circularidade, totalizando 46 atributos. Um AG seleciona o melhor conjunto com 1, 2, 3 e 4 atributos. O resultado foi avaliado pela comparação com a busca exaustiva e o AG se mostrou robusto e eficiente.

2.4 Otimização Multiobjetivo

Um problema de otimização pode ser considerado multiobjetivo quando há pelo menos dois objetivos conflitantes a considerar em um processo de tomada de decisão e não existe uma única solução que seja melhor para todos os objetivos. No problema multiobjetivo mostrado na formulação (2.7), a meta é encontrar o melhor vetor $\vec{x}$ de N_V variáveis, para minimizar N_{FO} funções objetivo. O espaço de busca é limitado por um limite inferior (${x_i}^L$) e superior (${x_i}^U$) para cada uma das N_{VP} variáveis de projeto x_i, além de N_{RD} restrições de desigualdade e N_{RI} restrições de igualdade.

$$
\begin{aligned}
&\textbf{Minimizar} \\
&\quad f_m(\vec{x}), && m \in \{1, 2, \ldots, N_{FO}\} \\
&\textbf{Sujeito a} \\
&\quad g_j(\vec{x}) \geq 0, && j \in \{1, 2, \ldots, N_{RD}\} \\
&\quad h_k(\vec{x}) = 0, && k \in \{1, 2, \ldots, N_{RI}\} \\
&\quad {x_i}^L x_i^L \leq x_i \leq {x_i}^U, && i \in \{1, 2, \ldots, N_{VP}\}
\end{aligned}
\tag{2.7}
$$

Existem várias alternativas para resolver esse tipo de problema. A abordagem de agregação ou escalarização consiste em combinar os objetivos em uma única função, por exemplo, utilizando pesos para transformar o problema em mono-objetivo. A função de agregação é usualmente linear. A definição dos pesos é fornecida pelo usuário do sistema e reflete a importância atribuída a cada objetivo. Sua determinação é o ponto fraco desse tipo de abordagem.

Na abordagem lexicográfica, os objetivos possuem uma ordem de importância, mas não um valor atribuído a essa importância. No processo de busca, os objetivos mais importantes são priorizados, e os outros são considerados apenas em caso de empate nos anteriores. Em geral, a comparação utiliza uma tolerância, considerando empate quando os valores não possuem diferença significativa.

A abordagem do conjunto de Pareto, utilizada neste trabalho, procura encontrar um conjunto não vazio de soluções não dominadas $\vec{x}_s$. Diz-se que uma solução $\vec{x}_{s1}$ domina outra solução $\vec{x}_{s2}$, relação que pode ser denotada por $\vec{x}_{s1} \triangleleft \vec{x}_{s2}$, se e somente se ambas as condições (2.8) e (2.9) são verdadeiras.

$$\forall m, 1 \leq m \leq N_{FO}, \quad f_m(\vec{x}_{s1}) \leq f_m(\vec{x}_{s2}) \tag{2.8}$$

$$\exists m, 1 \leq m \leq N_{FO}, \quad f_m(\vec{x}_{s1}) < f_m(\vec{x}_{s2}) \tag{2.9}$$

A condição (2.8) estabelece que não há uma função objetivo em que $\vec{x}_{s1}$ seja pior que $\vec{x}_{s2}$ e a condição (2.9), que $\vec{x}_{s1}$ é estritamente melhor que $\vec{x}_{s2}$ em pelo menos um objetivo. Se houver pelo menos um par $(m1, m2)$ para o qual $f_{m1}(\vec{x}_{s1}) < f_{m1}(\vec{x}_{s2})$ e $f_{m2}(\vec{x}_{s1}) > f_{m2}(\vec{x}_{s2})$, então nenhuma das soluções domina a outra, pois não é possível dizer que uma solução é melhor que a outra para ambos os objetivos. Como o problema é de minimização, $\vec{x}_{s1}$ é uma solução melhor para a função objetivo f_{m1}, enquanto $\vec{x}_{s2}$ é melhor para f_{m2}. O conjunto de Pareto é o conjunto de soluções em que nenhuma delas domina qualquer das outras (DEB, 2001) e pode ser formalmente definido pela expressão (2.10).

$$P^* = \{\vec{x}_1 | \nexists\, \vec{x}_2 \colon \vec{x}_2 \triangleleft \vec{x}_1)\} \tag{2.10}$$

No conjunto de Pareto, os elementos são soluções, ou seja, valores para as variáveis do problema de minimização. O conjunto dos valores da função objetivo para o conjunto de Pareto é denominado fronteira ou frente de Pareto e é dado pela expressão (2.11),

em que $\vec{f} = (f_1, f_2, \ldots, f_{NFO})$ é o vetor de funções objetivo.

$$PF^* = \{\vec{f}(\vec{x}) \mid \vec{x} \in P^*\} \tag{2.11}$$

Cada solução não dominada, ou solução de compromisso, favorece um ou outro objetivo. De posse do conjunto de soluções e da fronteira de Pareto, o usuário pode decidir qual solução adotar.

Neste trabalho, a implementação da solução da otimização multiobjetivo foi baseada no algoritmo M-GEO (GALSKI, 2006), uma versão multiobjetivo do algoritmo de Otimização Extrema Generalizada (GEO - *Generalized Extremal Optimization*) (SOUSA, 2002). O GEO é um algoritmo estocástico evolutivo, baseado na teoria da criticalidade auto-organizada, em que as espécies são representadas por uma cadeia de bits que codifica as variáveis do problema de otimização. Cada bit é forçado a evoluir com uma probabilidade proporcional à sua adaptabilidade. É um algoritmo de uso geral, que pode ser aplicado a problemas com espaço de busca complexo, disjunto, com vários mínimos locais e não linearidades na função objetivo e nas restrições.

Galski (2006) desenvolveu algumas variações para o algoritmo GEO, como a versão paralelizada GEOPAR-1. Sua versão multiobjetivo, o M-GEO (ver Algoritmo 1), gera o conjunto e a fronteira de Pareto para o problema de otimização. O GEO original pode ser obtido a partir do M-GEO fazendo o número de funções objetivo igual a um.

O algoritmo M-GEO utiliza uma estratégia de torneio: a cada iteração qualquer uma das funções objetivo pode ser utilizada como a função de atribuição de adaptação. A escolha é aleatória, com distribuição uniforme, e essa aleatoriedade permite ao M-GEO acessar a totalidade da fronteira de Pareto (GALSKI, 2006).

O M-GEO possui apenas um parâmetro livre, τ (passo 12), que controla o nível de determinismo da busca. Quando $\tau = 0$, a busca é completamente aleatória, ou seja, pode explorar o espaço de busca em qualquer direção. Quando $\tau \to \infty$, é uma busca determinística, já que o melhor valor encontrado na iteração é adotado e a busca segue nessa direção. Para cada problema específico há um valor de τ que maximiza a eficiência da busca (GALSKI, 2006).

O M-GEO deve ser executado diversas vezes de forma independente, ou seja, com

Algoritmo 1: Algoritmo M-GEO - adaptado de Galski (2006)

```
1: for i ← 1 to número de execuções independentes do
   2: inicialize aleatoriamente uma cadeia com L bits que codificam N variáveis
      de projeto;
   3: while não alcançou critério de parada do
      4: for i ← 1 to L do
         5: mude o valor do bit i, gerando uma nova cadeia x⃗_i;
         6: calcule o valor de todas as funções objetivo
         f⃗(x⃗_i) = [f_1(x⃗_i), ··· , f_M(x⃗_i)];
         7: teste e salve o conjunto e a fronteira de Pareto;
      end for
      8: escolha aleatoriamente uma função objetivo como f_Ic
      9: foreach string x⃗_i do
         10: atribua um valor de adaptação proporcional ao ganho ou perda
             que f_Ic(x⃗_i) tem se o bit muda, comparado ao melhor valor
             de f_Ic encontrado até o momento;
         11: ordene os bits de acordo com seu valor de adaptação, sendo 1
             a posição do pior valor;
         12: mude o valor de um bit da população com probabilidade P_k ∝ k^−τ,
             k ∈ {1, 2, ··· , L}, k é a posição do bit na ordenação;
      end foreach
   end while
end for
13:return conjunto e fronteira de Pareto;
```

inicializações aleatórias das variáveis de projeto. Isso garante a exploração mais ampla do espaço de busca, fazendo com que o algoritmo encontre uma extensão maior da fronteira de Pareto.

Algumas aplicações já desenvolvidas para o GEO e suas variações incluem o projeto térmico dos radiadores da Plataforma Multimissão do INPE (GALSKI et al., 2004), a obtenção da configuração de uma constelação de satélites de sensoriamento remoto (GALSKI et al., 2005) e o controle de atitude de satélites (MAINENTI-LOPES et al., 2011). O trabalho desenvolvido nessa tese contribui para a avaliação da eficácia deste novo algoritmo através de sua aplicação em uma área de domínio bastante diferente das já utilizadas.

Após a obtenção da fronteira de Pareto, uma opção é deixar para o usuário a escolha da solução que deve ser adotada para o problema de otimização. Entretanto, existem métodos para auxiliar nesta tarefa. Zeleny (1973) propõe uma metodologia que utiliza a solução mínima "ideal" para o problema. A solução ideal é uma solução

que, caso fosse viável, dominaria todas as soluções do conjunto de Pareto encontrado, como mostra a Figura 2.4. Nessa figura, a região de soluções viáveis está representada na cor cinza. Os pontos da fronteira de Pareto estão marcados com pontos sobre o limite da região viável e não são dominados por nenhuma solução no interior da região. A solução ótima "ideal" é o ponto que possui o menor valor possível observado, individualmente, para cada uma das funções f_1 e f_2. A técnica proposta é algébrica e consiste em encontrar, entre as soluções viáveis pertencentes à fronteira de Pareto, a que possui menor distância euclidiana até a solução ideal, no espaço de objetivos.

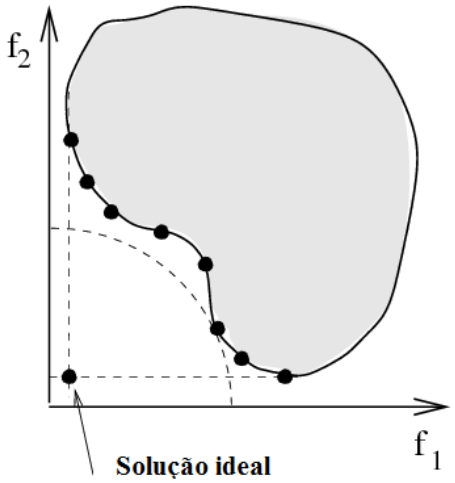

Figura 2.4 - Solução ideal utilizada no método de Zeleny (1973).
Fonte: Adaptado de Spolaôr (2010)

A abordagem de Rocco (2002) é considerar como a solução ótima do problema multiobjetivo aquela que acarreta menor perda para os objetivos envolvidos. Segundo o autor, esta é a solução localizada no ponto central da figura que tem como vértices as soluções ótimas para cada objetivo. O exemplo para três objetivos está ilustrado na Figura 2.5, em que o ponto B é a melhor solução indicada pelo método, e é o baricentro do triângulo formado pelos vértices $S1$, $S2$ e $S3$, respectivamente a solução que otimiza cada um dos objetivos 1, 2 e 3 isoladamente.

Um problema comumente associado a ambas as soluções de Zeleny (1973) e Rocco (2002) ocorre quando as funções objetivo representam grandezas de natureza bastante diferente. Por exemplo, se uma das funções objetivo representa uma quantidade, assumindo valores inteiros entre 1 e 10, enquanto outra das funções assume

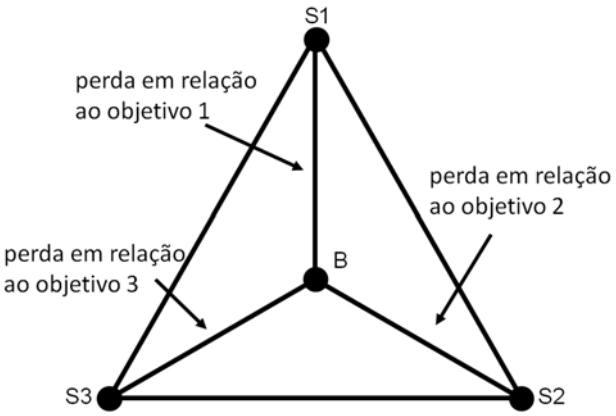

Figura 2.5 - Critério de menor perda.
Fonte: Rocco (2002)

valores reais entre 0 e 1. É o que se chama "misturar maçãs e laranjas". Um tratamento especial nos dados é necessário antes de se usar uma abordagem que avalia os dois tipos de valores em conjunto.

Os mecanismos propostos por Zeleny (1973) e Rocco (2002) não atribuem qualquer ordem de importância aos objetivos considerados. A semântica dos objetivos não é considerada, nem é feita qualquer interação com o usuário.

3 TRABALHOS RELACIONADOS

Neste capítulo são apresentados trabalhos que versam sobre temas relacionados ao desenvolvimento dessa tese. Vários estudos têm sido realizados na tentativa de automatizar a tarefa de escolher alguns dos elementos do cenário de classificação proposto nessa tese (Capítulo 4). Entretanto, nenhum dos trabalhos encontrados considera todos os elementos utilizados nessa tese, nem propõe uma representação formal para o problema.

Alguns trabalhos já foram desenvolvidos com o uso de hierarquia de classes. O objetivo do trabalho de Chen et al. (2009) é encontrar uma solução de compromisso entre o grau de especificidade das classes e a acurácia da classificação. Os autores partem do princípio, também utilizado nesta tese, de que quando os rótulos de classe estão organizados em uma estrutura hierárquica, a acurácia tende a ser maior nos níveis superiores da árvore. Quando as classes se tornam mais específicas, sua predição é mais difícil e a acurácia diminui. Chen et al. (2009) propõem um algoritmo de classificação para dados com rótulos de classe organizados hierarquicamente.

O classificador proposto é baseado em árvore de decisão. Durante o processo de construção da árvore, os rótulos dos dados em cada nó podem ter diferentes distribuições na árvore de classes. No momento da criação de cada nó interno, são considerados os dois critérios simultaneamente: a especificidade das classes utilizadas e a acurácia da classificação. Uma nova medida de entropia, denominada entropia hierárquica, é proposta para lidar com a situação de rótulos estruturados. A medida utiliza pesos que variam com o nível da classe na hierarquia e que são calculados conforme a Equação 3.2.

$$w_1 = 0 \tag{3.1}$$

$$w_i = (h - i + 1) \times \frac{2}{h(h-1)}, \quad i > 1 \tag{3.2}$$

O valor h da Equação 3.2 representa a altura total da hierarquia de rótulos (número de níveis). A expressão pode ser aplicada para todos os níveis i da árvore com $i > 1$, ou seja, será aplicada somente para árvores com $h > 1$. Por definição, o peso da raiz, w_1, é igual a zero. O critério de divisão da árvore é o ganho de informação hierárquica (*hierarchical information gain*), medida calculada com base na entropia hierárquica proposta.

Huang e Wang (2006) postulam que a seleção do melhor conjunto de atributos deve ocorrer ao mesmo tempo que a determinação dos parâmetros do classificador SVM. Sua proposta é o uso de um algoritmo genético para selecionar os atributos e otimizar os parâmetros simultaneamente. O processo de seleção de atributos é do tipo *wrapper* e usa a SVM com *kernel* radial (THEODORIDIS; KOUTROUMBAS, 2006). Portanto, dois parâmetros precisam ser otimizados: o parâmetro de custo e o parâmetro γ do *kernel* radial.

O cromossomo é representado por uma cadeia binária, que codifica o conjunto de atributos e os dois parâmetros do classificador. Na máscara para os atributos, 1 representa a presença e 0 a ausência do atributo e o tamanho da cadeia é o número de atributos disponíveis. O número de bits usado para representar os parâmetros dependerá da precisão desejada. A função objetivo (Expressão 3.3) combina as três variáveis, incluindo os dois objetivos na mesma expressão através da adição de pesos. Na notação utilizada pelos autores, tem-se:

$fitness$	é a medida de adaptação do cromossomo
W_A	é o peso da acurácia da classificação
$SVM_{accuracy}$	é a acurácia da classificação
W_F	é o peso para o número de atributos
C_i	é o custo do atributo i
F_i	é a posição i na máscara de atributos.

$$fitness = W_A \times SVM_{accuracy} + W_F \times \left(\sum_{i=1}^{m} C_i \times F_i \right)^{-1} \tag{3.3}$$

Os resultados obtidos pelos autores foram satisfatórios e trouxeram evidências de que os atributos selecionados influenciam na definição dos parâmetros adequados do classificador e vice-versa.

Ao combinar os objetivos de reduzir o conjunto de atributos e maximizar a acurácia em uma única função objetivo, o sistema de Huang e Wang (2006) precisa fazer o uso de pesos, que devem ser ajustados para melhor resultado. Os pesos W_A e W_F refletem o quanto o usuário considera importante cada um desses objetivos. Entretanto, essa decisão será tomada antecipadamente, sem a oportunidade de analisar o quanto um deles degrada o outro. Nessa tese, emprega-se o conjunto de Pareto, que permite que essa decisão seja tomada após mais informações estarem disponíveis. O sistema proposto por Huang e Wang (2006) não permite o ajuste das classes utilizadas, além de ser específico para um único tipo de classificador, o SVM, e apenas com o *kernel*

radial. O sistema proposto nessa tese analisa alguns conjuntos possíveis de classe e permite a utilização de qualquer tipo de classificador supervisionado.

A aplicação do trabalho de Ekbal e Saha (2012) é o reconhecimento de entidades nomeadas, no campo de processamento de linguagem natural. O objetivo é identificar um nome de entidade e classificá-lo em uma categoria como "nome de lugar", "nome de organização" ou "nome de pessoa".

Os autores utilizam uma abordagem alternativa para a clássica ideia de procurar um conjunto de atributos que seja apropriado para um determinado classificador. Ao invés disso, são construídos conjuntos de classificadores que são treinados com diferentes conjuntos de atributos. São utilizados três tipos de classificadores, de modo a formar conjuntos heterogêneos: entropia máxima, campos condicionais aleatórios (CRF - *conditional random field*) e SVM. Os classificadores geram vários modelos dependentes das representações de atributos que utilizam. A abordagem procura manter a diversidade das amostras, com o objetivo de obter um alto poder de generalização. O problema de encontrar a melhor configuração do sistema, ou seja, os melhores atributos para cada tipo de classificador, é modelado como um problema de otimização multiobjetivo.

O método de Ekbal e Saha (2012) adota a estratégia de fornecer como resultado o conjunto e a fronteira de Pareto. Os objetivos analisados são a cobertura (*recall*) e a precisão da classificação. Os valores das funções objetivo são calculados por validação cruzada 3-*fold*, ou seja, dois conjuntos para teste e um para treinamento, alternados. As amostras apresentam-se em três línguas indianas: Bengali, Hindi e Telugu. O algoritmo utilizado para resolver o problema de otimização multiobjetivo é um AG elitista adaptado para gerar a fronteira de Pareto. Os resultados encontrados mostraram-se superiores à abordagem que utiliza apenas um classificador. No sistema proposto nessa tese, embora seja possível escolher mais de um tipo de classificador para avaliação, apenas um é utilizado em cada cenário de classificação.

O trabalho de Spolaôr (2010) é o que mais se relaciona com o proposto nessa tese. O autor utiliza um algoritmo genético multiobjetivo para resolver o problema da seleção de atributos, de forma a eliminar atributos irrelevantes ou redundantes presentes em bases de dados. A abordagem utilizada é do tipo filtro, como apresenta a Figura 3.1. Essa abordagem foi escolhida devido ao menor custo computacional e foram adotadas preferencialmente critérios que avaliam o conjunto completo de atributos, evitando as avaliações individuais.

As medidas adotadas estão divididas em quatro categorias: consistência, dependência, distância e ganho de informação. Os critérios de avaliação da importância dos atributos consideram a consistência e dependência entre os atributos, a distância entre as classes e o ganho de informação associado ao atributo. São investigadas diferentes combinações de critérios, inclusive em dados não rotulados, pois algumas das medidas não utilizam a classe da amostra no cálculo.

A solução do problema multiobjetivo se dá pelo uso do algoritmo NSGA_II (*Non-Dominated Sorting Genetic Algorithm II*), desenvolvido por Deb et al. (2002). Ao final de sua execução, de posse do conjunto e fronteira de Pareto, é aplicada a técnica da solução ideal de Zeleny (1973), discutida na Seção 2.4 para a escolha de uma única solução.

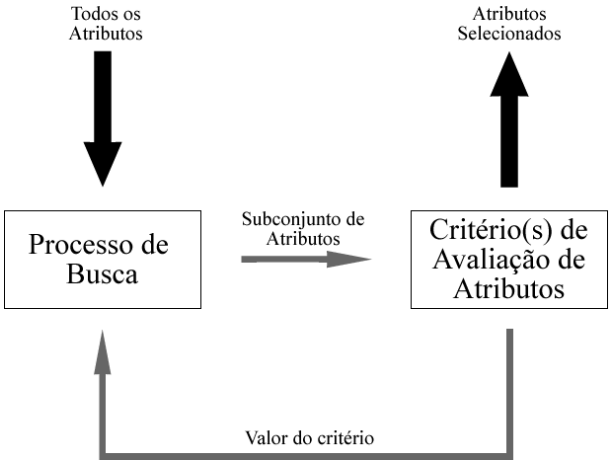

Figura 3.1 - Abordagem filtro para seleção de atributos.
Fonte: Spolaôr (2010)

A metodologia de Spolaôr (2010) não considera o classificador a ser utilizado, ou seja, o modelo que será extraído dos dados. Por esse motivo, as medidas podem não estar de acordo com o que efetivamente ocorre no momento da classificação. Um método do tipo *wrapper*, que utiliza a acurácia da classificação no processo de escolha dos atributos, oferece uma avaliação mais precisa do conjunto de atributos testado, pois já está de acordo com o modelo que será utilizado na classificação. Além disso, seu trabalho não permite a variação do conjunto de classes, que poderia aumentar a qualidade da classificação ao eliminar classes que não podem ser discriminadas satisfatoriamente com o conjunto de atributos disponíveis.

4 UM FORMALISMO PARA A CRIAÇÃO DE CENÁRIOS

Como será visto no próximo capítulo, esta tese propõe a automatização da tarefa de escolher os parâmetros de entrada de um processo de classificação de imagens, dados um conjunto básico de atributos Γ, um conjunto básico de classificadores Ψ e uma estrutura hierárquica de classes O, chamada de ontologia de classes.

Neste capítulo, propõe-se uma formalização do conceito de cenário para a tarefa de classificação com vistas à sua utilização na classificação de imagens de sensoriamento remoto. Nesse contexto, propõe-se que um cenário seja definido como as condições iniciais do problema de classificação, ou seja, as decisões básicas que precedem a realização da classificação em si. Um cenário de classificação considera basicamente três itens: o conjunto de atributos de entrada $F \subseteq \Gamma$, o classificador $\psi \in \Psi$ a ser utilizado e o conjunto de classes de interesse Ω, compatível com a ontologia O. Para definir formalmente o conceito de cenário, torna-se então necessário primeiramente definir o conceito de ontologia de classes.

O termo ontologia, que significa o estudo do ser, vem sendo usado na área de ciência da computação para designar um modelo de dados que representa um conjunto de conceitos dentro de um domínio e os relacionamentos entre estes. Neste trabalho, uma ontologia de classes é uma estrutura hierárquica que modela o conhecimento do usuário a respeito das classes de interesse presentes na imagem. Na ontologia de classes aqui utilizada, as ligações entre as classes representam o conceito de subclasse, com o significado de inclusão. Portanto, o conceito representado por uma superclasse consiste na união dos conceitos representados por suas subclasses. A representação formal da estrutura utilizada para representar uma ontologia neste trabalho é uma estrutura de árvore.

Os conjuntos de classes e árvores de classificação que podem fazer parte de um cenário devem ser compatíveis com a ontologia de classes fornecida pelo usuário. Para verificar essa compatibilidade, é necessário definir alguns formalismos. Uma função renomeadora de árvores possibilita a representação de uma árvore como uma sequência de partições refinantes de um conjunto de classes. Nesta forma, uma árvore pode ser associada a um caminho em um reticulado de partições. A seguir, o reticulado pode ser podado em função da árvore selecionada. Esses elementos são utilizados para comparar o conjunto de classes e a árvore de classificação com a ontologia.

A Seção 4.1 apresenta conceitos básicos de árvores, partições e reticulados. Uma função renomeadora de árvores é apresentada na Seção 4.2 para transformar árvores

em sequências de partições. A Seção 4.3 descreve a relação entre árvores e reticulados. A forma de compatibilizar conjuntos, árvores e ontologias é proposta na Seção 4.4.

4.1 Conceitos básicos

Esta Seção apresenta algumas definições conhecidas da teoria de grafos que serão utilizadas no decorrer do texto, consistindo em uma breve revisão teórica sobre árvores, partições e reticulados.

4.1.1 Árvores

- Uma *árvore* $T = (N, E)$ é um grafo orientado conexo e acíclico formado por um conjunto N de nós, dos quais um é designado como *raiz*. Os nós pertencentes a N estão conectados pelos arcos de E, $E \subseteq N \times N$, de forma que, para cada nó da árvore, existe um caminho que conecta a raiz a este nó.

- O conjunto L_T de *folhas* da árvore T é descrito por:

$$L_T = \{a \in N \mid (\nexists b)(b \in N \text{ e } (a, b) \in E\}.$$

- A *raiz* de uma árvore T é única, denotada por $r(T)$, e definida por:

$$r(T) = \{a \in N \mid (\nexists b)(b \in N \text{ e } (b, a) \in E\}.$$

- A *altura* de um nó a, denotada por $h(a)$, é o tamanho do caminho entre o nó raiz e o nó a. A altura de uma árvore T, denotada por $h(T)$, é a maior altura de seus nós.

- O *número de níveis* da árvore T é dado por $l(T) = h(T) + 1$ e o nível de um nó particular a, $l(a) = h(a) + 1$

- A função $filhos\colon N \to \mathscr{P}(N)$, na qual $\mathscr{P}(N)$ representa o conjunto potência do conjunto N, associa a cada elemento de uma árvore sua descendência imediata. Para $a \in N$ tem-se:

$$filhos(a) = \{b \mid (a, b) \in E\}.$$

A Figura 4.1 traz o exemplo de uma árvore com 11 nós, para a qual $r(T) = k$, $l(T) = 4$, $h(T) = 3$ e $L_T = \{a, b, c, d, e, f, g\}$.

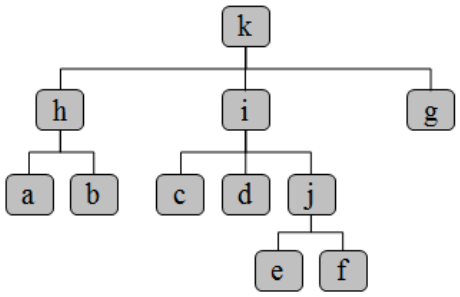

Figura 4.1 - Exemplo de árvore de classes.

4.1.2 Reticulados e Partições

Em Lipschutz e Lipson (1997) podem ser encontradas diversas definições sobre reticulados com variados exemplos. Já Jovanovic (2005) apresenta um texto mais resumido, com apenas as principais definições.

Um *conjunto parcialmente ordenado* A é um conjunto para o qual é definida uma relação binária $x \leq y$, que satisfaz, para todo x,y,$z \in A$, as seguintes condições:

- Reflexiva: $(\forall x \in A)(x \leq x)$
- Antissimétrica: Se $x \leq y$ e $y \leq x$ então $x = y$
- Transitiva: Se $x \leq y$ e $y \leq z$ então $x \leq z$.

Se $x \leq y$ e $x \neq y$, pode-se escrever $x < y$ ou, equivalentemente, $y > x$. Também pode-se dizer que x *está contido* em y ou que y *contém* x. Qualquer subconjunto de um conjunto parcialmente ordenado é também um conjunto parcialmente ordenado.

Um *reticulado* é um conjunto parcialmente ordenado A no qual quaisquer dois elementos de A possuem um máximo limite inferior (*ínfimo*) e um mínimo limite superior (*supremo*) em A. Pode-se usar diagramas de Hasse para representar um conjunto parcialmente ordenado ou um reticulado. Basta desenhar um ponto ou círculo para representar cada elemento de A, posicionando a "mais para cima" de b e desenhando linhas ligando a a b quando $a > b$. A seguir, apagam-se as linhas redundantes, ou seja, as que são consequentes da aplicação das propriedades reflexiva e transitiva da relação binária. Essas linhas ficam implícitas na representação do conjunto ou reticulado. Entre os exemplos de conjuntos parcialmente ordenados, podemos citar: o conjunto $\mathscr{P}(I)$ (conjunto de todos os subconjuntos de I), com a relação $(x \leq y) \leftrightarrow (x \subseteq y)$; e o conjunto $\{1, 2, 3, 6\}$ com a relação $(x \leq y) \leftrightarrow (x \text{ divide } y)$.

A Figura 4.2 mostra o diagrama de Hasse que representa o reticulado do último exemplo. Observa-se que, embora seja verdadeira a relação (1 divide 6), não existe ligação explícita entre os elementos 1 e 6, a ligação está implícita e é consequência da aplicação da propriedade transitiva, por exemplo, em (1 divide 2) e (2 divide 6). O mesmo pode ser dito para a propriedade reflexiva, que torna desnecessário representar, por exemplo, a relação (1 divide 1).

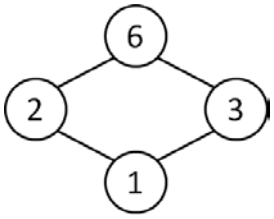

Figura 4.2 - Diagrama do reticulado com $A = \{1, 2, 3, 6\}$, $(x \leq y) \leftrightarrow (x \text{ divide } y)$.

Um reticulado R induzido por um conjunto A e uma relação $\leq$ está associado a um grafo $G_R = (N_R, E_R)$, em que o conjunto de nós $N_R = A$ e as arestas E_R correspondem aos pares ordenados da relação parcial $\leq$, ou seja, para $a, b \in N_R$, $(a, b) \in E_R \iff a \leq b$.

O diagrama para o exemplo do conjunto $\mathscr{P}(I)$ é apresentado na Figura 4.3 para o caso em que $I = \{1, 2, 3\}$ e usa pontos na representação dos elementos do reticulado. Da mesma forma que no exemplo anterior, não é necessário representar explicitamente que $\emptyset \subseteq \emptyset$ nem que $\emptyset \subseteq \{1, 2\}$, por exemplo, pois estas relações estão implícitas, respectivamente, pela reflexividade e transitividade da relação. Pode-se notar também que, apesar de usar linhas simples e não setas, o diagrama é direcionado e as relações são lidas "de baixo para cima". A relação inversa não é permitida, devido à propriedade antissimétrica.

Em um conjunto parcialmente ordenado A, denomina-se *minimal* um elemento $x \in A$ que não é precedido por nenhum outro elemento de A, e *maximal* o elemento que não é sucedido por nenhum outro elemento de A. Um elemento é denominado *primeiro*, denotado por $\bot_R$ se precede qualquer elemento de A, e *último*, denotado por $\top_R$ se sucede qualquer elemento de A. No exemplo mostrado na Figura 4.3, $\{1, 2, 3\}$ é um elemento maximal e também é o último, enquanto $\emptyset$ é um elemento minimal e também é o primeiro elemento.

Um conjunto de nós que conecta dois nós dados em um reticulado é chamado de caminho. A Figura 4.4 mostra dois possíveis caminhos entre $\emptyset$ e $\{1, 2, 3\}$, os nós

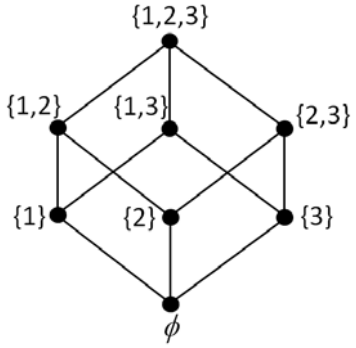

Figura 4.3 - Diagrama do reticulado com $A = \mathscr{P}(\{1, 2, 3\})$, $(x \leq y) \leftrightarrow (x \subseteq y)$.

primeiro e último do reticulado, respectivamente. O caminho ilustrado em 4.4a ocorre através dos arcos expícitos e, saindo de $\emptyset$, passa pelos nós $\{1\}$ e $\{1, 2\}$ para chegar a $\{1, 2, 3\}$. Já o caminho mostrado em 4.4b, faz uso de um arco implícito, garantido pela propriedade transitiva, para ir de $\{1\}$ diretamente para $\{1, 2, 3\}$.

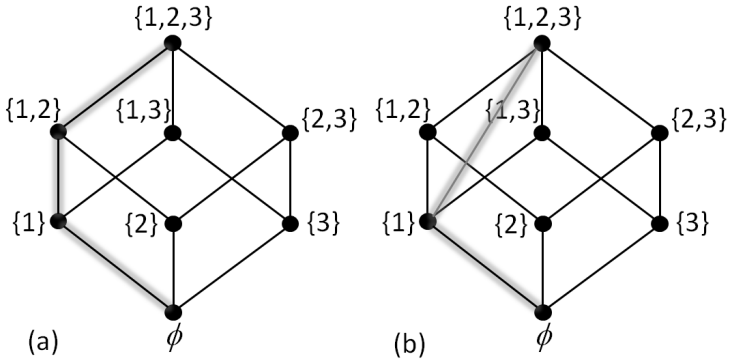

Figura 4.4 - Dois caminhos possíveis entre $\emptyset$ e $\{1, 2, 3\}$.

Uma partição de um conjunto A é um conjunto π de subconjuntos não vazios de A tal que cada elemento de A pertence a um e apenas um dos elementos de π. Ou seja, a união de todos elementos de π é igual a A, e a interseção entre quaisquer dois dos elementos de π é vazia. Por exemplo, $\{\{1, 3\}, \{2, 4, 7\}, \{5, 6\}\}$ é uma partição de $\{1, 2, 3, 4, 5, 6, 7\}$.

Dadas duas partições π_1 e π_2 de um conjunto A, diz-se que π_1 é um refinamento de π_2 se todo elemento de π_1 é subconjunto de algum elemento de π_2. Por exemplo, a partição $\{\{1, 3\}, \{2, 4, 7\}, \{5, 6\}, \{8\}\}$ de $\{1, 2, 3, 4, 5, 6, 7, 8\}$ é um refinamento da partição $\{\{1, 3, 2, 4, 7\}, \{5, 6, 8\}\}$. Formalmente, uma partição π_i refina uma partição

π_j, denotado por $\pi_i \prec \pi_j$, se, e somente se, $(\forall \alpha \in \pi_i)(\exists \beta \in \pi_j \mid \alpha \subseteq \beta)$. No exemplo, é verdadeira a relação $\{\{1,3\},\{2,4,7\},\{5,6\},\{8\}\} \prec \{\{1,3,2,4,7\},\{5,6,8\}\}$.

Dado um conjunto A, o conjunto de todas as partições que podem ser obtidas a partir de A é um reticulado com a relação $(\pi_i \leq \pi_j) \iff (\pi_i \prec \pi_j)$.

4.2 Árvores como sequências de partições

Seja $renom\colon N \to \mathscr{P}(L_T)$ uma função renomeadora de árvores que substitui um nó a pelo conjunto de todas as folhas que podem ser alcançadas a partir de a. A função recursiva $renom$ é dada por:

$$renom(a) = \begin{cases} \{a\}, & \text{se } a \in L_T, \\ \bigcup_{b \in filhos(a)} renom(b), & \text{se } a \notin L_T. \end{cases}$$

Por exemplo, para a árvore da Figura 4.1, tem-se:

$$\begin{array}{lll} renom(a) = \{a\}, & renom(b) = \{b\}, & renom(c) = \{c\}, \\ renom(d) = \{d\}, & renom(e) = \{e\}, & renom(f) = \{f\}, \\ renom(g) = \{g\}, & renom(h) = \{a,b\}, & renom(i) = \{c,d,e,f\}, \\ renom(j) = \{e,f\}, & renom(k) = \{a,b,c,d,e,f,g\}. & \end{array}$$

Usando a função renomeadora $renom$ na árvore $T = (N, E)$, obtém-se a árvore $T' = (N', E')$ em que $N' = \{renom(a) \mid a \in N\}$ e $E' = \{(renom(a), renom(b)) \mid (a,b) \in E\}$. Tem-se, portanto, $N' \subseteq \mathscr{P}(N)$. A Figura 4.5 traz a árvore T' obtida a partir da árvore T da Figura 4.1.

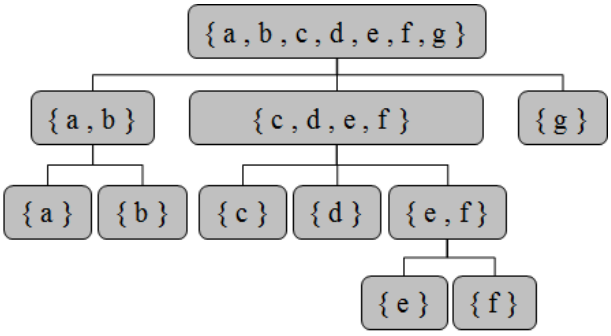

Figura 4.5 - Mesma árvore da Figura 4.1 após o uso da função renomeadora.

A árvore T' obtida dessa forma possui a mesma estrutura da árvore T original, ou seja, são isomorfas. Porém, seus conjuntos de nós e arestas são diferentes. Por

exemplo, $L_{T'} = \{\{a\}, \{b\}, \{c\}, \{d\}, \{e\}, \{f\}, \{g\}\}$.

Uma árvore $T' = (N', E')$, construída a partir de uma árvore $T = (N, E)$, pode ser descrita como um conjunto de partições $P(T') = \{\pi_1, \cdots, \pi_k\}$, $k = l(T)$ (vide anexo para definição de partição). As partições π_i são definidas como:

$$\begin{aligned} \pi_1 &= \{L_T\} \\ \pi_{i+1} &= \bigcup_{A \in \pi_i} \{\alpha(A)\}, \;\; 1 \leq i \leq h(T) \end{aligned}$$

em que a função $\alpha(a)$ resulta no conjunto dos nós descendentes de um nó a da árvore T', caso existam, ou o próprio nó caso contrário, conforme a definição:

$$\forall a \in N', \alpha(a) = \begin{cases} filhos(a), & \text{se } filhos(a) \neq \emptyset, \\ a, & \text{caso contrário.} \end{cases}$$

Por exemplo, a função α aplicada a alguns nós do exemplo da Figura 4.5 resulta em:

$$\begin{aligned} \alpha(\{a, b, c, d, e, f, g\}) &= \{\{a, b\}, \{c, d, e, f\}, \{g\}\} \\ \alpha(\{g\}) &= \{g\}. \end{aligned}$$

Para a árvore da Figura 4.5, obtém-se o conjunto de partições:

$$\begin{aligned} \{ \quad & \{\{a, b, c, d, e, f, g\}\}, \\ & \{\{a, b\}, \{c, d, e, f\}, \{g\}\}, \\ & \{\{a\}, \{b\}, \{c\}, \{d\}, \{e, f\}, \{g\}\}, \\ & \{\{a\}, \{b\}, \{c\}, \{d\}, \{e\}, \{f\}, \{g\}\} \quad \} \end{aligned}$$

O conjunto N' de nós da árvore T' pode ser obtido por $\bigcup_i \pi_i$. Para o exemplo citado, correspondente à árvore da Figura 4.5, $N' = \{\{a, b, c, d, e, f, g\}, \{c, d, e, f\}, \{a, b\}, \{e, f\}, \{a\}, \{b\}, \{c\}, \{d\}, \{e\}, \{f\}, \{g\}\}$.

Foi demonstrado como derivar um conjunto de partições para uma árvore dada T. Por outro lado, dado um conjunto de partições, é possivel verificar se ele corresponde a uma árvore. Para tanto, basta verificar se seus elementos podem constituir uma ordem parcial do conjunto de partições, com a relação $\prec$ (refinamento). Tanto o conjunto $P(T')$, obtido a partir da árvore T, quanto a árvore T, obtida a partir de um conjunto de partições $P(T')$, são únicos.

Para efeito de simplificação da notação, a partir de agora o conjunto $\{\{a, b\}\}$ será

representado por ab e o conjunto $\{\{a\},\{b\}\}$ será representado por a-b. No exemplo, tem-se $P(T') = \{$ $abcdefg$, ab-$cedf$-g, a-b-c-d-ef-g, a-b-c-d-e-f-g $\}$.

4.3 Árvores como caminhos em um reticulado

Como visto em 4.2, uma árvore de classes T pode ser modelada como uma sequência de partições $P(T')$. A partir de T, pode-se então obter o reticulado R_T induzido pelo conjunto de todas as partições possíveis do conjunto de classes folha L_T e a operação de refinamento $\prec$. A Figura 4.6 mostra o reticulado induzido por $\prec$ e o conjunto de classes folha $L_T = \{1, 2, 3, 4\}$, em notação simplificada.

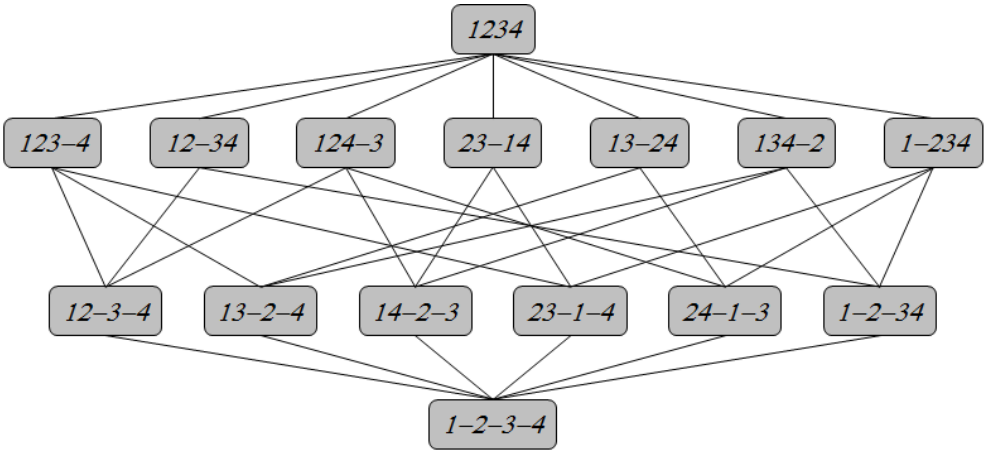

Figura 4.6 - Reticulado R_T para $L_T = \{1, 2, 3, 4\}$.

Em um reticulado R_T, o caminho entre um nó qualquer e o $\top_R$ representa uma árvore. Na Figura 4.6, pode-se gerar árvores com até 4 níveis utilizando os caminhos entre os nós de R_T e 1234. Por exemplo, o caminho destacado na Figura 4.7 representa a árvore $\{\{\{1, 2, 3, 4\}\}, \{\{1, 2, 3\}, \{4\}\}, \{\{1, 2\}, \{3\}, \{4\}\}, \{\{1\}, \{2\}, \{3\}, \{4\}\}\}$, apresentada na Figura 4.8. Nesse caso, o caminho utilizado é entre o primeiro e o último nó, mas também podem ser geradas árvores a partir de caminhos menores, como por exemplo a árvore $\{\{\{1, 2, 3, 4\}\}, \{\{1, 2, 3\}, \{4\}\}\}$, que tem dois nós folha, $\{1, 2, 3\}$ e $\{4\}$.

A representação gráfica do reticulado não mostra os arcos redundantes implícitos pela propriedade transitiva da ordenação parcial, mas eles existem e podem ser usados em um caminho (ver exemplo da Seção 4.1, Figura 4.4). Com o uso dos arcos implícitos, também é possível utilizar o caminho da Figura 4.9, obtendo-se assim a árvore da Figura 4.10, ou seja, $\{\{\{1, 2, 3, 4\}\}, \{\{2, 3\}, \{1\}, \{4\}\}, \{\{1\}, \{2\}, \{3\}, \{4\}\}\}$.

Na Figura 4.9, os arcos tracejados representam os três pares de arcos que justificam a existência do arco implícito destacado, pela propriedade transitiva.

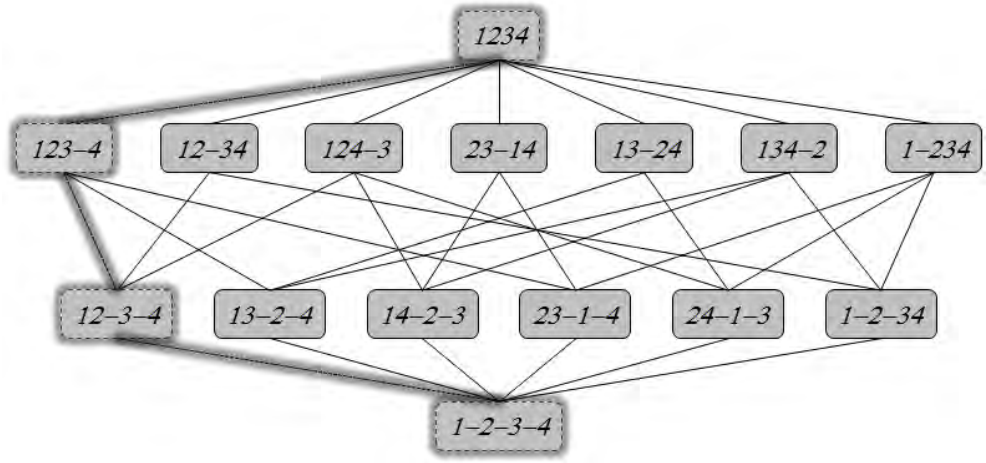

Figura 4.7 - Caminho sobre o reticulado, representando uma árvore.

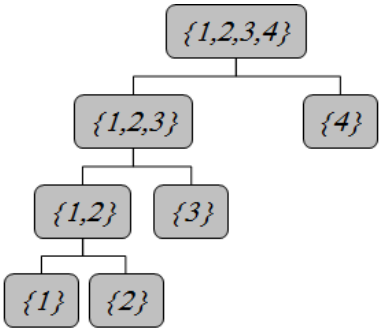

Figura 4.8 - Árvore equivalente ao caminho destacado na Figura 4.7.

Utilizando o conjunto L_T das folhas da árvore T, pode-se derivar todas as árvores possíveis que possuem este conjunto de folhas. Por exemplo, a Figura 4.11 traz os formatos de árvores possíveis com quatro folhas.

Além disso, para uma estrutura em particular, podem ser geradas várias árvores, mudando-se a disposição de seus nós. Por exemplo, as seis possíveis árvores formadas a partir da forma da Figura 4.11c podem ser vistas na Figura 4.12. O nó interno representado por $\{3, 4\}$ correponde à união das folhas $\{3\}$ e $\{4\}$.

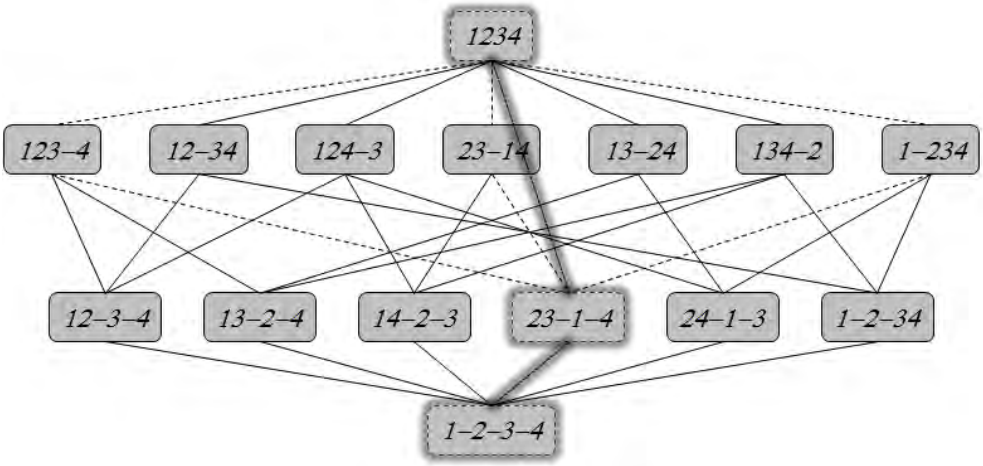

Figura 4.9 - Caminho sobre o reticulado usando um arco implícito.

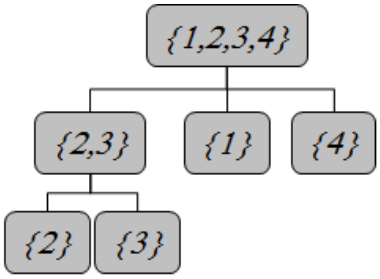

Figura 4.10 - Árvore equivalente ao caminho destacado na Figura 4.9.

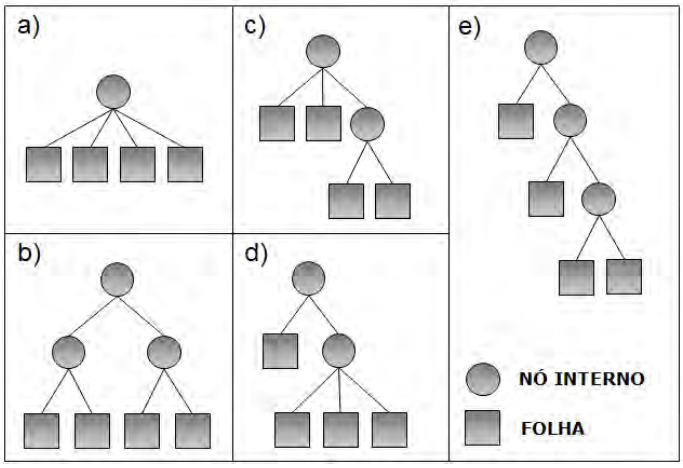

Figura 4.11 - Formas possíveis de árvores com quatro folhas.

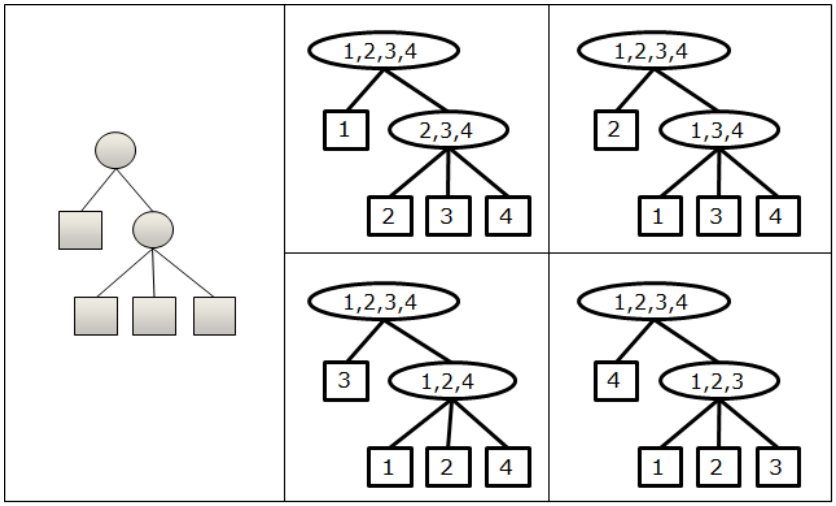

Figura 4.12 - Árvores possíveis a partir da mesma forma.

4.4 Compatibilidade entre árvores sobre um reticulado

Neste trabalho estamos interessados em verificar se uma árvore é compatível com outra no mesmo reticulado. O processo se inicia com a uma árvore dada T. A partir do conjunto das partições $P(T')$ (chamado de ontologia de classes) constrói-se o reticulado R. Em seguida, tomamos uma árvore qualquer T_0 (chamada de árvore de classificação) que representa um caminho entre um nó qualquer e $\top_R$, o último nó do reticulado R. Finalmente verificamos se T_0 é compatível com T, utilizando o reticulado R.

No contexto deste trabalho, as estruturas de árvores de classes são pois utilizadas para representar duas estruturas hierárquicas de classes com propósitos diferentes: ontologias e árvores de classificação. Uma ontologia é dada inicialmente e não é alterada no processo de otimização proposto. Já as árvores de classificação são geradas iterativamente pelo gerador de cenários, conforme explicado no Capítulo 5. No modelo proposto nesse trabalho, só podem ser geradas árvores de classificação que sejam compatíveis com a ontologia fornecida. O objetivo desta Seção é definir como essa compatibilidade pode ser verificada.

Já foi visto nessa tese que uma árvore de classes T pode ser representada por uma sequência de partições $P(T')$ e, a seguir, pode-se obter o reticulado R_T induzido pelo conjunto de todas as partições possíveis do conjunto de classes folha L_T e a relação de refinamento $\prec$.

Para definir a compatibilidade de uma árvore com uma ontologia dada, é necessário

introduzir o conceito de *reticulado podado*. Seja R o reticulado de partições em Ω_L, com a relação de refinamento $\prec$. Seja O uma ontologia. O reticulado R_O, obtido através da poda do reticulado R, eliminando os nós não compatíveis com O, é obtido como descrito abaixo.

Seja $O = P(T') = \{\pi_1, \cdots, \pi_{np}\}$ uma ontologia onde π_i é uma partição de Ω_L, com $np = |L_T|$. Seja $G_R = (N_R, E_R)$, em que N_R é o conjunto de nós do reticulado R e E_R é seu conjunto de arestas. O grafo $G_{RO} = (N_{RO}, E_{RO})$ associado ao novo reticulado R_O é formado por

$$N_{RO} = \{M \in N_R \mid (\forall \pi_i \in O)(\pi_i \prec M \text{ ou } M \prec \pi_i)\}.$$

Portanto, garante-se que cada nó em R_O está em uma linha de descendência ou ascendência dos nós pertencentes a O. O conjunto de arestas E_{RO} conserva todas as arestas de G_R entre os nós do conjunto N_{RO}, ou seja:

$$E_{RO} = \{(M_1, M_2) \mid M_1, M_2 \in N_{RO} \text{ e } (M_1, M_2) \in E_R\}.$$

Uma árvore de classificação é definida por um caminho de tamanho k entre um nó do reticulado R_O e $\top_{RO} = \top_R$, sendo $k \leq |L_T|$.

No exemplo da Figura 4.7, a ontologia escolhida passa por todos os níveis do reticulado, portanto o reticulado podado corresponde à própria ontologia (Figura 4.13), e a árvore de classificação não poderá incluir nenhum nó que não esteja presente na ontologia. Pode-se, entretanto, retirá-los. As possíveis árvores de classificação geradas a partir da ontologia da Figura 4.8 são mostradas na Figura 4.14.

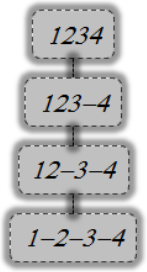

Figura 4.13 - Reticulado podado R_O para a ontologia da Figura 4.8.

Já no exemplo da Figura 4.9, a ontologia escolhida não passa pelo segundo nível

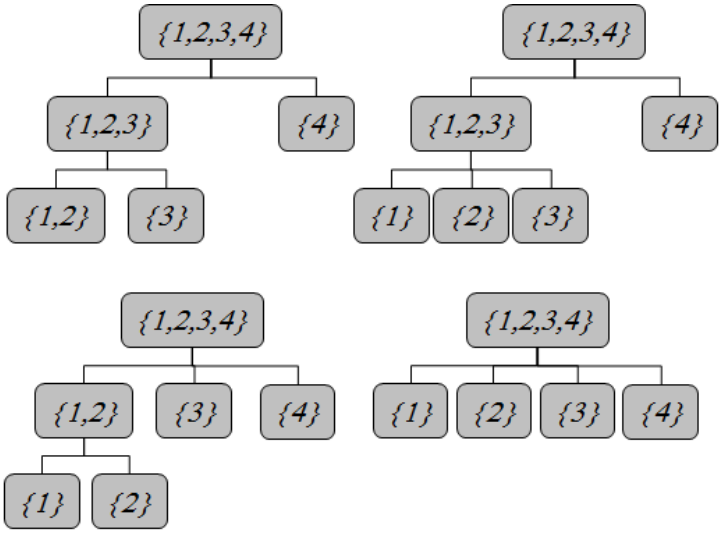

Figura 4.14 - Exemplos de árvores que podem ser geradas a partir da ontologia da Figura 4.8.

do reticulado, isto é, todos os nós deste nível que possuirem ligação com qualquer nó da ontologia podem ser incluídos no reticulado podado (Figura 4.15), e a árvore de classificação poderá incluir nós que não estejam presentes na ontologia original fornecida.

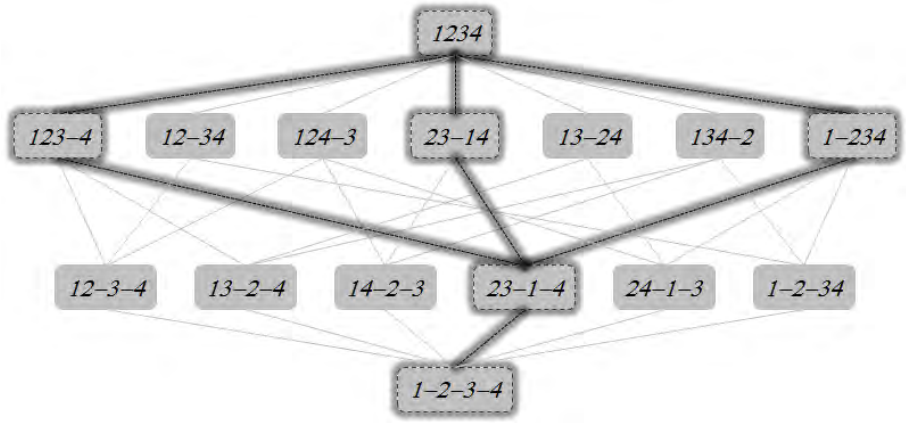

Figura 4.15 - Reticulado podado R_O para a ontologia da Figura 4.10.

Esse conceito de árvore de classificação permite a definição de outro tipo de cenário, com o objetivo de ser utilizado em um sistema de classificação sequencial, conforme detalhado na Seção 5.1.2.

5 MÉTODO PROPOSTO

Neste trabalho é proposto um sistema de classificação alternativo ao mostrado na Figura 1.1. Um diagrama que ilustra a arquitetura do método proposto pode ser visto na Figura 5.1. A captação de uma cena terrestre por um sensor gera um conjunto de dados original, a partir dos quais são gerados novos atributos (dados derivados). A entrada do sistema automatizado é um conjunto de dados de referência, um conjunto de atributos e uma ontologia. Estes três elementos derivam diretamente um cenário inicial completo. O método proposto consiste em um processo iterativo de análise automática de cenários, os quais são formados por um classificador, um conjunto de atributos e um conjunto de classes ou uma árvore de classificação. Com base nos dados de dados de entrada fornecidos pelo usuário e na avaliação obtida nas iterações anteriores, novos cenários são gerados e avaliados em um processo de otimização. Ao final, a análise é apresentada ao usuário, que escolhe o cenário mais apropriado para sua aplicação.

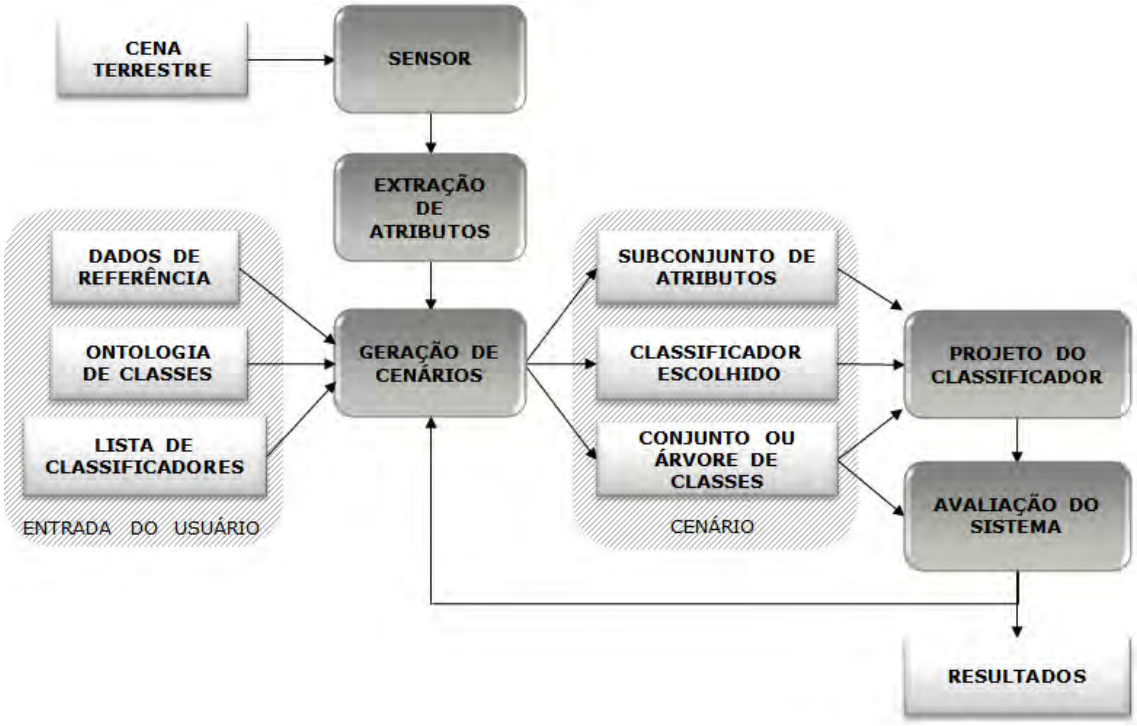

Figura 5.1 - Método proposto para as etapas da tarefa de classificação de imagens em sensoriamento remoto.

O cenário inicial fornecido pelo usuário deve conter os dados rotulados, compostos por um conjunto de atributos e um conjunto de classes de interesse. Esses dados são

os mesmos que seriam usados em um modelo tradicional. Para aproveitar as vantagens do sistema analisador de cenários, alguns dados adicionais podem ser fornecidos, como o classificador inicial utilizado na análise ou o subconjunto de classificadores que se deseja testar. Preferencialmente, as classes devem estar organizadas em forma de hierarquia, pois esta representa uma visão da cena, uma descrição rudimentar do que o pesquisador espera que o classificador discrimine na imagem. Entretanto, também é possível especificar apenas um conjunto de classes, sem qualquer relação estrutural. Nesse caso, a ontologia acrescenta apenas uma raiz com a união das classes fornecidas, que são as folhas. Pode-se ainda utilizar um programa para gerar uma hierarquia de forma automática, com base na similaridade das classes, usando os atributos disponíveis.

Este capítulo utiliza o formalismo desenvolvido no Capítulo 4 para descrever o método proposto neste trabalho. A Seção 5.1 apresenta os diferentes tipos de cenários de classificação permitidos pela metodologia. A seguir, a Seção 5.2 mostra como o problema inicial foi modelado como um problema de otimização multiobjetivo. Finalmente, a Seção 5.3 descreve o método proposto para reduzir a fronteira de Pareto encontrada como resultado da otimização.

5.1 Cenários de classificação

Esta Seção descreve alguns dos diferentes tipos de cenários que podem ser utilizados dentro da metodologia proposta nessa tese. A Seção 5.1.1 fala sobre um tipo básico de cenário, contendo apenas três elementos: um classificador, um conjunto de atributos e um conjunto de classes. Um cenário para classificação sequencial é descrito na Seção 5.1.2 e outras possibilidades para definições de cenários são apresentadas na Seção 5.1.3.

5.1.1 Cenário básico de classificação

A proposta deste trabalho é considerar, como cenário básico para a tarefa de classificação, três elementos principais, como mostra a Figura 5.2.

O *conjunto de atributos* é selecionado do conjunto total de atributos disponíveis, simples ou derivados. Através do processo de análise de cenários, deseja-se eleger o subconjunto mais adequado para discriminar as classes de interesse presentes no *conjunto de classes*. Da mesma forma, o *conjunto de classes* também é avaliado, e são selecionadas as classes que podem ser discriminadas pelo classificador escolhido, considerando os dados disponíveis e a acurácia desejada.

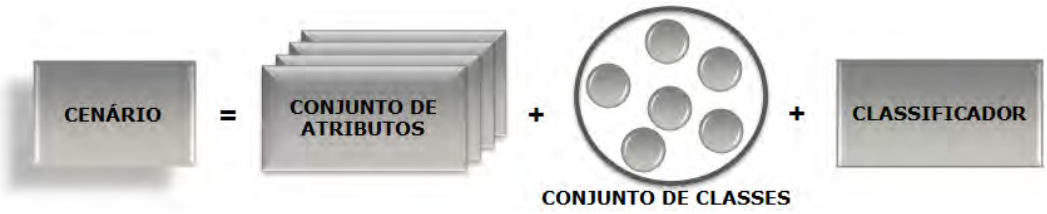

Figura 5.2 - Elementos principais de um cenário.

O componente *classificador* é composto pelo tipo do classificador, como Máxima Verossimilhança (Maxver), Máquina de Vetores Suporte (SVM) ou Árvore de Decisão (DT), e também pelos parâmetros necessários inerentes a cada tipo. Para o classificador SVM, por exemplo, é necessário especificar o tipo do *kernel* (linear, gaussiano, sigmóide), o valor da penalidade e ainda parâmetros relativos a tipos de *kernel* específicos, como o desvio padrão no caso do *kernel* gaussiano.

A ideia é analisar diferentes cenários e identificar os melhores para determinada tarefa de classificação. Uma imagem classificada I_S é gerada quando um classificador ψ utiliza os elementos especificados em um cenário S para classificar uma imagem I.

Figura 5.3 - Classificação a partir de um cenário.

Formalmente, um cenário básico de classificação S pode ser definido como uma tripla $S = (F, \Omega, \psi)$, em que:

- F é um conjunto de atributos, subconjunto do total de atributos disponíveis Γ
- Ω é um conjunto de classes de interesse
- ψ é um algoritmo selecionado do conjunto Ψ de algoritmos supervisionados para classificação de imagens.

Todos os elementos da descrição são variáveis, ou seja, podem mudar de um cenário para outro. Os valores dessas variáveis para cada cenário são atribuídos pelo *gerador de cenários*, componente do sistema de análise. Qualquer algoritmo de classificação pode ser escolhido dentre os disponíveis, assim como qualquer subconjunto de Γ com

pelo menos um atributo pode ser escolhido como F.

Já o conjunto de classes de cada cenário é restrito pela ontologia utilizada. Somente são permitidos conjuntos de classes compatíveis com a estrutura hierárquica. Pode ser utilizada qualquer uma das partições refinantes que define a ontologia ou qualquer outra partição que possa ser inserida na sequência de forma a conservá-la como uma sequência refinante de partições. No exemplo da Figura 4.10, os conjuntos de classes permitidos incluem $\{\{\omega_1\},\{\omega_2\},\{\omega_3\},\{\omega_4\}\}$ e $\{\{\omega_1\},\{\omega_2,\omega_3\},\{\omega_4\}\}$, mas não $\{\{\omega_1,\omega_2\},\{\omega_3\},\{\omega_4\}\}$ nem $\{\{\omega_1\},\{\omega_2\},\{\omega_3,\omega_4\}\}$. Estes últimos conjuntos contrariam a estrutura fornecida, que associa as classes ω_2 e ω_3 antes de uni-las às demais. Conceitualmente também seria possível considerar, por exemplo, os conjuntos $\{\{\omega_2,\omega_3\},\{\omega_1,\omega_4\}\}$ e $\{\{\omega_1,\omega_2,\omega_3\},\{\omega_4\}\}$, mas a implementação atual do sistema não contempla essa possibilidade.

5.1.2 Cenário básico para classificação sequencial

Para realizar a classificação sequencial, é necessário definir uma estrutura, denominada árvore de classificação, que determina a ordem em que os subconjuntos de classes são separados pelo classificador. A *árvore de classificação* é utilizada como um guia para a tarefa de classificação. Ela especifica a ordem em que os subconjuntos de classes são separados, em um processo sequencial de classificação hierárquica. A árvore deve ser compatível com uma estrutura definida previamente, denominada ontologia de classes. A ontologia de classes pode ser fornecida pelo usuário ou gerada pelo sistema. Da mesma forma que as ontologias, uma árvore de classificação pode ser definida por uma sequência de partições ou por uma árvore. A representação formal dessa estrutura foi apresentada no Capítulo 4. Nesse tipo de classificação, usa-se um cenário em que o conjunto de classes é estruturado, ou seja, está associado a uma árvore de classificação, conforme Figura 5.4.

Figura 5.4 - Elementos principais de um cenário para classificação hierárquica.

Da mesma forma que na classificação tradicional (Figura 5.3), a utilização de um

cenário S e de um conjunto de dados (imagem I) por um classificador produzirá uma imagem classificada I_S. Entretanto, o classificador é mais complexo do que no cenário básico, trata-se de um sistema de classificação que pode ser composto por diversos tipos de classificadores, ou, como se considera aqui, pela aplicação sequencial do mesmo tipo de classificador ψ.

Um cenário básico S para classificação sequencial pode ser definido como uma quádrupla $S = (F, \Omega, \psi, T)$, em que:

- F é um conjunto de atributos, $F \subseteq \Gamma$
- Ω é um conjunto de classes de interesse
- ψ é um algoritmo selecionado do conjunto Ψ de algoritmos supervisionados para classificação de imagens
- T é uma árvore de classificação.

5.1.3 Outros cenários de classificação

Nesta Seção são sugeridas algumas variações que podem ser incluídas nos cenários de classificação, conforme necessidade da aplicação.

No caso em que a classificação hierárquica considera de forma independente a escolha do classificador e dos atributos para cada etapa de classificação, o classificador escolhido ψ deve ser substituído por um conjunto $\Psi_0 \subseteq \Psi$ de algoritmos classificadores e uma função $g\Psi$ que associa os nós da árvore T', $T' = (N', E')$, a um classificador $(g\Psi : N' \rightarrow \Psi_0)$. Além disso, é necessário incluir uma função $g\Gamma$ que associa os nós da árvore T' a um conjunto de atributos, subconjunto de Γ_0, que por sua vez é subconjunto de Γ. Ou seja, a função é definida como $(g\Gamma : N' \rightarrow \mathscr{P}(\Gamma_0))$. Para essa situação, um cenário pode ser descrito como uma 7-upla: $S = (\Gamma_0, \Omega, \Psi_0, O, T', g\Psi, g\Gamma)$.

Em outro caso, quando o usuário não deseja adicionar uma ontologia de classes, ela pode ser considerada como $O = \{L_T, \{\{\omega_i\}, \cdots, \{\omega_n\}\}\}$ ou então simplesmente omitida da descrição. Nesse caso, permite-se que qualquer estrutura seja gerada a partir do conjunto L_T de classes de interesse, conforme mostrado na Seção 4.4.

5.2 Otimização Multiobjetivo

O problema de encontrar o melhor cenário para a tarefa de classificar uma imagem foi modelado como um problema de otimização multiobjetivo.

De acordo com Cohon (1978), os passos para realizar uma metodologia de planejamento multiobjetivo são:

a) Identificação e quantificação dos objetivos

b) Definição de variáveis de decisão e restrições

c) Coleta de dados

d) Geração e avaliação das alternativas

e) Seleção da melhor alternativa

f) Implementação da alternativa selecionada.

Cohon (1978) ressalta a diferença entre ideal e objetivo. Segundo o autor, ideal é uma afirmação avaliativa geral com a qual normalmente todos concordam, enquanto que objetivo é uma afirmação operacionalmente útil que é consistente com um ideal subjacente mas que pode ser que nem todos concordem. Em geral, o objetivo é expresso através de uma função matemática das variáveis de decisão de um problema. Por exemplo, na aplicação de classificação de imagens, querer aumentar a acurácia da classificação é um ideal, enquanto que utilizar uma medida para isso, como o coeficiente kappa ou a acurácia global, o transforma em um objetivo. No modelo da otimização multiobjetivo, a comparação entre duas soluções viáveis, para definir qual é a melhor, é baseada em todos os objetivos considerados.

No contexto desse trabalho, os ideais identificados foram:

- maximizar a qualidade da classificação;
- encontrar o conjunto ótimo de atributos para a classificação;
- maximizar a especificidade das classes discriminadas.

Outros ideais poderiam ter sido considerados, conforme a necessidade do usuário e da sua aplicação. As possibilidades incluem: maximizar o poder de generalização do modelo gerado pelo classificador; maximizar a uniformidade das áreas classificadas; minimizar o tempo de classificação; ou ainda minimizar o erro temático por píxel, como proposto por Brown et al. (2009).

Para cada um dos ideais considerados nesse trabalho, formulou-se um modo de avaliação, que consiste no objetivo a ser otimizado. Em alguns casos, mais de uma formulação está disponível no sistema, oferecendo diversas opções ao usuário.

Para avaliar a qualidade da classificação, propõe-se utilizar uma medida de acurácia. No sistema implementado estão disponíveis quatro medidas: acurácia global, acurácia do usuário, o coeficiente kappa, e o kappa do usuário, conforme as equações citadas na Seção 2.2. As duas medidas do usuário são médias ponderadas das medidas específicas de cada classe, e são recomendadas no caso em que se deseja dar maior importância para a acurácia de determinadas classes. Os pesos utilizados na média são valores de prioridade definidos pelo usuário. Por exemplo, pode-se atribuir prioridade máxima para a classe mais importante para a aplicação, dificultando assim sua eliminação do conjunto pelo analisador. Outra possibilidade é utilizar uma medida do usuário em conjunto com uma medida global, atribuindo a prioridade máxima a uma determinada classe e um valor baixo ou nulo para as demais. Assim, pode-se obter o melhor mapa possível sem degradar a principal classe de interesse.

No caso da acurácia global e do kappa, os objetivos são calculados na forma mostrada na Seção 2.2, Equações 2.3 e 2.4. Já no último caso, a medida κu_i mostrada na Equação 2.6 deve ser poderada pela prioridade da classe, p_i ($i = 1, \ldots, |\Omega|$), fornecida pelo usuário. Para utilizar esta medida, o objetivo a ser maximizado é a acurácia global ponderada pela prioridade da classe, agp, apresentada na Equação 5.1. O mesmo ocorre para a acurácia do usuário.

$$agp = \sum_{i=1}^{n_C} p_i \kappa u_i \tag{5.1}$$

Já no caso do conjunto F de atributos utilizados em um cenário particular, foi considerado como ótimo aquele com menor quantidade de atributos. Ou seja, o segundo objetivo consiste em minimizar a expressão $|F|$ (quantidade de elementos do conjunto F de atributos selecionados).

Para avaliar a especificidade das classes, optou-se também por considerar o tamanho do conjunto de classes Ω utilizado no cenário. Antes de optar por essa escolha, foram avaliadas medidas que consideram o nível da hierarquia atingido, porém estas não se mostraram satisfatórias do ponto de vista semântico. Em uma hierarquia de classes, nem todas as folhas possuem a mesma altura. Uma classe pode estar em uma folha, e ainda assim pertencer ao segundo nível em uma árvore de altura 5. Ou então pode estar no quarto nível, possuindo ainda dois níveis de descendentes. Por exemplo, na Figura 4.1, a classe i está no mesmo nível que g, mas elas não poderiam receber a mesma nota no quesito especificidade. Enquanto g foi detectada no grau máximo de detalhe proposto pelo usuário, para i o ideal seria que fosse dividida em quatro

subclasses. Da mesma forma, não foi considerado adequado atribuir a mesma nota a g e f, já que essa última pode ser generalizada para j. Portanto, o objetivo considerado foi maximizar $|\Omega|$.

Após definir as funções objetivo, foram formuladas as restrições do problema, ou seja, em que condições uma solução é considerada viável.

Na prática, o problema foi modelado como um problema de minimização, ou seja, foi adicionado um sinal negativo aos objetivos que se deseja maximizar, e a formulação do problema para o caso de utilizar o κ como medida de acurácia é expresso por:

$$
\begin{aligned}
&\textbf{Minimizar} \\
&\qquad -\kappa \\
&\qquad |F| \\
&\qquad -|\Omega| \\
&\textbf{Sujeito a} \\
&\qquad F \subseteq \Gamma \\
&\qquad |F| \geq 1 \\
&\qquad \Omega \in O \\
&\qquad |\Omega| \geq 2 \\
&\qquad \kappa > \kappa_{min}
\end{aligned}
$$

As restrições utilizadas na formulação são:

- o conjunto de atributos F utilizado deve ser subconjunto do conjunto de atributos disponíveis Γ;
- o conjunto de atributos F não pode ser vazio;
- o conjunto de classes Ω deve ser derivado da estrutura hierárquica de classes utilizada O;
- o conjunto de classes Ω deve possuir no mímino duas classes;
- a acurácia da classificação supera um valor mínimo.

5.3 Redução da fronteira de Pareto

Após a obtenção da fronteira de Pareto para o problema de otimização descrito na Seção 5.2, propõe-se um método para reduzir a sua quantidade de pontos, com o intuito de facilitar a escolha do usuário.

O método proposto consiste de uma ordenação dos elementos da fronteira por um critério selecionado pelo usuário e o agrupamento dos elementos com mesmo valor para o critério escolhido. A seguir, é realizado um teste de hipótese Z sobre o valor da acurácia dos elementos pertencentes ao mesmo grupo. Caso os valores não possuam diferença estatisticamente significativa para o nível de confiança escolhido, um dos pontos é eliminado da fronteira, pois considera-se que é equivalente utilizar este ponto ou algum outro do mesmo grupo. Os critérios para a ordenação e para a eliminação dos pontos são selecionados pelo usuário na interface do sistema, mostrada adiante na Figura 6.9.

O teste Z realizado para o índice kappa é unilateral e suas hipóteses são:

$$\begin{aligned} H_0 &: \kappa_1 = \kappa_2 \\ H_1 &: \kappa_1 > \kappa_2. \end{aligned}$$

A estatística z é calculada pela Equação 5.2, na qual $\hat{\sigma}^2_{\hat{\kappa}_1}$ é a variância de $\hat{\kappa}_1$ e $\hat{\sigma}^2_{\hat{\kappa}_2}$ é a variância de $\hat{\kappa}_2$. O valor encontrado é comparado ao z_{crit}, calculado para o nível de confiança desejado (Figura 5.5).

$$z = \frac{\hat{\kappa}_1 - \hat{\kappa}_2}{\sqrt{\hat{\sigma}^2_{\hat{\kappa}_1} + \hat{\sigma}^2_{\hat{\kappa}_2}}} \tag{5.2}$$

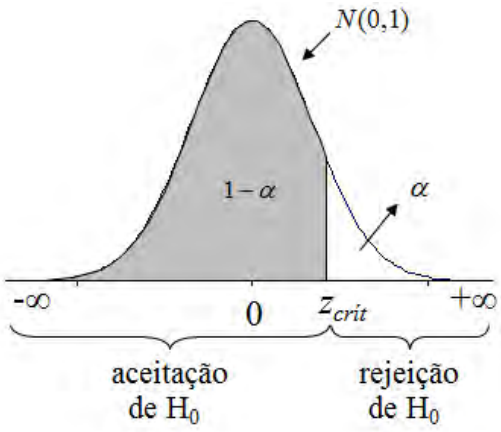

Figura 5.5 - Regiões de rejeição e aceitação da hipótese nula no teste Z.

6 IMPLEMENTAÇÃO

Este capítulo explica como foi modelado e implementado o sistema analisador de cenários. O objetivo de realizar a implementação é validar a metodologia proposta, avaliando seu desempenho na aplicação de problemas reais. O algoritmo foi codificado na linguagem IDL (EXELIS, 2012).

6.1 IDL

A linguagem IDL (*Interactive Data Language*) é uma linguagem de alto nível concebida especificamente para programação científica e análise de dados. A linguagem inclui um conjunto de ferramentas básicas de programação, como tipos numéricos, strings, vetores, estruturas, ponteiros, além dos comandos usuais para condicionais, laços, definição de funções e procedimentos. Mas sua principal vantagem para a área de processamento de imagens consiste na facilidade para manipular estruturas vetoriais e matriciais sem a necessidade de laços e de forma bastante eficiente. Usualmente, uma imagem é armazenada como uma estrutura vetorial com três dimensões: número de linhas, de colunas e de canais. IDL permite o uso de estruturas com até sete dimensões. Também está disponível um conjunto bastante extenso de funções gráficas, estatísticas, de álgebra linear, análise de dados e outras, além de características de programação orientada a objetos.

O termo IDL também pode se referir ao ambiente em que o programador desenvolve sua aplicação, utilizando a linguagem IDL. É um ambiente interativo, com um interpretador em que é possível digitar um comando e ver os resultados imediatamente. A desvantagem do seu uso é que as linguagens compiladas em geral são otimizadas pelo compilador, tornando sua execução mais rápida e eficiente do que o mesmo programa feito em uma linguagem interativa. O ambiente está disponível para diversos sistemas operacionais, como Unix, Linux, MacOS X e Windows. Pequenas alterações (ou precauções) são necessárias no código para migrar de uma plataforma para outra. Outra característica interessante é sua integração com o software ENVI (EXELIS, 2012), que permite a visualização, exploração, análise e apresentação de dados nas áreas de Sensoriamento Remoto e SIG (Sistemas de Informações Geográficas).

Para o presente trabalho, a escolha da linguagem IDL deveu-se principalmente à tradição no uso dessa linguagem e do ENVI no grupo de pesquisa ao qual o trabalho está vinculado. Além de apresentar grande vantagem na manipulação de estruturas matriciais, a linguagem abstrai certos detalhes de implementação, tornando seu uso mais simples para pesquisadores de formações heterogêneas, como é o caso do grupo.

Além de avaliar a metodologia de análise de cenários proposta, a intenção foi produzir um código flexível, que possa ser alterado com relativa facilidade de acordo com a necessidade de pesquisas futuras. A liguagem também oferece facilidade de integração com programas codificados em C++ e Java, o que foi utilizado no Analisador de Cenários para a classificação pelo método SVM.

6.2 Estruturas de dados

Foram implementadas duas estruturas de dados principais: uma para representar o cenário e uma para a árvore de classes. Para ambos, foram utilizadas as características de orientação a objetos oferecidas pela linguagem IDL. O diagrama da Figura 6.1 mostra os atributos e métodos da classe *Cenario*.

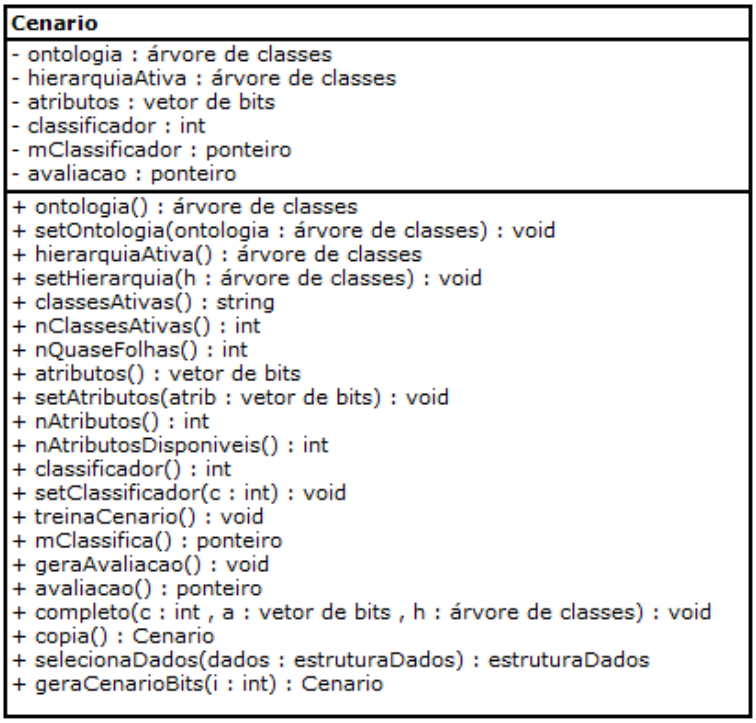

Figura 6.1 - Atributos e métodos da classe *Cenario*.

Os atributos *hierarquiaAtiva*, *atributos* e *classificador* são os atributos básicos da classe *Cenario* e correspondem às variáveis identificadas no problema multiobjetivo a ser resolvido. O método *geraCenarioBits* altera um desses três atributos básicos de acordo com uma posição na cadeia de bits utilizada como controle pelo otimizador.

O conjunto de atributos utilizados em um cenário de classificação pode ser repre-

sentado por um vetor de bits com o tamanho igual à quantidade de atributos disponíveis no problema. O valor de cada posição indica sua pertinência ao conjunto de um cenário específico. Por exemplo, se um cenário possui esse atributo igual a $[0, 1, 1, 0, 1]$, podemos concluir que o conjunto total de atributos disponíveis é $\Gamma = \{\gamma_1, \gamma_2, \gamma_3, \gamma_4, \gamma_5\}$ e o conjunto deste cenário em particular é $F = \{\gamma_2, \gamma_3, \gamma_5\}$. Na prática, foi utilizado o tipo *byte*, pois a linguagem IDL não possui o tipo básico *bit*.

O tipo de classificador utilizado é representado por um número inteiro que tem um papel seletor. Podem ser adicionados ao sistema tantos tipos de classificador quantos se queira. Para isso, é necessário apenas alterar três métodos: *treinaClassificador*, *geraAvaliacao* e *geraCenarioBits*. Os dois primeiros, *treinaClassificador* e *geraAvaliação*, chamam as rotinas, respectivamente, de treinamento e de classificação definidas externamente. Caso essa rotina externa tenha sido implementada especificamente para ser acoplada a esse sistema, é suficiente acrescentar uma linha a esses módulos, no comando seletor do classificador, adicionando a chamada do novo tipo. Mas para o caso de ser utilizado um classificador previamente existente ou que foi desenvolvido para outros tipos de dados, é necessário uma conversão nas entradas e saídas dos mesmos. As estruturas utilizadas são compatíveis com outros programas desenvolvidos pelo grupo de pesquisa.

O atributo *mClassificador* corresponde à saída do treinamento (*treinaClassificador*) e entrada da classificação no módulo *geraAvaliacao*. Ao se escolher a opção de classificar a imagem com um cenário específico, não é necessário realizar novamente o treinamento, pois o modelo obtido está armazenado neste atributo.

O atributo *ontologia* contém a árvore inicial fornecida pelo usuário, que não é alterada durante a execução do programa. Já no caso do atributo *hierarquiaAtiva*, modificações podem ser realizadas por métodos que ativam ou desativam classes, ou que renomeiam nós. Tanto a *ontologia* quanto a *hierarquiaAtiva* são representadas por uma árvore de classes, estrutura dinâmica obtida a partir de objetos da classe *Classe*, cujo diagrama de atributos e métodos é mostrado na Figura 6.2. Os métodos *nClassesAtivas* e *nQuaseFolhas* da classe *Cenario* são simples chamadas dos respectivos métodos de mesmo nome para seu atributo *hierarquiaAtiva*.

Todos os atributos da classe *Classe* são obtidos inicialmente a partir do arquivo com a árvore de classes fornecida pelo usuário, conforme explicado na Seção 6.3. Na *ontologia*, nenhum desses atributos será modificado ao longo da execução do programa. Já na *hierarquiaAtiva*, o atributo *nome* será alterado em todos os nós internos para

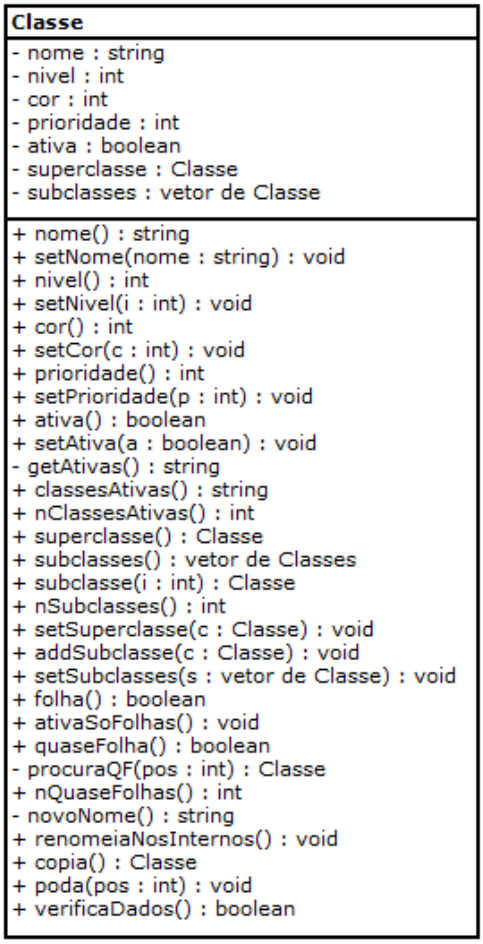

Figura 6.2 - Atributos e métodos da classe *Classe*.

uma string contendo todos os nomes de seus descendentes folhas. Essa operação é realizada pelo método *renomeiaNosInternos*, que utiliza *novoNome* de forma auxiliar. Da mesma forma, o atributo *ativa* será alterado pelos métodos *ativaSoFolhas*, que atribui o valor 1 a todas as folhas e 0 aos nós internos, e *poda*, método chamado pelo *geraCenarioBits* da classe *Cenario* e que desativa uma ramificação ativa, atribuindo 0 a um conjunto de classes ativas irmãs e 1 à sua superclasse. A operação de poda é realizada em nós da árvore denominados *quase-folhas*, que são classes cujas subclasses estão todas ativas. Não é permitido podar no nó raiz da árvore, pois esta operação deixaria o conjunto de classes ativas com um único elemento, o que impossibilita a classificação.

Além dos objetos, foi definida uma estrutura para armazenar os dados lidos no arquivo de amostras. A liguagem IDL oferece um recurso de variável compartilhada (*COMMON*), em que diversos módulos acessam o mesmo objeto na memória. Esse

recurso foi amplamente utilizado no sistema Analisador de Cenários, para evitar a alocação múltipla de espaço para o mesmo conjunto de dados, em especial os dados de treinamento e validação. A estrutura implementada armazena o número de canais presentes na amostra, a quantidade, o nome e o número de pontos de cada classe, e os dados amostrais em si.

Após a leitura do arquivo de amostras, um procedimento sorteia pontos do conjunto original para serem utilizados como pontos de treinamento, validação ou teste. Inicialmente uma quantidade média de pontos é calculada com base na quantidade de pontos de cada amostra. Essa quantidade padrão corresponderá ao total de pontos utilizados para cada classe. Aproximadamente metade dessa quantidade é sorteada de cada classe para compor o conjunto de treinamento, em um sorteio com reposição O mesmo procedimento é repetido para sortear 1/4 dos pontos para validação, e mais 1/4 para teste. Os pontos de treinamento são utilizados na obtenção do modelo de classificação. Os pontos de validação, para gerar a avaliação correspondente a cada cenário. E os pontos de teste são utilizados apenas ao final do processo, para avaliar o cenário escolhido para realizar a classificação da imagem.

6.3 Interface

Esta seção descreve como é feita a entrada e saída de dados do programa Analisador de Cenários. Para uma parte das entradas, existe uma interface gráfica em que o usuário escolhe as opções que deseja, digita os valores dos parâmetros e visualiza os resultados obtidos. Outra parte da interação com o programa é feita por meio de arquivos.

A tela inicial, mostrada na Figura 6.3, apresenta três botões para a seleção de arquivos. Inicialmente, os retângulos abaixo e à direita dos botões estão em branco. Devem ser fornecidos dois arquivos de texto: um com as amostras rotuladas das classes e outro com a estrutura da ontologia. Opcionalmente, pode-se escolher um nome para o novo arquivo de log, no qual serão gravadas as informações sobre a execução do programa.

O arquivo das amostras deve estar no formato exportado pelo software ENVI na opção *Output ROIs to ASCII*. O ENVI permite a seleção dos dados a serem gravados no arquivo texto e, para a compatibilidade com o Analisador de Cenários, deve-se escolher gravar apenas os valores das bandas, conforme mostra a Figura 6.4.

A Figura 6.5 traz uma imagem do aspecto do arquivo resultante, que deve ser gravado

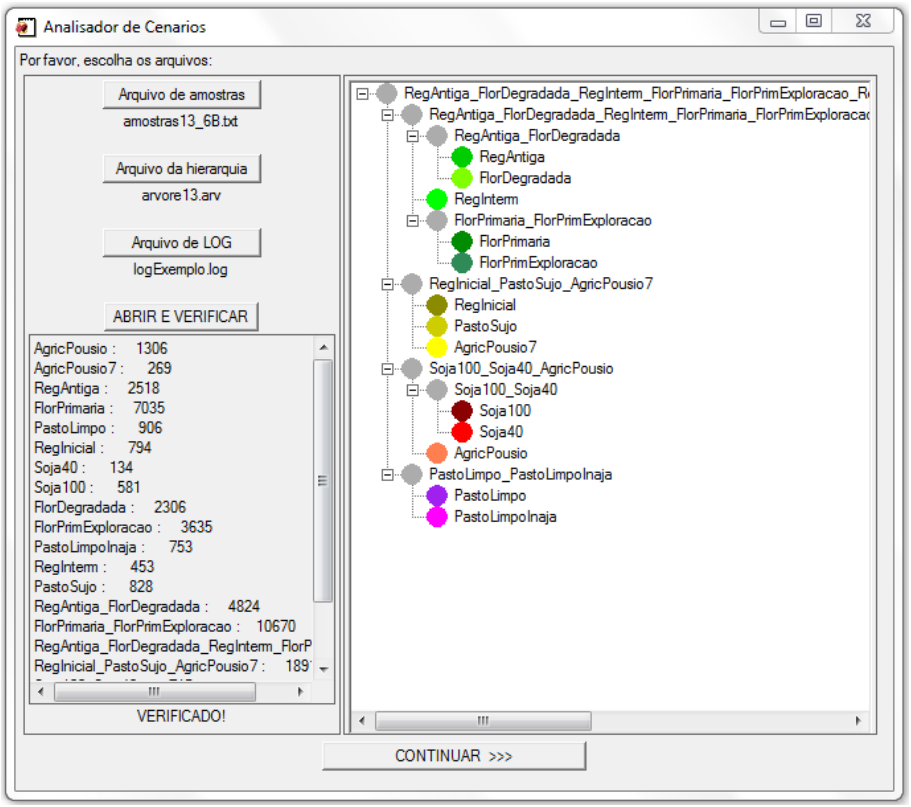

Figura 6.3 - Tela inicial do Analisador de Cenários.

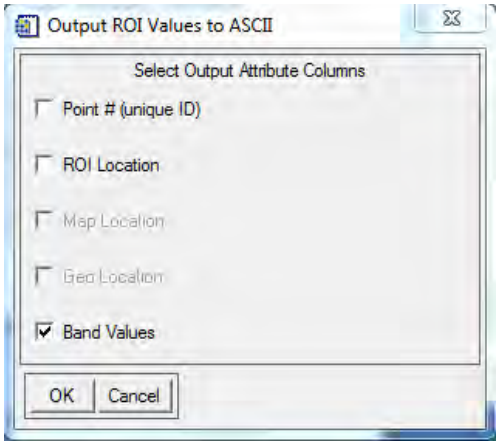

Figura 6.4 - Opções do ENVI para gravar amostras em formato ASCII.

com a extensão `.txt`. As três linhas iniciais consistem de um cabeçalho, do qual apenas a quantidade de classes é extraída. A seguir, os dados de cada amostra são apresentados em três linhas contendo o nome da classe, os valores RGB da cor que representa a classe na tela e a quantidade de pontos. Após a descrição de todas as classes, são listados os valores dos canais propriamente ditos, separando-se uma

classe da outra com uma linha em branco.

```
ENVI Output of ROIs (4.3) [Sun Sep 16 21:07:39 2012]
Number of ROIs: 13
File Dimension: 880 x 1900

ROI name: AgricPousio
ROI rgb value: {255, 127, 80}
ROI npts: 1306

ROI name: AgricPousio7
ROI rgb value: {255, 255, 0}
ROI npts: 269

ROI name: RegAntiga
ROI rgb value: {0, 205, 0}
ROI npts: 2518
  ◆
  ◆
  ◆
ROI name: PastoSujo
ROI rgb value: {205, 205, 0}
ROI npts: 828
  B1   B2   B3   B4   B5   B6
  27   38   75  114   74   69
  27   41   84  124   77   73
  24   41   75  119   73   69
  27   35   75  125   76   73
  27   38   71  109   74   69
  24   44   84  117   76   73
  24   41   81  122   76   69
  26   41   76  114   76   69
  27   41   81  125   79   73
  24   41   84  127   73   69
  29   41   86  117   77   76
  24   41   76  115   74   69
  ◆
  ◆
  ◆
  68  115  150  119   97  118
  66  103  150  122   95  115
  66  103  153  120   93  118
  68  109  159  124  100  122

  43   74  136  156   85  101
  43   77  131  156   83  101
  38   71  126  171   80  101
  45   74  128  172   83  108
```

Figura 6.5 - Trechos do arquivo de amostras gravado pelo ENVI.

O arquivo da árvore de classes também possui uma sintaxe bastante simples. A primeira classe a ser listada será a raiz da árvore, que não aparecerá na classificação, pois representa todo o domínio do problema, ou seja, tudo o que o usuário deseja identificar na cena a ser classificada. A seguir, as demais classes são listadas, precedidas por uma quantidade de símbolos "$>$" que representa o nível da classe na árvore. Por exemplo, na Figura 6.6, as classes *soja100* e *soja40* são subclasses de *soja*. Após o nome da classe, que deve ser o mesmo utilizado no arquivo das amostras para todas as folhas, seguem-se três valores representando, respectivamente, um índice de cor, a prioridade da classe e se ela está ativa ou inativa. O sistema utiliza a mesma tabela de cores do ENVI. O índice de cor, portanto, representa uma cor do software ENVI como consta no seu arquivo de configuração `colors50.txt`. O sistema utiliza a tabela de cores do arquivo de configuração `colors50.txt`do ENVI. O valor

seguinte, da prioridade da classe, é utilizado em uma média ponderada quando é escolhida uma medida por classe para avaliar a acurácia da classificação. É possível também otimizar ao mesmo tempo a acurácia global e uma acurácia específica de uma classe, colocando seu valor de prioridade igual a 1 e os das demais iguais a 0. E o último valor indica se no cenário inicial a classe deve estar ativa ou inativa. Na implementação atual, esse valor permanece armazenado na ontologia, mas é descartado na hierarquia ativa, cuja configuração inicial ativa todas as folhas e desativa os demais nós. Esse arquivo texto deve ser gravado com a extensão `.arv`.

```
universo
>Florestas;6;10;0
>>Antigas;12;10;0
>>>RegAntiga;21;10;1
>>>FlorDegradada;15;10;1
>>RegInterm;3;2;1
>>FlorestasPrimarias;29;10;0
>>>FlorPrimaria;22;10;1
>>>FlorPrimExploracao;9;10;1
>Veg1;26;10;0
>>RegInicial;28;10;1
>>PastoSujo;27;10;1
>>AgricPousio7;5;10;1
>Agricultura;8;10;0
>>Soja;13;10;0
>>>Soja100;19;10;1
>>>Soja40;2;10;1
>>AgricPousio;11;10;1
>PastosLimpos;8;1;0
>>PastoLimpo;10;10;1
>>PastoLimpoInaja;7;10;1
```

Figura 6.6 - Exemplo de arquivo de árvore de classes, que gera a árvore mostrada na Figura 6.3.

No arquivo de log serão gravadas informações durante a execução do programa, tais como os cenários gerados e o valor de suas funções objetivo, e também o conjunto e a fronteira de Pareto a cada alteração. Esse arquivo será gravado com a extensão `.log`.

Após a escolha dos três arquivos, deve-se clicar no botão “ABRIR E VERIFICAR” (Figura 6.3) para que as informações dos arquivos de amostras e da árvore sejam comparadas e verificadas. Caso os arquivos sejam compatíveis, a verificação é bem sucedida e o sistema mostra à direita uma representação gráfica da árvore lida e, à esquerda, a lista das amostras com as respectivas quantidades. A árvore já é mostrada na configuração que terá no primeiro cenário avaliado, ou seja, com os nós internos renomeados e apenas as folhas ativas. Os nós inativos são mostrados na cor cinza.

A segunda tela, que pode ser visualizada na Figura 6.7, permite ao usuário selecionar como ele deseja que seja o comportamento do analisador. É possível selecionar o que se deseja deixar variável, quais os objetivos a otimizar e o valor mínimo da acurácia para que uma solução seja considerada viável. Pode-se ainda configurar alguns parâmetros da otimização, detalhados na Seção 6.4, que trata também do processo de otimização que se segue à escolha da opção "CONTINUAR".

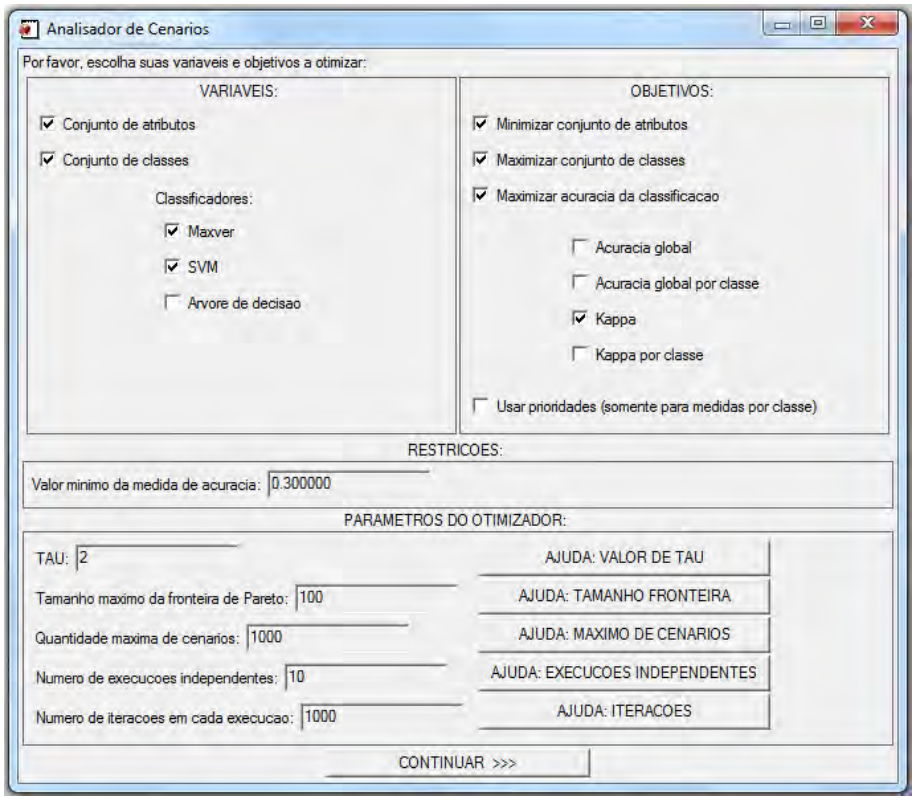

Figura 6.7 - Tela de opções do Analisador de Cenários.

Ao final do processo, é apresentada ao usuário uma tabela com o conjunto e a fronteira de Pareto, ou seja, as soluções Pareto-ótimas e sua avaliação pelas funções objetivo. A Figura 6.8 exibe um exemplo da saída do programa. O usuário pode então analisar as soluções encontradas e escolher a mais apropriada para sua aplicação. Caso queira ainda uma análise adicional, pode continuar para a segunda aba e escolher uma das opções disponíveis para reduzir a fronteira de Pareto, como mostra a Figura 6.9. Nessa tela ainda é possível carregar um arquivo de imagem e efetuar a classificação.

A redução da fronteira de Pareto realizada após o término do processo de otimiza-

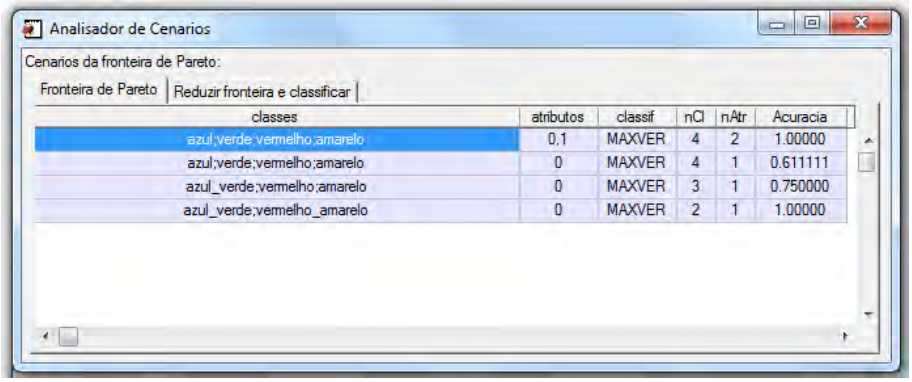

Figura 6.8 - Tela que mostra a fronteira de Pareto encontrada pelo sistema.

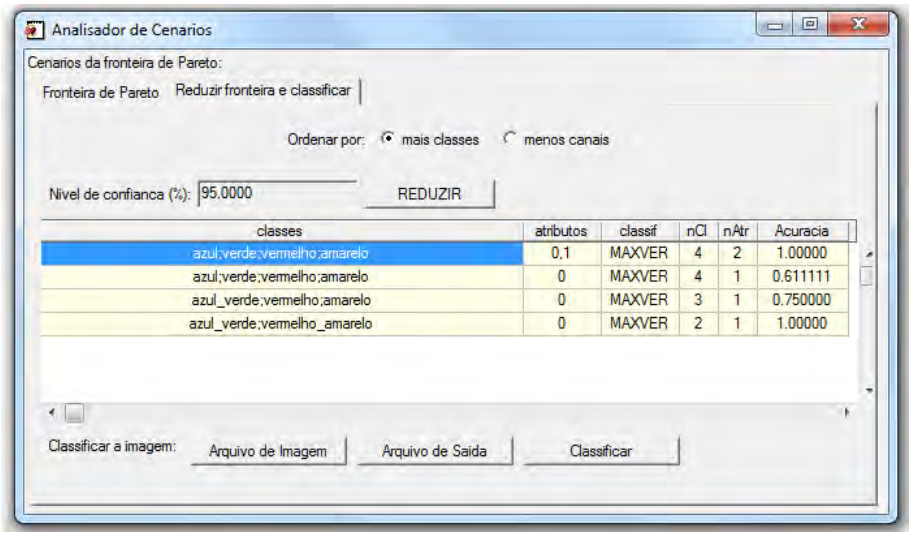

Figura 6.9 - Tela que permite a redução da fronteira de Pareto.

ção consiste na ordenação dos pontos da fronteira por uma das duas variáveis que fornecem diretamente uma avaliação da função objetivo: o conjunto de classes e o de atributos. A ordenação faz com que os pontos da fronteira sejam agrupados de acordo com o critério escolhido, e a ideia é reduzir a fronteira a apenas um ponto para cada um dos grupos, progressivamente. A redução se dá através de um teste estatístico Z que avalia a diferença entre os valores de acurácia dos dois pontos. O conjunto é reduzido cada vez mais, à medida em que se aumenta o valor do nível de confiança do teste. Na versão atual do sistema, somente o índice κ está disponível como opção para redução da fronteira.

6.4 Algoritmos

O programa principal do Analisador de Cenários consiste em um otimizador, implementado no módulo *encontraFronteira*. Essa implementação foi baseada no algoritmo M-GEO, algoritmo evolutivo inpirado na Teoria da Criticalidade Auto-organizada que foi apresentado na Seção 2.4. Algumas alterações foram necessárias no algoritmo devido à natureza dos dados e do problema a ser resolvido. O Algoritmo 2 apresenta a versão que foi efetivamente implementada no Analisador de Cenários.

Algoritmo 2: Adaptação do algoritmo M_GEO.

```
1: inicialize uma string de L bits com a representação de um cenário completo
2:for i ← 1 to número de execuções independentes do
    3: while não alcançou critério de parada do
        4: for i ← 1 to L do
            5: if o cenário não foi gerado em outra iteração then
                6: mude o valor do bit i, gerando uma nova string x_i;
                7: calcule o valor de todas as funções objetivo
                   f(x_i) = [f_1(x_i), ··· , f_M(x_i)];
                8: teste e salve o conjunto e a fronteira de Pareto;
            end if
        end for
        9: escolha aleatoriamente uma função objetivo como f_Ic
        10: foreach string x_i do
            11: atribua um valor de adaptação proporcional ao ganho ou perda
                que f_Ic(x_i) tem se o bit muda, comparado ao melhor valor
                de f_Ic encontrado até o momento;
            12: ordene os bits de acordo com seu valor de adaptação, sendo 1
                a posição do pior valor;
            13: mude o valor de um bit da população com probabilidade P_k ∝ k^-τ,
                k ∈ {1, 2, ··· , L}, k é a posição do bit na ordenação;
        end foreach
    end while
    14: inicialize aleatoriamente uma string com L bits para as N variáveis de
    projeto;
end for
15:return conjunto e fronteira de Pareto;
```

A primeira alteração foi na representação da população. Foi usado um vetor de *Bytes* que, na maior parte, é utilizado como um vetor de bits, ou seja, as posições só podem assumir os valores 0 ou 1. A exceção é na posição que representa o tipo do classificador, pois, para $|\Psi|$ classificadores disponíveis, essa posição pode assumir

valores de 0 a $|\Psi| - 1$. Na Figura 6.10, o classificador está associado com o bit da posição n. Os n bits de b_0 a b_{n-1} representam o conjunto de atributos e os últimos m bits representam as alterações possíveis no conjunto de classes para uma determinada iteração do algoritmo. O valor de m pode variar a cada iteração, pois depende do conjunto utilizado no cenário inicial, redefinido em cada passagem no passo 11.

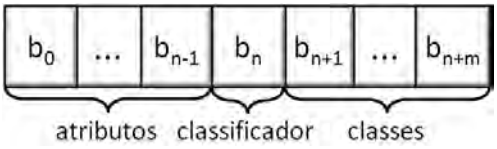

Figura 6.10 - Vetor de bits que representa uma população.

O vetor de bits da Figura 6.10 representa o cenário a ser analisado a cada iteração, e sua modificação reflete as alterações realizadas nos cenários durante a execução da otimização. Como um dos objetivos considerados nesse trabalho é a acurácia de classificação, cada avaliação das funções objetivo (passo 7 do Algoritmo 2) consiste no treinamento do classificador associado ao cenário e sua avaliação através da classificação de um conjunto de pontos. Sendo assim, a população efetivamente avaliada é uma lista de L (ou menos) cenários gerada pela modificação de cada posição do vetor de bits. Podem ser gerados menos de L cenários em dois casos: o cenário gerado pode estar fora do espaço viável ou pode já ter sido avaliado anteriormente. Em qualquer dos dois casos, a modificação não é realizada, e as funções objetivo não são avaliadas. Para o conjunto de Pareto também é mantida uma lista de cenários.

Uma outra modificação foi na inicialização do processo. Enquanto no M-GEO são realizadas diversas execuções independentes, todas com inicialização aleatória da população, no Analisador de Cenários a primeira execução é iniciada com o cenário completo, ou seja, com todos os atributos disponíveis, usando todas as classes folha da hierarquia e o classificador Maxver que, além de possuir uma avaliação mais rápida, apresentou valores de acurácia superiores nos experimentos realizados. As execuções seguintes são iniciadas com uma população aleatória.

Os parâmetros de entrada do algoritmo são, além do parâmetro de aleatoriedade τ, o número de execuções independentes do algoritmo (*nExec*), o número de iterações a executar em cada iteração (*nIter*), a quantidade máxima de pontos na fronteira de Pareto (*maxFronteira*) e a quantidade máxima de cenários que devem ser analisados (*maxCen*). Os dois últimos parâmetros, *maxFronteira* e *maxCen*, são utilizados para

definir o tamanho da lista de cenários a ser criada durante a análise. Assim como o parâmetro *nIter*, possuem o papel de limitar a busca a um tempo razoável, e fazem parte do critério de parada que controla as iterações do algoritmo. O parâmtro *nExec* indica quantas vezes o cenário inicial de cada iteração deve ser aleatoriamente gerado. Essa reinicialização é uma forma de explorar diferentes regiões do espaço de busca, favorecendo que se atinja todos os pontos da fronteira de Pareto. Sousa (2002) e Galski (2006) sugerem que, para problemas mais complexos, que exigem a representação da população com maior o vetor de bits, um número maior de execuções independentes é indicado. Em todas as gerações de números aleatórios do algoritmo, é utilizada a distribuição uniforme.

Os algoritmos de classificação utilizados no Analisador foram previamente implementados para outros sistemas do grupo de pesquisa. A implementação da árvore de decisão está descrita em (MEDEIROS et al., 2011) e não foi feita nenhuma modificação em seu código. No caso do SVM, foi utilizada a implementação $SVM^{Multiclass}$, como descrita em (CRAMMER; SINGER, 2001).

7 EXPERIMENTOS E RESULTADOS

7.1 Conjunto controlado

No desenvolvimento de um sistema computacional, sua aplicação a um conjunto de dados controlado permite que várias de suas características sejam verificadas e avaliadas. Para o Analisador de Cenários, foi utilizada para este fim uma imagem sintética, composta por quatro classes com a mesma quantidade de pontos.

7.1.1 Dados do problema

A Figura 7.1 mostra a composição colorida dos três canais, R, G e B, da imagem `sintetica.tif`, que possui 512 linhas e 512 colunas.

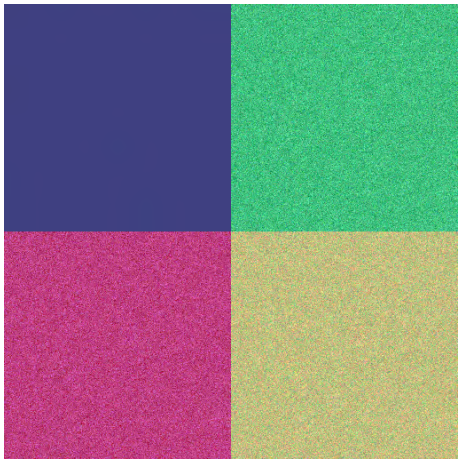

Figura 7.1 - Imagem sintética.

Os valores RGB iniciais de cada classe da imagem podem ser vistos na Tabela 7.1, que mostra também o valor do desvio padrão do ruído Gaussiano de média 0 adicionado a cada canal.

Tabela 7.1 - Valores dos canais da imagem sintética.

	Cor Original			**σ do ruído**		
	R	G	B	R	G	B
Azul	64	64	128	10	10	10
Verde	64	192	128	20	20	20
Vermelho	192	64	128	10	20	30
Amarelo	192	192	128	30	20	10

A visualização de cada um dos três canais é apresentada na Figura 7.2. Pode-se observar que, nos canais 1 e 2 (R e G), a imagem apresenta duas regiões bem caracterizadas. Já no terceiro canal (B), não é possível diferenciar as classes apenas pelas diferenças individuais nos píxeis. Por esse motivo, é possível separar as 4 classes da Tabela 7.1 com o uso apenas dos dois primeiros canais. O terceiro canal é irrelevante na tarefa de classificação por píxel.

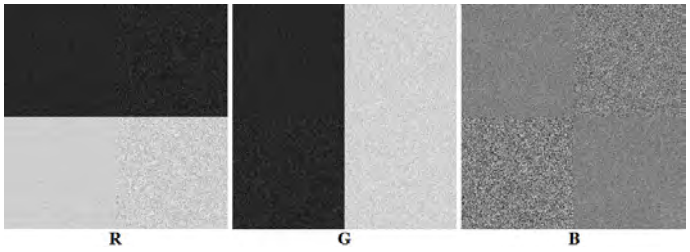

Figura 7.2 - Três canais de dados da imagem sintética.

O papel dos canais na classificação também pode ser observado na visualização do espaço de atributos, apresentada na Figura 7.3. Nessa visualização também é possível observar que os pontos de cada classe encontram-se agrupados de forma bem definida. A projeção dos pontos nos dois primeiros canais, mostrada na Figura 7.4, apresenta os quatro grupos distintos.

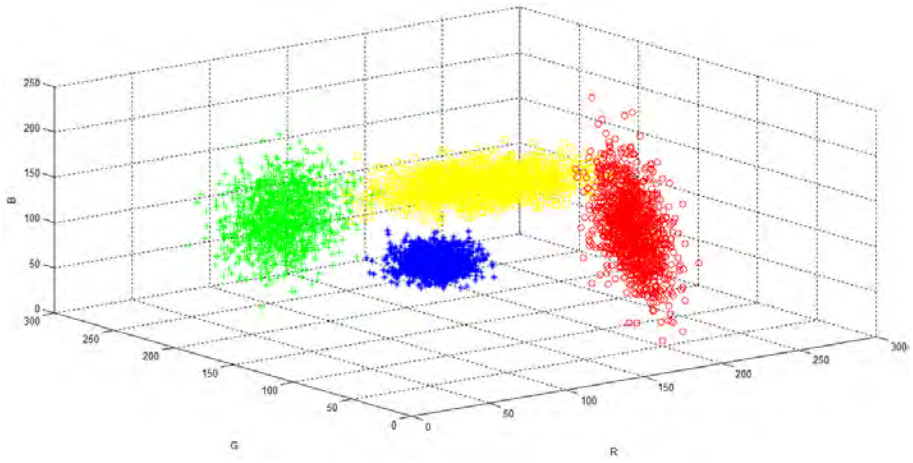

Figura 7.3 - Espaço de atributos da imagem sintética.

Para avaliar o analisador de cenários neste conjunto de dados controlado, foram criadas duas hierarquias para as classes, com base no seu comportamento espectral:

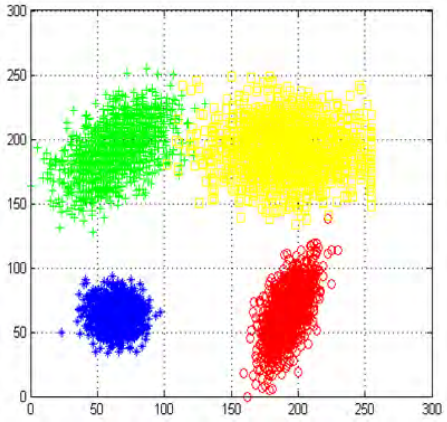

Figura 7.4 - Projeção do espaço de atributos da imagem sintética nos dois primeiros canais.

uma que inclui a partição das classes azul-vermelho, verde-amarelo e a outra que inclui azul-verde,vermelho-amarelo, como mostra a Figura 7.5.

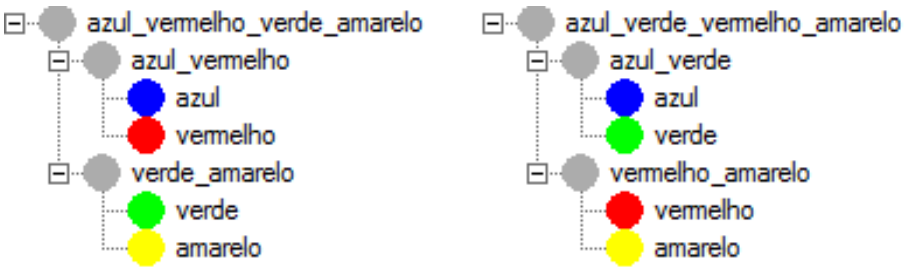

Figura 7.5 - Hierarquias de classe utilizadas nos testes com a imagem sintética.

No arquivo de hierarquia fornecido, a classe “azul_verde”, por exemplo, é chamada de “esquerda”. Esse nome não aparece na tela que mostra a hierarquia devido ao uso da função renomeadora, que altera o nome das superclasses para a concatenação dos nomes de suas subclasses. A ordem em que o nome aparece na concatenação é a ordem inicial fornecida pelo usuário no arquivo da hierarquia.

As duas árvores da Figura 7.5 possuem exatamente dois pontos de poda ($m = 2$) e estão disponíveis três canais de dados, R, G e B, ou seja, $n = 3$. Portanto, a quantidade máxima de bits L necessária para representar cada cenário para esse conjunto de dados é $n + 1 + m$, sendo $n = 3$ e $m = 2$, totalizando 6 bits. Os testes foram realizados com apenas dois classificadores, o Maxver e o SVM. A quantidade total de cenários que podem ser gerados com este vetor é de $2^6 = 64$. Porém, para 8 destes 64 cenários, o conjunto de atributos é vazio, o que os torna inviáveis. Portanto, somente

os 56 cenários viáveis são avaliados. A fórmula geral para calcular a quantidade de cenários possíveis em um determinado problema é $(2^n - 1) * n_{cla} * 2^m$, em que n é o número de atributos disponíveis, n_{cla} é a quantidade de classificadores possíveis e m, o número de locais de poda existentes na árvore de classes. O experimento com a segunda árvore da Figura 7.5 foi estudado com mais detalhes e será descrito a seguir.

7.1.2 Solução

Para confrontar a resposta do sistema com a verdadeira solução do problema, foram gerados todos os 56 cenários viáveis e foi feita sua avaliação fora do sistema analisador. O gráfico da Figura 7.6 mostra a avaliação das três funções objetivo para todos esses cenários. São distinguidos apenas 36 pontos porque em muitos casos houve sobreposição, como por exemplo no uso de apenas um dos dois primeiros canais ou na poda em um dos dois ramos da árvore, casos em que os cenários são diferentes mas possuem a mesma avaliação da função objetivo (tamanho do conjunto). A seta indica o ponto que seria a solução ótima do problema, caso existisse. O tamanho do conjunto de atributos foi expresso com sinal negativo para caracterizar o problema como sendo de maximização para todas as funções, facilitando a visualização.

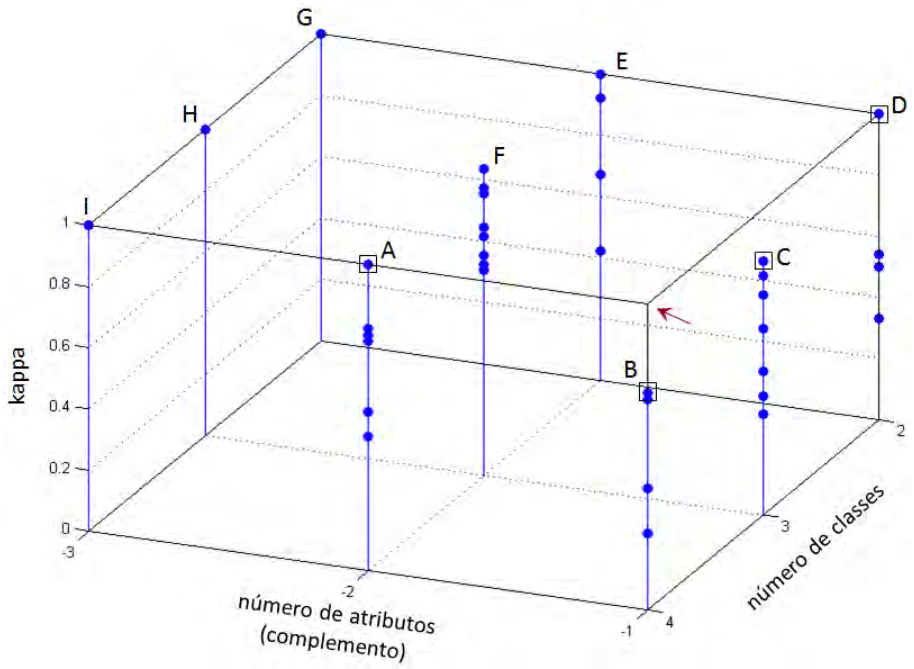

Figura 7.6 - Funções objetivo de todos os cenários da imagem sintética.

Os pontos pertencentes à fronteira de Pareto estão assinalados no gráfico como A, B, C e D. Os valores de suas funções objetivo e de outros pontos de destaque no gráfico estão listados na Tabela 7.2.

Tabela 7.2 - Valores das funções objetivo para os cenários da imagem sintética assinalados na Figura 7.6.

Ponto	$\|F\|$	$\|\Omega\|$	κ
A	2	4	1
B	1	4	$0,71$
C	1	3	$0,83$
D	1	2	1
E	2	2	1
F	2	3	1
G	3	2	1
H	3	3	1
I	3	4	1

Um ponto viável no espaço de busca domina outros se ele melhora pelo menos um dos objetivos sem piorar qualquer dos demais. Cada conjunto de pontos que aparece na mesma linha vertical no gráfico da Figura 7.6 possui o mesmo número de atributos e o mesmo número de classes, variando apenas o valor do índice kappa obtido na classificação.

Para cada um desse conjuntos, portanto, o ponto dominante é o que possui maior valor de kappa. Ao compararmos o ponto F com, por exemplo, o ponto C, verificamos que não existe dominância entre esses dois pontos. Enquanto o ponto C apresenta melhor valor para o número de atributos, o ponto F possui maior valor para kappa. Entretanto, F não faz parte da fronteira de Pareto, pois é dominado pelo ponto A, que possui o mesmo número de atributos e kappa, porém melhora a quantidade de classes.

Pode-se observar que o ponto A domina todos os pontos E, F, G, H e I. Porém, ainda que tenha melhor (ou igual) valor de kappa que B, C e D, sua quantidade de atributos é pior. Entre os quatro pontos A, B, C e D, não se pode dizer que um seja melhor que qualquer dos demais em todos os objetivos.

7.1.3 Resultados obtidos pelo Analisador de Cenários

Foram realizadas várias execuções do sistema, com variação de seus parâmetros. A variação recomendada por Sousa (2002) para o parâmetro τ é de valores entre $0,5$ e 8. Para avaliar a influência deste parâmetro no desempenho do sistema, foram testados quatro valores neste intervalo: $0,5$; 3; $5,5$ e 8. Já a quantidade de execuções independentes, que indica quantas vezes o cenário será reinicializado, assumiu os valores 5, 15 e 25. A outra variação foi no número de iterações em cada execução: 20, 60, 100 e 140.

Conforme discutido na Seção 2.4 quando foi introduzido o Algoritmo M-GEO, o parâmetro τ controla a aleatoriedade da busca. Para valores mais altos, ela é mais determinística e, para valores próximos de 0, torna-se mais aleatória, ou seja, guia a procura para direções com distribuição mais próxima à distribuição uniforme. O valor $0,5$ faz com que o programa seja mais aleatório, nem sempre caminhando na direção que melhora as funções objetivo. Isso faz com que o otimizador não fique preso em regiões de mínimos locais, podendo atingir pontos que estão mais distantes do cenário atual. Para valores mais altos de τ, a busca se torna mais determinística, o que, nesse caso específico, funcionou melhor.

O número de iterações em cada execução indica quantas vezes o otimizador vai tentar buscar um cenário melhor a partir da inicialização realizada antes de cada execução. É possível que nem todos os cenários gerados a partir do inicial sejam avaliados, pois pode-se gerar um cenário fora da região viável ou que já tenha sido testado anteriormente. Durante essa fase, a busca ocorre a partir do ponto gerado no fim da iteração anterior, no passo 13 do Algoritmo 2 (Página 61).

Já o parâmetro *nExec*, ou número de execuções independentes, define quantas vezes o processo será repetido com inicializações diferentes do cenário. Isso faz com que, em certos momentos, a busca "pule" para regiões diferentes do espaço de soluções, o que também ajuda a evitar o aprisionamento em mínimos locais, favorecendo a exploração de todo o espaço de busca.

Em todas as execuções realizadas, a fronteira de Pareto encontrada sempre apresentava o ponto A da Figura 7.6 e, na maioria dos casos, o ponto D. Devido à inicialização do processo com o cenário completo (ponto I), o ponto A é encontrado logo na primeira iteração. Por isso, mesmo com parâmetros extremos, como $\tau = 0,5$ e $nExec = 1$, esses dois pontos foram localizados na maioria das vezes.

A influência dos parâmetros pode ser verificada na obtenção dos demais pontos. Em alguns casos, o sistema encontrou apenas os pontos A e D, em outros, encontrou um dos pontos B ou C e, em outros ainda, encontrou pontos próximos a B ou C, porém com valores inferiores de kappa. É interessante ressaltar que, devido à natureza da imagem utilizada, os resultados com kappa igual a 1 podem facilmente ser reproduzidos. Já os outros valores citados podem sofrer pequenas variações de acordo com as amostras fornecidas como entrada e o modo como foi realizado seu sorteio.

Os experimentos relatados nesta seção usaram o mesmo conjunto de dados em todas as execuções. Para o valor baixo de τ, a dependência no caráter aleatório se evidencia, pois os resultados foram bastante variados. Em média, não é um valor indicado para ser adotado neste problema. O valor que gerou o melhor resultado foi $\tau = 5,5$.

Devido à forma de geração da imagem, com distribuição gaussiana dos pontos nas classes, o classificador Maxver foi melhor que o SVM em todas as execuções realizadas.

A análise final dos pontos da fronteira através do teste estatístico Z pode ser realizada de duas formas: pelo número de classes e pelo número de atributos. Porém, para este exemplo particular, a fronteira já é bastante reduzida e essa opção não é muito interessante. Na ordenação pelo número de classes apenas o valor 4 se repete. Como a acurácia de um dos cenários é igual a 1, o teste só dará equivalência entre os valores para o valor de confiança 100%. Já para a ordenação pelo número de atributos, com nível de confiança de 90%, o teste considera que não há diferença significativa entre os valores de kappa, podendo ser adotada qualquer uma das soluções de forma equivalente.

7.2 Conjunto real

Foram realizados alguns experimentos com o objetivo de avaliar o sistema desenvolvido aplicado a dados de problemas reais de sensoriamento remoto. A imagem utilizada nos experimentos é do sensor TM (*Thematic Mapper*) do satélite Landsat-5, adquirida em 29 de junho de 2010. Foram utilizadas as bandas 1, 2, 3, 4, 5 e 7. A imagem possui resolução espacial de 30 m e resolução radiométrica de 8 bits. A área de estudo está situada no estado do Pará, município de Belterra. A Figura 7.7 mostra a localização da área de estudo e a composição das bandas 5 (R), 4 (G) e 3 (B) da imagem TM.

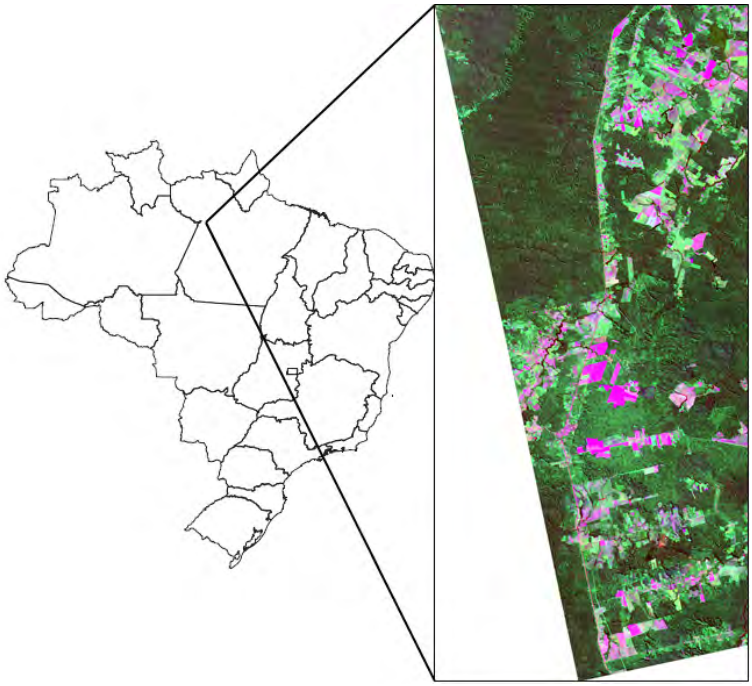

Figura 7.7 - Localização da área de estudo e composição 543 (RGB) da imagem TM.

Os dados de referência são provenientes de um trabalho de campo realizado no mesmo período. A Figura 7.8 mostra a hierarquia de classes utilizada. A descrição destas classes é apresentada na Tabela 7.3. Pode ser observado que a hierarquia possui sete pontos em que é possível realizar a poda.

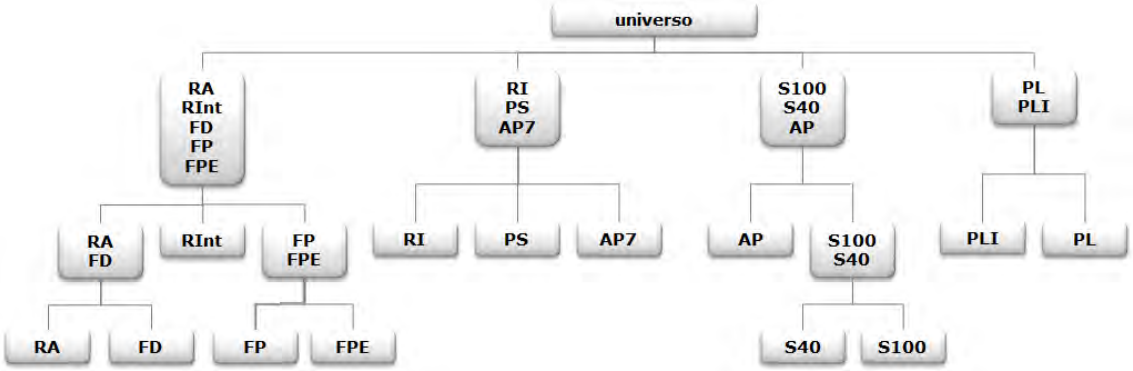

Figura 7.8 - Hierarquia de classes utilizada nos experimentos.

As amostras aferidas em campo foram assinaladas em uma imagem previamente segmentada. O processo completo de obtenção dos pontos rotulados foi descrito por Pereira (2012). A quantidade de amostras de cada classe folha pode ser vista na

Tabela 7.3.

Tabela 7.3 - Tamanho das amostras utilizadas.

Classe	Descrição	Píxeis
AP	Áreas agrícolas em pousio	1306
AP7	Áreas agrícolas com pousio de 7 a 24 meses	269
RA	Floresta secundária com mais de 20 anos de regeneração	2518
FP	Floresta primária	7035
PL	Pastagens	906
RI	Floresta secundária com menos de 8 anos de regeneração	794
S40	Soja com 40 dias da semeadora	134
S100	Soja após 100 dias da semeadura	581
FD	Florestas que sofreram queimadas	2306
FPE	Floresta com desmatamento seletivo	3635
PLI	Pastagens com presença de babaçu e inajá	753
RInt	Floresta Secundária com 8 a 20 anos de regeneração	453
PS	Pastagem com presença de árvores e arbustos	828

Foram realizados diversos experimentos com esse conjunto de dados, variando-se os parâmetros do algoritmo. Em todos os casos, foi utilizado o índice de concordância kappa como medida de acurácia e foram usadas como váriáveis de projeto os três componentes básicos do cenário, ou seja, conjunto de atributos, conjunto de classes e o classificador utilizado, entre Maxver, SVM e ODT. O classificador Maxver obteve maior acurácia que o SVM e o ODT na maioria dos casos analisados, provavelmente devido à natureza dos dados óticos.

Os conjuntos de classes gerados pelo algoritmo são apresentados na Tabela 7.4. Os conjuntos foram denominados como Ω_i, com $i \in \{A, B, C, D, E, F, G, H, I\}$. A coluna $|\Omega|$ mostra o tamanho do conjunto utilizado.

A fronteira de Pareto encontrada com maior número de pontos é apresentada na Tabela 7.5. Nesta Tabela, cada linha corresponde a um cenário encontrado. A primeira coluna mostra as bandas selecionadas pelo otimizador e a segunda apresenta o nome do conjunto de classes gerado (ver Tabela 7.4), constituindo as variáveis de projeto. As últimas três colunas são os valores das funções objetivo, ou seja, o número de atributos $|F|$, o número de classes $|\Omega|$ e o valor do índice κ obtido na classificação dos pontos de validação. Como pode ser observado, de modo geral a imagem não foi considerada adequada para identificar esse conjunto de classes em nenhum dos cenários analisados, sendo até mesmo insuficiente para discriminar satisfatoriamente

Tabela 7.4 - Conjuntos de classes gerados pelo sistema.

i	$\|\Omega\|$	Classes
A	13	RA FD RInt FP FPE RI PS AP7 S100 S40 AP PL PLI
B	12	RA FD RInt FP-FPE RI PS AP7 S100 S40 AP PL PLI
C	11	RA FD RInt FP FPE RI-PS-AP7 S100 S40 AP PL PLI
D	10	RA FD RInt FP-FPE RI-PS-AP7 S100 S40 AP PL PLI
E	9	RA FD RInt FP FPE RI-PS-AP7 S100-S40-AP PL PLI
F	9	RA-FD-RInt-FP-FPE RI PS AP7 S100 S40 AP PL PLI
G	7	RA-FD-RInt-FP-FPE RI-PS-AP7 S100 S40 AP PL PLI
H	6	RA-FD-RInt-FP-FPE RI-PS-AP7 S100 S40 AP PL-PLI
I	4	RA-FD-RInt-FP-FPE RI-PS-AP7 S100-S40-AP PL-PLI

as quatro classes iniciais da hierarquia. Nessa execução, a alteração do conjunto das classes foi bastante explorada, porém foi encontrado apenas um ponto que utiliza somente um atributo na classificação.

Tabela 7.5 - Maior fronteira de Pareto encontrada.

F	Ω	$\|F\|$	$\|\Omega\|$	κ
$B1, B2, B3, B4, B5, B7$	Ω_A	6	13	0,5230
$B1, B3, B4, B5, B7$	Ω_A	5	13	0,5155
$B1, B2, B3, B4, B5, B7$	Ω_B	6	12	0,5596
$B1, B2, B3, B4, B5$	Ω_C	5	11	0,5343
$B1, B2, B3, B4, B5, B7$	Ω_D	6	10	0,5899
$B1, B4, B5, B7$	Ω_C	4	11	0,5226
$B1, B3, B4, B5, B7$	Ω_D	5	10	0,5831
$B1, B4, B5, B7$	Ω_D	4	10	0,5719
$B1, B2, B3, B4, B7$	Ω_G	5	7	0,6784
$B1, B3, B4, B5, B7$	Ω_F	5	9	0,6288
$B1, B4, B7$	Ω_D	3	10	0,5616
$B1, B4, B5, B7$	Ω_G	4	7	0,6543
$B1, B2, B4, B7$	Ω_F	4	9	0,6081
$B1, B2$	Ω_D	2	10	0,5446
$B1, B2, B7$	Ω_G	3	7	0,6247
$B1, B2, B3$	Ω_E	3	9	0,5985
$B3$	Ω_D	1	10	0,3599
$B1, B3$	Ω_G	2	7	0,5654
$B1, B2, B4, B7$	Ω_H	4	6	0,6993
$B1, B2, B3$	Ω_H	3	6	0,6809
$B1, B3, B4, B5, B7$	Ω_H	5	6	0,7515
$B1, B2, B4, B5$	Ω_I	4	4	0,7300

Os 22 pontos da Tabela 7.5 foram encontrados com a realização de 15 execuções independentes de 50 iterações, com valor de τ igual a $0,5$. Para este conjunto de dados, utilizar uma busca mais aleatória favoreceu a avaliação de pontos mais distantes do cenário inicial, proporcionando uma cobertura maior da fronteira de Pareto. O pior resultado foi com 5 execuções independentes de 50 iterações, com τ igual a 8, em que apenas 7 pontos foram localizados, sendo todos com 12 ou 13 classes. Devido à interrupção precoce da execução pelo critério de parada *número de iterações*, uma boa parte do espaço de busca deixou de ser explorada.

Para o conjunto de dados real, mostrou-se bastante útil a ferramenta de reduzir a fronteira através do teste estatístico proposto, pois a quantidade de cenários na fronteira foi bem maior. Através desta metodologia, o usuário pode definir um critério para a escolha do cenário a ser implementado, e reduzir a fronteira até o tamanho que achar adequado para experimentação.

A Tabela 7.6 mostra o resultado de outra execução, com τ igual a $5,5$ e 100 iterações em 15 execuções independentes. Para esses valores de parâmetros, o algoritmo não explorou diferentes configurações do conjunto de classes, avaliando apenas os conjuntos com no mínimo sete classes. Por esse motivo, os valores de kappa apresentados são menores que os da Tabela 7.5. Os valores são mostrados na ordem do primeiro critério oferecido na interface, ou seja, ordem descendente do número de classes. A fronteira foi reduzida progressivamente com o aumento do nível de confiança do teste estatístico, respectivamente de 70%, 80% e 90%. Na Tabela 7.6, cada ponto está representado por seu valor de kappa. Os pontos eliminados foram substituídos por um $\times$. A última linha mostra a quantidade de pontos em cada situação.

Já na Tabela 7.7, a mesma execução foi reduzida pelo conjunto dos atributos. São mostrados os valores dos níveis de confiança que ocasionaram a redução da fronteira. Em todos os testes realizados, o tamanho da fronteira pode ser reduzido com níveis de confiança mais baixos quando se utiliza a ordem de classes. Nessa situação, os valores de kappa comparados estão mais próximos entre si do que quando são agrupados pelo número de atributos.

A imagem original foi classificada utilizando alguns dos cenários analisados pelo sistema. A Figura 7.9 mostra o resultado de três classificações, cada uma realizada com um cenário diferente. Na 7.9a, foram utilizadas todas as 13 classes e os 6 canais disponíveis. Já na Figura 7.9b, foi utilizado o conjunto Ω_F com 9 classes e os 6 canais; e na 7.9c, as 7 classes do conjunto Ω_G foram discriminadas com o uso de apenas duas bandas, a B1 e a B4.

Tabela 7.6 - Redução da fronteira de Pareto usando o tamanho do conjunto de classes como índice de ordenação.

			Nível de confiança		
$\|F\|$	$\|\Omega\|$	κ	70%	80%	90%
6	13	0,5230	0,5230	0,5230	0,5230
4	13	0,5086	0,5086	0,5086	×
6	12	0,5596	0,5596	0,5596	0,5596
5	12	0,5540	×	×	×
4	11	0,5226	0,5226	0,5226	0,5226
3	11	0,5119	0,5119	×	×
2	11	0,4972	0,4972	0,4972	×
1	11	0,3279	0,3279	0,3279	0,3279
6	10	0,5899	0,5899	0,5899	0,5899
5	10	0,5831	×	×	×
4	10	0,5719	0,5719	0,5719	×
3	10	0,5616	0,5616	×	×
1	10	0,3599	0,3599	0,3599	0,3599
2	7	0,5807	0,5807	0,5807	0,5807
Total		15	13	11	7

Tabela 7.7 - Redução da fronteira de Pareto usando o tamanho do conjunto de atributos como índice de ordenação.

			Nível de confiança		
$\|F\|$	$\|\Omega\|$	κ	90%	99%	99,9%
1	10	0,3599	0,3599	0,3599	0,3599
1	11	0,3279	0,3279	×	×
2	7	0,5807	0,5807	0,5807	0,5807
2	11	0,4972	0,4972	0,4972	0,4972
3	10	0,5616	0,5616	0,5616	0,5616
3	11	0,5119	0,5119	0,5119	0,5119
4	10	0,5719	0,5719	0,5719	0,5719
4	11	0,5226	0,5226	0,5226	0,5226
4	13	0,5086	×	×	×
5	10	0,5831	0,5831	0,5831	0,5831
5	12	0,5540	0,5540	×	×
6	10	0,5899	0,5899	0,5899	0,5899
6	12	0,5596	0,5596	0,5596	×
6	13	0,5230	0,5230	0,5230	0,5230
Total		15	13	12	10

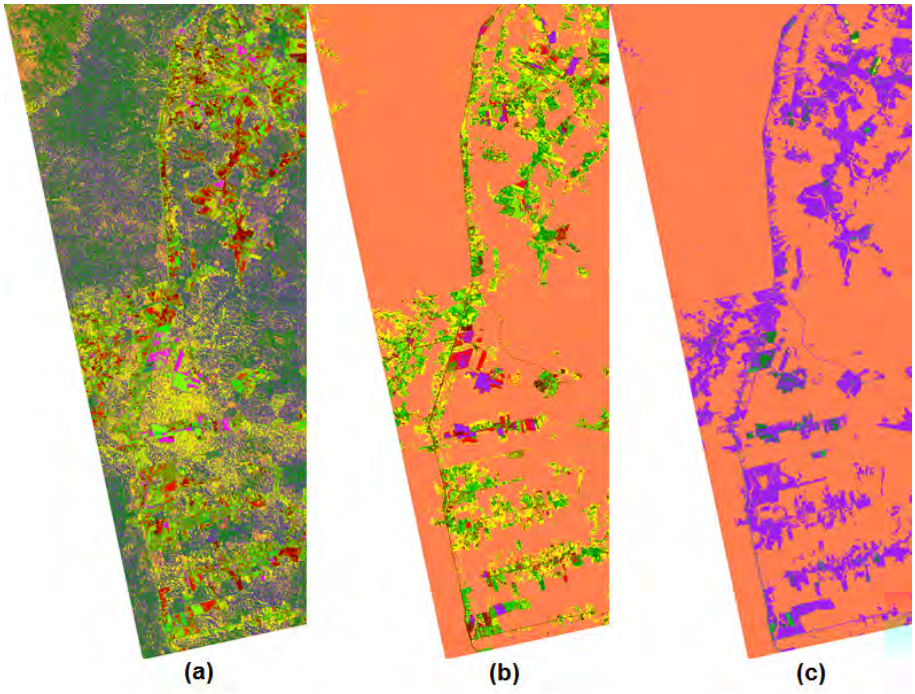

Figura 7.9 - Imagens classificadas para (Ω,F): (a) $(\Omega_A,\{B1, B2, B3, B4, B5, B7\})$; (b) $(\Omega_F,\{B1, B2, B3, B4, B5, B7\})$; (c) $(\Omega_G,\{B1, B4\})$.

A Figura 7.10 mostra detalhes das três classificações mostradas na Figura 7.9. Pode-se observar em 7.10a o excesso de píxeis isolados, devido ao grande número de classes, que aumenta a sua confusão. Já a 7.10c mostra uma imagem mais definida. A confusão entre as classes é menor, mesmo usando apenas dois canais na classificação.

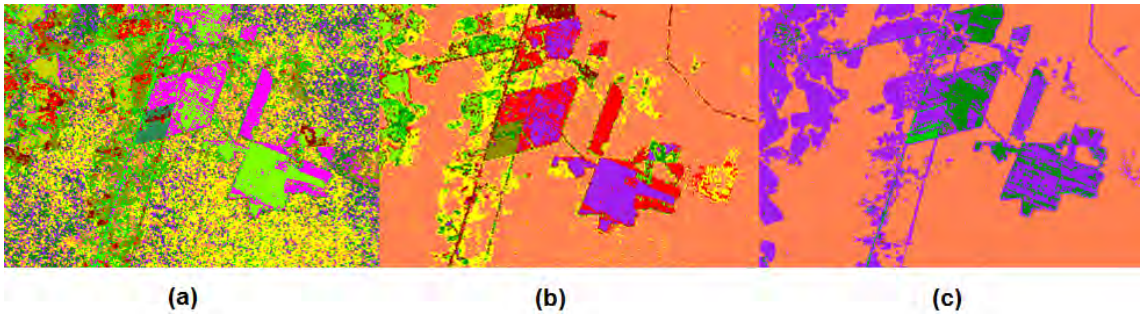

Figura 7.10 - Detalhes das classificações da Figura 7.9.

Outra comparação pode ser feita através da análise das classificações mostradas na Figura 7.11. As imagens mostradas em 7.11 foram obtidas pela classificação da imagem original utilizando, respectivamente, os conjuntos de classes Ω_H, Ω_H e Ω_G. Foram utilizados conjuntos com todos os 6 atributos (7.11a), com 4 atributos (7.11b)

e com apenas um atributo (7.11c).

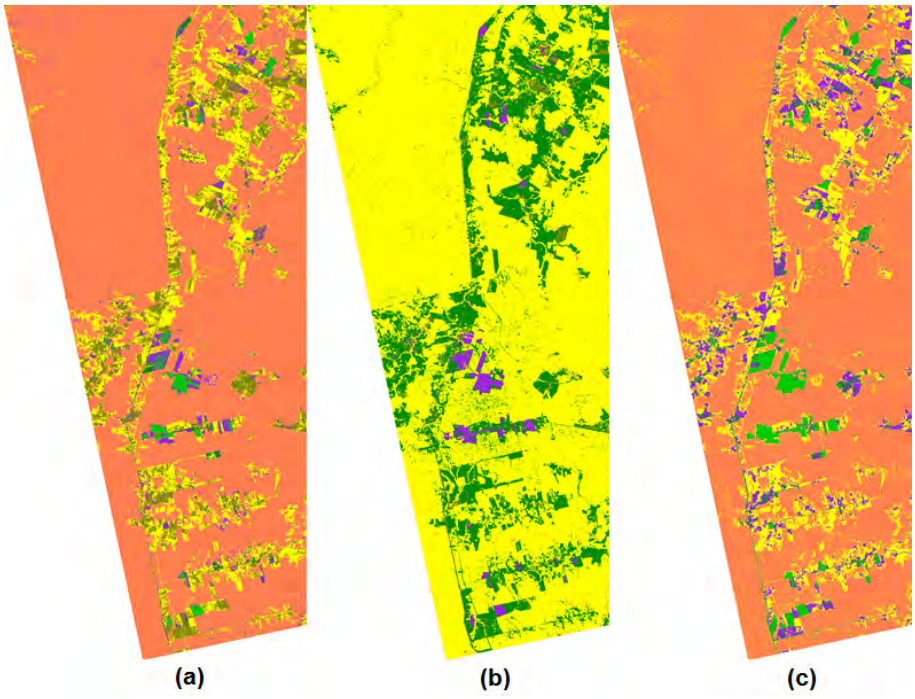

Figura 7.11 - Imagens classificadas para (Ω,F): (a) $(\Omega_H,\{B1,B2,B3,B4,B5,B7\})$; (b) $(\Omega_H,\{B1,B3,B4,B7\})$; (c) $(\Omega_G,\{B1\})$.

Os detalhes das três classificações da Figura 7.11 podem ser observados na Figura 7.12. Nesse caso, observa-se que, quando são utilizados conjuntos de classes semelhantes, o aumento no número de canais pode melhorar a qualidade da imagem classificada em relação à nitidez e definição.

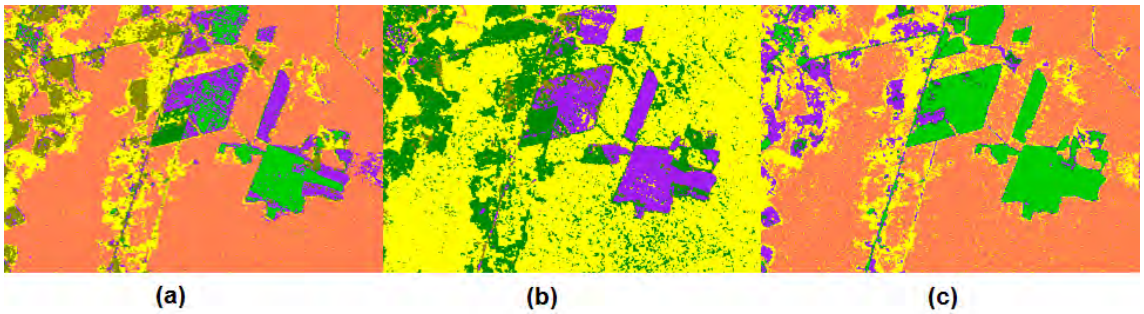

Figura 7.12 - Detalhes das classificações da Figura 7.11.

8 CONCLUSÕES E TRABALHOS FUTUROS

8.1 Conclusões

Este trabalho descreveu uma metodologia para análise de cenários de classificação de imagens, com aplicação na área de sensoriamento remoto. A primeira contribuição do trabalho é a proposta conceitual de cenários de classificação de imagens e sua formalização.

Foi proposto que o problema de definir os parâmetros e componentes do sistema de classificação seja modelado como um problema de otimização multiobjetivo flexível, na qual novos objetivos e variáveis de projeto podem ser adicionados conforme necessidade da aplicação.

O modelo proposto foi detalhado para um conjunto particular de variáveis de projeto e objetivos. Como variáveis de projeto, propôs-se utilizar o conjunto de atributos, o de classes e o classificador. Nesta proposta, os conjuntos de classes são gerados respeitando-se uma hierarquia fornecida pelo usuário. Essa característica visa evitar a criação de classes que combinam conceitos com pouca relação semântica, como por exemplo, uma classe "água_pasto". Os objetivos considerados neste modelo foram: maximizar a acurácia da classificação, em diversas modalidades, minimizar o tamanho do conjunto de atributos, definindo o melhor conjunto para a tarefa, e maximizar o tamanho do conjunto de classes, garantindo maior expecificidade das mesmas. Faz parte do modelo desenvolvido a possibilidade de utilizar árvores de classificação e de alterar essas árvores em um processo guiado por uma ontologia definida pelo usuário (Seção 5.1.2). Também é possível definir outros objetivos, conforme necessidade da aplicação.

Foi feita ainda a proposta de utilizar filtros adicionais para guiar o usuário na escolha de cenários dentro da fronteira de Pareto resultante da solução do problema multiobjetivo. A heurística é baseada em diferentes ordenações de prioridade dos objetivos, utilizando-se um teste estatístico para avaliar resultados semelhantes. A interação com o usuário ocorre após a obtenção da fronteira de Pareto, para auxiliá-lo a fazer sua escolha dentre os resultados obtidos, e o conjunto de Pareto é reduzido de acordo com o nível de confiança escolhido para o teste.

Outro resultado obtido foi a construção de um protótipo operacional para instanciação dos conceitos e aplicação em casos práticos, desenvolvido na linguagem IDL. Foi gerada uma ferramenta que pode ser utilizada por usuários de dados de sen-

soriamento remoto para avaliar suas escolhas antes de realizar uma classificação. A implementação utilizou o algoritmo de otimização geral M-GEO, que é um algoritmo baseado na evolução de sistemas naturais diante de eventos críticos. A aplicação do M-GEO no problema de análise de cenários de classificação também pode ser considerada como uma contribuição, pois permite sua avaliação em um domínio bastante diferente daqueles já testados.

A ferramenta gerada foi avaliada com um conjunto de dados controlado e com um conjunto de dados reais. Para o conjunto artificial, a ferramenta mostrou-se bastante eficiente, encontrando os pontos da fronteira com poucas iterações e pouco ajuste de seus parâmetros. No caso real, o ajuste não foi tão simples, sendo necessárias várias execuções independentes para cobrir a totalidade da fronteira de Pareto.

A inclusão do classificador SVM aumentou consideravelmente o tempo de execução, pois seu treinamento consiste também de um problema de otimização. A implementação multiclasse utilizada mostrou-se excessivamente lenta, o que prejudicou o desempenho do sistema sem melhorar seu resultado. Para os dados testados, o classificador Maxver foi bastante apropriado, apresentando valores de acurácia maiores que os do SVM e da ODT em um número maior de casos.

A flexibilidade proposta para o sistema foi alcançada no que diz respeito aos objetivos otimizados, à seleção das variáveis de projeto entre aquelas já implementadas e à adição de novos tipos de classificadores. Entretanto, a inclusão de novas variáveis de projeto requer um certo esforço de implementação, já que seriam necessárias novas estruturas de dados para armazená-las e algoritmos para gerar suas variações a cada iteração do algoritmo de otimização. Os testes mostraram que o sistema foi capaz de encontrar satisfatoriamente a fonteira de Pareto quando seus parâmetros foram bem ajustados.

8.2 Trabalhos Futuros

Como possibilidades de novas pesquisas, sugere-se avaliar a ferramenta com dados que apresentem distribuições não-Gaussianas, favorecendo assim o melhor desempenho de outros tipos de classificadores além do Maxver, como SVM e ODT. Também seria possível testar o sistema em outros tipos de aplicações além do sensoriamento remoto, e para a classificação de outros tipos de dados, além de imagens. A flexibilidade da implementação atual torna essa possibilidade bastante simples.

Pode-se ainda incluir outros tipos de classificadores, além de incluir os próprios

parâmetros dos classificadores no processo de otimização.

A eficiência do sistema poderia ser melhorada com a utilização de uma versão paralela do M-GEO, que ainda não foi desenvolvida. Existe, entretanto, uma versão paralela do GEO, que talvez possa servir de base para paralelizar também sua versão multiobjetivo.

Outra alternativa interessante seria incluir na implementação a classificação sequencial, conforme proposta do modelo (Seção 5.1.2), utilizando uma árvore de classificação. Essa implementação oferece diversas oportunidades de investigação, pois o cenário poderia ser definido para cada nó da árvore, alterando suas variáveis de projeto de um nó para o outro.

Mesmo sem a classificação sequencial, as folhas da árvore de classes fornecida pelo usuário poderiam ser averiguadas. No caso de classes multimodais, o sistema poderia sugerir não apenas a diminuição, mas também a extensão do conjunto de classes discriminado. Além disso, poderiam ser incluídos diversos critérios para a geração automática da hierarquia de classes com base nos atributos disponíveis.

Para o caso em que o usuário utiliza múltiplas fontes de dados, combinando, por exemplo, dados óticos e de radar, poderia ser interessante atribuir também prioridades para os canais utilizados, possibilitando o favorecimento da inclusão de atributos de uma fonte em particular no conjunto considerado.

Podem ser adicionados outros tipos de filtros e outras opções de medidas de acurácia e de ordenação ao algoritmo para reduzir a fronteira de Pareto, no intuito de auxiliar ainda mais o usuário em sua escolha final.

A ferramenta desenvolvida também pode ser utilizada para estudo de outros problemas de sensoriamento remoto como estudo de mudanças ou desmatamento, ou ainda para a escolha de parâmetros de classificadores, como o tipo de *kernel* do SVM.

REFERÊNCIAS BIBLIOGRÁFICAS

BROWN, K. M.; FOODY, G. M.; ATKINSON, P. M. Estimating per-pixel thematic uncertainty in remote sensing classifications. **International Journal of Remote Sensing**, v. 30, n. 1, p. 209–229, 2009. 46

CHEN, Y.-L.; HU, H.-W.; TANG, K. Constructing a decision tree from data with hierarchical class labels. **Expert Systems with Applications**, Elsevier, n. 36, p. 4838–4847, 2009. ISSN 0957-4174. Disponível em: <www.elsevier.com/locate/eswa>. 23

COHEN, J. A coefficient of agreement of nominal scales. **Educational and Psychological Measurement**, n. 20, p. 37–46, 1960. 12

COHON, J. L. **Multiobjective programming and planning**. New York, NY: Academic, 1978. (Mathematics in Science and Engineering). ISBN 0121783502. 45, 46

CRAMMER, K.; SINGER, Y. On the algorithmic implementation of multi-class kernel-based vector machines. **Journal of Machine Learning Research**, p. 265–292, 2001. 63

DASH, M.; LIU, H. Feature selection for classification. **Intelligent Data Analysis**, v. 1, n. 3, p. 131–156, 1997. 14, 15

DEB, K. **Multi-objective optimization using evolutionary algorithms**. 1. ed. Chichester: John Wiley and Sons, 2001. 17

DEB, K.; PRATAP, A.; AGARWAL, S.; MEYARIVAN, T. A fast and elitist multiobjective genetic algorithm: NSGA-II. **IEEE Transactions on Evolutionary Computation**, IEEE Press, Piscataway, NJ, USA, v. 6, n. 2, p. 182–197, abr. 2002. ISSN 1089-778X. Disponível em: <http://dx.doi.org/10.1109/4235.996017>. 3, 26

DUDA, R. O.; HART, P. E.; STORK, D. G. **Pattern classification**. 2. ed. New York, NY: John Wiley and Sons, 2001. ISBN 0471056693. 6

EKBAL, A.; SAHA, S. Multiobjective optimization for classifier ensemble and feature selection: an application to named entity recognition. **International Journal on Document Analysis and Recognition**, Springer-Verlag, Berlin, Heidelberg, v. 15, n. 2, p. 143–166, 2012. ISSN 1433-2833. Disponível em: <http://dx.doi.org/10.1007/s10032-011-0155-7>. 25

EXELIS VISUAL SOLUTIONS. **Exelis**: visual information solutions. 2012. Disponível em: <http://www.exelisvis.com/>. Acesso em: 23 set. 2012. 51

FAHEY, L.; RANDALL, R. M. (Ed.). **Learning from the future**: competitive foresight scenarios. Chichester: John Wiley & Sons, 1998. 1

FOODY, G. M. On the compensation for chance agreement in image classification accuracy assessment. **Photogrammetric Engineering and Remote Sensing**, n. 58, p. 1459–1460, 1992. 12

_____. Status of land cover classification accuracy assessment. **Remote Sensing of Environment**, n. 80, p. 185–201, 2002. 11

GALSKI, R. L. **Desenvolvimento de versões aprimoradas híbridas, paralela e multiobjetivo do método da otimização extrema generalizada e sua aplicação no projeto de sistemas espaciais**. 279 p. Tese (Doutorado) — Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2006. 3, 18, 19, 63

GALSKI, R. L.; SOUSA, F. L.; RAMOS, F. M. Application of a new evolutionary algorithm to the optimum design of a remote sensing satellite constelation. In: INTERNATIONAL CONFERENCE ON INVERSE PROBLEMS IN ENGINEERING: THEORY AND PRACTICE, 5., 2005, Cambridge. **Proceedings...** [S.l.], 2005. 19

GALSKI, R. L.; SOUSA, F. L. d.; RAMOS, F. M.; MURAOKA, I. Spacecraft thermal design with the generalized extremal optimization algorithm. In: INVERSE PROBLEMS, DESIGN AND OPTIMIZATION SYMPOSIUM, 17-19 Mar., Rio de Janeiro. 2004. Disponível em: <http://urlib.net/sid.inpe.br/marciana/2004/09.20.10.27>. Acesso em: 23 set. 2012. 19

HAWAII INSTITUTE OF GEOPHYSICS AND PLANETOLOGY. **Airborne Hyperspectral Imager**. 2001. Hawaii. Disponível em: <http://www.higp.hawaii.edu/ahi/>. Acesso em: february, 2008. 13

HEIJDEN, K. van der; BRADFIELD, R.; BURT, G.; CAIRNS, G.; WRIGHT, G. **The sixth sense**: accelerating organizational learning with scenarios. Chichester: John Wiley & Sons, 2002. 1

HUANG, C.-L.; WANG, C.-J. A GA-based feature selection and parameters optimization for support vector machines. **Expert Systems with Applications**, v. 31, n. 2, p. 231–240, 2006. 24

JAIN, A. K.; DUIN, R. P. W.; MAO, J. Statistical pattern recognition: a review. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 22, n. 1, p. 4–37, 2000. 14, 15

JOVANOVIC, N. **Lattice Tutorial**. 2005. Disponível em: <http://www.iseclab.org/people/enji/infosys/lattice_tutorial.pdf>. Acesso em: 22 fev. 2013. 29

KAVZOGLU, T.; MATHER, P. M. The role of feature selection in artificial neural network applications. **International Journal of Remote Sensing**, v. 23, n. 15, p. 2919–2937, 2002. 16

LEE, J.-S.; GRUNES, M.; POTTIER, E. Quantitative comparison of classification capability: fully polarimetric versus dual and single-polarization sar. **IEEE Transactions on Geoscience and Remote Sensing**, v. 39, n. 11, p. 2343–2351, 2001. 12

LIM, H. S.; MATJAFRI, M. Z.; ABDULLAH, K.; SALEH, N. M. High spatial resolution land cover mapping using ALOS data over Kedah, Malaysia. In: FIRST JOINT PI SYMPOSIUM OF ALOS DATA NODES FOR ALOS SCIENCE PROGRAM, 2007, Kyoto,Japan. **Proceedings...** Kyoto, 2007. 12

LIPSCHUTZ, S.; LIPSON, M. **Matemática discreta**. 2. ed. Porto Alegre, RS: Bookman, 1997. 29

LIU, C.; FRAZIER, P.; KUMAR, L. Comparative assessment of the measures of thematic classification accuracy. **Remote Sensing of Environment**, n. 107, p. 606–616, 2007. 13

MAINENTI-LOPES, I.; SOUZA, L. C. G. d.; SOUSA, F. L. d. Design of the satellite attitude control system using multi-objective generalized extremal optimization. **Advances in the astronautical sciences**, I, p. 1241–1252, 2011. ISSN 0065-3438. Setores de Atividade: Atividades profissionais, científicas e técnicas. Disponível em: <http://urlib.net/dpi.inpe.br/plutao/2011/09.22.17.20.57>. Acesso em: 23 set. 2012. 19

MATHER, P. M. **Computer processing of remotely-sensed images**: an introduction. 3. ed. NY: John Wiley & Sons, 2004. 1

MEDEIROS, I. P. d.; CASTRO FILHO, C. A. P. d.; ERTHAL, G. J.; DUTRA, L. V. Classificação de imagens pelo método de árvore de decisão oblíqua. In: XV

SIMPÓSIO BRASILEIRO DE SENSORIAMENTO REMOTO, 2011, Curitiba, Brasil. **Proceedings...** [S.l.], 2011. p. 4255–4262. 11, 63

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION. **AVIRIS**: Airborne visible/infrared imaging spectometer. 2007. Jet Propulsion Laboratory, Pasadena. Disponível em: <http://aviris.jpl.nasa.gov/>. Acesso em: 23 set. 2012. 13

PANTALEÃO, E.; SCOFIELD, G. B. Comparação entre medidas de acurácia de classificação para imagens do satélite ALOS. **XIV Simpósio Brasileiro de Sensoriamento Remoto**, p. 7039–7046, 2009. 13

PEREIRA, L. d. O. **Avaliação de métodos de integração de imagens ópticas e de Radar para a classificação do uso e cobertura da terra na Região Amazônica**. 270 p. Dissertação (Mestrado) — Instituto Nacional de Pesquisas Espaciais (INPE), São José dos Campos, 2012-08-27 2012. Disponível em: <http://urlib.net/sid.inpe.br/mtc-m19/2012/08.30.12.50>. Acesso em: 06 nov. 2012. 72

ROCCO, E. M. **Manutenção orbital de constelações simétricas de satélites utilizando manobras impulsivas ótimas com vínculo de tempo**. Tese (Doutorado) — Instituto Nacional de Pesquisas Espaciais, São José dos Campos, out. 2002 2002. Acesso em: 07 nov. 2012. 20, 21

ROSENFIELD, G. H.; FITZPATRICK-LINS, K. A coefficient of agreement as a measure of thematic classification accuracy. **Photogrammetric Engineering and Remote Sensing**, v. 52, p. 223–227, 1986. 13

SANTOS, J. C. **Extração de atributos de forma e seleção de atributos usando algoritmos genéticos para a classificação de regiões**. 99 p. Tese (Dissertação (Computação Aplicada)) — Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2007. 16

SOUSA, F. L. d. **Otimização extrema generalizada: um novo algoritmo estocástico para o projeto ótimo**. 142 p. Tese (Doutorado) — Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2002. Disponível em: <http://urlib.net/sid.inpe.br/marciana/2003/03.18.15.39>. Acesso em: 23 set. 2012. 18, 63, 70

SPOLAÔR, N. **Aplicação de algoritmos genéticos multiobjetivo ao problema de seleção de atributos**. 134 p. Tese (Dissertação (Mestrado em

Engenharia de Informação)) — UFABC, Santo André, 2010. Disponível em: <http://www.inf.ufpr.br/aurora/disciplinas/topicosia2/livros/Dissertacao.pdf>. Acesso em: 23 set. 2012. 20, 25, 26

STORY, M.; CONGALTON, R. G. Accuracy assessment: A user's perspective. **Photogrammetric Engineering and Remote Sensing**, v. 52, p. 397–399, 1986. 11, 13

THEODORIDIS, S.; KOUTROUMBAS, K. **Pattern recognition**. 3. ed. San Diego: Academic Press, 2006. 2, 7, 8, 9, 14, 16, 24

UNITED STATES GEOLOGICAL SURVEY. **EO-1 User's Guide**: Data products: Hyperion. 2008. United States. Disponível em: <http://eo1.usgs.gov/documents/EO1userguidev2pt320030715UC.pdf>. Acesso em: 23 set. 2012. 13

VAPNIK, V.; GOLOWICH, S. E.; SMOLA, A. Support vector method for function approximation, regression estimation, and signal processing. In: **Advances in neural information processing systems 9**. [S.l.]: MIT Press, 1996. p. 281–287. 7

WEBB, A. R. **Statistical pattern recognition**. 1. ed. Chichester: John Wiley & Sons, 2002. 8

ZELENY, M. Compromise programming. In: COCHRANE, J.; ZELENY, M. (Ed.). **Multiple criteria decision making**. Columbia: University of South Carolina Press, 1973. p. 262–301. xiii, 19, 20, 21, 26

PUBLICAÇÕES TÉCNICO-CIENTÍFICAS EDITADAS PELO INPE

Teses e Dissertações (TDI)

Teses e Dissertações apresentadas nos Cursos de Pós-Graduação do INPE.

Manuais Técnicos (MAN)

São publicações de caráter técnico que incluem normas, procedimentos, instruções e orientações.

Notas Técnico-Científicas (NTC)

Incluem resultados preliminares de pesquisa, descrição de equipamentos, descrição e ou documentação de programas de computador, descrição de sistemas e experimentos, apresentação de testes, dados, atlas, e documentação de projetos de engenharia.

Relatórios de Pesquisa (RPQ)

Reportam resultados ou progressos de pesquisas tanto de natureza técnica quanto científica, cujo nível seja compatível com o de uma publicação em periódico nacional ou internacional.

Propostas e Relatórios de Projetos (PRP)

São propostas de projetos técnico-científicos e relatórios de acompanhamento de projetos, atividades e convênios.

Publicações Didáticas (PUD)

Incluem apostilas, notas de aula e manuais didáticos.

Publicações Seriadas

São os seriados técnico-científicos: boletins, periódicos, anuários e anais de eventos (simpósios e congressos). Constam destas publicações o Internacional Standard Serial Number (ISSN), que é um código único e definitivo para identificação de títulos de seriados.

Programas de Computador (PDC)

São a seqüência de instruções ou códigos, expressos em uma linguagem de programação compilada ou interpretada, a ser executada por um computador para alcançar um determinado objetivo. Aceitam-se tanto programas fonte quanto os executáveis.

Pré-publicações (PRE)

Todos os artigos publicados em periódicos, anais e como capítulos de livros.