UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

CARLOS ALBERTO DE ARAÚJO PADILHA

Uma Abordagem Multinível usando Algoritmos Genéticos em um Comitê de LS-SVM

Tese apresentada como requisito parcial para a obtenção do grau de Doutor em Ciência da Computação

Orientador: Prof. Dr. Dante Augusto Couto Barone
Co-orientador: Prof. Dr. Adrião Duarte Dória Neto

Porto Alegre
2018

CIP — CATALOGAÇÃO NA PUBLICAÇÃO

Padilha, Carlos Alberto de Araújo

Uma Abordagem Multinível usando Algoritmos Genéticos em um Comitê de LS-SVM / Carlos Alberto de Araújo Padilha. – Porto Alegre: PPGC da UFRGS, 2018.

108 f.: il.

Tese (doutorado) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2018. Orientador: Dante Augusto Couto Barone; Co-orientador: Adrião Duarte Dória Neto.

1. Comitê de Máquinas. 2. Algoritmos Genéticos. 3. Máquinas de Vetor de Suporte por Mínimos Quadrados. 4. Seleção de Atributos. 5. Diversidade. 6. Aprendizagem Profunda. I. Dante Augusto Couto Barone, . II. Adrião Duarte Dória Neto, . III. Título.

“If I have seen farther than others,
it is because I stood on the shoulders of giants.”
— Sir Isaac Newton

AGRADECIMENTOS

Primeiramente, gostaria de agradecer a Deus por me ajudar e proteger todos os dias da minha vida.

Agradecimento especial para minha família e minha querida namorada Paula pelo amor e incentivo para que eu me torne cada dia uma pessoa melhor em todos os sentidos e atingir os meus objetivos como profissional.

Aos meus mestres, Prof. Dr. Dante Augusto Couto Barone e Prof. Dr. Adrião Duarte Dória Neto, pela oportunidade de orientação oferecida e pelo grande exemplo como profissional e como pessoa. A estrada foi longa...

Aos companheiros de laboratório, pela convivência diária, pela grande amizade e pelas conversas que sempre resultam em idéias mirabolantes.

Ao CNPq e CAPES, pelo apoio financeiro.

RESUMO

Há muitos anos, os sistemas de comitê já tem se mostrado um método eficiente para aumentar a acurácia e estabilidade de algoritmos de aprendizado nas décadas recentes, embora sua construção tem uma questão para ser elucidada: diversidade. O desacordo entre os modelos que compõe o comitê pode ser gerado quando eles são contruídos sob diferentes circunstâncias, tais como conjunto de dados de treinamento, configuração dos parâmetros e a seleção dos algoritmos de aprendizado. O *ensemble* pode ser visto como uma estrutura com três níveis: espaço de entrada, a base de componentes e o bloco de combinação das respostas dos componentes. Neste trabalho é proposta uma abordagem multi-nível usando Algoritmos Genéticos para construir um *ensemble* de Máquinas de Vetor de Suporte por Mínimos Quadrados ou LS-SVM, realizando uma seleção de atributos no espaço de entrada, parametrização e a escolha de quais modelos irão compor o comitê no nível de componentes e a busca por um vetor de pesos que melhor represente a importância de cada classificador na resposta final do comitê. De forma a avaliar a performance da abordagem proposta, foram utilizados alguns *benchmarks* do repositório da UCI para comparar com outros algoritmos de classificação. Além disso, também foram comparados os resultados da abordagem proposta com métodos de aprendizagem profunda nas bases de dados MNIST e CIFAR e se mostraram bastante satisfatórios.

Palavras-chave: Comitê de Máquinas. Algoritmos Genéticos. Máquinas de Vetor de Suporte por Mínimos Quadrados. Seleção de Atributos. Diversidade. Aprendizagem Profunda.

ABSTRACT

Many years ago, the *ensemble* systems have been shown to be an efficient method to increase the accuracy and stability of learning algorithms in recent decades, although its construction has a question to be elucidated: diversity. The disagreement among the models that compose the *ensemble* can be generated when they are built under different circumstances, such as training dataset, parameter setting and selection of learning algorithms. The *ensemble* may be viewed as a structure with three levels: input space, the base components and the combining block of the components responses. In this work is proposed a multi-level approach using genetic algorithms to build the *ensemble* of Least Squares Support Vector Machines (LS-SVM), performing a feature selection in the input space, the parameterization and the choice of which models will compose the *ensemble* at the component level and finding a weight vector which best represents the importance of each classifier in the final response of the ensemble. In order to evaluate the performance of the proposed approach, some benchmarks from UCI Repository have been used to compare with other classification algorithms. Also, the results obtained by our approach were compared with some deep learning methods on the datasets MNIST and CIFAR and proved very satisfactory.

Keywords: Ensemble Systems, Genetic Algorithms, Least Squares Support Vector Machines, Feature Selection, Diversity, Deep Learning.

LISTA DE ABREVIATURAS E SIGLAS

AG	Algoritmo Genético
AM	Aprendizado de Máquina
BD	*Bad Diversity*
BT	Busca Tabu
DCS	*Dynamic Classifer Selection*
GD	*Good Diversity*
GEFS	*Genetic Ensemble Feature Selection*
GP	*Genetic Programming*
ILS	*Iterated Local Search*
KKT	*Karush-Kuhn-Tucker*
K-NN	*K-Nearest Neighbor*
LS	*Least Squares*
LSTM	*Long-Short Term Memory*
MCE	*Minimum Classification Error*
MEE	*Meta-Evolutionary Ensembles*
MNIST	*Modified NIST*
NCL	*Negative Correlation Learning*
NIST	*National Institute of Standards and Technology database*
PFC	*Pairwise Failure Crediting* S
PG	*Programação Genética*
PLS	*Partial Least Square*
PSO	*Particle Swarm Optimization*
RBF	*Radial Basis Function*
RF	*Random Forest*

SA *Simulated Annealing*

SRM *Structural Risk Minimization*

SVM *Support Vector Machine*

UCI *University of California, Irvine*

V-C *Vapnik-Chervonenkis*

LISTA DE FIGURAS

Figura 2.1 Modelo de aprendizagem supervisionada. 24
Figura 2.2 Hiperplano de separação ótimo. 26
Figura 2.3 (a) Ponto de dado se encontra na região de separação no lado correto do hiperplano (b) Ponto de dado se encontra na região de separação, mas no lado errado. 29

Figura 3.1 Estrutura de um ensemble. 34

Figura 4.1 Alguns esquemas de resfriamento presentes na literatura (THOMPSON; DOWSLAND, 1996; SUMAN; KUMAR, 2006; TAN, 2008) 45
Figura 4.2 Fluxograma de um Algoritmo Genético canônico. 52
Figura 4.3 Operador de cruzamento de um ponto. 55
Figura 4.4 Operador de cruzamento multiponto com dois pontos de corte. 56
Figura 4.5 Operador de cruzamento uniforme. 56
Figura 4.6 Classificação das metaheurísticas híbridas. 64

Figura 5.1 Visualização dos níveis de um comitê. 65
Figura 5.2 Método proposto - Ações realizadas no nível 1 67
Figura 5.3 Método proposto - Ações realizadas no nível 3 68
Figura 5.4 Fluxograma do método proposto. 72
Figura 5.5 Fluxograma do método híbrido proposto. 75

Figura 7.1 Comparação do número de iterações passadas até o término do processo de otimização para as bases textitImage, *Ringnorm*, *Splice*, *Waveform* e as demais, indicadas como Outros. 84
Figura 7.2 Comparação dos tempos de processamento. 85
Figura 7.3 Comparação dos tempos de processamento. 88
Figura 7.4 Arquitetura da LeNet-5, uma rede convolutiva aplicada ao problema do MNIST. Fonte (LECUN et al., 1998) 91
Figura 7.5 Arquitetura de cada rede convolutiva utilizada para o problema do MNIST em (SCHMIDHUBER, 2012). 92

LISTA DE TABELAS

Tabela 2.1 Funções típicas usadas como *kernel* 31

Tabela 4.1 Exemplos de representações de indivíduos em diferentes problemas 52

Tabela 6.1 Bases de Dados 80

Tabela 7.1 Tabela contendo os valores médios de acurácia e desvio padrão obtidos nas bases de dados - Comparação da Abordagem Genética com Outros Métodos 83

Tabela 7.2 Tabela contendo os valores de p dos testes t realizados, comparando o método proposto com os outros algoritmos. 83

Tabela 7.3 Tabela contendo os valores médios de acurácia e desvio padrão obtidos nas bases de dados - Comparação da Abordagem Genética usando Outras Funções de Aptidão 86

Tabela 7.4 Tabela contendo os valores de p dos testes t realizados, comparando o método proposto e suas variações. 86

Tabela 7.5 Tabela contendo os valores médios de acurácia e desvio padrão obtidos pela abordagem genética e versões do método híbrido. 88

Tabela 7.6 Tabela contendo os valores médios de acurácia e desvio padrão obtidos nas bases de dados - Comparação da Abordagem Híbrida com Outros Métodos 89

Tabela 7.7 Tabela contendo os valores de p dos testes t realizados, comparando o GATSLS-SVM com os outros algoritmos. 89

Tabela 7.8 Tabela contendo as taxas de erro (%) alcançadas pela nossa abordagem híbrida e outros métodos na literatura na base MNIST. 91

Tabela 7.9 Tabela contendo as taxas de erro (%) alcançadas pela nossa abordagem híbrida e outros métodos na literatura na base CIFAR-10. 93

SUMÁRIO

LISTA DE FIGURAS **9**
LISTA DE TABELAS **10**
SUMÁRIO **11**
1 INTRODUÇÃO **13**
1.1 Motivação **13**
1.2 Objetivos **15**
1.2.1 Objetivos Gerais 15
1.2.2 Objetivos Específicos 16
1.3 Estado da Arte **16**
1.3.1 Seleção de Atributos 16
1.3.2 Seleção do Comitê 18
1.3.3 Método de Combinação 21
2 MÁQUINA DE VETOR DE SUPORTE **23**
2.1 Teoria do Aprendizado Estatístico **23**
2.1.1 Princípio da Minimização do Risco Empírico 23
2.1.2 Minimização do Risco Estrutural 25
2.2 Classificação de Padrões Linearmente Separáveis **25**
2.3 Classificação de Padrões Não-Linearmente Separáveis **28**
2.4 Núcleo e Espaço de Características **29**
2.5 Formulação por Mínimos Quadrados **31**
3 *ENSEMBLES* **34**
3.1 Introdução **34**
3.2 Geração de Componentes **36**
3.3 Métodos de Combinação **39**
3.4 Diversidade **39**
4 METAHEURíSTICAS **41**
4.1 Metaheurísticas Baseadas em Solução Única **42**
4.1.1 Simulated Annealing 42
4.1.2 Busca Tabu 44
4.1.3 GRASP 46
4.1.4 Variable Neighborhood Search 48
4.1.5 Iterated Local Search 49

4.2 Algoritmos Genéticos 50
4.2.1 Representação do Cromossomo 51
4.2.2 Seleção 53
4.2.3 Cruzamento 54
4.2.4 Mutação 57
4.2.5 Atualização 57
4.2.6 Características 57
4.3 Metaheurísticas Híbridas 59
4.3.1 Classificação das Metaheurísticas Híbridas 59
4.3.1.1 Tipos de algoritmos 59
4.3.1.2 Nível de Hibridização 61
4.3.1.3 Ordem de Execução 61
4.3.1.4 Estratégia de Controle 62
5 MÉTODO PROPOSTO 65
5.1 Modelagem do Problema 66
5.1.1 Primeiro Nível 66
5.1.2 Segundo Nível 67
5.1.3 Terceiro Nível 68
5.1.4 Estrutura do Cromossomo 69
5.2 Abordagem Genética 71
5.3 Abordagem Híbrida 73
6 DESCRIÇÃO DA ANÁLISE EMPíRICA 77
6.1 Bases de Dados - *UCI Repository* 78
6.2 MNIST 80
6.3 CIFAR-10 81
7 RESULTADOS EMPíRICOS 82
7.1 Comparação da Abordagem Genética com outros Métodos 82
7.2 Comparação da Abordagem Genética usando Outras Funções de Aptidão 85
7.3 Comparação da Abordagem Híbrida com Outros Métodos 87
7.4 Resultados da Abordagem Híbrida no MNIST e CIFAR-10 90
8 CONCLUSÕES E PERSPECTIVAS FUTURAS 94
9 TRABALHOS PUBLICADOS E SUBMETIDOS 97
REFERÊNCIAS 98

1 INTRODUÇÃO

1.1 Motivação

As Máquinas de Vetor de Suporte (do inglês, Support Vector Machines ou SVM) são máquinas de aprendizagem estatística criadas por (VAPNIK, 1995), que tem grande destaque dentro da área de Aprendizado de Máquina (AM), devido a sua facilidade de uso e boa capacidade de generalização. Elas tem sido aplicadas com sucesso em numerosas aplicações, incluindo bioinformática, reconhecimento de escrita, identificação de faces, aproximação de funções e regressão, etc.

A formulação por Mínimos Quadrados da SVM (do inglês, Least Squares Support Vector Machines ou LS-SVM) introduzida por Suykens e Vandewalle (1999), encontra um hiperplano de separação ótima através da solução de um sistema de equações lineares, evitando assim o uso da programação quadrática implementada na SVM. Para que a LS-SVM consiga obter um desempenho elevado em problemas de classificação/regressão não linearmente separáveis é necessário encontrar uma função de núcleo ou *kernel* adequada para tal e, consequentemente, seu(s) parâmetro(s) interno(s). Entretanto, em problemas de classificação com alto grau de sobreposição das classes, esse não é o único problema a ser resolvido para que se possa extrair o máximo desempenho de uma LS-SVM.

Nas últimas décadas, pesquisadores vem propondo melhorar a acurácia e estabilidade dos sistemas preditivos gerados por algoritmos de aprendizado de máquina ou *machine learning*. Um método que obteve bastante destaque é o uso de sistemas que fazem a combinação de múltiplos estimadores. conhecido como comitê ou ensemble. A principal ideia de usar *ensembles* é que a combinação de diferentes classificadores individuais ou componentes, pode oferecer informação complementar sobre instâncias desconhecidas, aprimorando a qualidade da classificação geral em termos de generalização e acurácia (TAN; STEINBACH; KUMAR, 2005). O comitê é composto essencialmente de três partes ou níveis: 1) O espaço de entrada é onde os dados do problema são distribuídos para cada componente; 2) Os componentes propriamente ditos; 3) O método de combinação das respostas de classificador para gerar a resposta final.

Um comitê efetivo deve balancear a acurácia individual contra a diversidade entre seus componentes; ou seja, o comitê deve consistir de um conjunto de classificadores que não possuem apenas alta acurácia, mas cujos erros são descorrelacionados (KUNCHEVA, 2004; NASCIMENTO et al., 2011; OPITZ, 1999a; KUNCHEVA; JAIN, 2000; LI; WANG; SUNG, 2005).

Portanto, ao combinarmos estes classificadores, as falhas individuais são minimizadas. A diversidade pode ser alcançada ou gerada, quando a base de componentes é construída sob diferentes circunstâncias, tais como: conjuntos de dados, configuração de parâmetros e tipos de algoritmo de aprendizado utilizados.

Nas últimas décadas, diversas abordagens tem sido propostas de forma a otimizar certas partes ou níveis dos *ensembles*, focando em um deles ou combinação de dois, utilizando para tal algoritmos de computação evolutiva, a maior parte deles utilizando Algorítmos Genéticos (AG) ou Programação Genética (PG). Em (HO, 1998; OPITZ, 1999a; KUNCHEVA; JAIN, 2000; BREIMAN, 2001; KIM; STREET; MENCZER, 2006; BACAUSKIENE et al., 2009; EMMANUELLA; SANTANA; CANUTO, 2014), as abordagens tiveram como foco o uso de metaheurísticas, especialmente AGs, para realizar a seleção de características ou *feature selection*. Enquanto que em (ZHOU; WU; TANG, 2002; HERNáNDEZ-LOBATO et al., 2006; CHEN; TINO; YAO, 2009; PADILHA; NETO; MELO, 2012; BHOWAN et al., 2013; BHOWAN et al., 2014; LIMA; LUDERMIR, 2015; CAVALCANTI et al., 2016), a maioria deles envolveram seleção dos componentes ou classificadores base e, alguns deles, também realizaram a tarefa de parametrização dos componentes. Finalmente, as abordagens (UEDA, 2000; DEB et al., 2002; PADILHA et al., 2010; PADILHA; NETO; MELO, 2012; BHOWAN et al., 2014; MOUSAVI; EFTEKHARI, 2015) focaram no método de combinação das respostas dos componentes, propondo melhoramentos sobre os métodos baseados em fusão, seleção ou híbridos.

Nos trabalhos anteriores a esta tese (PADILHA et al., 2010; PADILHA; NETO; MELO, 2012), era usado um comitê de LS-SVM (Least Squares Support Vector Machine) com *kernel* RBF (Radial Basis Function) e combinação linear como método de combinação. Em (PADILHA et al., 2010), era focado apenas no nível do bloco de combinação, usando um Algoritmo Genético (AG) para analisar a importância de cada LS-SVM no conjunto, através de um vetor de pesos, enquanto que os parâmetros dos classificadores eram arbitrariamente escolhidos. Em (PADILHA; NETO; MELO, 2012), o uso do AG foi estendido para o nível de componentes. Agora, o AG tinha que, além de encontrar o vetor de pesos que melhor represente a contribuição de cada classificador para a resposta final do comitê, encontrar bons valores para os parâmetros σ (largura da Gaussiana) e C (termo de regularização) de cada LS-SVM. Em ambos os trabalhos, nenhum tipo de otimização havia sido utilizada no espaço de entrada do ensemble. Em (PADILHA et al., 2010), foi utilizada a mesma estratégia do Bagging (BREIMAN, 1996) para selecionar os conjuntos de treinamento de cada classificador. Já em (PADILHA; NETO; MELO, 2012), o método *Random Subspace* (HO, 1998) para deixar cada LS-SVM responsável

pela classificação de um sub-problema com uma menor dimensão do que o problema original.

Uma das propostas que serão apresentadas nesta tese é o trabalho publicado recentemente (PADILHA; BARONE; NETO, 2016), onde foi proposto um passo adiante usando o AG para guiar toda a construção de um *ensemble* de uma forma unificada. Basicamente, o AG se tornou responsável por realizar a seleção de características dos dados de entrada, a seleção de classificadores, parametrização e a escolha de um vetor de pesos de saída que representa a contribuição de cada classificador na resposta final do ensemble.

Outra proposta desta tese e que será detalhada adiante no texto, é um passo adiante no sentido de aumentar a robustez e eficiência do AG. Apesar de ter mostrado uma grande performance, os AGs possuem algumas questões que podem atrapalhar o seu desempenho como o problema da deriva genética (ASOH; MüHLENBEIN, 1994; THIERENS et al., 1998), que provoca uma baixa na variedade da população, e a demora na convergência para um ótimo global. Pensando nisso, a segunda proposta contida nesta tese é a combinação de um AG e uma metaheurística de busca local em um método híbrido que combina o poder de exploração do AG e a intensificação das buscas locais para acelerar a convergência para um ótimo global.

1.2 Objetivos

Nesta seção, os objetivos gerais e específicos desta tese são enumerados.

1.2.1 Objetivos Gerais

1. Desenvolver um algoritmo para geração de arquiteturas otimizadas baseadas em comitês de LS-SVM através do uso de um Algoritmo Genético, para que este realize a otimização em todos os níveis de um comitê de forma unificada.

2. Experimentar o algoritmo desenvolvido fazendo o uso um método híbrido que usa uma estratégia integrativa entre os Algoritmos Genéticos e metaheurísticas baseadas em solução única, substituindo o Algoritmo Genético.

1.2.2 Objetivos Específicos

1. Modelagem da função de aptidão, englobando o espaço de entrada dos dados, quantidade de classificadores, valores para os parâmetros internos das funções de *kernel* e o módulo de combinação das saídas.

2. Modelagem da acoplamento entre Algoritmo Genético e os métodos focados em busca local.

3. Comparação dos resultados com outros métodos que utilizam máquinas de comitê e otimização utilizando bases de dados largamente utilizadas pela comunidade de Aprendizado de Máquina.

4. Analisar o desempenho do método frente à bases de dados maiores, como a base de dígitos escritos manualmente, mais conhecida como MNIST (LECUN; CORTES, 2010), e a base CIFAR-10 (KRIZHEVSKY, 2009) que possui milhares de imagens de objetos.

1.3 Estado da Arte

Nesta seção, diversas abordagens envolvendo construção de *ensembles* e algoritmos evolutivos são apresentadas. Essas abordagens usaram estes algoritmos de otimização visando o aprimoramento de 1 ou 2 níveis do comitê, a maioria usando Algoritmos Genéticos ou Programação Genética.

1.3.1 Seleção de Atributos

No contexto de *ensembles*, o propósito do uso de métodos para seleção de atributos é reduzir o número de características apresentadas para a base de classificadores, além de lidar com os problemas da dimensionalidade e diversidade entre os membros de tais sistemas.

Há muitos trabalhos na literatura envolvendo métodos de seleção de atributos e *ensembles* tais como em (HO, 1998; OPITZ, 1999a; KUNCHEVA; JAIN, 2000; BREIMAN, 2001; BRYLL; GUTIERREZ-OSUNA; QUEK, 2003; GUYON; ELISSEEFF, 2003; TSYMBAL; PUURONEN; PATTERSON, 2003; KIM; STREET; MENCZER, 2006; BACAUSKIENE et al., 2009; EMMANUELLA; SANTANA; CANUTO, 2014; NETO; CANUTO, 2017). Em (HO, 1998; TSYMBAL; PUURONEN; PATTERSON, 2003), os autores demonstraram que, até uma

simples amostragem aleatória no espaço de características, pode ser considerado um método satisfatório para o aumento da acurácia em sistemas ensemble. Em (BRYLL; GUTIERREZ-OSUNA; QUEK, 2003), a abordagem é dividida em dois estágios. No primeiro momento, um subconjunto de atributos de tamanho M é encontrado através de testes de acurácia com subconjuntos de tamanhos aleatórios. No segundo estágio, a acurácia de classificação com subconjunto selecionado é avaliado. Em (WANG; TANG, 2006), a quantidade de atributos selecionados para uso é composto pelos primeiros autovetores da matriz de covariância dos dados e dimensões aleatoriamente selecionadas. Em (BREIMAN, 2001), o algoritmo *Random Forests* faz o ranqueamento dos atributos em termos da contribuição deles para a acurácia na classificação e isso pode ser usado para selecionar as características mais úteis.

Um dos primeiros trabalhos a considerar o uso de AG para realizar a seleção de atributos é (SIEDLECKI; SKLANSKY, 1989). Em (VAFAIE; JONG, 1992), o AG é aplicado para selecionar atributos para classificadores baseados em regras. No contexto de *ensembles*, (GUERRA-SALCEDO; WHITLEY, 1999; OPITZ, 1999b) utilizaram um AG para buscar sobre todo o espaço de características, mas considerando apenas um ensemble.

Guerra-Salcedo e Whitley (GUERRA-SALCEDO; WHITLEY, 1999) testaram quatro tipos de métodos de ensemble, incluindo Bagging e AdaBoost (usando o conjunto completo de atributos), o método de Ho (HO, 1998) e a abordagem deles. Os resultados mostraram que *ensembles* construídos usando características selecionadas pelo AG tiveram melhor performance, seguidas pelo método *Random Subspace* (HO, 1998).

Em (OPITZ, 1999a), o *Genetic Ensemble Feature Selection* (GEFS) utiliza subconjuntos de características com tamanho variável para promover diversidade entre os classificadores, permitindo que os atributos possam ser selecionados mais de uma vez. Os componentes do comitê são avaliados em termos de acurácia e diversidade. A diversidade é calculada como a diferença média entre a predição dos classificadores e do ensemble. GEFS obteve melhores resultados numa razão de 2/3 entre os 21 conjuntos de dados testados.

Kim et al. (KIM; STREET; MENCZER, 2006) propuseram o *Meta-Evolutionary Ensembles* (MEE) que considera múltiplas *ensembles* simultaneamente e permite que cada classificador se desloque para o comitê mais adequado. Os operados genéticos alteram o tamanho dos comitês e a filiação de cada classificador individual através do tempo. A população é inicializada com atributos aleatoriamente selecionados e atribuição aleatória a *ensembles*. Os classificadores competem uns com os outros somente se eles pertencem ao mesmo comitê. Eles são avaliados e recompensados baseados em dois critérios, acurácia e diversidade. O *ensemble* com mais alta acurácia é definido com o modelo de classificação final. Comparado aos algoritmos

tradicionais (Bagging e Boosting) e GEFS, o comitê resultante mostra performance comparável, enquanto mantém uma estrutura menor.

Em (BACAUSKIENE et al., 2009), os autores usaram ambas as abordagens por filtro e *wrapper* para selecionar características proeminentes para *ensembles* na tarefa de classificação. Essas abordagens serão explicadas mais adiante no texto. Na primeira fase, o teste estatístico t pareado é explorado para eliminar características redundantes. Na segunda, o AG é empregado para determinar os conjuntos de atributos para cada membro do ensemble. Como em (KUNCHEVA; JAIN, 2000), cada *ensemble* é codificado em um cromossomo.

Em Emmanuella et al. (EMMANUELLA; SANTANA; CANUTO, 2014) aplicaram AGs e outras duas técnicas de otimização bem conhecidas (Enxame de Partículas e Colônia de Formigas), em ambas as versões mono e bi-objetiva, para escolher subconjuntos de características para cada membro do ensemble. O procedimento de seleção de atributos usou métodos baseados em filtro que simularam a ideia de diversidade individual(versão mono-objetiva) e em grupo (versão bi-objetiva), de forma que as técnicas de otimização tentem maximizar essas medidas.

Recentemente em (NETO; CANUTO, 2017), os autores propuseram um extenso estudo exploratório usando diversas técnicas de otimização com o objetivo de selecionar, de forma automática, os atributos e os membros do comitê, em dezenas de problemas de classificação. Além disso, analisaram a performance das versões mono e multi-objetivo dessas técnicas com diferentes combinações de função objetivo.

1.3.2 Seleção do Comitê

Há vários trabalhos que envolvem a escolha do conjunto de classificadores para serem incluídos no *ensemble* final (ZHOU; WU; TANG, 2002; COELHO; LIMA; ZUBEN, 2003; HERNáNDEZ-LOBATO et al., 2006; CHEN; TINO; YAO, 2009; XIE, 2009; YANG, 2009; KAPP; SABOURIN; MAUPIN, 2010; YANG; WANG, 2010; LU; WANG, 2011; ZHANG; NIU, 2011; CANUTO; NASCIMENTO, 2012; BHOWAN et al., 2013; BHOWAN et al., 2014; NETO; CANUTO, 2017). A abordagem proposta por Zhou et al. (ZHOU; WU; TANG, 2002) chamada GASEN treina várias redes neurais primeiro. Então, o algoritmo designa pesos aleatórios para aquelas redes e emprega o AG para que os pesos evoluam, de modo que eles possam representar a aptidão dessas redes na constituição do comitê. Finalmente, ele seleciona o melhor subconjunto de classificadores para formar o *ensemble* baseado na minimização do erro de generalização do subconjunto. GASEN é iniciado utilizando 20 redes neurais e, ao término do

processo de otimização, o comitê final é composto por muito menos que os 20 componentes iniciais.

O trabalho Coelho et al. (COELHO; LIMA; ZUBEN, 2003) propõe o uso de um mecanismo evolutivo na busca do subconjunto ótimo de modelos de SVM para a construção de um comitê heterogêneo. Primeiro, os parâmetros dos modelos são escolhidos usando validação cruzada com 10 partições. Depois, uma população de *ensembles* é evoluída até que a mais apta seja encontrada. O método de combinação usado é o voto majoritário.

Em (HERNáNDEZ-LOBATO et al., 2006), os autores propuseram uma abordagem genética para poda de comitês e a compararam com outras heurísticas (*Reduce error*, *Kappa* e *Early Stopping Pruning*). Um algoritmo de poda probabilística é introduzida em (CHEN; TINO; YAO, 2009) para aproximar os pesos dos componentes utilizando o algoritmo de *Expectation Propagation* (MINKA, 2001).

Em (YANG, 2009), foi proposto a utilização de Algoritmos Imunológicos para encontrar os valores dos parâmetros de uma LS-SVM usando *kernel* de Função de Base Radial (do inglês, *Radial Basis Function* ou RBF). Os parâmetros são codificados como os genes dos anticorpos. Seus resultados mostram que há melhora significativa na performance e ela é comparável com a acurácia de outros métodos existentes como o *multi-fold crossvalidation* e *grid-search*.

Já Xie (XIE, 2009), usa Algoritmos Genéticos para encontrar os parâmetros de uma LS-SVM com *kernel* híbrido construído para tentar absorver as vantagens de cada tipo de *kernel* existente. Os resultados desse método são melhores que o de uma LS-SVM apenas com *kernel* RBF.

(KAPP; SABOURIN; MAUPIN, 2010) propõem uma abordagem para a realização de aprendizagem incremental de forma adaptativa com um comitê de SVMs. A ideia principal é acompanhar, evoluir e combinar hipóteses ótimas ao longo do tempo, baseada na otimização de processos dinâmicos e seleção do comitê. Resultados experimentais demonstraram que a estratégia proposta é promissora, uma vez que supera uma variante de classificador único da abordagem proposta e outros métodos de classificação frequentemente utilizados para a aprendizagem incremental.

No trabalho de (YANG; WANG, 2010), foi proposto um comitê de SVM em 3 "estágios". Primeiro, utilizam a técnica *Fuzzy C-means* para clusterizar o espaço de entrada dos dados e criam subconjuntos de dados. Em seguida, as SVMs são construídas com base no algoritmo PSO (*Particle Swarm Optimization*) ou, em português, Otimização por Enxame de Partículas que irá otimizar os parâmetros de seus *kernels*. Por fim, utilizam a lógica *Fuzzy* para agregar as saidas das máquinas do comitê para obter uma resposta final.

(LU; WANG, 2011) propuseram uma técnica envolvendo um comitê de SVMs para simular a predição de chuva. Primeiro, utilizam o Bagging para gerar diferentes conjuntos de treinamento para cada máquina. Então, treinam as SVMs usando diferentes tipos de *kernel* para obter as respostas baseadas em cada conjunto de treino. Em seguida, a técnica PLS (*Partial Least Square*) para selecionar os componentes do comitê. Finalmente, criam uma v-SVM (SCHÖLKOPF et al., 2000) usando o aprendizado das máquinas treinadas previamente.

(ZHANG; NIU, 2011) propuseram uma nova técnica híbrida chamada SACPSO que é a junção do *Simulated Annealing* (SA) com a técnica PSO, também visando a seleção automática dos parâmetros de uma LS-SVM usando *kernel* RBF. O SA emprega uma certa probabilidade para aprimorar a capacidade do PSO de escapar de mínimos locais, tem uma rápida convergência e alta precisão computacional. As simulações mostram que o resultado do método hibrido é melhor que o do PSO sozinho.

No trabalho de (CANUTO; NASCIMENTO, 2012), a seleção de atributos foi feita via uso de Algoritmos Genéticos para gerar diferentes subconjuntos para os classificadores do comitê. Uma função de aptidão híbrida e adaptativa é usada, considerando as abordagens de filtro e *wrapper*. Os experimentos foram conduzidos utilizando 10 diferentes tipos de máquinas de aprendizado em 14 bases de dados. Os resultados do método foram comparados com um AG usando a abordagem de filtro e com o uso do algoritmo de Bagging padrão sem uso de seleção de atributos.

Xin Yao e colaboradores tem um extenso número de trabalhos envolvendo seleção de comitê (CHANDRA; YAO, 2006; BHOWAN et al., 2013; BHOWAN et al., 2014). Em (BHOWAN et al., 2013), uma nova abordagem é proposta usando Programação Genética Multiobjetivo (MOGP) para selecionar os classificadores para compor o comitê em problemas desbalanceados. Primeiro, Bhowan et al. usaram uma função de aptidão que leva em consideração somente a acurácia em classes minoritárias e majoritárias como objetivos conflitantes no processo de aprendizagem. Então, foi analisada a adaptação da função de aptidão para evoluir soluções diversas, as quais podem ser combinadas em um comitê para aprimorar ainda mais a performance na classificação. Duas medidas na função de aptidão para estimular a diversidade entre as soluções evoluídas são investigadas, chamadas *Negative Correlation Learning* (NCL) (LIU; YAO, 1998) e *Pairwise Failure Crediting* (PFC) (CHANDRA; YAO, 2006). Em (BHOWAN et al., 2014), uma abordagem em duas etapas para evoluir *ensembles* usando Programação Genética (GP) para dados desbalanceados é desenvolvida. O primeiro passo usa o trabalho anterior deles (BHOWAN et al., 2013) cuja função de aptidão é composta pela acurácia em classes minoritárias e majoritárias e a medida PFC. O segundo passo, o qual é foco deste trabalho, a GP é

reutilizada para selecionar o menor comitê diverso por limitar a profundidade das árvores de soluções compostas.

Em um trabalho anterior (PADILHA; NETO; MELO, 2012), nós utilizamos um AG para otimizar os parâmetros internos das LS-SVMs (Largura da Gaussiana do *kernel* RBF e o parâmetro de regularização C da formulação da própria LS-SVM) que constituem o ensemble. O processo de poda foi implicitamente realizado pelo vetor de pesos quando as saídas eram combinadas. Um classificador com um peso muito baixo era descartado.

Complementando a informação dada na subseção anterior sobre o trabalho de (NETO; CANUTO, 2017), as técnicas de otimização também eram responsáveis pela seleção de membros para comitês heterogêneos, ou seja, composto por diferentes tipos de classificadores, e que se combinam usando voto majoritário.

1.3.3 Método de Combinação

Há um vasto número de métodos de combinação reportados na literatura (KUNCHEVA, 2004). Eles podem ser categorizados em três principais estratégias: baseados em fusão, seleção e métodos híbridos.

Na primeira categoria, as decisões de todos os classificadores são levadas em consideração para qualquer padrão de entrada. Exemplos de métodos baseados em fusão são: soma, combinação linear, voto majoritário, Lógica *Fuzzy*, Naïve Bayes, entre outros. Em (UEDA, 2000), diversas redes neurais são selecionadas baseadas em quais delas trabalham melhor para cada classe em termos de minimização dos erros de classificação. Então, elas são linearmente combinadas de forma a explorar as virtudes individuais de cada classificador. O critério *Minimum Classification Error* (MCE) é usado para estimar os pesos ótimos. Em (BHOWAN et al., 2014), GP é usado para escolher conjuntos de operadores de função na representação de soluções compostas, permitindo que as saídas dos membros do comitê sejam agregadas e manipuladas de diferentes maneiras. Em nossos trabalhos anteriores (PADILHA et al., 2010; PADILHA; NETO; MELO, 2012), utilizamos um AG para evoluir um vetor de pesos para melhorar a classificação do comitê. Esses pesos representavam um distribuição de probabilidade, de forma que a soma dos pesos é igual a 1, medindo a contribuição de cada membro para a resposta final.

Na segunda classe, somente um classificador é necessário para classificar corretamente um certo padrão de entrada. A escolha de um membro do comitê para fazer a decisão usualmente envolve o padrão de entrada para ser classificado. Um dos principais métodos dessa cate-

goria é o *Dynamic Classifier Selection* (DCS) (WOODS; KEGELMEYER; BOWYER, 1997).

Métodos híbridos são aqueles que usam as abordagens por fusão e seleção de forma a fornecer a saída mais apropriada dado um padrão de entrada. Frequentemente, somente um deles é usado por vez. A seleção é usada somente se o melhor classificador é bom o suficiente para realizar a classificação de um certo padrão sozinho. Caso contrário, um método de combinação por fusão é usado. Os dois principais exemplos de métodos híbridos são o DCS-MCS (DCS baseado em comportamento múltiplo classificador) e DCS-DT (DCS usando modelos de decisão).

2 MÁQUINA DE VETOR DE SUPORTE

A Máquina de Vetor de Suporte (SVM, *Support Vector Machine*) foi proposta por (VAPNIK, 1998) e consiste de uma máquina de aprendizado estatístico e, mais precisamente, uma implementação do método de minimização estrutural de risco (SRM, *Structural Risk Minimization*). Seu método de treinamento é capaz de construir um hiperplano de separação ótima que maximiza a margem de separação entre as classes, no caso de padrões linearmente separáveis. Em problemas não-linearmente separáveis são utilizadas funções de núcleo ou *kernels*, que fazem um mapeamento do espaço de entrada em um outro espaço, sendo este de alta dimensionalidade, chamado espaço de características; onde os padrões são linearmente separáveis com alta probabilidade de acordo com o teorema de Cover (COVER, 1965).

2.1 Teoria do Aprendizado Estatístico

A teoria em que se baseia o treinamento das Máquinas de Vetor de Suporte foi criada por Vladimir Vapnik e Alexey Chervonenkis. Segundo (FACELI et al., 2011), a Teoria do Aprendizado Estatístico estabelece condições matemáticas para que seja feita a escolha de um classificador a partir de um conjunto de treinamento. Tais condições levam em conta o desempenho do classificador e a sua complexidade, com a finalidade de obter um bom desempenho de generalização. Nas subseções a seguir, será mostrada a estrutura da aprendizagem supervisionada, a dimensão V-C e os princípios da minimização do risco empírico e da minimização estrutural de risco.

2.1.1 Princípio da Minimização do Risco Empírico

Um modelo de aprendizagem supervisionada é composto de três componentes interrelacionados, como na Figura 2.1.

Segundo (HAYKIN, 1999), os componentes desse modelo podem ser descritos matematicamente como:

- Ambiente: Fornece o vetor de entrada $\vec{x}$ gerado com um função de distribuição de probabilidade fixa, cumulativa e desconhecida $p(\vec{x})$.

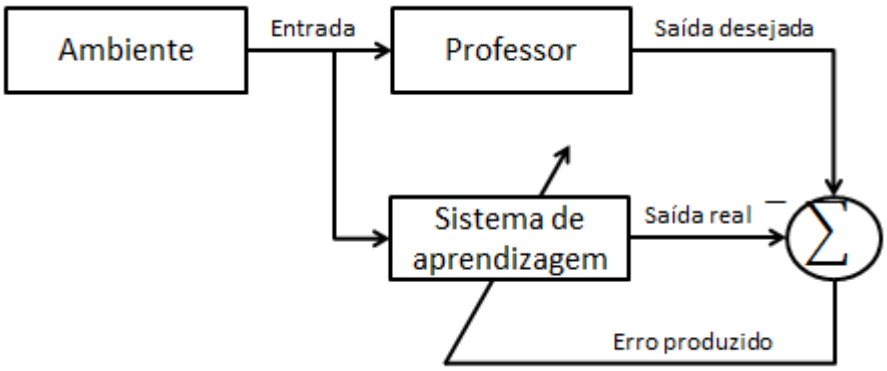


Figura 2.1: Modelo de aprendizagem supervisionada.
Fonte: O Autor

- Professor: Fornece uma resposta desejada d para cada vetor de entrada x que lhe é fornecido pelo ambiente. O vetor de entrada e a resposta desejada se relacionam através da equação:

$$d = f(\vec{x}, v) \tag{2.1}$$

onde v é um termo de ruído. Como d depende de $\vec{x}$ que depende da função p, pode-se escrever $p(\vec{x}, d)$.

- Sistema de aprendizagem: Provê o mapeamento entrada-saida através da função $y = F(\vec{x}, \vec{w})$, onde y é a saída real produzida pelo sistema de aprendizagem e $\vec{w}$, um conjunto de parâmetros livres.

O problema da aprendizagem supervisionada é encontrar a função que se aproxime da função f através do conjunto de entrada e saida desejada. Utiliza-se também um função de perda, que pode ser dada por exemplo:

$$L(x, d, f) = \begin{cases} 0, & \text{se } d = f(x); \\ 1, & \text{se } d \neq f(x). \end{cases} \tag{2.2}$$

O valor esperado de perda é definido pelo funcional de risco:

$$R(f) = \int L(x, d, f)\, dp(x, d) \tag{2.3}$$

Porém, como a função de distribuição de probabilidade p é normalmente desconhecida, o cálculo do funcional só é aplicado para dados do conjunto de entrada, que é a única informação disponível. Assim, o funcional de risco é calculado como:

$$R_e = \frac{1}{N} \sum_{i=1}^{N} L(x_i, d_i, f_i) \tag{2.4}$$

Onde N é a quantidade de exemplos do conjunto de entrada.

2.1.2 Minimização do Risco Estrutural

O princípio da minimização do risco estrutural é baseado no fato de que a taxa de erro de generalização de uma máquina de aprendizagem é limitada pela soma da taxa de erro de treinamento e por um valor que depende da dimensão V-C.

A dimensão V-C é uma medida da capacidade da família de funções de classificação realizadas pela máquina de aprendizagem. Ela representa o número máximo de exemplos de treinamento que podem ser aprendidos pela máquina corretamente, independentemente da classe à qual os exemplos pertençam. Em (HAYKIN, 1999), exemplifica-se que, para o plano cartesiano, a dimensão V-C de uma reta é 3, porque três pontos no plano sempre podem ser classificados corretamente por uma reta, independente da classe que pertençam.

Seja um espaço de hipóteses de uma estrutura aninhada de n máquinas:

$$M_1 \subset M_2 \subset \cdots \subset M_n \tag{2.5}$$

Assim, as dimensões V-C dos classificadores de padrões individuais satisfazem a condição:

$$h_1 \leq h_2 \leq \cdots \leq h_n \tag{2.6}$$

Implicando que a dimensão V-C de cada classificador de padrões é finita. Então, o método de minimização estrutural de risco objetiva encontrar uma estrutura de uma máquina de aprendizagem tal que o decréscimo da dimensão V-C ocorra à custa do menor aumento do erro de treinamento.

2.2 Classificação de Padrões Linearmente Separáveis

Para explicar a idéia básica por trás de uma máquina de vetor de suporte em um cenário mais simples, pressupõe-se que os padrões são linearmente separáveis. Então, tem-se um padrão representado por $d_i = +1$ e outro por $d_i = -1$. A equação de um hiperplano que realiza esta

separação é:

$$\mathbf{w}^T\mathbf{x} + b = 0 \tag{2.7}$$

Onde **w** é o vetor de pesos ajustáveis, **x** é o vetor de entrada e b é o bias. Assim, pode-se escrever que:

$$\begin{cases} \mathbf{w}^T\mathbf{x}_i + b \geq 0, & \text{para } d_i = +1; \\ \mathbf{w}^T\mathbf{x}_i + b < 0, & \text{para } d_i = -1. \end{cases} \tag{2.8}$$

A margem de separação, representada por ρ, é a distância entre o hiperplano e o ponto de dado mais próximo ao mesmo. O objetivo de uma máquina de vetor de suporte é encontrar um hiperplano para o qual a margem de separação ρ seja máxima. Sob esta condição, a superfície de decisão é referida como o hiperplano ótimo. A Figura 2.2 ilustra a construção de um hiperplano ótimo para um espaço de entrada bidimensional.

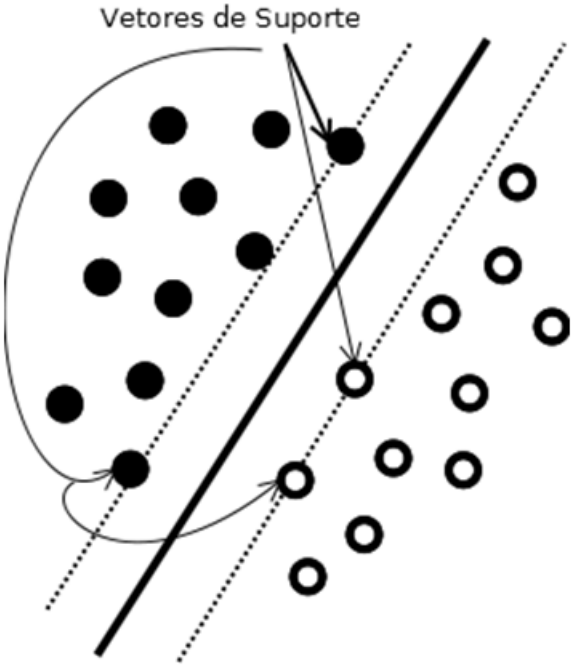


Figura 2.2: Hiperplano de separação ótimo.
Fonte: O Autor

Observando a Figura 2.2, vê-se que há dois outros hiperplanos em tracejado que serviram de base, pois passam por pontos que são amostras de treinamento. Esses pontos são chamados de vetores de suporte. Sendo então o hiperplano de separação ótimo descrito por:

$$\mathbf{w}_0^T\mathbf{x} + b_0 = 0 \tag{2.9}$$

Onde $\mathbf{w}_0$ e b_0 são os valores ótimos do vetor peso e do bias, respectivamente. Sendo, então, a distância entre um vetor de suporte e o hiperplano ótimo dada por:

$$r = \frac{|\mathbf{w}_0^T \mathbf{x}^{(s)} + b_0|}{\|\mathbf{w}_0\|} \tag{2.10}$$

Da equação acima, verifica-se que para maximizar a margem de separação entre as classes, o que corresponderia a $2r$, deve-se minimizar a norma do vetor de pesos do hiperplano ótimo. Então, a função custo é dada por:

$$\Phi(\mathbf{w}) = \frac{1}{2}\|\mathbf{w}\|^2 \tag{2.11}$$

Sujeita às restrições: $d_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1$ para $i = 1, 2, ..., N$.

Segundo (HAYKIN, 1999), este problema de otimização com restrições é chamado de problema primordial, pois a função de custo $\Phi(\mathbf{w})$ é uma função convexa de $\mathbf{w}$ e as restrições são lineares em relação a $\mathbf{w}$. Utilizando-se a teoria dos multiplicadores de Lagrange (CRISTIANINI; SHAWE-TAYLOR, 2000), o problema pode ser representado por:

$$J(\mathbf{w}, b, \alpha) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{N} \alpha_i \left[d_i(\mathbf{w}^T \mathbf{x}_i + b) - 1 \right] \tag{2.12}$$

Onde as variáveis auxiliares não-negativas α_i são chamadas de multiplicadores de Lagrange. A solução para o problema de otimização restrito é determinada pelo ponto de sela da função lagrangiana da $J(\mathbf{w}, b, \alpha)$, que deve ser minimizada em relação a $\mathbf{w}$ e a b e maximizada em relação a α. Assim, diferenciando $J(\mathbf{w}, b, \alpha)$ em relação $\mathbf{w}$ e b e igualando os resultados a zero, tem-se:

$$\frac{\partial J}{\partial \mathbf{w}} = 0 \Longrightarrow \mathbf{w} = \sum_{i=1}^{N} \alpha_i d_i \mathbf{x}_i = \mathbf{0} \tag{2.13}$$

$$\frac{\partial J}{\partial b} = 0 \Longrightarrow \sum_{i=1}^{N} \alpha_i d_i = 0 \tag{2.14}$$

O vetor solução $\mathbf{w}$ é definido em termos de uma expansão envolvendo os N exemplos de treinamento. Entretanto, apesar dessa solução ser única devido à convexidade lagrangiana, o mesmo não pode ser dito sobre os coeficientes α_i. Para tal, é possível construir um outro problema chamado de problema dual, que tem o mesmo valor ótimo do problema primordial, mas com os multiplicadores de Lagrange fornecendo a solução ótima. Assim, substituindo as equações 2.13 e 2.14 na equação 2.12, obtém-se o problema dual, que é a maximização de:

$$Q(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j d_i d_j x_i^T x_j \tag{2.15}$$

Sujeito às restrições:

1. $\sum_{i=1}^{N} \alpha_i d_i = 0$
2. $\alpha_i \geq 0$ para $i = 1, 2, ..., N$

Uma vez determinados os multiplicadores de Lagrange ótimos, representados por $\alpha_{0,i}$, pode-se calcular o vetor peso ótimo $\mathbf{w}_0$ da seguinte forma:

$$\mathbf{w}_0 = \sum_{i=1}^{N} \alpha_{0,i} d_i \mathbf{x}_i \tag{2.16}$$

Para calcular o bias ótimo b_0, utiliza-se o $\mathbf{w}_0$ obtido e assim escrever:

$$b_0 = 1 - \mathbf{w}_0^T \mathbf{x}^{(s)} \tag{2.17}$$

2.3 Classificação de Padrões Não-Linearmente Separáveis

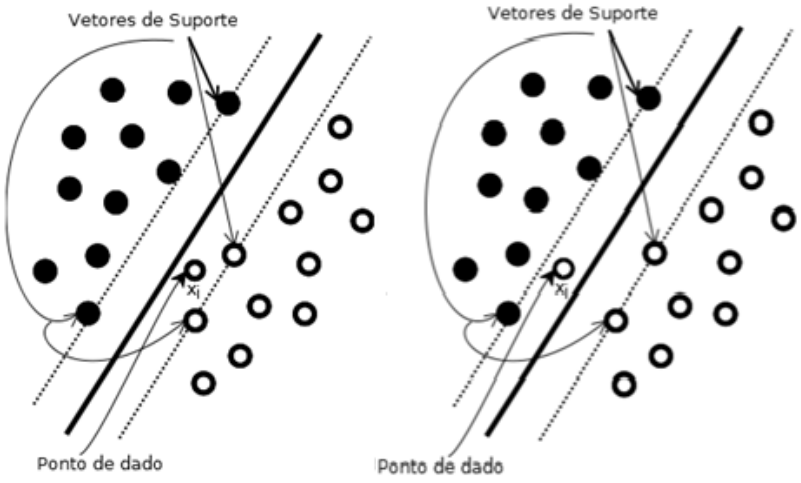
Para o caso de padrões não-linearmente separáveis, não é possível construir uma superfície de separação sem que se cometam erros de classificação. Neste caso, precisa-se encontrar o hiperplano ótimo que minimize a probabilidade de erro de classificação, calculada como a média sobre o conjunto de treinamento. Para tal, é necessário adicionar variáveis ao problema, chamadas de "variáveis soltas" ξ_i, escalares não-negativos, que medem o desvio de um ponto de dado da condição ideal de separabilidade de padrões, para gerar uma margem suave de decisão para a SVM. Para $0 \leq \xi_i \leq 1$, o ponto de dado se encontra dentro da região de separabilidade, mas no lado correto do hiperplano. Já para $\xi_i > 1$, o ponto se encontra no lado errado do hiperplano. Os dois casos estão ilustrados na Figura 2.3.

Dessa forma, o problema primordial se torna a minimização de:

$$\Phi(\mathbf{w}, \xi) = \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^{N} \xi_i \tag{2.18}$$

Sujeito a:

1. $d_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1 - \xi_i$



(a) (b)

Figura 2.3: (a) Ponto de dado se encontra na região de separação no lado correto do hiperplano (b) Ponto de dado se encontra na região de separação, mas no lado errado.
Fonte: O Autor

2. $\xi_i \geq 0$ para todo i

Onde C é um coeficiente de regularização da SVM, escolhido pelo usuário, que estabelece um equilíbrio entre a complexidade do modelo e o erro de treinamento. Assim como foi feito na seção anterior, usando os mesmos artifícios matemáticos tem-se o problema dual, que é a maximização de:

$$Q(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j d_i d_j \mathbf{x}_i^T \mathbf{x}_j \tag{2.19}$$

Sujeito a:

1. $\sum_{i=1}^{N} \alpha_i d_i = 0$

2. $0 \leq \alpha_i \leq C$ para $i = 1, 2, ..., N$

2.4 Núcleo e Espaço de Características

Como visto em seções anteriores, as máquinas de vetor de suporte são estruturas de natureza linear. A SVM forma um hiperplano, capaz de maximizar a margem de separação entre duas classes.

Em (MINSKY; PAPERT,), fala-se que uma máquina linear é bastante limitada, tendo em vista os complexos problemas de classificação de padrões que são encontrados hoje em dia.

Portanto, se faz necessário encontrar alguma forma de permitir a uma SVM atuar em problemas de classificação de forma não-linear.

Para a resolução desse problema, são utilizadas estruturas denominadas núcleos ou *kernels*. Esses núcleos geram um mapeamento entre o espaço de entrada e um espaço de alta dimensionalidade, chamado espaço de características, onde os padrões têm uma alta probabilidade de serem linearmente separáveis. O hiperplano gerado pela SVM nesse espaço de características, ao ser mapeado de volta ao espaço de entrada, se torna uma superfície não-linear.

Um *kernel* é uma função que recebe dois pontos x_i e x_j do espaço de entradas e calcula o produto escalar desses dados no espaço de características. Dado por:

$$K\left(\mathbf{x}_i, \mathbf{x}_j\right) = \phi^T\left(\mathbf{x}_i\right)\phi\left(\mathbf{x}_j\right) \tag{2.20}$$

Assim, utilizando a figura do *kernel* no problema de otimização do treinamento da SVM passa a ser a maximização de:

$$Q(\alpha) = \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j d_i d_j K(\mathbf{x}_i, \mathbf{x}_j) \tag{2.21}$$

Sujeito a:

1. $d_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i$
2. $\xi_i \geq 0$ para todo i

Para garantir a convexidade do problema de otimização e que o *kernel* apresente mapeamento nos quais seja possível o cálculo de produtos escalares, o *kernel* a ser utilizado precisa satisfazer as condições estabelecidas pelo teorema de Mercer (MERCER, 1909).

Theorem 2.4.1 (Teorema de Mercer) *Seja $X \subset R^n$ um subconjunto fechado e K uma função contínua em XxX que satisfaça as equações 2.22 e 2.23.*

$$\sum_{i,j=1}^{N} a_i a_j K(x_i, x_j) \geq 0 \tag{2.22}$$

Onde a são números reais. K é dito um $kernel$ definido positivo em X.

$$\int_X \int_X K(x,t)^2 d\mu(x) d\mu(t) < \infty \tag{2.23}$$

Então temos:

$$K(x,t) = \sum_{k=1}^{\infty} \lambda_k \phi_k(x) \phi_k(t) \tag{2.24}$$

Onde λ são os autovalores não negativos. Assim, a série converge absolutamente para cada par $(x,t) \in XxX$ e uniformemente em cada subconjunto fechado de X.

Sendo assim, os *kernels* que satisfazem o teorema de Mercer são caracterizados por dar origem a matrizes positivas semi-definidas k, em que cada elemento k_{ij} é definido por $k_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$ para todo $(i, j = 1, 2, \ldots, n)$. Na Tabela 2.1 abaixo, são mostradas algumas funções comumente utilizadas como função *kernel*.

Tabela 2.1: Funções típicas usadas como *kernel*

Tipo de *Kernel*	Expressão Matemática	Comentários
Polinomial	$(\mathbf{x}^T \mathbf{x}_i + 1)^p$	A potência p é especificada a priori pelo usuário
RBF	$\exp\left(-\frac{1}{2\sigma^2}\|\mathbf{x} - \mathbf{x}_i\|^2\right)$	A largura σ, comum a todos os núcleos, é especificada a priori pelo usuário
Perceptron	$\tanh(\beta_0 \mathbf{x}^T \mathbf{x}_i + \beta_1)$	O teorema de Mercer é satisfeito apenas para alguns valores de β_0 e β_1

2.5 Formulação por Mínimos Quadrados

Uma boa característica das SVM não-lineares é que elas conseguem resolver problemas não-lineares de classificação e regressão através de uma programação quadrática. Buscando simplificar o esforço computacional sem perder qualidade nas soluções, (SUYKENS; VANDEWALLE, 1999) propuseram uma modificação da SVM, levando a resolução de um conjunto de equações lineares ao invés da programação quadrática. A modificação da SVM chamada de Máquina de Vetor de Suporte por Mínimos Quadrados (Least Squares Support Vector Machine, LS-SVM), formula o problema de classificação como:

$$\min_{\mathbf{W},b,e} J_{LS}(\mathbf{w}, b, e) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\frac{1}{2}\sum_{k=1}^{N} e_k^2 \tag{2.25}$$

Tal que:

$$d_k \left[\mathbf{w}^T \phi\left(\mathbf{x}_k\right) + b\right] = 1 - e_k, k = 1, ..., N. \tag{2.26}$$

Onde $\phi(\cdot)$ corresponde ao mapeamento para o espaço de características de alta dimensionalidade como no caso da SVM padrão. Suykens modifica a formulação da SVM proposta por Vapnik em dois pontos, simplificando o problema. Primeiro, ao invés de restrições de desigualdade, usa restrições de igualdade onde o valor 1 no lado direito é mais considerado como um valor alvo do que um valor limite. Sobre esse valor alvo, uma variável de erro e_k é permitida de forma que erros de classificação podem ser tolerados em caso de sobreposição de distribuições. Essas variáveis de erro funcionam de forma semelhante como as variáveis de folga ξ_k nas formulações de SVM. Segundo, a função de perda quadrática é tomada para essa variável de erro.

O Lagrangiano é definido por:

$$L(\mathbf{w}, b, e; \alpha) = J_{LS}(\mathbf{w}, b, e) - \sum_{k=1}^{N} \alpha_k \{d_k[\mathbf{w}^T \phi(\mathbf{x}_k) + b] - 1 + e_k\} \tag{2.27}$$

Onde α_k são os multiplicadores de Lagrange (os quais podem positivos ou negativos agora devido às restrições de igualdade como decorre das condições Karush-Kuhn-Tucker (KKT) (FLETCHER, 1987)).

As condições de otimalidade são:

$$\begin{cases} \frac{\partial L}{\partial \mathbf{W}} = 0 \longrightarrow \mathbf{w} = \sum_{k=1}^{N} \alpha_k d_k \phi(\mathbf{x}_k), \\ \\ \frac{\partial L}{\partial b} = 0 \longrightarrow \sum_{k=1}^{N} \alpha_k d_k = 0, \\ \\ \frac{\partial L}{\partial e_k} = 0 \longrightarrow \alpha_k = C e_k, k = 1, ..., N, \\ \\ \frac{\partial L}{\partial \alpha_k} = 0 \longrightarrow d_k \left[\mathbf{w}^T \phi\left(\mathbf{x}_k\right) + b\right] = 1 - e_k = 0, k = 1, ..., N \end{cases} \tag{2.28}$$

Elas podem ser imediatamente escritas como a solução para o seguinte conjunto de equações lineares (FLETCHER, 1987):

$$\begin{bmatrix} I & 0 & 0 & Z^T \\ 0 & 0 & 0 & -D^T \\ 0 & 0 & CI & -I \\ Z & D & I & 0 \end{bmatrix} \begin{bmatrix} \mathbf{w} \\ b \\ \mathbf{e} \\ \alpha \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \tag{2.29}$$

Onde $Z = \left[\phi(x_i)^T d_1; ...; \phi(x_N)^T d_N\right]$, $D = [d_1; ...; d_N]$, $\overrightarrow{1} = [1; ...; 1]$, $\mathbf{e} = [e_1; ...; e_N]$, $\alpha = [\alpha_1; ...; \alpha_N]$. A solução é dada também por:

$$\begin{bmatrix} 0 & -D^T \\ D & ZZ^T + C^{-1}I \end{bmatrix} \begin{bmatrix} b \\ \alpha \end{bmatrix} = \begin{bmatrix} 0 \\ \overrightarrow{1} \end{bmatrix} \tag{2.30}$$

A condição de Mercer pode ser aplicada de novo para a matriz $\Omega = ZZ^T$, onde:

$$\Omega_{kl} = d_k d_l \phi(x_k)^T \phi(x_l) = d_k d_l K(x_k, x_l) \tag{2.31}$$

Onde $K(x_k, x_l)$ é a função *kernel*.

Portanto, o classificador é encontrado resolvendo as equações 2.30 e 2.31 ao invés da programação quadrática. Essa simplificação da SVM feita por (SUYKENS; VANDEWALLE, 1999), proporcionando um menor esforço computacional sem perder qualidade, motivou o uso das LS-SVM neste trabalho.

3 *ENSEMBLES*

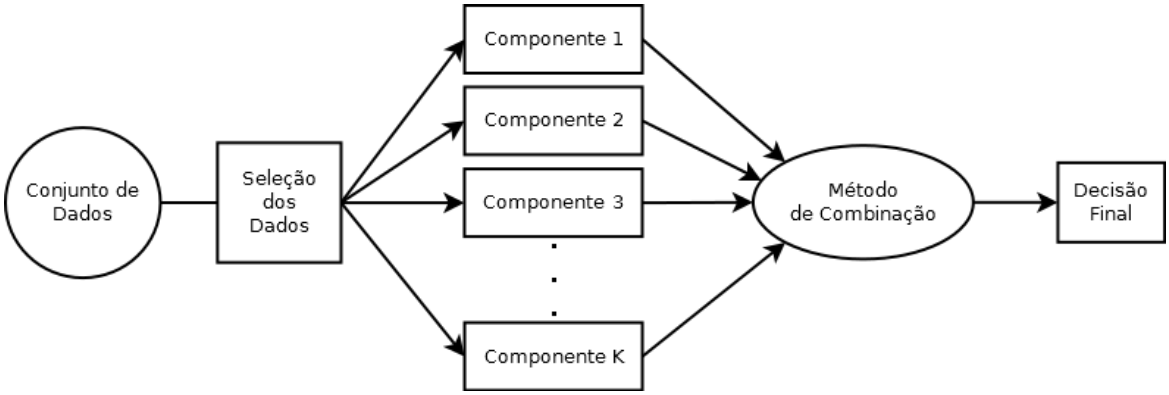
3.1 Introdução

Ensembles ou máquinas de comitê são estruturas baseadas no princípio de dividir para conquistar. Tal principio diz que um problema complexo pode ser dividido em problemas mais simples de serem resolvidos para, então, ter suas soluções combinadas em uma decisão final (KUNCHEVA, 2004). Portanto, os *ensembles* são constituídos por um conjunto de classificadores individuais organizados de forma paralela, que recebem padrões de entrada e geram saídas que são combinadas de alguma maneira em uma resposta final, teoricamente superior do que as respostas geradas individualmente (KUNCHEVA, 2004). A estrutura de um *ensemble* é apresentada na Figura 3.1, constituindo-se de três níveis, a entrada e seleção do conjunto de dados destinado a cada componente do comitê, o conjunto de classificadores e, finalmente, o método de combinação.

A estratégia de combinar máquinas de aprendizagem visando melhorar a performance não é nova. Uma ideia semelhante pode ser vista no trabalho de Nilsson (NILSSON, 1965), que utilizava uma estrutura de rede formada por uma camada de perceptrons elementares seguida de um perceptron de votação na camada seguinte. Entretanto, as pesquisas nessa área obtiveram grande impulso somente a partir da década de 90, devido à evolução dos computadores, com o surgimento de trabalhos importantes como (SCHAPIRE, 1990), (HANSEN; SALAMON, 1990), (JACOBS et al., 1991), (WOLPERT, 1992) e (PERRONE; COOPER, 1993).

O bom desempenho dos *ensembles* tem incentivado a sua utilização em todas as linhas de atuação em aprendizado de máquina, de forma que outras técnicas estão sendo empregadas

Figura 3.1: Estrutura de um ensemble.



Fonte: O Autor

para compor um comitê além de Redes Neurais Artificiais (SHARKEY, 1998). Quanto às linhas de atuação, pode-se citar:

- Classificação de padrões
  - Reconhecimento de face (GUTTA; WECHSLER, 1996), (HUANG et al., 2000)
  - Reconhecimento óptico de caracteres citepHansen1990, (DRUCKER; SCHAPIRE; SIMARD, 1993), (MAO, 1998)
  - Análise de imagens (CHERKAUER, 1996)
  - Diagnóstico médico (CUNNINGHAM; CARNEY; JACOB, 2000), (ZHOU; WU; TANG, 2002)
  - Classificação de sinais sísmicos (SHIMSHONI; INTRATOR, 1996)
- Regressão
  - Aproximação de funções (HASHEM; SCHMEISER, 1995), (HASHEM, 1997), (LIMA; COELHO; ZUBEN, 2002)
  - Predição de séries temporais (WICHARD; OGORZAŁEK, 2004), (INOUE; NARIHISA, 2000)

Em (DIETTERICH, 2000), vemos que existem 3 razões fundamentais que motivam a utilização de *ensembles*. São elas:

- Motivo estatístico: quando temos dados insuficientes, o algoritmo de aprendizagem pode encontrar várias hipóteses diferentes no espaço de busca H. Com o uso de um *ensemble* composto de bons classificadores, o algoritmo pode realizar uma média das decisões dos componentes, reduzindo, dessa forma, a chance de fazer a escolha por uma hipótese errada.
- Motivo computacional: muitos algoritmos de aprendizagem trabalham na busca por um mínimo local. Assim, usando um comitê formado por estimadores obtidos com diferentes pontos iniciais de busca, permite uma melhor aproximação da função desejada.
- Motivo representacional: em muitas casos, uma função f não pode ser representada fora do espaço de hipóteses H. Dessa forma, *ensembles* permitem obter uma representação dessa função fora de H, permitindo uma melhor aproximação de f.

Na construção de *ensembles*, há dois pontos importantes a serem considerados: a base de classificadores e o método de combinação que serão selecionados. A correta escolha do conjunto de classificadores é fundamental para o desempenho do *ensemble* como um todo. Um comitê efetivo deve consistir de um conjunto de classificadores que não possuam apenas alta acurácia, mas que apresentem erros descorrelacionados, ou seja, o *ensemble* precisa apresentar diversidade entre seus componentes (OPITZ, 1999a; KUNCHEVA; JAIN, 2000; KUNCHEVA, 2004; LI; WANG; SUNG, 2005; NASCIMENTO et al., 2011).

O próximo ponto é a escolha do método de combinação. Não há consenso na literatura sobre qual é a melhor forma de combinar as respostas dos componentes de um ensemble, exigindo um teste exaustivo para selecionar o método que melhor atende as necessidades do problema em questão. Na literatura, há vários métodos de combinação diferentes, que devido às suas características, podem ser agrupadas em três estratégias principais (KUNCHEVA, 2004): (a) baseada em fusão; (b) baseada em seleção; (c) métodos híbridos. Os métodos de combinação com estratégia baseada em fusão consideram as respostas de todos os classificadores do ensemble, enquanto aqueles métodos que usam a estratégia baseada em seleção, escolhem apenas uma resposta. A abordagem híbrida é uma mistura das duas anteriores, usando a estratégia baseada em seleção somente se o melhor classificador é bom o suficiente. Caso contrário, a estratégia baseada em fusão é usada.

3.2 Geração de Componentes

A geração de componentes para constituir um *ensemble* deve ter a preocupação de manter/proporcionar um certo nível de diversidade entre eles. Dentre os métodos mais importantes encontrados na literatura, os algoritmos de Bagging (BREIMAN, 1996) e Boosting (SCHAPIRE, 1990), são os mais mencionados.

O método de Bagging (do inglês, *bootstrap aggregating*) foi um dos primeiros algoritmos criados para a geração de um comitê. Nesta abordagem, cada membro do *ensemble* é treinado com um conjunto de dados gerado através de uma amostragem com reposição dos dados de entrada, de modo que o conjunto de treinamento de cada classificador possua a mesma quantidade de exemplos. Como a probabilidade de cada exemplo ser escolhido é igual para todos, alguns podem ser repetidos e outros deixados de fora no treinamento geral. A combinação de resultados é feita através de uma votação simples onde a resposta com maior freqüência é a escolhida.

Em (SCHAPIRE, 1990), os conjuntos de treinamento não são gerados a partir de uma

amostragem uniforme com reposição, como no método Bagging. A probabilidade de um padrão ser escolhido depende da participação deste para o erro de treinamento dos classificadores treinados anteriormente. Isto é, a probabilidade de uma amostra ser escolhida aumenta em relação as outras, caso ela tenha sido classificada incorretamente antes. Assim, o primeiro classificador é treinado seguindo uma amostragem uniforme como no Bagging. Então, as instâncias classificadas de maneira incorreta recebem um peso maior para que, no próximo conjunto de treinamento, tenham uma maior probabilidade de serem escolhidos e, assim, sucessivamente.

A partir do trabalho de Schapire (SCHAPIRE, 1990), outros algoritmos na mesma linha do Boosting foram desenvolvidos e, segundo (HAYKIN, 1999), classificados em:

- Boosting por filtragem: esta abordagem envolve filtrar os exemplos de treinamento por diferentes versões de um algoritmo de aprendizagem fraca. Ele assume, teoricamente, uma quantidade infinita de fontes de exemplos, com os exemplos sendo descartados ou não durante o treinamento e, se comparado com os outros dois modos, é o que menos requer memória.

- Boosting por subamostragem: esta abordagem trabalha com amostra de treinamento de tamanho fixo. Os exemplos são amostrados novamente durante o treinamento, de acordo com uma determinada distribuição de probabilidade de acordo com o erro dos membros do comitê para tais exemplos. Amostras que apresentaram mais erros nos treinamentos de classificadores anteriores, tem uma maior probabilidade de serem escolhidos para compor o conjunto de treinamento do próximo componente.

- Boosting por ponderação: esta abordagem também trabalha com uma amostra de treinamento fixa como a anterior, porém assume que o algoritmo de aprendizagem fraca pode receber exemplos “ponderados”.

Um dos problemas no algoritmo proposto por Schapire (SCHAPIRE, 1990), conhecido como *Boosting*, é assumir que se dispõe de um grande conjunto de dados. Dessa forma, Freund e Schapire (FREUND; SCHAPIRE, 1996) propuseram um novo método chamado de AdaBoost (*Adaptive Boosting*), que reune as principais qualidades dos algoritmos Bagging e Boosting. Assim como no Bagging, o AdaBoost realiza uma reamostragem com reposição, mas a escolha das amostras é feita de maneira adaptativa, assim como o Boosting por subamostragem. O Algoritmo 3.1 logo abaixo apresenta o AdaBoost (FREUND; SCHAPIRE, 1996).

Algoritmo 3.1: AdaBoost

Entrada: Conjunto de treinamento $\{(x_i, d_i)\}_{i=1}^{N}$, Distribuição D sobre os N exemplos rotulados, Inteiro T especificando o número de iterações e o Modelo de aprendizagem fraca

1: Inicialização: $D_1(i) = \frac{1}{N}$ para todo i

2: **para** $n = 1$ até T **faça**

3: Chame o modelo de aprendizagem fraca, utilizando a distribuição D_n

4: Retorne a hipótese $F_n : \mathbf{X} \rightarrow Y$

5: Calcule o erro da hipótese F_n

$$E_n = \sum_{i:F_n(x_i)\neq d_i} D_n(i) \tag{3.1}$$

6: Faça $\beta_n = \frac{E_n}{1-En}$

7: Atualize a distribuição D_n:

$$D_{n+1}(i) = \frac{D_n(i)}{Z_n} * \begin{cases} \beta_n, & \text{se } d_i = F_n(x_i); \\ 1, & \text{caso contrário.} \end{cases} \tag{3.2}$$

onde Z_n é uma constante de normalização.

8: **fim para**

Saída: A hipótese final é

$$F_n(\mathbf{x}) = \arg\max_{d\in D} \sum_{n:F_n(\mathbf{X})=d} \log \frac{1}{\beta_n} \tag{3.3}$$

3.3 Métodos de Combinação

Como dito antes, em (KUNCHEVA, 2004), há vários métodos de combinação reportados na literatura e eles podem ser classificados em três principais estratégias: baseados em fusão, seleção ou híbrido.

No caso dos métodos baseados em fusão, também conhecidos como métodos baseados em combinação, é assumido que todos os classificadores são igualmente capazes de realizar as tomadas de decisão. Portanto, as decisões de todos os classificadores são tomadas em consideração para qualquer padrão de entrada. Os trabalhos presentes na literatura podem ser classificados, de acordo com suas características, como linear ou não linear. A forma linear mais simples de combinar múltiplos classificadores é a soma e média das saídas. Para o caso não linear (KUNCHEVA, 2004), temos as estratégias por voto majoritário, lógica *fuzzy*, redes neurais, entre outras.

Para o caso dos métodos baseados em seleção, ao contrário do que os métodos por fusão fazem, apenas um classificador é usado para responder por um padrão de entrada. Para tal, é importante definir o processo de escolha desse membro do comitê, usualmente usando o padrão de entrada para isso. Um dos principais métodos foi proposto por (WOODS; KEGELMEYER; BOWYER, 1997), chamado de *Dynamic Classifier Selection* (DCS).

Métodos híbridos são aqueles que utilizam as técnicas de seleção e fusão, de forma a fornecer a saída mais adequada para classificar um certo padrão. Usualmente, há um critério que decide qual das técnicas utilizar em um dado momento. A ideia principal é combinar o melhor de ambas as estratégias, ou seja, usar apenas um classificador somente se ele for bom o suficiente para representar o comitê, caso contrário, o comitê inteiro é considerado. Os principais métodos híbridos presentes na literatura são o DCS-MCS (DCS *based on multiple classifier behavior* e DCS-DT (DCS *using also Decision Templates*).

3.4 Diversidade

Para que um comitê seja efetivo, ele não pode ser formado por classificadores idênticos, mas por aqueles que apresentam erros descorrelacionados. Segundo (KUNCHEVA, 2004; NASCIMENTO et al., 2011; JOHANSSON; LOFSTROM, 2012), há basicamente quatro formas de criar diversidade em comitês (três se considerarmos que escolher subconjuntos de padrões e atributos, como a seleção de conjuntos de treinamento):

- **Diferentes conjuntos de treinamento** - Nesta abordagem, múltiplos conjuntos de treinamento são criados por reamostragem dos dados originais seguindo alguma distribuição e cada classificador é treinado usando um desses conjuntos gerados. Métodos tais como *Bagging* (BREIMAN, 1996) e *Boosting* (SCHAPIRE, 1999) selecionam conjuntos de treinamento com diferentes padrões.

- **Diferentes atributos de entrada** - Nesta abordagem, somente um subconjunto dos atributos dos dados é selecionado para cada conjunto de treinamento. Essa seleção de atributos pode ser feita utilizando as abordagens filtro e *wrapper*. Métodos filtro são independentes de qualquer algoritmo de aprendizado, suas funções de avaliação consideram apenas as propriedades dos dados. Já os modelos *wrappers*, utilizam um modelo para avaliar os subconjuntos de atributos.

- **Diferentes configurações de parâmetros** - Nesta abordagem, a diversidade pode ser alcançada através do uso de diferentes configurações de parâmetros iniciais nos métodos de classificação. Como exemplo, no caso de redes neurais, isso significaria ter uma variação no conjunto de pesos e na topologia do modelo.

- **Diferentes algoritmos de aprendizagem** - A escolha de diferentes tipos de algoritmos para constituir um *ensemble* (comitê heterogêneo) pode contribuir com a geração de diversidade. Isso significa que em um *ensemble* composto por SVM, árvores de decisão e redes neurais é mais provável que haja desacordo entre os classificadores do que em um composto por somente um desses algoritmos.

4 METAHEURÍSTICAS

Uma metaheurística é um algoritmo desenvolvido para resolver, de maneira aproximada, problemas de otimização complexos sem a necessidade de uma grande adaptação para cada problema, ao contrário das heurísticas específicas. Problemas de otimização complexos são aqueles que não podem ser resolvidos de forma ótima ou por qualquer método exato em um tempo razoável. Eles podem ser divididos em diversas categorias dependendo se são problemas contínuos ou discretos, mono ou multi-objetivo, com ou sem restrição, estáticos ou dinâmicos. Em problemas complexos reais como os encontrados em áreas indo das finanças até ao gerenciamento de produção e engenharia, as metaheurísticas são geralmente utilizadas onde faltam soluções heurísticas satisfatórias. O aumento significativo do poder de processamento dos computadores unido ao desenvolvimento de arquiteturas altamente paralelizadas tem amenizado o tempo de processamento das metaheurísticas, naturalmente alto, e tem impulsionado o seu desenvolvimento.

De uma forma geral, as metaheurísticas possuem as seguintes características: (a) inspiradas na natureza, ou seja, são baseadas em princípios da física ou biologia; (b) usam componentes estocásticos; (c) não fazem uso de derivada; (d) possuem parâmetros que precisam ser ajustados ao problema.

O bom desempenho de uma metaheurística em um problema de otimização dependerá se ela conseguir balancear a exploração e a intensificação. Exploração consiste em sondar diferentes porções do espaço de busca visando identificar soluções promissoras e diversificar a busca, a fim de evitar ficar preso em um ótimo local. Já a intensificação, sonda uma porção limitada do espaço de busca com a esperança de melhorar a solução que já se tem em mãos. De acordo com (BIRATTARI et al., 2001), a forma particular como cada metaheurística alcança esse balanço é o que mais as diferencia. Há vários aspectos que podem ser usados para classificar esses algoritmos, tais como o processo de busca, uso de memória, vizinhança, quantidade de soluções mantidas para a próxima geração, maneira de fugir de mínimos locais, etc. Por focarmos no uso de Algoritmos Genéticos, nesse trabalho as metaheurísticas serão divididas em dois grupos, baseadas em solução única (Seção 4.1) e Algoritmos Genéticos (Seção 4.2), e haverá um melhor detalhamento logo a seguir.

4.1 Metaheurísticas Baseadas em Solução Única

As metaheurísticas baseadas em solução única, ao contrário daquelas baseadas em população, inicializam o processo de busca com uma solução inicial e, então, se movem no espaço de busca descrevendo uma trajetória, por isso eles também são conhecidos como métodos de trajetória. Os principais algoritmos que se enquadram nessa categoria são o *Simulated Annealing*, Busca Tabu, GRASP, *Iterated Local Search*, *Variable Neighborhood Search* e outras variantes, os quais serão detalhados nesta seção.

4.1.1 Simulated Annealing

O algoritmo *Simulated Annealing* ou têmpera simulada, em português, foi proposto independentemente por Kirkpatrick et al. (KIRKPATRICK; GELATT; VECCHI, 1983) e por Cerny (CERNY, 1985) e se baseia no processo metalúrgico de têmpera para obter um estado sólido de energia mínima onde a liga metática é bastante forte.

O processo de têmpera consiste em inicialmente elevar a temperatura de um sólido em banho térmico, fazendo com que as partículas do material se distribuam aleatoriamente numa fase líquida, e então, gradualmente resfriando o objeto até a temperatura ambiente, de maneira que todas as partículas fiquem arranjadas em um estado de baixa energia, onde a solidificação acontece.

Simulated Annealing transpõe o processo de têmpera para resolver problemas de otimização: a função objetivo do problema f que, de forma similar à energia do material, é minimizada e o algoritmo usa um parâmetro T que simula a temperatura do objeto.

O algoritmo começa com uma solução inicial s, escolhida de forma aleatória ou através do uso de alguma heurística, e inicializando o parâmetro de temperatura T. Então, a cada iteração, uma solução s' é escolhida aleatoriamente na vizinhança de s que é a solução atual do problema. A solução atual do problema é atualizada dependendo dos valores da função objetivo para s e s' ($f(s)$ e $f(s')$) e também do valor atual de T. Caso $f(s') <= f(s)$ (considerando um problema de minimização), o valor de s é atualizado com s'. Por outro lado, se $f(s') > f(s)$, s' ainda poderá ser aceita como solução atual com uma probabilidade $p(f(s), f(s'), T) = \exp(-\frac{f(s')-f(s)}{T})$ (algoritmo de Metropolis (METROPOLIS et al., 1953)). Esse processo por um número máximo de iterações ($SAmax$) para cada temperatura T. A temperatura T é reduzida lentamente durante o processo de busca usando um fator de resfriamento α, tal que $T_k = \alpha * T_{k-1}$, sendo $0 < \alpha < 1$. Isso quer dizer que, no início da busca, a probabilidade do algoritmo

aceitar soluções de piora é alta e, então, diminui gradativamente com o passar das iterações. Este processo é apresentado no Algoritmo 4.1.

Algoritmo 4.1: Simulated Annealing

```
1: Escolha de forma aleatória uma solução inicial s para o problema
2: Inicialize a temperatura T
3: IterT = 0
4: enquanto critério de parada não é atingido faça
5:    enquanto IterT < SAmax faça
6:       IterT = IterT + 1
7:       Selecione aleatoriamente uma solução s' da vizinhança N(s)
8:       se f(s') <= f(s) então
9:          s = s'
10:      senão
11:         Gere um valor aleatório v entre 0 e 1
12:         Probabilidade de aceitação p(f(s), f(s'), T) = exp(-(f(s')-f(s))/T)
13:         se v <= p(f(s), f(s'), T) então
14:            s = s'
15:         fim se
16:      fim se
17:   fim enquanto
18:   IterT = 0
19:   T = α * T
20: fim enquanto
21: retorne A melhor solução encontrada
```

Em (THOMPSON; DOWSLAND, 1996; SUMAN; KUMAR, 2006; TAN, 2008), há diversas outras maneiras de realizar o resfriamento da temperatura presentes na literatura. Na Figura 4.1, são apresentados alguns esquemas de resfriamento que usam informações como a temperatura no ciclo i (T_i), onde $0 <= i <= N$ (N é o número máximo de ciclos determinada pelo usuário), além das temperaturas inicial e final, T_0 e T_N, respectivamente.

O resfriamento monotônico aditivo, apresentado com quatro variantes nas figuras 4.1(a)-(d), calcula a temperatura do sistema no ciclo i adicionando à T_N um termo que tem seu valor decrementado em relação a i. As seguintes variantes são mostradas na Figura 4.1:

- Aditivo linear: $T_i = T_0 - i * \frac{(T_0 - T_N)}{N}$
- Aditivo quadrático: $T_i = T_N + (T_0 - T_N) * (\frac{N-i}{N})^2$
- Aditivo exponencial: $T_i = T_N + (T_0 - T_N) * \frac{1}{(1+\exp(\frac{2*\log(T_0 - T_N)}{N} * (i - \frac{1}{2} * N))}$
- Aditivo trigonométrico: $T_i = T_N + \frac{1}{2}(T_0 - T_N) * (1 + \cos(\frac{i*\pi}{N}))$

Além disso, o resfriamento monotônico multiplicativo é apresentado nas figuras 4.1(e)-(h). Este esquema calcula a temperatura T_i multiplicando T_0 por um fator que decrementa em relação a i. Segue algumas variantes desse esquema:

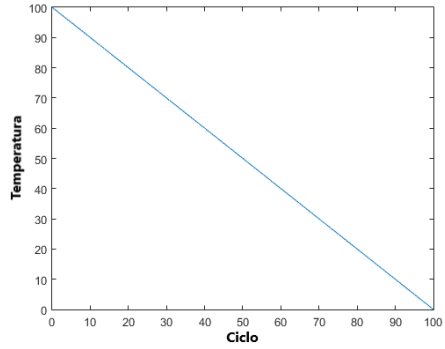
- Multiplicativo linear: $T_i = \frac{T_0}{1+\alpha*i}$ com $\alpha > 0$
- Multiplicativo quadrático: $T_i = \frac{T_0}{1+\alpha*i^2}$ com $\alpha > 0$
- Multiplicativo exponencial: $T_i = T_0 * \alpha^i$ com $0.8 \leq \alpha \leq 0.9$
- Multiplicativo logarítmico: $T_i = \frac{T_0}{1+\alpha*\log(1+i)}$ com $\alpha > 1$

4.1.2 Busca Tabu

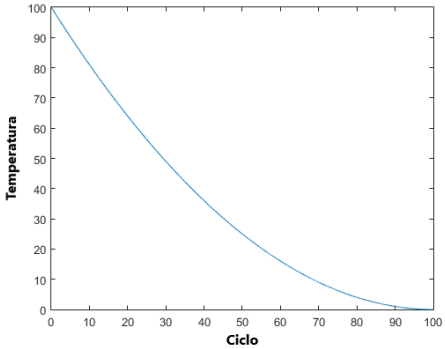
A busca tabu, do inglês *tabu search* (TS), foi formalizado em 1986 por Glover (GLOVER, 1986), mesmo trabalho que introduziu o termo "meta-heurística". TS usa explicitamente o histórico da busca como uma memória adaptativa, possibilitando escapar de mínimos locais e implementar uma estratégia de exploração. De fato, o uso dessa memória das soluções já encontradas no espaço de busca, torna a busca mais econômica e efetiva. Olhando desse ponto de vista, o SA faz o caminho oposto, que não usa memória e, dessa forma, é incapaz de aprender com o passado e pode perder uma solução melhor que foi encontrada durante o processo de busca.

O algoritmo inicia com uma solução $s = s_0$ e, a cada iteração, explora um subconjunto de soluções V na vizinhança $N(s)$ da solução atual s. A melhor solução encontrada nesse subconjunto s' se torna a solução atual s, mesmo que possua um valor $f(s') > f(s)$, considerando um problema de minimização. Essa estratégia de escolha do melhor vizinho visa escapar de mínimos locais; entretanto, isso pode fazer com que o algoritmo fique em um ciclo infinito, retornando à soluções visitadas anteriormente.

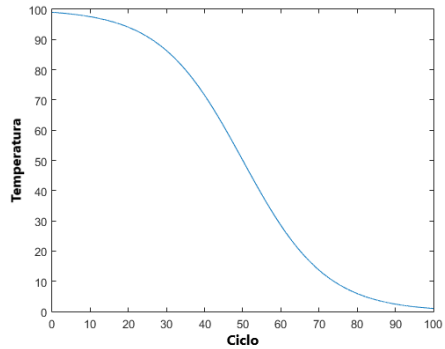
Para evitar isso, a TS utiliza a lista tabu T, que dá nome ao algoritmo, a qual armazena as últimas soluções encontradas e, assim, impede que elas sejam revisitadas. Ela funciona como uma lista de tamanho fixo que armazena as últimas $|T|$ soluções visitadas, ou seja, a cada nova solução incluída na lista, a mais antiga é removida. Assim, na exploração do subconjunto V da vizinhança $N(s)$ da solução atual, a TS fica proibida de revisitar as soluções s' já vistas anteriormente, enquanto permanecerem na lista tabu.



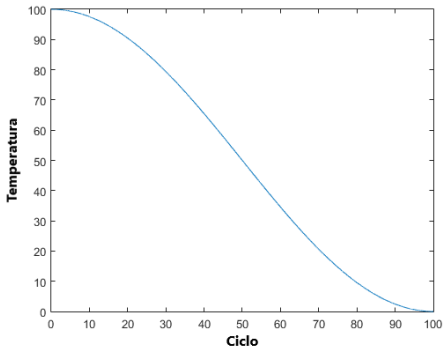
(a) Aditivo Linear



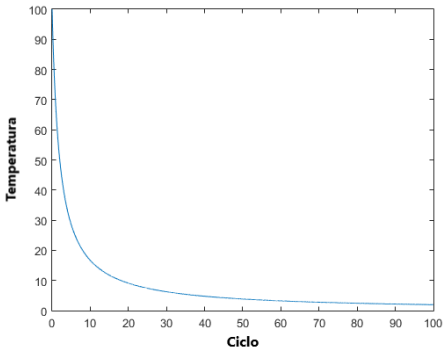
(b) Aditivo Quadrático



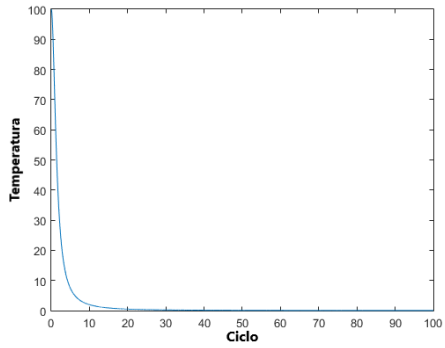
(c) Aditivo Exponencial



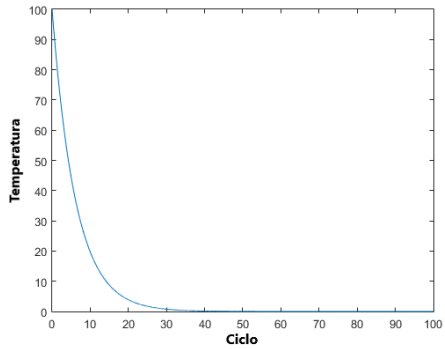
(d) Aditivo Trigonométrico



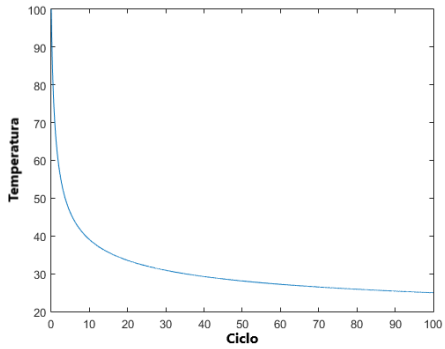
(e) Multiplicativo Linear



(f) Multiplicativo Quadrático



(g) Multiplicativo Exponencial



(h) Multiplicativo Logarítmico

Figura 4.1: Alguns esquemas de resfriamento presentes na literatura (THOMPSON; DOWSLAND, 1996; SUMAN; KUMAR, 2006; TAN, 2008)

Fonte: O Autor

O tamanho da lista controla a memória do processo de busca. Se pequeno, a busca tenderá a se concentrar em pequenas áreas do espaço de busca (intensificação); entretanto se ela for grande, a busca irá explorar regiões bem maiores, devido a lista inibir com que soluções sejam revisitadas e forçará a procura em novas regiões do espaço de busca. Uma alternativa proposta por Battiti (BATTITI; TECCHIOLLI, 1994) é a busca tabu reativa, onde o histórico de busca e o conhecimento adquirido durante o processo de busca são usados para adaptar automaticamente o tamanho da lista.

De fato, apesar das vantagens citadas anteriormente, a lista tabu também pode atrapalhar o processo de busca, proibindo movimentos de melhora da solução ou até mesmo levando a total estagnação da busca (SOUZA, 2005). Dessa forma, a inclusão de um critério de aspiração A, pode aprimorar bastante o processo de busca. Ele é um instrumento usado para passar pelas restrições impostas pela lista tabu, ou seja, retirar o status de tabu de uma certa solução s', em certas circunstâncias. Duas formas comumente utilizadas para modelar o critério de aspiração são a aspiração por objetivo e a *default*. Na aspiração por objetivo, um movimento tabu para uma solução s' em V é permitido se $f(s') < A(f(s))$. Já na aspiração por $default$, o movimento tabu mais antigo é permitido se não houver qualquer movimento possível fora da lista tabu.

Segundo (SOUZA, 2005), esse processo de busca é normalmente interrompido seguindo basicamente dois critérios de parada. O primeiro é o $BTmax$, número máximo de iterações sem que haja melhora no valor da melhor solução encontrada. Já o segundo, é interromper a busca quando o valor da melhor solução encontrada até então atinge um limite inferior conhecido ou se aproxima dele, evitando a continuação desnecessária da busca. Obviamente, não é sempre que se conhece o limite inferior de um problema, tornando o primeiro critério mais comum. O Algoritmo 4.2 mostra como se dá o processo de busca da TS.

4.1.3 GRASP

GRASP, acrônimo para *Greedy Randomized Adaptive Search Procedure*, é um método interativo para problemas de otimização combinatorial proposto por Feo e Resende (FEO; RESENDE, 1989; RESENDE; RIBEIRO, 2003). Cada iteração do GRASP consiste de dois passos: um passo chamado de construção, onde uma solução é produzida elemento a elemento, e um segundo passo, onde essa solução é usada como solução inicial de um procedimento de busca local. Após um certo número de iterações, o algoritmo termina e a melhor solução encontrada

Algoritmo 4.2: Busca Tabu

```
1: Inicie com solução inicial s
2: Melhor solução já encontrada s*
3: Inicialize a lista tabu vazia
4: Defina um valor para BTmax
5: Caso conheça o limite inferior, forneça f_min
6: Iter = 0
7: MelhorIter = 0
8: enquanto (Iter − MelhorIter) < BTmax e f(s) > f_min faça
9:    Iter = Iter + 1
10:   Gere um subconjunto V na vizinhança V(s)
11:   Recupere a melhor solução s' de V tal que não seja tabu ou f(s') < A(f(s))
12:   Atualize a lista tabu
13:   s = s'
14:   se f(s) < f(s*) então
15:      s* = s
16:      MelhorIter = Iter
17:   fim se
18: fim enquanto
19: retorne A melhor solução encontrada s*
```

é retornada como resultado. O pseudo-código de GRASP é brevemente descrito no Algoritmo 4.3 a seguir.

Na etapa de construção, a solução é construída iterativamente, ou seja, um novo elemento é incorporado em uma solução parcial a cada iteração, até que esteja completa. A cada iteração dessa fase, uma lista de elementos candidatos é criada com todos os elementos que podem ser incluídos na solução parcial que seja viável. A ordenação dessa lista é dada por uma função gulosa, que mede o benefício em escolher cada elemento. Assim, o elemento para ser adicionado é selecionado aleatoriamente entre os melhores candidatos. A lista contendo os melhores candidatos é chamada de lista de candidatos restrita. Essa seleção aleatória de um elemento dentro da lista representa a componente probabilística do GRASP.

O tamanho da lista de candidatos restrita pode ser baseado diretamente pela cardinalidade ou pela qualidade dos elementos. No primeiro caso, a lista é composta pelos p melhores candidatos, sendo p um parâmetro a ser determinado. Já no segundo, consiste em os candidatos possuírem um custo maior ou igual a $c_{min} + \alpha * (c_{max} - c_{min})$, onde $\alpha \in [0, 1]$. O parâmetro α controla o compromisso entre intensificação e diversificação. Um valor $\alpha = 0$ faz com que apenas soluções gulosas sejam geradas, já com $\alpha = 1$, são criadas de maneira totalmente aleatória.

O desempenho do GRASP é bastante sensível ao valor do parâmetro α, por isso procedi-

mentos mais sofisticados incluem um ajuste adaptativo para α como em (PRAIS et al., 1998). O método GRASP reativo realiza um auto-ajuste de α, onde o valor é periodicamente atualizado de acordo com a qualidade das soluções obtidas.

Algoritmo 4.3: GRASP

1: $f^* = \inf$
2: **enquanto** critério de parada não é atingido **faça**
3: $s = Construção(\alpha, s)$
4: $s = BuscaLocal(N, s)$
5: **se** $f(s) < f^*$ **então**
6: $s^* = s$
7: $f^* = f(s)$
8: **fim se**
9: **fim enquanto**
10: **retorne** A melhor solução encontrada s^*

4.1.4 Variable Neighborhood Search

O método de Busca em Vizinhança Variável (Variable Neighborhood Search, VNS) proposto por Pierre Hansen e Nenad Mladenovic (HANSEN; OSTERMEIER, 2001) é uma metaheurística de busca local que consiste na exploração de uma vizinhança dinamicamente variável para uma dada solução. Na inicialização do algoritmo, as estruturas de vizinhança podem ser escolhidas arbitrariamente, mas normalmente como uma sequência de vizinhanças $N_1, N_2, \ldots, N_{Nmax}$ com cardinalidade crescente. Segundo (BOUSSAÏD; LEPAGNOT; SIARRY, 2013), essas estruturas podem ser incluídas uma na outra, ou seja, $N_1 \in N_2 \in \ldots \in N_{Nmax}$, mas tal inclusão pode gerar uma busca ineficiente, já que um grande número de soluções acabariam sendo revisitadas. Assim, o VNS se torna eficiente se as estruturas de vizinhança usadas são complementares, ou seja, o ótimo local para a vizinhança N_i não é um ótimo local para a vizinhança N_j. Em sua versão original, o VNS faz uso do método de Descida em Vizinhança Variável (Variable Neighborhood Descent, VND) (HANSEN; OSTERMEIER, 2001) para realizar a busca local.

O algoritmo começa com uma solução inicial arbitrária e a cada iteração uma solução s' é escolhida aleatoriamente dentro da vizinhança $N_n(s)$ da solução atual s. Então, s' é usada como solução inicial de um método de busca local, produzindo s'' como solução ótima local. A busca local pode usar qualquer estrutura de vizinhança, não precisando se limitar à sequência definida

para o VNS. Se s'' for melhor que s, s'' substitui s como solução atual e o ciclo recomeça com $n = 1$. Caso contrário, a busca continua na próxima vizinhança $n = n + 1$. Este procedimento termina quando uma condição de parada for atingida como o número máximo de iterações, número máximo de iterações sem melhoria na solução atual, tempo máximo de busca, entre outros. O pseudocódigo do VNS é apresentado no Algoritmo 4.4.

Algoritmo 4.4: VNS

```
1: Selecione as estruturas de vizinhança (N_1, N_2, ..., N_n), onde n = 1, 2, ..., Nmax
2: Escolha uma solução inicial s
3: enquanto critério de parada não é atingido faça
4:    n = 1
5:    enquanto n < Nmax faça
6:       Escolha aleatoriamente uma solução s' em N_n(s)
7:       Faça uma busca local começando em s' e obtenha s'' como ótimo local
8:       se s'' é melhor que s então
9:          s = s''
10:         n = 1
11:      senão
12:         n = n + 1
13:      fim se
14:   fim enquanto
15: fim enquanto
16: retorne A melhor solução encontrada s*
```

4.1.5 Iterated Local Search

A definição da metaheurística *Iterated Local Search* (ILS) foi dada por Lourenço e colaboradores em (LOURENçO; MARTIN; STüTZLE, 2002), onde também mostram que outros autores propuseram técnicas que se enquadram como instâncias específicas do ILS, tais como *iterated descent* (BAUM, 1986), *large-step Markov chains* (MARTIN; OTTO; FELTEN, 1992), *iterated Lin-Kernighan* (JOHNSON, 1990), *chained local optimization* (MARTIN; OTTO, 1993) e também suas combinações (APPLEGATE; COOK; ROHE, 2003). Segundo (LOURENçO; MARTIN; STüTZLE, 2002), há duas características principais que fazem de um algoritmo uma instância da ILS: (1) deve existir apenas uma trajetória sendo seguida, excluindo assim, os algoritmos baseados em populações; (2) a busca por melhores soluções deve acontecer em espaços de busca reduzidos que são definidos por uma heurística "caixa-preta". Na

prática, essa heurística é implementada como uma busca local, mas também qualquer algoritmo de otimização específico de um problema poderia ser usado.

Ao invés de realizar buscas locais em soluções geradas aleatoriamente repetidamente, a ILS produz uma solução inicial para a próxima iteração através de uma perturbação do mínimo local encontrado na iteração atual. Esse procedimento é realizado na esperança que a perturbação gere uma solução que esteja localizada na bacia de atração de um mínimo local melhor. Esse mecanismo de perturbação é um ponto-chave da ILS, de um lado, se a perturbação for muito fraca, pode não ser suficiente para escapar da bacia de atração do mínimo local atual; por outro lado, se a perturbação for muito forte, pode fazer com que o algoritmo reinicie as buscas locais de maneira similar à aleatória.

O critério de aceitação é outra característica chave da ILS e define as condições que o novo mínimo local precisa atender para substituir a atual solução. Juntamente com o mecanismo de perturbação, permitem que o algoritmo controle o balanço entre intensificação e diversificação. Condição extrema de aceitação visando intensificação é aceitar apenas soluções melhores, já uma condição extrema de diversificação, aceita qualquer solução seja ela melhor ou não. O pseudocódigo da ILS é apresentado no Algoritmo 4.5.

Algoritmo 4.5: ILS

1: Escolha de forma aleatória uma solução s
2: Realize uma busca local a partir de s, gerando s^*
3: **enquanto** critério de parada não for atendido **faça**
4: Aplique uma perturbação sobre s^* para gerar uma solução p
5: Faça uma busca local começando em p e obtenha p^*
6: **se** critério de aceitação é satisfeito **então**
7: $s^* = p^*$
8: **fim se**
9: **fim enquanto**
10: **retorne** A melhor solução encontrada s^*

4.2 Algoritmos Genéticos

O Algoritmo Genético (AG) (GOLDBERG, 1989) pode ser definido como uma técnica de busca e otimização de problemas baseada nos princípios da genética e seleção natural. Cada solução em potencial para o problema a ser resolvido se encontra codificada em uma estrutura chamada de cromossomo ou indivíduo, e o conjunto de soluções é chamado de população.

Geralmente, a população inicial é gerada aleatoriamente (SIVANANDAM; DEEPA, 2008) e será modificada seguindo os processos biológicos da evolução através de gerações. A cada geração, novos indivíduos são gerados através de operadores genéticos e a população resultante será constituída por indivíduos considerados melhores do que aqueles presentes na geração passada.

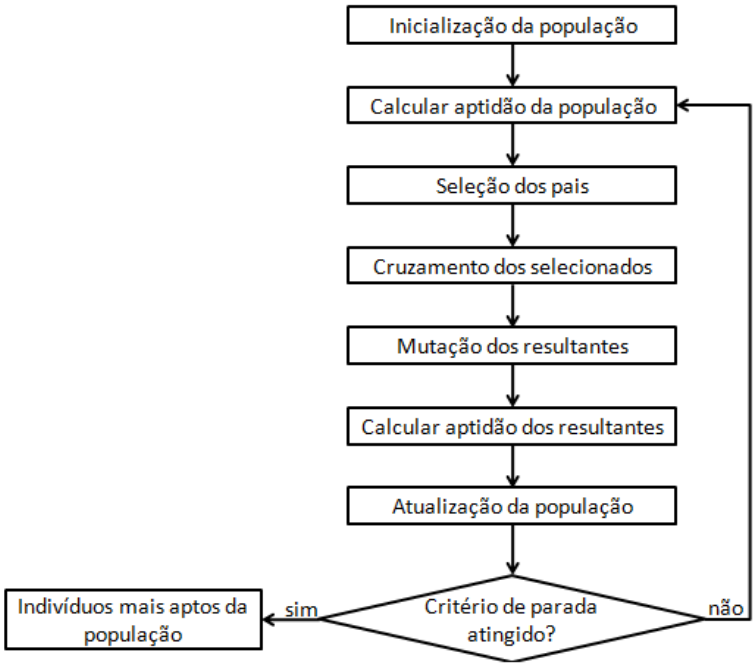
Os melhores indivíduos são aqueles que melhor se ajustam ao problema. A adequação de cada indivíduo é medida por uma função de aptidão, que indica quão boa é a solução para o problema em mãos. Portanto, os indivíduos mais aptos são mais prováveis de serem escolhidos (SIVANANDAM; DEEPA, 2008) e eles passam através das gerações até que alcançam a população final, de onde a melhor solução para o problema é encontrada. Dependendo da técnica de seleção utilizada, os melhores indivíduos podem ser movidos diretamente para a próxima geração ou selecionados para produzir descendentes pela aplicação de operadores genéticos tais como *crossover* e mutação. Tanto a função de aptidão quanto a forma de codificar as soluções ficam dependendo do domínio do problema.

O fluxograma do funcionamento de um AG canônico é apresentado na Figura 4.2. O processo de busca se inicia com uma população inicial gerada de alguma maneira, por exemplo, de forma aleatória. Os valores de aptidão de cada indivíduo da população é calculado. O método de seleção extrai pares de cromossomos da população atual para gerar novos indivíduos. O operador de *crossover* realiza a troca de segmentos entre dois cromossomos com certa probabilidade. Então, o operador de mutação é aplicado nos novos indivíduos para provocar mudanças em diferentes partes do cromossomo com certa probabilidade. Os indivíduos mais aptos são mantidos na próxima geração. O processo continua até que uma condição de parada seja satisfeita. Por fim, a melhor solução obtida através do processo é recuperada.

4.2.1 Representação do Cromossomo

Segundo (DAWKINS, 1996; KUNCHEVA, 2004), a escolha da representação do cromossomo é uma etapa muito importante para o desenvolvimento de um AG, uma vez que ela está ligada diretamente ao desempenho do algoritmo. Um indivíduo consiste de um conjunto de genes, onde a informação se encontra codificada (genótipo) e sua avaliação é baseada em seu fenótipo (conjunto de características observáveis no objeto resultante do processo de decodificação dos genes). O conjunto de todas as configurações que o cromossomo pode assumir

Figura 4.2: Fluxograma de um Algoritmo Genético canônico.



Fonte: O Autor

forma o seu espaço de busca. Se o cromossomo representa *n* parâmetros de uma função, então o espaço de busca é *n*-dimensional. A Tabela 4.1 mostra exemplos de representações de indivíduos.

Tabela 4.1: Exemplos de representações de indivíduos em diferentes problemas

Genótipo	Fenótipo	Problema
101101	45	Otimização numérica
ABCDE	Inicie pela cidade A, passando em seguida pelas cidades B, C, D e termine em E	Caixeiro viajante
$C_1R_3C_2R_4$	Se condição 1 (C_1) execute a regra 3 (R_3), se condição 2 (C_2) execute a regra 4 (R_4)	Regras de aprendizagem para agentes

4.2.2 Seleção

A seleção é a responsável pela perpetuação das boas características na população. Segundo (DAWKINS, 1996), a seleção é um processo dirigido e cumulativo. Ao contrário da mutação, que pode ocorrer de forma aleatória, a seleção opera de forma determinística ou próxima disso. O que significa que enquanto as mutações ocorrem ao acaso, um indivíduo, através de suas características fenotípicas, só conseguirá sobreviver e reproduzir em seu ambiente se e somente se for apto a responder de forma adequada aos fenômenos de seu meio. Ela é cumulativa devido às boas características serem mantidas de uma geração para a outra no processo de seleção. Essas duas propriedades combinadas garantem a possibilidade de surgimento de organismos complexos como as formas de vida que hoje conhecemos.

Segundo (DAWKINS, 1996), existem várias formas para efetuar a seleção, dentre as quais se destacam:

- **Seleção por *ranking***: os individuos da população são ordenados pelo seu grau de aptidão e então a probabilidade de escolha de um individuo é atribuída conforme a posição que ocupa.

- **Seleção por roleta**: cada individuo da população é representado proporcionalmente ao seu grau de aptidão. Logo, indivíduos com valores altos de aptidão ocupam grandes porções da roleta (maior probabilidade de ser um pai da próxima geração), enquanto aqueles de menor aptidão ocupam uma porção relativamente menor da roleta (menor probabilidade de ser um pai da próxima geração). Finalmente, a roleta é girada certo número de vezes, dependendo do tamanho da população, e aqueles sorteados na roleta serão indivíduos na próxima geração.

- **Seleção por torneio**: usa sucessivas disputas para fazer a seleção. Esse método estabelece k disputas para selecionar k indivíduos, cada disputa envolvendo n indivíduos selecionados aleatoriamente. O indivíduo com o maior valor de aptidão é selecionado na disputa, e deve permanecer na população para a próxima geração.

- **Seleção uniforme**: todos os indivíduos da população têm a mesma probabilidade de serem selecionados. Claramente, esta forma de selecionar indivíduos possui uma probabilidade muito remota de causar uma melhora da população.

4.2.3 Cruzamento

O cruzamento (*crossover*) é um intercâmbio de informações entre cromossomos pais, criando novos cromossomos diferentes daqueles existentes na geração atual. Para que isto aconteça, o operador precisa escolher pares de reprodutores, havendo assim a troca de partes de seus materiais genéticos. Segundo (LUCAS, 2002), alguns dos principais métodos de escolha desses pares de reprodutores são:

- **Escolha aleatória**: os pares de individuos são escolhidos ao acaso.
- ***Inbreeding***: parentes são combinados, ou seja, individuos que possuem um ancestral em comum são escolhidos.
- ***Line breeding***: onde o indivíduo mais apto é o único que pode cruzar com os outros.
- ***Self-fertilization***: o indivíduo é combinado consigo mesmo, ou seja, há uma espécie de "clonagem"desse individuo.
- ***Positive assortive mating***: indivíduos semelhantes são combinados.
- ***Negative assortive mating***: indivíduos diferentes são combinados.

Para realizar a operação do cruzamento, são definidos pontos na estrutura do cromossomo para a troca do material genético. Baseada nestes pontos, uma seleção por máscara é criada para determinar para qual filho irá cada gene dos pais ao se combinarem. Seu funcionamento pode ser visto segundo o seguinte algoritmo:

Algoritmo 4.6: Algoritmo da seleção por máscara

1: **se** critério de aceitação é satisfeito **então**
2: $filho_1(i) = pai_1(i)$;
3: $filho_2(i) = pai_2(i)$;
4: **senão**
5: $filho_1(i) = pai_2(i)$;
6: $filho_2(i) = pai_1(i)$;
7: **fim se**

De (GOLDBERG, 1989), pode-se ver que os mais conhecidos operadores de cruzamento para genomas de tamanho fixo são:

- **Cruzamento de um ponto**: dados dois genomas p_1 e p_2 de comprimento l, sorteia-se um número k qualquer tal que $0 < k < l$, o primeiro filho f_1 receberá todos os genes p_1 de 1 até k e todos os genes p_2 de $k + 1$ até l, e o segundo filho f_2 o inverso. Portanto, a máscara de cruzamento seria uma sequência de $0s$ (zeros) de 1 até k, sucedida por uma sequência de $1s$ (uns) de $k + 1$ até l. Dentre os operadores de cruzamento tradicionais, o de um ponto é o que normalmente apresenta o pior desempenho. A Figura 4.3 ilustra esse operador.

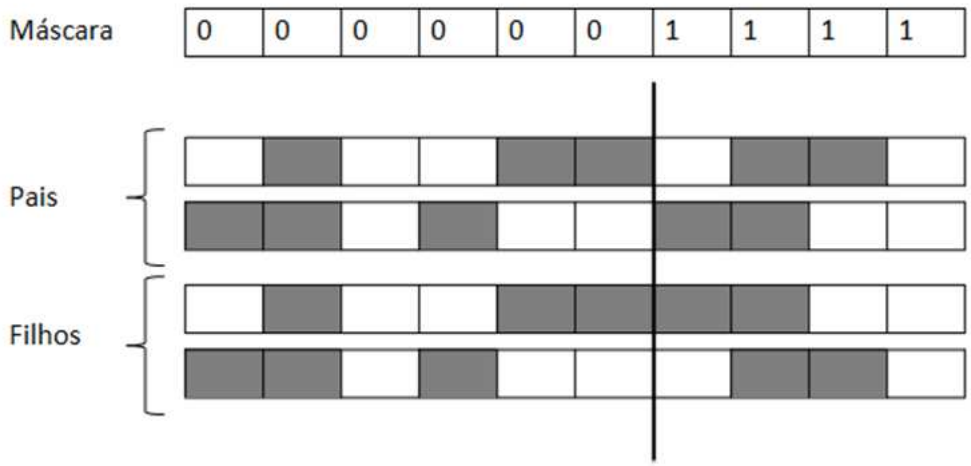


Figura 4.3: Operador de cruzamento de um ponto.
Fonte: O Autor

- **Cruzamento multiponto**: o cruzamento multiponto é uma generalização do operador de um ponto. Um número fixo n de pontos de corte é sorteado. Um operador com n pontos de cruzamento apresentaria uma máscara de cruzamento com n trocas em sua sequência de zeros e uns. A Figura 4.4 apresenta esse operador.

- **Cruzamento segmentado**: o cruzamento segmentado funciona de forma parecida ao multiponto. Segundo (ESHELMAN; CARUANA; SCHAFFER, 1989), neste operador não existem pontos de corte, cada gene do cromossomo pai é testado e, caso seja selecionado segundo uma dada probabilidade, é trocado pelo respectivo gene do outro pai.

- **Cruzamento uniforme**: para cada gene dos filhos, o operador de cruzamento uniforme escolhe de qual dos pais este se origina. A máscara de cruzamento, portanto, é uma sequência qualquer de zeros e uns. A Figura 4.5 mostra esse operador.

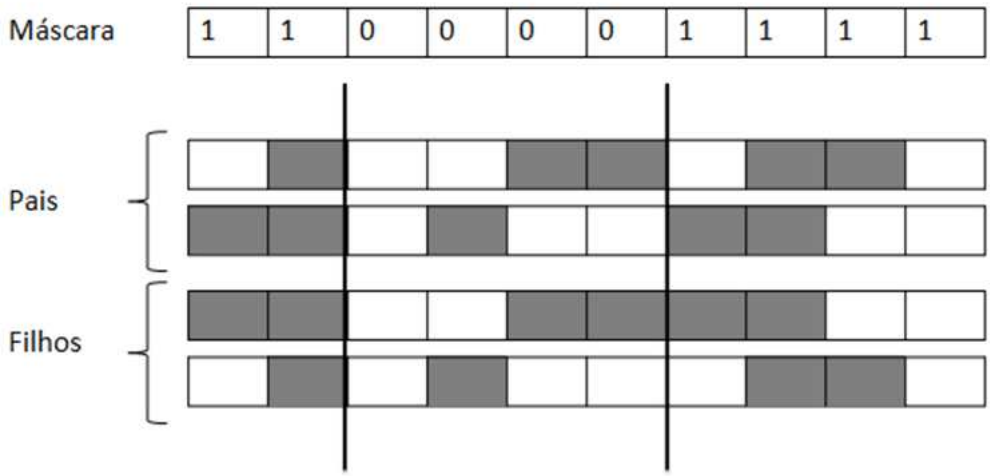


Figura 4.4: Operador de cruzamento multiponto com dois pontos de corte.
Fonte: O Autor

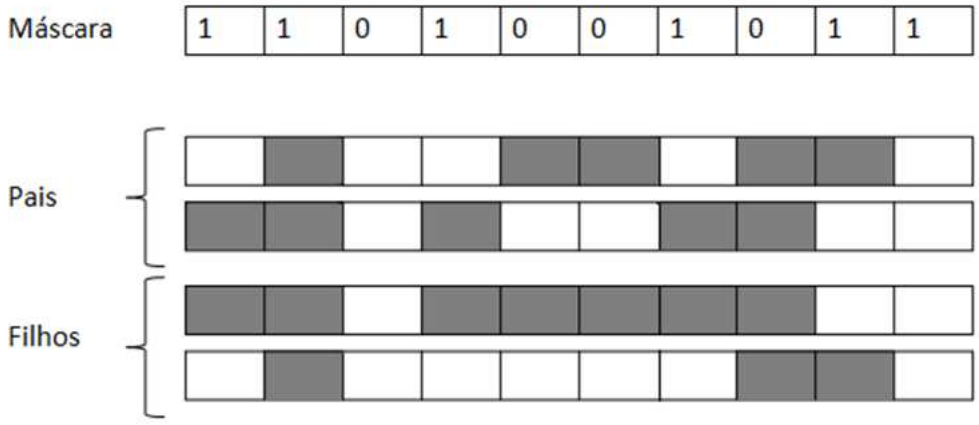


Figura 4.5: Operador de cruzamento uniforme.
Fonte: O Autor

4.2.4 Mutação

A mutação opera sobre os indivíduos resultantes do processo de cruzamento e com certa probabilidade pré-estabelecida efetua algum tipo de alteração em suas estruturas. A mutação é fator fundamental para garantir a biodiversidade, assegurando assim que o espaço de busca possivelmente será explorado em uma parte significativa de sua extensão. Este operador possui também um papel fundamental para evitar uma convergência prematura, que ocorre quando a população se estabiliza com uma média de adaptação pouco adequada por causa da pressão evolutiva e baixa diversidade. Alguns exemplos de operadores de mutação são:

- **Mutação por troca (*swap mutation*)**: são sorteados n pares de genes, e os elementos do par trocam de valor entre si.
- **Mutação *creep***: um valor aleatório é somado ou subtraído do valor do gene.
- **Mutação *flip***: cada gene que sofrerá mutação recebe um valor sorteado do alfabeto válido.

4.2.5 Atualização

Após a aplicação dos operadores de cruzamento e mutação, a população antiga é substituída por uma nova população segundo a política adotava pelo AG. Na formulação clássica de um algoritmo genético, a população mantém um tamanho fixo e os novos indivíduos gerados pela aplicação dos operadores genéticos substituem inteiramente a população anterior. Entretanto, existem várias alternativas mais sofisticadas a essa abordagem como: o tamanho da população ser variado, critério de inserção ser variado (como, por exemplo, os novos indivíduos serão inseridos somente se possuírem maior valor de aptidão que os cromossomos a serem substituídos), ou escolher um conjunto dos n melhores indivíduos para se manter na população.

4.2.6 Características

Pela maneira como os AGs operam, pode-se destacar as seguintes características:

- **Busca codificada**: Segundo (LUCAS, 2002), os AGs não trabalham sobre o domínio do problema, mas sobre representações de seus elementos. Tal fato impõe ao problema que o

conjunto de soluções viáveis para ele seja de alguma forma codificada em uma população de indivíduos.

- **Generalidade**: Uma vez que a representação e a avaliação das possíveis soluções são as únicas partes que requerem conhecimento dependente do domínio do problema, basta apenas a alteração destas para os AGs atenderem outros casos.

- **Paralelismo explícito**: o alto grau de paralelismo intrínseco aos AGs pode ser facilmente visto considerando que cada indivíduo da população existe como um ente isolado e é avaliado independentemente.

- **Busca estocástica**: A busca não se dá de forma completamente aleatória, pois as probabilidades de aplicação dos operadores genéticos fazem com que estes operem de forma previsível estatisticamente, apesar de não permitirem que o comportamento do sistema seja determinado com exatidão absoluta.

- **Busca cega**: um AG tradicional opera sem conhecer o significado das estruturas que manipula e, assim, não conhece qual seria a melhor maneira de trabalhar com elas. Esta característica de não se valer de conhecimento específico ao domínio do problema, traz de um lado generalidade, mas por outro tende a possuir menor eficiência.

- **Eficiência mediana**: por constituir um método de busca cega, um AG tende a apresentar um desempenho inferior se comparado com alguns tipos de busca heurística orientadas ao problema. Para solucionar isso, a tática mais utilizada é a hibridização como mostrada em (WEARE; BURKE; ELLIMAN,), onde heurísticas de outras técnicas são incorporadas.

- **Paralelismo implícito**: ao se fazer uma busca por populações, a evolução de um AG tende a favorecer indivíduos que compartilhem certas características, sendo assim capaz de avaliar implicitamente determinadas combinações ou esquemas como sendo mais ou menos desejáveis, assim, efetuando uma busca por hiperplanos que é de natureza paralela (GOLDBERG, 1989).

- **Facilidade no uso de restrições**: os AGs facilitam a codificação de problemas com diversos tipos de restrições, mesmo que elas apresentem diferentes graus de importância (BARBOSA, 1996). Neste caso, se dois indivíduos violam as restrições, o mais apto é aquele que viola as restrições mais flexíveis (*soft constraints*) enquanto que o outro viola as mais graves (*hard constraints*).

4.3 Metaheurísticas Híbridas

Um grande número de artigos foram publicados propondo algoritmos cujos conceitos não são de uma única metaheurística, pelo contrário, eles combinam diferentes estratégias, algumas até oriundas de fora do campo das metaheurísticas (KRASNOGOR; SMITH, 2005; RAIDL, 2006). Esses métodos são mais conhecidos como metaheurísticas híbridas.

A motivação por trás dessas hibridizações é alcançar um melhor desempenho em resolver problemas de otimização complexos através da combinação dos pontos fortes de cada algoritmo e minimizando suas fraquezas. A efetividade de cada hibridização depende de quão sinérgica é a combinação dos conceitos algorítmicos, o que não é uma tarefa fácil. Vários trabalhos envolvendo metaheurísticas híbridas tem sido reportados em eventos dedicados à esse campo de pesquisa, como o evento *Workshops on Hybrid Metaheuristics* que é realizado desde 2004, mostrando a importância e popularidade do tópico.

4.3.1 Classificação das Metaheurísticas Híbridas

Com base nas taxonomias publicadas em (TALBI, 2002; BLUM; ROLI; ALBA, 2005; COTTA; TALBI; ALBA, 2005; PUCHINGER; RAIDL, 2005; EL-ABD; KAMEL, 2005), nesta subseção as várias classes e propriedades, pelas quais as metaheurísticas híbridas são classificadas, são ilustradas.

4.3.1.1 Tipos de algoritmos

Segundo (TALBI, 2002; BLUM; ROLI; ALBA, 2005; COTTA; TALBI; ALBA, 2005; PUCHINGER; RAIDL, 2005; EL-ABD; KAMEL, 2005), pode-se começar diferenciando o que de fato é hibridizado, ou seja, os tipos de algoritmos envolvidos. Nesta categoria, eles podem ser divididos da seguinte maneira:

(a) Metaheurísticas com diferentes estratégias

Uma das formas mais populares de hibridização consiste no uso de metaheurísticas de baseadas em população trabalhando junto com métodos de trajetória. De fato, a maioria das aplicações bem-sucedidas de Computação Evolutiva fazem uso de procedimentos de busca local e a eficiência de muitos algoritmos híbridos dependem de fato dessa sinergia.

O poder dos métodos baseados em população é a capacidade de recombinar as soluções

para obter outras. Isso permite que o processo de busca realize uma amostragem guiada do espaço de busca, geralmente resultando em uma exploração "granulada". Portanto, essas técnicas podem encontrar de forma efetiva regiões promissoras do espaço de busca.

Já a força dos métodos de trajetória pode ser encontrada na maneira com que a exploração de uma região promissora do espaço de busca é feita. Como nesses métodos a busca local é um componente chave, uma região promissora no espaço de busca é explorada de uma maneira mais estruturada do que nos métodos baseados em população. Portanto, a chance de estar próximo à boas soluções mas perdê-las não é tão alta quanto nas abordagens baseadas em população.

Em resumo, os métodos baseados em população são melhores em identificar áreas promissoras do espaço de busca nos quais os métodos de trajetória podem rapidamente atingir um bom ótimo local. Portanto, metaheurísticas híbridas que combinem de forma eficiente as forças desses dois grupos serão bem-sucedidas.

(b) Metaheurísticas com outras técnicas mais gerais oriundas de outros campos como Pesquisa Operacional (PO) e Inteligência Artificial (IA).

Uma das direções de pesquisa mais proeminentes é a integração de metaheurísticas com métodos mais clássicos de Pesquisa Operacional e Inteligência Artificial, como Programação com Restrições (PR), técnicas de busca em árvores entre outras.

Técnicas de PR podem ser utilizadas para reduzir o espaço de busca ou a vizinhança para ser explorada por um método de busca local. As restrições encapsulam partes bem definidas do problema em sub-problemas, tornando possível a construção de algoritmos especializados em solucionar sub-problemas que ocorrem frequentemente. Cada restrição é associada à um "algoritmo de filtro"que remove valores do domínio de um variável que não contribuem para soluções válidas. As metaheurísticas, especialmente aqueles métodos de trajetória, podem usar PR para explorar de forma eficiente a vizinhança da solução atual, ao invés de enumerar os vizinhos ou amostrá-los.

Uma outra combinação possível consiste em introduzir estratégias ou conceitos de uma classe de algoritmos em outra. É o caso da combinação entre Busca Tabu e o algoritmo de busca em árvore. A lista tabu e o critério de aspiração podem ser usados para gerenciar a lista de nós folha ainda não explorados na busca em árvore.

Essas formas de integração que foram mencionadas pertencem ao grupo de combinações integrativas, onde acontece uma relação mestre-escravo onde um algoritmo comanda o processo de busca enquanto que o outro fica subordinado ao primeiro. Um outro tipo

de hibridização é a estratégia cooperativa ou colaborativa, baseada na troca de estados, modelos, sub-problemas, soluções ou características do espaço de busca. Normalmente, os algoritmos de busca colaborativa consistem na execução paralela dos métodos de busca com algum nível de comunicação entre eles.

4.3.1.2 Nível de Hibridização

Além da diferenciação por tipo, algumas taxonomias, como as apresentadas em (COTTAPORRAS, 1998; TALBI, 2002), faziam primeiro a distinção pelo nível ou força da combinação de algoritmos diferentes. Nas combinações de alto nível, a princípio, as identidades dos algoritmos envolvidos são mantidas e há uma forma de cooperação bem definida entre eles, mas não há um relacionamento direto forte entre os mecanismos internos dos algoritmos. Já os algoritmos que participam de combinações de baixo nível dependem fortemente um do outro, já que ocorrem trocas de componentes ou funções individuais.

4.3.1.3 Ordem de Execução

Uma outra propriedade pela qual pode-se distinguir métodos híbridos é a ordem da execução dos componentes. Basicamente, podemos dividir os modos de ordem de execução em duas categorias: (a) por lote ou sequencial e (b) paralelo.

No modo sequencial, um algoritmo é executado apenas depois do outro terminar, fazendo com que a informação seja passada apenas em uma direção. Métodos que possuem algum tipo de pré-processamento inteligente ou pós-processamento dos resultados obtidos por outro algoritmo se enquadram nessa categoria de lote. Outro exemplo são os problemas multiníveis que são resolvidos considerando um nível depois do outro por algoritmos de otimização dedicados.

Ao contrário do modo sequencial, os modos intercalados e paralelos permitem que algoritmos possam interagir de formas mais sofisticadas. As metaheurísticas paralelas são atualmente um campo de pesquisa bastante importante, como vemos em (ALBA, 2005). Seguindo as definições comuns de aparecerem na literatura (TALBI, 2002; BLUM; ROLI; ALBA, 2005; COTTA; TALBI; ALBA, 2005; PUCHINGER; RAIDL, 2005; EL-ABD; KAMEL, 2005), podemos discrimina-las de acordo com as seguintes características:

- Arquitetura - Single Instruction, Multiple Data streams (SIMD) *vs* Multiple Instructions, Multiple Data streams (MIMD): Em máquinas SIMD, os processadores ficam restritos a executar o mesmo programa e são bastante eficientes em algoritmos paralelos síncronos

que não exijam tanto poder computacional e possuam uma certa regularidade na transferência de informações. Já máquinas com arquiteturas MIMD são mais indicadas nos casos quando os trabalhos se tornam irregulares, momento em que a arquitetura SIMD se torna menos eficiente. Os processadores podem executar outros tipos de instrução em dados diferentes.

- Uso do espaço de busca - Global *vs* Parcial: Na forma global, todos os algoritmos fazem buscas em todo o espaço de busca, ou seja, todos eles tentam resolver o problema de otimização de forma global. Já no modo parcial, cada algoritmo faz busca em uma porção diferente do espaço de busca e pode prover uma solução local.

- Tipo de algoritmo - Homogêneo *vs* Heterogêneo: Envolvem o uso de diferentes instâncias do mesmo algoritmo, no caso homogêneo, ou de algoritmos diferentes, no caso heterogêneo.

- Memória - Compartilhada *vs* Distribuída: As estratégias de memória compartilhada são mais simples, já aquelas com estratégia distribuída são mais flexíveis e proporcionam tolerância à falhas.

- Distribuição de trabalhos - Estática *vs* Dinâmica: Na categoria estática, as demandas são geradas e distribuídas seguindo um "agendamento"previamente estabelecido. A alocação de processadores para determinadas tarefas, por exemplo, permanece inalterada durante a execução do processo de busca. Já no modo dinâmico, a performance é aprimorada devido ao balanceamento da demanda de carga de trabalho.

- Sincronia dos trabalhos - Sincronismo *vs* Assincronismo: O primeiro caso compreende os modelos onde um deles aguarda os outros em um determinado ponto de sincronização. Já os modelos paralelos assíncronos trabalham com a informação que foi disponibilizada em algum tempo pelos demais.

4.3.1.4 Estratégia de Controle

Por fim, nós ainda podemos distinguir as metaheurísticas híbridas pela maneira como se dá o papel de cada algoritmo envolvido ou a estratégia de controle adotada. Segundo (TALBI, 2002; BLUM; ROLI; ALBA, 2005; COTTA; TALBI; ALBA, 2005; PUCHINGER; RAIDL, 2005; EL-ABD; KAMEL, 2005), existem basicamente as formas de combinação integrativa e colaborativa, também conhecida como cooperativa.

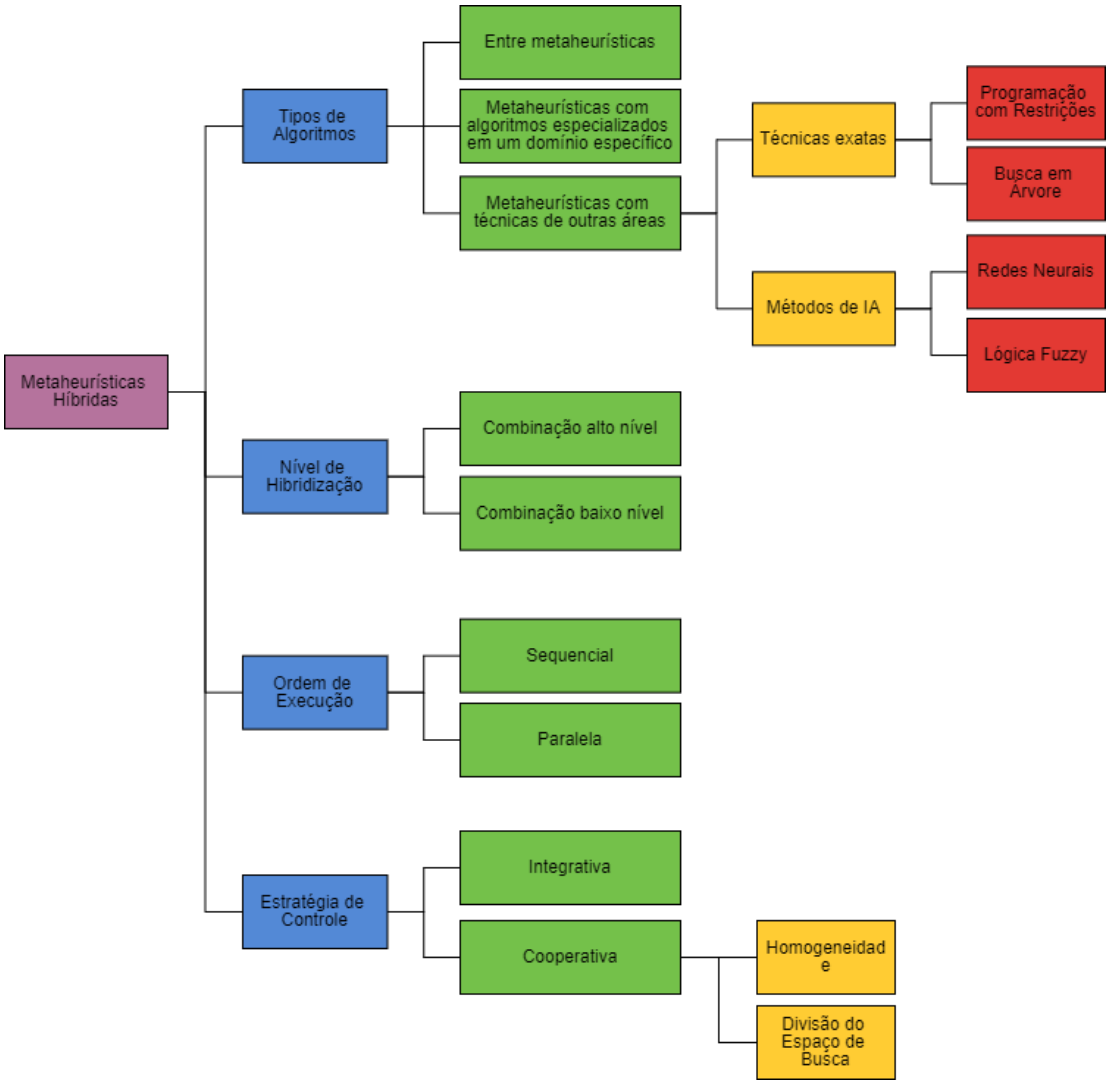
Nas abordagens integrativas, um dos algoritmos é considerado subordinado a outro ou é embutido em outro, sendo essa a abordagem mais amplamente utilizada. Alguns exemplos de abordagens integrativas são:

- Algoritmos meméticos são os exemplos mais claros desse tipo de abordagem, onde métodos de busca local são incorporados por Algoritmos Evolutivos (AE).
- Uso de técnicas exatas como Programação Dinâmica para encontrar melhores soluções em vizinhanças grandes dentro de métodos de busca local, como o *Very Large Scale Neighborhood Search* (VLSN) (AHUJA et al., 2002).
- Algoritmos baseados em população como AE, possuem um operador de recombinação que faz a combinação de duas ou mais soluções para gerar uma nova. Esse operador pode ser substituído por outros algoritmos mais poderosos como o Path Relinking (GLOVER et al., 2011), métodos exatos ou programação inteira para encontrar a melhor combinação entre as soluções.

Já nas abordagens cooperativas, os algoritmos não fazem parte um dos outros e realizam trocas de informação. Os algoritmos envolvidos podem ser instâncias da mesma metaheurística (abordagem homogênea) ou de diferentes algoritmos (abordagem heterogênea). Em relação ao espaço de busca, os métodos envolvidos podem explorar o mesmo espaço, mas utilizando diferentes soluções iniciais, parâmetros internos, etc., a fim de obter diferentes soluções, ou ainda, cada algoritmo trabalhar em uma porção do espaço de busca diferente.

A Figura 4.6 a seguir, ilustra a classificação das metaheurísticas híbridas como um todo, explicitando as categorias mencionadas anteriormente.

Figura 4.6: Classificação das metaheurísticas híbridas.

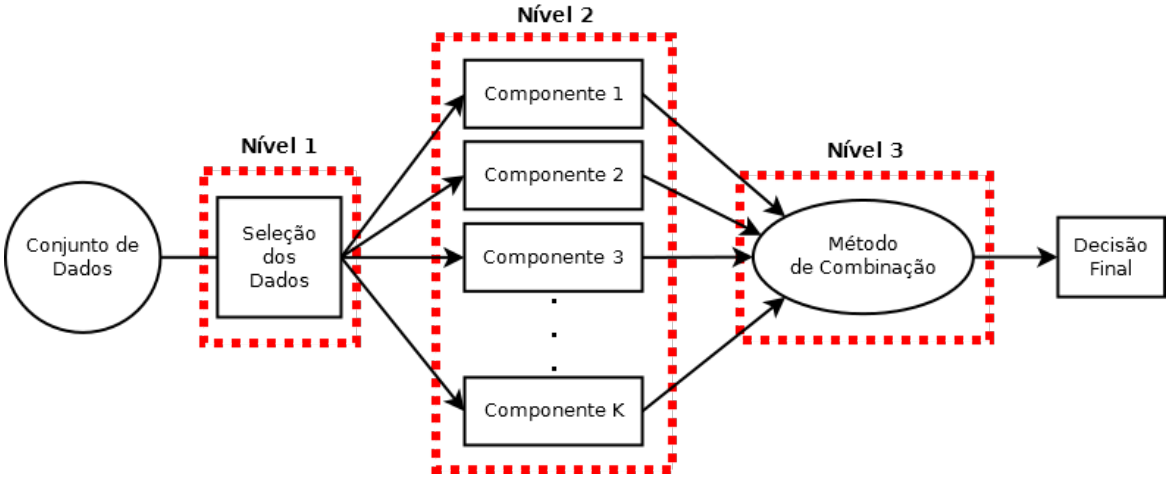


Fonte: O Autor

5 MÉTODO PROPOSTO

Como visto nos trabalhos relacionados, várias abordagens foram propostas ao longo do últimos anos, com o objetivo de otimizar certas partes ou níveis de *ensembles*. Essas divisões podem ser melhor vistas em uma típica arquitetura de ensemble, mostrada na Figura 5.1.

Figura 5.1: Visualização dos níveis de um comitê.



Fonte: O Autor

Nesta estrutura, pode-se claramente verificar a presença de três níveis, a seleção dos dados fornecidos para cada classificador, o conjunto de classificadores que compõe o *ensemble* e o método pelo qual as respostas dos componentes são combinadas.

As abordagens propostas até então focaram na otimização de um desses níveis ou combinação de dois deles, através do uso de algoritmos de Computação Evolutiva, onde a grande maioria utilizou Algoritmos Genéticos (AG) ou Programação Genética (PG).

A proposta desta tese é dar um passo adiante e considerar a otimização de todos os níveis do *ensemble* simultaneamente. Para que possamos introduzir a abordagem proposta, precisaremos revisitar um pouco da teoria de *Ensembles*.

Há uma série de cuidados que devemos ter ao criar um *ensemble* de maneira que este seja efetivo. A escolha correta de um conjunto de classificadores e de um método de combinação apropriado são os principais desafios na construção de *ensembles*. Além disso, um comitê efetivo deve consistir de um conjunto de classificadores com alta acurácia e erros descorrelacionados, ou seja, que eles também apresentem certo grau de diversidade. Essa diversidade pode ser gerada de quatro maneiras diferentes (KUNCHEVA, 2004; KUNCHEVA; WHITAKER, 2009): 1) Usando diferentes conjuntos de treinamento; 2) Escolhendo diferentes atributos de

entrada; 3) Variando as configurações dos parâmetros dos classificadores; 4) Ensemble composto por diferentes algoritmos de aprendizado.

Portanto, qualquer abordagem que se proponha a desenvolver um comitê efetivo precisa usar algum meio para gerar diversidade entre os componentes, escolher o conjunto de classificadores e o método de combinação. Nossa proposta é abordar todos esses pontos de uma forma unificada pelo uso de Algoritmos Genéticos, a técnica de Computação Evolutiva mais conhecida e empregada na solução de problemas de otimização, e que também é bastante robusta e trabalha muito bem em espaços de buscas grandes. O AG é bastante genérico e há vários aspectos que podem ser implementados diferentemente de acordo com o problema: representação da solução (cromossomo), estratégia de seleção, tipo dos operadores de recombinação e mutação, etc. Na próxima seção, nós apresentamos a modelagem do problema de otimização em detalhes.

5.1 Modelagem do Problema

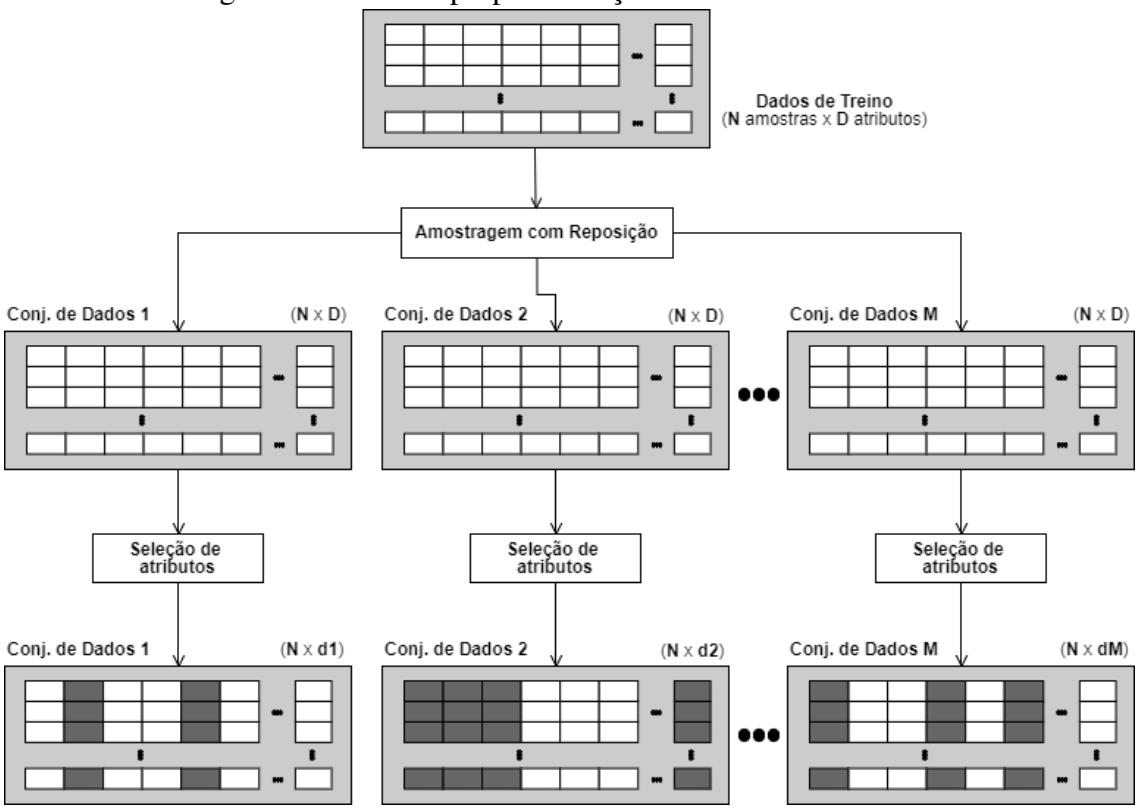
A partir de cada nível do ensemble, nós iremos considerar determinadas informações que estam diretamente ligadas às principais questões na construção dos mesmos.

5.1.1 Primeiro Nível

O primeiro nível considerado é o espaço de entrada do comitê, responsável por apresentar os dados do problema aos estimadores. Neste ponto, segundo (KUNCHEVA, 2004), há duas questões principais para serem atendidas de forma a criarmos um comitê efetivo: (1) prover conjuntos de dados diferentes de forma a possuirmos classificadores redundantes, ou seja, dois ou mais classificadores sendo treinados sobre os mesmos conjuntos de dados; (2) fazer com que os classificadores sejam treinados usando conjuntos de atributos dos dados diferentes. Dessa forma, nós garantimos um certo nível de diversidade entre os componentes do ensemble.

Assim, primeiramente nós empregamos a mesma ideia presente no algoritmo Bagging (BREIMAN, 1996) e geramos diferentes conjuntos de dados amostrados aleatoriamente. Em seguida, realizamos uma seleção de atributos em cada um dos conjuntos de dados gerados com a ajuda do AG, de forma que os classificadores operem em diferentes dimensões de um problema D-dimensional. Esse processo é ilustrado na Figura 5.2.

Figura 5.2: Método proposto - Ações realizadas no nível 1.



Fonte: O Autor

5.1.2 Segundo Nível

O segundo nível é a escolha do conjunto de classificadores. Neste ponto é definido o tamanho do ensemble, ou seja, a quantidade de componentes que o constitui, o(s) tipo(s) de classificador e a configuração dos parâmetros internos. O objetivo é selecionar o menor *ensemble* diverso (BHOWAN et al., 2014), reduzindo o risco de haver classificadores que não contribuem positivamente para a acurácia do comitê, inclusos no mesmo. O classificador base usado é o LS-SVM com *kernel* de Função de Base Radial ou *Radial Basis Function* (RBF), um dos tipos de *kernel* mais populares.

Em (VALENTINI; DIPARTIMENTO; DIETTERICH, 2004; LI; WANG; SUNG, 2005), nós vemos que outro ponto importante é a escolha do parâmetro **C** presente na formulação das LS-SVMs (Equação 2.25) e do(s) parâmetro(s) da função de *kernel*, no nosso caso é a largura da Gaussiana σ. A análise de performance realizada por Valentini et al. (VALENTINI; DIPARTIMENTO; DIETTERICH, 2004) mostra que uma sintonização apropriada desses parâmetros evita o *overfitting*. Além disso, diferentes configurações de parâmetros contribuem para gerar ainda mais diversidade.

Assim, codificando cada par dos parâmetros σ e C dentro de um vetor $\theta = (\sigma, C)$,

considere que temos a seguinte função de decisão parametrizada por α (multiplicadores de Lagrange), b (bias) e θ para cada LS-SVM que compõe o ensemble:

$$f_{\alpha,b,\theta}(x) = sign(\sum_{i=1}^{N} \alpha_i y_i K(\mathbf{x}, \mathbf{x}_i) + b) \tag{5.1}$$

Tal que:

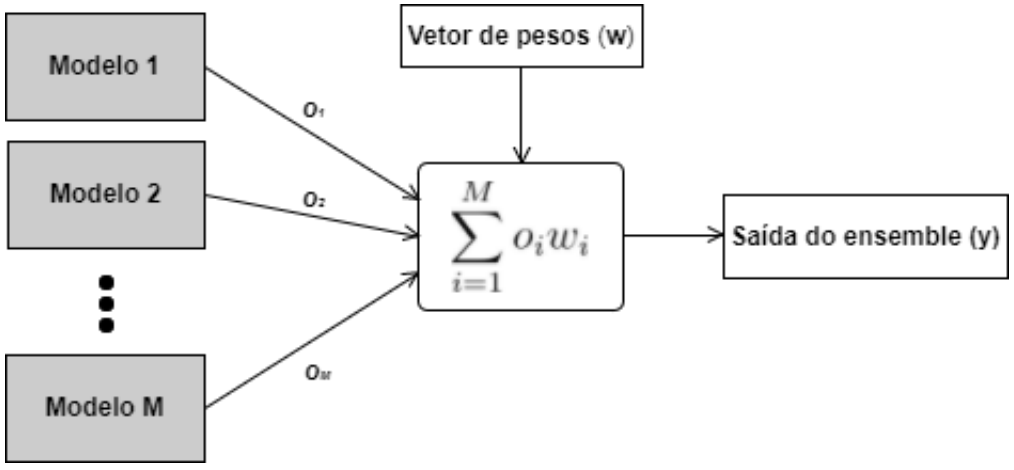
1. $K(\mathbf{x}, \mathbf{x}_i) = \exp\left(-\frac{1}{2\sigma^2}\|\mathbf{x} - \mathbf{x}_i\|^2\right)$
2. $0 \leq \alpha_i \leq C$ para $i = 1, 2, ..., N$

Os valores encontrados pelo AG para cada $\theta_1, \ldots, \theta_M$ e, posteriormente, com os valores de α encontrados após a solução do sistema linear (Equação 2.30) devem maximizar o valor de Q (Equação 2.21).

5.1.3 Terceiro Nível

O último nível é o método de combinação das respostas dos classificadores. Neste trabalho, a resposta final do *ensemble* será obtida através de uma simples combinação linear dos valores de decisão de cada classificador com um vetor de pesos **w**, que medirá a importância de cada LS-SVM na classificação final, de forma que a soma desses pesos seja igual a 1 (100%).

Figura 5.3: Método proposto - Ações realizadas no nível 3.



Fonte: O Autor

5.1.4 Estrutura do Cromossomo

Uma vez apresentados os níveis do *ensemble* considerados por nossa abordagem, agora precisamos definir como codificar toda essa informação (seleção de atributos, tamanho do comitê, parâmetros dos classificadores e o vetor de pesos do método de combinação) para representar um indivíduo da população do AG.

A representação de cada indivíduo da população do AG é um vetor com tamanho igual a $M * (n + 1) + 3 * M = M * (n + 4)$, onde M é o número de classificadores e n é o número de atributos do problema. O cromossomo tem duas partes, uma delas composta por $M * (n + 1)$ números binários, enquanto que a outra é composta por $3 * M$ números reais. Na parte binária do cromossomo, temos por classificador, um bit por classificador para dizer se ele está ou não compondo o ensemble, e mais n bits para representar quais atributos do problema estão sendo levados em consideração para aquele classificador. Já na parte real do cromossomo, os genes correspondem as parâmetros e pesos dos LS-SVMs (σ_i, C_i, w_i), onde $i = 1, 2, \ldots, M$. A representação do cromossomo é apresentada abaixo.

$$
\begin{aligned}
crom_{ParteBin} &= [p_1, a_{11}, \ldots, a_{1n}, \ldots, p_M, a_{M1}, \ldots, a_{Mn}] \\
crom_{ParteReal} &= [\sigma_1, \ldots, \sigma_M, C_1, \ldots, C_M, w_1, \ldots, w_M] \\
cromossomo &= [crom_{ParteBin} \quad crom_{ParteReal}]
\end{aligned}
$$

Onde p_i é o bit que indica a presença ou ausência do classificador no comitê, a_ij é o bit que indica a presença ou ausência do atributo do j no conjunto de dados para o classificador i, $j = 1, 2, \ldots, n$.

A função de aptidão escolhida é a norma quadrática do erro do comitê, descrita na equação 5.2, logo abaixo.

$$\Psi = \|\mathbf{d} - \mathbf{y}\|^2 \tag{5.2}$$

Onde $\mathbf{d} = [d_1, ..., d_N]$, contém os padrões de saída e $\mathbf{y} = [y_1, ..., y_N]$, contém as respostas do comitê. Cada valor de decisão do comitê y_k é calculado pela combinação linear das saídas dos classificadores para cada padrão de entrada x_k (**o**) e o vetor de pesos **w**, equação 5.3.

$$y_k = \mathbf{o}^T\mathbf{w}, \ para \ k = 1, ..., N \tag{5.3}$$

Assim, nós podemos formular o problema de otimização para ser resolvido da seguinte forma:

$$\min \Psi = \|\mathbf{d} - \mathbf{y}\|^2 \tag{5.4}$$

Sujeito a:

1. $\sum_{i=1}^{M} w_i = 1$
2. $\vee_{j=1}^{n} a_{ij} = 1, \;\; i = 1, 2, ..., M$
3. $\sigma_i, C_i > 0$ e $w_i \geq 0, \;\; i = 1, 2, ..., M$

5.2 Abordagem Genética

A escolha da configuração do AG foi feita experimentalmente. A população inicial é gerada aleatoriamente com 20 indivíduos. Nós empregamos a seleção por torneio, onde dois indivíduos participam em cada torneio e o mais apto é escolhido. Os pares de indivíduos para realizar o cruzamento são escolhidos aleatoriamente e utilizamos o operador de cruzamento uniforme. Três operadores de mutação são utilizados no cromossomo: 1) o operador de mutação *flip* é utilizado sobre a parte binária do cromossomo, invertendo o valor do bit; 2) o operador de mutação *creep* é utilizado sobre os genes que codificam os parâmetros dos classificadores, adicionado ou subtraindo um valor aleatório, desde que respeitem a restrição que proíbe valores negativos para os parâmetros; 3) o operador de mutação por troca atua sobre os valores dos pesos, sorteando pares de pesos para trocarem de valor. O AG irá executar por 100 gerações e em cada uma delas, os dois indivíduos mais aptos são passados para a próxima geração. O algoritmo do método proposto é apresentado no Algoritmo 5.1 e seu fluxograma na Figura 5.4.

Algoritmo 5.1: Abordagem Genética

Entrada: $S = \{(x_1, d_1), ..., (x_N, d_N); x_k \in \Re^n, d_k \in \{-1, 1\}\}$, a base de dados
Saída: A classificação do *ensemble* para o conjunto de teste

1: Chame o AG para inicializar a população aleatoriamente com I indivíduos
2: **enquanto** critério de parada não é atingido **faça**
3: **para** $i = 1$ até I **faça**
4: Extraia do i-ésimo cromossomo, os atributos selecionados para cada classificador, m números de LS-SVMs e seus parâmetros (σ, C) e realize o treinamento usando P
5: Realize a combinação linear das respostas das LS-SVMs com o vetor de pesos **w** contido no i-ésimo cromossomo
6: $y_k^{(i)} = \mathbf{o}^{(i)T}\mathbf{w}$, $k = 1, \ldots, N$
7: $\mathbf{y}^{(i)} = \left[y_1^{(i)}, y_2^{(i)}, \ldots, y_k^{(i)}\right]^T$, a i-ésima resposta do ensemble
8: Calcule o valor de aptidão do i-ésimo indivíduso usando função de aptidão
9: $\Psi^{(i)} = \|\mathbf{d} - \mathbf{y}^{(i)}\|^2$
10: **fim para**
11: Selecione os pais
12: Faça o cruzamento dos pais selecionados
13: Realize a mutação nos novos indivíduos
14: Atualize a população
15: **fim enquanto**
16: Resgate o indivíduo mais apto para realizar a fase de teste usando V
17: **retorne** Classificação final do ensemble

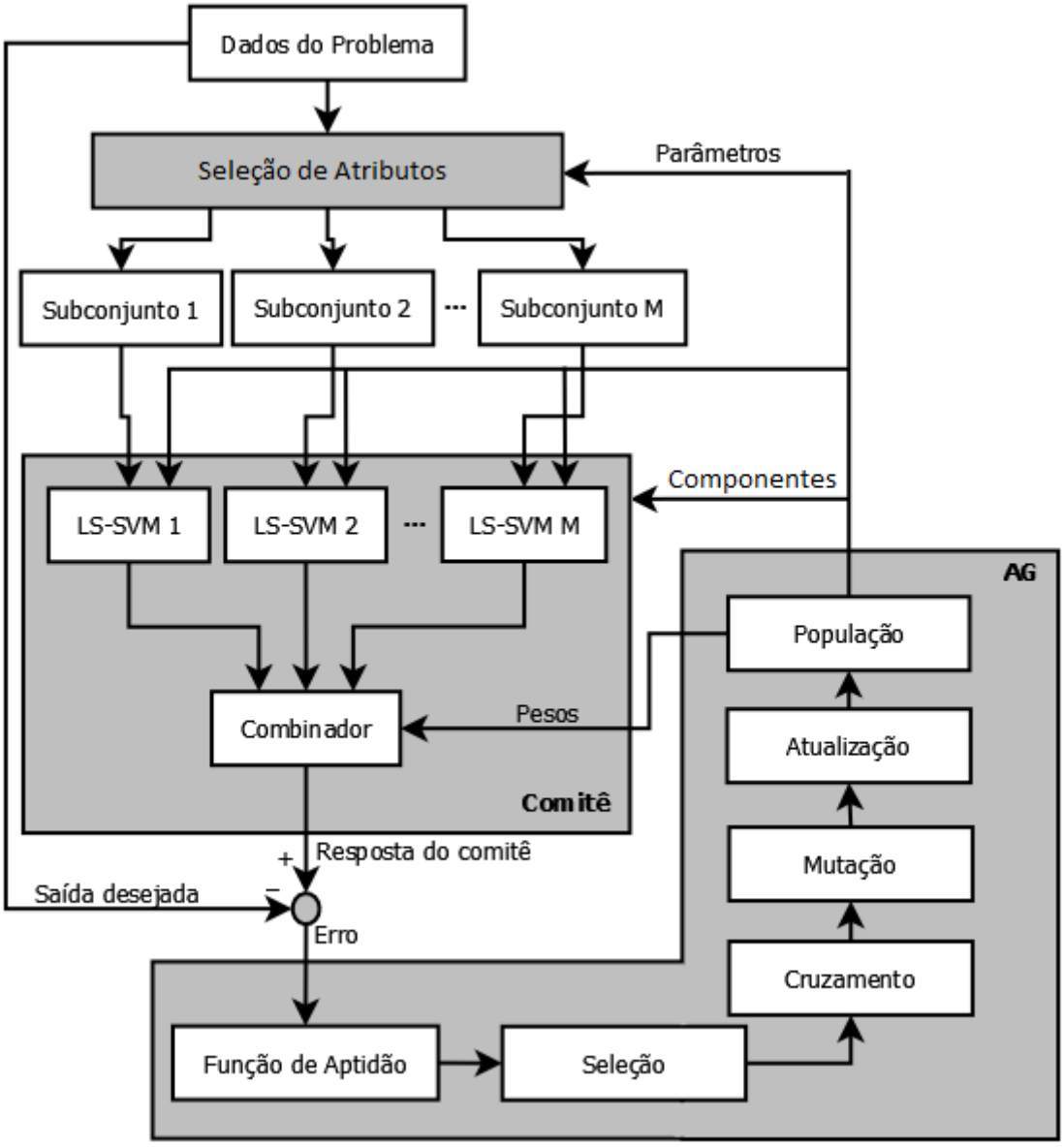


Figura 5.4: Fluxograma do método proposto.

5.3 Abordagem Híbrida

O bom desempenho dos AGs, tal como qualquer algoritmo de otimização global, depende do balanceamento entre a exploração de novas áreas promissoras e a intensificação das buscas em regiões com as melhores soluções já encontradas. Em particular, o poder dos AGs se deve por conseguir resolver esse balanceamento de maneira ótima (HOLLAND, 1975). Entretanto, apesar disso ser teoricamente verdade, há alguns problemas que surgem na prática.

Segundo (BEASLEY; BULL; MARTIN, 1993), esses problemas são causados por Holland ter assumido que a população tem tamanho infinito, que a função de aptidão reflete com precisão quão adequada é uma solução e que as interações entre genes são bem pequenas. Na prática, o tamanho da população de indivíduos é finito, influenciando a capacidade de amostragem do AG e, consequentemente, afeta sua performance. Incorporando um método de busca local dentro de um AG pode ajudar a superar a maior parte dos problemas que surgem devido ao tamanho fixo da população.

A incorporação de métodos de busca local pode ajudar a combater a perda da diversidade entre as soluções presentes na população, também conhecido como o problema da deriva genética (ASOH; MüHLENBEIN, 1994; THIERENS et al., 1998) que é causado pelo acúmulo de erros estocásticos em populações finitas. Além disso, isso também pode impulsionar a busca pelo mínimo global (HART, 1994) que, por sua vez, pode garantir que a taxa de convergência seja grande o suficiente para impedir a deriva genética.

Devido ao tamanho limitado da população, o AG pode fazer uma amostragem de indivíduos "ruins"em regiões "boas"e de "bons"indíviduos em regiões "ruins"do espaço de busca. Segundo (GRUAU; WHITLEY, 1993), um método de busca local pode garantir uma representação mais adequada das diferentes áreas de busca por fazer a amostragem de seus mínimos locais e, assim, também reduzir a probabilidade de ocorrer uma convergência prematura.

Além disso, uma população de tamanho finito pode fazer com que o AG produza soluções de qualidade inferior àquelas que podem ser produzidas por métodos de busca local. A dificuldade em encontrar as melhores soluções contidas nas melhores regiões já encontradas pelo AG, se deve à incapacidade do mesmo em executar pequenos movimentos na vizinhança da solução atual (REEVES, 1994). Utilizando um algoritmo de busca local junto a um AG pode aprimorar a capacidade de intensificação das buscas sem comprometer a habilidade de exploração (HART, 1994). Caso um correto balanço entre exploração global e intensificação local seja alcançado, o algoritmo híbrido poderá produzir soluções com alta acurácia (LOBO; GOLDBERG, 1997).

Um outro ponto diz respeito à performance em termos da velocidade de convergência. Embora os AGs consigam encontrar a região onde se encontra o ótimo global, eles levam um tempo relativamente grande para localizar exatamente o ótimo local na região de convergência (PREUX; TALBI, 1999; JONG, 2005). A combinação de um AG e método de busca local pode acelerar a busca para localizar o exato mínimo global. Em tal híbrido, o AG guia o uso da busca local nas regiões mais promissoras do espaço de busca e pode acelerar a convergência para um ótimo global.

Pensando nisso, além da abordagem genética, nós pensamos em criar uma segunda abordagem que faz uso de um método híbrido que combina o poder de exploração do AG e a intensificação das buscas locais que as metaheurísticas de solução única podem oferecer.

A estratégia usada aqui é fazer uma combinação integrativa, uma das abordagens mais amplamente utilizadas, deixando o AG como "principal"otimizador e a metaheurística baseada em solução única (Busca Tabu, *Iterated Local Search*, *Simulated Annealing*, entre outras) ficando subordinada a ele.

O fluxo da nossa abordagem híbrida é o seguinte:

1. O AG inicia a processo de busca e se mantém executando por até *i* iterações sem que haja melhora na melhor solução encontrada até então.

2. Caso o número de iterações sem melhoria seja atingido, o AG chama o método de busca local.

3. A busca local será executada em um subconjunto da população (incluindo a melhor solução) amostrado aleatoriamente, onde cada indivíduo é usado como solução inicial da busca local.

4. As soluções finais de cada busca local sobrescrevem os respectivos indivíduos da população, caso seus valores de aptidão sejam superiores.

5. O AG retoma o processo de busca.

Nós testamos 3 das metaheurísticas baseadas em solução única apresentadas no Capítulo 4, para compor a abordagem híbrida com o AG: Busca Tabu (GLOVER, 1986), *Iterated Local Search* (LOURENçO; MARTIN; STüTZLE, 2002), *Simulated Annealing* (KIRKPATRICK; GELATT; VECCHI, 1983). Através de análise empírica, nós determinamos o uso de 10 iterações sem que haja melhora na solução atual como condição para chamar o método local e também que o subconjunto que é amostrado, represente 5% da população. A idéia de aplicar

uma busca local na melhor solução corrente e em outros indivíduos escolhidos aleatoriamente tem como função encontrar melhores soluções e também ajudar a escapar de ótimos locais. Algoritmos como *Simulated Annealing* possuem explicitamente um mecanismo para tal. O algoritmo do abordagem híbrida é apresentado no Algoritmo 5.2 e seu fluxograma na Figura 5.5.

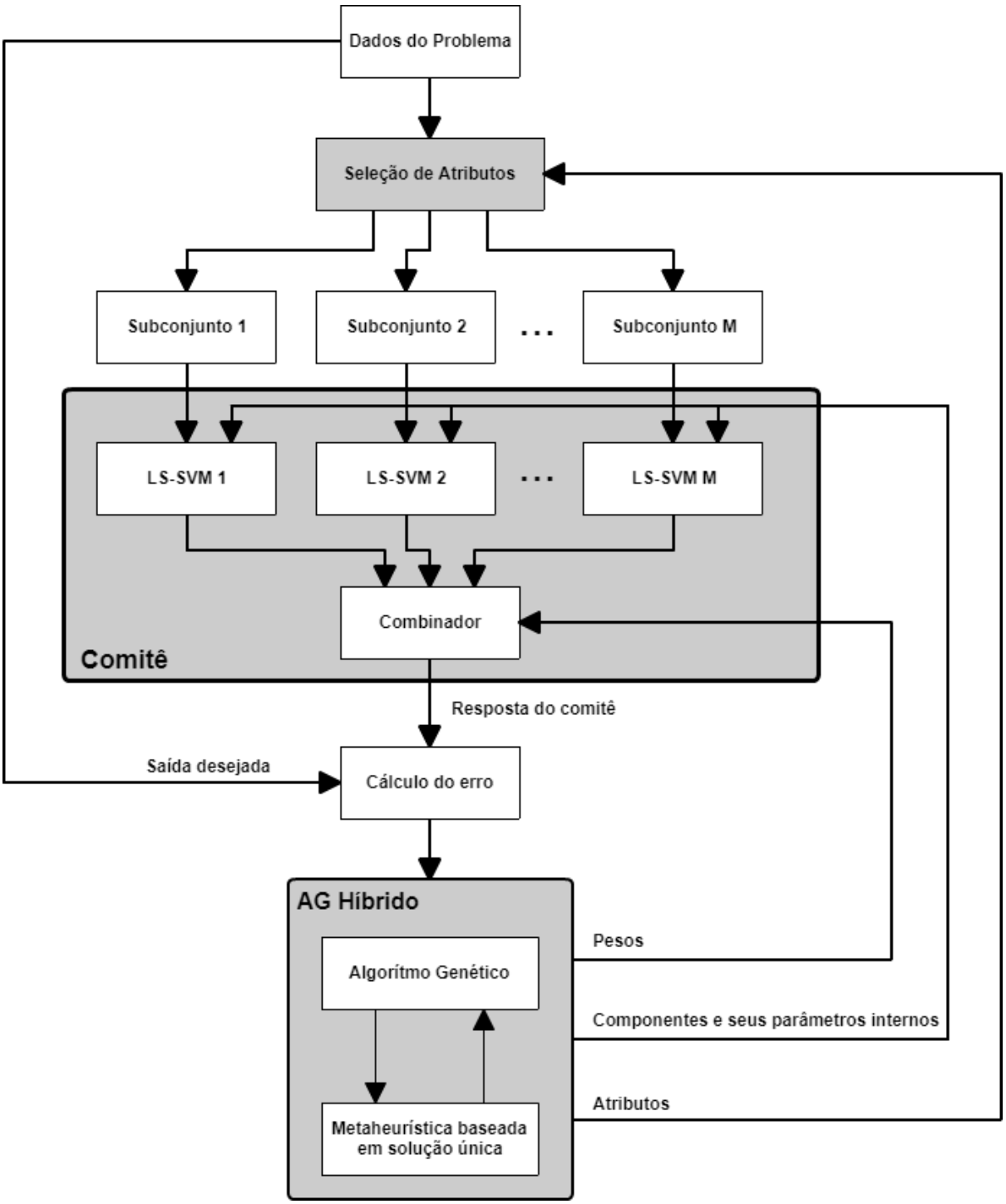


Figura 5.5: Fluxograma do método híbrido proposto.

Algoritmo 5.2: Abordagem Híbrida

Entrada: $S = \{(x_1, d_1), ..., (x_N, d_N); x_k \in \Re^n, d_k \in \{-1, 1\}\}$, a base de dados
Saída: A classificação do *ensemble* para o conjunto de teste

1: Chame o AG para inicializar a população aleatoriamente com I indivíduos
2: **enquanto** critério de parada não é atingido **faça**
3: **para** $i = 1$ até I **faça**
4: Extraia do i-ésimo cromossomo, os atributos selecionados para cada classificador, m números de LS-SVMs e seus parâmetros (σ, C) e realize o treinamento usando P
5: Realize a combinação linear das respostas das LS-SVMs com o vetor de pesos **w** contido no i-ésimo cromossomo
6: $y_k^{(i)} = \mathbf{o}^{(i)T}\mathbf{w}$, $k = 1, \ldots, N$
7: $\mathbf{y}^{(i)} = \left[y_1^{(i)}, y_2^{(i)}, \ldots, y_k^{(i)}\right]^T$, a i-ésima resposta do ensemble
8: Calcule o valor de aptidão do i-ésimo indivíduso usando função de aptidão
9: $\Psi^{(i)} = \|\mathbf{d} - \mathbf{y}^{(i)}\|^2$
10: **fim para**
11: Selecione os pais
12: Faça o cruzamento dos pais selecionados
13: Realize a mutação nos novos indivíduos
14: Atualize a população
15: **se** Numéro máximo de gerações sem melhora excedido **então**
16: Selecione aleatoriamente R indivíduos
17: **para** $r = 1$ to R **faça**
18: Chame o algoritmo de busca local e use cada r como solução inicial
19: Recupere a nova solução encontrada r'
20: **se** $fitness(r') < fitness(r)$ **então**
21: $r = r'$
22: Atualize a população
23: **fim se**
24: **fim para**
25: **se** Se alguma solução inicial r resultou em uma solução global melhor **então**
26: Atualize a melhor solução encontrada
27: Reinicie o número de gerações sem melhora
28: **fim se**
29: **fim se**
30: **fim enquanto**
31: Resgate o indivíduo mais apto para realizar a fase de teste usando V
32: **retorne** Classificação final do ensemble

6 DESCRIÇÃO DA ANÁLISE EMPÍRICA

Uma análise empírica foi realizada de maneira a validar a abordagem proposta. Para essa análise, os métodos propostos e todos os outros métodos de classificação que foram utilizados para fins de comparação, foram implementados no MATLAB R2015a. O número máximo de classificadores/estimadores para cada método baseado em *ensembles* foi empiricamente definido como 30. Em relação à seleção de atributos, não há restrição sobre o número de classificadores em que um atributo pode ser designado, ou seja, um atributo pode ser atribuído a todos os classificadores ou nenhum deles. A única restrição é que um classificador precisa ter ao menos 1 atributo (restrição 2 na formulação do problema de otimização para ser resolvido pelo AG), caso contrário ele é removido do ensemble. Finalmente, de forma a obter uma melhor estimativa das taxas de acurácia, é adotada a validação cruzada com 10 partições.

Nos experimentos, foi comparara a performance dos métodos propostos com outros que utilizam comitês e também com um classificador individual, para mostrar a vantagem em se utilizar multi-classificadores.

Primeiro, foi comparado a abordagem genética proposta com o AdaBoost (SCHAPIRE, 1999) usando SVMs como classificadores base, *Random Forests* (RF) (BREIMAN, 2001), um trabalho anterior (RSGALS-SVM) (PADILHA; NETO; MELO, 2012) e uma SVM individual otimizada por um AG. Após isso, foi comparado a abordagem proposta com quatro outras versões da mesma, onde a função de aptidão é variada e incorpora conceitos introduzidos por (BROWN; KUNCHEVA, 2010).

No trabalho (BROWN; KUNCHEVA, 2010), os autores adotam a perspectiva que a medida de diversidade deveria ser naturalmente derivada como consequência de duas decisões na construção de um ensemble: a escolha da função de erro e do método de combinação. Assim, eles propuseram uma decomposição do erro de classificação para comitês, usando um combinador por voto majoritário, em três termos: acurácia individual, diversidade boa e ruim. As definições de diversidade boa (do inglês, *good diversity* ou GD) e ruim (do inglês, *bad diversity* ou BD), são descritas nas equações 6.1 e 6.2, respectivamente.

$$GD = \frac{1}{M} \sum_{k=1}^{\#(D^+)} v_k^- \tag{6.1}$$

$$BD = \frac{1}{M} \sum_{k=1}^{\#(D^-)} v_k^+ \tag{6.2}$$

Onde v_k^- é a quantidade de votos incorretos para a amostra k no conjunto D^+ que o

comitê classificou corretamente e v_k^+ é a quantidade de votos corretos dos classificadores na instância de classificação k no conjunto de instâncias D^- que o comitê classificou incorretamente.

Após isso, foi testado a performance da nossa abordagem genética com a híbrida, considerando três metaheurísticas de busca local: Busca Tabu, *Iterated Local Search* e *Simulated Anneaning*. Esse experimento visa avaliar o ganho da hibridização e também os resultados alcançados ao variar o método de busca local.

De maneira a comparar a acurácia dos métodos de classificação nos experimentos, testes estatísticos foram aplicados, chamados de testes de hipótese ou testes t (KENDALL et al., 1999). Esse é um teste o qual envolve testar duas hipóteses aprendidas em conjuntos de teste idênticos. Para realizar este teste, um conjunto de amostras de ambos os algoritmos deve ser usado e baseado na informação fornecida, junto com o número de amostras, a significância da diferença entre dois conjuntos de amostras, em função do parâmetro α, é definida. Neste trabalho, o nível de confiança é de 95% ($\alpha = 0.05$).

Por fim, realizamos um experimento com 2 datasets de imagens bem conhecidos na comunidade, MNIST (LECUN; CORTES, 2010) e CIFAR-10 (KRIZHEVSKY, 2009), que contém centenas de atributos e milhares de amostras, para mostrar a escalabilidade do método proposto e também comparar nossos resultados com outros métodos que já publicaram resultados nessas bases, inclusive métodos de aprendizagem profunda que usam redes convolutivas, redes neurais projetadas para trabalhar imagens.

6.1 Bases de Dados - *UCI Repository*

Os experimentos foram realizados usando 14 bases de dados oriundas do UCI *Repository* (FRANK; ASUNCION,), dos quais 11 bases (*Breast Cancer*, *Diabetis*, *Flare Solar*, *German*, *Heart*, *Image*, *Ringnorm*, *Splice*, *Thyroid*, *Twornorm* e *Waveform*) foram modificadas por Ratsch et al. em (RäTSCH; ONODA; MüLLER, 2001) para o caso binário e as outras 3 (*Ionosphere*, *Hill-Valley* e *Libras Movement*) permanecem iguais. *Ionosphere* e *Hill-Valley* também são problemas binários, enquanto *Libras Movement* possui 15 classes. A Tabela 6.1 apresenta a descrição das bases de dados utilizadas nessa análise, nela apresentamos os números de atributos, instâncias dos conjuntos de treinamento e teste de cada uma delas.

A seguir, uma breve descrição de cada base utilizada:

- *Breast Cancer*: é uma base de dados sobre o câncer de mama e é composta por diversas

informações relacionadas à condição das pacientes e ao tumor que são associados a um diagnóstico, tumor benígno ou malígno.

- *Diabetis* é outra base para diagnóstico de diabetes em mulheres com pelo menos 21 anos da linhagem do índios Pima que vivem no Arizona.
- *Flare-Solar* contém características capturadas de regiões ativas do sol para classificação de tempestades solares.
- *German* é uma base que classifica pessoas descritas por um conjunto de atributos como tendo baixo ou alto risco de crédito.
- *Heart* tem informações para classificar se pacientes tem ou não doença cardíaca.
- *Hill-Valley* é uma base que contém pontos em um gráfico bidimensional formando colinas ou vales.
- *Image* é outra base que contém informações para a classificação de uma região 3x3 segmentada de imagens ao ar livre.
- *Ionosphere* é uma base de dados usada no passado na classificação de retornos de radares da ionosfera (SIGILLITO et al., 1989), tendo os elétrons livres na mesma o alvo desse estudo. Aqueles que mostram a evidência de alguma estrutura na ionosfera são bons retornos de radar, do contrário, são considerados retornos ruins.
- *Libras Mov.* é uma base composta por 15 classes que são referentes a tipos de movimento feito com a mão em LIBRAS (Língua Brasileira de Sinais).
- *Ringnorm* classifica um dado padrão como vindo de uma das duas distribuições normais sobrepostas.
- *Splice* reconhece duas classes de juntas de processamento numa sequência de DNA. Essas juntas são pontos em uma sequência de DNA em que o DNA dito "supérfluo", é removido durante o processo de criação de proteína em organismos superiores.
- *Thyroid* diagnostica se um paciente tem doença na tireoide.
- *Twonorm* classifica um dado padrão como proveniente de uma das distribuições normais, uma se encontra dentro da outra.
- *Waveform* é uma base de dados gerada artificialmente e classifica se uma onda é proveniente de uma das duas ondas base.

Tabela 6.1: Bases de Dados

Bases de Dados	Atributos	Instâncias de Treinamento	Instâncias de teste
Breast Cancer	9	200	77
Diabetis	8	468	300
Flare-Solar	9	666	400
German	20	700	300
Heart	13	170	100
Hill-Valley	100	606	606
Image	18	1300	1010
Ionosphere	34	200	151
Libras Mov.	90	360	450
Ringnorm	20	400	7000
Splice	60	1000	2175
Thyroid	5	140	75
Twonorm	20	400	7000
Waveform	21	400	4600

6.2 MNIST

O MNIST (LECUN; CORTES, 2010) é uma grande base de dados de imagens de dígitos manualmente escritos, números inteiros de 0 até 9, que é amplamente conhecida e utilizada para avaliar algoritmos de classificação, especialmente aqueles voltados para reconhecimento de imagens. Ele foi construído a partir das bases de dados especiais SD-1 e SD-3 do NIST (*National Institute of Standards and Technology database*) as quais contém imagens binárias de dígitos escritos à mão. Originalmente, o NIST designou o SD-3 como conjunto de treinamento e o SD-1 como o conjunto de teste. Entretanto, há uma grande diferença entre a qualidade das imagens dos dois conjuntos, o SD-3 é mais limpo e fácil de reconhecer que o SD-1. Isso se deve ao fato que os conjuntos foram coletados por grupos totalmente distintos, sendo o SD-3 coletado entre os funcionários do *Census Bureau*, enquanto que o SD-1 fora coletado entre alunos do ensino médio. Portanto, os dados precisaram ser misturados e deram origem ao NIST modificado ou *Modified NIST* (MNIST).

Assim, o conjunto de treino no MNIST é composto por trinta mil amostras do SD-3 e outras trinta mil amostras do SD-1. O conjunto de teste é composto por cinco mil exemplos do SD-3 e outros cinco mil exemplos do SD-1. Segundo (LECUN; CORTES, 2010), todas essas sessenta mil amostras foram coletadas de aproximadamente 250 escritores e foi garantido que o conjunto deles nos conjuntos de treino e teste são disjuntos.

6.3 CIFAR-10

O CIFAR-10 (KRIZHEVSKY, 2009) é uma base de dados para trabalhos envolvendo reconhecimento de objetos já bem estabelecida. Os dados foram coletados por Alex Krizhevsky, Vinod Nair e Geoffrey Hinton, sendo um subconjunto de uma base de dados contendo oitenta milhões de imagens pequenas. CIFAR-10 consiste de sessenta mil imagens coloridas com resolução 32x32 pixels contendo um objeto das dez classes possíveis, 6000 imagens de cada objeto, que são: avião (*airport*), automóvel (*automobile*), pássaro (*bird*), gato (*cat*), cervo (*deer*), cachorro (*dog*), sapo (*frog*), cavalo (*horse*), barco (*ship*) e caminhão (*truck*).

Essas imagens são divididas em cinquenta mil imagens para o conjunto de treinamento e o restante para o conjunto de teste. O conjunto todo está dividido em cinco lotes de treinamento e um de teste, cada um contendo dez mil imagens. O lote de teste contém mil amostras aleatoriamente selecionadas de cada classe. Os lotes para treinamento contém o restante das imagens e alguns deles podem conter mais amostras de uma determinada classe que outros, mas ao todo são cinco mil imagens de cada classe.

As classes nas imagens se apresentam mutualmente exclusivas, não havendo sobreposição entre automóveis e caminhões, por exemplo. Os automóveis englobam veículos não tão grandes, como sedans e SUVs, enquanto que na classe caminhão são incluídos apenas grandes caminhões.

7 RESULTADOS EMPÍRICOS

Neste capítulo, os resultados obtidos em diversos experimentos são apresentados e cada experimento terá sua própria seção para uma melhor organização. As soluções encontradas pelas abordagens propostas estão disponibilizadas em http://cpadilha.github.io/MLGALSSVM/.

7.1 Comparação da Abordagem Genética com outros Métodos

Afim de avaliar a performance da abordagem genética, foram escolhidos alguns outros métodos também baseados em comitês para termos uma comparação mais justa. São eles: um *ensemble* de SVMs treinado com o AdaBoost (SCHAPIRE, 1990) e uma *Random Forest*. Além deles, nosso trabalho anterior (PADILHA; NETO; MELO, 2012), indicado pela sigla RSGALS-SVM, que usa o método *Random Subspace* para fazer a seleção de atributos e um AG para otimizar os parâmetros internos das LS-SVMs (σ e C) e vetor de pesos de saída, também é considerado nessa comparação. Por fim, também incluímos os resultados obtidos por uma única SVM cujos parâmetros internos foram otimizados por um AG.

Na Tabela 7.1, nós apresentamos os valores médios de acurácia e desvio padrão obtidos pelo método proposto, indicado como MLGALS-SVM, AdaBoost com SVM, *Random Forest*, indicado como RF, uma SVM otimizada por um AG e o RSGALS-SVM (PADILHA; NETO; MELO, 2012), quando aplicados às bases de dados. Os melhores valores em cada base são marcados em negrito. Para atestar se a diferença entre os valores médios de acurácia apresentados na Tabela 7.1 é de fato significativa, foi feito a aplicação de um teste t bicaudal com 5% de nível de significância e os valores de p colocados na Tabela 7.2. Os valores menores que 0,05 indicam que as diferenças entre as médias são significativas.

Como podemos ver na Tabela 7.1 e a análise da Tabela 7.2, o método proposto obteve resultados significativamente superiores em 10 de 14 bases, perdendo apenas nos testes com *Ringnorm* (tendo o AdaBoost como vencedor) e *Waveform* (RSGALS-SVM como vencedor) e tendo resultados estatisticamente igual em *Image* (empate com RSGALS-SVM) e *Splice* (empate com AdaBoost). Assim como visto em (PADILHA; NETO; MELO, 2012), o RSGALS-SVM superou os resultados do AG-SVM e o AdaBoost na maioria dos testes. Apesar de ter seus parâmetros (σ e C) otimizados pelo AG, uma única SVM não pôde superar o AdaBoost

Tabela 7.1: Tabela contendo os valores médios de acurácia e desvio padrão obtidos nas bases de dados - Comparação da Abordagem Genética com Outros Métodos.

Bases de Dados	MLGALS-SVM	AdaBoost	RSGALS-SVM	AG-SVM	RF
Breast Cancer	$\mathbf{88.3 \pm 4.6}$	74 ± 4.7	74.1 ± 0.1	69.6 ± 4.7	70.3 ± 6.7
Diabetis	$\mathbf{82.7 \pm 1.9}$	76.5 ± 1.7	76.3 ± 0.3	73.5 ± 2.3	75.6 ± 1.7
Flare-Solar	$\mathbf{73.0 \pm 2.4}$	67.6 ± 1.8	70.3 ± 0.6	64.3 ± 1.8	65.8 ± 1.7
German	$\mathbf{83.3 \pm 2.5}$	76.4 ± 2.1	78 ± 0.6	72.5 ± 2.5	76.7 ± 2.4
Heart	$\mathbf{96.0 \pm 3.7}$	84 ± 3.3	87 ± 0.5	79.7 ± 3.4	82.1 ± 3.6
Hill-Valley	$\mathbf{90.4 \pm 4.7}$	67.8 ± 1.6	72.1 ± 1.8	69.5 ± 1.0	56.8 ± 0.8
Image	$\mathbf{98.4 \pm 1.6}$	97 ± 0.6	96.3 ± 0.1	97.3 ± 0.4	97.4 ± 0.4
Ionosphere	$\mathbf{96.4 \pm 1.3}$	87.6 ± 1.2	90.2 ± 0.8	88.0 ± 1.0	92.1 ± 3.8
Libras Mov.	$\mathbf{93.4 \pm 0.8}$	85.4 ± 1.6	88.7 ± 0.8	84.4 ± 1.0	90.7 ± 0.6
Ringnorm	97.9 ± 0.9	$\mathbf{98.3 \pm 0.1}$	97.5 ± 0.3	98.1 ± 0.3	93.6 ± 0.9
Splice	$\mathbf{91.9 \pm 2.0}$	89.1 ± 0.7	90.1 ± 0.1	89.9 ± 0.5	89.1 ± 0.5
Thyroid	$\mathbf{97.3 \pm 2.7}$	95.2 ± 2.2	94.7 ± 0.6	95.6 ± 0.6	95.1 ± 2.5
Twonorm	$\mathbf{98.1 \pm 0.5}$	97 ± 0.2	97.9 ± 0.2	97 ± 0.3	95.6 ± 0.4
Waveform	89.4 ± 0.4	90.1 ± 0.4	$\mathbf{92.7 \pm 0.2}$	89.2 ± 0.6	88.5 ± 0.6

Tabela 7.2: Tabela contendo os valores de p dos testes t realizados, comparando o método proposto com os outros algoritmos.

Bases de Dados	AG-SVM	RF	AdaBoost	RSGALS-SVM
Breast Cancer	$\mathbf{0.0000}$	$\mathbf{0.0000}$	$\mathbf{0.0000}$	$\mathbf{0.0000}$
Diabetis	$\mathbf{0.0000}$	$\mathbf{0.0000}$	$\mathbf{0.0002}$	$\mathbf{0.0000}$
Flare-Solar	$\mathbf{0.0000}$	$\mathbf{0.0000}$	$\mathbf{0.0435}$	$\mathbf{0.0000}$
German	$\mathbf{0.0000}$	$\mathbf{0.0000}$	$\mathbf{0.0303}$	$\mathbf{0.0000}$
Heart	$\mathbf{0.0000}$	$\mathbf{0.0000}$	$\mathbf{0.0000}$	$\mathbf{0.0000}$
Hill-Valley	$\mathbf{0.0000}$	$\mathbf{0.0000}$	$\mathbf{0.0000}$	$\mathbf{0.0000}$
Image	$\mathbf{0.0092}$	$\mathbf{0.0089}$	$\mathbf{0.0324}$	0.4664
Ionosphere	$\mathbf{0.0000}$	$\mathbf{0.0000}$	$\mathbf{0.0000}$	$\mathbf{0.0000}$
Libras Mov.	$\mathbf{0.0000}$	$\mathbf{0.0000}$	$\mathbf{0.0000}$	$\mathbf{0.0000}$
Ringnorm	$\mathbf{0.0000}$	$\mathbf{0.0000}$	$\mathbf{0.0000}$	$\mathbf{0.0000}$
Splice	$\mathbf{0.0426}$	$\mathbf{0.0411}$	0.5144	$\mathbf{0.0205}$
Thyroid	$\mathbf{0.0000}$	$\mathbf{0.0000}$	$\mathbf{0.0000}$	$\mathbf{0.0000}$
Twonorm	$\mathbf{0.0000}$	$\mathbf{0.0000}$	$\mathbf{0.0000}$	$\mathbf{0.0000}$
Waveform	$\mathbf{0.0000}$	$\mathbf{0.0000}$	$\mathbf{0.0000}$	$\mathbf{0.0000}$

e RF (perdeu em 8 de 14 testes), mostrando a efetividade em se utilizar uma estrutura com multi-classificadores.

Então, resolvemos investigar um pouco mais o histórico dos experimentos com as bases *Image*, *Ringnorm*, *Splice* e *Waveform*, onde o método não foi superior aos demais, para descobrir a causa para isso. Assim, foi feito um comparativo do número de iterações até o processo de otimização ser finalizado que cada um dessas bases levou com as outras 10 bases onde o método proposto se mostrou superior. Esse comparativo é apresentado na Figura 7.1.

De acordo com o gráfico na Figura 7.1, podemos perceber claramente três tipos de comportamento: (1) O processo de otimização nas bases Image e Ringnorm foi finalizado com mediana abaixo de 20 iterações; (2) Já o comportamento do processo nas bases Splice e Waveform foi bem diferente, praticamente em 100 iterações; (3) Por fim, o comportamento do

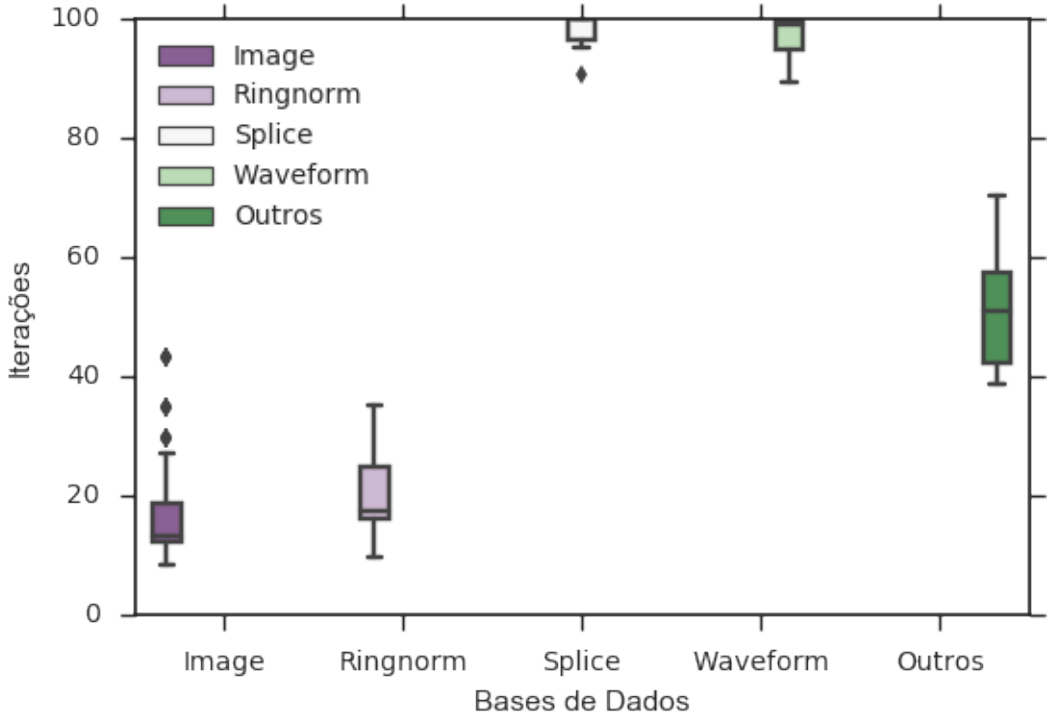


Figura 7.1: Comparação do número de iterações passadas até o término do processo de otimização para as bases textitImage, *Ringnorm*, *Splice*, *Waveform* e as demais, indicadas como Outros.

processo de otimização nas demais 10 bases se mostrou intermediário, com mediana próxima a 50 iterações. O primeiro comportamento pode mostrar que houve uma convergência prematura da otimização para um mínimo local por perda de diversidade na população. Já o segundo, mostra que na maior parte das vezes o processo de otimização foi finalizado por ultrapassar o número máximo de iterações (adotamos o número de 100 iterações como um dos critérios de parada), ou seja, talvez ainda houvesse a possibilidade de encontrar melhores soluções caso tivéssemos executado com mais iterações. Esse atraso pode ter ocorrido devido ao método ter ficado preso em um mínimo local ou pela deficiência dos Algoritmos Genéticos em fazer buscas locais de forma eficiente.

Portanto, podemos concluir que a nossa abordagem genética, apesar de superior na maioria dos problemas experimentados, poderia ser incrementada com mecanismos para garantir diversidade e também acelerar a busca por melhores soluções locais.

Além disso, os tempos de processamento médios despendidos por cada método em cada base de dados são apresentados na Figura 7.2. Aqui vale a observação que apesar do método proposto executar uma busca que engloba os três níveis do comitê, ele dispende menos tempo que o nosso trabalho anterior (PADILHA; NETO; MELO, 2012). Isso pode acontecer porque desta vez a seleção de atributos foi incluída no processo de otimização e contribuiu acelerando

o processo.

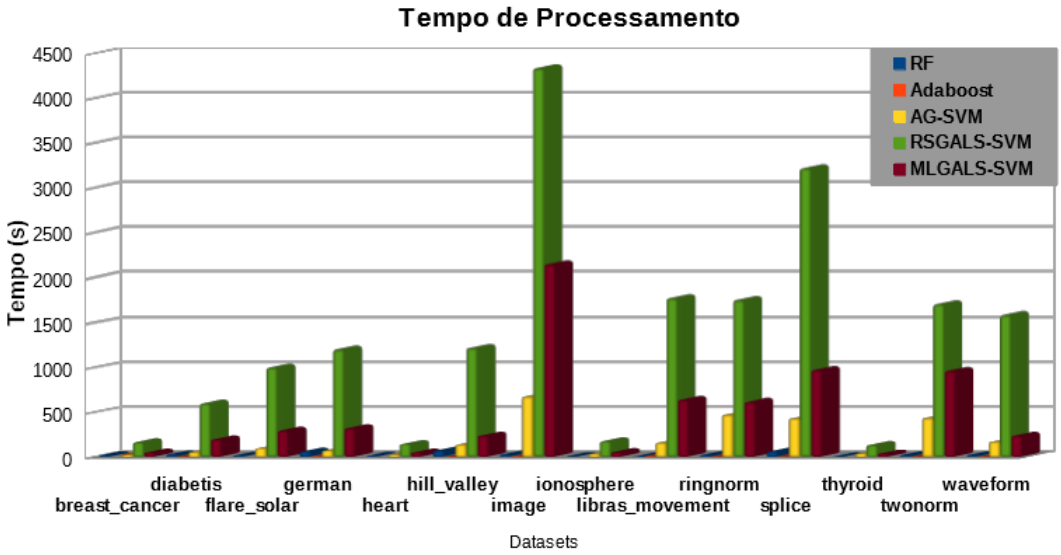


Figura 7.2: Comparação dos tempos de processamento.

7.2 Comparação da Abordagem Genética usando Outras Funções de Aptidão

Então, nós conduzimos uma segunda sequência de testes para analizar a performance do método proposto quando incluímos os conceitos de diversidades boa e ruim (BROWN; KUNCHEVA, 2010) na função de aptidão. Assim, preparamos três novas versões da função de aptidão, a primeira versão usa somente a diversidade boa (Equação 6.1), a segunda usa somente a diversidade ruim (equação 6.2), enquanto que a terceira consiste da norma quadrática do erro do *ensemble* menos a diversidade ruim, como segue:

$$\Psi^{v1} = -GD = -\frac{1}{M}\sum_{k=1}^{\#(D^+)} v_k^- \tag{7.1}$$

$$\Psi^{v2} = BD = \frac{1}{M}\sum_{k=1}^{\#(D^-)} v_k^+ \tag{7.2}$$

$$\Psi^{v3} = \|\mathbf{d} - \mathbf{y}\|^2 + BD \tag{7.3}$$

Os resultados desse segundo experimento, os valores médios de acurácia e desvio padrão, estão presentes na Tabela 7.3.

Tabela 7.3: Tabela contendo os valores médios de acurácia e desvio padrão obtidos nas bases de dados - Comparação da Abordagem Genética usando Outras Funções de Aptidão.

Bases de Dados	MLGALS-SVM	Versão 1	Versão 2	Versão 3
Breast Cancer	88.3 ± 4.6	82.7 ± 4.3	81.8 ± 4.5	$\mathbf{88.3 \pm 4.0}$
Diabetis	$\mathbf{82.7 \pm 1.9}$	80.1 ± 15.0	79.7 ± 15.7	81.3 ± 1.9
Flare-Solar	$\mathbf{73.0 \pm 2.4}$	70.6 ± 1.8	70 ± 2.1	72.2 ± 3.0
German	$\mathbf{83.3 \pm 2.5}$	76.6 ± 7.3	76 ± 7.3	81.7 ± 2.5
Heart	$\mathbf{96 \pm 3.7}$	90.3 ± 16.4	91 ± 16.7	95 ± 2.8
Hill-Valley	$\mathbf{90.4 \pm 4.7}$	74.8 ± 2.1	74.2 ± 2.3	76.4 ± 9.0
Image	$\mathbf{98.4 \pm 1.3}$	65.7 ± 1.2	65.2 ± 1.5	97.8 ± 1.3
Ionosphere	$\mathbf{96.4 \pm 1.3}$	86.0 ± 0.8	85.5 ± 1.0	92.3 ± 1.5
Libras Mov.	$\mathbf{93.4 \pm 0.8}$	89.5 ± 0.3	89.0 ± 0.5	95.0 ± 0.3
Ringnorm	$\mathbf{97.9 \pm 1.7}$	96.0 ± 2.5	95.3 ± 2.7	97.6 ± 2.6
Splice	$\mathbf{91.9 \pm 2.0}$	58.7 ± 0.7	58.2 ± 0.9	91.2 ± 3.3
Thyroid	$\mathbf{97.3 \pm 2.7}$	86.8 ± 3.5	86.1 ± 3.9	92.7 ± 3.0
Twonorm	$\mathbf{98.1 \pm 0.5}$	$\mathbf{98.1 \pm 1.2}$	97.7 ± 1.2	98 ± 0.4
Waveform	$\mathbf{89.4 \pm 0.4}$	84.9 ± 0.9	84.5 ± 0.9	89.4 ± 5.4

Para avaliar o ganho de performance do método proposto, os resultados apresentados nas Tabelas 4.1 e 7.3 foram comparados e apresentados na Tabela 7.4. Os resultados se referem aos valores de p da aplicação do teste t bicaudal. Os valores menores que 0,05 indicam que as diferenças entre as médias são significativas.

Tabela 7.4: Tabela contendo os valores de p dos testes t realizados, comparando o método proposto e suas variações.

Bases de Dados	Versão 1	Versão 2	Versão 3
Breast Cancer	**0.0000**	**0.0000**	0.4195
Diabetis	**0.0000**	**0.0000**	0.1006
Flare-Solar	**0.0000**	**0.0000**	0.9186
German	**0.0000**	**0.0000**	**0.0207**
Heart	**0.0000**	**0.0000**	0.2419
Hill-Valley	**0.0000**	**0.0000**	0.1897
Image	**0.0000**	**0.0000**	0.2595
Ionosphere	**0.0000**	**0.0000**	**0.0000**
Libras Mov.	**0.0000**	**0.0000**	**0.0000**
Ringnorm	**0.0000**	**0.0000**	0.2275
Splice	**0.0426**	**0.0000**	0.4343
Thyroid	**0.0000**	**0.0000**	0.3647
Twonorm	0.2649	**0.0000**	0.3177
Waveform	0.0892	0.0773	0.1064

Agora analisando os resultados da segunda sequência de testes vistos nas Tabelas 7.3 e 7.4, podemos ver que as versões 1 e 2 que utilizam unicamente os conceitos de diversidade boa e ruim, respectivamente, obtiveram resultados bem parecidos, com uma pequena vantagem da

versão 1 sobre a 2. A versão 1 obteve resultados significativamente piores em 12 de 14 bases em relação ao método proposto e estatisticamente igual nas bases *Twonorm*, obtendo nessa um resultado ligeiramente superior inclusive, e *Waveform*. Já avaliando a versão que utilizou apenas a diversidade ruim (versão 1) teve resultados significativamente piores em 13 de 14 bases e estatisticamente igual aos outros na base *Waveform*. Comparando a versão original e a versão que inclui o cálculo da diversidade ruim na função de aptidão (versão 3), podemos ver que não houve ganho na performance da versão modificada parada à original. Eles tem resultados estatisticamente similares em 11 de 14 bases e, além disso, a versão original tem vantagem significativa em *German*, *Ionosphere* e *Libras Movement*.

7.3 Comparação da Abordagem Híbrida com Outros Métodos

De forma a avaliar o desempenho da nossa abordagem híbrida, primeiro nós experimentamos a hibridização do AG com três metaheurísticas baseadas em solução única bastante conhecidas: a) *Simulated Annealing* (SA); b) Busca Tabu (BT); c) *Iterated Local Search* (ILS).

Os nomes dos métodos híbridos são os seguintes:

- AG + SA é indicado como GASALS-SVM;
- AG + BT é indicado como GATSLS-SVM;
- AG + ILS é indicado como GAI2LS-SVM.

Para este primeiro experimento com o método híbrido, usamos como *baseline* os resultados da abordagem genética apresentados anteriormente. Os valores médios de acurácia e desvio padrão obtidos pelo método híbrido (MLGALS-SVM), GASALS-SVM, GATSLS-SVM e GAI2LS-SVM são apresentados na Tabela 7.5. Aplicamos testes t de com nível de significância de 5% para avaliar se um resultado é significantemente superior a outro e marcamos em negrito esses valores. Adicionalmente, comparamos os tempos médios de execução de cada método em cada base de dados na Figura 7.3.

Como podemos ver na Tabela 7.5, as abordagens híbridas mostraram, no geral, resultados significativamente melhores que a abordagem genética, enquanto que nas bases *Libras Movement* e *Waveform* todos os resultados são estatisticamente equivalentes. Entre as abordagens híbridas testadas, a Busca Tabu (GLOVER, 1986) se mostrou a melhor parceria para

Tabela 7.5: Tabela contendo os valores médios de acurácia e desvio padrão obtidos pela abordagem genética e versões do método híbrido.

Datasets	MLGALS-SVM	GASALS-SVM	GATSLS-SVM	GAI2LS-SVM
Breast Cancer	88.3 ± 4.6	89.6 ± 4.2	$\mathbf{90.9 \pm 1.7}$	88 ± 4.1
Diabetis	82.7 ± 1.9	$\mathbf{83.6 \pm 1.9}$	83.3 ± 0.8	82 ± 2.2
Flare-Solar	$\mathbf{73.0 \pm 2.4}$	72.2 ± 2.1	$\mathbf{73.0 \pm 2.4}$	70 ± 2.1
German	$\mathbf{83.3 \pm 2.5}$	83.3 ± 2.3	$\mathbf{84.2 \pm 2.3}$	81.7 ± 2.5
Heart	96.0 ± 3.7	95 ± 2.5	$\mathbf{96 \pm 1.7}$	95 ± 2.8
Hill-Valley	90.4 ± 4.7	94.5 ± 3.2	95.4 ± 4.7	$\mathbf{96.4 \pm 3.3}$
Image	98.4 ± 1.6	$\mathbf{99.2 \pm 0.6}$	$\mathbf{99.4 \pm 0.7}$	97.8 ± 1.3
Ionosphere	96.4 ± 1.3	96.9 ± 1.3	$\mathbf{97.4 \pm 1.1}$	92.3 ± 1.5
Libras Mov.	93.4 ± 0.8	93.9 ± 0.9	94.0 ± 0.8	93.6 ± 1.0
Ringnorm	97.9 ± 0.9	97.9 ± 0.5	$\mathbf{98.5 \pm 0.2}$	97.6 ± 2.6
Splice	91.9 ± 2.0	92.7 ± 1.2	$\mathbf{93.9 \pm 1.5}$	91.2 ± 3.3
Thyroid	97.3 ± 2.7	98.4 ± 0.4	$\mathbf{98.8 \pm 0.7}$	92.7 ± 3.0
Twonorm	98.1 ± 0.5	98 ± 0.1	$\mathbf{98.5 \pm 0.5}$	98 ± 0.4
Waveform	89.4 ± 0.4	89.3 ± 0.7	89.7 ± 0.6	89.4 ± 5.4

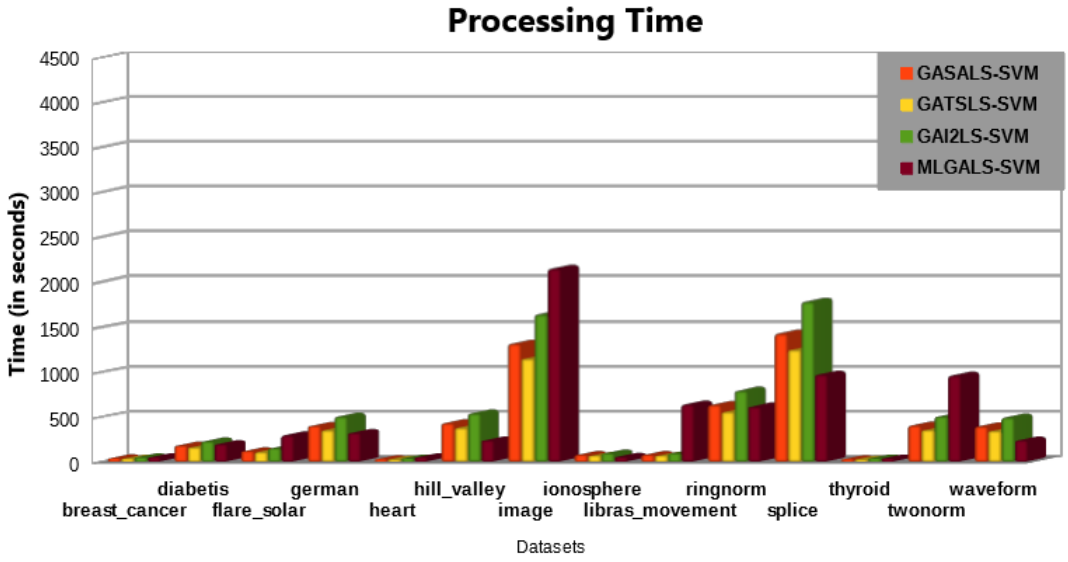


Figura 7.3: Comparação dos tempos de processamento.

o AG, com 7 vitórias/2 empates/2 derrotas. Analisando o gráfico dos tempos de execução de cada método, é possível observar uma diminuição no tempo de convergência do processo de otimização.

Em seguida, utilizando o método híbrido que teve melhor performace, GATSLS-SVM, comparamos os resultados obtidos por ele nas 14 bases de dados com os demais métodos: um *ensemble* de SVMs treinado com o AdaBoost (SCHAPIRE, 1990), *Random Forest*, AG-SVM, nosso trabalho anterior (PADILHA; NETO; MELO, 2012), indicado pela sigla RSGALS-SVM, e a abordagem genética proposta indicada pela sigla MLGALS-SVM. Na Tabela 7.6, nós

apresentamos os valores médios de acurácia e desvio padrão obtidos por eles, marcando em negrito os melhores valores, e na Tabela 7.7 os p-valores retornados pelo teste t bicaudal com 5% de nível de significância, comparando o GATSLS-SVM com os demais algoritmos. Como podemos ver nesses resultados, o método híbrido foi significativamente superior em 12 bases de dados e equivalente à abordagem genética nas bases *Flare-Solar* e *German*, corrigindo de certa forma os problemas apresentados anteriormente pela abordagem genética.

Tabela 7.6: Tabela contendo os valores médios de acurácia e desvio padrão obtidos nas bases de dados - Comparação da Abordagem Híbrida com Outros Métodos

Datasets	GATSLS-SVM	MLGALS-SVM	AdaBoost	RSGALS-SVM	RF	AG-SVM
Breast Cancer	$\mathbf{90.9 \pm 1.7}$	88.3 ± 4.6	74 ± 4.7	74.1 ± 0.1	70.3 ± 6.7	69.6 ± 4.7
Diabetis	$\mathbf{83.3 \pm 0.8}$	82.7 ± 1.9	76.5 ± 1.7	76.3 ± 0.3	75.6 ± 1.7	73.5 ± 2.3
Flare-Solar	$\mathbf{73.0 \pm 2.4}$	$\mathbf{73.0 \pm 2.4}$	67.6 ± 1.8	70.3 ± 0.6	65.8 ± 1.7	64.3 ± 1.8
German	$\mathbf{84.2 \pm 2.3}$	$\mathbf{83.3 \pm 2.5}$	76.4 ± 2.1	78 ± 0.6	76.7 ± 2.4	72.5 ± 2.5
Heart	$\mathbf{96.0 \pm 1.7}$	96.0 ± 3.7	84 ± 3.3	87 ± 0.5	82.1 ± 3.6	79.7 ± 3.4
Hill-Valley	$\mathbf{95.4 \pm 4.7}$	90.4 ± 4.7	67.8 ± 1.6	72.1 ± 1.8	56.8 ± 0.8	69.5 ± 1.0
Image	$\mathbf{99.4 \pm 0.7}$	98.4 ± 1.6	97 ± 0.6	96.3 ± 0.1	97.4 ± 0.4	97.3 ± 0.4
Ionosphere	$\mathbf{97.4 \pm 1.1}$	96.4 ± 1.3	87.6 ± 1.2	90.2 ± 0.8	92.1 ± 3.8	88.0 ± 1.0
Libras Mov.	$\mathbf{94.0 \pm 0.8}$	93.4 ± 0.8	85.4 ± 1.6	88.7 ± 0.8	90.7 ± 0.6	84.4 ± 1.0
Ringnorm	$\mathbf{98.5 \pm 0.2}$	97.9 ± 0.9	98.3 ± 0.1	97.5 ± 0.3	93.6 ± 0.9	98.1 ± 0.3
Splice	$\mathbf{93.9 \pm 1.5}$	91.9 ± 2.0	89.1 ± 0.7	90.1 ± 0.1	89.1 ± 0.5	89.9 ± 0.5
Thyroid	$\mathbf{98.8 \pm 0.7}$	97.3 ± 2.7	95.2 ± 2.2	94.7 ± 0.6	95.1 ± 2.5	95.6 ± 0.6
Twonorm	$\mathbf{98.5 \pm 0.5}$	98.1 ± 0.5	97 ± 0.2	97.9 ± 0.2	95.6 ± 0.4	97 ± 0.3
Waveform	$\mathbf{89.7 \pm 0.6}$	89.4 ± 0.4	90.1 ± 0.4	92.7 ± 0.2	88.5 ± 0.6	89.2 ± 0.6

Tabela 7.7: Tabela contendo os valores de p dos testes t realizados, comparando o GATSLS-SVM com os outros algoritmos.

Bases de Dados	AG-SVM	RF	AdaBoost	RSGALS-SVM	MLGALS-SVM
Breast Cancer	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
Diabetis	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0001**
Flare-Solar	**0.0000**	**0.0000**	**0.0000**	**0.0000**	0.7578
German	**0.0000**	**0.0000**	**0.0000**	**0.0000**	0.5932
Heart	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0383**
Hill-Valley	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
Image	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
Ionosphere	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
Libras Mov.	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
Ringnorm	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0001**
Splice	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
Thyroid	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
Twonorm	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
Waveform	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**

7.4 Resultados da Abordagem Híbrida no MNIST e CIFAR-10

Para este experimento, o processo de treinamento foi bastante demorado onde cada base de dados levou em torno de 2 semanas para ser executada a validação cruzada com 10 partições. Para ambas as bases, tínhamos cinquenta mil imagens para o treinamento e dez mil para teste e a configuração do método híbrido foi mantida, com cada *ensemble* possuindo até trinta classificadores. A ideia desse experimento é avaliar a performance do método proposto frente aos algoritmos de aprendizado profundo que hoje em dia estão na vitrine da comunidade de aprendizado de máquina.

Para a comparação dos resultados na base de dados MNIST (LECUN; CORTES, 2010), nós selecionamos alguns exemplos de métodos que foram executados nessa base e que tem seus resultados disponibilizados na página do Prof. Lecun (http://yann.lecun.com/exdb/mnist/). Lá eles estão categorizados por tipo de classificador, da seguinte forma: classificadores lineares e não-lineares, *K-Nearest Neighbors* (K-NNs) (ALTMAN, 1992), *Boosted Stumps* (KÉGL; BUSA-FEKETE, 2009), SVMs, redes neurais e redes convolutivas (LECUN et al., 1998). Dessas categorias, iremos selecionar o melhor resultado de cada uma para fazer a comparação.

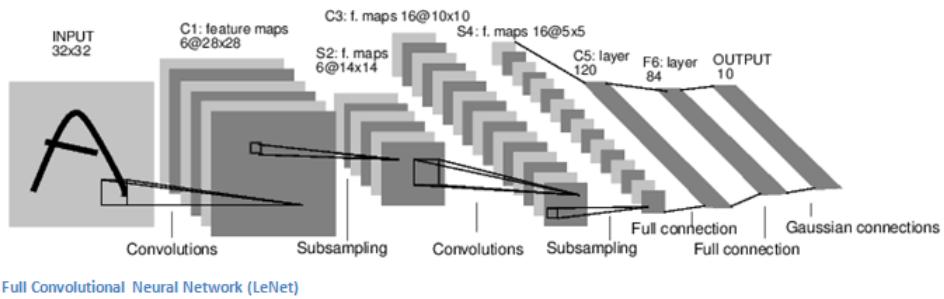
Alguns desses métodos realizaram algum tipo de pré-processamento nas imagens como: a) Um processo conhecido como *deskew* (BOUKHAROUBA, 2017) que nada mais que uma correção no enquadramento das imagens; b) Uso de *data augmentation* para aumentar o número de imagens artificialmente através da aplicação de algumas transformações como rotações, translações, distorções e compressão. Nós utilizamos o enquadramento e também fizemos uma escala nos valores dos pixels que são entre 0 e 255, para valores entre 0 e 1. Os resultados são mostrados na Tabela 7.8.

Nesses resultados pode-se ver que apesar do número grande de instâncias, o MNIST não é uma base de dados difícil de ser classificada, visto que até classificadores lineares conseguem atingir taxas de erro menores que 10%. Adicionando não-linearidade aos métodos essa taxa de erro já cai para menos de 3.6%, resultado do classificador que possui 100 núcleos RBF + classificador linear. Após isso, temos os resultados do produto de "decision stumps", uma árvode de decisão composta por um nó e 2 folhas na forma $\theta_{j,b}(x) = 1$ se $x^(j) \geq b$ e $\theta_{j,b}(x) = -1$ caso contrário, nas características de Haar extraídas das imagens (KÉGL; BUSA-FEKETE, 2009) que atingiu 0.87%, 0.56% com Virtual SVM (DECOSTE; SCHöLKOPF, 2002), 0.35% usando uma rede neural com 6 camadas (784-2500-2000-1500-1000-500-10) e 0.23%, a menor taxa de

Tabela 7.8: Tabela contendo as taxas de erro (%) alcançadas pela nossa abordagem híbrida e outros métodos na literatura na base MNIST.

Categoria	Classificador	Pré-processamento	*Data Augmentation*	Taxa de erro (%)
Classificadores Lineares	Classificadores lineares para cada par de classes	Enquadramento	Nenhum	7.6
K-NNs	K-NN com correspondência no formato do contexto	Extração das características do formato do contexto	Nenhum	0.63
Boosted Stumps	Produto de "stumps"nas características de Haar	Características de Haar	Nenhum	0.87
Classificadores Não-Lineares	1000 RBFs + classificador linear	Nenhum	Nenhum	3.6
SVMs	Virtual SVM com kernel polinomal de grau 9	Enquadramento	Nenhum	0.56
Redes Neurais	Rede com 6 camadas 784-2500-2000-1500-1000-500-10	Normalização da largura e desenclinação	Distorções elásticas	0.35
Redes Convolutivas	Ensemble de 35 redes convolutivas	Normalização da largura	Distorções elásticas	**0.23**
Abordagem híbrida	Ensemble de LS-SVMs otimizada por AG + TS	Enquadramento e escala dos pixels	Nenhum	0.31

erro disponibilizada na página, por um *ensemble* composto por 35 redes convolutivas (SCHMIDHUBER, 2012), cada uma composta por 2 camadas de convolução e 1 camada totalmente conectada (arquitetura das redes convolutivas utilizadas é mostrada na Figura 7.5). As redes convolutivas são redes profundas projetadas para trabalhar com imagens que permitem a extração das características contidas nas imagens e diminuindo a dimensionalidade com o passar das camadas (Figura 7.4 apresenta a arquitetura da LeNet-5, usada no problema do MNIST). O método híbrido atingiu 0.31% com uma arquitetura menor (*ensemble* final continha 20 LS-SVMs) que, apesar de ficado atrás no resultado, essa taxa de erro é considerada satisfatória.



A Full Convolutional Neural Network (LeNet)

Figura 7.4: Arquitetura da LeNet-5, uma rede convolutiva aplicada ao problema do MNIST. Fonte (LECUN et al., 1998)

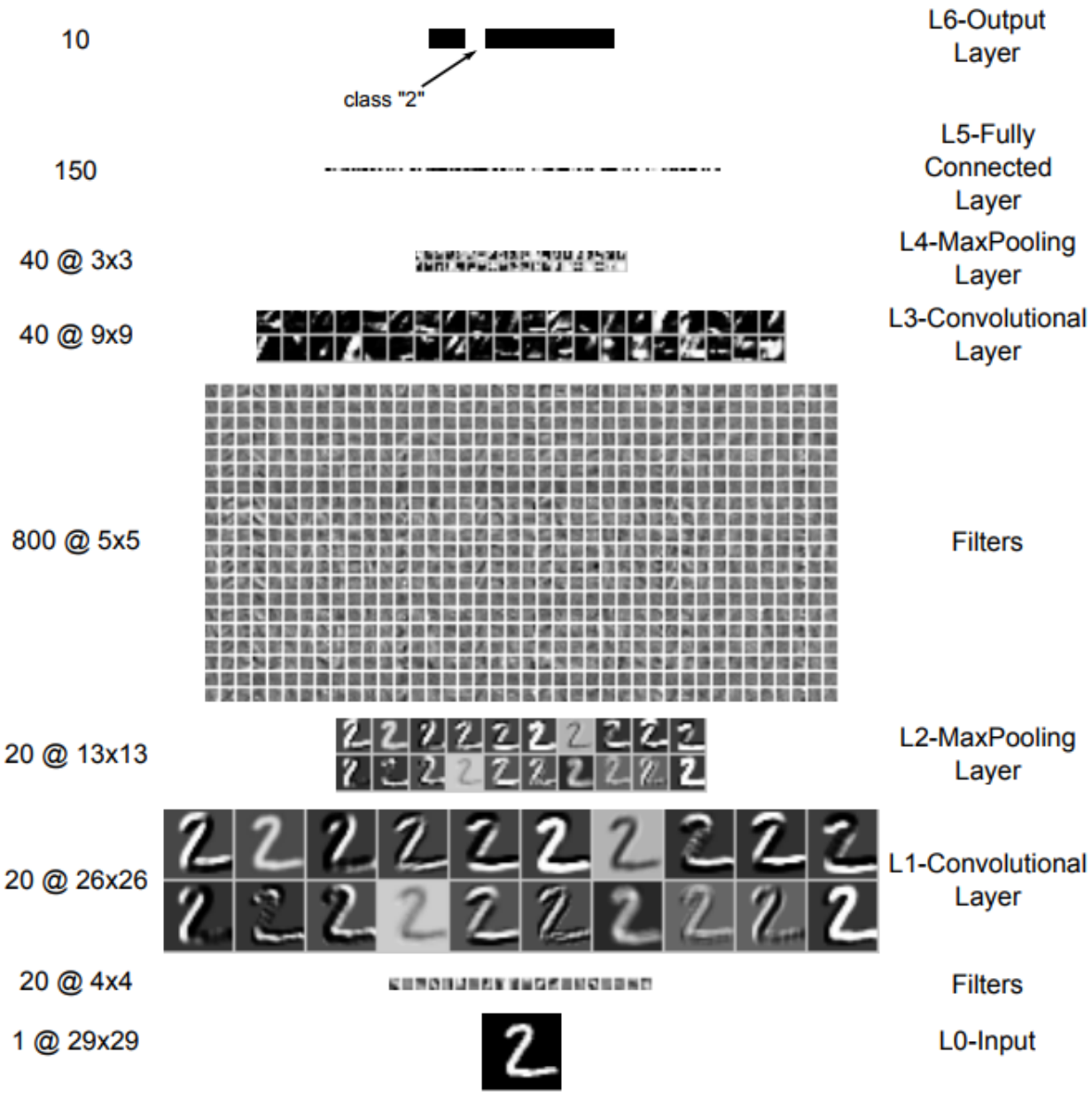


Figura 7.5: Arquitetura de cada rede convolutiva utilizada para o problema do MNIST em (SCHMIDHUBER, 2012).

De maneira similar ao MNIST, fizemos uma pesquisa bibliográfica atrás de métodos de aprendizado profundo que utilizaram a base CIFAR-10 para teste e selecionamos 5 trabalhos que reportam melhores resultados. Assim, foram selecionados os seguintes trabalhos: *Spatially-sparse convolutional neural networks* (GRAHAM, 2014), *Generalizing Pooling Functions in Convolutional Neural Networks: Mixed, Gated, and Tree* (LEE; GALLAGHER; TU, 2015), *All you need is a good init* (MISHKIN; MATAS, 2015), *Striving for Simplicity: The All Convolutional Net* (SPRINGENBERG et al., 2014) e *Fractional Max-Pooling* (GRAHAM, 2015). Todos esses trabalhos utilizam redes convolutivas e se diferenciam principalmente nas estratégias adotadas para a construção das arquiteturas e no treinamento. Não iremos nos aprofundar

nos detalhes dessas redes por fugir do escopo da tese. Como é possível observar, os resultados alcançados pela abordagem proposta não ficam tão atrás assim em relação à essas redes que foram publicadas recentemente e, portanto, acredita-se que o resultado alcançado é satisfatório no CIFAR-10, assim como aconteceu com o MNIST.

Tabela 7.9: Tabela contendo as taxas de erro (%) alcançadas pela nossa abordagem híbrida e outros métodos na literatura na base CIFAR-10.

Trabalho	Publicado em	Pré-processamento	*Data Augmentation*	Taxa de erro (%)
Spatially-sparse convolutional neural networks	arXiv 2014	Preenchimento com zeros	Transformações afins	6.28
Generalizing Pooling Functions in Convolutional Neural Networks: Mixed, Gated, and Tree	AISTATS 2016	Nenhum	Nenhum	6.05
All you need is a good init	ICLR 2016	Nenhum	Espelhamento e translações aleatórias	5.84
Striving for Simplicity: The All Convolutional Net	ICLR 2015	Nenhum	Espelhamento horizontal e translações aleatórias	7.25
Fractional Max-Pooling	arXiv 2015	Nenhum	Transformações afins e translações aleatórias nos canais de cor	3.47
Abordagem híbrida	-	Enquadramento e escala dos pixels	Nenhum	9.73

8 CONCLUSÕES E PERSPECTIVAS FUTURAS

Este trabalho apresentou uma abordagem multi-nível usando Algoritmos Genéticos (AG) aplicado a um *ensemble* de LS-SVMs. Considerando que um comitê efetivo precisa balancear a acurácia individual contra diversidade entre seus componentes, o método proposto objetiva agir sobre os níveis de um comitê, realizando uma seleção de atributos no espaço de entrada, seleção de classificadores e parametrização dos mesmos e a busca por um vetor de pesos que melhor represente a importância de cada componente para o comitê. Aqui nós adotamos as LS-SVMs como classificadores-base do ensemble, mas nossa abordagem, em tese, pode ser estendida a qualquer outro tipo de classificador.

De forma a avaliar a performance do método proposto, nós conduzimos duas baterias de teste. Primeiro, nossa abordagem foi comparada com o AdaBoost composta de SVM, *Random Forests*, nosso trabalho anterior (RSGALS-SVM) e uma SVM otimizada pelo AG. Os resultados da abordagem proposta alcançou resultados significativamente melhores na maioria dos casos, com a razão de vitórias / derrotas / empates com a proporção de aproximadamente $64\%/18\%/18\%$. Na segunda sequência, nós comparamos o método proposto com outras duas versões do mesmo, a primeira versão usando somente o conceito de diversidade ruim como função de aptidão e a segunda, adicionando o cálculo de diversidade ruim na versão original da função de aptidão. Os resultados apresentaram que a primeira versão modificada obteve piores resultados de acurácia em 13 de 14 bases em relação a versão original, enquanto que a segunda versão modificada não apresentou ganhos de performance significativos. A versão original ainda obteve vantagem significativa sobre as outras em *German*, *Ionosphere* e *Libras Movement*. Se nós compararmos a abordagem deste trabalho com o nosso anterior, onde o AG atuava sobre dois níveis do ensemble, os resultados mostraram que a extensão do AG para realizar a seleção de atributo promoveu um ganho significativo nos valores de acurácia.

Foi feita uma análise nos registros que guardamos que cada execução dos algoritmos para tentar descobrir as possíveis causas para um desempenho não muito satisfatório nas bases *Image*, *Ringnorm*, *Splice* e *Waveform* frente aos outros algoritmos. Nessa análise, é possível verificar 2 comportamentos distintos nessas quatro bases se comparado com o comportamento das outras 10 bases onde houve desempenho satisfatório. Nas bases *Image* e *Ringnorm* foi observado uma convergência prematura da abordagem genética, enquanto que nas bases *Splice* e *Waveform* o comportamento foi o oposto; o método provavelmente precisaria de bem mais gerações para convergir para um "ótimo"global do que o número máximo pré-determinado. Observando essas 2 questões, convergência prematura devido à baixa diversidade da população

e demora para atingir um "ótimo"global, nós propusemos também uma abordagem híbrida do Algoritmo Genético com uma metaheurística de busca local.

Para esta abordagem híbrida, optamos por uma estratégia de hibridização integrativa onde o Algoritmo Genético ficaria como principal técnica, enquanto que a técnica de busca local ficaria subordinada ao AG, para ser chamada quando for mais conveniente. O processo de otimização se inicia da mesma forma que na abordagem genética, com o AG buscando por áreas promissoras e, caso fique um certo número de gerações sem melhoria na solução, o método de busca local é acionado. De maneira a manter a diversidade na população, a busca local é executada em uma porcentagem da população de indivíduos escolhidos de maneira aleatória e também na melhor solução atual, que são usados como solução inicial em cada execução. Caso ocorra melhoria da solução, a busca local poderá ser acionada outra vez, caso contrário, uma vez que o AG volte a ficar um certo número de iterações sem melhoria, o processo será interrompido.

Então, nós avaliamos três metaheurísticas baseadas em solução única bem conhecidas para hibridizar com o AG: *Simulated Annealing*, Busca Tabu e *Iterated Local Search*. Nos experimentos, verificamos que entre as abordagens híbridas testadas, a Busca Tabu (GLOVER, 1986) se mostrou a melhor parceria para o AG, com 7 vitórias/2 empates/2 derrotas. Já analisando os tempos de execução de cada método, é possível observar uma diminuição no tempo de convergência do processo de otimização da abordagem proposta. Portanto, a abordagem híbrida correspondeu com as expectativas, aumentando a performance e ao mesmo tempo acelerando a convergência do método.

Por fim, de maneira a avaliar a performance da nossa abordagem frente à bases de dados bem maiores, selecionamos as bases MNIST e CIFAR-10, bem conhecidas na comunidade, e comparamos com os resultados obtidos por métodos de aprendizado profunda (LECUN et al., 1998; SCHMIDHUBER, 2012; GRAHAM, 2014; SPRINGENBERG et al., 2014; GRAHAM, 2015; LEE; GALLAGHER; TU, 2015; MISHKIN; MATAS, 2015). Normalmente são usadas redes convolutivas para classificar imagens já que são redes projetadas para processar dados bidimensionais. Os resultados da nossa abordagem se mostraram bem satisfatórios e conseguiram ficar bem próximos ao trabalhos que usam aprendizado profundo.

Como perspectivas futuras, poderíamos investigar a utilização do aprendizado semi-supervisionado no lugar do aprendizado supervisionado adotado neste trabalho. O aprendizado semi-supervisionado é um meio termo entre os aprendizados supervisionado e não-supervisionado. Além dos dados não-rotulados, é fornecido ao algoritmo alguma informação supervisionada, mas não necessariamente para todos os exemplos. Muitos problemas reais contém uma grande

quantidade de dados não rotulados porque os rótulos necessitam de anotadores humanos, dispositivos especiais ou experimentos caros e lentos (ZHU; GOLDBERG, 2009), e o aprendizado semi-supervisionado tira grande proveito da quantidade de dados disponíveis. Uma outra possibilidade seria aplicar o nosso método em algoritmos de aprendizado profunda que necessitem de um *fine-tuning* em seus parâmetros internos como, por exemplo, a rede *Long-Short Term Memory* também conhecida como LSTM, que possui uma série de parâmetros que poderiam ser otimizados via AG.

9 TRABALHOS PUBLICADOS E SUBMETIDOS

- **A Multi-level Approach Using Genetic Algorithms in a Ensemble of Least Squares Support Vector Machines**
  - Autores: Carlos Padilha, Dante Barone e Adrião Dória Neto
  - Periódico: *Knowledge-Based Systems*
  - Editora: Elsevier
  - Qualis: A2
  - Ano: 2016
- **Ensemble System based on Genetic Algorithm For Stock Market Forecasting**
  - Autores: Rafael Thomazi Gonzalez, Carlos Alberto Padilha and Dante Augusto Couto Barone
  - Conferência: IEEE CEC 2015
  - Editora: IEEE
  - Qualis: A1
- **Hybrid Genetic Algorithms Applied to an Ensemble of Least Squares Support Vector Machines (Submetido)**
  - Autores: Carlos Padilha, Dante Barone e Adrião Dória Neto
  - Periódico: *Knowledge-Based Systems*
  - Editora: Elsevier
  - Qualis: A2
  - Ano: 2017

REFERÊNCIAS

AHUJA, R. K. et al. A survey of very large-scale neighborhood search techniques. **Discrete Appl. Math.**, Elsevier Science Publishers B. V., Amsterdam, The Netherlands, The Netherlands, v. 123, n. 1-3, p. 75–102, nov. 2002. ISSN 0166-218X.

ALBA, E. **Parallel Metaheuristics: A New Class of Algorithms**. [S.l.]: Wiley-Interscience, 2005. ISBN 0471678066.

ALTMAN, N. S. An Introduction to Kernel and Nearest-Neighbor Nonparametric Regression. **The American Statistician**, American Statistical Association, v. 46, n. 3, p. 175–185, 1992. ISSN 00031305. Available from Internet: <http://dx.doi.org/10.2307/2685209>.

APPLEGATE, D.; COOK, W.; ROHE, A. Chained lin-kernighan for large traveling salesman problems. **INFORMS J. on Computing**, INFORMS, Institute for Operations Research and the Management Sciences (INFORMS), Linthicum, Maryland, USA, v. 15, n. 1, p. 82–92, jan. 2003. ISSN 1526-5528. Available from Internet: <http://dx.doi.org/10.1287/ijoc.15.1.82.15157>.

ASOH, H.; MüHLENBEIN, H. On the mean convergence time of evolutionary algorithms without selection and mutation. In: **PARALLEL PROBLEM SOLVING FROM NATURE, LECTURE NOTES IN COMPUTER SCIENCE 866**. [S.l.]: Springer-Verlag, 1994. p. 88–97.

BACAUSKIENE, M. et al. A feature selection technique for generation of classification committees and its application to categorization of laryngeal images. **Pattern Recognition**, v. 42, n. 5, p. 645–654, 2009. ISSN 00313203.

BARBOSA, H. J. C. Algoritmos genéticos para otimização em engenharia: uma introdução. In: . Juiz de Fora: IV Seminários sobre Elementos Finitos e Métodos Numéricos em Engenharia, 1996.

BATTITI, R.; TECCHIOLLI, G. The reactive tabu search. **ORSA journal on computing**, INFORMS, v. 6, n. 2, p. 126–140, 1994.

BAUM, E. B. Towards practical 'neural' computation for combinatorial optimization problems. In: **AIP Conference Proceedings**. AIP, 1986. v. 151, p. 53–58. ISSN 0094243X. Available from Internet: <http://aip.scitation.org/doi/abs/10.1063/1.36219>.

BEASLEY, D.; BULL, D. R.; MARTIN, R. R. **An Overview of Genetic Algorithms : Part 1, Fundamentals**. 1993.

BHOWAN, U. et al. Evolving Diverse Ensembles Using Genetic Programming for Classification With Unbalanced Data. **IEEE Transactions on Evolutionary Computation**, v. 17, n. 3, p. 368–386, 2013. ISSN 1089-778X.

BHOWAN, U. et al. Reusing Genetic Programming for Ensemble Selection in Classification of Unbalanced Data. **IEEE Transactions on Evolutionary Computation**, v. 18, n. 6, p. 893–908, 2014. ISSN 1089-778X.

BIRATTARI, M. et al. Classification of metaheuristics and design of experiments for the analysis of components. In: **AIDA-01-05**. [S.l.: s.n.], 2001. p. 1–12.

BLUM, C.; ROLI, A.; ALBA, E. An introduction to metaheuristic techniques. In: ____. **Parallel Metaheuristics**. [S.l.]: John Wiley & Sons Inc., 2005. p. 1–42. ISBN 9780471739388.

BOUKHAROUBA, A. A new algorithm for skew correction and baseline detection based on the randomized hough transform. **Journal of King Saud University - Computer and Information Sciences**, v. 29, n. 1, p. 29 – 38, 2017. ISSN 1319-1578. Available from Internet: <http://www.sciencedirect.com/science/article/pii/S1319157816300015>.

BOUSSAÏD, I.; LEPAGNOT, J.; SIARRY, P. A survey on optimization metaheuristics. **Information Sciences**, v. 237, p. 82–117, 2013. ISSN 00200255.

BREIMAN, L. Bagging predictors. **Machine Learning**, Springer, v. 24, n. 2, p. 123–140, 1996. ISSN 08856125.

BREIMAN, L. Random forests. **Machine Learning**, v. 45, n. 1, p. 5–32, 2001. ISSN 08856125.

BROWN, G.; KUNCHEVA, L. I. "Good" and "Bad" Diversity in Majority Vote Ensembles. p. 124–133, 2010.

BRYLL, R.; GUTIERREZ-OSUNA, R.; QUEK, F. Attribute bagging: Improving accuracy of classifier ensembles by using random feature subsets. **Pattern Recognition**, v. 36, n. 6, p. 1291–1302, 2003. ISSN 00313203.

CANUTO, A. M. P.; NASCIMENTO, D. S. C. A genetic-based approach to features selection for ensembles using a hybrid and adaptive fitness function. In: **The 2012 International Joint Conference on Neural Networks (IJCNN)**. [S.l.]: IEEE, 2012. p. 1–8. ISBN 978-1-4673-1490-9.

CAVALCANTI, G. D. et al. Combining Diversity Measures for Ensemble Pruning. **Pattern Recognition Letters**, Elsevier B.V., v. 74, p. 38–45, 2016. ISSN 01678655.

CERNY, V. Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm. **Journal of Optimization Theory and Applications**, v. 45, n. 1, p. 41–51, 1985. ISSN 1573-2878.

CHANDRA, A.; YAO, X. Ensemble learning using multi-objective evolutionary algorithms. **Journal of Mathematical Modelling and Algorithms**, Springer Netherlands, v. 5, n. 4, p. 417–445, 2006. ISSN 1570-1166.

CHEN, H.; TINO, P.; YAO, X. Predictive ensemble pruning by expectation propagation. **IEEE Transactions on Knowledge and Data Engineering**, v. 21, n. 7, p. 999–1013, 2009. ISSN 10414347.

CHERKAUER, K. Human expert-level performance on a scientific image analysis task by a system using combined artificial neural networks. **Working Notes, Integrating Multiple Learned Models for Improving and Scaling Machine Learning Algorithms Wkshp, 13th Nat Conf on Artificial Intelligence**, p. 15–21, 1996.

COELHO, A.; LIMA, C.; ZUBEN, F. V. Ga-based selection of components for heterogeneous ensembles of support vector machines. In: **Evolutionary Computation, 2003. CEC '03. The 2003 Congress on**. [S.l.: s.n.], 2003. v. 3, p. 2238–2245 Vol.3.

COTTA, C.; TALBI, E.-G.; ALBA, E. Parallel hybrid metaheuristics. In: ____. **Parallel Metaheuristics**. [S.l.]: John Wiley & Sons Inc., 2005. p. 347–370. ISBN 9780471739388.

COTTAPORRAS, C. A study of hybridisation techniques and their application to the design of evolutionary algorithms. **AI Commun.**, IOS Press, Amsterdam, The Netherlands, The Netherlands, v. 11, n. 3,4, p. 223–224, dec. 1998. ISSN 0921-7126.

COVER, T. M. Geometrical and Statistical Properties of Systems of Linear Inequalities with Applications in Pattern Recognition. **IEEE Transactions On Electronic Computers**, IEEE, EC-14, n. 3, p. 326–334, 1965. ISSN 03677508.

CRISTIANINI, N.; SHAWE-TAYLOR, J. **An Introduction to Support Vector Machines and Other Kernel-based Learning Methods**. [S.l.]: Cambridge University Press, 2000. 204 p. ISSN 15562646. ISBN 0521780195.

CUNNINGHAM, P.; CARNEY, J.; JACOB, S. Stability problems with artificial neural networks and the ensemble solution. **Artificial Intelligence in Medicine**, v. 20, p. 217–225, 2000. ISSN 09333657.

DAWKINS, R. **A Escalada do Monte Improvável: uma defesa da teoria da evolução**. São Paulo: Companhia das Letras, 1996.

DEB, K. et al. A fast and elitist multiobjective genetic algorithm: NSGA-II. **IEEE Transactions on Evolutionary Computation**, v. 6, n. 2, p. 182–197, 2002. ISSN 1089778X.

DECOSTE, D.; SCHöLKOPF, B. Training invariant support vector machines. **Mach. Learn.**, Kluwer Academic Publishers, Hingham, MA, USA, v. 46, n. 1-3, p. 161–190, mar. 2002. ISSN 0885-6125. Available from Internet: <http://dx.doi.org/10.1023/A:1012454411458>.

DIETTERICH, T. G. Ensemble Methods in Machine Learning. **Lecture Notes in Computer Science**, v. 1857, p. 1–15, 2000. ISSN 0738-4602.

DRUCKER, H.; SCHAPIRE, R.; SIMARD, P. Boosting Performance in Neural Networks. **International Journal of Pattern Recognition and Artificial Intelligence**, v. 07, n. 04, p. 705–719, 1993. ISSN 0218-0014.

EL-ABD, M.; KAMEL, M. A taxonomy of cooperative search algorithms. In: **Proceedings of the Second International Conference on Hybrid Metaheuristics**. Berlin, Heidelberg: Springer-Verlag, 2005. (HM'05), p. 32–41. ISBN 3-540-28535-0, 978-3-540-28535-9. Available from Internet: <http://dx.doi.org/10.1007/11546245_4>.

EMMANUELLA, L.; SANTANA, A. S.; CANUTO, A. M. D. P. Filter-based optimization techniques for selection of feature subsets in ensemble systems. **Expert Systems With Applications**, Elsevier Ltd, v. 41, n. 4, p. 1622–1631, 2014. ISSN 0957-4174.

ESHELMAN, L. J.; CARUANA, R. A.; SCHAFFER, J. D. Biases in the crossover landscape. In: SCHAFFER, J. D. (Ed.). **Proceedings of the Third International Conference on Genetic Algorithms**. [S.l.]: Morgan Kaufmann Publishers Inc., 1989. p. 10–19.

FACELI, K. et al. **Inteligência Artificial: Uma Abordagem de Aprendizado de Máquina**. 1. ed. [S.l.]: LTC, 2011. 394 p. ISBN 9788521618805.

FEO, T. A.; RESENDE, M. G. A probabilistic heuristic for a computationally difficult set covering problem. **Operations Research Letters**, v. 8, n. 2, p. 67–71, apr 1989. ISSN 01676377.

FLETCHER, R. **Practical Methods of Optimization**. [S.l.: s.n.], 1987. 285–286 p. ISBN 0471494631.

FRANK, A.; ASUNCION, A. **UCI Machine Learning Repository**. Irvine: University of California.

FREUND, Y.; SCHAPIRE, R. E. Experiments with a New Boosting Algorithm. **Update**, Citeseer, pages, p. 148–156, 1996.

GLOVER, F. Future paths for integer programming and links to artificial intelligence. **Comput. Oper. Res.**, Elsevier Science Ltd., Oxford, UK, UK, v. 13, n. 5, p. 533–549, may 1986. ISSN 0305-0548.

GLOVER, F. et al. Scatter search and path relinking: A tutorial on the linear arrangement problem. **Int. J. Swarm. Intell. Res.**, IGI Global, Hershey, PA, USA, v. 2, n. 2, p. 1–21, abr. 2011. ISSN 1947-9263. Available from Internet: <http://dx.doi.org/10.4018/jsir.2011040101>.

GOLDBERG, D. E. **Genetic Algorithms in Search, Optimization, and Machine Learning**. [S.l.]: Addison-Wesley, 1989. 432 p. (Artificial Intelligence). ISBN 0201157675.

GRAHAM, B. Spatially-sparse convolutional neural networks. **CoRR**, abs/1409.6070, 2014. Available from Internet: <http://arxiv.org/abs/1409.6070>.

GRAHAM, B. Fractional max-pooling. **CoRR**, abs/1412.6071, 2015. Available from Internet: <http://arxiv.org/abs/1412.6071>.

GRUAU, F.; WHITLEY, D. Adding learning to the cellular development of neural networks: Evolution and the baldwin effect. **Evol. Comput.**, MIT Press, Cambridge, MA, USA, v. 1, n. 3, p. 213–233, sep. 1993. ISSN 1063-6560. Available from Internet: <http://dx.doi.org/10.1162/evco.1993.1.3.213>.

GUERRA-SALCEDO, C.; WHITLEY, D. Genetic approach to feature selection for ensemble creation. In: **In Proc. of Genetic and Evolutionary Computation Conference**. [S.l.]: Morgan Kaufmann, 1999. p. 236–243.

GUTTA, S.; WECHSLER, H. Face recognition using hybrid classifier systems. In: **Neural Networks, 1996., IEEE International Conference on**. [S.l.: s.n.], 1996. v. 2, p. 1017 –1022 vol.2.

GUYON, I.; ELISSEEFF, A. An introduction to variable and feature selection. **Journal of Machine Learning Research**, v. 3, p. 1157–1182, 2003. ISSN 15324435.

HANSEN, L. K.; SALAMON, P. Neural network ensembles. **IEEE Transactions on Pattern Analysis and Machine**, v. 12, n. 10, p. 993–1001, 1990. ISSN 01628828.

HANSEN, N.; OSTERMEIER, A. Completely derandomized self-adaptation in evolution strategies. **Evol. Comput.**, MIT Press, Cambridge, MA, USA, v. 9, n. 2, p. 159–195, jun. 2001. ISSN 1063-6560.

HART, W. E. **Adaptive Global Optimization with Local Search**. Thesis (PhD), La Jolla, CA, USA, 1994. UMI Order No. GAX94-32928.

HASHEM, S. Optimal Linear Combinations of Neural Networks. **Neural Networks**, v. 10, n. 4, p. 599–614, 1997. ISSN 0893-6080.

HASHEM, S.; SCHMEISER, B. Improving model accuracy using optimal linear combinations of trained neural networks. **IEEE Transactions on Neural Networks**, v. 6, n. 3, p. 792–794, 1995.

HAYKIN, S. **Neural Networks: A Comprehensive Foundation**. [S.l.]: Prentice Hall, 1999. 409–412 p. ISSN 02698889. ISBN 0132733501.

HERNáNDEZ-LOBATO, D. et al. Pruning Adaptive Boosting Ensembles by Means of a Genetic Algorithm. **Proceedings of the 7th International Conference on Intelligent Data Engineering and Automated Learning**, v. 4224, p. 322–329, 2006. ISSN 03029743.

HO, T. K. The random subspace method for constructing decision forests. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 20, n. 8, p. 832–844, 1998. ISSN 01628828.

HOLLAND, J. H. **Adaptation in Natural and Artificial Systems**. [S.l.]: University of Michigan Press, 1975. 1–200? p. ISBN 0262581116.

HUANG, F. J. H. F. J. et al. Pose invariant face recognition. **Proceedings of the IEEE 2000 National Aerospace and Electronics Conference NAECON 2000 Engineering Tomorrow Cat No00CH37093**, v. 6, n. 3, p. 245–250, 2000.

INOUE, H.; NARIHISA, H. Predicting chaotic time series by ensemble self-generating neural networks. In: **Neural Networks, 2000. IJCNN 2000, Proceedings of the IEEE-INNS-ENNS International Joint Conference on**. [S.l.: s.n.], 2000. v. 2, p. 231–236 vol.2. ISSN 1098-7576.

JACOBS, R. a. et al. Adaptive Mixtures of Local Experts. **Neural Computation**, v. 3, n. 1, p. 79–87, 1991. ISSN 0899-7667.

JOHANSSON, U.; LOFSTROM, T. Producing implicit diversity in ANN ensembles. **The 2012 International Joint Conference on Neural Networks (IJCNN)**, Ieee, p. 1–8, jun. 2012.

JOHNSON, D. S. Local optimization and the traveling salesman problem. In: **Proceedings of the 17th International Colloquium on Automata, Languages and Programming**. London, UK, UK: Springer-Verlag, 1990. (ICALP '90), p. 446–461. ISBN 3-540-52826-1. Available from Internet: <http://dl.acm.org/citation.cfm?id=646244.684359>.

JONG, K. D. **Genetic algorithms: a 30 year perspective**. [S.l.]: Oxford University Press, 2005.

KAPP, M. N.; SABOURIN, R.; MAUPIN, P. Adaptive incremental learning with an ensemble of support vector machines. **Proceedings - International Conference on Pattern Recognition**, p. 4048–4051, 2010. ISSN 10514651.

KÉGL, B.; BUSA-FEKETE, R. Boosting products of base classifiers. In: **Proceedings of the 26th Annual International Conference on Machine Learning**. New York, NY, USA: ACM, 2009. (ICML '09), p. 497–504. ISBN 978-1-60558-516-1. Available from Internet: <http://doi.acm.org/10.1145/1553374.1553439>.

KENDALL, M. et al. **Kendall's Advanced Theory of Statistics:Volume 2A -Classical Inference and and the Linear Model (Kendall's Library of Statistics)**. 6. ed. A Hodder Arnold Publication, 1999. Hardcover. ISBN 0340662301. Available from Internet: <http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20&path=ASIN/0340662301>.

KIM, Y.; STREET, W.; MENCZER, F. Optimal ensemble construction via meta-evolutionary ensembles. **Expert Systems with Applications**, v. 30, n. 4, p. 705–714, may 2006. ISSN 09574174.

KIRKPATRICK, S.; GELATT, C. D.; VECCHI, M. P. Optimization by simulated annealing. **Science**, American Association for the Advancement of Science, v. 220, n. 4598, p. 671–680, 1983. ISSN 00368075.

KRASNOGOR, N.; SMITH, J. A tutorial for competent memetic algorithms: Model, taxonomy, and design issues. **Trans. Evol. Comp**, IEEE Press, Piscataway, NJ, USA, v. 9, n. 5, p. 474–488, oct. 2005. ISSN 1089-778X. Available from Internet: <http://dx.doi.org/10.1109/TEVC.2005.850260>.

KRIZHEVSKY, A. **Learning Multiple Layers of Features from Tiny Images**. [S.l.], 2009.

KUNCHEVA, L.; WHITAKER, C. Ten measures of diversity in classifier ensembles: Limits for two classifiers. **DERA/IEE Work. Intell. Sens. Process.**, Iee, v. 2001, p. 10–10, 2009. ISSN 09633308.

KUNCHEVA, L. I. **Combining Pattern Classifiers: Methods and Algorithms**. [S.l.: s.n.], 2004. 517–518 p. ISSN 0040-1706. ISBN 0-471-21078-1.

KUNCHEVA, L. I.; JAIN, L. C. Designing classifier fusion systems by genetic algorithms. **IEEE Transactions on Evolutionary Computation**, v. 4, n. 4, p. 327–336, 2000. ISSN 1089778X.

LECUN, Y. et al. Gradient-based learning applied to document recognition. **Proceedings of the IEEE**, v. 86, n. 11, p. 2278–2324, Nov 1998. ISSN 0018-9219.

LECUN, Y.; CORTES, C. MNIST handwritten digit database. 2010. Available from Internet: <http://yann.lecun.com/exdb/mnist/>.

LEE, C.-Y.; GALLAGHER, P. W.; TU, Z. Generalizing pooling functions in convolutional neural networks: Mixed, gated, and tree. **CoRR**, abs/1509.08985, 2015. Available from Internet: <http://dblp.uni-trier.de/db/journals/corr/corr1509.html#LeeGT15>.

LI, X.; WANG, L.; SUNG, E. AdaBoost with Learners Based Weak. **Electron. Eng.**, p. 196–201, 2005.

LIMA, C.; COELHO, A.; ZUBEN, F. V. Ensembles of support vector machines for regression problems. In: **Neural Networks, 2002. IJCNN '02. Proceedings of the 2002 International Joint Conference on**. [S.l.: s.n.], 2002. v. 3, p. 2381–2386. ISSN 1098-7576.

LIMA, T. P. F.; LUDERMIR, T. B. Differential evolution and meta-learning for dynamic ensemble of neural network classifiers. **Proceedings of the International Joint Conference on Neural Networks**, v. 2015-September, 2015.

LIU, Y.; YAO, X. Negatively correlated neural networks for classification. In: **Proc 9th Australian Conference on Neural Networks (ACNN'98)**. [S.l.: s.n.], 1998. p. 183–187.

LOBO, F. G.; GOLDBERG, D. E. Decision making in a hybrid genetic algorithm. In: **Evolutionary Computation, 1997., IEEE International Conference on**. [S.l.: s.n.], 1997. p. 121–125.

LOURENçO, H. R.; MARTIN, O. C.; STüTZLE, T. Iterated local search. In: **Handbook of Metaheuristics, volume 57 of International Series in Operations Research and Management Science**. [S.l.]: Kluwer Academic Publishers, 2002. p. 321–353.

LU, K.; WANG, L. A Novel Nonlinear Combination Model Based on Support Vector Machine for Rainfall Prediction. In: **2011 Fourth International Joint Conference on Computational Sciences and Optimization**. [S.l.]: IEEE, 2011. p. 1343–1346. ISBN 978-1-4244-9712-6.

LUCAS, D. C. **Algoritmos Genéticos: uma Introdução**. [S.l.]: UFRGS, 2002.

MAO, J. A case study on bagging, boosting and basic ensembles of neural networks for ocr. In: **Neural Networks Proceedings, 1998. IEEE World Congress on Computational Intelligence. The 1998 IEEE International Joint Conference on**. [S.l.: s.n.], 1998. v. 3, p. 1828–1833 vol.3. ISSN 1098-7576.

MARTIN, O.; OTTO, S. W.; FELTEN, E. W. Large-step markov chains for the tsp incorporating local search heuristics. **Oper. Res. Lett.**, Elsevier Science Publishers B. V., Amsterdam, The Netherlands, The Netherlands, v. 11, n. 4, p. 219–224, may 1992. ISSN 0167-6377. Available from Internet: <http://dx.doi.org/10.1016/0167-6377(92)90028-2>.

MARTIN, O. C.; OTTO, S. W. **Combining Simulated Annealing with Local Search Heuristics**. [S.l.], 1993.

MERCER, J. Functions of Positive and Negative Type, and their Connection with the Theory of Integral Equations. **Philosophical Transactions of the Royal Society A Mathematical Physical and Engineering Sciences**, The Royal Society, v. 209, n. 441-458, p. 415–446, 1909. ISSN 1364503X.

METROPOLIS, N. et al. Equation of state calculations by fast computing machines. **Journal of Chemical Physics**, v. 21, p. 1087–1092, 1953.

MINKA, T. P. Expectation Propagation for approximate Bayesian inference. **Uncertainty in Artificial Intelligence, Proc. of (UAI)**, p. 362–369, 2001.

MINSKY, M. L.; PAPERT, S. A. **Perceptrons: expanded edition**. [S.l.]: The MIT Press. 20 p. ISBN 0262631113.

MISHKIN, D.; MATAS, J. All you need is a good init. **CoRR**, abs/1511.06422, 2015. Available from Internet: <http://arxiv.org/abs/1511.06422>.

MOUSAVI, R.; EFTEKHARI, M. A new ensemble learning methodology based on hybridization of classifier ensemble selection approaches. **Applied Soft Computing**, Elsevier B.V., v. 37, p. 652–666, 2015. ISSN 15684946.

NASCIMENTO, D. S. C. et al. Combining different ways to generate diversity in bagging models: An evolutionary approach. **2011 Int. Jt. Conf. Neural Networks**, IEEE, p. 2235–2242, jul. 2011.

NETO, A. F.; CANUTO, A. M. P. An exploratory study of mono and multi-objective metaheuristics to ensemble of classifiers. **Applied Intelligence**, p. 1–16, 2017.

NILSSON, N. J. **Learning machines: foundations of trainable pattern-classifying systems**. [S.l.]: McGraw-Hill, 1965. xi, 137 p. p. ISBN 0070465703.

OPITZ, D. Feature selection for ensembles. In: **Proceedings of the National Conference on Artificial Intelligence**. [S.l.: s.n.], 1999. p. 379–384. ISBN 0262511061. ISSN 09333657.

OPITZ, D. W. Feature selection for ensembles. In: **In Proceedings of 16th National Conference on Artificial Intelligence (AAAI**. [S.l.]: Press, 1999. p. 379–384.

PADILHA, C. et al. An genetic approach to Support Vector Machines in classification problems. In: **2010 Int. Jt. Conf. Neural Networks**. [S.l.]: IEEE, 2010. p. 1–4. ISBN 978-1-4244-6916-1.

PADILHA, C.; NETO, A. a. D. D.; MELO, J. D. RSGALS-SVM: Random subspace method applied to a LS-SVM ensemble optimized by genetic algorithm. In: **Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)**. [S.l.: s.n.], 2012. v. 7435 LNCS, p. 253–260. ISBN 9783642326387. ISSN 03029743.

PADILHA, C. A. de A.; BARONE, D. A. C.; NETO, A. D. D. A multi-level approach using genetic algorithms in an ensemble of least squares support vector machines. **Knowledge-Based Systems**, v. 106, p. 85 – 95, 2016. ISSN 0950-7051.

PERRONE, M. P.; COOPER, L. N. When networks disagree: Ensemble methods for hybrid neural networks. **Artificial Neural Networks for Speech and Vision**, Chapman & Hall, n. 4, p. 126–142, 1993.

PRAIS, M. et al. Reactive grasp: An application to a matrix decomposition problem in tdma traffic assignment. **INFORMS Journal on Computing**, v. 12, p. 164–176, 1998.

PREUX, P.; TALBI, E.-G. Towards hybrid evolutionary algorithms. **International Transactions in Operational Research**, Blackwell Publishing Ltd, v. 6, n. 6, p. 557–570, 1999. ISSN 1475-3995. Available from Internet: <http://dx.doi.org/10.1111/j.1475-3995.1999.tb00173.x>.

PUCHINGER, J.; RAIDL, G. R. Combining metaheuristics and exact algorithms in combinatorial optimization: A survey and classification. In: **Proceedings of the First International Work-conference on the Interplay Between Natural and Artificial Computation Conference on Artificial Intelligence and Knowledge Engineering Applications: A Bioinspired Approach - Volume Part II**. Berlin, Heidelberg: Springer-Verlag, 2005. (IWINAC'05), p. 41–53. ISBN 3-540-26319-5, 978-3-540-26319-7. Available from Internet: <http://dx.doi.org/10.1007/11499305_5>.

RAIDL, G. R. A unified view on hybrid metaheuristics. **Hybrid metaheuristics**, Springer, v. 4030, p. 1–12, 2006.

RäTSCH, G.; ONODA, T.; MüLLER, K. R. Soft margins for AdaBoost. **Mach. Learn.**, v. 42, p. 287–320, 2001. ISSN 08856125.

REEVES, C. R. Genetic algorithms and neighbourhood search. In: ____. **Evolutionary Computing: AISB Workshop Leeds, U.K., April 11–13, 1994 Selected Papers**. Berlin, Heidelberg: Springer Berlin Heidelberg, 1994. p. 115–130. ISBN 978-3-540-48999-3.

RESENDE, M. G. C.; RIBEIRO, C. C. Greedy randomized adaptive search procedures. In: ____. **Handbook of Metaheuristics**. Boston, MA: Springer US, 2003. p. 219–249. ISBN 978-0-306-48056-0.

SCHAPIRE, R. E. The strength of weak learnability. **Machine Learning**, v. 5, n. 2, p. 197–227, 1990. ISSN 08856125.

SCHAPIRE, R. E. A brief introduction to boosting. In: **IJCAI Int. Jt. Conf. Artif. Intell.** [S.l.: s.n.], 1999. v. 2, p. 1401–1406. ISBN 3540440119. ISSN 10450823.

SCHMIDHUBER, J. Multi-column deep neural networks for image classification. In: **Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. Washington, DC, USA: IEEE Computer Society, 2012. (CVPR '12), p. 3642–3649. ISBN 978-1-4673-1226-4. Available from Internet: <http://dl.acm.org/citation.cfm?id=2354409.2354694>.

SCHÖLKOPF, B. et al. New Support Vector Algorithms. **Neural Computation**, v. 12, n. 5, p. 1207–1245, 2000. ISSN 0899-7667.

SHARKEY, A. Multi-Net Systems. In: **Combining artificial neural nets: ensemble and modular multi-net systems**. [S.l.: s.n.], 1998. ISBN 978-1852330040.

SHIMSHONI, Y.; INTRATOR, N. Classification of seismic waveforms by integrating ensembles of neural networks. In: **Neural Networks for Signal Processing VI. Proceedings of the 1996 IEEE Signal Processing Society Workshop**. [S.l.: s.n.], 1996. p. 453–462. ISBN 0-7803-3550-3. ISSN 1089-3555.

SIEDLECKI, W.; SKLANSKY, J. A note on genetic algorithms for large-scale feature selection. **Pattern Recognition Letters**, v. 10, n. 5, p. 335–347, nov. 1989. ISSN 01678655.

SIGILLITO, V. G. et al. Classification of radar returns from the ionosphere using neural networks. **Johns Hopkins APL Technical Digest**, v. 10, p. 262–266, 1989.

SIVANANDAM, S. N.; DEEPA, S. N. **Introduction to Genetic Algorithms**. [S.l.: s.n.], 2008. 442 p. ISBN 9783540731894.

SOUZA, M. J. F. Inteligência Computacional para Otimizaçãoo 2 Heurísticas Construtivas. p. 1–28, 2005.

SPRINGENBERG, J. T. et al. Striving for simplicity: The all convolutional net. **CoRR**, abs/1412.6806, 2014. Available from Internet: <http://arxiv.org/abs/1412.6806>.

SUMAN, B.; KUMAR, P. A survey of simulated annealing as a tool for single and multiobjective optimization. **Journal of the Operational Research Society**, v. 57, n. 10, p. 1143–1160, 2006. ISSN 1476-9360.

SUYKENS, J. A. K.; VANDEWALLE, V. Least-Squares Support Vector Machine Classifiers. **Neural Process. Lett.**, v. 9, n. 3, p. 293–300, 1999. ISSN 13704621.

TALBI, E.-G. A taxonomy of hybrid metaheuristics. **Journal of Heuristics**, Kluwer Academic Publishers, Hingham, MA, USA, v. 8, n. 5, p. 541–564, sep. 2002. ISSN 1381-1231.

TAN, C. **Simulated Annealing**. 1st. ed. [S.l.]: IN-TECH Education and Publishing, 2008.

TAN, P.-N.; STEINBACH, M.; KUMAR, V. **Introduction to Data Mining, (First Edition)**. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2005. ISBN 0321321367.

THIERENS, D. et al. **Domino Convergence, Drift, and the Temporal-Salience Structure of Problems**. 1998.

THOMPSON, J.; DOWSLAND, K. A. General cooling schedules for a simulated annealing based timetabling system. In: ____. **Practice and Theory of Automated Timetabling: First International Conference Edinburgh, U.K., August 29–September 1, 1995 Selected Papers**. Berlin, Heidelberg: Springer Berlin Heidelberg, 1996. p. 345–363. ISBN 978-3-540-70682-3.

TSYMBAL, A.; PUURONEN, S.; PATTERSON, D. W. Ensemble feature selection with the simple Bayesian classification. **Information Fusion**, v. 4, n. 2, p. 87–100, 2003. ISSN 15662535.

UEDA, N. Optimal linear combination of neural networks for improving classification performance. **Pattern Analysis and Machine Intelligence, IEEE Transactions on**, v. 22, n. 2, p. 207–215, 2000. ISSN 01628828.

VAFAIE, H.; JONG, K. D. Genetic algorithms as a tool for feature selection in machine learning. In: . [S.l.]: Society Press, 1992. p. 200–204.

VALENTINI, G.; DIPARTIMENTO, D. S. I.; DIETTERICH, T. G. Bias-Variance Analysis of Support Vector Machines for the Development of SVM-Based Ensemble Methods. **J. Mach. Learn. Res.**, JMLR. org, v. 5, p. 725–775, 2004. ISSN 15324435.

VAPNIK, V. N. **The Nature of Statistical Learning Theory**. [S.l.: s.n.], 1995. 188 p. ISSN 10459227. ISBN 0387945598.

VAPNIK, V. N. **Statistical Learning Theory**. [S.l.]: Wiley-Interscience, 1998. 736 p. (Wiley Series on Adaptive and Learning Systems for Signal Processing, Communications and Control, 4). ISSN 10762787. ISBN 0471030031.

WANG, X.; TANG, X. Random sampling for subspace face recognition. **International Journal of Computer Vision**, v. 70, n. 1, p. 91–104, 2006. ISSN 09205691.

WEARE, R. F.; BURKE, E. K.; ELLIMAN, D. G. A hybrid genetic algorithm for highly constrained timetabling problems. **Science**, Morgan Kaufmann, p. 605–610.

WICHARD, J. D.; OGORZAŁEK, M. Time series prediction with ensemble models. **IEEE International Conference on Neural Networks - Conference Proceedings**, v. 2, n. 4, p. 1625–1630, 2004. ISSN 10987576.

WOLPERT, D. H. Stacked generalization. **Neural Networks**, v. 5, n. 2, p. 241–259, 1992. ISSN 08936080.

WOODS, K.; KEGELMEYER, W. P.; BOWYER, K. Combination of multiple classifiers using local accuracy estimates. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 19, n. 4, p. 405–410, 1997. ISSN 0162-8828.

XIE, J. X. J. **Kernel optimization of LS-SVM based on damage detection for smart structures**. 2009.

YANG, F. Y. F. **Fast optimizing parameters algorithm for least squares support vector machine based on artificial immune algorithm**. 2009.

YANG, L. Y. L.; WANG, H. T. W. H. T. **Classification Based on Particle Swarm Optimization for Least Square Support Vector Machines Training**. [S.l.]: Ieee, 2010. 246–249 p.

ZHANG, W.; NIU, P. **LS-SVM based on Chaotic Particle Swarm Optimization with simulated annealing and application**. [S.l.]: IEEE, 2011. 931–935 p.

ZHOU, Z. H.; WU, J.; TANG, W. Ensembling neural networks: Many could be better than all. **Artificial Intelligence**, v. 137, n. 1-2, p. 239–263, 2002. ISSN 00043702.

ZHU, X.; GOLDBERG, A. B. Introduction to Semi-Supervised Learning. In: **Synthesis Lectures on Artificial Intelligence and Machine Learning**. [S.l.: s.n.], 2009. v. 3, n. 1, p. 1–130. ISBN 9781598295474.