



MINISTÉRIO DA CIÊNCIA, TECNOLOGIA, INOVAÇÕES E COMUNICAÇÕES
INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

sid.inpe.br/mtc-m21b/2017/01.17.16.20-TDI

HSMI: MÉTODO DE CLASSIFICAÇÃO HIERÁRQUICO BASEADO EM SVM MULTIKERNEL COM OTIMIZAÇÃO META-HEURÍSTICA

Michelle de Oliveira Parreira

Tese de Doutorado do Curso de
Pós-Graduação em Computação
Aplicada, orientada pelos Drs.
Luciano Vieira Dutra, e Eliana
Pantaleão, aprovada em 24 de
novembro de 2016.

URL do documento original:

[<http://urlib.net/8JMKD3MGP3W34P/3N7M8QP>](http://urlib.net/8JMKD3MGP3W34P/3N7M8QP)

INPE
São José dos Campos
2017

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@inpe.br

**COMISSÃO DO CONSELHO DE EDITORAÇÃO E PRESERVAÇÃO
DA PRODUÇÃO INTELECTUAL DO INPE (DE/DIR-544):****Presidente:**

Maria do Carmo de Andrade Nono - Conselho de Pós-Graduação (CPG)

Membros:

Dr. Plínio Carlos Alvalá - Centro de Ciência do Sistema Terrestre (CST)

Dr. André de Castro Milone - Coordenação de Ciências Espaciais e Atmosféricas (CEA)

Dra. Carina de Barros Melo - Coordenação de Laboratórios Associados (CTE)

Dr. Evandro Marconi Rocco - Coordenação de Engenharia e Tecnologia Espacial (ETE)

Dr. Hermann Johann Heinrich Kux - Coordenação de Observação da Terra (OBT)

Dr. Marley Cavalcante de Lima Moscati - Centro de Previsão de Tempo e Estudos Climáticos (CPT)

Silvia Castro Marcelino - Serviço de Informação e Documentação (SID)

BIBLIOTECA DIGITAL:

Dr. Gerald Jean Francis Banon

Clayton Martins Pereira - Serviço de Informação e Documentação (SID)

REVISÃO E NORMALIZAÇÃO DOCUMENTÁRIA:

Simone Angélica Del Ducca Barbedo - Serviço de Informação e Documentação (SID)

Yolanda Ribeiro da Silva Souza - Serviço de Informação e Documentação (SID)

EDITORAÇÃO ELETRÔNICA:

Marcelo de Castro Pazos - Serviço de Informação e Documentação (SID)

André Luis Dias Fernandes - Serviço de Informação e Documentação (SID)



MINISTÉRIO DA CIÊNCIA, TECNOLOGIA, INOVAÇÕES E COMUNICAÇÕES
INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

sid.inpe.br/mtc-m21b/2017/01.17.16.20-TDI

HSMI: MÉTODO DE CLASSIFICAÇÃO HIERÁRQUICO BASEADO EM SVM MULTIKERNEL COM OTIMIZAÇÃO META-HEURÍSTICA

Michelle de Oliveira Parreira

Tese de Doutorado do Curso de
Pós-Graduação em Computação
Aplicada, orientada pelos Drs.
Luciano Vieira Dutra, e Eliana
Pantaleão, aprovada em 24 de
novembro de 2016.

URL do documento original:

<<http://urlib.net/8JMKD3MGP3W34P/3N7M8QP>>

INPE
São José dos Campos
2017

Dados Internacionais de Catalogação na Publicação (CIP)

Parreira, Michelle de Oliveira.

P248h HSMI: Método de classificação hierárquico baseado em SVM
multikernel com otimização meta-heurística / Michelle de Oliveira
Parreira. – São José dos Campos : INPE, 2017.
xxiv + 87 p. ; (sid.inpe.br/mtc-m21b/2017/01.17.16.20-TDI)

Tese (Doutorado em Computação Aplicada) – Instituto
Nacional de Pesquisas Espaciais, São José dos Campos, 2016.

Orientadores : Drs. Luciano Vieira Dutra, e Eliana Pantaleão.

1. Máquinas de vetores de suporte. 2. Combinação de
classificadores. 3. Classificação binária. 4. Sensoriamento remoto.
5. Reconhecimento de padrões. I.Título.

CDU 519.142:004.023



Esta obra foi licenciada sob uma Licença [Creative Commons Atribuição-NãoComercial 3.0 Não Adaptada](#).

This work is licensed under a [Creative Commons Attribution-NonCommercial 3.0 Unported License](#).

Aluno (a): **Michelle de Oliveira Parreira**

"HSMI: MÉTODO DE CLASSIFICAÇÃO HIERÁRQUICO BASEADO EM SVM MULTIKERNEL COM OTIMIZAÇÃO META-HEURÍSTICA"

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

Computação Aplicada

Dr. Rafael Duarte Coelho dos Santos



Presidente / INPE / SJCampos - SP

Dr. Luciano Vieira Dutra



Orientador(a) / INPE / SJCampos - SP

Dra. Eliana Pantaleão




Orientador(a) / UFU / Patos de Minas - MG

Dr. Carlos Henrique Quartucci Forster




Convidado(a) / ITA / São José dos Campos - SP

Dr. Rogério Galante Negri



Convidado(a) / UNESP / São José dos Campos - SP

Dr. Nelson Delfino d'Ávila Mascarenhas



Convidado(a) / FACCAMP / Campo Limpo Paulista - SP

Este trabalho foi aprovado por:

() maioria simples

☒ unanimidade

São José dos Campos, 24 de Novembro de 2016

“Não importa quanto a vida possa ser ruim, sempre existe algo que você pode fazer, e triunfar. Enquanto há vida, há esperança”.

STEPHEN HAWKING

*A meu filho Gibran,
a meu esposo Sherfis
e a minha mãe Maria Dexite*

AGRADECIMENTOS

Agradeço a Deus por sempre estar presente em minha vida, ensinando-me a ter paciência e acreditar no caminho a ser seguido.

Agradeço a todos que de alguma forma colaboraram para a conclusão deste trabalho. Em especial:

A meu filho Gibran, meu lindo presente que Deus concedeu-me durante o programa de doutorado, que veio para iluminar meus dias com seu sorriso encantador, dando-me forças para lutar e conhecer o verdadeiro sentido da vida.

A meu esposo Sherfis que me estendeu a mão e deu-me coragem, confiança, companheirismo e auxílio no desenvolvimento desta tese.

A minha maravilhosa mãe Maria Dezite que esteve cuidando do meu pequeno Gibran, dando carinho e atenção para que eu pudesse concluir este trabalho.

A minha família, em especial meu pai Mauri pelo incentivo aos estudos ao longo da vida, e irmãos Diego, Gabriel e Iruam pelo nosso elo.

Aos meus orientadores Dr. Luciano Vieira Dutra e Dra. Eliana Pantaleão pela paciência, dedicação e todo conhecimento compartilhado.

Aos professores da CAP, em especial: Dra. Corina Costa Freita, Dr. Sidnei João Siqueira Sant'Anna, Dra. Sandra Sandri, Dr. Nandamudi L. Vijaykumar, Dr. Rafael Duarte Coelho dos Santos, Dr. Reinaldo Roberto Rosa, Dr. Solon Venâncio de Carvalho, Dr. Ezzat Selim Chalhoub, Dr. José Carlos Becceneri e Dr. Haroldo Fraga de Campos Velho.

Aos amigos do INPE, turma da CAP 2010/2011, galerinha inesquecível: Luciana Rebelo, Amarísio Araújo, Jonas Mendonça, Érica Gouvêa, Sabrina Sanbatti, Luis França, Pettras dos Santos, Ligia Corrêa, Diego Stalder, Érica de Souza, Marlon da Silva, Saymon Santana, André Chiarelli, Marcos Borges, Marina Nascimento, Peterson Sarmento, Juliana Bueno, Juliana Anochi, Sóstenes Gomes, Victor Hugo Silva, Anna Karina Gomes, Maria Teodora Ferreira, Alessandro Arantes e Aline da Silva.

Aos amigos do INPE, turma da Senzala/DPI: Carlos Pires, Wagner Barreto, Rogério Negri, Luciana Pereira, Marinalva Soares, Flávia Martins, Daniela Anjos, Leonardo

Torres, Eliana Pantaleão, Henrique, Maria Antônia e Mariane. Ao parceiro da ETE Wagner Mahler.

Ao Instituto Nacional de Pesquisas Espaciais (INPE) e às secretarias da CAP e da Pós-Graduação do INPE.

À *Michigan State University* (MSU), em especial meu co-orientador Dr. Dengsheng Lu e a secretária Jean Lepard do *Center for Global Change and Earth Observations*.

Aos laços de amizade construído durante o programa de doutorado sanduiche, na cidade de East Lansing-MI/EUA, nossos amigos: Lorena Barbosa, Juliana Ribeiro, Ramon Bicudo, Marilene Santos, Thaisa Pegoraro, Dalton Ferreira e Thais Ribeiro. Aos amigos chineses Xue Li, Yuxing Wang, Shengpan Lin, Ruoxuan Zhang e Marie Wang. À Trinity Church, em especial ao casal Bonie e John Bankson. À Friendship House MSU, em especial aos amigos Rich Bearup, Janine Boehmer, Patty Mazarriegos, Mamata Acharya Dangi, Laman Tatanaki, Maryam Irannejad, Lena Grau, Stéfanie Fares Sabbag, Lucie Aussignargues e Francesco D'Onofrio. À família Ricka, Janel e Merle Boehmer.

Ao Instituto Federal de São Paulo (IFSP), campus Registro, e à Universidade Federal de Mato Grosso (UFMT), campus Rondonópolis.

À Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) pela concessão da bolsa de doutorado durante o programa no Brasil e ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) pela concessão da bolsa de doutorado sanduíche nos EUA.

Aos professores membros internos e externos da banca pelas valiosas contribuições.

Enfim, aos amigos e familiares, meus sinceros agradecimentos.

RESUMO

Esse trabalho propõe o método HSMI (*Hierarchical Support vector machine with Multiple kernels optimized by Invasive weed optimization*) de classificação baseado em máquinas de vetores suporte (SVM) que usa múltiplos kernels e atribui os rótulos às classes de modo hierárquico. Uma árvore binária é criada automaticamente pelo algoritmo proposto e cada nó realiza a classificação entre duas partições do conjunto de classes pré-classificado pelo nó superior. A classificação é realizada pelo classificador SVM com múltiplos kernels combinados aproveitando as diferentes características de cada kernel. A escolha pelas classes que compõem cada partição em cada nó é feita por otimização junto com os parâmetros dos kernels e os coeficientes da combinação linear entre eles. Para isso é empregado o algoritmo Infestação por Ervas Daninhas (*Invasive Weed Optimization*, IWO). Esse novo método consegue separar hierarquicamente as classes com melhor separabilidade segundo um classificador SVM multikernel otimizado para cada classificação binária. Os resultados foram comparados com o método SVM com kernel gaussiano e SVM com kernel polinomial. Os resultados demonstraram que o método HSMI ao particionar as classes de forma embutida permite a fusão de classes confusas identificadas no processo de classificação.

Palavras-chave: Máquinas de Vetores de Suporte. Combinação de Classificadores. Classificação Binária. Sensoriamento Remoto. Reconhecimento de Padrões.

HSMI: HIERARCHICAL CLASSIFICATION METHOD BASED ON MULTI-KERNEL SVM WITH META-HEURISTIC OPTIMIZATION

ABSTRACT

This work proposes the classifier method HSMI (Hierarchical support vector machine with multiple kernels optimized by Invasive weed optimization) based on support vector machine (SVM) that uses multiple kernels and assigns the labels to classes in a hierarchical way. A binary tree is automatically created by the proposed algorithm and each node performs the classification between two partitions of the set of pre-sorted classes by the upper node. The classification is performed by the SVM classifier with multiple kernels combined taking advantage of the different characteristics of each kernel. The choice of the classes that make up each partition at each node is done by optimization along with the parameters of the kernels and the coefficients of the linear combination between them. For this the Invasive Weed Optimization algorithm (IWO) is used. This new method can separate hierarchically classes with better separability according to a multi-kernel SVM classifier optimized for each binary classification. The results were compared with the SVM method with Gaussian kernel and SVM with polynomial kernel. The results showed that the HSMI method in partitioning the classes of embedded form allows the fusion of confused classes identified in the classification process.

Keywords: Support Vector Machine. Ensembles. Binary Classification. Remote sensing. Pattern Recognition.

LISTA DE FIGURAS

	<u>Pág.</u>
2.1 Abordagens para a construção de <i>ensembles</i> em combinação de classificadores.	7
2.2 Três razões para construir bons <i>ensembles</i>	8
2.3 C1. Arquitetura de classificação - combinação cooperativa com divisão do espaço de características.	10
2.4 C2. Arquitetura de classificação - combinação cooperativa a partir do mesmo espaço de características.	10
2.5 Metodologia desenvolvida.	16
2.6 Procedimento de produção de sementes em uma colônia.	18
2.7 (a) Exemplo de partição dos espaços de atributos (b) Exemplo de uma árvore de decisão.	20
2.8 Exemplo de uma Árvore Comum de Classificação Binária.	23
2.9 Exemplo de uma Árvore de Decisão Oblíqua.	23
2.10 Impacto do Overfitting em aprendizagem por árvore de decisão - Relacionamento entre tamanho da árvore de decisão e a taxa de erro.	25
2.11 Erro empírico e funcional.	27
3.1 Particionamento das classes.	33
3.2 Construção da árvore HSMI.	36
3.3 Estrutura do Cluster.	38
3.4 Algoritmo SMI em cluster.	39
3.5 Metodologia: <i>Hierarchical Support vector machine with Multiple kernels and Invasive weed Optimization</i> (HSMI).	43
4.1 Imagem Sintética. Composição colorida dos três canais: R, G e B. 512 linhas e 512 colunas.	46
4.2 Imagem Sintética. Três Canais: (1)R, (2)G e (3)B.	47
4.3 Imagem Sintética com a visualização das amostras das classes utilizadas.	47
4.4 Número de <i>pixels</i> das amostras que foram utilizadas para treinamento, validação e teste. Conjunto de 4 classes.	47
4.5 Árvores treinadas para imagem sintética.	49
4.6 Imagem sintética classificada pelo método HSMI.	50

4.7	Localização da área de estudo. a) localização em relação à Amazônia Legal brasileira; b) localização em relação aos limites políticos e geográficos; c) recorte aproximado da área de estudo de imagem LANDSAT5/TM de 29 de junho de 2010. Composição colorida 5(R)4(G)3(B).	51
4.8	Número de pixels das amostras que foram utilizadas para treinamento, validação e teste. Para imagem TM (Coluna 30m de resolução) e para imagem de Radar (Coluna 15m de resolução). Conjunto de 10 classes. . .	53
4.9	Número de pixels das amostras que foram utilizadas para treinamento, validação e teste. Para imagem TM (Coluna 30m de resolução) e para imagem de Radar (Coluna 15m de resolução). Conjunto de 6 classes. . .	54
4.10	Imagem LANDSAT5/TM de 29 de junho de 2010, composição colorida 5(R)4(G)3(B). Coordenadas referentes a UTM/WGS84.	55
4.11	Árvore treinada para imagem TM com 10 classes.	57
4.12	Imagem TM - 10 classes - classificada pelo método HSML.	58
4.13	Árvore treinada para imagem TM com 6 classes.	60
4.14	Imagem TM - 6 classes - classificada pelo método HSML.	61
4.15	Imagem ALOS/PALSAR de 21 de junho de 2010, em amplitude e na composição colorida HH(R)HV(G)HH(B). Coordenadas referentes a UTM/WGS84.	63
4.16	Árvore treinada para imagem de Radar com 10 classes.	64
4.17	Imagem de Radar - 10 classes - classificada pelo método HSML.	65
4.18	Árvore treinada para imagem de RADAR com 6 classes.	67

LISTA DE TABELAS

	<u>Pág.</u>
4.1 Imagem sintética - SVM-RBF $\gamma = 0,004$	48
4.2 Imagem sintética - SVM-POLY $d = 2$	48
4.3 Imagem sintética - HSMI	48
4.4 Imagem sintética - HSMI - parâmetros da solução para a primeira árvore.	49
4.5 Imagem sintética - HSMI - parâmetros da solução para a segunda árvore.	49
4.6 Imagem TM - 10 classes - SVM-RBF $\gamma = 0,004$	56
4.7 Imagem TM - 10 classes - HSMI	57
4.8 Imagem TM - 10 classes - HSMI - parâmetros da solução.	58
4.9 Imagem TM - 6 classes - SVM-RBF $\gamma = 0,004$	59
4.10 Imagem TM - 6 classes - HSMI.	59
4.11 Imagem TM - 6 classes - HSMI - parâmetros da solução.	60
4.12 Imagem TM - 10 classes - SVM-RBF $\gamma = 0,004$	62
4.13 Imagem de Radar - 10 classes - HSMI	64
4.14 Imagem de Radar - 10 classes - HSMI - parâmetros da solução.	65
4.15 Imagem RADAR - 6 classes - SVM-RBF $\gamma = 0,004$	66
4.16 Imagem RADAR - 6 classes - HSMI	66
4.17 Imagem RADAR - 6 classes - HSMI - parâmetros da solução.	67

LISTA DE ABREVIATURAS E SIGLAS

AC	–	Área Cultivada
ACO	–	<i>Ant Colony Optimization</i>
AD	–	Árvore de Decisão
ALOS	–	<i>Advanced Land Observing Satellite</i>
AP	–	Área em Pousio
CCP	–	<i>Cost Complexity Pruning</i>
CT	–	Conjunto de Treinamento
DAG	–	<i>Direct Acyclic Graph</i>
EBP	–	<i>Error Based Pruning</i>
ERTS	–	<i>Earth Resources Technology Satellites</i>
FBD	–	<i>Fine Beam Dual</i>
FD	–	Floresta Degradada
FP	–	Floresta Primária
HSMI	–	<i>Hierarchical Support vector machine with Multiple kernels optimized by Invasive weed optimization</i>
IAD	–	Indução de Árvore de Decisão
IWO	–	<i>Invasive Weed Optimization</i>
JAXA	–	Agência Espacial Japonesa
JM	–	Jeffreys-Matusita
LIN	–	Linear
MAXVER	–	Máxima Verossimilhança
MKL	–	<i>Multiple Kernel Learning</i>
NASA	–	<i>National Aeronautics and Space Administration</i>
PALSAR	–	<i>Phase Array L-Band Synthetic Aperture Radar</i>
PL	–	Pasto Limpo
POLY	–	Polinomial
PS	–	Pasto Sujo
PSO	–	<i>Particle Swarm Optimization</i>
RBF	–	<i>Radial Basis Function</i>
REP	–	<i>Reduced Error Pruning</i>
SA	–	<i>Simulated Annealing</i>
SAR	–	<i>Synthetic Aperture Radar</i>
SDNLM	–	<i>Stochastic Distances Nonlocal Means</i>
SE	–	Solo Exposto
SV	–	Vetores de Suporte (<i>Support Vectors</i>)
SVM	–	Máquina de Vetores de Suporte (<i>Support Vector Machine</i>)
TDIDT	–	<i>Top-Down Induction of Decision Tree</i>
VS	–	Vegetação Secundária
VS1	–	Vegetação Secundária Inicial
VS2	–	Vegetação Secundária Intermediária

VS3 – Vegetação Secundária Avançada

LISTA DE SÍMBOLOS

N	– quantidade de amostras rotuladas
\mathbf{x}	– vetor de atributos de uma amostra
x_{ij}	– valor do atributo j da amostra i
i, j	– índices
y	– rótulo da classe
D	– dimensão do espaço de características
f	– função
X	– conjunto de amostras rotuladas
ω	– identificador de uma classe em um problema multi-classe
Ω	– conjunto das classes em um problema multi-classe
C	– quantidade de classes do conjunto Ω
Cl_i	– um classificador
S_i	– dados de entrada para um classificador
P_{Cl_i}	– probabilidade do classificador Cl_i
$\hat{\omega}$	– classe atribuída a uma amostra
k	– quantidade de classificadores em uma combinação de classificadores; ou dimensão do espaço de atributos transformado por uma função.
$Pop_{inicial}$	– população inicial
Pop_{max}	– população máxima
$iter_{max}$	– número máximo de iterações
$D_{problema}$	– dimensão do problema
ns_{max}	– número máximo de sementes que uma erva daninha pode gerar em uma reprodução
ns_{min}	– número mínimo de sementes que uma erva daninha pode gerar em uma reprodução
mod	– módulo de não linearidade para IWO
$\sigma_{inicial}$	– variância inicial para IWO
σ_{final}	– variância final para IWO
F_{min}	– maior aptidão encontrada na população
F_{max}	– menor aptidão encontrada na população
F_{weed}	– aptidão de uma erva daninha
$iter$	– iteração corrente do algoritmo
ns_i	– número de sementes para reprodução de uma erva daninha i
σ_{iter}	– variância aplicada em uma dada iteração
rnd	– valor aleatório
m	– dimensão do vetor de amostra
F	– conjunto de funções
R_{emp}	– erro empírico
R_f	– erro funcional
\mathbf{w}	– vetor perpendicular ao hiperplano do SVM

b	–	escalar
λ_i	–	multiplicador de Lagrange
Ω_L	–	partição esquerda
Ω_R	–	partição direita
Π	–	dados das amostras de treinamento, validação e teste de uma partição
K	–	função kernel
l	–	é o número de amostras de treinamento
c_1, c_2, c_3	–	coeficientes
a, c, d	–	parâmetros da função kernel polinomial
γ	–	parâmetro da função kernel gaussiana
Q	–	conjunto de parâmetros
P_k	–	possível solução para IWO
κ	–	índice de concordância <i>kappa</i>
M	–	quantidade de classes em um problema multi-classe
R	–	registro
A	–	conjunto de treinamento
B	–	conjunto de validação
Γ	–	conjunto de teste
θ	–	partição

SUMÁRIO

	<u>Pág.</u>
1 Introdução	1
1.1 Problema	3
1.2 Objetivos	5
1.2.1 Objetivo Geral	5
1.2.2 Objetivos Específicos	6
1.3 Organização da Tese	6
 2 Métodos de Classificação	 7
2.1 Combinação de Classificadores	7
2.2 Esquemas de Combinação	9
2.2.1 Espaço de Características	9
2.2.2 Nível	11
2.2.3 Grau de Treinamento	11
2.2.3.1 Combinadores Não Treináveis	12
2.2.3.2 Combinador Treinável	13
2.2.4 Forma	14
2.2.5 Estrutura	14
2.2.6 Otimização	15
2.2.6.1 Método de Otimização por Meta-Heurística: Infestação de Ervas Daninhas (IWO)	15
2.3 Classificador por Indução de Árvore de Decisão (IAD)	19
2.3.1 Representações dos Nós de Decisão	19
2.3.2 Construção de uma Árvore de Decisão	20
2.3.3 Limitações na Dimensão das Árvores de Decisão (Poda)	24
2.4 Máquina de Vetores de Suporte - <i>Support Vector Machine</i> (SVM)	26
2.4.1 Múltiplos <i>Kernels</i>	29
 3 Método de Classificação HSMI	 31
3.1 Descrição do nó e construção da árvore	32
3.2 Arquitetura computacional e implementação	36
 4 Experimentos e Resultados	 45
4.1 Estudo de Caso 1 - Imagem Sintética	46

4.1.1	Dados	46
4.1.2	Resultados	48
4.2	Estudo de Caso 2 - Imagem Óptica	50
4.2.1	Dados	53
4.2.2	Resultados	56
4.3	Estudo de Caso 3 - Imagem Radar	59
4.3.1	Dados	61
4.3.2	Resultados	66
5	Considerações Finais	69
5.1	Resumo dos Resultados Obtidos	69
5.2	Trabalhos Relacionados	70
5.3	Principais Contribuições e Limitações	74
5.4	Perspectivas Futuras	75
	REFERÊNCIAS BIBLIOGRÁFICAS	77

1 Introdução

Para monitorar o crescimento urbano, observar o desmatamento e planejar área de cultivo faz-se necessário o monitoramento e observação do uso e cobertura da Terra. Devido ao vasto território a ser trabalhado, não é viável para os especialistas visitarem todos os lugares para recolhimento das amostras de áreas de interesse. Com o avanço das tecnologias de satélite nos últimos cinquenta anos, grandes áreas de terra podem ser verificadas pelos sensores ópticos e de radar. No entanto, estes produzem grande quantidade de informações que são impossíveis de ser analisadas manualmente. Sensoriamento remoto como uma área de conhecimento pode ser aplicado para realizar essa tarefa (COLWELL, 1997). Algoritmos computacionais são desenvolvidos para processar esta grande quantidade de dados e extrair novas informações sobre o uso e a cobertura da Terra.

Classificação de imagens em Sensoriamento Remoto é a extração de dados de áreas da superfície terrestre que tem por intuito reconhecer padrões homogêneos dentro da imagem. Classificação supervisionada é uma atividade preditiva, que consiste na generalização de registros a partir de respostas conhecidas utilizadas para classificar novos exemplos. A partir de um conjunto de treinamento constrói-se um modelo para cada classe baseado nas características dos dados. As atividades preditivas podem ser de classificação ou regressão (THEODORIDIS; KOUTROUMBAS, 2009). A classificação ainda pode ser do tipo não supervisionada, que consiste de atividades descritivas, sem conhecimento a priori do rótulo da classe como regras de associação, clusterização e sumarização (THEODORIDIS; KOUTROUMBAS, 2009).

Diante da importância e necessidade de se obter resultados cada vez mais precisos, diferentes métodos de classificação de imagens e variações de métodos já existentes são constantemente desenvolvidos. De modo geral, os métodos de classificação de imagens possuem características básicas, as quais permitem subdividi-los em pontual, que efetua a avaliação somente do pixel, contextual, que avalia o pixel e sua vizinhança, ou por regiões, que usa toda a informação de um segmento ou objeto da imagem.

Normalmente, especialistas visitam áreas que representam cada classe e definem essas áreas na imagem como dados rotulados. Em seguida, classificadores supervisionados usam esses dados rotulados para treinar um algoritmo para reconhecer cada classe. Finalmente, todos os conjuntos de informações são processados e cada padrão é previsto para uma das classes de entrada. Dentre os classificadores supervisionados mais utilizados na literatura, tem-se: Máquinas de Vetores Suporte - *Support Vec-*

tor Machine (SVM), K-Vizinhos mais próximos (KNN), Máxima Verossimilhança (MAXVER), Redes Neurais Artificiais (RNA) (KOVÁCS, 2002) e Indução de Árvores de Decisão (IAD) (QUINLAN, 1986). Dentre estes, o SVM tem se destacado na classificação pontual, por (SMOLA, 2000): algoritmo de arquitetura simples, boa capacidade de generalização, convexidade da função objetivo, teoria bem definida baseada na Teoria de Aprendizado Estatístico (TAE).

De modo geral, os procedimentos tradicionais de classificação utilizam o conceito de classificação competitiva. Primeiro é selecionado um conjunto de classificadores supervisionados disponíveis. As imagens são classificadas utilizando o conjunto de treinamento. Posteriormente, calculam-se métricas de avaliação sobre o conjunto de validação. Por fim, o classificador que produziu os melhores resultados dentre as métricas avaliadas é escolhido para estender a análise a novas amostras (PARREIRA et al., 2015). O uso deste procedimento deve-se ao fato de que diferentes classificadores supervisionados geralmente produzem resultados com diferentes acurácias para um mesmo conjunto de dados de entrada. O problema em usar classificação competitiva é que optando por apenas um classificador pode-se gerar perda de informação para uma determinada classe, já que classificadores geram erros amostrais diferentes para cada classe (PARREIRA et al., 2015).

Na última década vem sendo proposto o uso de combinação de classificadores, que combina as decisões de vários classificadores em apenas um único classificador. Este tipo de classificador resultante é denominado *ensemble* (LEBLANC; TIBSHIRANI, 1996; BREIMAN, 2000). Para a construção de *ensembles* é possível manipular tanto classificadores homogêneos quanto heterogêneos. Utilizar classificadores homogêneos significa usar o mesmo tipo de classificador, ou seja, a mesma técnica de classificação. A combinação neste caso é em nível de dados e características. No nível de dados, pode-se combinar as amostras, enquanto no nível de características pode-se utilizar técnicas de seleção ou extração de atributos. Por sua vez, os classificadores heterogêneos utilizam algoritmos diferentes, mantendo fixos os dados e as características, variando o nível de combinação ou o nível do classificador (KUNCHEVA, 2004).

Otimizar é a tentativa de encontrar a melhor solução a partir de várias soluções existentes, sem a identificação de um algoritmo eficiente que encontre a solução exata em tempo polinomial. Por este motivo é considerada como um problema da classe NP-Difícil (ZIVIANI, 2005). Problemas de otimização tem por finalidade encontrar a solução - conjunto de valores para as variáveis do problema - que forneça o valor máximo, ou mínimo, de uma função objetivo (NETO; BECCENERI, 2009). Algoritmos

exponenciais com técnicas de tentativa e erro, algoritmos de aproximação e heurísticas são técnicas e conhecimentos para buscar a melhor solução de um determinado problema de otimização.

Meta-heurística é um método que resolve problemas de otimização de forma genérica através da aproximação progressiva de um dado problema. É uma das técnicas mais utilizadas atualmente e está dentre as mais completas em nível de busca de solução. Por explorar de maneira abrangente o espaço de busca e por utilizar algoritmos aproximativos para evitar o confinamento da busca em mínimos ou máximos locais é considerada como de alta eficiência na descoberta de soluções (NETO; BECCENERI, 2009) em relação às outras técnicas. Muitos trabalhos utilizam algoritmos de otimização inteligentes, dentre os quais se destacam os bio-inspirados (algoritmos baseados em simulações de ações da natureza) como Algoritmos Genéticos, Colônia de Formigas e Colisão de Partículas.

Um dos mais recentes algoritmos de otimização bio-inspirados é denominado Infestação de Ervas Daninhas - *Invasive Weed Optimization* (IWO) (MEHRABIAN; LUCAS, 2006). Este algoritmo baseia-se na capacidade da infestação dessas ervas, conhecidas como plantas invasoras devido à sua alta capacidade de reprodução. Estudos demonstraram que, mesmo depois de 50 anos de existência de herbicidas, essas ervas continuam se multiplicando em um mesmo campo de análise, gerando espécies mais resistentes aos herbicidas. Isso se deve ao fato destas ervas possuírem capacidade de crescimento rápido, eficiência no uso da água, alta adaptação climática, curto intervalo entre floração e germinação, estruturas para dispersão que germinam em quase todos os substratos úmidos sem uma fertilização específica, alta dormência, alta longevidade e alta produção contínua (MEHRABIAN; LUCAS, 2006).

Nesta tese foi aplicada uma meta-heurística baseada no algoritmo bio-inspirado Infestação de Ervas Daninhas (MEHRABIAN; LUCAS, 2006) para otimização dos parâmetros dos múltiplos-*kernels* utilizados no SVM, em uma combinação de classificadores hierárquica.

1.1 Problema

Dado um conjunto de N amostras rotuladas $\{(\mathbf{x}_i, y_i)\}_{i=1}^N$ em que \mathbf{x}_i é um vetor de atributos de dimensão D e $y_i \in \{-1, 1\}$ é o rótulo da classe daquela amostra, a classificação é feita por uma função $f(\mathbf{x}) : \mathbb{R}^D \rightarrow \mathbb{R}$ onde a amostra é classificada como $y = -1$ se $f(\mathbf{x}) < 0$ ou $y = 1$ se $f(\mathbf{x}) \geq 0$. Classificar uma amostra significa prever sua classe com base em uma função previamente treinada $f(\mathbf{x})$ usando um

conjunto rotulado ou parte dele. Usualmente, o conjunto rotulado é dividido entre conjunto de treinamento e teste. O primeiro é usado para treinar f enquanto o segundo é usado para medir a acurácia da classificação.

Em (PARREIRA et al., 2015) foram realizados vários experimentos de classificações um-contra-um para explorar e melhorar a acurácia de classificação. Foi utilizada uma imagem LANDSAT/TM 5 (30 m) adquirida em 2009. A cena faz parte da Mesorregião do Baixo Amazonas. As classes de entradas foram combinadas em pares. Dois classificadores foram escolhidos para os experimentos: SVM e MAXVER. Cada par de classe foi avaliado por ambos os classificadores, que atribuíram cada amostra para uma classe do par. Ao final, foram usadas várias técnicas para unir os resultados.

Parreira et al. (2015) define um problema multiclasse $X = \{\mathbf{x}_i, \omega\}_{i=1}^N$ com M classes em que $\omega_i \in \Omega = \{\omega_1, \dots, \omega_M\}$ e supõe uma amostra rotulada $\{\mathbf{x}_1, \omega_1\}$ com atributos próximos à distribuição ω_2 . Isso conduz o classificador a errar a classificação, atribuindo à amostra \mathbf{x}_1 a classe ω_2 ao invés de ω_1 . Mas como o classificador poderia fazer a escolha correta pela classe ω_1 baseado nas informações da amostra? A hipótese em (PARREIRA et al., 2015) foi que a combinação dos resultados dos diferentes classificadores para todos os pares de classes poderia corrigir o erro de classificação ocorrido na classificação de um dos pares de classes. Entretanto os experimentos provaram que isto não acontece e de fato o par de classe que produz resultado errôneo é o responsável pela falha na classificação da amostra.

Apesar de o SVM ser na sua essência um classificador um-contra-um, foram criados métodos para resolver SVM multi-classe. A maioria deles tentam combinar vários resultados de SVM binários um-contra-um ou um-contra-todos. Alguns pesquisadores tentaram outras soluções, como modificar os algoritmos SVM para resolver problemas multi-classe em uma única etapa de otimização (WESTON et al., 1999). No entanto, os resultados não mostraram melhora estatística. Eles mostram uma redução de vetores de suporte e, portanto, o tempo de computação.

Conclui-se que para melhorar a precisão da classificação é necessário novas informações a partir da amostra. Existem diversas maneiras de extrair novas informações a partir de uma amostra de pixel de imagens. Pode-se usar informações de contexto de pixel (MATERKA et al., 1998; HARALICK, 1979; HARALICK et al., 1973; COGO, 1994) ou mapear os atributos do pixel para um novo espaço dimensional. A segunda solução pode ser realizada através do desenvolvimento de funções *kernel* (VAPNIK, 1999). Classificadores baseados em funções *kernel* são usados em diversas áreas, tais como bio-informática (SCHÖLKOPF et al., 2004; DING; DUBCHAK, 2001; ZIEN; ONG, 2007;

NAHAR et al., 2007; DAMOULAS; GIROLAMI, 2008), diagnóstico de falha de circuitos analógicos (CAI et al., 2013), classificação de texto (LODHI et al., 2002), reconhecimento de face (SENECHAL et al., 2011; LU et al., 2003) e imagens de sensoriamento remoto (LU; WENG, 2007; NEGRI et al., 2014).

SVM é um classificador baseado em funções *kernel*, amplamente utilizado em sensoriamento remoto. É inicialmente um classificador binário que otimiza um hiperplano entre duas classes de amostras, a fim de maximizar a margem de separação entre elas. Sempre que as classes não são linearmente separáveis, diferentes funções *kernel* podem transformar o espaço de características em uma dimensão mais elevada, onde as classes podem ser separáveis de forma linear (MULLER et al., 2001).

Parâmetros da função *kernel* e do SVM têm impacto sobre o resultado da classificação. Sua definição pode ser feita por qualquer método de otimização, com destaque para a utilização de abordagens de meta-heurísticas como Simulated Annealing (SA) (LIN et al., 2008), Ant Colony Optimization (ACO), Particle Swarm Optimization (PSO) (OMKAR et al., 2007) e Algoritmo Genético (HOWLEY; MADDEN, 2005; HUANG; WANG, 2006). Outros artigos desenvolvem métodos para escolher uma função *kernel* que melhor se enquadra em cada problema (ALI; SMITH-MILES, 2006; AYAT et al., 2005). O uso de múltiplos *kernels* combinados são tratados em (GÖNEN; ALPAYDIN, 2011; ZIEN; ONG, 2007; ZHUANG et al., 2011; VARMA; BABU, 2009; BACH et al., 2004; DAMOULAS; GIROLAMI, 2008).

1.2 Objetivos

1.2.1 Objetivo Geral

O objetivo desta tese é propor um método de classificação estruturado hierarquicamente com Máquinas de Vetores de Suporte (SVM) implementados com múltiplos *kernels* - *Multiple Kernel Learning* (MKL) e otimizados pelo algoritmo bio-inspirado Infestação de Ervas Daninhas (IWO), denominado HSMI (*Hierarchical support vector machine with Multiple kernels optimized by Invasive weed optimization*). O método classificador é composto por um ou mais estágios, os quais são dispostos em arranjos mais complexos de diferentes classificadores individuais de forma a se ajustar à natureza dos dados de entrada. A utilização de um único classificador permite usualmente treinar apenas um tipo de característica do dado de entrada. Aplicar combinação de classificadores permitirá que cada classificador trate um tipo de característica. Ao final os resultados são combinados, melhorando a discriminabilidade e evitando perda de informação de classes.

1.2.2 Objetivos Específicos

Visando atingir o objetivo geral, podem ser relacionados os seguintes objetivos específicos:

- Implementar e analisar o comportamento das classificações por pares de classes.
- Aplicar e analisar classificação competitiva e cooperativa.
- Desenvolver método de classificação binária com otimização de parâmetros do classificador.
- Desenvolver modelo hierárquico para combinação dos classificadores binários.
- Desenvolver método de otimização IWO para as funções multikernels.
- Construir arquitetura paralela para manipulação das Ervas Daninhas.

1.3 Organização da Tese

Esta tese está dividida em cinco capítulos. Uma breve descrição do conteúdo de cada capítulo é apresentada a seguir:

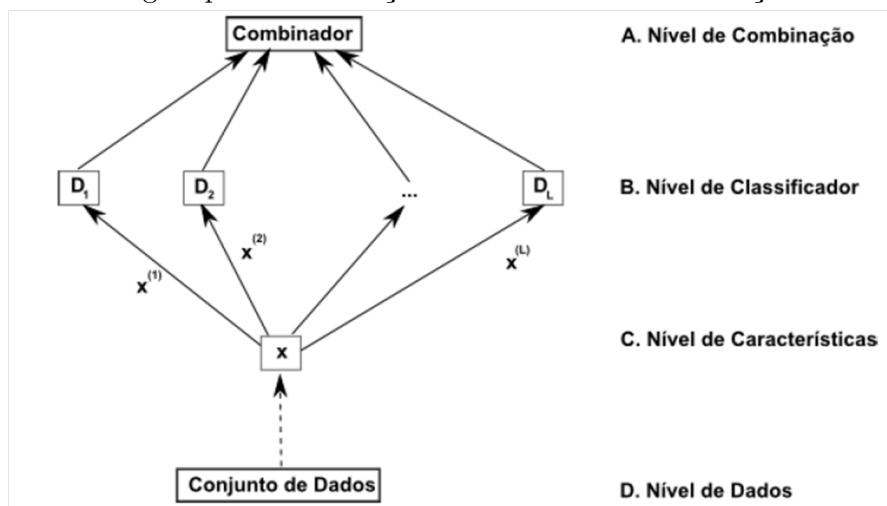
- Capítulo 2 - Base Teórica: faz uma revisão sucinta de alguns conceitos utilizados no desenvolvimento deste trabalho.
- Capítulo 3 - Apresenta a metodologia proposta, bem como a formalização dos algoritmos desenvolvidos e implementação realizada.
- Capítulo 4 - Experimentos e Resultados: descreve os cenários, dados e testes realizados, e resultados encontrados.
- Capítulo 5 - Considerações Finais: apresenta um resumo dos resultados obtidos, trabalhos relacionados, principais contribuições e limitações, e perspectivas futuras deste trabalho.

2 Métodos de Classificação

2.1 Combinação de Classificadores

Ensemble é a combinação das decisões de vários classificadores, identificados na Figura 2.1 por D_1, D_2, \dots, D_L , sendo L o número de classificadores utilizados, em apenas um único resultado (LEBLANC; TIBSHIRANI, 1996; BREIMAN, 2000). Para a construção dos *ensembles* é possível manipular tanto classificadores homogêneos quanto heterogêneos. Utilizar classificadores homogêneos significa usar o mesmo tipo de classificador, ou seja, a mesma técnica de classificação para D_1, D_2, \dots, D_L . Em geral, nestes casos, a combinação é a nível de dados e características. Ao nível de dados pode-se combinar as amostras, como *Bagging* (BREIMAN, 1996) e *Boosting* (FREUND et al., 1996; SCHAPIRE, 1990). Já para combinar características há as técnicas de seleção de atributos (PANTALEÃO, 2012) ou extração de atributos (HARALICK et al., 1973). Por sua vez, os classificadores heterogêneos utilizam técnicas diferentes, mantendo fixos os dados e as características, variando o nível de combinação ou o nível de classificador (KUNCHEVA, 2004). A Figura 2.1 apresenta os diferentes níveis do processo de classificação em que se pode combinar estratégias para construção de *ensembles*, onde $X^{(1)}, X^{(2)}, X^{(L)}$ são sub-espacos de atributos tratados pelos classificadores correspondentes.

Figura 2.1 - Abordagens para a construção de *ensembles* em combinação de classificadores.

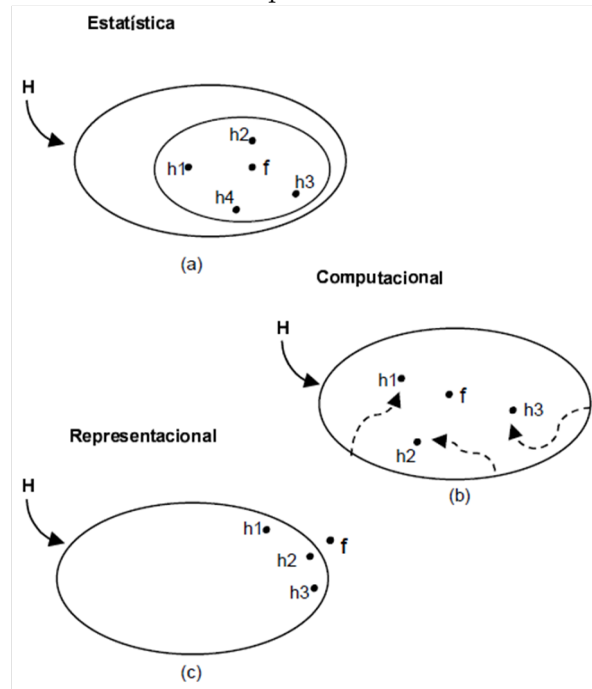


Fonte: Kuncheva (2004).

Existem três razões pelas quais é possível construir bons *ensembles*, são elas: estatística, computacional e representacional (KUNCHEVA, 2004) (Figura 2.2). Suponha

que temos um conjunto de dados H rotulado e diversas hipóteses ($h1$, $h2$, $h3$ e $h4$) com um bom desempenho em H . Um algoritmo de aprendizado faz uma busca no espaço de hipóteses para escolher a melhor precisão. Se o conjunto de treinamento é muito pequeno se comparado ao espaço de hipóteses, sem uma quantidade suficiente de dados, o algoritmo pode escolher diferentes hipóteses com a mesma acurácia no conjunto de treinamento, tem-se um problema estatístico. Mesmo que os classificadores tenham o mesmo desempenho sobre o conjunto de treinamento, eles podem ter diferentes desempenhos na generalização. A Figura 2.2.a mostra uma razão estatística. A curva externa representa o conjunto H ; a curva interna um conjunto de hipóteses com boa precisão no conjunto de treinamento; e o ponto f a hipótese verdadeira.

Figura 2.2 - Três razões para construir bons *ensembles*.



Fonte: Dietterich (2000).

A razão computacional está no fato de que, mesmo existindo um conjunto de treinamento suficiente, pode ser difícil o algoritmo de aprendizado encontrar a melhor hipótese, já que algoritmos de treinamento executam alguma estratégia de busca aleatória que podem levar a diferentes ótimos locais (Figura 2.2.b).

Para o caso representacional, imagine que a hipótese f não pode ser representada por nenhuma das hipóteses do espaço H (Figura 2.2.c). Por exemplo, um classificador

ótimo para dados não-lineares será também não-linear. Caso o espaço dos classificadores possíveis seja restrito a somente classificadores lineares, o classificador ótimo não pertencerá a este espaço. Entretanto, um conjunto (*ensemble*) de classificadores lineares pode aproximar qualquer fronteira de decisão com alguma precisão.

Para (KUNCHEVA, 2004) existem duas principais estratégias em combinação de classificadores: Fusão e Seleção, que na literatura apresentam os sinônimos: classificação cooperativa versus competitiva, abordagem por conjunto versus módulo e topologia múltipla versus híbrida, respectivamente. A **fusão** assume que cada classificador do *ensemble* tem conhecimento sobre todo o espaço de características e combina previsões de múltiplos classificadores para produzir a predição de uma única classe (WEBB et al., 2011). A classificação resulta da opinião coletiva. **Seleção** é um classificador de escolha dinâmica, assume que cada classificador *ensemble* é especialista em uma região do espaço de características (WEBB et al., 2011). Seleção demonstra ser uma arquitetura híbrida, ou seja, existe a escolha de um classificador para rotular uma determinada entrada de dados (KUNCHEVA, 2004).

2.2 Esquemas de Combinação

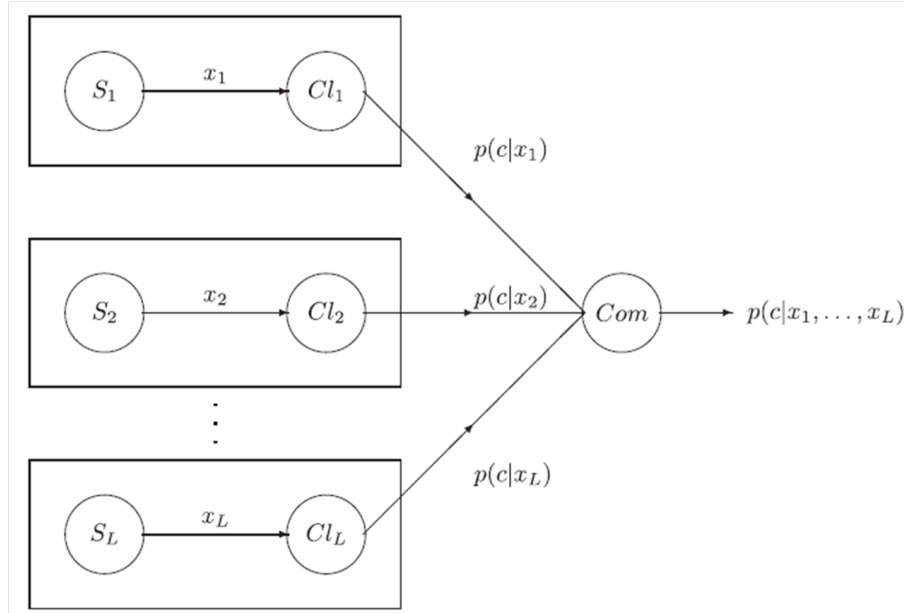
Segundo Webb et al. (2011) os esquemas de combinação podem ser classificados a partir da análise de seus aspectos: espaço de características, nível, grau de formação, forma, estrutura e otimização.

2.2.1 Espaço de Características

São apresentadas duas abordagens sobre o espaço de características. A primeira delas (C1) tem por objetivo analisar diferentes espaços de características utilizando para cada espaço um determinado classificador. Esta abordagem é mostrada na Figura 2.3, em que cada espaço de característica (S_1, S_2, \dots, S_L) é associado a um classificador (Cl_1, Cl_2, \dots, Cl_L) para o qual é estimada a probabilidade a posteriori da classe associada $p(c|x_i)$ pelo espaço de característica S_i . O elemento *Com* representa a regra de combinação.

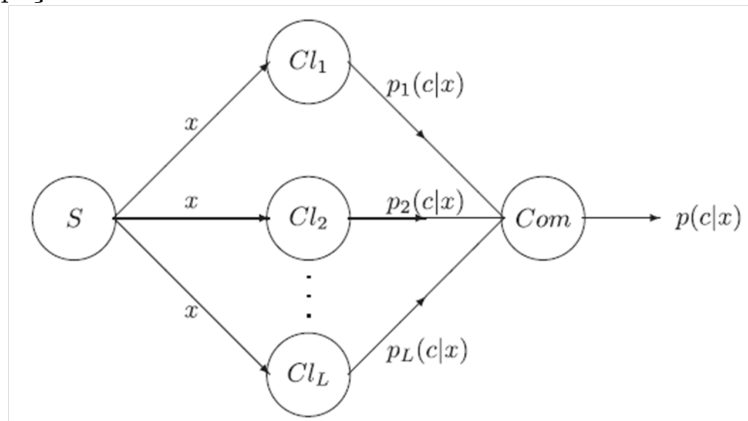
A segunda abordagem (C2) refere-se ao uso do mesmo espaço de característica por um conjunto de classificadores. O combinador *COM* obterá o melhor classificador através da combinação das saídas de cada classificador (Figura 2.4).

Figura 2.3 - C1. Arquitetura de classificação - combinação cooperativa com divisão do espaço de características.



Fonte: Webb et al. (2011).

Figura 2.4 - C2. Arquitetura de classificação - combinação cooperativa a partir do mesmo espaço de características.



Fonte: Webb et al. (2011).

2.2.2 Nível

A combinação de classificadores pode ser feita pela combinação das suas saídas em diferentes níveis. A nível de dados, todos os atributos são tratados por um classificador que toma a decisão da classe. Isso é conhecido na literatura como sistema centralizado. A nível de características, vários classificadores executam uma transformação do espaço de características para um novo espaço de menor dimensão. Entretanto, eles não atribuem uma classe para a amostra. Isso é feito por um combinador que recebe os espaços transformados de todos os classificadores para então tomar a decisão final.

Esquemas de combinação a nível de decisão possuem vários classificadores no primeiro estágio, que atribuem uma classe para cada amostra de entrada. Um combinador, em um segundo estágio, utiliza um dentre diversos métodos para decidir a classe da amostra a partir das classes atribuídas pelos classificadores do primeiro estágio. Os métodos de combinação podem ser histogramas, árvores com pesos ou redes bayesianas, dentre outros.

Kuncheva (2004) acrescenta outros métodos para combinar os resultados do primeiro estágio no esquema a nível de decisão. Nível abstrato: cada classificador possui a informação do rótulo da classe associada ao padrão; *rank*: lista ordenada, em que o topo representa a classe mais provável; e medidas: uma medida/informação dada por um especialista a fim de gerar um grau de confiança da classificação - mapa de confiança.

2.2.3 Grau de Treinamento

Combinadores não treináveis ou fixos são aqueles que não necessitam de um passo de treinamento ou ajuste. São representados pelos métodos de votação pela maioria, máximo, soma, soma ponderada, produto e produto ponderado. Já combinadores treináveis passam por uma etapa de treinamento para a construção de *templates* ou definição dos valores de seus parâmetros.

Para tratar as seções subsequentes, considera-se um vetor d -dimensional de características \mathbf{x} , em que existem c classes possíveis rotuladas $\omega_1, \omega_2, \dots, \omega_c$. Assume-se um conjunto de k classificadores Cl_1, Cl_2, \dots, Cl_k onde cada classificador Cl_i produz $[P_{Cl_i}(\omega_1|\mathbf{x}), P_{Cl_i}(\omega_2|\mathbf{x}), \dots, P_{Cl_i}(\omega_c|\mathbf{x})]$ na saída, em que $P_{Cl_i}(\omega_j|\mathbf{x})$, representa a hipótese de que o vetor \mathbf{x} seja da classe ω_j .

2.2.3.1 Combinadores Não Treináveis

A **Regra da Votação da Maioria** é uma regra de decisão simples em que somente os rótulos atribuídos pelo classificador são considerados e aquele que obtiver mais votos é o vencedor. A classe $\hat{\omega}$ é atribuída pela avaliação da expressão 2.1, em que i indica o índice do classificador e P_{Cl_i} indica o nível de confiança fornecido na saída do classificador Cl_i . A função max count retorna a classe que obteve maior frequência no argumento, já a função arg max retorna a classe ω dentre as classes de entrada do problema Ω correspondente ao maior argumento.

$$\hat{\omega} = \max_{i=1,\dots,k} \text{count} \left[\arg \max_{\omega \in \Omega} [P_{Cl_i}(\omega|\mathbf{x})] \right] \quad (2.1)$$

O método de votação pode ser dividido em dois tipos: os que alteram a distribuição do conjunto de treinamento baseado nos resultados da iteração prévia, como o algoritmo *boosting* (*AdaBoost*), ao qual pode ser atribuído pesos; e os que combinam classificadores treinados no mesmo conjunto, como o algoritmo *bagging* (*bootstrap aggregating*), que geram conjuntos de treino aleatoriamente, podendo produzir classificadores paralelos.

A **Regra do Máximo** é também uma regra de decisão simples, em que a classe com o nível de confiança máximo é vencedora, independente do classificador (Definido por: 2.2).

$$\hat{\omega} = \arg \max_{i=1,\dots,k, \omega \in \Omega} [P_{Cl_i}(\omega|\mathbf{x})] \quad (2.2)$$

A **Regra da Soma** é baseada no somatório dos níveis de confiança fornecidos pelos classificadores. Os níveis de confiança são somados para cada classe. A classe cuja soma resultante for máxima é atribuída a amostra \mathbf{x} (Definido por: 2.3).

$$\hat{\omega} = \arg \max_{\omega \in \Omega} \left[\sum_{i=1}^k P_{Cl_i}(\omega|\mathbf{x}) \right] \quad (2.3)$$

A **Regra da Soma Ponderada** é a soma ponderada na saída dos classificadores. Dá-se maior peso aos classificadores mais competentes. O nível de confiança de cada classificador é multiplicado por um peso w_i , e os pesos são específicos para cada classificador (Definido por: 2.4).

$$\hat{\omega} = \arg \max_{\omega \in \Omega} \left[\sum_{i=1}^k w_i P_{Cl_i}(\omega|\mathbf{x}) \right] \quad (2.4)$$

A **Regra do Produto** é baseada na multiplicação dos níveis de confiança fornecidos pelos classificadores. Os níveis de confiança são multiplicados para cada classe, e a classe que o produto resultante for máxima ganha a rotulação (Definido por: 2.5).

$$\hat{\omega} = \arg \max_{\omega \in \Omega} \left[\prod_{i=1}^k P_{Cl_i}(\omega|\mathbf{x}) \right] \quad (2.5)$$

A **Regra do Produto Ponderado** é definida pela multiplicação dos níveis de confiança dos classificadores. O nível de confiança de cada classificador é elevado a um coeficiente w_i , correspondente ao classificador (Definido por: 2.6).

$$\hat{\omega} = \arg \max_{\omega \in \Omega} \left[\prod_{i=1}^k [P_{Cl_i}(\omega|\mathbf{x})]^{w_i} \right] \quad (2.6)$$

2.2.3.2 Combinador Treinável

Combinador treinável foi proposto por (KUNCHEVA, 2004) para combinação de classificadores com saídas contínuas. Utiliza-se de um perfil de decisão mais comum para cada classe ω_j , denominado DT_j (Equação 2.7), que será a média de todos os perfis de decisão $DP(R_k)$ calculado em uma etapa de treinamento.

$$DT_j = \frac{1}{N_j} \sum_{\substack{CT_k \in \omega_j \\ CT_k \in CT}} DP(R_k) \quad (2.7)$$

onde R_k é um registro pertencente a ω_j do conjunto de treinamento CT e $DP(\mathbf{x})$ (Equação 2.8) é uma matriz representada pelas saídas (grau de confiança) para cada c classes de nCl classificadores individuais Cl_i de um dado padrão de entrada \mathbf{x} . E r_j é o número de registros de ω_j em CT .

$$DP(\mathbf{x}) = \begin{bmatrix} d_{1,1}(\mathbf{x}) & \cdots & d_{1,j}(\mathbf{x}) & \cdots & d_{1,c}(\mathbf{x}) \\ \vdots & & \vdots & & \vdots \\ d_{i,l}(\mathbf{x}) & \cdots & d_{1,j}(\mathbf{x}) & \cdots & d_{1,c}(\mathbf{x}) \\ \vdots & & \vdots & & \vdots \\ d_{nCl,1}(\mathbf{x}) & \cdots & d_{nCl,j}(\mathbf{x}) & \cdots & d_{nCl,c}(\mathbf{x}) \end{bmatrix} \quad (2.8)$$

Um valor de similaridade $\mu_j(\mathbf{x})$, que pode ser a distância euclidiana, será calculado entre o perfil de decisão $DP(\mathbf{x})$ do padrão a ser classificado e do DT_j . Por fim, x será associado à classe que fornecer o maior valor de $\mu_j(\mathbf{x})$ (Equação 2.9 e Equação 2.10).

$$\mu_j(\mathbf{x}) = S(DP(\mathbf{x}), DT_j), j = 1..c \quad (2.9)$$

$$S(DP(\mathbf{x}), DT_j) = 1 - \frac{1}{nCl \times c} \sum_{i=1}^{nCl} \sum_{k=1}^{nCl} [DT_j(i, k) - d_{i,k}(\mathbf{x})]^2 \quad (2.10)$$

2.2.4 Forma

A forma diz respeito ao tipo dos classificadores utilizados. Forma comum indica que todos os classificadores do conjunto são do mesmo tipo, isto é, ou são todos redes neurais, ou todos árvores de decisão, ou todos de um outro tipo. Essa combinação de forma comum também é conhecida como combinação homogênea de classificadores. Essa forma é escolhida para se obter interoperabilidade, implementabilidade e adaptabilidade.

Quando diferentes tipos de classificadores são utilizados, o esquema é caracterizado como de forma dissimilar. Podem ser combinados algoritmos como redes neurais, vizinhos mais próximos, árvores de decisão, dentre outros. Esta forma também pode ser classificada como combinação heterogênea.

2.2.5 Estrutura

A estrutura paralela define que os resultados dos classificadores que compõe a combinação são repassados para o combinador de classificadores, que por sua vez gera a decisão final. O modo serial depende dos resultados que são gerados pelos classificadores em uma etapa anterior. As saídas são sequenciais, tendo uma análise prévia

sobre as classes. Na estrutura hierárquica, os classificadores são combinados numa hierarquia de uma maneira semelhante à classificadores de árvore de decisão.

2.2.6 Otimização

É importante que se tenha uma maneira de escolher e otimizar a combinação de um conjunto fixo de classificadores base, que terá como entrada a resposta de vários classificadores e apenas uma decisão final como saída.

A otimização pode ser feita apenas no combinador final, ou para todos os classificadores constituintes, buscando em ambos os casos maximizar a performance do processo de classificação. Esse processo de otimização deve tratar com grande número de parâmetros e de problemas não lineares. Para essa perspectiva, pode-se destacar o uso de metas-heurísticas.

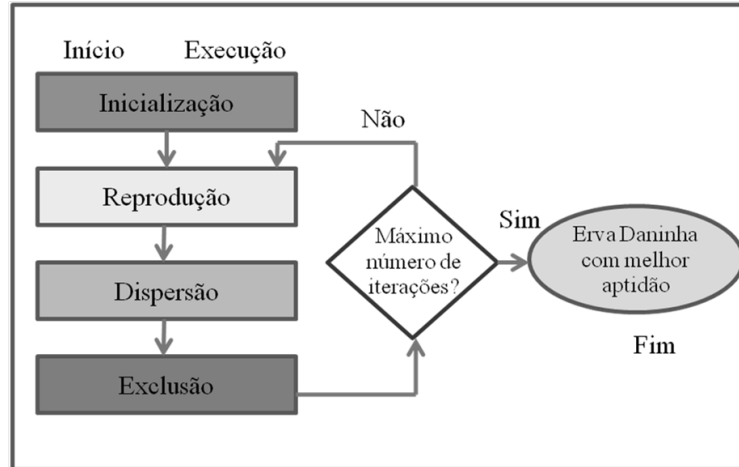
Meta-heurística é uma das técnicas mais utilizadas atualmente e está dentre as mais completas em nível de busca de solução. Por explorar de maneira abrangente o espaço de busca e por utilizar algoritmos aproximativos para evitar o confinamento da busca em mínimos ou máximos locais é considerada como de alta eficiência na descoberta de soluções (NETO; BECCENERI, 2009) em relação às outras técnicas.

Vários são os algoritmos de meta-heurística: Algoritmos Genéticos, Recozimento Simulado, Colônia de Formigas, Colisão de Partícula, Infestação de Ervas Daninhas (IWO), dentre outros. Nesta sub-seção será abordado o método IWO, meta-heurística de otimização utilizada na metodologia de combinação de classificadores deste trabalho.

2.2.6.1 Método de Otimização por Meta-Heurística: Infestação de Ervas Daninhas (IWO)

Para se entender o algoritmo de ervas daninhas IWO (MEHRABIAN; LUCAS, 2006), são definidas algumas inter-relações entre o problema a ser otimizado e o processo natural de infestação de ervas daninhas. Por erva daninha entende-se uma solução do problema. A posição da erva daninha na sua colônia é definida pelo vetor de variáveis do problema. A aptidão de uma erva daninha, descrita na sequência do trabalho, é equivalente ao valor da função dada em que a erva daninha encontra-se em uma posição na colônia. A aptidão representa um conjunto de valores para as variáveis do problema. A metodologia de infestação aplicada pode ser visualizada na Figura 2.5.

Figura 2.5 - Metodologia desenvolvida.



Primeiramente é configurado um conjunto de parâmetros que regem o processo de evolução do algoritmo. O processo é iniciado com a criação da primeira geração de ervas daninhas. A partir daí é realizado um ciclo de iterações para representar a evolução da colônia. Cada iteração realiza a reprodução das ervas daninhas gerando as sementes novas, que em seguida são dispersadas e se tornam parte da população. A exclusão de ervas daninhas elimina as menos aptas dentre toda população mantendo a quantidade de indivíduos limitada conforme os parâmetros iniciais. O ciclo é repetido até alcançar o número de iterações definido nos parâmetros. Ao final, com a quantidade máxima de iterações alcançada, espera-se encontrar a solução ótima global através da erva com melhor aptidão. A seguir são detalhadas cada uma das etapas do algoritmo.

Inicialização de Parâmetros

A escolha dos parâmetros de inicialização é fundamental para que se encontre uma convergência para a solução ótima global. Esses parâmetros são:

- $Pop_{initial}$ = População inicial;
- Pop_{max} = População máxima;
- $iter_{max}$ = Número máximo de iterações;
- $D_{problema}$ = Dimensão do problema;
- ns_{max} = Número máximo de sementes que uma erva daninha pode gerar em uma reprodução;

- ns_{min} = Número mínimo de sementes que uma erva daninha pode gerar em uma reprodução;
- mod = Módulo de não linearidade para cálculo da variância;
- σ_{final} = Desvio padrão final;
- $\sigma_{inicial}$ = Desvio padrão inicial;
- $A[\]$ = Área para dispersão da população inicial, onde [dimensão][0=limite mínimo; 1 =limite máximo].

Os parâmetros auxiliares são informações geradas durante a execução do algoritmo e são apresentados a seguir.

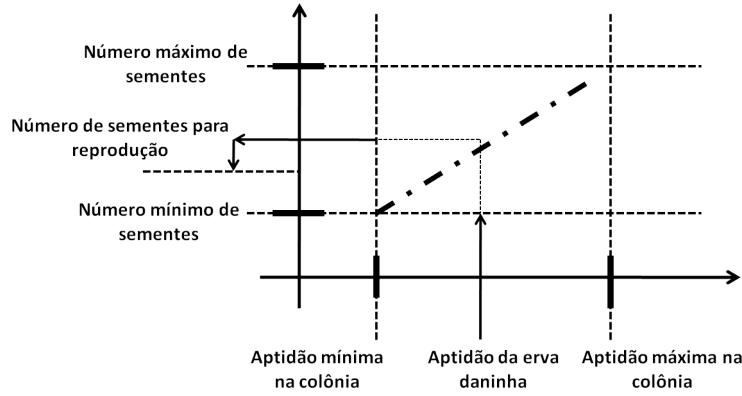
- F_{min} = Maior aptidão encontrada na população;
- F_{max} = Menor aptidão encontrada na população;
- F_{weed} = Aptidão de uma erva daninha;
- $iter$ = Iteração corrente;
- ns_i = Número de sementes para reprodução de uma erva daninha i ;
- σ_{iter} = Variância aplicada em uma dada iteração.

Reprodução

A reprodução ocorre no momento que cada erva daninha deve replicar e gerar novas sementes. Calcula-se, através da equação da reta, inscrita no retângulo da Figura 2.6, o número de sementes a ser gerado em cada reprodução por cada erva daninha, dada sua aptidão em relação às aptidões máxima e mínima na população naquela iteração (Equação 2.11) (MEHRABIAN; LUCAS, 2006). Destaca-se que nesse momento as sementes ainda não possuem posição na colônia.

$$ns_i = \left(\frac{F_{weed} - F_{min}}{F_{max} - F_{min}} \right) (ns_{max} - ns_{min}) + ns_{min} \quad (2.11)$$

Figura 2.6 - Procedimento de produção de sementes em uma colônia.



Fonte: Mehrabian e Lucas (2006).

Dispersão

A dispersão é baseada no cálculo do desvio padrão σ_{iter} de uma determinada iteração que dependerá de $iter_{max}$, índice de modulação não linear mod , desvio padrão inicial $\sigma_{inicial}$ e desvio padrão final σ_{final} (Equação 2.12) (MEHRABIAN; LUCAS, 2006). Após calcular o desvio padrão de uma iteração i , é necessário gerar um vetor de valores randômicos de uma distribuição normal centrada na posição $x_{i,0}$ da erva daninha pai. Cada um desses é o valor de uma variável do vetor de solução conforme Equação 2.13, onde rnd é um valor aleatório entre 0 e 1 que é aplicado à equação para gerar o valor de x_i de forma a seguir uma distribuição normal.

$$\sigma_{iter} = \frac{(iter_{max} - iter)^{mod}}{(iter_{max})^{mod}} (\sigma_{inicial} - \sigma_{final}) + \sigma_{final} \quad (2.12)$$

$$x_i = x_{i,0} \pm \sigma_{iter} \sqrt{-2\ln(rnd)} \quad (2.13)$$

Exclusão Competitiva

O parâmetro de população máximo definido inicialmente é determinante para a continuidade de novas gerações. Após as novas ervas daninhas estarem distribuídas na colônia, elas são incorporadas à população total, o que pode tornar o número de indivíduos da população superior ao parâmetro Pop_{max} . Se ocorrer essa superpopulação, a exclusão das ervas com menor aptidão é realizada até que se mantenha o limite de população Pop_{max} . Com isto, apenas os sobreviventes poderão se reproduzir nas

próximas iterações.

2.3 Classificador por Indução de Árvore de Decisão (IAD)

O classificador por Indução de Árvore de Decisão (IAD) é um classificador supervisionado não-linear, não-paramétrico, com método de seleção de atributos embutido (embutido), onde a seleção faz parte do aprendizado. Uma árvore de decisão representa uma tabela de decisão ou conjunto de regras na forma de relação IF-THEN, em que cada nó folha (nó de decisão) é o resultado de uma regra, representado por uma classe (QUINLAN, 1993). As Árvores de Decisão (ADs) utilizam a estratégia dividir-para-conquistar. Uma das vantagens de sua utilização é que não assumem nenhuma distribuição particular para os dados e os atributos em questão podem ser quantitativos, categóricos ou ausentes.

2.3.1 Representações dos Nós de Decisão

Na maioria dos casos, para um melhor aproveitamento da técnica de classificação a ser utilizada é necessário transformar os dados. As transformações mais comumente utilizadas são: normalização, agregação, codificação numérico-categórica e codificação-numérica, construção de atributos e correlação de prevalência (SOARES, 2007).

A forma de representação dos nós pode influenciar de maneira decisiva no desempenho das árvores de decisão induzidas. Dependendo do tipo de atributo, existem diferentes tipos de representação dos nós para o particionamento dos dados. A seguir, são apresentadas algumas formas de representação considerando atributos quantitativos, categóricos e ausentes.

Atributos Quantitativos - a estratégia empregada é realização de testes através de um ponto de referência na distribuição dos valores. Para isso, devem-se ordenar todos os valores do atributo em questão. Posteriormente, é escolhido o ponto de referência. Utiliza-se então uma busca exaustiva para testar todas as partições possíveis geradas de um ponto qualquer de referência para cisão. Qualquer ponto intermediário entre dois valores diferentes e consecutivos dos valores observados no conjunto de treino pode ser utilizado como possível ponto de referência. Geralmente utiliza-se o valor médio entre dois valores diferentes e consecutivos.

Atributos Categóricos - os atributos contínuos podem ser discretizados e, se possuírem poucos intervalos numéricos, podem ser tratados como categóricos. Algumas abordagens de tratamento das categorias: gerar um ramo para cada categoria do

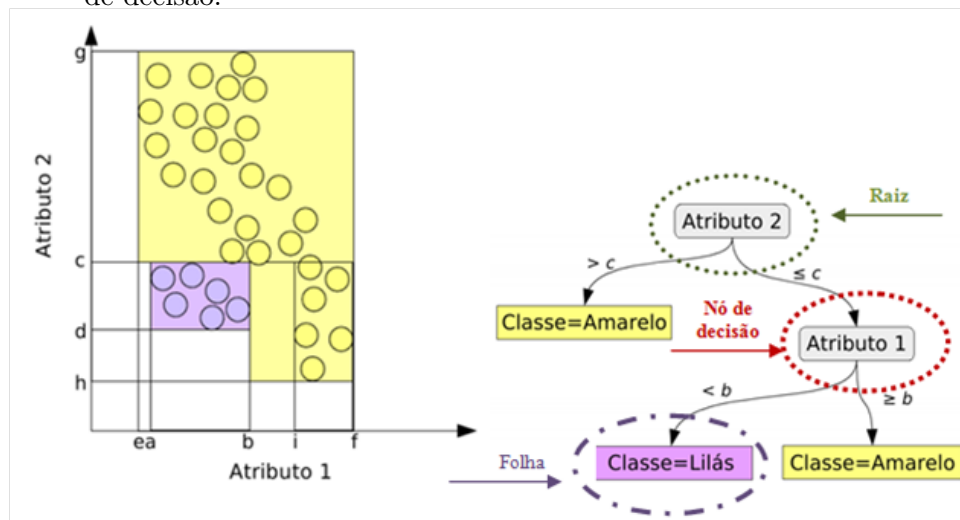
atributo; gerar nós binários - Solução de Hunt (CLS) (HUNT et al., 1966); gerar partição de atributos categóricos ordinais; gerar agrupamento de valores em dois ou mais conjuntos.

Atributos com Valores Ausentes - algumas técnicas aplicadas para correção dos valores ausentes são (HAN; KAMBER, 2006): exclusão de casos (ListWise Deletion); preenchimento manual de valores; preenchimento com valores globais constantes; preenchimento com medidas estatísticas; e preenchimento com métodos de Mineração de Dados. Técnicas e métodos de tratamentos de valores ausentes podem ser consultados em Parreira (2010).

2.3.2 Construção de uma Árvore de Decisão

O nó em uma AD é um nó teste em um atributo. Cada ramo representa um possível valor para este atributo e cada nó folha é determinado por uma classe. Um percurso da raiz da árvore até um nó folha é uma regra. A representação de uma árvore de decisão pode ser visualizada na Figura 2.7. Vale ressaltar que uma AD representa a disjunção de conjunções de restrições nos valores dos atributos, ou seja, cada ramo na árvore é uma conjunção de condições, já as regras (conjunto de ramos) são disjuntas.

Figura 2.7 - (a) Exemplo de partição dos espaços de atributos (b) Exemplo de uma árvore de decisão.



Fonte: Santos, R. D. C. (2011).

De acordo com (RUSSEL; NORVIG, 2004) qualquer função booleana pode ser escrita como uma AD, o que poderia causar uma AD exponencialmente grande. Por exem-

plo, sobre um conjunto de dados binários, uma **função paridade** que retorna 1 se e somente se um número par de entradas é igual a 1; ou a **função maioria**, que retorna 1 se mais da metade de suas entradas é igual a 1. Provavelmente para estes casos a AD não seria uma boa representação. Importante observar que uma tabela da verdade tem 2^n linhas, por que para cada teste existem n atributos. Baseado neste raciocínio, agrupar e classificar uma tabela de dados através do método da força bruta, isto é, gerar uma tabela de decisão com todas as combinações possíveis de todos os atributos em questão relacionados com suas classes, não é a melhor opção para minimizar o esforço e os possíveis erros gerados na utilização de técnicas que visam outras estratégias para classificação.

a) Indução de Árvores de Decisão (IAD)

Uma das maneiras mais simples de se pensar em construir uma AD é construir uma árvore que possua um caminho da raiz até a folha para cada exemplo de uma tabela que teste cada atributo. Na existência de exemplos ou descrições iguais, a AD apresentará uma classificação correta. O problema desta maneira simples de representação é que esta árvore irá memorizar as observações e, portanto, não estará extraindo nenhum conhecimento a partir dos exemplos. Portanto, o objetivo do método de classificação por AD não é só encontrar uma árvore que represente todo o espaço de atributos/exemplos e sim adotar métodos de indução que classifiquem novos exemplos com menores árvores possíveis e que tenham maior poder de generalização.

O *Top-Down Induction of Decision Tree* (TDIDT) é um algoritmo utilizado como base para muitos algoritmos de indução de árvores de decisão, dentre eles os mais conhecidos são ID3 (QUINLAN, 1986), C4.5 (QUINLAN, 1993) e CART (BREIMAN et al., 1984). De forma geral, o algoritmo gera regras de decisão de uma AD através de sucessivas divisões dos registros (R), levando em consideração os valores dos atributos.

Considera-se que CT é um conjunto de treinamento, assumindo que as classes sejam denotadas por C_1, C_2, \dots, C_k , o algoritmo TDIDT é baseado nos seguintes passos (REZENDE, 2003):

1. CT contém um ou mais registros, todos pertencentes à classe C_j . Assim, a árvore de decisão para CT é um nó folha que identifica a classe C_j .
2. CT não contém registros. A árvore de decisão também é um nó folha, mas

a classe associada deve ser determinada por uma informação adicional a CT . Por exemplo, a classe mais frequente para o nó-pai desse nó pode ser utilizada.

3. CT contém registros pertencentes a mais de uma classe. Assim, a proposta é dividir CT em subconjuntos de registros que são ou tendem a ser coleções de registros com classes únicas. Para isso, é necessário escolher um atributo preditivo A . Cada indutor tem sua própria técnica para escolher o atributo a ser utilizado no teste. Considera-se Q_1, Q_2, \dots, Q_n como os possíveis resultados do teste. CT é então particionado em subconjuntos CT_1, CT_2, \dots, CT_n , onde CT_i contém todos os registros de CT que têm resultado Q_i para o atributo A . A árvore de decisão para CT consiste de um nó de decisão identificando o teste sobre o atributo A , e uma aresta para cada possível resultado, ou seja, ‘ a ’ arestas. Uma ressalva: pode ser considerado um subconjunto de atributos ao invés de apenas um atributo A .

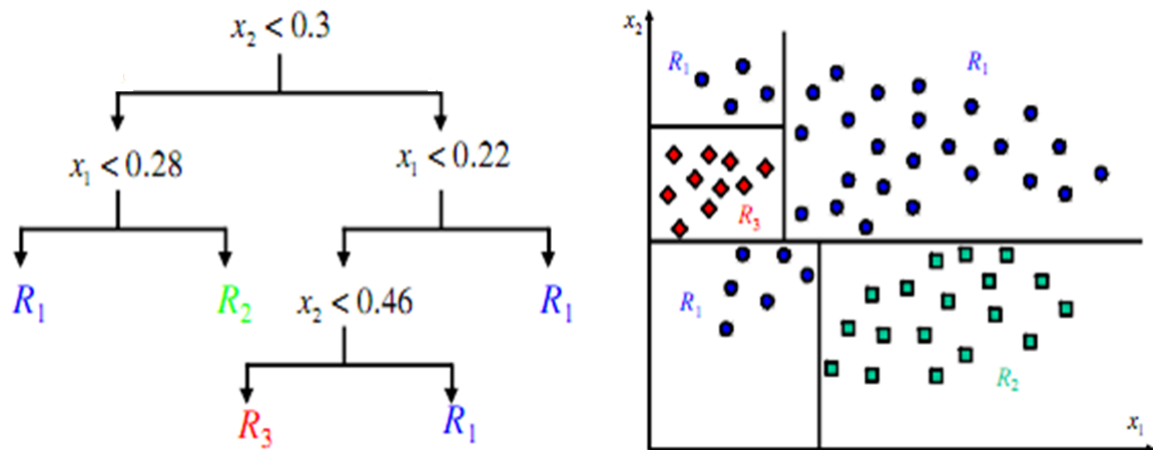
Os passos 1, 2 e 3 são aplicados recursivamente para cada subconjunto de exemplos CT_i . TDIDT é um algoritmo recursivo de busca exaustiva, que visa obter o atributo mais eficiente ou mais importante e assim permita uma melhor divisão do conjunto de registros em subconjuntos, de modo que todos os registros sejam classificados.

A maioria dos algoritmos de indução de árvores de decisão trabalha com funções de divisão univariável, ou seja, cada nó interno da árvore é dividido de acordo com um único atributo. Este tipo de árvore é denominado Árvore Comum de Classificação Binária (Figura 2.8). Nesse caso, o algoritmo tenta encontrar o melhor atributo para realizar essa divisão. A superfície de decisão é ortogonal ao eixo de um atributo, ou seja, o hiperplano de decisão intercepta apenas um dos eixos coordenados. A Equação 2.14 apresenta este modelo, onde x_j é o valor de um dado atributo. O eixo t intercepta o hiperplano em w_0 .

$$f(\mathbf{x}|t, w_0) = x_j + w_0 = 0 \quad (2.14)$$

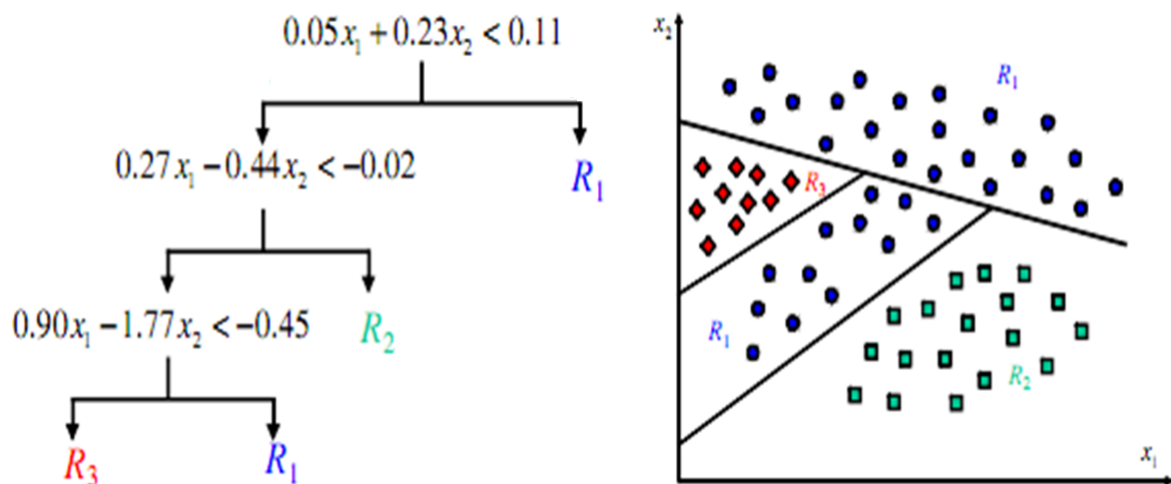
Outro tipo de partição é a Árvore de Decisão Oblíqua, Figura 2.9, que produz hiperplanos não-ortogonais (BREIMAN et al., 1984; MURTHY et al., 1994). Esse modelo é apresentado na Equação 2.15, onde nd é o nó de decisão, \mathbf{w}_{nd} é o vetor de pesos do hiperplano (d -dimensional), w_{nd0} o valor limitante quando \mathbf{x} está projetado sobre \mathbf{w}_{nd} e T indica o vetor transposto (MEDEIROS et al., 2011).

Figura 2.8 - Exemplo de uma Árvore Comum de Classificação Binária.



Fonte: Medeiros et al. (apud DUDA et al., 2001)

Figura 2.9 - Exemplo de uma Árvore de Decisão Oblíqua.



Fonte: Medeiros et al. (apud DUDA et al., 2001).

$$f_m(\mathbf{x}|\mathbf{w}_{nd}, w_{nd0}) = \mathbf{w}_{nd}^T \mathbf{x} + w_{nd0} = 0 \quad (2.15)$$

b) Determinação da Classe Associada à Folha

Quando se encontra um nó folha, durante a indução de uma árvore de decisão, é necessário determinar qual classe deverá estar associada a ele. Essa determinação pode estar associada à classe que minimiza a taxa de erro da classificação ou à classe que minimiza os custos da classificação (GARCIA, 2003).

Atribuição da Classe mais Provável

A atribuição da classe mais provável à folha refere-se à classe que possui o máximo de exemplos associados a ela (FONSECA, 1994), conforme apresentado na Equação 2.16, onde c é o número de classes, r o número total de registros na folha, r_j o número de registros da classe j na folha.

$$\hat{\omega} = \arg \max_j (P_j) = \max_j \frac{r_j}{r} ; j = 1..c \quad (2.16)$$

Determinação Baseada na Noção de Custos

A determinação de uma classe baseada na noção de custos está associada à minimização dos custos gerados ao se adotar uma determinada classe e não a probabilidade do erro resultante. A Equação 2.17 apresenta o cálculo deste custo, onde c é o número de classes, p_i a probabilidade da classe i e $C_{i,j}$ o valor da linha i , coluna j da matriz de custos. Isto significa que não será escolhida necessariamente uma classe que apresente maior probabilidade de existência dentre os registros, mas sim a que terá menor custo de adoção para uma folha. A matriz de custos define o custo do número de erros diante de um número de casos testados para cada classe.

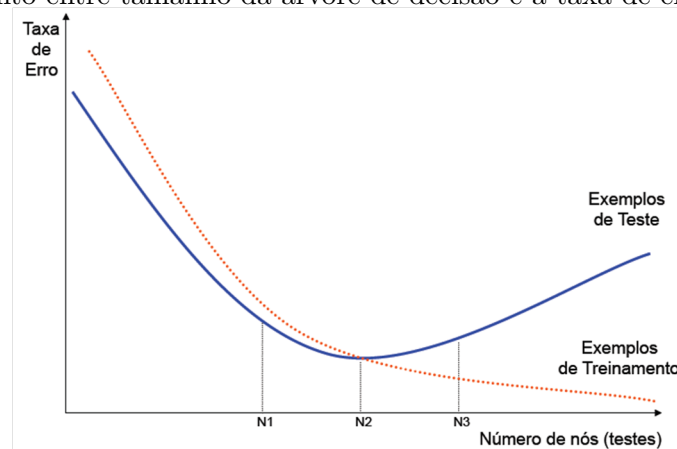
$$custo(m) = \sum_{i=1}^N p_i C_{i,j} \quad (2.17)$$

2.3.3 Limitações na Dimensão das Árvores de Decisão (Poda)

O número de parâmetros de uma árvore de decisão cresce linearmente com o número de exemplos, assim uma árvore maior é induzida de forma a superajustar os exemplos (os dados de treinamento), o que é denominado *overfitting* (super-especialização).

A fim de resolver esse problema, a árvore então é podada até obter uma árvore menor (mais simples). A Figura 2.10 mostra o impacto do *overfitting* no aprendizado por AD. O eixo horizontal indica o número total de nós na AD à medida que a árvore é construída. O eixo vertical indica o erro nas previsões feitas pela árvore. Na linha pontilhada é mostrada a taxa de erro da AD calculada sobre o conjunto de treinamento, enquanto que na linha sólida é mostrada a taxa de erro calculada sobre um conjunto de teste (REZENDE, 2003).

Figura 2.10 - Impacto do Overfitting em aprendizagem por árvore de decisão - Relacionamento entre tamanho da árvore de decisão e a taxa de erro.



Fonte: Rezende (2003).

O treinamento do classificador por árvore de decisão binária tradicional é comumente composto de duas operações: crescimento da árvore (*tree growing*) e poda de árvore (*tree pruning*). Primeiramente, o conjunto de dados precisa ser dividido em *growing set* e *pruning set*. Apenas os dados do *growing set* são utilizáveis para aprender as regras. O *pruning set* é utilizado para validação do aprendizado. O custo computacional é maior, mas este processamento adicional usualmente é compensado pela melhor generalização no aprendizado.

A poda tanto pode ser realizada durante o crescimento (*pre-pruning*) como após o crescimento (*pos-pruning*). Conjuntos independentes de dados podem ser utilizados em cada uma das fases: *growing set* para a fase *tree growing* e *pruning set* para a fase de *post-pruning*.

Pre-pruning faz a poda no momento em que as regras são geradas, ou seja, estabelece uma condição de parada quando uma regra é adicionada. Esta condição pode ser: significância estatística, ganho de informação, redução de erro ou outra métrica

qualquer. Alguns critérios de *pre-pruning*: 1) caso o nó não seja puro, isto é, somente poderá ser dividido se nele houver instâncias rotuladas a duas ou mais classes distintas; 2) caso a porcentagem mínima de instâncias estiver dentro da tolerância definida pelo usuário; e 3) caso a porcentagem mínima de diminuição de entropia estiver dentro do valor definido pelo usuário, quando estiver utilizando ganho de informação como condição de parada. Se a medida encontrada estiver abaixo de um valor limite (*threshold*), o particionamento é interrompido e a árvore para aquele subconjunto é composta apenas pela folha mais apropriada. Evita-se assim a construção de subárvores que não serão utilizadas na árvore final. Sua desvantagem é a dificuldade em se determinar a condição de parada, o que pode gerar *underfitting*.

Já *pos-pruning* faz a poda somente quando toda a árvore de decisão já foi gerada, possui a vantagem de ser mais precisa e confiável (QUINLAN, 1986), porém para crescer e podar é mais lenta que *pre-pruning*. Existem na literatura várias técnicas de poda, as quais utilizam estimativas de erro num determinado nó, são elas: *Reduced Error Pruning* (REP), *Cost Complexity Pruning* (CCP) e *Error Based Pruning* (EBP). REP é uma das técnicas mais simples, proposta por (QUINLAN, 1986), descrita no livro de (ROKACH; MAIMON, 2008), que propõe obter estimativas de erro a partir de um conjunto de validação independente do conjunto de treino utilizado para construir uma árvore, reduzindo o volume de informação disponível para crescer a árvore. Um algoritmo de poda basicamente percorre a árvore em profundidade, posteriormente calcula para cada nó uma estimativa pessimista do erro no nó e soma as estimativas dos erros dos nós descendentes. Caso a estimativa do erro no nó é menor ou igual à soma das estimativas de erros dos nós descendentes, o nó é transformado em folha (RIBEIRO, C. H., 2005).

2.4 Máquina de Vetores de Suporte - *Support Vector Machine* (SVM)

Seja uma função $f(\mathbf{x}) : R^m \rightarrow R$ um classificador que atribui as amostras \mathbf{x} à classe $y = -1$ se $f(\mathbf{x}) < 0$ ou $y = 1$ se $f(\mathbf{x}) > 0$, onde $\mathbf{x} = (x_1, x_2, \dots, x_m)$ é um vetor de atributos de dimensão m . E F um conjunto de funções $f(\mathbf{x})$. Treinar um classificador - supervisionado - significa escolher f para um conjunto de amostras rotuladas $(\mathbf{x}_i, y_i) \in X$ tal que f classifique corretamente o maior número de amostras em suas classes y_i .

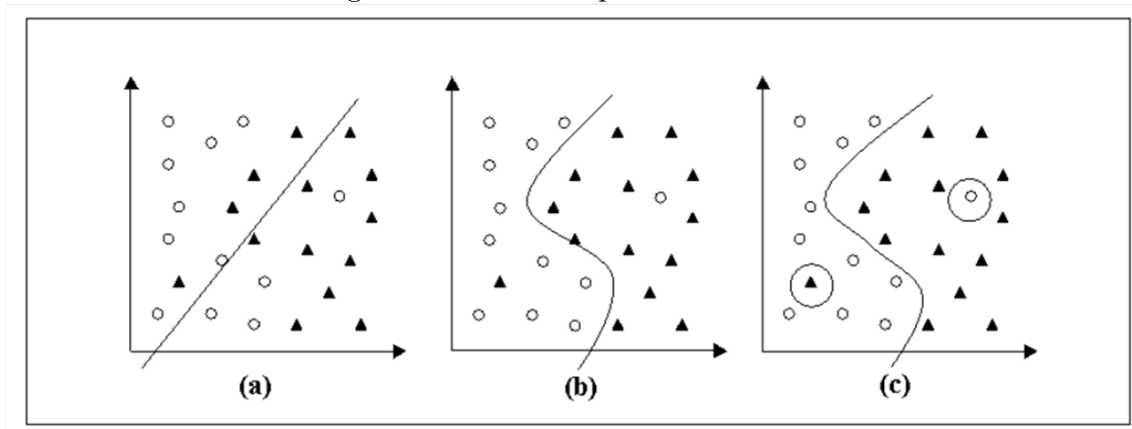
Define-se como erro empírico R_{emp} de uma classificação a razão entre o número de amostras classificadas erroneamente durante o treinamento e número total de amostras treinadas. A Figura 2.11.c apresenta uma função de classificação f que consegue distinguir com detalhes todas as amostras do conjunto de treinamento.

Portanto, ela possui erro empírico nulo, pois não comete erros de classificação durante o treinamento. Entretanto, ao usar essa função treinada para classificar um novo conjunto de amostras rotuladas, pequenas variações nas amostras produzirão erros de classificação. Isso indica que seu poder de generalizar outras amostras além das de treinamento é baixo. Essa característica de generalização é medida pelo erro funcional R_f (VAPNIK, 1999).

Observando a Figura 2.11.a as amostras são separadas em suas classes por uma função mais simples, uma reta. As amostras de treinamento são classificadas com erros, produzindo um erro empírico $R_{emp} > 0$. Porém, para outros conjuntos de entrada, essa função tem menor sensibilidade a ruídos nas amostras de entrada, reduzindo o erro funcional. Isso indica melhor poder de generalização.

A Figura 2.11.b mostra uma função que busca equilibrar os erros empírico e funcional. Assim, ela permite erros durante o treinamento para melhorar o poder de generalizar outras amostras. O classificador Máquina de Vetores de Suporte (do inglês *Support Vector Machine*, SVM) define um hiperplano a partir de uma função de classificação $f(\mathbf{x})$ que separa o espaço de atributos em dois conjuntos disjuntos, um para $f(\mathbf{x}) < 0$ e outro para $f(\mathbf{x}) \geq 0$. A função f deve ser tal que a distância entre o hiperplano e as amostras de cada classe seja maximizada. A essa distância dá-se o nome de margem de separação.

Figura 2.11 - Erro empírico e funcional.



Fonte: Lorena e Carvalho (2007).

O método SVM tem recebido grande atenção nos últimos anos devida sua capacidade de generalização e independência da distribuição dos dados (BRUZZONE; PERSELLO,

2009). O hiperplano de separação é uma função definida por:

$$f(\mathbf{x}) = \langle \mathbf{w}, \phi(\mathbf{x}) \rangle + b \quad (2.18)$$

onde \mathbf{w} representa o vetor ortogonal ao hiperplano de separação, $b/\|\mathbf{w}\|$ é a distância do hiperplano à origem do sistema de coordenadas e $\phi(\mathbf{x})$ é uma função adotada, caso necessário, para remapear os vetores em um novo espaço (VAPNIK, 1998). Os parâmetros da Equação 2.18 são obtidos a partir da solução do problema de otimização quadrática, onde l é o número de amostras de treinamento (THEODORIDIS; KOUTROUMBAS, 2009):

$$\max \left(\sum_{i=1}^l \lambda_i - \frac{1}{2} \sum_{i=1}^l \sum_{j=1}^l \lambda_i \lambda_j y_i y_j \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle \right) \quad (2.19)$$

sujeito a

$$\begin{cases} 0 \leq \lambda_i \leq \zeta, i = 1, \dots, l \\ \sum_{i=1}^l \lambda_i y_i = 0 \end{cases} \quad (2.20)$$

que consiste em encontrar λ_i para então calcular \mathbf{w}^T e b usando:

$$\mathbf{w} = \sum_{i \in SV} \lambda_i y_i \mathbf{x}_i \quad (2.21)$$

$$b = \frac{1}{\#SV} \left(\sum_{i \in SV} y_i + \sum_{i=1}^l \sum_{j=1}^l \lambda_i \lambda_j y_i y_j \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle \right) \quad (2.22)$$

O problema de maximização em 2.19 e 2.20 é formulado pela equação de Lagrange, na qual λ_i são os multiplicadores de Lagrange e $y_i = \{-1, +1\}$ define a classe da amostra \mathbf{x}_i . ζ age como um limite superior de valores λ_i . Os vetores de suporte (SV) são as amostras mais próximas do hiperplano e são usados para defini-lo através dos parâmetros \mathbf{w} e b . O algoritmo SVM é muito sensível à escolha adequada dos valores dos parâmetros (CRISTIANINI; SHAW-TAYLOR, 2000).

$$SV = \{(x_i, y_i) | \lambda_i \neq 0; i = 1, \dots, l\} \quad (2.23)$$

2.4.1 Múltiplos *Kernels*

Em muitos casos as amostras não são linearmente separáveis, isto é, não podem ser separadas por um hiperplano. Nesses casos, uma função $\phi(\mathbf{x}) : R^m \rightarrow R^k$ é usada para mapear o espaço de atributos em um novo espaço vetorial R^k . A função ϕ deve ser escolhida de modo que as amostras transformadas para R^k possam ser separadas por um hiperplano.

Uma função *Kernel* é definida como $K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j)$, ou seja, um função que produza o mesmo resultado que o produto escalar dos vetores das amostras em R^k . Assim, o SVM permite separar vetores de suporte com separação não lineares usando um hiperplano linear (YU et al., 2012). Uma função *Kernel* mapeia o espaço de características para um novo espaço vetorial, geralmente acompanhada de um aumento da dimensão vetorial do espaço de atributos ($k \geq m$) (GÖNEN; ALPAYDIN, 2011). As funções *Kernels* mais utilizadas são: linear (Equação 2.24), polinomial (2.25) e gaussiana (2.26). *Kernel* Gaussiano é um tipo de kernel, da classe das funções de base radial *Radial Basis Function* (RBF). O operador $\|\mathbf{x}\|_2^2$ representa o quadrado da norma euclidiana do vetor \mathbf{x} .

$$K_{LIN} = \langle \mathbf{x}_i, \mathbf{x}_j \rangle \quad (2.24)$$

$$K_{POL} = [a\langle \mathbf{x}_i, \mathbf{x}_j \rangle + c]^d; d \in N \quad (2.25)$$

$$K_{RBF} = \exp \left[-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|_2^2 \right]; \gamma > 0 \quad (2.26)$$

A seleção de funções *Kernel* para diferentes aplicações afetam os resultados da classificação (HSU et al., 2003). Geralmente, um único *Kernel* é selecionado através de uma otimização (MULLER et al., 2001). Utilizar várias funções *kernels* de forma cooperativa é uma abordagem alternativa para aproveitar as características de discriminação de cada função *Kernel* em uma combinação entre elas (GÖNEN; ALPAYDIN, 2011).

Compor uma função *Kernel* a partir da soma não ponderada de vários *kernels* não é uma forma otimizada. Se um dos *kernels* não for correlacionado às classes, um peso positivo pode aumentar o ruído sobre os dados (ZIEN; ONG, 2007). A combinação dos *kernels* pode ser feita de diversas formas, como soma ponderada dos *kernels*

obtidos por otimização:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \sum_{v=1}^p c_v \cdot K_v(\mathbf{x}_i, \mathbf{x}_j); c_v \geq 0 \quad (2.27)$$

Neste caso, o valor dos coeficientes c_v deve ser definido de forma a aumentar a contribuição dos *kernels* que possibilite separação das classes, e reduzir os demais. Eles podem ser definidos por otimização (HUANG et al., 2013). Outro método é definir os valores dos coeficientes a partir de funções das amostras de treinamento (LEWIS et al., 2006). Gönen e Alpaydm (2011) faz uma revisão de diversas outras maneiras de combinar *kernels*, por exemplo, pela multiplicação de diferentes funções *kernel*:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \prod_{v=1}^p K_v(\mathbf{x}_i, \mathbf{x}_j) \quad (2.28)$$

3 Método de Classificação HSMI

Ensembles e máquinas de comitê tratam de agrupamentos de classificadores para produzir como saída a indicação da classe de uma amostra de entrada. *Ensembles* são composições de classificadores que individualmente realizam a classificação da entrada com todas as classes do problema. Assim, individualmente, cada classificador componente do *ensemble* é autônomo na tarefa de classificação. O resultado da combinação dos resultados dos componentes em uma classificação final é geralmente superior em relação a cada componente isolado (LIMA, 2004). A máquina de comitê difere do *ensemble* por utilizar um conjunto de classificadores que isoladamente não realizam a tarefa de classificação. A saída de cada classificador contribui para um combinador produzir o resultado da classificação.

O método *Hierarchical Support vector machine with Multiple kernels optimized by Invasive weed optimization* (HSMI) se enquadra como uma mistura de especialistas, dentro do esquema de máquinas de comitê. Cada nó da árvore HSMI é especializada em particionar um subconjunto de classes, e não pode isoladamente fazer uma classificação completa para todas as classes do problema. A saída de cada nó é uma classificação final atribuída à amostra de entrada, ou é direcionada para a entrada de outro nó especialista.

Neste trabalho de doutorado, será adotada a seguinte taxonomia:

- Classificadores competitivos: analisam todo espaço de características/atributos em uma única vez; todos os classificadores implementados classificam todas as classes; possui estrutura paralela;
- Classificadores cooperativos: a partir da definição de um espaço de características e regiões de competência pré-definidos gera-se um comitê de classificadores; possui estrutura serial;
- Classificadores em cascata: verifica como dividir o espaço de atributos; funciona de maneira semelhante ao cooperativo, mas em tempos diferentes; espera ter um passo à frente para tomada de decisão; possui estrutura hierárquica.

Seguindo esta padronização, no método HSMI, em cada nó da árvore a classificação é competitiva. No conceito hierárquico, como resultado final, a árvore é cooperativa. Portanto, tem-se um modelo em cascata cooperativo com análise competitiva em cada nó.

O método proposto trata da construção de uma árvore de classificadores especializados. É utilizado o método SVM descrito na seção 2.4, o algoritmo de otimização IWO apresentado na seção 2.2.6.1, múltiplos *kernels* conforme a seção 2.4.1 e combinação de classificadores segundo características detalhadas na seção 2. O nome HSMI foi escolhido a partir das técnicas componentes, do inglês *Hierarchical SVM-MKL-IWO*. Primeiramente será explorado o modelo de um nó, composto por um classificador, e seu método de treinamento. Em seguida será apresentada a hierarquia proposta e a sequência de treinamento dos nós. Na sequência é descrita a arquitetura construída para execução paralela em *cluster*, aproveitando as características dos vários estágios do treinamento. Por último, será apresentado o algoritmo de teste sobre o método HSMI completo.

3.1 Descrição do nó e construção da árvore

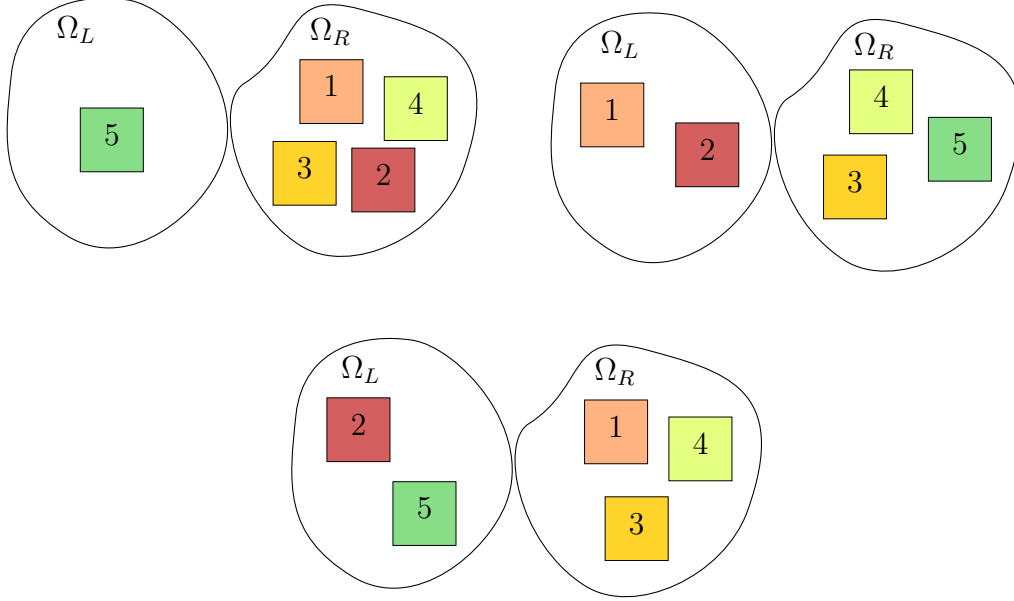
Toma-se como hipótese que um classificador especializado na separação de apenas duas classes produz melhor resultado que outro classificador cuja tarefa é classificar mais de duas classes a partir do mesmo treinamento. Utiliza-se da estratégia dividir-para-conquistar, de forma que cada classificador tenha responsabilidade em distinguir entre duas classes e assim especializar-se nessa tarefa.

Parte-se de um conjunto de amostras rotuladas $(\mathbf{x}_i, y_i) \in X$. As amostras são compostas por um vetor $\mathbf{x}_i = (x_{i1}, x_{i2}, \dots, x_{ij})$, onde cada x_{ij} é chamado de atributo, característica ou ainda canal para o caso de imagens, e um rótulo $y_i \in \Omega = \{1, 2, \dots, C\}$ que indica a classe a qual a amostra pertence. C é o número de classes do problema.

Um nó é composto por um classificador especializado para separar as amostras X em duas partes, identificadas por partição Ω_L e partição Ω_R . A Figura 3.1 mostra três exemplos dentre as possíveis maneiras de particionar as classes 1, 2, 3, 4, 5. As classes de um nó são combinadas de todas as maneiras, sem repetição, ou seja, $\Omega_L = C_m^C$, $m = 1, 2, \dots, C/2$. Cada combinação dá origem a uma partição. As demais classes que não fazem parte de uma combinação são tomadas em outra partição complementar. Cada par de partições, que contemplam assim todas as C classes do nó, é definido como uma combinação θ . Portanto, uma partição é composta pela união de um sub-conjunto de classes $\Omega_L = \{\omega_i\} \subset \Omega$. A outra partição é composta pelas demais classes $\Omega_R = \Omega - \Omega_L$.

No texto será adotado o termo partição para identificar $\theta = (\Omega_L, \Omega_R)$. O conjunto de amostras X é dividido em dois novos conjuntos, de treinamento A e de validação B . Para cada partição θ são formados quatro novos conjuntos de amostras:

Figura 3.1 - Particionamento das classes.



$$\begin{aligned}
 A_L &= \{(\mathbf{x}_i, y_i) \in A \mid y_i \in \Omega_L\} \\
 A_R &= \{(\mathbf{x}_i, y_i) \in A \mid y_i \in \Omega_R\} \\
 B_L &= \{(\mathbf{x}_i, y_i) \in B \mid y_i \in \Omega_L\} \\
 B_R &= \{(\mathbf{x}_i, y_i) \in B \mid y_i \in \Omega_R\}
 \end{aligned} \tag{3.1}$$

Os conjuntos A_L e A_R são usados para treinar o classificador do nó, enquanto B_L e B_R são usados para validar o treinamento. Define-se $\Pi_\theta = [A_L, A_R, B_L, B_R]$ os dados das amostras para cada partição θ . Esses conjuntos são definidos de forma que A_L e A_R possuam o mesmo número de amostras, assim $\text{count}(A_L) = \text{count}(A_R)$, em que $\text{count}()$ retorna o número de amostras do argumento. Além disso, cada classe dentro da sua partição deve ter o mesmo número de amostras que as demais classes na mesma partição, ou seja, $\text{count}(\{(\mathbf{x}, y) \in A_L \mid y = \omega_i \in \Omega_L\}) = \text{count}(\{(\mathbf{x}, y) \in A_L \mid y = \omega_j \in \Omega_L\}), i \neq j$. O mesmo procedimento é adotado para formação de B_L e B_R em relação às amostras de B .

É utilizado para o nó um classificador SVM, que define um hiperplano de separação pela função

$$f(\mathbf{x}) = \langle \mathbf{w}, \phi(\mathbf{x}) \rangle + b \tag{3.2}$$

Conforme discutido na seção 2.4, o processo de treinamento deve encontrar os parâmetros \mathbf{w} e b de f que maximize a margem de separação das classes. Nesse processo, foi utilizada uma função *kernel* construída pela combinação linear da função *kernel* linear, da função *kernel* polinomial e da função *kernel* de base radial.

$$K(\mathbf{x}_i, \mathbf{x}_j) = c_1 \langle \mathbf{x}_i, \mathbf{x}_j \rangle + c_2 [a \langle \mathbf{x}_i, \mathbf{x}_j \rangle + c]^d + c_3 \exp \left[-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|_2^2 \right] \quad (3.3)$$

Deseja-se encontrar uma solução $Q_k = (c_1, c_2, c_3, a, c, d, \gamma)$, com os valores dos parâmetros das funções *kernel* e dos coeficientes lineares, que sejam usados para treinar o SVM com as amostras de treinamento A e que produzam a máxima acurácia sobre o conjunto de validação B . Para isso, é utilizado o algoritmo de otimização IWO. Cria-se uma população de soluções $Q = \{Q_k\}, k = 1, 2, 3 \dots q$ com valores aleatórios para cada parâmetro de Q_k , onde q é a quantidade de população inicial pré-definida para o algoritmo. Cada partição $\theta = (\Omega_L, \Omega_R)$ é combinada com todas as soluções Q_k , formando a população inicial de soluções P_k para o nó

$$P = Q \times \{\theta\} = \{P_w = (Q_k, \theta)\} \quad (3.4)$$

O SVM é então treinado para cada P_w . Isso envolve selecionar o conjunto $\bar{A} = A_L \cup A_R$ da partição θ definida em P_w e usá-lo para treinar o SVM com os parâmetros Q_k de P_w . O SVM treinado SVM_w passa para a fase de validação e cálculo da função objetivo. As amostras \mathbf{x}_i de $\bar{B} = B_L \cup B_R$ correspondentes a mesma partição θ usada no treinamento são processadas pelo SVM para prever suas classes \bar{y}_i . Calcula-se a acurácia A_{cc} da classificação comparando-se \bar{y}_i com y_i . Não se faz necessário o cálculo de κ , pois todas as classificações são binárias. O valor da acurácia é usado como função objetivo para o algoritmo IWO. Após se obter as acurácias de todos P_w , eliminam-se aqueles com menor valor de A_{cc} até se obter a quantidade limite de população pré-definida para o IWO.

Para cada iteração do algoritmo IWO, faz-se a reprodução das soluções P_w segundo os parâmetros de número mínimo e máximo de sementes pré-definidos no IWO, e segundo a acurácia máxima e mínima da população existente, Equação 2.11. As novas soluções P_w^* são aplicadas ao SVM, que é então treinado e validado segundo o método já detalhado, obtendo-se as respectivas acurácias. O processo iterativo de eliminação, reprodução, treinamento e validação é repetido até o número de iterações pré-definido para o IWO.

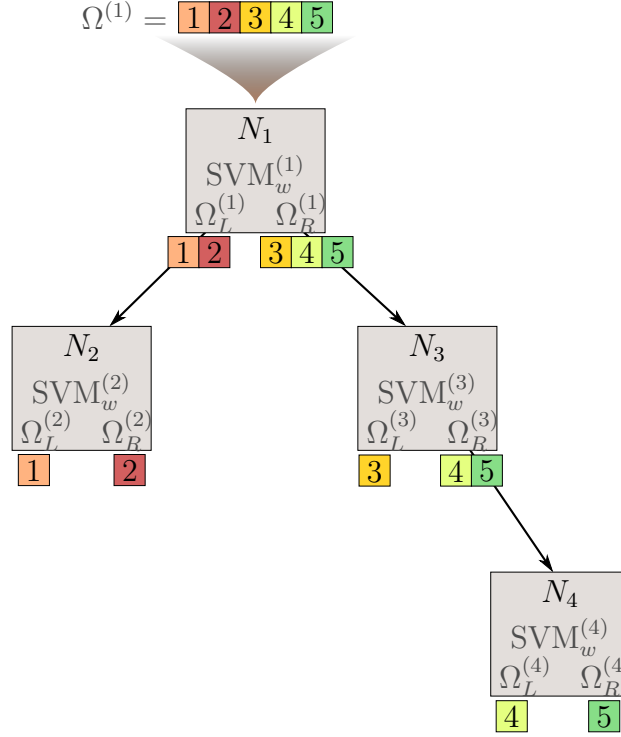
Ao final, a solução $P_w = ((c_1, c_2, c_3, a, c, d, \gamma), (\Omega_L, \Omega_R))$ com melhor acurácia é selecionada para compôr o nó com o respectivo classificador SVM_w . Nesse processo, destacam-se duas importantes contribuições deste trabalho. Ao mesmo tempo em que os parâmetros do *kernel* são otimizados, a função objetivo da otimização é calculada sobre um conjunto de amostras diferente do conjunto de treinamento. Isso promove um equilíbrio entre o erro empírico minimizado por uma solução $P_w^{(t)}$ no treinamento e o erro funcional avaliado sobre o conjunto de validação \bar{B} . Assim, $P_w^{(t)}$ não necessariamente é a solução escolhida para o SVM. A segunda contribuição é a escolha das partições a serem separadas pelo nó ao mesmo tempo em que o classificador é especializado para cada particionamento. Dessa forma, ao escolher uma partição $\theta = (\Omega_L, \Omega_R)$, o método usa durante a comparação entre os possíveis particionamentos classificadores especializados em cada partição. Este método híbrido de seleção de espaço de amostras é semelhante aos chamados métodos embutido de seleção de atributos. Pode-se então nomear essa estratégia de **Método Embutido de Seleção do Espaço de Amostras**.

Obtém-se portanto para um nó dois subconjuntos de classes, Ω_L e Ω_R , junto com o classificador SVM_w especializado em separá-las, que produz a melhor acurácia na classificação binária entre Ω_L e Ω_R . O processo descrito é classificado como competitivo, pois cada solução - erva daninha - P_w compete com as demais, e dentre elas é escolhida uma para uso no classificador.

Descrito o método de treinamento de um nó, segue o processo de construção da árvore HSML. A árvore é composta por nós treinados segundo o método descrito. Dado o conjunto X de amostras (\mathbf{x}_i, y_i) e as classes $\omega_i \in \Omega$ para o nó raiz N_1 na Figura 3.2, este é treinado e recebe a tarefa de separar as partições $\Omega_L^{(1)}$ e $\Omega_R^{(1)}$. No exemplo $\Omega = \{1, 2, 3, 4, 5\}$, $\Omega_L^{(1)} = \{1, 2\}$ e $\Omega_R^{(1)} = \{3, 4, 5\}$. As classes definidas para $\Omega_L^{(1)}$ são atribuídas ao nó filho esquerdo, identificado na Figura 3.2 por N_2 , enquanto as classes definidas para $\Omega_R^{(1)}$ são atribuídas ao nó filho direito N_3 .

O nó filho usa as classes recebidas do nó pai como entrada para seu treinamento. Assim, um nó N_2 do exemplo da Figura 3.2 deverá trabalhar com as classes herdadas de N_1 , ou seja, $\Omega^{(2)} = \Omega_L^{(1)}$. Cada nó é construído e treinado até que todos os nós extremos possuam apenas uma classe para Ω_L e uma classe para Ω_R . Segundo a classificação apresentada no início deste capítulo, ao passo que cada nó é treinado por uma estratégia competitiva, a árvore é composta de forma cooperativa, pois cada nó é responsável por uma parte da classificação, e a estrutura toda é necessária para produzir o resultado final da classificação.

Figura 3.2 - Construção da árvore HSMI.



3.2 Arquitetura computacional e implementação

Os algoritmos e estruturas de dados descritos neste trabalho são apresentados em pseudo código junto com instruções em linguagem C++. Isso torna a escrita mais enxuta sem perder a clareza, considerando que o leitor tenha conhecimento na linguagem C++. O código utiliza as bibliotecas LIBTIFF para ler arquivos TIFF. A biblioteca OpenCV para tratar os dados de amostras e realizar a classificação SVM. O tipo de dados Mat, presente nos algoritmos a seguir, é uma implementação do OpenCV, que gerencia uma matriz de dados, geralmente utilizada para armazenar imagens e amostras. O classificador proposto é identificado por HSMI, enquanto cada nó compreende parte do algoritmo, identificado por SMI (SVM-MKL-IWO).

Para melhorar o desempenho computacional, e aproveitar o potencial de paralelização do método HSMI, foi criada uma arquitetura de implementação paralela, *cluster*. Por não fazer parte do escopo deste trabalho, a arquitetura em *cluster* não foi otimizada, o que segue como sugestão de trabalho futuro. O *cluster* compreende de um nó MESTRE e um conjunto de nós de TRABALHO, assim identificados na Figura 3.3. O nó MESTRE faz a inicialização das amostras em A e B , envia elas para todos os nós de TRABALHO assim como as partições $\Pi = (A_L, A_R, B_L, B_R)$. Em seguida

inicia o processo de treinamento construindo a árvore HSMI. Um terceiro conjunto de dados, conjunto de teste Γ , é usado para se obter o κ para o classificador HSMI completo. O algoritmo do nó MESTRE é apresentado no Algoritmo 1.

Algorithm 1 Execução do processo MASTER.

- 1: $X \leftarrow$ lê arquivo de amostras rotuladas
 - 2: Separa amostras em treinamento A , avaliação B e teste Γ
 - 3: Envia (A, B) para nós de TRABALHO
 - 4: Inicia treinamento (Algoritmo 3)
 - 5: Inicia teste (Algoritmo 4) com amostras Γ
 - 6: Calcula κ com os resultados de teste
 - 7: Executa outros classificadores com A , B e Γ
 - 8: Gera arquivo com resultados
-

A Figura 3.3 mostra a responsabilidade dos nós MESTRE e de TRABALHO na realização do treinamento da árvore HSMI. O nó MESTRE cria o nó raiz, inicializa o algoritmo IWO para o nó raiz e distribui a população de soluções P_w entre os nós de TRABALHO. Depois, recebe dos nós de TRABALHO as novas soluções P_w^* com as acurácias de validação. A população excedente é eliminada e uma nova iteração do IWO é executada.

O processo de treinamento em *cluster* é detalhado na Figura 3.4, que compreende o treinamento SMI de um nó da árvore HSMI. A população inicial de soluções $P = \{P_w\}$ é inicializada a partir dos parâmetros aleatórios em Q_k e as combinações de partições θ criadas a partir da entrada Ω aplicada ao nó. P é então dividida em conjuntos P_p . Cada P_p é enviado para um processo de TRABALHO. Os processos de TRABALHO fazem a reprodução das soluções, treinamento e avaliação do SVM_w , e cálculo da respectiva acurácia de validação (A_{ccw}) para os conjuntos de parâmetros e combinação de partições P_w . Para otimizar o processo, apenas os novos conjuntos P_w , identificados por P_w^* , são processados, pois os demais já possuem sua respectiva acurácia calculada em iterações anteriores.

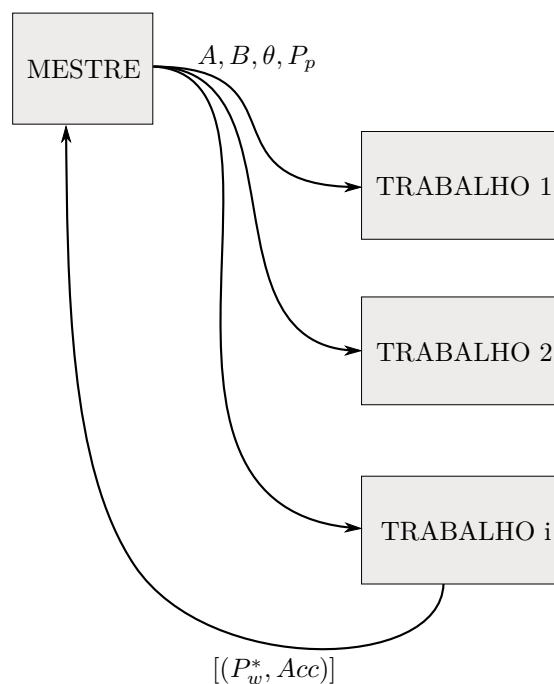
Algumas estruturas de dados usadas são apresentadas no Algoritmo 2. O nó MESTRE recebe todas as soluções P_w , insere-as na população, elimina da população as soluções com menor acurácia até atingir o limite populacional, e depois repete a iteração. O algoritmo usado pelo nó MESTRE no processo de treinamento SMI de um nó está listado em 3 e o algoritmo para execução de testes é listado no Algoritmo 4

Figura 3.3 - Estrutura do Cluster.

Repete para cada nó da árvore

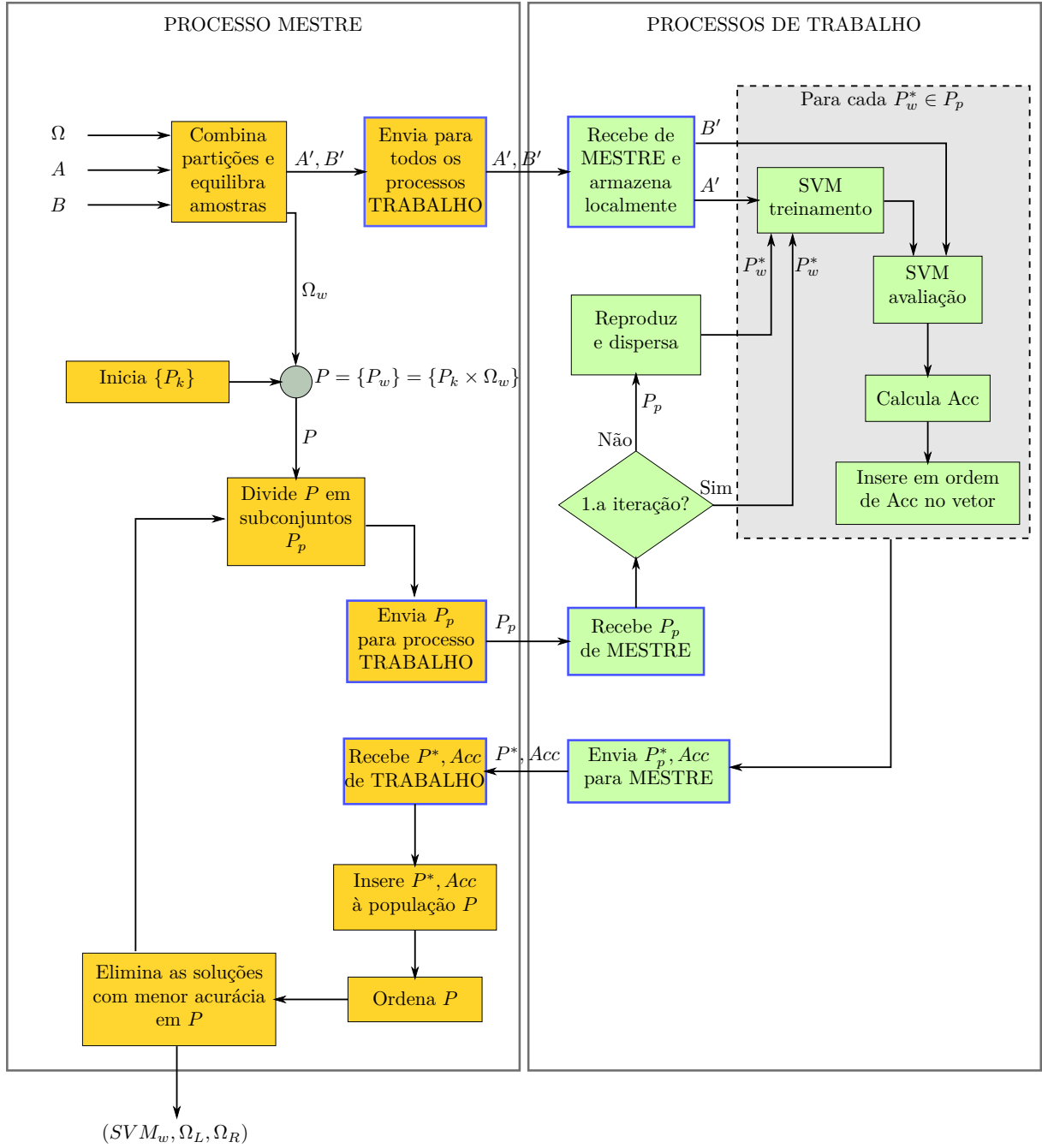
Repete para cada iteração de IWO

- Combinar classes em partições e equilibra amostras
- Cria e distribui população inicial de ervas daninhas
- Envia subconjuntos de ervas daninhas para processos
- Recebe ervas daninhas com acurácia dos processos
- Insere as novas ervas daninhas P_w^* na população
- Ordena a população
- Elimina P_w para ajustar a população máxima



- Recebe conjunto de ervas daninhas de MASTER
- Reproduz e dispersa ervas daninhas
- Processa uma iteração para cada P_w^* :
 - Treina SVM
 - Avalia SVM treinado e calcula acurácia Acc
- Envia resultados de acurácia para MASTER

Figura 3.4 - Algoritmo SMI em cluster.



Algorithm 2 Estruturas de dados usadas nos algoritmos.

```
1: vector<Mat&> A, B                                ▷ amostras separadas por classes
2:
3: class PartitionPair  $\Pi_\theta$ 
4:   methods:
5:     Mat getSamples()      ▷ verifica matCreated para criar  $A_{samples}, A_{labels}...$ 
6:     Mat getLabels()
7:   attributes:
8:     int  $\theta$                                 ▷ índice do objeto na lista de partições
9:     boolean matCreated    ▷ true: Mat  $A_{samples}, A_{labels}...$  já criadas
10:    vector<int>  $\tilde{A}_L, \tilde{A}_R, \tilde{B}_L, \tilde{B}_R$       ▷ índices das amostras
11:    Mat  $A_{samples}, A_{labels}, B_{samples}, B_{labels}$   ▷ dados e rótulos das amostras
12:    vector<int> classL, classR                ▷ IDs das classes das partições
13: end class
14:
15: struct Weed  $P_w$ 
16:   vector<int>  $P_k$                                 ▷ parâmetros da função multikernel
17:   vector<int>  $\Omega_L, \Omega_R$                     ▷ IDs das classes das partições
18:   int  $\theta$                                     ▷ índice da partição
19:   float Acc                                     ▷ acurácia
20: end struct
21:
22: struct Node
23:   SVM* SVM $_w$ 
24:   vector<int> X                                ▷ conjunto de índices das amostras para classificação
25:   vector<int>  $\Omega_L$                             ▷ IDs das classes da partição esquerda
26:   vector<int>  $\Omega_R$                             ▷ IDs das classes da partição direita
27:   Node* ptrL                                  ▷ Referência para nó esquerdo
28:   Node* ptrR                                  ▷ Referência para nó direito
29: end struct
30:
31: struct NodeTrain
32:   vector<int>  $\Omega$                                 ▷ IDs das classes a serem classificadas pelo nó
33:   Node* node                                  ▷ Referência para o nó correspondente na árvore treinada
34: end struct
35:
36: class HSMI
37:   methods:
38:     train()
39:     predict()
40:   attributes:
41:     NodeTrain* rootTrain                      ▷ árvore para treinamento
42:     Node* root                                ▷ árvore treinada
43:     Mat classified                            ▷ resultado da classificação
44: end class
45:
46: class Communication                                ▷ Serializa e envia dados no cluster
47: end class
```

Algorithm 3 Treinamento da árvore HSMI - nó MESTRE.

```
1: rootTrain  $\leftarrow$  new NodeTrain( $\Omega$ )
2: rootTrain.node  $\leftarrow$  new Node()
3: root  $\leftarrow$  rootTrain.node
4: stack.push(rootTrain)
5: repeat
6:   ptr  $\leftarrow$  stack.pop()
7:    $\Omega \leftarrow$  ptr. $\Omega$ 
8:   [ $SVM_w, \Omega_L, \Omega_R$ ]  $\leftarrow$  SMI<CLUSTER>( $\Omega$ ) ▷ Figura 3.4
9:   ptr.node. $SVM_w \leftarrow SVM_w$ 
10:  ptr.node. $\Omega_L \leftarrow \Omega_L$ 
11:  ptr.node. $\Omega_R \leftarrow \Omega_R$ 
12:  if count( $\Omega_L$ ) > 1 then
13:    nodeTrainL  $\leftarrow$  new NodeTrain( $\Omega_L$ )
14:    nodeTrainL.node  $\leftarrow$  new Node()
15:    ptr.node.ptrL  $\leftarrow$  nodeTrainL.node
16:    stack.push(nodeTrainL)
17:  end if
18:  if count( $\Omega_R$ ) > 1 then
19:    nodeTrainR  $\leftarrow$  new NodeTrain( $\Omega_R$ )
20:    nodeTrainR.node  $\leftarrow$  new Node()
21:    ptr.node.ptrR  $\leftarrow$  nodeTrainR.node
22:    stack.push(nodeTrainR)
23:  end if
24:  free(ptr) ▷ libera memória de NodeTrain sem desalocar o Node associado
25: until stack not empty
```

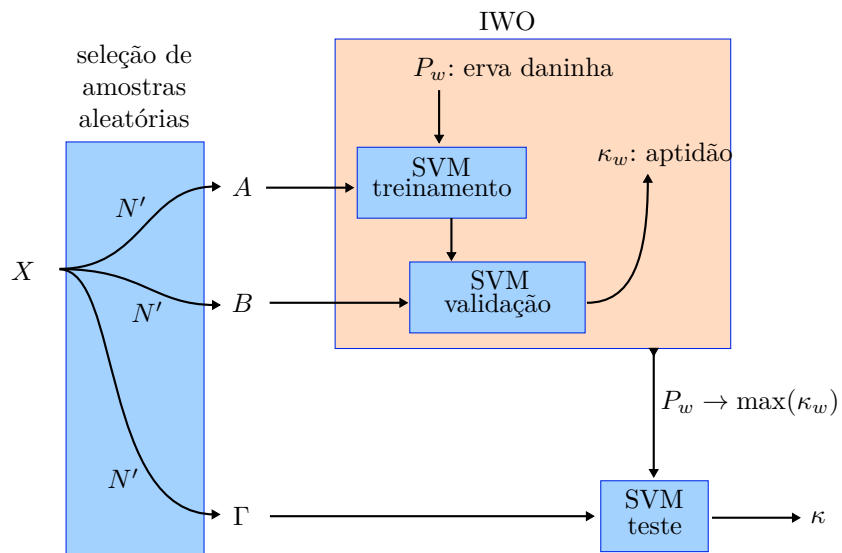
Algorithm 4 Classificação de amostras a partir da árvore HSMI treinada.

```

1: root.X = X                                ▷ conjunto de índices das amostras para classificação
2: stack.push(root)
3: repeat
4:   node ← stack.pop()
5:   D ← data(node.X)                          ▷ busca as amostras a partir dos índices em X
6:   R = node.SVMw.predict(D)
7:   ptr = &R[0]
8:   ptrX = &X[0]
9:   for  $i = 0$  to count(R) - 1 do
10:    if *ptr < 0 then
11:      XL.add(*ptrX)
12:    else
13:      XR.add(*ptrX)
14:    end if
15:    ptr++
16:    ptrX++
17:  end for
18:  if node.ptrL not null then
19:    node.ptrL.X ← XL
20:    stack.push(node.ptrL)
21:  else
22:    for each  $x \in X_L$  do
23:      classified[x] ←  $\Omega_L$ 
24:    end for
25:  end if
26:  if node.ptrR not null then
27:    node.ptrR.X ← XR
28:    stack.push(node.ptrR)
29:  else
30:    for each  $x \in X_R$  do
31:      classified[x] ←  $\Omega_R$ 
32:    end for
33:  end if
34: until stack not empty

```

Figura 3.5 - Metodologia: *Hierarchical Support vector machine with Multiple kernels and Invasive weed Optimization* (HSMI).



4 Experimentos e Resultados

Os experimentos realizados neste trabalho foram executados em *cluster* com 18 máquinas em processamento paralelo. Cada máquina possui as seguintes configurações: um processador 3.2GHz, AMD Phenom(tm) II X4 (4 núcleos) B97, com 4 GB de RAM e sistema operacional Windows 7 de 64 bits.

O método HSMI foi comparado aos métodos SVM com *kernel* RBF e SVM com *kernel* polinomial, já estabelecidos como padrões na literatura. Foram usadas duas configurações para o SVM com *kernel* RBF, uma com $\gamma = 0.167$ e outra com $\gamma = 0.004$. O primeiro parâmetro foi baseado na configuração padrão do software ENVI. Este software é amplamente utilizado nas aplicações de Sensoriamento Remoto. O SVM-RBF com $\gamma = 0.004$ foi escolhido a partir de uma otimização em um conjunto de testes realizados. A terceira comparação foi feita com o SVM polinomial, utilizando o parâmetro $d = 2.0$. Este valor foi atribuído por ser utilizado como padrão na classificação do ENVI. Deve-se observar que a função *kernel* adotada como padrão do ENVI é a RBF. A matriz de confusão apresentada para comparação em cada experimento foi a do classificador SVM-RBF com $\gamma = 0.004$, que foi o melhor resultado dentre os métodos de comparação.

Quando são criados polígonos sobre a imagem para definir regiões de interesse (ROIs), ocorre que os pixels da região, por serem vizinhos, possuem alta correlação. Se um destes conjuntos for usado para treinamento e outro conjunto para validação, pode acontecer que o conjunto de treinamento tenha baixa correlação com o conjunto de validação. Isso levará a uma redução na acurácia calculada sobre o conjunto de validação para um classificador treinado no primeiro conjunto.

Para a realização dos experimentos desta tese, cada dado de um estudo de caso foi classificado 10 vezes, com variação do sorteio das amostras de treinamento, validação e testes. A quantidade de amostras escolhidas para treinamento, validação e teste de cada classe foi selecionada a partir da classe com o menor número de amostras (S_{min}). Cada conjunto (treinamento, validação e teste) de uma classe recebe aleatoriamente um terço ($1/3$) de (S_{min}). Esta escolha aleatória tem por intuito gerar combinação de todos os tipos de amostras, evitando a correlação dentro do conjunto.

No caso do *kernel* polinomial da função multikernel optou-se por não restringir os valores dos coeficientes a e c como positivos, de forma a se encontrar funções discriminantes as melhores possíveis dentro do processo de otimização, mesmo que, no caso de coeficientes negativos, algumas propriedades típicas de *kernel* não possam

ser asseguradas.

4.1 Estudo de Caso 1 - Imagem Sintética

Uma imagem sintética refere-se a um conjunto de dados controlados, que permite a verificação e avaliação de diversas características (PANTALEÃO, 2012).

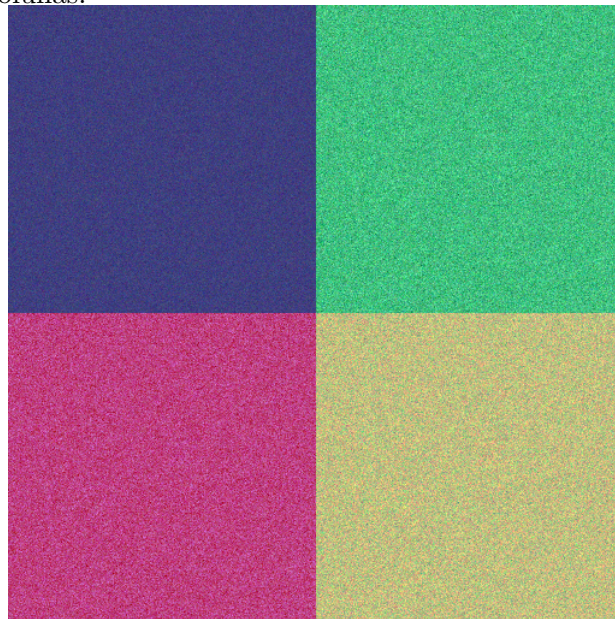
4.1.1 Dados

Para este estudo de caso foi utilizada uma imagem com quatro classes com a mesma quantidade de *pixels*, conforme apresentada na Figura 4.1.

A Figura 4.2 mostra cada um dos três canais da imagem sintética. Os canais um (1) e dois (2) apresentam duas regiões bem caracterizadas, enquanto o canal três (3) não apresenta boa capacidade de diferenciação dos *pixels* (PANTALEÃO, 2012). A Figura 4.3 mostra as regiões utilizadas como amostras no processo de classificação.

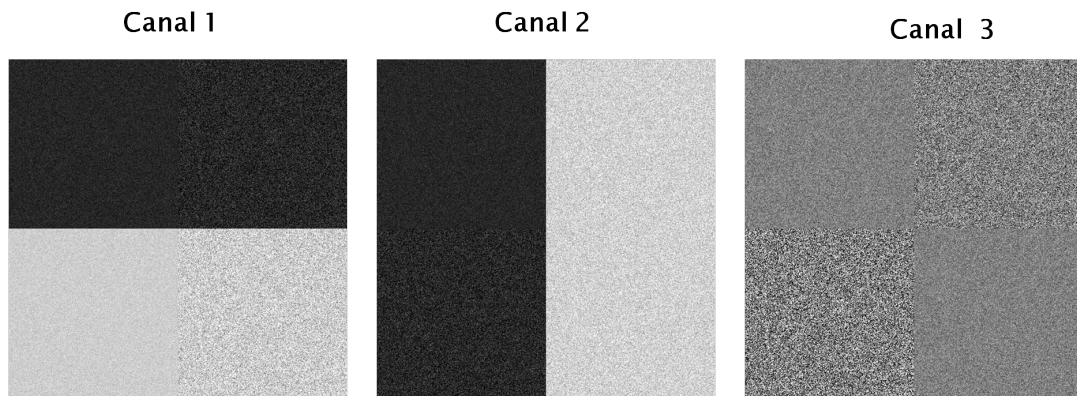
O número de *pixels* utilizado das amostras é apresentado na Figura 4.4. Neste estudo de caso, a classe 1 (*Green*) possui a menor quantidade de amostras ($S_{min} = 1739$). Foram sorteados um terço ($1/3$) de 1739 *pixels* para cada conjunto, de treinamento, validação e testes, para cada uma das quatro classes.

Figura 4.1 - Imagem Sintética. Composição colorida dos três canais: R, G e B. 512 linhas e 512 colunas.



Fonte: Pantaleão (2012).

Figura 4.2 - Imagem Sintética. Três Canais: (1)R, (2)G e (3)B.



Fonte: Pantaleão (2012)

Figura 4.3 - Imagem Sintética com a visualização das amostras das classes utilizadas.

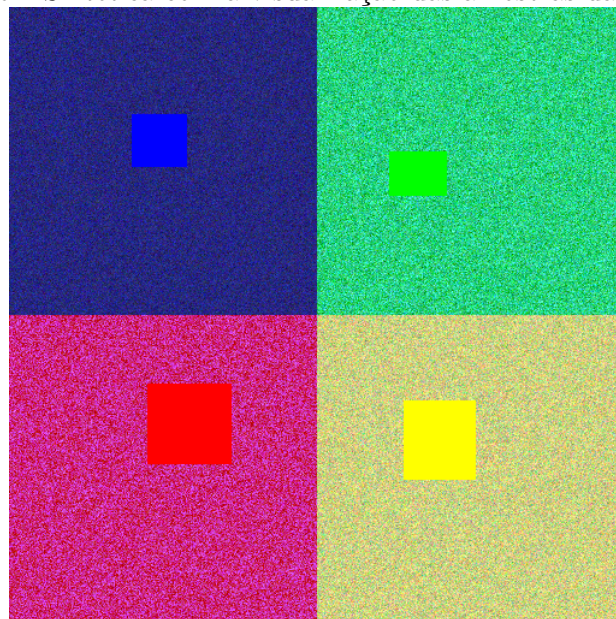






Figura 4.4 - Número de *pixels* das amostras que foram utilizadas para treinamento, validação e teste. Conjunto de 4 classes.

		Classe	Número de <i>pixels</i>
	0	Azul	1980
	1	Verde	1739
	2	Vermelho	4623
	3	Amarelo	3894

4.1.2 Resultados

O experimento usando SVM com *kernel* de base radial (RBF) configurado com $\gamma = 0.004$ apresentou estatística kappa $\kappa = 0,996546$. Sua matriz de confusão é exibida na Tabela 4.1. As matrizes de confusão apresentadas possuem os conjuntos de referência das classes organizados em coluna enquanto as linhas são as classificações realizadas.

Tabela 4.1 - Imagem sintética - SVM-RBF $\gamma = 0,004$

	0	1	2	3
0	100,00%	0,00%	0,00%	0,00%
1	0,00%	99,76%	0,00%	0,79%
2	0,00%	0,00%	100,00%	0,00%
3	0,00%	0,24%	0,00%	99,21%

O experimento usando SVM com *kernel* polinomial configurado com $d = 2$ apresentou $\kappa = 0,996$. A matriz de confusão pode ser visualizada na Tabela 4.2.

Tabela 4.2 - Imagem sintética - SVM-POLY $d = 2$

	0	1	2	3
0	100,0%	0,0%	0,0%	0,0%
1	0,0%	100,0%	0,0%	1,4%
2	0,0%	0,0%	99,8%	0,0%
3	0,0%	0,0%	0,2%	98,6%

O experimento realizado com o método HSMI apresentou $\kappa = 0,996$. A matriz de confusão é apresentada na Tabela 4.3.

Tabela 4.3 - Imagem sintética - HSMI

	0	1	2	3
0	100,0%	0,0%	0,0%	0,0%
1	0,0%	99,7%	0,0%	0,9%
2	0,0%	0,0%	100,0%	0,0%
3	0,0%	0,3%	0,0%	99,1%

Os experimentos produziram duas configurações de árvores, apresentadas na Figura

4.5. Apesar de as duas árvores serem diferentes, as classes com maior confusão foram separadas no último nível em ambos os casos. As soluções encontradas para os parâmetros dos *kernels* da primeira e da segunda árvore estão listadas respectivamente nas tabelas 4.4 e 4.5. Um exemplo da imagem classificada pelo método HSMI é apresentado na Figura 4.6.

Figura 4.5 - Árvores treinadas para imagem sintética.

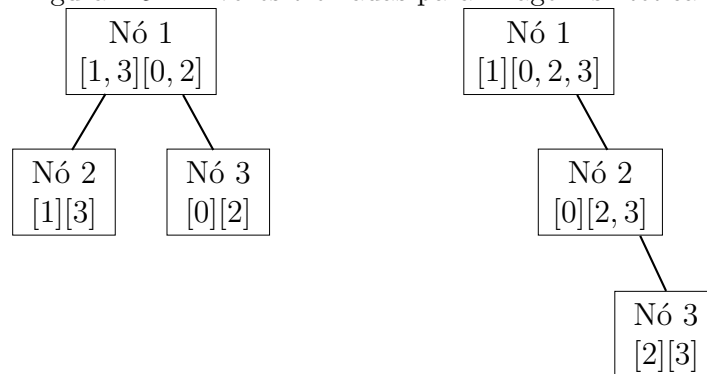


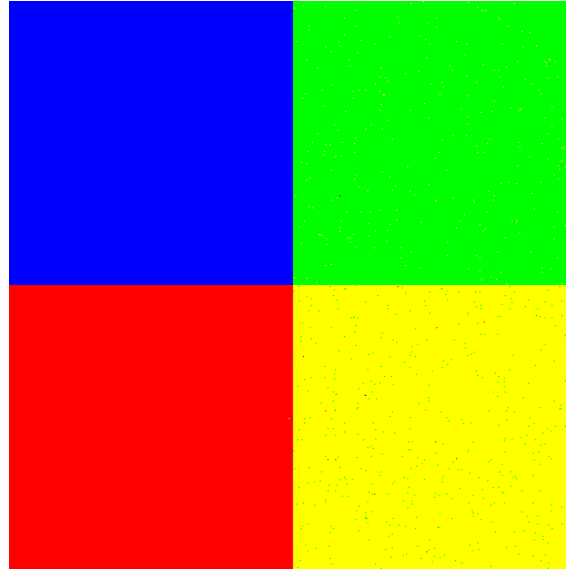
Tabela 4.4 - Imagem sintética - HSMI - parâmetros da solução para a primeira árvore.

Nó	A_{cc}	Coeficientes Lineares			POLY			RBF
		LIN	POLY	RBF	a	c	d	γ
1	1,00	0,26	2,57	3,19	4,84	3,01	2,00	2,82
2	1,00	3,88	13,16	15,83	8,46	9,78	1,00	7,37
3	1,00	1,57	0,88	1,62	3,16	4,98	2,00	0,19

Tabela 4.5 - Imagem sintética - HSMI - parâmetros da solução para a segunda árvore.

Nó	A_{cc}	Coeficientes Lineares			POLY			RBF
		LIN	POLY	RBF	a	c	d	γ
1	1,00	1,92	4,84	2,33	2,46	0,10	1,00	4,44
2	1,00	3,45	2,76	1,35	0,55	3,08	2,00	4,02
3	1,00	2,06	1,14	0,29	4,52	4,33	2,00	2,95

Figura 4.6 - Imagem sintética classificada pelo método HSML.



4.2 Estudo de Caso 2 - Imagem Óptica

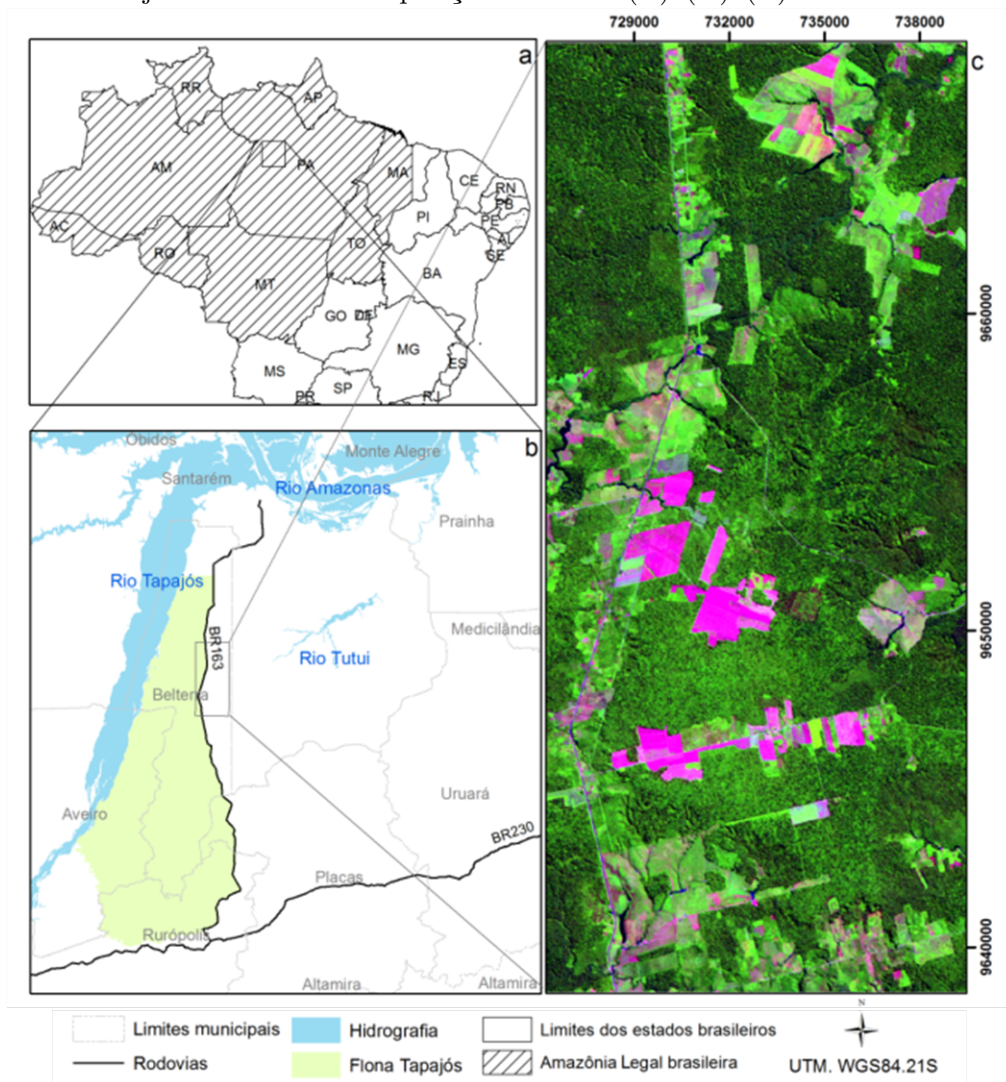
Sensoriamento Remoto é definido por (COLWELL, 1997) como “a medida ou aquisição de informações de algumas propriedades de algum objeto ou fenômeno, por um aparelho de registro remoto que não está em contato físico com o objeto ou fenômeno sob investigação”. A ideia básica é mensurar as variações temporais, espectrais e espaciais de objetos terrestres através de diferentes tipos de sensores que registram a radiação eletromagnética refletida ou emitida por esses objetos. Assim, as imagens capturadas por Sensoriamento Remoto têm como objetivo extrair informações das características ambientais. O monitoramento ambiental está se expandindo a cada ano diante da disponibilidade de imagens de satélites para pesquisa e dos avanços da aplicação de sensores ópticos e micro-ondas.

Neste trabalho de doutorado foi realizado um estudo de sensoriamento remoto em uma área da região amazônica localizada a leste da FLoresta NAcional de Tapajós (FLONA) (Figura 4.7). Esta área é uma unidade de conservação federal criada pelo Decreto N.º 73.684 - de 19 de fevereiro de 1974, que abrange uma área aproximada de 600 mil hectares, localizados em terras de quatro municípios: Aveiro, Belterra, Rurópolis e Placas, no oeste do estado brasileiro do Pará (SECRETARIA DE ESTADO DE MEIO AMBIENTE, 1974).

Esta área é muito explorada por estudiosos devido à preocupação com as regiões de desmatamento florestais, visto que esta região ocupa as florestas tropicais as quais

abrigam grande parte da diversidade natural do planeta, além da grande participação dos ciclos hidrológicos e climáticos. Além disto, apresenta diversos trabalhos realizados pelo INPE, bem como disponibilidade de imagens de sensoriamento remoto e informações de trabalhos realizados em campo sob diversas classes de ocupação do solo, tais como: floresta primária, florestas em diversas fases de regeneração, agricultura e pastagem.

Figura 4.7 - Localização da área de estudo. a) localização em relação à Amazônia Legal brasileira; b) localização em relação aos limites políticos e geográficos; c) recorte aproximado da área de estudo de imagem LANDSAT5/TM de 29 de junho de 2010. Composição colorida 5(R)4(G)3(B).



Fonte: Reis (2014).

Esta Mesorregião do Baixo Amazonas localizada entre as coordenadas 55°1'51"W

3°22'51"S e 54°42'16"W 2°56'53"S é caracterizada por ter o clima quente e úmido, com predominância de floresta tropical, mais precisamente por floresta ombrófila densa. Este tipo de floresta é uma mata de serranias próximas ao Oceano Atlântico, caracterizada por espécies tropicais afro-brasileiras (VACCARO, 2002), com mata de aspecto verde, árvores emergentes de até 40 metros de altura e vegetação arbustiva densa.

Em virtude da grande expansão agrícola mecanizada nas duas últimas décadas, caracterizada principalmente pela produção de soja e outros grãos, como arroz, milho, sorgo e feijão, existem nestas regiões grandes áreas de floresta degradada, principalmente causada por queimadas (VENTURIERI, 2007). Assim, há florestas em diferentes estágios de regeneração, como resultado das modificações antrópicas nas proximidades da rodovia BR-163 (Santarém/Cuiabá).











O calendário agrícola começa em dezembro, no início da estação chuvosa, com uma ou duas atividades de plantio durante o ano, sendo as colheitas realizadas, respectivamente, em março/abril e julho/agosto. As classes descritas na Figura 4.8 foram definidas a partir de trabalhos de campo realizados em datas compatíveis com as aquisições das imagens.

Para a construção de um primeiro cenário (Cenário 1), dez (10) classes são esquematizadas (REIS, 2014):

- 0 Área em Pousio (AP): áreas agrícolas em pousio, cobertas por palha ou vegetação esparsa;
- 1 Área Cultivada (AC): culturas de grãos;
- 2 Floresta Degradada (FD): floresta degradada por atividades de fogo ou por desmatamento seletivo, de forma que suas características originais tenham sido alteradas, mas ainda possuam porte e fisionomia florestal;
- 3 Floresta Primária (FP): floresta em que a ação humana não provocou alterações significativas em suas características originais de estrutura e espécies;
- 4 Pasto Limpo (PL): áreas com vegetação típica de pastagens, com predomínio de herbáceas;
- 5 Pasto Sujo (PS): áreas com vegetação típica de pastagens, com presença de espécies arbustivas e espécies invasoras;

- 6 Solo Exposto (SE): áreas predominantemente de solo exposto;
- 7 Vegetação Secundária Inicial (VS1): áreas de vegetação secundária com predominância de herbáceas e arbustos;
- 8 Vegetação Secundária Intermediária (VS2): áreas com presença de vegetação secundária com poucas espécies herbáceas e predominância de árvores de pequeno porte e arbustos;
- 9 Vegetação Secundária Avançada (VS3): florestas secundárias em avançado estágio de desenvolvimento, com predomínio de árvores geralmente entre 13 e 17 m, mas ocorrência de árvores emergentes e, em menor grau, arbustos e herbáceas.

Figura 4.8 - Número de pixels das amostras que foram utilizadas para treinamento, validação e teste. Para imagem TM (Coluna 30m de resolução) e para imagem de Radar (Coluna 15m de resolução). Conjunto de 10 classes.

Classe			2010	
			30 m	15 m
	AP	Área em Pousio	1650	6457
	AC	Área Cultivada	1112	4286
	FD	Floresta Degradada	6260	25050
	FP	Floresta Primária	3512	13882
	PL	Pasto Limpo	1283	4933
	PS	Pasto Sujo	1836	7242
	SE	Solo Exposto	2336	9236
	VS1	Vegetação Secundária Inicial	499	1927
	VS2	Vegetação Secundária Intermediária	710	2780
	VS3	Vegetação Secundária Avançada	2121	8482







Fonte: Reis (2014).

Para um segundo teste, construção de um segundo cenário, as dez (10) classes foram reduzidas a um conjunto de (seis) classes (Cenário 2). As classes VS3, VS2 e VS1 foram integradas na classe denominada VS. E as classes FD e AP foram eliminadas desta análise. Esta nova configuração é apresentada na Figura 4.9

4.2.1 Dados

No dia 23 de Julho de 1972, a *National Aeronautics and Space Administration* (NASA) lançou nos Estados Unidos o primeiro satélite denominado *Earth Resour-*

Figura 4.9 - Número de pixels das amostras que foram utilizadas para treinamento, validação e teste. Para imagem TM (Coluna 30m de resolução) e para imagem de Radar (Coluna 15m de resolução). Conjunto de 6 classes.

Classe			2010	
			30 m	15 m
	AC	Área Cultivada	1112	4286
	FP	Floresta Primária	3512	13882
	PL	Pasto Limpo	1283	4933
	PS	Pasto Sujo	1836	7242
	SE	Solo Exposto	2336	9236
	VS	Vegetação Secundária	3330	13189

Fonte: Reis (2014).

ces Technology Satellites (ERTS 1), pertencente ao quadro do Programa Espacial “*Earth Resources Technology Satellite*”. O *Land Remote Sensing Satellite-5* (LANDSAT/TM 5), pertencente a esse programa espacial, foi lançado em 1984 com o objetivo de mapeamento multiespectral em alta resolução da superfície da Terra. As principais características das imagens LANDSAT-5 TM (JENSEN, 2009), são:

- Resolução espacial: 30 m
- Resolução radiométrica: 8 bits
- Resolução temporal: 16 dias
- Faixa imageada: 185 km
- Resolução espectral: 7 bandas (Visível (3), Infravermelho próximo (1), Infravermelho de ondas curtas (2), e Infravermelho termal (1))

No Estudo de Caso 2 foi utilizada a imagem multi-espectral LANDSAT-5 TM, bandas 1, 2, 3, 4, 5 e 7, adquirida em 29/06/2010, apresentada na Figura 4.10. Para uma melhor compreensão do tratamento das imagens óticas utilizadas nesta tese, ver Reis (2014).

Neste estudo de caso foram realizados dois processos de classificação, um utilizando a legenda com dez (10) classes (Cenário 1) e outro com seis (6) classes (Cenário 2). O número de *pixels* utilizado das amostras para a imagem TM de dez (10) classes é apresentado na Figura 4.8. Neste estudo de caso, a classe 7 (VS1) possui a menor quantidade de amostras ($S_{min} = 499$). Foram sorteados um terço (1/3) de 499 *pixels*

Figura 4.10 - Imagem LANDSAT5/TM de 29 de junho de 2010, composição colorida 5(R)4(G)3(B). Coordenadas referentes a UTM/WGS84.



Fonte: Reis (2014).

para cada conjunto, de treinamento, validação e testes, para cada uma das dez classes. O número de *pixels* utilizado das amostras para a imagem TM de seis (6) classes é apresentado na Figura 4.9. Neste estudo de caso, a classe 0 (AC) possui a menor quantidade de amostras ($S_{min} = 1112$). Foram sorteados um terço ($1/3$) de 1112 *pixels* para cada conjunto, de treinamento, validação e testes, para cada uma das seis classes.

4.2.2 Resultados

Primeiramente a imagem TM foi classificada em um conjunto de 10 classes. O experimento usando SVM com *kernel* de base radial (RBF), configurado com $\gamma = 0.004$, apresentou $\kappa = 0,738153$. A matriz de confusão é apresentada na Tabela 4.12. No teste do SVM-RBF, com $\gamma = 0,167$, o resultado foi de $\kappa = 0,677644$. Usando o SVM com *kernel* polinomial, obteve-se $\kappa = 0,316867$.

Tabela 4.6 - Imagem TM - 10 classes - SVM-RBF $\gamma = 0,004$

	0	1	2	3	4	5	6	7	8	9
0	90,6%	0,0%	0,0%	0,0%	0,8%	0,0%	6,6%	0,0%	0,0%	0,0%
1	0,0%	97,8%	0,0%	0,0%	3,3%	1,7%	0,0%	0,1%	0,0%	0,0%
2	0,0%	0,0%	35,4%	2,5%	0,0%	0,0%	0,0%	2,2%	23,3%	12,0%
3	0,0%	0,0%	9,3%	82,8%	0,0%	0,0%	0,0%	0,0%	2,2%	12,0%
4	1,7%	0,5%	0,0%	0,0%	89,2%	1,0%	0,0%	0,0%	0,0%	0,0%
5	0,0%	1,0%	0,0%	0,0%	6,7%	89,5%	0,0%	5,4%	1,4%	1,6%
6	7,7%	0,0%	0,0%	0,0%	0,0%	0,0%	93,4%	0,0%	0,0%	0,0%
7	0,0%	0,7%	1,4%	0,0%	0,0%	6,1%	0,0%	87,8%	4,0%	0,1%
8	0,0%	0,0%	23,3%	0,2%	0,0%	1,7%	0,0%	4,5%	42,8%	19,2%
9	0,0%	0,0%	30,6%	14,5%	0,0%	0,0%	0,0%	0,0%	26,4%	55,1%

O experimento usando o método HSMI resultou em $\kappa = 0,729139$. A matriz de confusão é visualizada na Tabela 4.7. A Figura 4.11 mostra a árvore produzida nos experimentos, cujos nós têm como solução os parâmetros listados na Tabela 4.8. A imagem classificada pelo método HSMI é visualizada na Figura 4.12.

Tabela 4.7 - Imagem TM - 10 classes - HSMI

	0	1	2	3	4	5	6	7	8	9
0	92,6%	0,0%	0,0%	0,0%	2,4%	0,0%	10,0%	0,0%	0,0%	0,0%
1	0,8%	98,4%	0,0%	0,0%	2,8%	2,8%	0,0%	1,8%	0,0%	0,0%
2	0,0%	0,0%	38,8%	3,6%	0,0%	0,0%	0,0%	1,6%	23,3%	16,3%
3	0,0%	0,0%	15,7%	90,0%	0,0%	0,0%	0,0%	0,0%	5,4%	24,7%
4	1,0%	0,2%	0,0%	0,2%	88,6%	3,2%	0,0%	0,0%	0,0%	0,0%
5	0,0%	1,0%	0,0%	0,0%	6,0%	89,8%	0,0%	6,4%	1,8%	1,0%
6	5,6%	0,0%	0,0%	0,0%	0,0%	0,0%	90,0%	0,0%	0,0%	0,0%
7	0,0%	0,4%	5,2%	0,0%	0,2%	2,6%	0,0%	87,8%	5,6%	0,4%
8	0,0%	0,0%	26,7%	1,0%	0,0%	1,0%	0,0%	2,4%	45,0%	22,1%
9	0,0%	0,0%	13,7%	5,2%	0,0%	0,6%	0,0%	0,0%	18,9%	35,5%

Figura 4.11 - Árvore treinada para imagem TM com 10 classes.

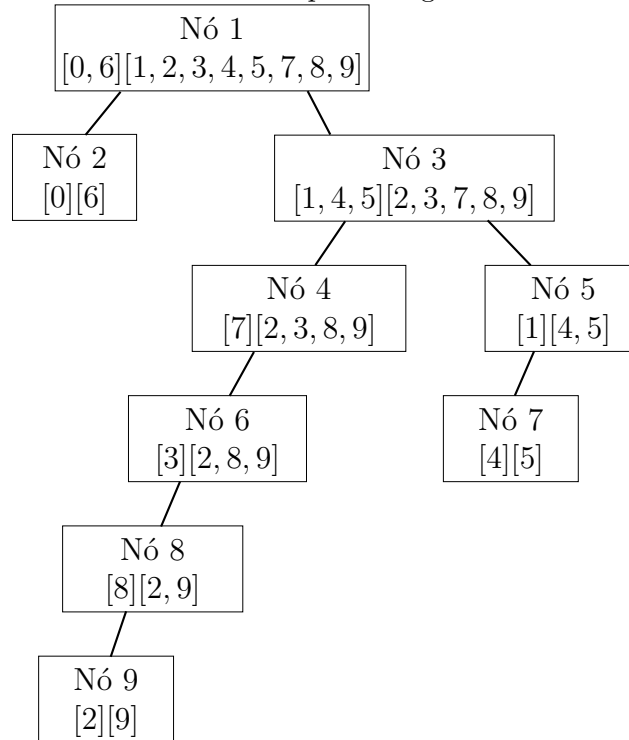
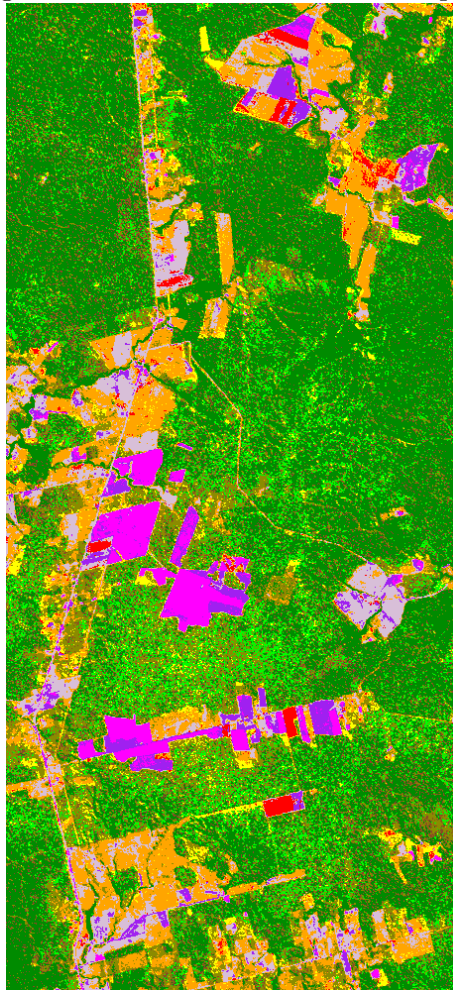


Tabela 4.8 - Imagem TM - 10 classes - HSMI - parâmetros da solução.

Nó	A_{cc}	Coeficientes Lineares			POLY			RBF
		LIN	POLY	RBF	a	c	d	γ
1	1.00	18.09	2.89	5.68	1.88	10.66	1.00	23.46
2	0.94	0.03	0.00	0.21	15.33	12.11	1.00	0.03
3	0.99	0.00	0.00	0.01	6.87	17.26	1.00	0.02
4	0.98	13.30	5.17	4.31	-1.97	-7.61	1.00	0.04
5	0.99	0.00	0.00	27.60	9.98	12.79	2.00	0.07
6	0.92	4.19	1.75	2.20	4.32	2.10	4.00	1.98
7	0.98	4.54	1.15	5.11	-8.38	5.96	4.00	2.90
8	0.74	4.23	1.22	54.46	2.23	-6.26	1.00	4.37
9	0.73	0.68	0.00	21.25	5.48	-10.31	1.00	12.55

Figura 4.12 - Imagem TM - 10 classes - classificada pelo método HSMI.



Um segundo cenário foi proposto unindo as classes com menores contrastes até obter-se 6 classes (Cenário 2). Usando o classificador SVM-RBF, com $\gamma = 0,004$, o resultado foi de $\kappa = 0,908649$. A matriz de confusão é apresentada na Tabela 4.9. No teste do SVM-RBF, com $\gamma = 0,167$, o resultado foi de $\kappa = 0,886487$. Já para o SVM com *kernel* polinomial, obteve-se $\kappa = 0,469369$.

Tabela 4.9 - Imagem TM - 6 classes - SVM-RBF $\gamma = 0,004$

	0	1	2	3	4	5
0	97,6%	0,0%	3,1%	0,8%	0,0%	0,1%
1	0,0%	85,3%	0,0%	0,0%	0,0%	13,0%
2	0,0%	0,0%	91,7%	1,3%	0,0%	0,0%
3	2,4%	0,0%	5,1%	97,0%	0,0%	4,2%
4	0,0%	0,0%	0,0%	0,0%	100,0%	0,0%
5	0,0%	14,7%	0,1%	0,9%	0,0%	82,7%

O experimento para imagem TM com 6 classes usando o método HSMI obteve $\kappa = 0,911216$ e a matriz de confusão é apresentada na Tabela 4.10.

Tabela 4.10 - Imagem TM - 6 classes - HSMI.

	0	1	2	3	4	5
1	98,6%	0,0%	3,4%	2,6%	0,0%	0,4%
2	0,0%	87,4%	0,0%	0,0%	0,0%	13,4%
0	0,4%	0,0%	92,5%	2,0%	0,0%	0,0%
3	0,7%	0,0%	3,9%	93,9%	0,0%	2,9%
4	0,0%	0,0%	0,1%	0,0%	100,0%	0,0%
5	0,3%	12,6%	0,0%	1,6%	0,0%	83,2%

A Figura 4.13 mostra a árvore produzida nos experimentos com 6 classes, cujos nós têm como solução os parâmetros listados na Tabela 4.11. A imagem classificada pelo método HSMI é visualizada na Figura 4.13.

4.3 Estudo de Caso 3 - Imagem Radar

A área de estudo deste terceiro estudo de caso é a mesma descrita na seção 4.2. A imagem utilizada nesse experimento é uma imagem de radar, enquanto a imagem usada na seção 4.2 é uma imagem óptica.

Apesar das imagens obtidas por sensores ópticos possuem maior facilidade de inter-

Figura 4.13 - Árvore treinada para imagem TM com 6 classes.

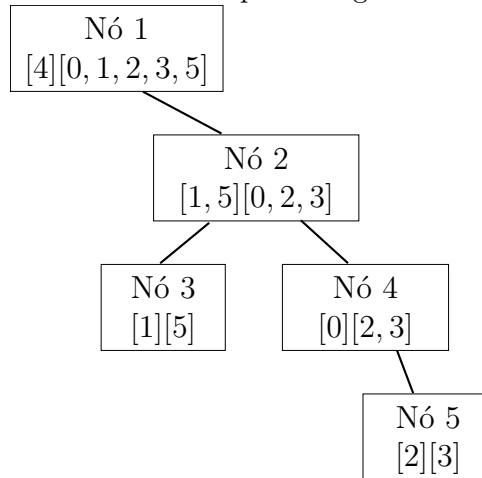


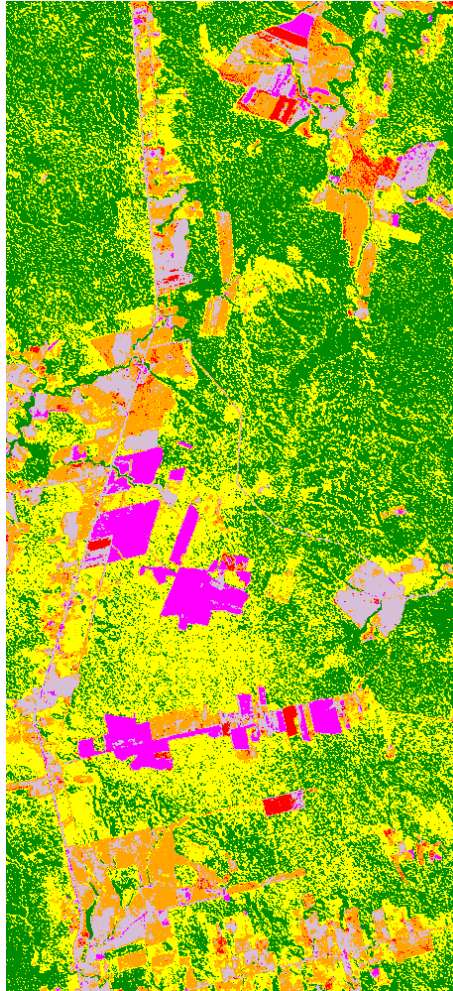
Tabela 4.11 - Imagem TM - 6 classes - HSMI - parâmetros da solução.

Nó	A_{cc}	Coeficientes Lineares			POLY			RBF
		LIN	POLY	RBF	a	c	d	γ
1	1.00	2.94	4.78	2.78	0.12	0.57	3.00	1.27
2	0.99	0.18	0.22	2.00	1.44	3.21	1.00	0.09
3	0.86	0.00	0.06	16.77	5.77	-15.09	1.00	0.03
4	0.99	0.00	0.00	0.07	9.68	-5.65	1.00	0.03
5	0.98	0.55	0.27	7.30	1.10	-18.30	1.00	0.37

pretação, as quais se relacionam com as características químicas, físicas e biológicas dos alvos, os sensores apresentam vulnerabilidade na visibilidade atmosférica. Sensores micro-ondas são imageadores ativos, denominados Radar de Abertura Sintética (*Synthetic Aperture Radar*, SAR). As imagens dos sensores SAR identificam a forma, textura e propriedades dielétricas dos alvos. Sua aplicação deve-se ao fato de apresentarem independência das condições atmosféricas e maior poder de penetração na cobertura vegetal da região (HESS et al., 1990).

As imagens de radar apresentam vantagens em relação aos dados ópticos em sua utilização nestas regiões tropicais (FLONA) por estas áreas apresentarem alta probabilidade de ocorrência de nuvens (LEWIS, 1998). O grande problema da imagem de radar é o ruído *speckle*, um ruído multiplicativo, proporcional à intensidade do sinal recebido. Ele é uma das principais causas de distorções radiométricas da imagem SAR. O ruído *speckle* gera uma textura granulosa que prejudica a interpretação das imagens SAR, apresentando menor poder discriminatório das classes.

Figura 4.14 - Imagem TM - 6 classes - classificada pelo método HSML.



4.3.1 Dados

No dia 24 de janeiro de 2006, com o intuito de observar e extrair imagens de todo o planeta, foi lançado pela Agência Espacial Japonesa (JAXA) o *Phase Array L-Band Synthetic Aperture Radar* (PALSAR), que é um radar a bordo do satélite japonês *Advanced Land Observing Satellite* (ALOS) para fins de monitoramento de desastres ambientais, levantamento de recursos naturais e, em especial, de suporte à cartografia. As principais características da imagem de Radar ALOS/PALSAR (JAXA, 2008) usada nos experimentos, são:

- Comprimento de onda: Banda L (aprox. 23 cm)
- Modo de operação: *Fine Beam Dual* (FBD)
- Polarizações: HH+HV

- Ângulo de incidência: 38,7°
- Espaçamento entre pixels: 9,36 m em *range* x 3,19 m em azimuth
- Faixa imageada: 70 km
- Resolução radiométrica: 32 bits
- Resolução espacial: ≈ 10 m em *range* x $\approx 4,5$ m em azimuth

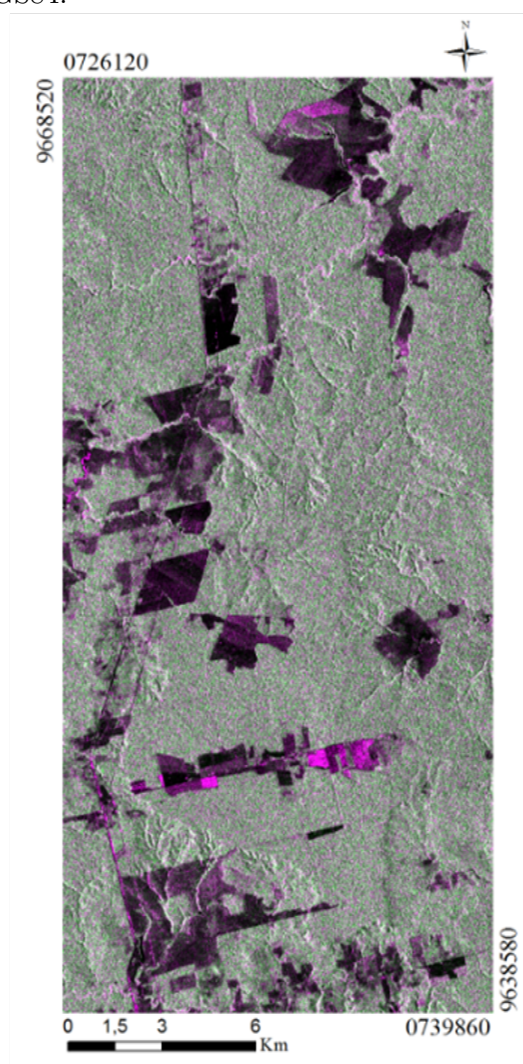
No Estudo de Caso 3 foram utilizadas duas imagens de SAR em amplitude multi-polarizadas (HH e HV) do sensor ALOS/PALSAR, modo *Fine Beam Dual* (FBD), nível de processamento 1.1, adquiridas no dia 21/06/2010, apresentada na Figura 4.15. As imagens foram ortorretificadas e para redução do ruído *speckle* foi utilizado o filtro *Stochastic Distances Nonlocal Means* (SDNLM) desenvolvido por [Torres et al. \(2014\)](#). Nesta filtragem foi utilizado o tamanho de janela igual 5x5 *pixels*, tamanho do *patch* em 3x3 *pixels* e nível de confiança dos testes estatísticos em noventa por 90%. Mais detalhes do filtro SDNLM em [Torres et al. \(2014\)](#) e para melhor compreensão do tratamento das imagens de radar utilizado neste cenário em [Reis \(2014\)](#).

Primeiramente a imagem de Radar foi classificada em um conjunto de 10 classes (Cenário 1). O experimento usando SVM com *kernel* de base radial (RBF), configurado com $\gamma = 0.004$, apresentou $\kappa = 0,333247$. A matriz de confusão é apresentada na Tabela 4.12. Usando o SVM com *kernel* polinomial, obteve-se $\kappa = 0,069$.

Tabela 4.12 - Imagem TM - 10 classes - SVM-RBF $\gamma = 0,004$

	0	1	2	3	4	5	6	7	8	9
0	90,6%	0,0%	0,0%	0,0%	0,8%	0,0%	6,6%	0,0%	0,0%	0,0%
1	0,0%	97,8%	0,0%	0,0%	3,3%	1,7%	0,0%	0,1%	0,0%	0,0%
2	0,0%	0,0%	35,4%	2,5%	0,0%	0,0%	0,0%	2,2%	23,3%	12,0%
3	0,0%	0,0%	9,3%	82,8%	0,0%	0,0%	0,0%	0,0%	2,2%	12,0%
4	1,7%	0,5%	0,0%	0,0%	89,2%	1,0%	0,0%	0,0%	0,0%	0,0%
5	0,0%	1,0%	0,0%	0,0%	6,7%	89,5%	0,0%	5,4%	1,4%	1,6%
6	7,7%	0,0%	0,0%	0,0%	0,0%	0,0%	93,4%	0,0%	0,0%	0,0%
7	0,0%	0,7%	1,4%	0,0%	0,0%	6,1%	0,0%	87,8%	4,0%	0,1%
8	0,0%	0,0%	23,3%	0,2%	0,0%	1,7%	0,0%	4,5%	42,8%	19,2%
9	0,0%	0,0%	30,6%	14,5%	0,0%	0,0%	0,0%	0,0%	26,4%	55,1%

Figura 4.15 - Imagem ALOS/PALSAR de 21 de junho de 2010, em amplitude e na composição colorida HH(R)HV(G)HH(B). Coordenadas referentes a UTM/WGS84.



Fonte: Reis (2014).

O experimento usando o método HSMI resultou em $\kappa = 0,339737$. A matriz de confusão é visualizada na Tabela 4.13. A Figura 4.16 mostra a árvore produzida nos experimentos, cujos nós têm como solução os parâmetros listados na Tabela 4.14. A imagem classificada pelo método HSMI é visualizada na Figura 4.17.

Tabela 4.13 - Imagem de Radar - 10 classes - HSMI

	0	1	2	3	4	5	6	7	8	9
0	35,5%	7,7%	0,0%	0,0%	8,8%	4,8%	3,9%	0,0%	0,0%	0,0%
1	14,8%	57,4%	0,0%	0,0%	9,7%	4,1%	0,6%	0,1%	0,0%	0,0%
2	0,0%	0,2%	15,3%	11,1%	0,1%	0,1%	0,0%	8,6%	8,5%	13,3%
3	0,0%	0,1%	17,1%	20,2%	0,0%	1,8%	0,0%	12,3%	19,0%	15,7%
4	14,3%	6,9%	0,1%	0,0%	24,2%	5,4%	3,7%	0,0%	0,0%	0,0%
5	8,1%	18,8%	0,7%	1,6%	30,2%	60,8%	0,6%	6,0%	3,5%	0,9%
6	26,9%	6,2%	0,0%	0,0%	23,4%	1,1%	91,2%	0,0%	0,0%	0,0%
7	0,2%	2,4%	29,5%	32,8%	3,4%	20,4%	0,0%	47,0%	35,7%	37,2%
8	0,2%	0,2%	21,0%	21,0%	0,2%	1,2%	0,0%	15,5%	21,3%	16,7%
9	0,0%	0,1%	16,4%	13,4%	0,0%	0,3%	0,0%	10,5%	11,9%	16,1%

Figura 4.16 - Árvore treinada para imagem de Radar com 10 classes.

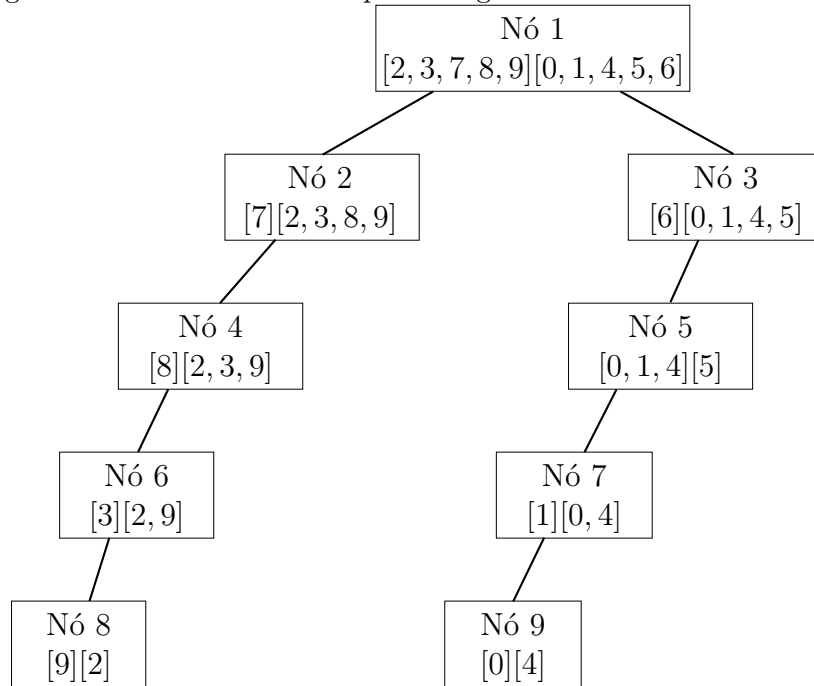
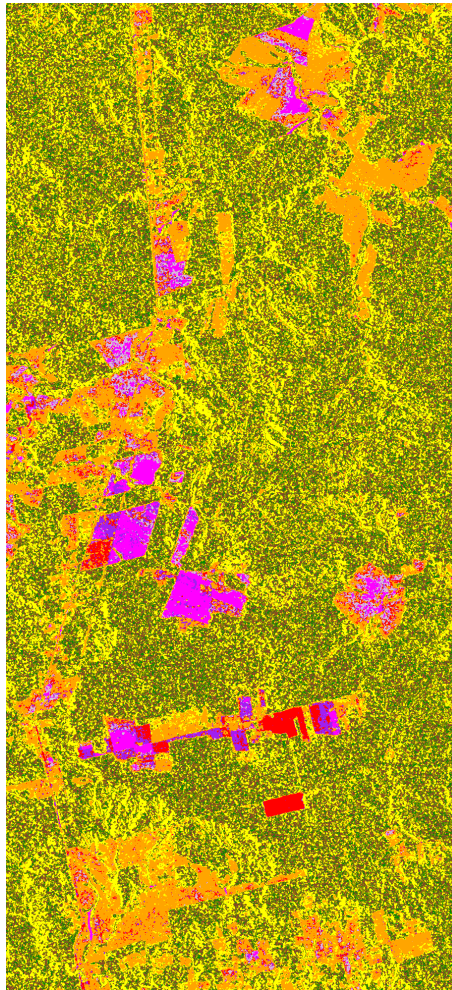


Tabela 4.14 - Imagem de Radar - 10 classes - HSMI - parâmetros da solução.

Nó	A_{cc}	Coeficientes Lineares			POLY			RBF
		LIN	POLY	RBF	a	c	d	γ
1	0.96	7.15	8.15	14.98	8.69	-1.40	1.00	9.07
2	0.60	11.73	3.67	5.55	-10.55	11.74	10.00	5.73
3	0.89	0.00	0.00	2.67	2.10	-3.87	1.00	0.00
4	0.58	0.00	4.56	20.29	3.51	-0.09	0.00	0.91
5	0.82	0.00	0.00	0.03	4.51	3.91	0.00	0.30
6	0.58	0.03	0.01	2.94	10.49	-6.37	1.00	0.19
7	0.78	0.11	0.00	17.08	18.83	14.67	2.00	0.63
8	0.54	0.34	22.97	0.46	12.73	4.65	0.00	4.47
9	0.74	0.05	0.03	17.97	20.28	-12.35	1.00	7.26

Figura 4.17 - Imagem de Radar - 10 classes - classificada pelo método HSMI.



Um segundo cenário foi proposto unindo as classes com menos contraste, resultando em seis (6) classes (Cenário 2). O número de *pixels* utilizado das amostras para a imagem radar de seis (6) classes é apresentado na Figura 4.9. Neste estudo de caso, a classe 0 (AC) possui a menor quantidade de amostras ($S_{min} = 4286$). Foram sorteados um terço ($1/3$) de 4286 *pixels* para cada conjunto, de treinamento, validação e testes, para cada uma das seis classes.

4.3.2 Resultados

O conjunto de 6 classes foi usado para experimento. O resultado para classificação com SVM-RBF, $\gamma = 0.004$, apresentou $\kappa = 0.532353$. A matriz de confusão é visualizada na Tabela 4.15. Usando o SVM com *kernel* de base radial (RBF), configurado com $\gamma = 0.167$, obteve-se $\kappa = 0.52916$. No SVM com *kernel* polinomial, obteve-se $\kappa = 0.0077$.

Tabela 4.15 - Imagem RADAR - 6 classes - SVM-RBF $\gamma = 0,004$

	0	1	2	3	4	5
0	51,7%	0,0%	1,0%	0,6%	0,1%	0,0%
1	0,8%	49,1%	0,4%	5,3%	0,0%	42,6%
2	35,4%	0,0%	58,6%	21,5%	12,7%	0,0%
3	7,4%	7,8%	25,2%	69,4%	0,4%	6,8%
4	4,0%	0,0%	14,2%	0,1%	86,8%	0,0%
5	0,6%	43,1%	0,6%	3,1%	0,0%	50,6%

O resultado do método HSMI obteve $\kappa = 0.534174$. A matriz de confusão é visualizada na Tabela 4.16.

Tabela 4.16 - Imagem RADAR - 6 classes - HSMI

	0	1	2	3	4	5
0	63,0%	0,0%	8,2%	7,1%	0,4%	0,0%
1	0,9%	45,9%	0,4%	8,3%	0,0%	39,3%
2	21,5%	0,0%	44,3%	12,1%	7,8%	0,0%
3	7,5%	3,6%	26,2%	65,6%	0,3%	3,7%
4	5,8%	0,0%	19,5%	1,3%	91,4%	0,0%
5	1,5%	50,6%	1,3%	5,7%	0,0%	56,9%

A árvore resultante do classificador HSMI está descrita na Figura 4.18. Os nós foram otimizados com parâmetros listados na Tabela 4.17.

Figura 4.18 - Árvore treinada para imagem de RADAR com 6 classes.

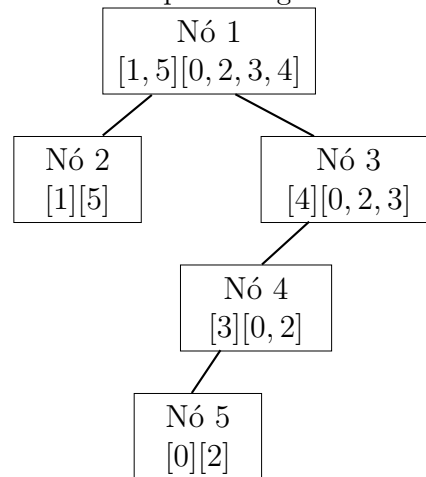


Tabela 4.17 - Imagem RADAR - 6 classes - HSMI - parâmetros da solução.

Nó	A_{cc}	Coeficientes Lineares			POLY			RBF
		LIN	POLY	RBF	a	c	d	γ
1	0.96	3.15	24.25	4.93	-4.70	-6.19	2.00	15.62
2	0.56	18.59	1.62	2.53	-2.66	-11.97	1.00	0.14
3	0.91	1.19	0.00	10.01	-1.71	17.97	0.00	0.75
4	0.80	0.00	1.98	15.12	1.98	6.94	1.00	0.69
5	0.79	0.13	0.06	16.91	1.39	12.89	1.00	1.98

5 Considerações Finais

5.1 Resumo dos Resultados Obtidos

Dentre os resultados apresentados, a função *kernel* RBF notoriamente apresenta resultados melhores quando comparado a função *kernel* polinomial, algo já conhecido na literatura. Quanto ao método HSMI comparado a função *kernel* RBF houve uma pequena melhora.

Os resultados da imagem TM com 10 classes produziu uma árvore que comprova o poder de discriminabilidade das classes confusas, colocando-as em níveis mais profundo da árvore durante o treinamento. O teste com 6 classes foi selecionado eliminando-se ou agrupando-se as classes que foram separadas nos nós mais profundos da árvore de 10 classes. Enquanto o resultado do HSMI para 10 classes não foi melhor que o SVM RBF otimizado, o HSMI para 6 classes conseguiu elevar a acurácia de classificação quando comparado ao SVM RBF otimizado.

O que deve ser levado em consideração é que o método de particionamento de entrada embutida no processo de treinamento do classificador, adicionando à classificação competitiva em cada nó da árvore e à classificação hierárquica cooperativa produz um maior conhecimento do comportamento das classes dentro do processo de classificação, gerando inclusive uma seleção automática das classes com maior e menor separabilidade. Na literatura, os outros métodos geralmente utilizam medidas estatísticas do espaço de classes, como Distância Jeffreys-Matusita - JM (DUTRA; HUBER, 1999), numa etapa prévia ao treinamento do classificador.

Para medir a discriminabilidade das classes na seleção de atributos em uma árvore de decisão, sabe-se que “uma divisão que mantém as proporções de classes em todas as partições é inútil”, já “uma divisão onde cada partição todos os exemplos são da mesma classe tem utilidade máxima”. Isto por que na árvore de decisão busca-se para cada nó o atributo que possui o maior ganho de informação. O ganho de informação mede a redução da entropia causada pela partição dos exemplos de acordo com os valores do atributo (SHANNON; WEAVER, 1948).

Na árvore de seleção de classes, o método HSMI permite que na classificação binária de cada nível da árvore as classes vão sendo ajustadas, buscando conjuntos de partições com todos os exemplos da mesma classe, fornecendo ao longo do processo o poder de discriminabilidade. Ao término da construção da árvore, no resultado final da cooperação dos nós, quando a classificação apresenta algumas confusões no nó

raiz, conclui-se que estas classes devem ser agrupadas para um melhor conhecimento de suas características antes de uma nova divisão de amostras em um novo processo de classificação.

As árvore de decisão utiliza a estratégia dividir-para-conquistar. Nesta árvore implementada também utiliza-se da estratégia dividir-para-conquistar no uso das classificações binárias das partições selecionadas. Uma diferença é que não há um processo indutor de poda para evitar o *overfitting* (super-especialização), já que a árvore pode ter altura máxima igual ao número de classes menos um.

Quando se usa uma maior quantidade vetores de suporte no processo de classificação, aumenta-se a complexidade do SVM, o risco empírico é reduzido e isto gera um maior aumento da acurácia no conjunto de treinamento. Porém na validação das amostras a acurácia de validação é reduzida, aumentando o risco funcional. Isto é avaliado dentro do SVM, em que o ajuste da função discriminante se faz necessário.

O uso da meta-heurística IWO dentro do processo de classificação do SVM permite realizar uma discussão sobre o risco empírico e funcional. Cada erva daninha representa um SVM-MKL, desta forma cada erva gera um hiperplano de ajuste. O processo de otimização busca o mínimo do risco funcional, buscando paralelamente o mínimo do risco empírico, dado que o SVM já realizou o treinamento da função discriminante.

5.2 Trabalhos Relacionados

Bruzzone e Cossu (2002) combina em cascata algoritmos paramétricos (máxima verossimilhança) e não paramétricos (rede neural RBF) para uma classificação parcialmente supervisionada. Lima (2004) faz uma revisão abrangente das técnicas de seleção e combinação de componentes em ensembles para então propor novas configurações usando componentes SVM. Ele justifica a falta de procedimentos sistemáticos na literatura para definição dos tipos de *kernels* e do parâmetro C que designa o erro aceitável na formulação do hiperplano de separação, e passa a propor métodos para essas definições. Realiza ainda experimentos com modelos de ensembles baseados em redes neurais e SVM, seleção dos componentes através de Algoritmo Genético e estuda a ordenação das componentes.

Bernardini (2006) combina classificadores simbólicos utilizando medidas de regras de conhecimento e Algoritmos Genéticos para envolver esses classificadores em um único classificador. Villanueva (2006) aplica comitês de máquinas em predição de séries

temporais indicando que esta estratégia pode conduzir a ganhos de desempenho quando comparado ao uso de um único preditor. [Coelho \(2006\)](#) propõe um ensemble de redes neurais artificiais do tipo perceptron multicamadas, que foi otimizado por uma meta-heurística imuno-inspirada (*opt-aiNet*).

[Santos et al. \(2007\)](#) combina em cascata múltiplas redes neurais MLP para a fusão de dados multitemporais. [Waske e Benediktsson \(2007\)](#) baseia-se na fusão da decisão de saídas diferentes. Cada fonte de dados é tratada separadamente e classificada por uma Máquina de Vetores Suporte (SVM). Usa dois regimes de votos para decisão final das classificações competitivas.

[Wong e Yan \(2008\)](#) combina quatro classificadores competitivos: Distância Mínima, Distância de Mahalanobis, Máxima Verossimilhança e KNN. Cinco regras de decisão bayesiana, incluindo as regras do produto, soma, máxima, mínima e a regra mediana foram utilizadas para construir um *ensemble* com a combinação dos classificadores competitivos. [Henriques \(2008\)](#) classifica imagens de ambientes coralinos, combinando classificadores de distância mínima com Máquina de Vetores de Suporte. Utiliza uma combinação de classificadores em três fases. Em cada fase é aplicado um conjunto de classificações competitivas pré-definidas e ao final é criada uma unidade combinatória das classificações parciais.

[Salvadeo \(2009\)](#) combina múltiplos classificadores, como MAXVER e SVM, utilizando combinadores treináveis e não treináveis para reconhecimento de face humana variando a extração de atributos com PCA. [Nascimento \(2009\)](#) realiza uma configuração heterogênea de *ensembles* de classificadores com a investigação em *bagging*, *boosting* e *multiboosting*, aplicando algoritmos genéticos como meio de configuração automática dos diferentes tipos de componentes.

[Araújo \(2010\)](#) gera um algoritmo para reconhecimento de características faciais baseado em filtros de correlação, usando cascata de classificadores com os métodos discriminante de Fisher e AdaBoost.

[Cavalcanti et al. \(2011\)](#) propõe uma metodologia que usa múltiplas técnicas de extração de características, com diferentes abordagens. Ao final, faz a fusão dos sub-problemas classificados. Outra parte do trabalho usa seleção dinâmica de classificadores. [Bakos et al. \(2011\)](#) realiza classificação hierárquica por árvore de decisão binária (HBDT). Utiliza o algoritmo HBDT para selecionar cadeias de processamento adequadas para a fusão de decisão. A fusão de decisão real é realizada de forma não supervisionada, usando um mapeamento ponderado dos valores de pro-

bilidade na adesão de múltiplos classificadores.

Lima (2012) constrói um comitê de SVM gerado pelo algoritmo *Adaboost*. Utiliza uma camada de aprendizagem por reforço para ajustar parâmetros do comitê evitando que desequilíbrios nos classificadores componentes do comitê prejudiquem o desempenho de generalização da hipótese final. Santos (2012) aplica aprendizado semi-supervisionado em tarefas de classificação hierárquica multirrótulo. Em um dos experimentos, utiliza ensembles de modo hierárquico para analisar o comportamento do aprendizado semi-supervisionado. Santana (2012) utiliza meta-heurísticas bio-inspiradas para seleção de atributos, uma abordagem baseada em filtro, para a construção de comitês de classificadores.

Padilha (2013) aplica Algoritmos Genéticos a um comitê de SVM com formulação por Mínimos Quadrados (LS) para selecionar os parâmetros e mensurar cada classificador SVM. Ao final, é realizada uma combinação linear das respostas de cada máquina ponderada pelos pesos. Morais (2013) propõe um comitê de classificadores dinâmico com a integração de duas etapas. A primeira etapa usa o algoritmo *bagging* e a segunda, durante a fase de operação, atribui notas a cada classificador do comitê.

Alguns trabalhos encontrados na literatura se aproximam da proposta HSMI. O método DAGSVM proposto por Platt et al. (1999) cria um gráfico direcionado acíclico (*Directed Acyclic Graph*, DAG), onde cada nó é um classificador SVM que produz uma classificação binária na sua saída. A abordagem é a mesma da estratégia um-contrum. Assim, todas as combinações de pares de classes são testadas em algum nó do grafo, sendo necessários $K(K - 1)/2$ nós. Essa abordagem não cria especialistas em particionar o espaço de entrada como proposto para o método HSMI. A quantidade de nós necessários na hierarquia DAGSVM é $K(K - 1)/2$, enquanto o modelo HSMI utiliza apenas $K - 1$ nós. Observa-se ainda que o número de nós do grafo DAGSVM cresce de forma quadrática, enquanto o HSMI possui a ordem de grandeza do número de classes do problema.

Vural e Dy (2004) usa a mesma estratégia de dividir as classes em dois subconjuntos a cada nó de uma árvore binária. Uma das suas contribuições é a comparação do tempo de teste dessa estrutura, que usa apenas $K - 1$ nós, com as abordagens um-contrum, um-contratodos e DAGSVM. Os resultados demonstraram significativa redução no tempo de teste para o método proposto.

Entretanto, a estratégia de treinamento dos nós e a escolha das classes a serem separadas em cada nó propostos por Vural e Dy (2004) não otimizam o classificador.

No que diz respeito a separação das classes, são sugeridas métricas de separação por média dos atributos das amostras de cada classe, separando as classes que possuem média maior e menor em relação a média de todas as amostras de treinamento. Sugere-se também o uso do método de vizinhos mais próximos (KNN) a partir da mesma métrica de médias das classes. Nenhuma dessas estratégias de seleção de classes e treinamento busca encontrar o particionamento que resulta na minimização do erro de classificação sob a ótica do classificador utilizado, como é feito no método HSMI.

Em relação ao treinamento do classificador de cada nó, [Vural e Dy \(2004\)](#) sugere o uso adaptativo de uma função *kernel* para cada nó, mas não propõe otimizar seus parâmetros, nem utiliza múltiplos *kernels* para especializar o classificador. O método HSMI por sua vez, contribui com a otimização dos parâmetros de uma função multi *kernel* para cada nó, com foco na acurácia da classificação. Isso resolve os dois problemas, de treinamento do classificador e de seleção das classes pertencentes as duas partições, em uma única otimização. Assim, ao mesmo tempo em que as classes são separadas buscando a maior separabilidade entre elas, isso se dá perante a especialização do classificador para tal tarefa.

[Ayat et al. \(2005\)](#) apresenta diversas referências sobre *kernels* e otimização de parâmetros. Propõe um método para calcular os hiper-parâmetros do SVM usando erro empírico. Usa experimentos para comparar os resultados com os métodos *Generalized Approximate Cross-Validation* (GAVC) e *Vapnik-Chernovenkis* (VC). [Friedrichs e Igel \(2005\)](#) e [Zhuang et al. \(2011\)](#) abordam ajustes dos parâmetros multikernels.

Alguns trabalhos que utilizam meta-heurísticas para otimização dos parâmetros do SVM: [Huang e Wang \(2006\)](#), [Omkar et al. \(2007\)](#), [Lin et al. \(2008\)](#) e [Mantovani et al. \(2015\)](#).

[Cheong et al. \(2004\)](#) converte o problema multi-classe em problema binário e utiliza uma arquitetura de árvore para essa solução. As decisões binárias são feitas pelo SVM. A separação nos nós em dois grupos é realizada de duas formas, a primeira usa o *kernel* para plotar em duas dimensões e depois divide em dois grupos com uma confusão mínima. O segundo método maximiza a distância entre dois centros de grupos buscando a minimização da variância de cada grupo. O resultado apresentado da nova metodologia estatisticamente é igual ao do SVM padrão um-contra-um.

[Madzarov et al. \(2009\)](#) utiliza classificação binária em modo árvore aplicando SVM multi-classe. As subtarefas de decisão binária do SVM usa algoritmo de clustering.

Para a consistência em relação ao modelo de SVM, o modelo de cluster utiliza medidas de distância no espaço do *kernel*, em vez do espaço de entrada. Outros trabalhos que realizam classificação binária utilizando SVM são Fei e Liu (2006), Wang et al. (2009) e Moustakidis et al. (2012).

Huang et al. (2013) utiliza IWO para otimizar os coeficientes de uma combinação linear entre um *kernel* polinomial, que apresenta boa generalização e um *kernel* gaussiano, que colabora com um bom treinamento local. Cai et al. (2013) aplica IWO para otimizar o parâmetro g do *kernel* RBF do SVM para encontrar falha em circuito elétrico. Apresenta uma tabela em que o parâmetro otimizado melhora a acurácia da classificação.

5.3 Principais Contribuições e Limitações

A primeira contribuição é a integração de diferentes conceitos, como meta-heurística bio-inspirada IWO, SVM, MKL, classificação hierárquica e particionamento do espaço de entrada, em um único método.

A segunda contribuição é a criação de um novo conceito: particionamento do espaço de entrada embutido no processo de treinamento do classificador.

A vantagem do método HSMI é permitir encontrar, durante o treinamento, níveis ou nós da árvore e as classes em que a confusão entre elas aumenta. Isso permite combinar classes durante o processo de treinamento para melhorar a acurácia da classificação.

Uma das limitações do método HSMI é o grande tempo necessário para treinar o classificador. Por outro lado, após treinado, o tempo para predição de amostras é o menor entre as estratégias um-contra-um, um-contra-todos e DAGSVM.

Outra limitação é o fato de que as imagens não foram exploradas no uso das técnicas de extração e seleção de atributos. Estes fatores poderiam consideravelmente aumentar o valor da acurácia no processo de classificação.

Também não foram testadas aqui qual efeito de limitar os coeficientes do *kernel* polinomial da função multikernel como positivos. Observa-se, no entanto, que, mesmo que em alguns casos valores negativos foram obtidos, os resultados da classificação foram consistentes.

5.4 Perspectivas Futuras

Como sugestões para trabalhos futuros, são citadas:

- Adicionar o parâmetro C do SVM no processo de otimização.
- Gerar um método *threshold* similar ao da classificação por indução de árvore de decisão, com o intuito de interromper o particionamento quando os valores de acurácia são baixos e as classes passarem a se confundir. Como a árvore possui aprendizado cooperativo, um *pre-pruning* utilizando o *threshold* não causaria *underfitting*, pois o processo de indução da árvore é conhecido e a altura é pré-definida.
- Desta forma, a boa capacidade de generalização do SVM estaria sendo explorada até o ponto de *threshold*. Desse modo, a partir de um determinado nó, podem ser adicionados outros métodos de classificação com diferentes abordagens na classificação competitiva, incluindo outros métodos de combinação de classificadores para aumentar o poder de discriminabilidade das classes confusas.
- Desenvolver um quantificador para incertezas de classificação.
- Desenvolver modelos mais apurados de computação paralela, *cluster*, para redução do tempo computacional.
- Adicionar outras funções *kernels* na função multikernel.
- Avaliar o custo computacional do método HSML.
- Adicionar a combinação de dados ópticos e de radar, realizando a fusão utilizando a metodologia HSML.

REFERÊNCIAS BIBLIOGRÁFICAS

- ALI, S.; SMITH-MILES, K. A. A meta-learning approach to automatic kernel selection for support vector machines. **Neurocomputing**, Elsevier, v. 70, n. 1, p. 173–186, 2006. [5](#)
- ARAÚJO, G. M. **Algoritmo para reconhecimento de características faciais baseado em filtros de correlação**. Dissertação (Mestrado em Engenharia Elétrica) — Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2010. [71](#)
- AYAT, N.-E.; CHERIET, M.; SUEN, C. Y. Automatic model selection for the optimization of SVM kernels. **Pattern Recognition**, Elsevier, v. 38, n. 10, p. 1733–1745, 2005. [5](#), [73](#)
- BACH, F. R.; LANCKRIET, G. R.; JORDAN, M. I. Multiple kernel learning, conic duality, and the smo algorithm. In: INTERNATIONAL CONFERENCE ON MACHINE LEARNING, 2004. **Proceedings...** [S.l.]: ACM, 2004. [5](#)
- BAKOS, K. L.; GAMBA, P.; BURA, P. Rapid estimation of point source chemical pollutant coverage in catastrophe situation using hierarchical binary decision tree ensemble and probability membership value based ensemble approaches. **Evolution in Remote Sensing (WHISPERS)**, p. 1–4, 2011. [71](#)
- BERNARDINI, F. C. **Combinação de classificadores simbólicos utilizando medidas de regras de conhecimento e algoritmos genéticos**. Tese (Doutorado) — Universidade de São Paulo, 2006. [70](#)
- BREIMAN, L. Bagging predictors. **Machine Learning**, Springer, v. 24, n. 2, p. 123–140, 1996. [7](#)
- _____. **Some infinity theory for predictor ensemble**. University of California: Berkeley, 2000. [2](#), [7](#)
- BREIMAN, L.; FRIEDMAN, J. H.; OLSHEN, R. A.; STONE, C. J. **Classification and regression trees**. Belmont: Wadsworth, 1984. [21](#), [22](#)
- BRUZZONE, L.; COSSU, R. A multiple-cascade-classifier system for a robust and partially unsupervised updating of land-cover maps. **IEEE Transactions on Geoscience and Remote Sensing**, v. 40, n. 9, p. 1984–1996, 2002. [70](#)
- BRUZZONE, L.; PERSELLO, C. A novel context-sensitive semisupervised SVM classifier robust to mislabeled training samples. **IEEE Transactions on Geoscience and Remote Sensing**, IEEE, v. 47, n. 7, p. 2142–2154, 2009. [28](#)

CAI, S.; YUAN, H.; LV, J.; CUI, Y. Application of IWO-SVM approach in fault diagnosis of analog circuits. In: CHINESE CONTROL AND DECISION CONFERENCE, 25., 2013. **Proceedings...** [S.l.]: IEEE, 2013. p. 4786–4791. [5](#), [74](#)

CAVALCANTI, G. D. d. C. et al. **Methods for dynamic selection and fusion of ensemble of classifiers**. Dissertação (Mestrado em Ciência da Computação) — Universidade Federal de Pernambuco, Recife, 2011. [71](#)

CHEONG, S.; OH, S. H.; LEE, S.-Y. Support vector machines with binary tree architecture for multi-class classification. **Neural Information Processing-Letters and Reviews**, v. 2, n. 3, p. 47–51, 2004. [73](#)

COELHO, G. P. **Geração, seleção e combinação de componentes para ensembles de redes neurais aplicadas a problemas de classificação**. Tese (Doutorado) — Universidade Estadual de Campinas, 2006. [71](#)

COGO, S. E. V. **Feições de textura para classificação de imagens**. Dissertação (Mestrado em Sensoriamento Remoto) — Universidade Federal do Rio Grande do Sul, 1994. [4](#)

COLWELL, R. N. History and place of photographic interpretation. **Manual of Photographic Interpretation**, Bethesda, American Society for Photogrammetry and Remote Sensing, v. 2, p. 33–48, 1997. [1](#), [50](#)

CRISTIANINI, N.; SHAWE-TAYLOR, J. **An introduction to support vector machines and other kernel-based learning methods**. [S.l.]: Cambridge university press, 2000. [28](#)

DAMOULAS, T.; GIROLAMI, M. A. Probabilistic multi-class multi-kernel learning: on protein fold recognition and remote homology detection. **Bioinformatics**, Oxford Univ Press, v. 24, n. 10, p. 1264–1270, 2008. [4](#), [5](#)

DIETTERICH, T. G. An experimental comparison of three methods for constructing ensembles of decision trees: bagging, boosting and randomization. **Machine Learning**, v. 40, n. 2, p. 139–157, 2000. [8](#)

DING, C. H.; DUBCHAK, I. Multi-class protein fold recognition using support vector machines and neural networks. **Bioinformatics**, Oxford Univ Press, v. 17, n. 4, p. 349–358, 2001. [4](#), [5](#)

DUDA, R. O.; HART, P. E.; STORK, D. G. **Pattern classification**. New York: Wiley Interscience, 2001. [23](#)

DUTRA, L. V.; HUBER, R. Feature extraction and selection for ERS-1/2 InSAR classification. **Int. J. Remote Sensing**, v. 20, n. 5, p. 993–1016, 1999. [69](#)

FEI, B.; LIU, J. Binary tree of SVM: a new fast multiclass training and classification algorithm. **IEEE Transactions on Neural Networks**, IEEE, v. 17, n. 3, p. 696–704, 2006. [74](#)

FONSECA, J. M. R. **Indução de árvores de decisão - HistClass - Proposta de um algoritmo não paramétrico**. Dissertação (Mestrado em Informática) — Universidade Nova de Lisboa, Lisboa, 1994. [24](#)

FREUND, Y.; SCHAPIRE, R. E. et al. Experiments with a new boosting algorithm. In: INTERNATIONAL CONFERENCE ON MACHINE LEARNING, 1996. **Proceedings...** [S.l.]: IEEE, 1996. p. 148–156. [7](#)

FRIEDRICH, F.; IGEL, C. Evolutionary tuning of multiple SVM parameters. **Neurocomputing**, Elsevier, v. 64, p. 107–117, 2005. [73](#)

GARCIA, S. C. **O uso de árvore de decisão na descoberta de conhecimento na área da saúde**. Dissertação de Mestrado — UFRGS, Lisboa, 2003. [24](#)

GÖNEN, M.; ALPAYDIN, E. Multiple kernel learning algorithms. **Journal of Machine Learning Research**, v. 12, n. Jul, p. 2211–2268, 2011. [5](#), [29](#), [30](#)

HAN, J.; KAMBER, M. **Data mining: concepts and techniques**. San Francisco: Morgan Kaufmann, 2006. [20](#)

HARALICK, R. M. Statistical and structural approaches to texture. **Proceedings of the IEEE**, IEEE, v. 67, n. 5, p. 786–804, 1979. [4](#)

HARALICK, R. M.; SHANMUGAM, K. et al. Textural features for image classification. **IEEE Transactions on Systems, Man, and Cybernetics**, IEEE, n. 6, p. 610–621, 1973. [4](#), [7](#)

HENRIQUES, A. d. P. d. M. **Classificação de imagens de ambientes coralinos: uma abordagem empregando uma combinação de classificadores e máquina de vetor de suporte**. Tese (Doutorado) — Universidade Federal do Rio Grande do Norte, 2008. [71](#)

HESS, L. L.; MELACK, J. M.; SIMONETT, D. S. Radar detection of flooding beneath the forest canopy: a review. **International Journal of Remote Sensing**, v. 11, p. 1313–1325, 1990. [60](#)

HOWLEY, T.; MADDEN, M. G. The genetic kernel support vector machine: description and evaluation. **Artificial Intelligence Review**, Springer, v. 24, n. 3-4, p. 379–395, 2005. 5

HSU, C.; CHANG, C.; LIN, C. **A practical guide to support vector classification**. 2003. Disponível em: <http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf>>. Acesso em: Novembro, 2016. 29

HUANG, C.-L.; WANG, C.-J. A GA-based feature selection and parameters optimization for support vector machines. **Expert Systems with applications**, Elsevier, v. 31, n. 2, p. 231–240, 2006. 5, 73

HUANG, H.; DING, S.; ZHU, H.; XU, X. Invasive weed optimization algorithm for optimizing the parameters of mixed kernel twin support vector machines. **Journal of Computers**, v. 8, n. 8, p. 2077–2084, 2013. 30, 74

HUNT, E. B.; MARIN, J.; STONE, P. J. **Experiments in induction**. New York: Academic Press, 1966. 20

JAXA. **ALOS/PALSAR**. 2008. Disponível em: http://www.eorc.jaxa.jp/ALOS/en/doc/fdata/PALSAR_x_Format_EL.pdf>. Acesso em: Dezembro, 2015. 61

JENSEN, J. R. **Sensoriamento remoto do ambiente: uma perspectiva em recursos terrestres**. São José dos Campos: Parêntese, 2009. 54

KOVÁCS, Z. L. **Redes neurais artificiais**. [S.l.]: Editora Livraria da Física, 2002. 2

KUNCHEVA, L. I. **Combining pattern classifiers: methods and algorithms**. [S.l.]: John Wiley & Sons, 2004. 2, 7, 9, 11, 13

LEBLANC, M.; TIBSHIRANI, R. Combining estimates in regression and classification. **Journal of the American Statistical Association**, Taylor & Francis Group, v. 91, n. 436, p. 1641–1650, 1996. 2, 7

LEWIS, D. P.; JEBARA, T.; NOBLE, W. S. Nonstationary kernel combination. In: INTERNATIONAL CONFERENCE ON MACHINE LEARNING, 23., 2006, Pittsburgh, Pennsylvania, USA. **Proceedings...** New York, NY, USA: ACM, 2006. p. 553–560. ISBN 1-59593-383-2. 30

LEWIS, H. Principle and applications of imaging radar. **Manual of Remote Sensing, American Society for Photogrammetry and Remote Sensing**, p. 811, 1998. [60](#)

LIMA, C. A. de M. **Comitê de máquinas: uma abordagem unificada empregando máquinas de vetores-suporte**. Tese (Doutorado) — Universidade Estadual de Campinas, 2004. [31](#), [70](#)

LIMA, N. H. C. **Classificação de padrões através de um comitê de máquinas aprimorado por aprendizagem por reforço**. Dissertação (Mestrado) — Universidade Federal do Rio Grande do Norte, Natal, 2012. [72](#)

LIN, S.-W.; LEE, Z.-J.; CHEN, S.-C.; TSENG, T.-Y. Parameter determination of support vector machine and feature selection using simulated annealing approach. **Applied soft computing**, Elsevier, v. 8, n. 4, p. 1505–1512, 2008. [5](#), [73](#)

LODHI, H.; SAUNDERS, C.; SHAW-TAYLOR, J.; CRISTIANINI, N.; WATKINS, C. Text classification using string kernels. **Journal of Machine Learning Research**, v. 2, n. Feb, p. 419–444, 2002. [5](#)

LORENA, A. C.; CARVALHO, A. C. de. Uma introdução às support vector machines. **Revista de Informática Teórica e Aplicada**, v. 14, n. 2, p. 43–67, 2007. [27](#)

LU, D.; WENG, Q. A survey of image classification methods and techniques for improving classification performance. **International Journal of Remote Sensing**, Taylor & Francis, v. 28, n. 5, p. 823–870, 2007. [5](#)

LU, X.; WANG, Y.; JAIN, A. K. Combining classifiers for face recognition. In: INTERNATIONAL CONFERENCE ON MULTIMEDIA AND EXPO, 2003. **Proceedings...** [S.l.]: IEEE, 2003. p. 13–16. [5](#)

MADZAROV, G.; GJORGJEVIKJ, D.; CHORBEV, I. A multi-class SVM classifier utilizing binary decision tree. **Informatica**, v. 33, n. 2, 2009. [73](#)

MANTOVANI, R. G.; ROSSI, A. L.; VANSCHOREN, J.; BISCHL, B.; CARVALHO, A. C. To tune or not to tune: recommending when to adjust SVM hyper-parameters via meta-learning. In: INTERNATIONAL JOINT CONFERENCE ON NEURAL NETWORKS, 2015. **Proceedings...** [S.l.]: IEEE, 2015. p. 1–8. [73](#)

MATERKA, A.; STRZELECKI, M. et al. Texture analysis methods – a review. **Technical University of Lodz, Institute of Electronics, COST B11 report, Brussels**, p. 9–11, 1998. 4

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: SIMPÓSIO BRASILEIRO DE SENSORIAMENTO REMOTO, 15., 2011, Curitiba. **Anais...** São José dos Campos: INPE, 2011. p. 4255–4262. ISBN 978-85-17-00056-0. Disponível em: <<http://urlib.net/dpi.inpe.br/marte/2011/06.27.14.17>>. Acesso em: Dezembro, 2015. 22, 23

MEHRABIAN, A. R.; LUCAS, C. A novel numerical optimization algorithm inspired from weed colonization. **Ecological informatics**, Elsevier, v. 1, n. 4, p. 355–366, 2006. 3, 15, 17, 18

MORAIS, P. F. T. B. d. **Geração dinâmica de comitês de classificadores através da ordenação de competências e estabelecimento de critério de corte**. Dissertação (Mestrado em Ciência da Computação) — Universidade Federal de Pernambuco, Recife, 2013. 72

MOUSTAKIDIS, S.; MALLINIS, G.; KOUTSIAS, N.; THEOCHARIS, J. B.; PETRIDIS, V. SVM-based fuzzy decision trees for classification of high spatial resolution remote sensing images. **IEEE Transactions on Geoscience and Remote Sensing**, IEEE, v. 50, n. 1, p. 149–169, 2012. 74

MULLER, K.-R.; MIKA, S.; RATSCH, G.; TSUDA, K.; SCHOLKOPF, B. An introduction to kernel-based learning algorithms. **IEEE Transactions on Neural Networks**, IEEE, v. 12, n. 2, p. 181–201, 2001. 5, 29

MURTHY, S. K.; KASIF, S.; SALZBERG, S. A system for induction of oblique decision trees. **Journal of Artificial Intelligence Research**, v. 2, n. 2, p. 1–32, 1994. 22

NAHAR, J.; ALI, S.; CHEN, Y.-P. P. Microarray data classification using automatic SVM kernel selection. **DNA and Cell Biology**, Mary Ann Liebert, Inc. 140 Huguenot Street, 3rd Floor New Rochelle, NY 10801 USA, v. 26, n. 10, p. 707–712, 2007. 4, 5

NASCIMENTO, D. S. C. **Configuração heterogênea de ensembles de classificadores**: investigação em bagging, boosting e multiboosting. Tese

(Doutorado) — Dissertação de mestrado, Universidade de Fortaleza. UNIFOR., Fortaleza, CE, 2009. 71

NEGRI, R. G.; DUTRA, L. V.; SANT'ANNA, S. J. S. An innovative support vector machine based method for contextual image classification. **ISPRS Journal of Photogrammetry and Remote Sensing**, Elsevier, v. 87, p. 241–248, 2014. 5

NETO, A. S.; BECCENERI, J. Técnicas de inteligência computacional inspiradas na natureza—aplicação em problemas inversos em transferência radiativa. **Notas em Matemática Aplicada**, v. 41, 2009. 2, 3, 15

OMKAR, S.; KUMAR, M. M.; MUDIGERE, D.; MULEY, D. Urban satellite image classification using biologically inspired techniques. In: INTERNATIONAL SYMPOSIUM ON INDUSTRIAL ELECTRONICS, 2007. **Proceedings...** [S.l.]: IEEE, 2007. p. 1767–1772. 5, 73

PADILHA, C. A. d. A. **Algoritmos genéticos aplicados a um comitê de LS-SVM em problemas de classificação**. Dissertação (Mestrado em Engenharia Elétrica e Computação) — Universidade Federal do Rio Grande do Norte, Natal, 2013. 72

PANTALEÃO, E. **Análise de cenários para classificação de dados de sensoriamento remoto usando otimização multiobjetivo e hierarquia de classes**. 109 p. Tese (Doutorado) — Instituto Nacional de Pesquisas Espaciais (INPE), São José dos Campos, 2012-02-29 2012. Disponível em: <<http://urlib.net/sid.inpe.br/mtc-m19/2012/02.29.20.00.49>>. 7, 46, 47

PARREIRA, M. O. **Avaliação experimental da imputação múltipla e composta de valores ausentes no processo de mineração de dados**. Dissertação (Mestrado em Engenharia da Computação) — Instituto Tecnológico de Aeronáutica, São José dos Campos, 2010. 20

PARREIRA, M. O.; DUTRA, L. V.; PANTALEÃO, E.; RUWER, S. G.; LU, D. Sistema classificador parreira: um método de combinação de classificações por pares de classes. In: SIMPÓSIO BRASILEIRO DE SENSORIAMENTO REMOTO. (SBSR, 17), 25-29 abr. 2015, João Pessoa. **Anais...** São José dos Campos: Instituto Nacional de Pesquisas Espaciais (INPE), 2015. p. 2929–2936. ISBN 978-85-17-0076-8. 2, 4

PLATT, J. C.; CRISTIANINI, N.; SHAWE-TAYLOR, J. Large margin DAGs for multiclass classification. In: INTERNATIONAL CONFERENCE ON NEURAL

INFORMATION PROCESSING SYSTEMS, 12., 1999, Denver. **Proceedings...** [S.l.]: MIT press, 1999. p. 547–553. [72](#)

QUINLAN, J. R. Induction of decision trees. **Machine Learning**, v. 1, n. 1, p. 81–106, 1986. [2](#), [21](#), [26](#)

_____. **C4.5**: Programs for machine learning. San Francisco: Morgan Kaufman, 1993. [19](#), [21](#)

REIS, M. **Detecção de mudanças de uso e cobertura da Terra utilizando dados óticos e de micro-ondas em uma região da Amazônia brasileira**. 331 p. Dissertação (Mestrado em Sensoriamento Remoto) — Instituto Nacional de Pesquisas Espaciais (INPE), São José dos Campos, 2014. [51](#), [52](#), [53](#), [54](#), [55](#), [62](#), [63](#)

REZENDE, S. **Sistemas inteligentes**: fundamentos e aplicações. Barueri, SP: Manole, 2003. [21](#), [25](#)

RIBEIRO, C. H. **Aprendizado em IA. Apresentação da Disciplina CT215 ITA**. 2005. Disponível em: <http://www.comp.ita.br/~carlos/texts/5-Aprendizado%20em%20IA.pdf>. Acesso em: Setembro, 2016. [26](#)

ROKACH, L.; MAIMON, O. **Data mining with decision trees. Theory and applications**. [S.l.]: World Scientific Publishing, 2008. [26](#)

RUSSEL, S.; NORVIG, P. **Inteligência artificial: uma abordagem moderna**. São Paulo: Campus, 2004. [20](#)

SALVADEO, D. H. P. **Combinação de múltiplos classificadores para reconhecimento de face humana**. Dissertação (Mestrado em Ciência da Computação) — Universidade Federal de São Carlos, São Carlos, 2009. [71](#)

SANTANA, L. E. A. d. S. **Otimização em comitês de classificadores: uma abordagem baseada em filtro para seleção de subconjuntos de atributos**. Tese (Doutorado) — Universidade Federal do Rio Grande do Norte, Natal, 2012. [72](#)

SANTOS, A. d. M. **Investigando a combinação de técnicas de aprendizado semissupervisionado e classificação hierárquica multirrótulo**. Tese (Doutorado) — Universidade Federal do Rio Grande do Norte, Natal, 2012. [72](#)

- SANTOS, R. D. C. 2011. Disponível em: <http://www.lac.inpe.br/~rafael.santos/Docs/ELAC/2010/Elac01_DM_Dia2.pdf>. Acesso em: Setembro, 2014. 20
- SANTOS, R. d. O. V. d.; FEITOSA, R. Q.; VELLASCO, M. M. B. R.; TANSCHKEIT, R. Sistemas multi-redes para classificação de imagens multitemporais. In: SIMPÓSIO BRASILEIRO DE SENSORIAMENTO REMOTO, 13., 2007, Florianópolis. **Anais...** São José dos Campos: INPE, 2007. p. 6135–6142. ISBN 978-85-17-00031-7. 71
- SCHAPIRE, R. E. The strength of weak learnability. **Machine learning**, Springer, v. 5, n. 2, p. 197–227, 1990. 7
- SCHÖLKOPF, B.; TSUDA, K.; VERT, J.-P. **Kernel methods in computational biology**. [S.l.]: MIT press, 2004. 4, 5
- SECRETARIA DE ESTADO DE MEIO AMBIENTE. **Decreto n. 73.684 - Decreto de criação da FLONA do Tapajós - 19/02/1974**. 1974. Disponível em: <http://www.sema.pa.gov.br/interna.php?idconteudocoluna=2018&idcoluna=9&titulo_conteudocoluna=73684>. Acesso em: Janeiro, 2013. 50
- SENECHAL, T.; RAPP, V.; SALAM, H.; SEGUIER, R.; BAILLY, K.; PREVOST, L. Combining AAM coefficients with LGBP histograms in the multi-kernel SVM framework to detect facial action units. In: AUTOMATIC FACE & GESTURE RECOGNITION AND WORKSHOPS. **Proceedings...** [S.l.]: IEEE, 2011. p. 860–865. 5
- SHANNON, C. E.; WEAVER, W. The mathematical theory of information. **The Bell System Technical**, v. 27, p. 379–423, 1948. Disponível em: <<http://math.harvard.edu/~ctm/home/text/others/shannon/entropy/entropy.pdf>>. Acesso em: Janeiro, 2013. 69
- SMOLA, A. J. **Advances in large margin classifiers**. [S.l.]: MIT press, 2000. 2
- SOARES, J. A. **Pré-processamento em mineração de dados: um estudo comparativo em complementação**. Tese (Doutorado em Ciências em Engenharia de Sistemas de Computação) — Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2007. 19
- THEODORIDIS, S.; KOUTROUMBAS, K. **Pattern recognition**. [S.l.]: Academic Press, 2009. 1, 28

TORRES, L.; SANT'ANNA, S. J.; FREITAS, C. da C.; FRERY, A. C. Speckle reduction in polarimetric SAR imagery with stochastic distances and nonlocal means. **Pattern Recognition**, Elsevier, v. 47, n. 1, p. 141–157, 2014. [62](#)

VACCARO, S. **Crescimento de uma floresta estacional decidual, em três estágios sucessionais, no município de Santa Tereza**. Tese (Doutorado em Engenharia Florestal, Área de Concentração de Manejo Florestas) — Universidade Federal de Santa Maria, Santa Maria, 2002. [52](#)

VAPNIK, V. N. **Statistical learning theory**. [S.l.]: Wiley New York, 1998. [28](#)

_____. An overview of statistical learning theory. **IEEE Transactions on Neural Networks**, IEEE, v. 10, n. 5, p. 988–999, 1999. [4](#), [27](#)

VARMA, M.; BABU, B. R. More generality in efficient multiple kernel learning. In: INTERNATIONAL CONFERENCE ON MACHINE LEARNING, 26., 2009. **Proceedings...** [S.l.]: ACM, 2009. p. 1065–1072. [5](#)

VENTURIERI, A. Zoneamento ecológico-econômico da área de influência da rodovia BR-163 (Cuiabá-Santarém): diagnóstico do meio sócio-econômico, jurídico e arqueologia. **Embrapa Amazônia Oriental**, p. 229–252, 2007. [52](#)

VILLANUEVA, W. J. P. **Comitê de máquinas em predição de séries temporais**. Tese (Doutorado) — Universidade Estadual de Campinas, 2006. [70](#)

VURAL, V.; DY, J. G. A hierarchical method for multi-class support vector machines. In: PROCEEDINGS OF THE TWENTY-FIRST INTERNATIONAL CONFERENCE ON MACHINE LEARNING, 2004. **Proceedings...** [S.l.]: ACM, 2004. p. 105. [72](#), [73](#)

WANG, D.-L.; LI, J.-X.; ZHENG, J.-G.; ZHOU, Y. Posterior-probability-based binary tree of support vector machine. In: INTERNATIONAL CONFERENCE ON MACHINE LEARNING AND CYBERNETICS, 2009. **Proceedings...** [S.l.]: IEEE, 2009. p. 1167–1171. [74](#)

WASKE, B.; BENEDIKTSSON, J. A. Fusion of support vector machines for classification of multisensor data. **IEEE Transactions on Geoscience and Remote Sensing**, v. 45, n. 12, p. 3858–3866, 2007. [71](#)

WEBB, A.; COPSEY, K.; CAWLEY, G. **Statistical pattern recognition**. [S.l.]: Wiley, 2011. [9](#), [10](#)

WESTON, J.; WATKINS, C. et al. Support vector machines for multi-class pattern recognition. In: EUROPEAN SYMPOSIUM ON ARTIFICIAL NEURAL NETWORKS, 1999, Bruges (Belgium). **Proceedings...** [S.l.]: D-Facto, 1999. p. 219–224. ISBN 2-600049-9-X. [4](#)

WONG, M. S.; YAN, W. Y. Investigation of diversity and accuracy in ensemble of classifiers using bayesian decision rules. **Earth Observation and Remote Sensing Applications**, p. 1–6, 2008. [71](#)

YU, L.; PORWAL, A.; HOLDEN, E.-J.; DENTITH, M. C. Towards automatic lithological classification from remote sensing data using support vector machines. **Computers & Geosciences**, Elsevier, v. 45, p. 229–239, 2012. [29](#)

ZHUANG, J.; TSANG, I. W.; HOI, S. C. Two-layer multiple kernel learning. In: **AISTATS**. [S.l.: s.n.], 2011. p. 909–917. [5](#), [73](#)

ZIEN, A.; ONG, C. S. Multiclass multiple kernel learning. In: INTERNATIONAL CONFERENCE ON MACHINE LEARNING, 24., 2007. **Proceedings...** [S.l.]: ACM, 2007. p. 1191–1198. [4](#), [5](#), [29](#)

ZIVIANI, N. **Projeto de algoritmos**: com implementações em pascal e C. São Paulo: Editora Pioneira Thompson Learning. 2^a. Edição, 2005. [2](#)

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.