Abordagens para combinar classificadores e agrupadores em problemas de classificação

Luiz Fernando Sommaggio Coletta

SERVIÇO DE PÓS-GRADUAÇÃO DO ICMC-USP

Data de Depósito:

Assinatura: _____

Luiz Fernando Sommaggio Coletta

Abordagens para combinar classificadores e agrupadores em problemas de classificação

Tese apresentada ao Instituto de Ciências Matemáticas e de Computação - ICMC-USP, como parte dos requisitos para obtenção do título de Doutor em Ciências - Ciências de Computação e Matemática Computacional. *VERSÃO REVISADA*

Área de Concentração: Ciências de Computação e Matemática Computacional

Orientador: Prof. Dr. Eduardo Raul Hruschka

Coorientador: Prof. Dr. Moacir Antonelli Ponti

USP – São Carlos
Janeiro de 2016

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados fornecidos pelo(a) autor(a)

C694a Coletta, Luiz Fernando Sommaggio
Abordagens para combinar classificadores e
agrupadores em problemas de classificação / Luiz
Fernando Sommaggio Coletta; orientador Eduardo Raul
Hruschka; co-orientador Moacir Ponti. -- São
Carlos, 2016.
149 p.

Tese (Doutorado - Programa de Pós-Graduação em
Ciências de Computação e Matemática Computacional) --
Instituto de Ciências Matemáticas e de Computação,
Universidade de São Paulo, 2016.

1. Aprendizado de Máquina. 2. Algoritmos
Evolutivos. 3. Classificadores. 4. Agrupadores. 5.
Mineração de Dados. I. Hruschka, Eduardo Raul,
orient. II. Ponti, Moacir, co-orient. III. Título.

Luiz Fernando Sommaggio Coletta

**Approaches for combining classifiers and clusterers in
classification problems**

Doctoral dissertation submitted to the Instituto de
Ciências Matemáticas e de Computação - ICMC-
USP, in partial fulfillment of the requirements for the
degree of the Doctorate Program in Computer
Science and Computational Mathematics. *FINAL
VERSION*

Concentration Area: Computer Science and
Computational Mathematics

Advisor: Prof. Dr. Eduardo Raul Hruschka
Coadvisor: Prof. Dr. Moacir Antonelli Ponti

**USP – São Carlos
January 2016**

Aos meus pais

AGRADECIMENTOS

Gostaria de agradecer primeiramente ao meu orientador, Prof. Eduardo Raul Hruschka, que com muita inteligência e bom humor me conduziu pacientemente na longa jornada da pós-graduação. Além de ser um excepcional exemplo de professor e orientador, é um grande amigo que admiro muito.

Gostaria de expressar também meus sinceros agradecimentos ao meu coorientador, Prof. Moacir Ponti, e aos colegas de longa distância, Prof. Joydeep Ghosh e Ayan Acharya, pelo valioso auxílio e colaboração neste trabalho. Além do Prof. Estevam Hruschka Jr. e a colega de laboratório Nádia Félix da Silva, pessoas que também estiveram comigo nesta empreitada.

Sou muito grato também aos meus pais, Maria e Edson, pelo apoio incondicional e conforto durante todos esses anos de estudo. Expresso gratidão também pelo reconhecimento, apoio e preocupação de meus avós e meu irmão, além de toda a minha família.

Agradeço especialmente à minha esposa, Sarah, por iluminar ainda mais meus dias e por ter sido paciente e compreensiva ao longo destes anos. Com dedicação e simpatia, ela tem estado sempre ao meu lado, tanto nos momentos bons quanto nos difíceis.

Gostaria de deixar meu reconhecimento também aos companheiros de trabalho que de alguma forma colaboraram com esta tese. Em especial, aos colegas André Rossi, Bruno Feres, Daniel Bonetti, Danilo Horta, Eduardo Fraccaroli, Fernando Zuher, Jonathan Andrade, Lucas Vendramin, Luís Paulo, Marcelo Oliveira, Murillo Carneiro, Rafael Mantovani, Rafael Rossi, Ricardo Cerri, Ricardo Marcacini, Rodrigo Calvo, Thiago Covões, Valéria Carvalho, Victor Barella e Vinícius Siqueira. Foram longas conversas técnicas, não técnicas e reflexões sobre a vida, as quais certamente irei carregar comigo.

A todos os funcionários do ICMC/USP, pela dedicação e competência.

À CAPES (Processo DS-5713814/D) e FAPESP (Processo 2010/20830-0) pelo apoio financeiro para a realização deste trabalho.

*São todos maus descobridores,
os que pensam que não há terra
quando conseguem ver apenas o mar.
– Francis Bacon*

RESUMO

COLETTA, L. F. S.. **Abordagens para combinar classificadores e agrupadores em problemas de classificação**. 2015. 149 f. Tese (Doutorado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação (ICMC/USP), São Carlos – SP.

Modelos para aprendizado não supervisionado podem fornecer restrições complementares úteis para melhorar a capacidade de generalização de classificadores. Baseando-se nessa premissa, um algoritmo existente, denominado de C^3E (*Consensus between Classification and Clustering Ensembles*), recebe como entradas estimativas de distribuições de probabilidades de classes para objetos de um conjunto alvo, bem como uma matriz de similaridades entre esses objetos. Tal matriz é tipicamente construída por agregadores de agrupadores de dados, enquanto que as distribuições de probabilidades de classes são obtidas por um agregador de classificadores induzidos por um conjunto de treinamento. Como resultado, o C^3E fornece estimativas refinadas das distribuições de probabilidades de classes como uma forma de consenso entre classificadores e agrupadores. A ideia subjacente é de que objetos similares são mais propensos a compartilhar o mesmo rótulo de classe. Nesta tese, uma versão mais simples do algoritmo C^3E , baseada em uma função de perda quadrática (C^3E -SL), foi investigada em uma abordagem que permitiu a estimação automática (a partir dos dados) de seus parâmetros críticos. Tal abordagem faz uso de uma nova estratégia evolutiva concebida especialmente para tornar o C^3E -SL mais prático e flexível, abrindo caminho para que variantes do algoritmo pudessem ser desenvolvidas. Em particular, para lidar com a escassez de dados rotulados, um novo algoritmo que realiza aprendizado semissupervisionado foi proposto. Seu mecanismo explora estruturas intrínsecas dos dados a partir do C^3E -SL em um procedimento de autotreinamento (*self-training*). Esta noção também inspirou a concepção de um outro algoritmo baseado em aprendizado ativo (*active learning*), o qual é capaz de se adaptar para aprender novas classes que possam surgir durante a predição de novos dados. Uma extensa análise experimental, focada em problemas do mundo real, mostrou que os algoritmos propostos são bastante úteis e promissores. A combinação de classificadores e agrupadores resultou em modelos de classificação com grande potencial prático e que são menos dependentes do usuário ou do especialista de domínio. Os resultados alcançados foram tipicamente melhores em comparação com os obtidos por classificadores tradicionalmente usados.

Palavras-chave: Combinação de Classificadores e Agrupadores, Algoritmos Evolutivos, Aprendizado Semissupervisionado, Autotreinamento, Aprendizado Ativo.

ABSTRACT

COLETTA, L. F. S.. **Abordagens para combinar classificadores e agrupadores em problemas de classificação**. 2015. 149 f. Tese (Doutorado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação (ICMC/USP), São Carlos – SP.

Unsupervised learning models can provide a variety of supplementary constraints to improve the generalization capability of classifiers. Based on this assumption, an existing algorithm, named C^3E (from Consensus between Classification and Clustering Ensembles), receives as inputs class probability distribution estimates for objects in a target set as well as a similarity matrix. Such a similarity matrix is typically built from clusterers induced on the target set, whereas the class probability distributions are obtained by an ensemble of classifiers induced from a training set. As a result, C^3E provides refined estimates of the class probability distributions, from the consensus between classifiers and clusterers. The underlying idea is that similar new objects in the target set are more likely to share the same class label. In this thesis, a simpler version of the C^3E algorithm, based on a Squared Loss function (C^3E -SL), was investigated from an approach that enables the automatic estimation (from data) of its critical parameters. This approach uses a new evolutionary strategy designed to make C^3E -SL more practical and flexible, making room for the development of variants of the algorithm. To address the scarcity of labeled data, a new algorithm that performs semi-supervised learning was proposed. Its mechanism exploits the intrinsic structure of the data by using the C^3E -SL algorithm in a self-training procedure. Such a notion inspired the development of another algorithm based on active learning, which is able to self-adapt to learn new classes that may emerge when classifying new data. An extensive experimental analysis, focused on real-world problems, showed that the proposed algorithms are quite useful and promising. The combination of supervised and unsupervised learning yielded classifiers of great practical value and that are less dependent on user-defined parameters. The achieved results were typically better than those obtained by traditional classifiers.

Key-words: Combining Classifiers and Clusterers, Evolutionary Algorithms, Semi-supervised Learning, Self-training, Active Learning.

LISTA DE ILUSTRAÇÕES

Figura 1 – Visão geral do algoritmo C ³ E (ACHARYA <i>et al.</i> , 2011).	42
Figura 2 – Acurácias de classificação (%) do C ³ E a partir da combinação de diferentes valores dos parâmetros λ e α respectivamente para as bases de dados <i>News5</i> (a), <i>Cora4</i> (b) e <i>Pima</i> (c).	50
Figura 3 – Acurácias médias (%) de acordo com o número de iterações (I) do algoritmo C ³ E respectivamente para as bases de dados <i>News1</i> (a), <i>DBLP</i> (b), <i>Cora4</i> (c) e <i>Spam</i> (d).	51
Figura 4 – Acurácias de classificação (%) do algoritmo C ³ E-SL ao se variar os valores dos parâmetros α e I na base de dados <i>Wine Red Quality</i> (FRANK; ASUNCION, 2010).	60
Figura 5 – Esquema de calibração do algoritmo C ³ E-SL via validação cruzada. Os experimentos realizados assumiram que o conjunto alvo foi formado por 20% dos dados — os 80% restantes foram divididos em duas partes iguais dando origem aos conjuntos de treinamento e de validação.	61
Figura 6 – Valores mínimos, máximos e médios para α^* (CMA-ES).	67
Figura 7 – Valores mínimos, máximos e médios para α^* (DE).	67
Figura 8 – Valores mínimos, máximos e médios para I^* (CMA-ES).	68
Figura 9 – Valores mínimos, máximos e médios para I^* (DE).	68
Figura 10 – Proporções de acurácias maiores ou iguais do D ² E sobre o DE a partir de 21 diferentes configurações dos parâmetros de controle (F e Cr).	73
Figura 11 – Acurácias médias com desvios padrões do DE e D ² E a partir de 21 diferentes configurações dos parâmetros de controle (F e Cr).	73
Figura 12 – Classificação incorreta média – <i>Wisconsin</i>	75
Figura 13 – Valores de F – <i>Wisconsin</i>	75
Figura 14 – Valores de Cr – <i>Wisconsin</i>	75
Figura 15 – Classificação incorreta média – <i>Blood</i>	76
Figura 16 – Valores de F – <i>Blood</i>	76
Figura 17 – Valores de Cr – <i>Blood</i>	76
Figura 18 – Calibração do C ³ E-SL — parâmetros α e I são estimados em um conjunto de validação por meio do algoritmo D ² E (COLETTA <i>et al.</i> , 2015a).	86
Figura 19 – Os valores “ótimos” para α e I são fixados e usados no C ³ E-SL para refinar resultados do SVM em um conjunto alvo (com dados não rotulados que se deseja prever).	86

Figura 20 – Esquema do C ³ E-SL semissupervisionado.	86
Figura 21 – Espaço bidimensional formado pela contagem dos léxicos positivos e negativos presentes nos <i>tweets</i> da Tabela 12. Em (a) são apresentadas as classes reais destes <i>tweets</i> . Em (b) e (c) são apresentadas as classificações com o uso do SVM (independente e como componente na abordagem de <i>self-training</i>). Em (d), por fim, é apresentada a predição feita pela abordagem do C ³ E-SL semissupervisionado.	87
Figura 22 – $\overline{F1}$ (%) para diferentes percentuais de dados rotulados — <i>LiveJournal</i>	94
Figura 23 – $\overline{F1}$ (%) para diferentes percentuais de dados rotulados — <i>SMS2013</i>	94
Figura 24 – $\overline{F1}$ (%) para diferentes percentuais de dados rotulados — <i>Twitter2013</i>	94
Figura 25 – $\overline{F1}$ (%) para diferentes percentuais de dados rotulados — <i>Twitter2014</i>	95
Figura 26 – $\overline{F1}$ (%) para diferentes percentuais de dados rotulados — <i>TwitterSarcasm2014</i>	95
Figura 27 – Um classificador de imagens hipotético e ideal treinado para reconhecer apenas cavalos, pessoas e elefantes comete erros significativos na presença de uma nova classe, a qual representa dinossauros.	100
Figura 28 – Imagens da base de dados pedagógica usada. Apenas exemplos de patos e gatos estavam disponíveis na base de treinamento para construir o classificador. O IC-EDS deveria, então, ser capaz de descobrir a nova classe “ <i>Piggy</i> ” presente inicialmente apenas nos conjuntos alvo.	106
Figura 29 – Proporções (por classe) das imagens selecionadas nas três primeiras iterações do algoritmo IC-EDS.	106
Figura 30 – A matriz de similaridades gerada pelos agrupamentos identificou três grupos (regiões mais escuras) e ajudou no refinamento das classificações do SVM, bem como na detecção da nova classe (“ <i>Piggy</i> ”). Os grupos a partir da esquerda inferior até a direita superior englobam as instâncias, respectivamente, das classes “ <i>Duck</i> ”, “ <i>Cat</i> ” e “ <i>Piggy</i> ”.	107
Figura 31 – Instâncias de cada classe das três bases de dados usadas nos experimentos.	110
Figura 32 – Proporções, por classe, de instâncias selecionadas após 10 iterações quando a classe 13 (“ <i>Piggy</i> ”) foi intencionalmente omitida com o intuito de ser descoberta pelos algoritmos.	111
Figura 33 – Erro da classe 13, “ <i>Piggy</i> ”.	111
Figura 34 – Erro da classe 2, “ <i>Wooden Part I</i> ”.	111
Figura 35 – Erro da classe 4, “ <i>Cat</i> ”.	112
Figura 36 – Número de instâncias rotuladas para alcançar uma certa taxa de erro para a classe omitida “ <i>Wooden Part I</i> ”.	113
Figura 37 – Número de instâncias rotuladas para alcançar uma certa taxa de erro para a classe omitida “ <i>Cat</i> ”.	113
Figura 38 – F-scores globais obtidos por cada algoritmo na base de dados <i>Coil-20</i>	113

Figura 39 – Proporções, por classe, de instâncias selecionadas após 10 iterações quando a classe 10 (“ <i>Food</i> ”) foi intencionalmente omitida com o intuito de ser descoberta pelos algoritmos.	114
Figura 40 – Erro da classe 10, “ <i>Food</i> ”.	115
Figura 41 – Erro da classe 3, “ <i>Buildings</i> ”.	115
Figura 42 – Número de instâncias rotuladas para alcançar uma certa taxa de erro para a classe omitida “ <i>Food</i> ”.	115
Figura 43 – Número de instâncias rotuladas para alcançar uma certa taxa de erro para a classe omitida “ <i>Buildings</i> ”.	115
Figura 44 – F-scores globais obtidos por cada algoritmo na base de dados <i>Corel-1000</i>	116
Figura 45 – Proporções, por classe, de instâncias selecionadas após 10 iterações quando a classe 9 (“ <i>Nectarine</i> ”) foi intencionalmente omitida com o intuito de ser descoberta pelos algoritmos.	117
Figura 46 – Erro da classe 9, “ <i>Nectarine</i> ”.	117
Figura 47 – F-scores globais para a base de dados <i>Supermarket Produce</i>	117
Figura 48 – Número de instâncias rotuladas para alcançar uma certa taxa de erro para a classe omitida “ <i>Nectarine</i> ”.	118
Figura 49 – Número de instâncias rotuladas para alcançar uma certa taxa de erro para a classe omitida “ <i>Watermelon</i> ”.	118

LISTA DE ALGORITMOS

Algoritmo 1 – C ³ E (ACHARYA <i>et al.</i> , 2011)	46
Algoritmo 2 – C ³ E com função de Perda Quadrática (C ³ E-SL)	48
Algoritmo 3 – Abordagem para a calibração do C ³ E-SL	63
Algoritmo 4 – Seleção baseada em entropia e densidade (EDS)	104
Algoritmo 5 – Classificador iterativo (IC) para aprendizagem de novas classes	105

LISTA DE TABELAS

Tabela 1	– Acurácias de classificação do agregador de classificadores (NB+J48+IB1) e dos algoritmos C ³ E e C ³ E-SL. Os melhores resultados estão em negrito.	52
Tabela 2	– Acurácias médias do agregador de classificadores (NB+J48) e as diferentes versões do C ³ E. Os melhores resultados estão em negrito.	53
Tabela 3	– Acurácias médias do agregador de classificadores (NB+IB5) e as diferentes versões do C ³ E. Os melhores resultados estão em negrito.	53
Tabela 4	– Principais características das bases de dados usadas.	63
Tabela 5	– Acurácias (%) do agregador de classificadores (NB+DT) e dos algoritmos C ³ E e C ³ E-SL (desvios padrões estão entre parênteses) — os melhores resultados estão destacados em negrito.	64
Tabela 6	– Acurácias (%) do C ³ E-SL com parâmetros otimizados pelas meta-heurísticas estudadas — desvios padrões estão entre parênteses. Os melhores resultados entre todas as meta-heurísticas estão em negrito. Note que as meta-heurísticas rodaram em apenas 50% do tempo requerido pela busca em grade.	65
Tabela 7	– Classificação das meta-heurísticas de acordo com a soma dos ranks do teste de Friedman computado a partir das acurácias reportadas na Tabela 6.	66
Tabela 8	– Acurácias (%) do C ³ E-SL otimizado pelo DE a partir de diferentes combinações de F e Cr (os quais permaneceram fixos durante todo o processo). Os melhores resultados estão destacados em negrito — LIT refere-se aos valores de parâmetros sugeridos pela literatura (PRICE, 1996; STORN; PRICE, 1997), i.e., $F = 0.5$ e $Cr = 0.9$	71
Tabela 9	– Acurácias (%) do C ³ E-SL otimizado pelo D ² E com diferentes combinações de valores iniciais de F e Cr (os quais foram dinamicamente atualizados). Os melhores resultados estão destacados em negrito — LIT refere-se aos valores de parâmetros sugeridos pela literatura (PRICE, 1996; STORN; PRICE, 1997), i.e., $F = 0.5$ e $Cr = 0.9$. A última linha apresenta o número de vezes em que o D ² E foi melhor/igual ao DE a partir das acurácias da Tabela 8.	72
Tabela 10	– Desvios padrões do DE e D ² E a partir das 21 combinações distintas de valores dos parâmetros de controle (F e Cr). Menores valores estão destacados em negrito.	74

Tabela 11 – Acurácias (%) do C ³ E-SL otimizado pelas variantes do DE (desvios padrões entre parênteses). Vitórias/Empates/Derrotas denota quantos valores alcançados pelo D ² E são superiores, iguais ou inferiores ao algoritmo da coluna correspondente — os melhores resultados estão destacados em negrito. . . .	77
Tabela 12 – Exemplo de <i>tweets</i> da base de dados <i>Twitter2014</i> (ROSENTHAL <i>et al.</i> , 2014).	87
Tabela 13 – Valores de similaridades entre <i>tweets</i> contidos na matriz de similaridades usada pelo C ³ E-SL. Em destaque estão os valores máximos calculados para os <i>tweets</i> t_1 , t_{20} e t_{1853}	88
Tabela 14 – Distribuição das classes <i>positiva</i> , <i>negativa</i> e <i>neutra</i> nas seis bases de dados utilizadas. Os classificadores foram treinados com <i>tweets</i> rotulados da base de dados <i>SemEval2013</i> (NAKOV <i>et al.</i> , 2013). Os resultados reportados foram obtidos a partir de classificações em cinco conjuntos de teste — <i>LiveJournal</i> , <i>SMS2013</i> , <i>Twitter2013</i> , <i>Twitter2014</i> e <i>TwitterSarcasm2014</i>	89
Tabela 15 – F-scores (%) obtidos pelas abordagens semissupervisionadas e o SVM supervisionado na base de dados <i>LiveJournal</i> (MIURA <i>et al.</i> , 2014). Melhores resultados estão em negrito.	92
Tabela 16 – F-scores (%) obtidos pelas abordagens semissupervisionadas e o SVM supervisionado na base de dados <i>SMS2013</i> (NAKOV <i>et al.</i> , 2013). Melhores resultados estão em negrito.	92
Tabela 17 – F-scores (%) obtidos pelas abordagens semissupervisionadas e o SVM supervisionado na base de dados <i>Twitter2013</i> (NAKOV <i>et al.</i> , 2013). Melhores resultados estão em negrito.	92
Tabela 18 – F-scores (%) obtidos pelas abordagens semissupervisionadas e o SVM supervisionado na base de dados <i>Twitter2014</i> (MIURA <i>et al.</i> , 2014). Melhores resultados estão em negrito.	93
Tabela 19 – F-scores (%) obtidos pelas abordagens semissupervisionadas e o SVM supervisionado na base de dados <i>TwitterSarcasm2014</i> (MIURA <i>et al.</i> , 2014). Melhores resultados estão em negrito.	93
Tabela 20 – F-scores para diferentes percentuais de dados rotulados para abordagens semissupervisionadas e também supervisionadas — incluindo os melhores resultados da literatura ao se utilizar o conjunto completo de dados rotulados. Os melhores resultados estão em negrito.	96
Tabela 21 – Proporção de maiores F-scores e menores desvios padrões ao se comparar apenas os resultados da Tabela 20 para o C ³ E-SL semissupervisionado e sua contraparte, a qual emprega <i>self-training</i> com SVM. Os melhores resultados estão em negrito.	97

Tabela 22 – Matrizes de confusão produzidas com resultados do C ³ E-SL em conjuntos de treinamento iniciais (contendo, portanto, apenas as classes “ <i>Duck</i> ” e “ <i>Cat</i> ”) e com resultados das três primeiras iterações do IC-EDS — em cada iteração, 5 imagens selecionadas foram manualmente rotuladas e incorporadas no modelo de classificação.	108
Tabela 23 – Imagens selecionadas (%), F-scores e taxas de erro para cada classe deixada de fora da base de dados <i>Coil-20</i> . Os algoritmos executaram 5 iterações (25 imagens foram selecionadas e manualmente rotuladas). Desvios padrões estão entre parênteses e os melhores resultados estão destacados em negrito.	114
Tabela 24 – Imagens selecionadas (%), F-scores e taxas de erro para cada classe deixada de fora da base de dados <i>Corel-1000</i> . Os algoritmos executaram 5 iterações (25 imagens foram selecionadas e manualmente rotuladas). Desvios padrões estão entre parênteses e os melhores resultados estão destacados em negrito.	116
Tabela 25 – Imagens selecionadas (%), F-scores e taxas de erros para cada classe deixada de fora da base de dados <i>Supermarket Produce</i> . Os algoritmos executaram 5 iterações (25 imagens foram selecionadas e manualmente rotuladas). Desvios padrões estão entre parênteses e os melhores resultados estão destacados em negrito.	119

LISTA DE SÍMBOLOS

$\mathcal{X}^T$ — Conjunto de treinamento

$\mathcal{X}$ — Conjunto alvo (de teste)

M — Número de atributos dos objetos em $\mathcal{X}^T$

N — Número de objetos em $\mathcal{X}^T$

n — Número de objetos em $\mathcal{X}$

c — Número de classes

k — Número de grupos

$\mathbf{D}$ — Matriz de distâncias

$\mathbf{S}$ — Matriz de similaridades

$\boldsymbol{\pi}$ — Vetor de estimativas de distribuição de probabilidades de classes

$\mathbf{y}$ — Vetor de estimativas refinadas de distribuição de probabilidades de classes *a posteriori*

λ — Multiplicador de Lagrange requerido pelo algoritmo C³E

α — Parâmetro do algoritmo C³E para controle da importância de classificadores e agrupadores

I — Número de iterações do algoritmo C³E

P — Número de indivíduos da população usada pelas meta-heurísticas

F — Taxa de escalonamento do algoritmo *Differential Evolution*

Cr — Taxa de *crossover* do algoritmo *Differential Evolution*

$\mathbf{e}$ — Vetor de entropias

$\mathbf{d}$ — Vetor de densidades

h — Número de vizinhos mais próximos usado no algoritmo IC-EDS

SUMÁRIO

1	INTRODUÇÃO	29
1.1	Agregação de Classificadores	31
1.2	Agregação de Agrupadores	32
1.3	Contextualização da Pesquisa	33
1.4	Hipótese e Ineditismo	35
1.5	Objetivos Gerais	36
1.6	Organização da Tese	36
2	COMBINAÇÃO DE CLASSIFICADORES E AGRUPADORES	39
2.1	O Algoritmo C^3E	41
2.1.1	<i>Análise de Complexidade Computacional</i>	46
2.1.2	<i>C^3E com Função de Perda Quadrática (C^3E-SL)</i>	47
2.2	Avaliação Empírica	48
2.3	Considerações Finais	53
3	META-HEURÍSTICAS PARA COMBINAR CLASSIFICADORES E AGRUPADORES	55
3.1	Meta-heurísticas para Otimização Paramétrica	56
3.2	Avaliação Empírica das Meta-heurísticas	60
3.2.1	<i>Configuração Experimental</i>	60
3.2.2	<i>Resultados e Análises</i>	63
3.3	Um Novo Algoritmo para Calibrar o C^3E-SL	69
3.4	Avaliação Empírica do Algoritmo D^2E	71
3.5	Considerações Finais	77
4	APRENDIZADO SEMISSUPERVISIONADO COMBINANDO CLASSIFICADORES E AGRUPADORES	81
4.1	Abordagens para Aprendizado Semissupervisionado	83
4.1.1	<i>C^3E-SL Semissupervisionado</i>	84
4.1.2	<i>Exemplo Pedagógico</i>	85
4.2	Avaliação Empírica na Classificação de Tweets	88
4.2.1	<i>Análise Comparativa</i>	91
4.2.2	<i>Impactos da Quantidade de Dados Rotulados Disponíveis</i>	93
4.3	Considerações Finais	95

5	COMBINANDO CLASSIFICADORES E AGRUPADORES PARA APRENDER NOVAS CLASSES	99
5.1	Aprendizagem de Novas Classes de Imagens	101
5.1.1	<i>Uma Versão Iterativa do C³E-SL Baseada em Active Learning</i>	103
5.1.2	<i>Configurando o C³E-SL e seus Componentes</i>	104
5.1.3	<i>Exemplo Pedagógico</i>	105
5.2	Avaliação Empírica	108
5.2.1	<i>Base de Dados Coil-20</i>	110
5.2.2	<i>Base de Dados Corel-1000</i>	113
5.2.3	<i>Base de Dados Supermarket Produce</i>	116
5.3	Considerações Finais	118
6	CONCLUSÕES	121
6.1	Principais Contribuições	121
6.2	Publicações Geradas e Artigos Submetidos	123
6.3	Limitações e Trabalhos Futuros	124
	Referências	127

INTRODUÇÃO

Convencionalmente, denomina-se de “Aprendizado de Máquina” o campo de estudo que trata do projeto e desenvolvimento de métodos e algoritmos capazes de *aprender* com base em experiência anterior (ALPAYDIN, 2010; BISHOP, 2007; MITCHELL, 1996). Algoritmos de aprendizado de máquina são capazes de adquirir *conhecimento* a partir de dois tipos elementares de aprendizado: o aprendizado *supervisionado* e o *não supervisionado*. Estes dois paradigmas de aprendizado oferecem, respectivamente, fundamentos para o estudo e desenvolvimento dos chamados *classificadores* e *agrupadores* (WITTEN; FRANK, 2005).

Para construir um classificador a partir dos dados assume-se que, para cada objeto¹ $\mathbf{x}_i^T$ de um conjunto de treinamento $\mathcal{X}^T = \{\mathbf{x}_i^T\}_{i=1}^N$ — sendo $\mathbf{x}_i^T$ um vetor² de valores para um conjunto de M atributos³ $\mathbf{A} = \{A_j\}_{j=1}^M$ — é fornecido também um rótulo da classe à qual o objeto pertence. Assim, cada objeto se torna uma tupla $\langle \mathbf{x}_i^T, C_\ell^i \rangle$, sendo $C_\ell \in C = \{C_1, C_2, \dots, C_c\}$ a classe do objeto; e o problema de inferi-la para objetos não observados em $\mathcal{X}^T$ é comumente chamado de *classificação*⁴. Portanto, a classificação de dados é um processo que consiste de duas etapas (HAN; KAMBER, 2006). Inicialmente, um modelo de classificação é induzido a partir do conjunto de treinamento $\mathcal{X}^T$, no qual cada objeto é rotulado de acordo com a classe à qual pertence. Na segunda etapa, o classificador obtido na primeira etapa é utilizado para inferir a classe de novos objetos não rotulados (e não observados durante o treinamento).

Existem centenas de algoritmos para construir classificadores, os quais, em sua maioria, empregam técnicas baseadas em: *árvores de decisão* (QUINLAN, 1993), *redes neurais artificiais* (HAYKIN, 1999), *aprendizado estatístico* ou *probabilístico* (HOLMES; JAIN, 2008), *aprendizado baseado em instâncias* (AHA; KIBLER; ALBERT, 1991), entre outras. Descrições detalhadas sobre esses e outros algoritmos de classificação de dados são fornecidas em diversos

¹ Os termos *instância*, *exemplo*, *registro* e *tupla* são também comumente usados.

² Usando a convenção usual de vetor coluna.

³ Também denominados de *atributos previsores*, *características* ou *variáveis independentes*.

⁴ Se a variável dependente é contínua, tem-se um problema de *regressão*.

livros (BISHOP, 2007; WITTEN; FRANK, 2005; HAND; MANNILA; SMYTH, 2001).

É bem conhecido que cada algoritmo de classificação possui suas próprias vantagens e desvantagens, e que estas são dependentes dos dados, ou seja, não existe um único algoritmo capaz de obter o melhor classificador para todos os problemas de classificação (WOLPERT; MACREARY, 1995). Por conta disso, em aplicações reais, é prática comum testar diferentes algoritmos de classificação e escolher aquele(s) que apresenta(m) melhor desempenho segundo alguma medida de interesse (*e.g.*, menor erro médio de classificação) — alternativamente, abordagens baseadas na *agregação* de classificadores (*ensembles*) têm se mostrado promissoras, como será abordado a seguir.

Agrupadores de dados, por sua vez, são algoritmos não supervisionados projetados para induzir grupos nos dados. A ausência de supervisão se refere ao fato de que, diferentemente de um problema de classificação, os rótulos de classes não estão disponíveis para se construir o modelo. Nestes casos, o conceito de *conjunto de treinamento* tem pouca importância prática, e pode-se denotar o conjunto de dados simplesmente por $\mathcal{X}$, o qual aqui é particionado de tal forma a se obter um conjunto de grupos (*clusters*), também chamado de agrupamento (*clustering*)⁵. A forma pela qual os grupos são induzidos depende do algoritmo de agrupamento (agrupador) usado. Em termos gerais, usualmente procura-se induzir uma partição cujos grupos possuam objetos similares (JAIN; DUBES, 1988). Mais especificamente, e considerando-se um contexto de otimização, tipicamente se busca maximizar a homogeneidade entre os objetos de um mesmo grupo e, concomitantemente, maximizar a heterogeneidade entre objetos de grupos distintos.

Algoritmos para agrupamento de dados podem ser categorizados em dois tipos: *hierárquicos* e *particionais*. Algoritmos hierárquicos geram uma sequência de partições em que cada partição está aninhada⁶ na partição subsequente (JAIN; DUBES, 1988). Algoritmos particionais, por sua vez, obtêm uma partição dos dados contendo k grupos (k pode ser fornecido pelo usuário ou estimado a partir dos próprios dados). Nas últimas décadas, muito trabalho tem sido feito dentro de cada uma dessas categorias de agrupadores. Algoritmos como o *single-link* (SNEATH, 1957), *complete-link* (SORENSEN, 1948) e *average-link* (SOKAL; MICHENER, 1958) são comumente usados quando se deseja realizar o agrupamento hierárquico dos dados. Da mesma forma, os algoritmos K-Médias (KM) (MACQUEEN, 1967), *Expectation-Maximization* (EM) (DEMPSTER; LAIRD; RUBIN, 1977) e *Fuzzy C-Means* (FCM) (BEZDEK, 1981) têm sido frequentemente escolhidos quando se necessita agrupar os dados de maneira particional. Estudos detalhados desses e vários outros algoritmos para agrupamento de dados podem ser encontrados em diversos livros (XU; WUNSCH, 2009; EVERITT; LANDAU; LEESE, 2001; KAUFMAN; ROUSSEUW, 1990; JAIN; DUBES, 1988).

⁵ Apesar de parecer natural traduzir o termo *cluster* como grupo e o termo *clustering* como agrupamento, há muita confusão, especialmente na Língua Portuguesa, em relação ao uso desses termos. Neste trabalho, um agrupamento (ou equivalentemente, uma partição) é definido(a) como sendo um conjunto de grupos.

⁶ Uma partição $\mathcal{A}$ está aninhada em uma partição $\mathcal{B}$ se todo grupo de $\mathcal{A}$ é um subconjunto de um grupo de $\mathcal{B}$ (JAIN; DUBES, 1988).

Embora, nos últimos anos, tenha havido um notável desenvolvimento das pesquisas, não existe um algoritmo de agrupamento universal, *i.e.*, que resolva todos os problemas (XU; WUNSCH, 2009). Normalmente os algoritmos são projetados de acordo com suposições sobre as formas dos grupos ou a distribuição dos dados e, inevitavelmente, favorecem algum tipo de *bias* — para minimizar essa restrição (semelhantemente ao que é feito com os classificadores) abordagens para a *agregação* de agrupadores (*cluster ensembles*) têm sido comumente empregadas na prática.

1.1 Agregação de Classificadores

A combinação de múltiplos classificadores (*ensembles*) para gerar um classificador consolidado tem sido uma área de pesquisa ativa nas últimas décadas⁷. Diversos trabalhos têm fornecido resultados teóricos e evidências empíricas sobre a utilidade de tais abordagens para resolver problemas de classificação difíceis — *e.g.*, (OZA; TUMER, 2008; HANSEN; SALAMON, 1990). Além disso, a extensa literatura existente tem mostrado que um agregador obtido a partir de classificadores diversificados é tipicamente melhor do que seus componentes individuais (KUNCHEVA, 2004; DIETTERICH, 2000; HANSEN; SALAMON, 1990).

O princípio, bem como a importância, de se agregar classificadores distintos reside no fato de que, por exemplo, dada a agregação de três classificadores (h_1 , h_2 e h_3) e um novo objeto $\mathbf{x}_i$ a ser classificado, se os três classificadores base são idênticos, então quando $h_1(\mathbf{x}_i)$ erra a predição, $h_2(\mathbf{x}_i)$ e $h_3(\mathbf{x}_i)$ também erram. Contudo, se os erros dos três classificadores não estão correlacionados, então, quando $h_1(\mathbf{x}_i)$ está errado, $h_2(\mathbf{x}_i)$ e $h_3(\mathbf{x}_i)$ podem estar certos de tal forma que por meio de *voto majoritário* o novo objeto $\mathbf{x}_i$ poderá ser corretamente classificado (DIETTERICH, 2000; HANSEN; SALAMON, 1990). Por ser uma prática simples e normalmente aplicável a qualquer tipo de classificador, muitos trabalhos levam em conta essa estratégia geral (OZA; TUMER, 2008; BAUER; KOHAVI, 1999; BREIMAN, 1996). Entretanto, diversas outras estratégias podem ser consideradas para introduzir diversidade nos classificadores base. Como exemplo, *bagging* (BREIMAN, 1996) faz uso de diversos modelos — cada um induzido por um conjunto de treinamento obtido por meio de *bootstrap* (EFRON; TIBSHIRANI, 1993) — para classificar um objeto não rotulado usando-se voto majoritário. Outro conceito, chamado de *boosting* (FREUND; SCHAPIRE, 1997), atribui pesos na predição de cada modelo levando-se em conta acurácias de classificações passadas — por isso, essa abordagem é considerada iterativa (cada novo modelo é influenciado pelo desempenho daqueles previamente construídos). Outras abordagens conhecidas são *stacking* (WOLPERT, 1992), *model averaging* (PERRONE; COOPER, 1993), *forecast combining* (CLEMEN, 1989) e *random forests*; esta última tem proporcionado importantes resultados em diferentes áreas do conhecimento humano (VERIKAS; GELZINIS; BACAUSKIENE, 2011). A agregação de classificadores é tratada mais detalhadamente em (OZA; TUMER, 2008; BISHOP, 2007; KUNCHEVA, 2004).

⁷ O *International Workshop on Multiple Classifiers Systems* tem acontecido desde 2000.

1.2 Agregação de Agrupadores

Da mesma forma que a aplicação de agregadores de classificadores tem tipicamente mostrado melhores resultados do que o uso de um único classificador, diversas pesquisas também mostram que agregadores de agrupadores (*cluster ensembles*) podem melhorar a qualidade dos resultados obtidos quando comparados com a aplicação de um único agrupador (GHOSH; ACHARYA, 2011). A agregação de agrupadores consiste em conciliar diversas partições com o objetivo de obter uma única partição (normalmente denominada de *partição de consenso*) que melhor represente o conjunto de partições disponíveis (NGUYEN; CARUANA, 2007). Embora o problema de se agregar agrupadores apresente semelhanças com o problema de se agregar classificadores, cumpre notar que: (i) o número de grupos pode ser diferente para cada partição, enquanto que em problemas de classificação o número de classes é fixo; (ii) o número de grupos da partição de consenso é desconhecido *a priori*; (iii) diferentemente de um problema de classificação, os rótulos dos grupos são símbolos que não guardam significado semântico por si só, tornando o alinhamento entre grupos de partições distintas um difícil problema de identificação de correspondência. Estas características fazem com que o problema de se agregar agrupadores seja, geralmente, mais difícil do que o problema de se agregar classificadores.

O problema de se encontrar uma partição de consenso pode ser formalizado da seguinte maneira (STREHL; GHOSH, 2002): dado um conjunto de n_π partições iniciais (partições base)⁸, $\Pi = \{\pi_1, \pi_2, \dots, \pi_{n_\pi}\}$, resultantes de várias aplicações de um ou mais algoritmos de agrupamento⁹ em um conjunto de dados $\mathcal{X}$, encontrar uma única partição final π_f que apresente *melhor qualidade* em relação às partições base. O termo “melhor qualidade” pode estar relacionado a diferentes aspectos, tais como (GHOSH; ACHARYA, 2011; TOPCHY; JAIN; PUNCH, 2005):

- *Qualidade da Solução*: analogamente ao que se observa ao agregar classificadores, a agregação de agrupadores também pode melhorar a qualidade da partição obtida relativamente a uma única partição base.
- *Robustez*: é bem conhecido que os algoritmos de agrupamento apresentam desempenho ruim para bases de dados que não atendem às suas premissas de modelagem. Um agregador de agrupadores pode ser visto como um modelo mais robusto no sentido de fornecer melhores resultados para diferentes bases de dados.
- *Seleção de Modelo*: o processo de agregação de múltiplas partições, formadas por diferentes números de grupos, oferece uma abordagem alternativa para a tradicional tarefa de seleção do modelo que possui o número de grupos mais apropriado. Na literatura sobre reconhecimento de padrões, o problema de se estimar o número de grupos usualmente en-

⁸ Também denominadas de componentes do agregador.

⁹ Pode-se usar o mesmo algoritmo de agrupamento, *e.g.*, K-Médias com diferentes subconjuntos de atributos e/ou diferentes números de grupos.

volve a aplicação de um índice de validade relativo — *e.g.*, (VENDRAMIN; CAMPELLO; HRUSCHKA, 2010).

- *Reuso de Conhecimento*: em algumas aplicações, partições de dados obtidas em projetos anteriores podem estar disponíveis. Tais partições podem ser refinadas com o auxílio de partições induzidas mais recentemente, de tal forma que resultados consolidados possam ser derivados.
- *Múltiplas Visões*: frequentemente os objetos a serem agrupados apresentam múltiplos aspectos (ou diferentes visões), de acordo com os atributos sob análise. Por exemplo, os clientes de uma determinada empresa podem ser segmentados de acordo com diferentes subconjuntos de atributos (necessidades, perfis demográficos, atitudes, etc.). Em tais cenários, pode-se obter um conjunto de partições base (uma para cada subconjunto de atributos) e então consolidar os resultados obtidos via uma partição de consenso.
- *Computação Distribuída*: em certas situações, os dados estão geograficamente distribuídos e não é possível centralizá-los por conta de restrições de segurança e/ou confidencialidade, bem como por conta de limitações tecnológicas (por exemplo, largura de banda e/ou espaço para armazenamento de dados). Nessas situações, uma partição de consenso pode ser obtida exclusivamente a partir das partições base, sem necessidade de acesso aos dados brutos utilizados para induzi-las — *e.g.*, (COLETTA *et al.*, 2012; VENDRAMIN *et al.*, 2011).

Devido à sua ampla aplicabilidade, diversos algoritmos têm sido propostos para se obter uma partição de consenso, cada qual envolvendo alguma forma de se gerar as partições base e também alguma estratégia de agregação. Na literatura pode-se encontrar abordagens baseadas em: *coassociação* (FRED; JAIN, 2005; STREHL; GHOSH, 2002), *votação* (FROSSYNIOTIS; PATERITSAS; STAFYLOPATIS, 2005; DIMITRIADOU; WEINGESSEL; HORNIK, 2001), *grafos* (TOPCHY; JAIN; PUNCH, 2005; STREHL; GHOSH, 2002), *modelos probabilísticos* (TOPCHY; JAIN; PUNCH, 2004), entre outras. Algoritmos para a agregação de agrupamentos estão descritos de maneira mais aprofundada em (GHOSH; ACHARYA, 2011; GHAEMI *et al.*, 2009; HORE; HALL; GOLDFOF, 2009).

1.3 Contextualização da Pesquisa

Algoritmos de aprendizado de máquina, como classificadores e agrupadores, têm sido ostensivamente utilizados nas mais diversas áreas do conhecimento humano. Por exemplo, na identificação de genes, tecidos doentes, nichos de mercado e corpos celestes; na detecção de fraudes, *spams* e intrusões; no reconhecimento de voz, face e escrita (LU; WENG, 2007; JAIN; DUIN; MAO, 2000; JAIN; MURTY; FLYNN, 1999). Esses algoritmos também têm sido substancialmente úteis e importantes no grande campo da *mineração de dados* (HAN;

KAMBER, 2006; WITTEN; FRANK, 2005; FAYYAD *et al.*, 1996; KLÖSGEN; ZYTKOW, 1996). Para se alcançar melhores resultados nestas aplicações, diversas abordagens baseadas em agregação (*ensemble*) têm sido concebidas e utilizadas na prática — tanto em problemas de classificação (OZA; TUMER, 2008; KUNCHEVA, 2004), como em problemas de agrupamento (GHAEMI *et al.*, 2009; HORE; HALL; GOLDFOF, 2009; GIONIS; MANNILA; TSAPARAS, 2007; STREHL; GHOSH, 2002). O presente trabalho é focado na agregação de algoritmos de aprendizado de máquina. Em particular, o objetivo é conciliar os benefícios do uso de dados rotulados e não rotulados, a exemplo de abordagens relacionadas, as quais se baseiam na combinação de classificadores e agrupadores (GAN *et al.*, 2013; RAHMAN; VERMA, 2013; DUVAL-POO *et al.*, 2012; QIAN; CHEN; CAI, 2012; SOARES; CHEN; YAO, 2012; VERMA; RAHMAN, 2012; ACHARYA *et al.*, 2014; ACHARYA *et al.*, 2011; CAI; CHEN; ZHANG, 2010; CAI; CHEN; ZHANG, 2009; GAO *et al.*, 2013; GAO *et al.*, 2009).

Diversos trabalhos presentes na literatura propõem formas de se combinar classificadores e agrupadores, de tal forma que agrupamentos previamente induzidos possam ajudar na construção de modelos de classificação em uma fase subsequente (XIANG; ZHOU, 2014a; CAI; CHEN; ZHANG, 2007; YANG, 2006; CASTRO; HRUSCHKA; CAMPELLO, 2004; KIM; OOMMEN, 2003; TANG *et al.*, 2002; SETNES; BABUSKA, 2002). Outras pesquisas têm explorado métodos que possam realizar simultaneamente supervisão e não supervisão, de maneira que tanto a classificação quanto o agrupamento possam se beneficiar do processo de aprendizado (QIAN; CHEN; CAI, 2012; CAI; CHEN; ZHANG, 2010; CAI; CHEN; ZHANG, 2009). Alguns trabalhos têm focado na agregação de diferentes classificadores e diferentes agrupadores para se obter uma classificação consolidada dos dados (ACHARYA *et al.*, 2014; ACHARYA *et al.*, 2011; GAO *et al.*, 2013; GAO *et al.*, 2009) figurando-se como abordagens bem próximas do interesse desta tese. Em particular, Acharya *et al.* (2011) introduziram o algoritmo C³E (*Consensus between Classification and Clustering Ensembles*), que classifica novos dados a partir do processamento de dois modelos: um modelo resultante da agregação de classificadores (abordados na Seção 1.1) e outro resultante da agregação de agrupadores (discutidos na Seção 1.2). Este algoritmo assume que objetos similares são mais propensos a compartilhar o mesmo rótulo de classe. As similaridades entre objetos são capturadas por meio do agrupamento dos dados. Sob esta perspectiva, agrupamentos fornecem restrições complementares que podem ser úteis para melhorar a capacidade de generalização do classificador — especialmente quando dados rotulados são escassos no conjunto de treinamento — cenário típico de aprendizado *semisupervisionado* (CHEN; GUO; CHEN, 2009; BASU; DAVIDSON; WAGSTAFF, 2008; CHAPELLE; SCHÖLKOPF; ZIEN, 2006).

O algoritmo C³E tem apresentado resultados empíricos atraentes, além de ser computacionalmente eficiente na prática (ACHARYA *et al.*, 2014; ACHARYA *et al.*, 2011). Tais aspectos favorecem o uso deste algoritmo em problemas do mundo real e, sobretudo, fortalecem as motivações para se conduzir mais pesquisas nesta linha. O presente trabalho introduz melhorias no algoritmo C³E, bem como o adapta para determinados cenários de aplicação (nos quais

acredita-se que o algoritmo seja mais adequado e promissor). Neste sentido, durante a pesquisa, foram concebidas variantes do C³E para atuarem de forma mais realista, atacando-se problemas de classificação de dados que na maioria das vezes são negligenciados em muitas aplicações. Por exemplo, a maioria dos classificadores assumem que os dados de treinamento e os dados do conjunto alvo (contendo objetos que se deseja classificar) possuem a mesma distribuição de probabilidades (i.e., não consideram nenhum tipo de *transferência de conhecimento* (PAN; YANG, 2010; SILVER; BENNETT, 2008)); bem como não consideram o surgimento de novas classes não observadas durante a fase de treinamento do modelo (MASUD *et al.*, 2010; SONG; RAJASEKARAN, 2008; JIA *et al.*, 2005). Nestes casos, técnicas que tratem concomitantemente dados rotulados e não rotulados podem ser úteis e promissoras (MASUD *et al.*, 2011; MASUD *et al.*, 2010; GAO *et al.*, 2008).

1.4 Hipótese e Ineditismo

Esta pesquisa assume como hipótese que variantes do algoritmo C³E possam se beneficiar da combinação de classificadores e agrupadores para serem “conscientes” das possíveis diferenças entre as distribuições dos dados de treinamento e dos dados do conjunto alvo, sendo capazes de até mesmo lidar com o surgimento de novas classes não observadas durante o treinamento do modelo. Algoritmos de classificação requerem um conjunto de treinamento representativo para produzir modelos que possam discriminar apropriadamente classes de objetos em uma aplicação em particular (BISHOP, 2007). No entanto, o processo de obtenção de objetos rotulados para compor um conjunto de treinamento é uma tarefa dispendiosa (TUIA *et al.*, 2011; FOODY; MATHUR, 2004). Por conta disso, em aplicações do mundo real, tipicamente apenas um conjunto limitado de objetos rotulados está disponível. Devido a esta dificuldade, e também por conta de particularidades do processo de amostragem, em muitos domínios práticos não se tem conhecimento (*a priori*) de todas as possíveis classes para um problema específico de classificação (SCHEIRER *et al.*, 2013). A concepção de algoritmos que permitam que o modelo de classificação possa se adaptar com o tempo para superar o problema de escassez de dados rotulados (em cenários *semisupervisionados*), bem como aprender novas classes (não observadas de antemão) é de grande importância prática (LAMPERT; NICKISCH; HARMELING, 2014). Entretanto, poucos estudos têm sido conduzidos nesta direção.

Nesta pesquisa buscou-se conceber ferramentas de classificação mais eficazes e eficientes. O ponto de partida do trabalho foi uma versão mais simples e flexível do algoritmo C³E, baseada na função de Perda Quadrática (C³E-SL) — a qual não foi explicitamente estudada em (ACHARYA *et al.*, 2011). Baseando-se em meta-heurísticas de busca (e.g., algoritmos evolutivos), o algoritmo C³E-SL pôde ser calibrado automaticamente a partir dos dados. Assim, parâmetros requeridos foram suprimidos, tornando o algoritmo menos dependente do usuário. A partir de um procedimento evolutivo, que facilita o uso do C³E-SL em problemas práticos do mundo real, variantes do algoritmo foram desenvolvidas. Sob esta perspectiva, o ineditismo

desta pesquisa reside no conjunto de vantagens atrelado aos algoritmos concebidos, as quais envolvem:

- Menor dependência do usuário/especialista de domínio, pois a aplicação é livre de parâmetros considerados críticos por conta do uso de uma abordagem que automaticamente otimiza (a partir dos dados) os parâmetros da ferramenta;
- Ampla abrangência de aplicação, pois as propriedades investigadas tornam os algoritmos úteis em cenários com poucos dados rotulados (aprendizado *semisupervisionado*), bem como em cenários suscetíveis à mudanças na distribuição dos dados e o surgimento de novas classes;
- Versatilidade da ferramenta, pois o algoritmo C³E aceita o uso de (praticamente) qualquer componente nos agregadores, independente de atribuições rígidas ou suaves de classes e/ou grupos;
- Suporte à computação distribuída e restrições de privacidade, pois os dados de treinamento não necessitam estar disponíveis após a construção do agregador de classificadores, favorecendo aplicações em que o acesso aos atributos é proibitivo.

1.5 Objetivos Gerais

Os objetivos desta pesquisa de doutorado foram:

- 1) Estudar e desenvolver abordagens evolutivas (preferencialmente escaláveis) que facilitem a combinação de classificadores e agrupadores para problemas de classificação;
- 2) Investigar e conceber formas de se lidar com a escassez de dados rotulados (em cenários semisupervisionados), bem como tratar o surgimento de novas classes (não previstas *a priori* pelo modelo) a partir da adaptação das abordagens desenvolvidas em 1.

1.6 Organização da Tese

Este documento está organizado da seguinte maneira:

Capítulo 2 - Combinação de Classificadores e Agrupadores: são apresentadas as fundamentações teóricas e práticas que envolvem a combinação de classificadores e agrupadores. O algoritmo C³E ([ACHARYA et al., 2011](#)) é detalhadamente descrito, incluindo seu caso particular, com uso da função de Perda Quadrática (C³E-SL). A avaliação empírica conduzida mostra que o C³E-SL oferece resultados competitivos de classificação, mas que pode ser aperfeiçoado, motivando os desenvolvimentos propostos em capítulos posteriores.

Capítulo 3 - Meta-heurísticas para Combinar Classificadores e Agrupadores: são abordadas diferentes meta-heurísticas de busca. Tais meta-heurísticas podem ser usadas na calibração do algoritmo C³E-SL. A partir disto, um algoritmo evolutivo adaptado para a combinação de classificadores e agrupadores é introduzido com o intuito de tornar o C³E-SL livre de parâmetros considerados críticos. Sua avaliação empírica envolve aplicações em bases de dados reais.

Capítulo 4 - Aprendizado Semissupervisionado Combinando Classificadores e Agrupadores: envolve investigações sobre as capacidades do algoritmo C³E-SL em lidar com a escassez de dados rotulados. Um algoritmo baseado em *self-training* (ZHU; GOLDBERG, 2009) é introduzido. Este algoritmo requer uma quantidade pequena de dados rotulados e realiza classificações com o auxílio de informação não supervisionada. Os experimentos consideraram especificamente a análise de sentimentos em *tweets*.

Capítulo 5 - Combinando Classificadores e Agrupadores para Aprender Novas Classes: aborda o problema do surgimento de novas classes (não observadas durante o treinamento do modelo de classificação). São apresentados trabalhos da literatura que buscam suprimir este tipo de problema. A partir do estudo realizado, uma extensão do algoritmo C³E-SL capaz de se autoadaptar para aprender novas classes é introduzida. A avaliação empírica envolve a área de visão computacional, na qual o algoritmo proposto, baseado em *active learning* (SETTLES, 2009), foi capaz de descobrir novos conceitos/classes a partir de poucos objetos rotulados.

Capítulo 6 - Conclusões: são sumarizadas as principais conclusões e contribuições desta tese, além de apresentar possíveis linhas de pesquisa para trabalhos futuros.

COMBINAÇÃO DE CLASSIFICADORES E AGRUPADORES

A principal motivação para se combinar classificadores e agrupadores provém da possibilidade de conciliação dos diferentes aspectos invocados ao se utilizar agregadores de classificadores (Seção 1.1) ou agregadores de agrupadores (Seções 1.2) isoladamente. Desta forma, espera-se que algoritmos que combinem informações de dados rotulados e não rotulados possam também se beneficiar das propriedades que tornaram os agregadores de classificadores e de agrupadores ferramentas conceituais amplamente usadas na prática. Adicionalmente, outras propriedades desejáveis podem emergir da combinação dessas duas ferramentas. Por exemplo, modelos não supervisionados podem fornecer uma série de restrições complementares para auxiliar na classificação de novos dados em um conjunto alvo (ACHARYA *et al.*, 2014; ACHARYA *et al.*, 2011; BASU; DAVIDSON; WAGSTAFF, 2008). A partir deste ponto de vista, a premissa subjacente é que objetos similares no conjunto alvo são mais propensos a compartilharem o mesmo rótulo de classe. Esta premissa é intuitiva e é assumida por esta pesquisa, bem como por diferentes estudos recentes (ACHARYA *et al.*, 2014; GAN *et al.*, 2013; GAO *et al.*, 2013; RAHMAN; VERMA, 2013; DUVAL-POO *et al.*, 2012; SOARES; CHEN; YAO, 2012; VERMA; RAHMAN, 2012).

Diversos trabalhos na literatura têm reportado os benefícios de se usar informações de dados rotulados e não rotulados em aplicações de aprendizado de máquina. Fundamentalmente, resultados de agrupamentos podem ajudar a aumentar a capacidade de generalização de classificadores (BOUSQUET; BOUCHERON; LUGOSI, 2004), enquanto que vários trabalhos que abordam *agrupamento supervisionado* (PEDRYCZ; VUKOVICH, 2004; PAPADIMITRIOU *et al.*, 2001; TIMM, 2001) e *agrupamento semissupervisionado* (KULIS *et al.*, 2005; ZHU, 2008; BASU; BILENKO; MOONEY, 2004) fornecem evidências teóricas e práticas sobre as vantagens de se usar a informação das classes para agrupar dados. Do ponto de vista de classificação, a ideia de se combinar classificadores e agrupadores tem como benefícios (CAI; CHEN; ZHANG, 2010; CAI; CHEN; ZHANG, 2009; GAO *et al.*, 2013; GAO *et al.*, 2009): (i) aumentar a qualidade e a

robustez do classificador; e (ii) facilitar a interpretação do modelo na medida em que se torna possível a identificação das relações subjacentes entre grupos e classes. Em aplicações de fluxos contínuos de dados (*data streams*), algoritmos de agrupamento de dados são úteis para identificar mudanças de *conceito* e novas classes não previstas no conjunto de treinamento¹. Sob este ponto de vista, a informação de grupos pode ser útil para se projetar métodos de aprendizado que sejam “conscientes” das possíveis diferenças entre as distribuições dos dados de treinamento e dos dados do conjunto alvo (ACHARYA *et al.*, 2014). Outras aplicações envolvem a identificação de *outlier*, evitando a sua associação com alguma das classes (KOLLIOS *et al.*, 2003; MILLER; BROWNING, 2003) e a preocupação em tornar o modelo mais interpretável (YONG *et al.*, 2009; BAI; CHAI; MA, 2007; ZHANG; NEAGU; BALESCU, 2005).

A maioria das trabalhos que se propõem a combinar agrupadores e classificadores, realizam esta tarefa de maneira *sequencial* ou *simultânea*:

- Nas abordagens sequenciais, os dados são primeiramente agrupados e posteriormente classificados a partir da estrutura de grupos obtida. Por exemplo, Setnes e Babuska (1999) inicialmente usaram o algoritmo *Fuzzy C-Means* (FCM) (BEZDEK, 1981) para induzir grupos e, na sequência, uma matriz de relação *fuzzy* — a qual indica as relações entre grupos e classes — é construída para auxiliar na classificação de novos objetos. Cai, Chen e Zhang (2007) aperfeiçoaram este algoritmo, tornando-o mais estável e capaz de atuar em conjuntos de dados com distribuições não necessariamente hiper-esféricas. Li e Ye (2006) introduziram um algoritmo que se baseia em agrupamento supervisionado para construir um classificador. Em (CASTRO; HRUSCHKA; CAMPELLO, 2004) um algoritmo genético para agrupamento é combinado com o K-Médias (KM) (MACQUEEN, 1967) para automaticamente estimar a localização e o número de funções de base radial em uma rede neural RBF. Embora esses trabalhos indiquem que as tarefas de classificação e agrupamento possam ser complementares, tais abordagens sequenciais nem sempre favorecem a otimalidade dos dois tipos de aprendizagem (CAI; CHEN; ZHANG, 2009), *i.e.*, na maioria das vezes, a tarefa de agrupamento apenas auxilia na classificação, mas não se beneficia desta. Cai, Chen e Zhang (2009) também argumentam que, por meio do conhecimento das relações entre grupos e classes, pode-se obter classificadores mais interpretáveis e robustos, desta forma motivando o uso de abordagens simultâneas para se combinar classificadores e agrupadores.
- Buscando combinar um classificador com um agrupador de maneira simultânea, Cai, Chen e Zhang (2009) introduziram um algoritmo que se baseia na otimização de uma única função objetivo, projetada para avaliar tanto o classificador quanto o agrupador, de maneira que se possa alcançar simultaneamente resultados robustos e eficazes para os dois modelos.

¹ A mudança de conceito também é conhecida como *concept drift*, e o surgimento de novas classes envolve a evolução de conceito também chamada de *concept evolution* (MASUD *et al.*, 2013; MASUD *et al.*, 2010; MASUD *et al.*, 2010).

Posteriormente, em (CAI; CHEN; ZHANG, 2010), esse algoritmo foi estendido para um cenário de otimização multiobjetivo, no qual se permite a otimização do problema por meio de vários critérios (tanto para agrupamento quanto para classificação). Para ilustrar a abordagem proposta, os autores utilizaram duas funções objetivo baseadas nos protótipos dos grupos obtidos pelo algoritmo de agrupamento FCM (taxa de erro de classificação e compactação/separabilidade entre os grupos). Inspirado por estes trabalhos, Qian, Chen e Cai (2012) desenvolveram uma abordagem mais rápida e flexível. O algoritmo tem garantias de convergência e foi generalizado para aprendizado semissupervisionado.

Como visto, grande parte dos trabalhos existentes na literatura apenas combinam um classificador com um agrupador, muitas vezes assumindo o agrupamento dos dados como uma tarefa de pré-processamento para posterior classificação. Dois trabalhos de destaque (ACHARYA *et al.*, 2011; GAO *et al.*, 2009) são capazes de combinar diferentes classificadores e diferentes agrupadores para lidar exclusivamente com o problema de classificação de dados. Este foi o foco desta pesquisa. Em particular, o algoritmo C³E (*Consensus between Classification and Clustering Ensembles*) (ACHARYA *et al.*, 2011) classifica novos dados a partir da combinação de dois modelos: um resultante da agregação de classificadores (Seção 1.1) e outro resultante da agregação de agrupadores (Seções 1.2). Como principal objeto de estudo deste trabalho, este algoritmo é descrito mais detalhadamente a seguir.

2.1 O Algoritmo C³E

A abordagem do algoritmo C³E (*Consensus between Classification and Clustering Ensembles*) (ACHARYA *et al.*, 2011) para combinar classificadores e agrupadores gerando uma classificação consolidada dos dados é ilustrada na Figura 1. Em resumo, este algoritmo recebe como entradas distribuições de probabilidades de classes, $\{\pi_i\}_{i=1}^n$, e uma matriz de similaridades, S , referentes aos objetos de um conjunto alvo, $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^n$, que se deseja classificar. Após processar estas entradas, o algoritmo fornece, como resultado, um conjunto de vetores $\{\mathbf{y}_i\}_{i=1}^n$ correspondentes as estimativas (refinadas) das probabilidades de classes *a posteriori* para cada objeto de $\mathcal{X}$, ou seja, $\mathbf{y}_i = p(C | \mathbf{x}_i)$.

Mais formalmente, considere que r_1 ($r_1 \geq 1$) classificadores, indexados por q_1 , e r_2 ($r_2 \geq 1$) agrupadores, indexados por q_2 , são empregados para obter uma classificação consolidada dos objetos do conjunto alvo. Os seguintes passos (I – III) sumarizam a abordagem proposta em (ACHARYA *et al.*, 2011) — os passos I e II podem ser vistos como passos preliminares para se obter as entradas (*inputs*) para o algoritmo C³E, enquanto que o passo III se constitui no algoritmo de otimização propriamente dito.

Passo I - Obter entradas do C³E a partir da agregação de classificadores: a saída (*output*) de um classificador, q_1 , para o objeto $\mathbf{x}_i \in \mathcal{X}$ é um vetor c -dimensional que captura a

Figura 1 – Visão geral do algoritmo C³E (ACHARYA *et al.*, 2011).

distribuição de probabilidades de classes $\pi_i^{(q_1)}$. Mais especificamente, este vetor denota as probabilidades atribuídas a cada uma das classes de um dado problema. Tais probabilidades podem refletir tanto atribuições rígidas de classes (caso específico no qual os vetores são formados por valores binários) ou suaves (caso geral). A partir do conjunto de tais vetores, $\{\pi_i^{(q_1)}\}_{q_1=1}^{r_1}$, um vetor médio pode ser computado para cada objeto $\mathbf{x}_i$ do conjunto alvo:

$$\pi_i = \frac{1}{r_1} \sum_{q_1=1}^{r_1} \pi_i^{(q_1)}. \quad (2.1)$$

Passo II - Obter entradas do C³E a partir da agregação de agrupadores: após a execução dos agrupadores, r_2 partições dos dados do conjunto alvo estão disponíveis para se construir uma matriz de similaridades (coassociação) $\mathbf{S}$. Dada uma dessas partições, (rígida) com n objetos, uma matriz binária e simétrica, $\mathbf{S}_{n \times n}^{(q_2)}$, — com o elemento $s_{ij}^{(q_2)}$ sendo igual a 1 se os objetos i e j estão no mesmo grupo e igual a 0 caso contrário — pode ser obtida². Em outras palavras, dado que a q_2 -ésima partição possui $k^{(q_2)}$ grupos, o resultado desse agrupamento pode ser denotado por um vetor de rótulos $\lambda^{(q_2)} \in \mathbb{Z}_+^n$, e os elementos da respectiva matriz binária, $\mathbf{S}^{(q_2)}$, podem ser computados da seguinte forma:

$$s_{ij}^{(q_2)} = \begin{cases} 1 & (i,j) \in C_g(\lambda^{(q_2)}) \text{ para algum } g \in \{1, 2, \dots, k^{(q_2)}\} \\ 0 & \text{caso contrário.} \end{cases}, \quad (2.2)$$

onde $C_g(\lambda^{(q_2)})$ é o g -ésimo grupo na q_2 -ésima partição. Uma matriz de similaridades pode ser computada a partir da média de todas as r_2 matrizes:

$$\mathbf{S} = \frac{1}{r_2} \sum_{q_2=1}^{r_2} \mathbf{S}^{(q_2)}. \quad (2.3)$$

² Uma matriz de similaridades pode também ser definida para partições suaves (*soft*) ou probabilísticas — *e.g.*, (PUNERA; GHOSH, 2008).

Passo III - Obter resultados de classificação consolidados a partir do C^3E : após definir as entradas para o C^3E ($\{\pi_i\}_{i=1}^n$ e $\mathbf{S}$), o problema de se combinar classificadores e agrupadores pode ser formulado como um problema de otimização cujo objetivo é minimizar J na Equação (2.4) com relação ao conjunto de vetores de probabilidades $\{\mathbf{y}_i\}_{i=1}^n$, no qual $\mathbf{y}_i$ é a estimativa refinada da distribuição de probabilidades de classes *a posteriori* para um determinado objeto de $\mathcal{X}$:

$$J = \sum_{i \in \mathcal{X}} \mathcal{L}(\pi_i, \mathbf{y}_i) + \alpha \sum_{(i,j) \in \mathcal{X}} s_{ij} \mathcal{L}(\mathbf{y}_i, \mathbf{y}_j). \quad (2.4)$$

Na Equação (2.4), a quantidade $\mathcal{L}(\cdot, \cdot)$ denota uma função de perda. Informalmente, o primeiro termo captura as dissimilaridades entre as distribuições de probabilidades de classes fornecidas pelo agregador de classificadores e os vetores de saída $\{\mathbf{y}_i\}_{i=1}^n$ do C^3E . Em outras palavras, esse termo procura fazer com que a distribuição de probabilidades de classes para cada objeto seja parecida com aquela obtida pelo agregador de classificadores. Por outro lado, o segundo termo captura a dissimilaridade ponderada acumulada entre todos os pares possíveis de $(\mathbf{y}_i, \mathbf{y}_j)$. Os pesos desses pares são atribuídos proporcionalmente às suas respectivas similaridades — elementos $s_{ij} \in [0, 1]$ da matriz $\mathbf{S}$. Ou seja, este segundo termo captura a ideia de que objetos semelhantes (sob o ponto de vista do agregador de agrupadores) deveriam ter distribuições de probabilidades de classes também semelhantes. O coeficiente $\alpha \in \mathbb{R}_+$ (que é um parâmetro requerido pelo algoritmo) controla a importância do agregador de classificadores em relação ao agregador de agrupadores. Assim, minimizar a função objetivo para $\{\mathbf{y}_i\}_{i=1}^n$ envolve combinar a evidência fornecida pelos classificadores com a evidência fornecida pelos agrupadores, de tal forma que uma classificação mais consolidada possa ser obtida³.

No tocante à função de perda, $\mathcal{L}(\cdot, \cdot)$, [Acharya et al. \(2011\)](#) adotam uma abordagem suficientemente geral no sentido de que qualquer Divergente de Bregman possa ser usado na Equação (2.4). Mais especificamente, Divergentes de Bregman incluem diversas funções de perda — tais como a Perda Quadrática, o Divergente de Kullback-Leibler, a Perda Logística, a Distância de Mahalanobis e o Divergente-I ([BANERJEE et al., 2005](#)). Usando Divergentes de Bregman, o problema de otimização em (2.4) pode ser re-escrito como:

$$\min_{\{\mathbf{y}_i\}_{i=1}^n} \left[\sum_{i \in \mathcal{X}} d_\phi(\pi_i, \mathbf{y}_i) + \alpha \sum_{(i,j) \in \mathcal{X}} s_{ij} d_\phi(\mathbf{y}_i, \mathbf{y}_j) \right]. \quad (2.5)$$

Todos os Divergentes de Bregman possuem a propriedade de que o vetor representativo de um conjunto de vetores é simplesmente o valor esperado para os vetores deste conjunto — dado que o divergente seja computado com este vetor representativo sendo o segundo argumento de $d_\phi(\cdot, \cdot)$. Entretanto, esta forma simples de solução ótima não é válida se a variável a ser

³ Para um tratamento mais geral assume-se neste texto que se dispõe de vários classificadores e vários agrupadores, entretanto, convém notar que os agregadores podem ser formados também por apenas um único componente.

otimizada ocorre como seu primeiro argumento. Neste caso, pode-se trabalhar no espaço dual, onde a solução ótima possui uma forma simples, conforme discutido em (BANERJEE *et al.*, 2005). Examinando a Equação (2.5), pode-se notar que as variáveis sobre os quais a função deve ser minimizada — $\{\mathbf{y}_i\}_{i=1}^n$ — no segundo termo aparecem tanto no primeiro quanto no segundo argumento do Divergente de Bregman. Assim sendo, a otimização sobre $\{\mathbf{y}_i\}_{i=1}^n$ não pode ser obtida de forma fechada. Acharya *et al.* (2011) propuseram uma forma de se contornar esse problema por meio da criação de duas cópias para cada $\mathbf{y}_i$ — a cópia da esquerda, $\mathbf{y}_i^{(l)}$, e a cópia da direita, $\mathbf{y}_i^{(r)}$. As cópias da esquerda (direita) são usadas sempre que as variáveis são encontradas no primeiro (segundo) argumentos dos Divergentes de Bregman. As cópias esquerdas e direitas são atualizadas iterativamente, e uma restrição adicional é usada para assegurar que as duas cópias da variável sejam similares durante as atualizações. Inicialmente, mantendo $\{\mathbf{y}_i^{(l)}\}_{i=1}^n$ e $\{\mathbf{y}_i^{(r)}\}_{i=1}^n \setminus \{\mathbf{y}_j^{(r)}\}$ fixas, a parte da função objetivo que somente depende de $\mathbf{y}_j^{(r)}$ pode ser escrita como:

$$J_{[\mathbf{y}_j^{(r)}]} = d_\phi(\pi_j, \mathbf{y}_j^{(r)}) + \alpha \sum_{i^{(l)} \in \mathcal{X}} s_{i^{(l)} j^{(r)}} d_\phi(\mathbf{y}_i^{(l)}, \mathbf{y}_j^{(r)}). \quad (2.6)$$

Note que a otimização de $J_{[\mathbf{y}_j^{(r)}]}$ em (2.6) com respeito a $\mathbf{y}_j^{(r)}$ deveria teoricamente atender à restrição de que as cópias esquerda e direita de $\mathbf{y}_j$ sejam iguais. Para procurar atender tal restrição, a diferença entre as cópias é penalizada no procedimento de atualização iterativa das variáveis. Mais especificamente, uma restrição é adicionada à função objetivo em (2.6), de tal forma que o problema de otimização passa a ser:

$$\min_{\mathbf{y}_j^{(r)}} \left[d_\phi(\pi_j, \mathbf{y}_j^{(r)}) + \alpha \sum_{i^{(l)} \in \mathcal{X}} s_{i^{(l)} j^{(r)}} d_\phi(\mathbf{y}_i^{(l)}, \mathbf{y}_j^{(r)}) + \lambda_j^{(r)} d_\phi(\mathbf{y}_j^{(l)}, \mathbf{y}_j^{(r)}) \right], \quad (2.7)$$

Na Equação (2.7), $\lambda_j^{(r)}$ é o respectivo termo de penalização (ou multiplicador de Lagrange). Para cada atribuição válida de $\{\mathbf{y}_i^{(l)}\}_{i=1}^n$, Acharya *et al.* (2011) demonstram que existe uma solução única para $\mathbf{y}_j^{(r)*}$ no problema de otimização em (2.7), o qual pode ser obtido da seguinte forma:

$$\mathbf{y}_j^{(r)*} = \frac{\pi_j + \gamma_j^{(r)} \sum_{i^{(l)} \in \mathcal{X}} \delta_{i^{(l)} j^{(r)}} \mathbf{y}_i^{(l)} + \lambda_j^{(r)} \mathbf{y}_j^{(l)}}{1 + \gamma_j^{(r)} + \lambda_j^{(r)}}, \quad (2.8)$$

sendo $\gamma_j^{(r)} = \alpha \sum_{i^{(l)} \in \mathcal{X}} s_{i^{(l)} j^{(r)}}$ e $\delta_{i^{(l)} j^{(r)}} = s_{i^{(l)} j^{(r)}} / \left[\sum_{i^{(l)} \in \mathcal{X}} s_{i^{(l)} j^{(r)}} \right]$. O mesmo procedimento de otimização em (2.7) é repetido sobre todas as variáveis $\mathbf{y}_j^{(r)}$, $j = 1, \dots, n$. Após as cópias direitas terem sido atualizadas, a função objetivo é (sequencialmente) otimizada com relação a todas as variáveis $\mathbf{y}_i^{(l)}$, $i = 1, \dots, n$. Assim como no primeiro passo, $\{\mathbf{y}_j^{(l)}\}_{j=1}^n \setminus \{\mathbf{y}_i^{(l)}\}$ e $\{\mathbf{y}_j^{(r)}\}_{j=1}^n$ são

mantidos fixos, e a diferença entre as cópias direita e esquerda de $\mathbf{y}_i$ é penalizada, de tal forma que a otimização com respeito a $\mathbf{y}_i^{(l)}$ pode ser re-escrita como:

$$\min_{\mathbf{y}_i^{(l)}} \left[\alpha \sum_{j^{(r)} \in \mathcal{X}} s_{i^{(l)} j^{(r)}} d_\phi(\mathbf{y}_i^{(l)}, \mathbf{y}_j^{(r)}) + \lambda_i^{(l)} d_\phi(\mathbf{y}_i^{(l)}, \mathbf{y}_i^{(r)}) \right], \quad (2.9)$$

na qual $\lambda_i^{(l)}$ é o correspondente termo de penalização — cumpre destacar que o algoritmo assume $\lambda_j^{(l)} = \lambda_i^{(r)}$, *i.e.*, as cópias da esquerda e da direita compartilham de um mesmo valor para o multiplicador de Lagrange, o qual deve ser fornecido pelo usuário ou especialista de domínio.

Conforme abordado em (ACHARYA *et al.*, 2011), é necessário se trabalhar com o espaço dual, usando a função convexa $\psi(\cdot)$ — dual de $\phi(\cdot)$ — definida como:

$$\psi(\mathbf{y}_i) = \langle \mathbf{y}_i, \nabla \phi^{-1}(\mathbf{y}_i) \rangle - \phi(\nabla \phi^{-1}(\mathbf{y}_i)). \quad (2.10)$$

O problema de otimização em (2.9) pode então ser re-escrito, em termos do Divergente de Bregman associado com $\psi(\cdot)$, como:

$$\min_{\nabla \phi(\mathbf{y}_i^{(l)})} \left[\alpha \sum_{j^{(r)} \in \mathcal{X}} s_{i^{(l)} j^{(r)}} d_\psi(\nabla \phi(\mathbf{y}_j^{(r)}), \nabla \phi(\mathbf{y}_i^{(l)})) + \lambda_i^{(l)} d_\psi(\nabla \phi(\mathbf{y}_i^{(r)}), \nabla \phi(\mathbf{y}_i^{(l)})) \right]. \quad (2.11)$$

Considerando que $\nabla \phi(\cdot)$ é monotônica e inversível para $\phi(\cdot)$ sendo estritamente convexa, a inversa do minimizador único para o problema de otimização em (2.11) é também única e igual ao minimizador único do problema em (2.9). Desta forma, a solução para o problema em (2.9) com relação à variável $\mathbf{y}_i^{(l)}$ é dada por (ACHARYA *et al.*, 2011):

$$\mathbf{y}_i^{(l)*} = \nabla \phi^{-1} \left[\frac{\gamma_i^{(l)} \sum_{j^{(r)} \in \mathcal{X}} \delta_{i^{(l)} j^{(r)}} \nabla \phi(\mathbf{y}_j^{(r)}) + \lambda_i^{(l)} \nabla \phi(\mathbf{y}_i^{(r)})}{\gamma_i^{(l)} + \lambda_i^{(l)}} \right], \quad (2.12)$$

na qual $\gamma_i^{(l)} = \alpha \sum_{j^{(r)} \in \mathcal{X}} s_{i^{(l)} j^{(r)}}$ e $\delta_{i^{(l)} j^{(r)}} = s_{i^{(l)} j^{(r)}} / \left[\sum_{j^{(r)} \in \mathcal{X}} s_{i^{(l)} j^{(r)}} \right]$. Nos experimentos reportados em (ACHARYA *et al.*, 2011) usou-se o algoritmo C³E (em sua forma geral) empregando o Divergente-I generalizado, que é definido como:

$$d_\phi(\mathbf{y}_i, \mathbf{y}_j) = \sum_{\ell=1}^c y_{i\ell} \log \left(\frac{y_{i\ell}}{y_{j\ell}} \right) - \sum_{\ell=1}^c (y_{i\ell} - y_{j\ell}), \forall \mathbf{y}_i, \mathbf{y}_j \in \mathbb{R}_+^c, \quad (2.13)$$

Neste caso, a função convexa é dada por:

$$\phi(\mathbf{y}_i) = \sum_{\ell=1}^c y_{i\ell} \log(y_{i\ell}), \quad (2.14)$$

e sendo $\nabla\phi(\mathbf{y}_i) = (1 + \log(y_{i\ell}))_{\ell=1}^c$, a Equação (2.12) pode ser re-escrita como:

$$\mathbf{y}_i^{(l)*,I} = \exp \left(\frac{\gamma_i^{(l)} \sum_{j^{(r)} \in \mathcal{X}} \delta_{i(l)j^{(r)}} \nabla\phi(\mathbf{y}_j^{(r)}) + \lambda_i^{(l)} \nabla\phi(\mathbf{y}_i^{(r)})}{\gamma_i^{(l)} + \lambda_i^{(l)}} \right) - \mathbf{1}, \quad (2.15)$$

Na Equação (2.15) o super-escrito I indica que o valor ótimo corresponde ao Divergente-I. A otimização sobre os argumentos esquerdos e direitos de todos os objetos se constitui numa iteração do algoritmo. Tais iterações são repetidas até a convergência. Dado que em cada passo o algoritmo minimiza a função objetivo em (2.5), cujo limite inferior é 0, e considerando que o minimizador é único devido à convexidade estrita de $\phi(\cdot)$ e de seu dual, o algoritmo possui garantias de convergência (ACHARYA *et al.*, 2011). Após o algoritmo convergir, todas as variáveis $\mathbf{y}_i, i = 1, \dots, n$ são normalizadas de acordo com a norma L_1 — após computar a média das cópias direitas e esquerdas. Desta forma, são obtidas distribuições de probabilidades de classes para cada objeto $\mathbf{x}_i \in \mathcal{X}$. Os principais passos do algoritmo C³E podem então ser sumarizados por meio do Algoritmo 1, cuja complexidade computacional é analisada na Seção 2.1.1.

Algoritmo 1: C³E (ACHARYA *et al.*, 2011)

Entrada: $\{\pi_i\}, \mathbf{S}$

Saída: $\{\mathbf{y}_i\}$

- 1 Inicializar $\{\mathbf{y}_i^{(r)}\}, \{\mathbf{y}_i^{(l)}\}$ tal que $\mathbf{y}_{i\ell}^{(r)} = \mathbf{y}_{i\ell}^{(l)} = \frac{1}{c} \forall i \in \{1, 2, \dots, n\}, \forall \ell \in \{1, 2, \dots, c\}$;
 - 2 **Repita**
 - 3 Atualizar $\mathbf{y}_j^{(r)}$ usando a Equação (2.8) $\forall j \in \{1, 2, \dots, n\}$;
 - 4 Atualizar $\mathbf{y}_i^{(l)}$ usando a Equação (2.12) $\forall i \in \{1, 2, \dots, n\}$;
 - 5 **até a convergência**;
 - 6 Computar $\mathbf{y}_i = 0.5[\mathbf{y}_i^{(l)} + \mathbf{y}_i^{(r)}] \forall i \in \{1, 2, \dots, n\}$;
 - 7 Normalizar $\mathbf{y}_i \forall i \in \{1, 2, \dots, n\}$;
-

A escolha do Divergente de Bregman para cada aplicação é dependente dos dados, e é difícil antecipar quais seriam os divergentes mais apropriados para cada caso particular. Na prática, pode-se experimentar com diversos divergentes e então escolher aquele que apresenta os melhores resultados. Conforme será abordado na Seção 2.1.2, o algoritmo de otimização pode ser simplificado ao se escolher a função de Perda Quadrática.

2.1.1 Análise de Complexidade Computacional

Considerando que um agregador de classificadores esteja disponível, o cálculo do conjunto de vetores $\{\pi_i\}_{i=1}^n$ requer $O(n \cdot r_1 \cdot c)$ operações, sendo n o número de objetos no conjunto alvo, r_1 o número de componentes do agregador de classificadores e c o número de rótulos de classe. Para computar a matriz de similaridades $\mathbf{S}$ são necessárias $O(r_2 \cdot n^2)$ operações, sendo r_2 o

número de componentes do agregador de agrupadores. Finalmente, tendo $\{\pi_i\}_{i=1}^n$ e $\mathbf{S}$ disponíveis, o custo computacional (por iteração) do C^3E é $O(c \cdot n^2)$. Na verdade, o gargalo computacional do C^3E não é o algoritmo de otimização propriamente dito, cujos passos principais podem ser paralelizados, mas a computação da matriz de similaridades. Note, entretanto, que esta é computada para o conjunto alvo e independe do conjunto de treinamento.

Obviamente que, ao se considerar o processo de mineração de dados como um todo, também é necessário levar em conta o custo computacional para se induzir os classificadores e os agrupadores que farão parte dos agregadores. Este custo computacional depende dos componentes escolhidos para formar os agregadores. Por exemplo, ao se utilizar um classificador Bayesiano simples, a complexidade de tempo computacional para induzir o classificador é linear tanto com relação ao número de atributos quanto com relação ao número de objetos do conjunto de treinamento. Classificadores mais custosos podem ser induzidos de forma paralela. De maneira análoga, o algoritmo K-Médias possui complexidade de tempo linear em relação ao número de objetos e de atributos (neste caso com respeito ao conjunto alvo) e, analogamente ao que se observa em relação à indução do agregador de classificadores, é possível obter partições do conjunto alvo em paralelo (*e.g.*, rodando um algoritmo de agrupamento por processador).

2.1.2 C^3E com Função de Perda Quadrática (C^3E -SL)

Caso se opte por usar uma função de Perda Quadrática, é possível simplificar o algoritmo C^3E . Este caso particular, que não foi explicitamente estudado em (ACHARYA *et al.*, 2011), considera que o problema de otimização em (2.4) possa ser reformulado como:

$$J_{PQ} = \frac{1}{2} \sum_{i \in \mathcal{X}} \|\mathbf{y}_i - \pi_i\|^2 + \alpha \frac{1}{2} \sum_{(i,j) \in \mathcal{X}} s_{ij} \|\mathbf{y}_i - \mathbf{y}_j\|^2 \quad (2.16)$$

O segundo termo dessa função objetivo mostra que as variáveis são acopladas. Para contornar essa dificuldade, e seguindo-se uma abordagem análoga àquela adotada para o caso mais geral, pode-se projetar um procedimento de atualização iterativo das variáveis a serem otimizadas. Mais especificamente, mantendo-se $\{\mathbf{y}_j\}_{j=1}^n \setminus \{\mathbf{y}_i\}$ fixos, pode-se minimizar J_{PQ} para cada $\mathbf{y}_i$ ao estabelecer:

$$\frac{\partial J_{PQ}}{\partial \mathbf{y}_i} = \mathbf{0}. \quad (2.17)$$

Observando-se que $\frac{\partial \|\mathbf{x}\|^2}{\partial \mathbf{x}} = 2\mathbf{x}$ e que a matriz de similaridades $\mathbf{S}$ é simétrica, obtém-se:

$$\mathbf{y}_i = \frac{\pi_i + \alpha' \sum_{j \neq i} s_{ij} \mathbf{y}_j}{1 + \alpha' \sum_{j \neq i} s_{ij}}, \quad (2.18)$$

na qual $\alpha' = 2\alpha$. De acordo como feito para o caso mais geral, a Equação (2.18) pode ser aplicada iterativamente. O Algoritmo 2 sumariza o C³E com uso da função de Perda Quadrática (C³E-SL). Dado que cada atualização de $\mathbf{y}_i$ contribui para minimizar J_{PQ} , o algoritmo converge. Note que, para este caso particular, os parâmetros $\lambda_i^{(l)}$ e $\lambda_j^{(r)}$ não são mais necessários, o que evita a necessidade de fornecimento do parâmetro $\lambda = \lambda_i^{(l)} = \lambda_j^{(r)}$ ao algoritmo. Além disso, semelhantemente ao que acontece no caso mais geral, válido para qualquer Divergente de Bregman, no processo de minimização de J_{PQ} , a atualização de $\{\mathbf{y}_i\}_{i=1}^n$ pode ser executada em paralelo (atualizando-se uma ou mais variáveis por processador). Pode-se observar, portanto, que o uso da Perda Quadrática torna o algoritmo de otimização bastante atraente para aplicações nas quais a eficiência computacional é de particular relevância. A avaliação empírica reportada a seguir mostra que o C³E-SL produz resultados competitivos e com chances de melhorias.

Algoritmo 2: C³E com função de Perda Quadrática (C³E-SL)

Entrada: $\{\pi_i\}, S$

Saída: $\{\mathbf{y}_i\}$

- 1 Inicializar $\{\mathbf{y}_i\}$ tal que $y_{i\ell} = \frac{1}{c} \forall i \in \{1, 2, \dots, n\}, \forall \ell \in \{1, 2, \dots, c\}$;
 - 2 **Repita**
 - 3 | Atualizar $\mathbf{y}_i \forall i \in \{1, 2, \dots, n\}$ usando a Equação (2.18);
 - 4 **até a convergência;**
-

2.2 Avaliação Empírica

Em (ACHARYA *et al.*, 2011), nenhum estudo sobre a sensibilidade paramétrica do C³E foi apresentado, de maneira que os principais parâmetros do algoritmo foram apenas manualmente definidos. Adicionalmente, as avaliações foram realizadas a partir de resultados de *ensembles* já disponíveis; e em um número limitado de bases de dados. Portanto, preliminarmente, na presente pesquisa, buscou-se investigar como o C³E responde à variação de seus parâmetros. A versão simplificada do algoritmo, apresentada na Seção 2.1.2 (C³E-SL), foi também avaliada.

O algoritmo C³E (ACHARYA *et al.*, 2011) necessita que o usuário informe o valor de três parâmetros principais, a saber: o multiplicador de Lagrange, $\lambda \in \mathbb{R}_+$, um parâmetro que controla a importância do agregador de classificadores em relação ao agregador de agrupadores, $\alpha \in \mathbb{R}_+$, e o número de iterações do algoritmo, $I \in \mathbb{Z}_+$. Alternativamente, o parâmetro I pode ser omitido ao se definir um critério de parada para o algoritmo. Neste sentido, nestes experimentos preliminares, foi definido um limiar $\varepsilon = 10^{-10}$ de diferença mínima da função objetivo em (2.4) calculada em duas iterações consecutivas. Isto possibilitou avaliar o comportamento do algoritmo ao se variar os outros dois parâmetros: λ e α .

Foram utilizadas 16 bases de dados reais (11 dessas referem-se à problemas de classificação de documentos — *News1*, ..., *News6*, *Cora1*, ..., *Cora4* e *DBLP* — empregou-se as mesmas probabilidades de classes iniciais e matrizes de similaridades usadas em (ACHARYA *et*

al., 2011) e (GAO *et al.*, 2009); as outras 5 bases são do repositório de dados da UCI (FRANK; ASUNCION, 2010) — *Wisconsin, Pima, Yeast, Spambase e Ionosphere*). Para as bases de dados da UCI, o agregador de classificadores foi composto pelos algoritmos *Naive Bayes* (NB), *Árvore de Decisão* (J48) e *1-Vizinho Mais Próximo* (IB1); e o agregador de agrupadores teve como componentes os algoritmos *K-Médias* (KM) e *Expectation Maximization* (EM)⁴ — o número de grupos k informado foi igual ao número de classes (c) do problema. A capacidade de generalização do C³E foi estimada por meio do procedimento de validação cruzada de 10 pastas (*10-fold cross-validation*) (WITTEN; FRANK, 2005)⁵.

A Figura 2 ilustra superfícies de acurácia — taxa de classificação correta (média) — traçadas pelos diferentes valores de $\lambda = \{0, 10^{-4}, 5 \times 10^{-4}, 10^{-3}, 5 \times 10^{-3}, 10^{-2}, 5 \times 10^{-2}, 10^{-1}, 5 \times 10^{-1}, 1\}$ e $\alpha = \{0, 0.01, 0.02, 0.03, \dots, 1\}$ informados ao algoritmo C³E. Tais superfícies são representativas do comportamento do algoritmo. Observa-se que a variação de λ , ao contrário da variação de α , não influencia significativamente nos resultados de classificação do C³E. Este fato sugere que α possa ser um parâmetro de maior importância e que, a partir de maiores estudos, possa alavancar melhores resultados para o algoritmo.

Objetivando-se avaliar a acurácia de classificação em função do número de iterações do algoritmo conduziu-se um experimento, no qual fixou-se valores considerados “bons” para α e o valor de I foi variado até 1000⁶. A Figura 3 sumariza os resultados. Convém destacar que, para uma dada iteração I , a acurácia média reportada refere-se à média computada a partir dos resultados gerados pelos diferentes valores de λ . Dada a pouca influência de λ nos resultados finais, estas médias estão muito próximas das acurácias obtidas para qualquer valor (entre 0 e 1) deste parâmetro. Nas figuras, a reta tracejada indica a acurácia obtida pelo agregador de classificadores e o losango indica o número de iterações do algoritmo de acordo com o critério de parada definido no início desta seção (limiar $\varepsilon = 10^{-10}$ de diferença mínima da função objetivo em duas iterações consecutivas). Tipicamente, após um determinado número de iterações do C³E, a acurácia passou a diminuir. Na maioria destes casos esta diminuição proporcionou resultados inferiores aos obtidos pelo agregador de classificadores. Em todas as bases de dados (exceto *News5*), a maior acurácia foi alcançada em até 20 iterações. Portanto, melhores acurácias são obtidas a partir de um número de iterações relativamente baixo; e nem sempre o critério de parada definido produz os melhores resultados (veja a posição dos losangos nas figuras). Estas observações sugerem que o algoritmo é significativamente sensível ao parâmetro I e a sua estimativa (automática) para um dado problema é considerada bastante relevante.

Os experimentos reportados até o momento sugerem que além do parâmetro α , o número de iterações, I , também influencia significativamente nos resultados de classificação do C³E.

⁴ Os agregadores (*ensembles*) foram implementados como uma extensão do *software* para mineração de dados Weka disponível em <http://www.cs.waikato.ac.nz/ml/weka/>.

⁵ A implementação computacional em Matlab do C³E em seu caso mais geral (a qual usa o Divergente-I como função de perda) foi gentilmente cedida pelos autores do algoritmo.

⁶ Os valores de α considerados interessantes possuem alta variância e foram definidos manualmente a partir de inspeção prévia dos resultados fornecidos pelo algoritmo.

Figura 2 – Acurácias de classificação (%) do C³E a partir da combinação de diferentes valores dos parâmetros λ e α respectivamente para as bases de dados *News5* (a), *Cora4* (b) e *Pima* (c).

Cumpra observar também que os experimentos realizados até o momento não exploraram completamente o espaço de soluções proporcionado pelos diferentes valores de cada um destes parâmetros. Dado que o C³E com Perda Quadrática (C³E-SL) suprime a necessidade de uso do multiplicador de Lagrange (λ), tal algoritmo abre espaço para uma análise mais criteriosa dos impactos da variação destes parâmetros em seu processo de otimização⁷.

⁷ Como notado, λ aparentemente oferece pouca utilidade prática na versão mais geral do algoritmo.

Figura 3 – Acurácias médias (%) de acordo com o número de iterações (I) do algoritmo C³E respectivamente para as bases de dados *News1* (a), *DBLP* (b), *Cora4* (c) e *Spam* (d).

A Tabela 1 apresenta as melhores acurácias obtidas nas bases de dados de classificação de documentos ao se variar $\alpha = \{0, 0.001, 0.002, \dots, 1\}$ e $I = \{1, 2, 3, \dots, 50\}$ no algoritmo C³E (assumiu-se $\lambda = 1$) e na sua versão simplificada (com Perda Quadrática — C³E-SL). Para fins de comparação, também são apresentadas as acurácias obtidas pelo agregador de classificadores. Esta tabela mostra que o algoritmo C³E foi capaz de fornecer melhorias nos resultados de classificação independentemente da função de perda usada (Divergente-I, em seu caso mais geral, ou Perda Quadrática). Entretanto, o C³E-SL foi melhor ou igual ao C³E em 73% das bases de dados. Em todos os casos não foi necessário mais do que 14 iterações para se alcançar os resultados.

Experimentos mais elaborados foram conduzidos em 16 bases de dados do repositório de dados da UCI (FRANK; ASUNCION, 2010). Duas agregações de classificadores distintas foram avaliadas: NB+J48, que tem como componentes os classificadores *Naive Bayes* (NB) e *Árvore de Decisão* (J48); e NB+IB5, que tem como componentes o *Naive Bayes* (NB) e 5-Vizinhos Mais Próximos (IB5). Duas diferentes agregações de agrupadores também foram avaliadas. Tais agregações têm em comum o uso de dois componentes, os quais são resultados de duas versões

Tabela 1 – Acurácias de classificação do agregador de classificadores (NB+J48+IB1) e dos algoritmos C³E e C³E-SL. Os melhores resultados estão em negrito.

Base de Dados	Agregador de Classificadores	C ³ E	C ³ E-SL
<i>News1</i>	78.19	85.20	85.27
<i>News2</i>	87.22	94.02	94.08
<i>News3</i>	82.39	91.42	91.42
<i>News4</i>	86.39	94.07	94.07
<i>News5</i>	85.67	92.87	92.11
<i>News6</i>	86.31	92.72	92.33
<i>Cora1</i>	86.43	89.14	89.44
<i>Cora2</i>	90.79	90.89	91.40
<i>Cora3</i>	81.61	91.89	91.89
<i>Cora4</i>	88.00	92.41	91.69
<i>DBLP</i>	91.05	93.81	94.76

particulares do K-Médias (KM); uma que agrupa os dados em k grupos, e outra que agrupa os dados em $2k$ grupos. Os agregadores de agrupadores diferenciam-se entre si pela forma com que definem o valor de k . Um assume que k é igual ao número de classes do problema (c), e o outro assume que k foi estimado por meio de um índice de validade relativo (capaz de estimar o número de grupos da partição — *e.g.*, Silhueta Simplificada (CAMPELLO; HRUSCHKA, 2006; HRUSCHKA; CAMPELLO; CASTRO, 2006)). Utilizou-se o procedimento de validação cruzada de 10 pastas (WITTEN; FRANK, 2005).

A Tabela 2 apresenta as acurácias de classificação (taxa de classificação correta média) do agregador de classificadores NB+J48 em comparação com os melhores resultados — pela variação de $\alpha = \{0, 0.001, 0.002, \dots, 1\}$ e $I = \{1, 2, 3, \dots, 50\}$ — obtidos pelos algoritmos C³E e C³E-SL, cada qual processando resultados de cada um dos dois agregadores de agrupadores disponíveis⁸. Os desvios padrões estão entre parenteses. A Tabela 3 reporta a mesma análise, porém considera os resultados obtidos a partir do agregador de classificadores NB+IB5.

Observando-se as tabelas, nota-se que ao se utilizar agrupadores com k estimado há maiores chances de se obter melhores resultados. Além disso, o algoritmo C³E-SL mostrou-se competitivo. Esta constatação indica que abordagens que usam função de Perda Quadrática (e processam resultados de agrupadores com k estimado a partir de algum índice) podem render modelos de classificação bastante atraentes. Portanto, levando-se em conta a maior simplicidade do algoritmo baseado em Perda Quadrática (C³E-SL), acredita-se que investir esforços em tal instanciação pode ser de grande utilidade prática.

⁸ Os resultados reportados para cada base de dados correspondem à média das maiores acurácias alcançadas em cada uma das 10 pastas — α e I foi variado em cada pasta; e assim cada pasta teve a sua própria combinação de α^* e I^* (“ótimos”).

Tabela 2 – Acurácias médias do agregador de classificadores (NB+J48) e as diferentes versões do C³E. Os melhores resultados estão em negrito.

Base de Dados	Agregador de Classificadores	C ³ E		C ³ E-SL	
		$k = c$	k estimado	$k = c$	k estimado
Wisc	95.61 (2.0)	96.92 (1.1)	97.07 (1.4)	97.07 (1.7)	97.07 (1.4)
Pima	73.05 (4.6)	77.08 (3.6)	76.69 (4.0)	75.52 (3.6)	77.99 (2.5)
Yeast	96.12 (3.0)	98.02 (2.6)	98.50 (2.4)	98.02 (2.6)	98.50 (2.4)
Spam	81.26 (1.5)	91.63 (4.5)	93.31 (1.1)	92.07 (4.1)	91.89 (4.1)
Iono	88.89 (5.1)	92.03 (3.7)	92.89 (4.1)	91.75 (3.6)	92.31 (3.3)
Iris	96.00 (5.6)	96.67 (4.7)	97.33 (3.4)	96.67 (4.7)	96.67 (4.7)
WRed	56.85 (5.7)	59.23 (6.0)	59.67 (6.8)	59.23 (6.1)	59.79 (6.8)
WWhite	58.13 (2.3)	58.31 (2.3)	58.82 (2.0)	58.23 (2.3)	58.80 (2.1)
Blood	76.07 (3.1)	78.61 (1.8)	78.34 (1.8)	78.61 (1.8)	78.08 (2.1)
Glass	64.96 (10.4)	67.32 (10.9)	67.77 (9.8)	66.86 (10.6)	66.82 (10.1)
Sonar	70.67 (7.7)	76.98 (6.5)	76.45 (7.9)	74.10 (5.3)	74.10 (4.9)
ImSeg	92.55 (2.3)	92.55 (2.3)	92.55 (2.3)	92.55 (2.3)	92.55 (2.3)
Vehicle	71.28 (6.9)	71.28 (6.9)	71.28 (6.9)	71.28 (6.9)	71.28 (6.9)
VertCol	81.94 (5.1)	85.48 (3.8)	86.13 (3.7)	85.81 (4.1)	86.77 (4.4)
Wave	77.16 (1.2)	79.70 (2.0)	79.22 (2.0)	80.28 (2.1)	79.58 (2.1)
PBlocks	94.96 (0.9)	95.54 (0.8)	95.36 (0.6)	95.60 (0.8)	95.71 (0.4)

Tabela 3 – Acurácias médias do agregador de classificadores (NB+IB5) e as diferentes versões do C³E. Os melhores resultados estão em negrito.

Base de Dados	Agregador de Classificadores	C ³ E		C ³ E-SL	
		$k = c$	k estimado	$k = c$	k estimado
Wisc	96.34 (2.0)	97.65 (1.2)	97.51 (1.2)	97.36 (1.4)	97.36 (1.4)
Pima	76.82 (3.4)	77.99 (3.2)	77.86 (3.9)	77.73 (3.6)	78.12 (3.3)
Yeast	97.52 (2.6)	97.52 (2.6)	98.00 (2.6)	97.52 (2.6)	98.00 (2.6)
Spam	80.07 (1.5)	90.72 (1.9)	90.89 (0.9)	91.15 (1.6)	91.39 (0.8)
Iono	91.44 (4.7)	92.87 (4.5)	92.59 (4.9)	92.87 (4.5)	92.87 (4.5)
Iris	95.33 (4.5)	96.67 (3.5)	96.00 (3.4)	96.00 (3.4)	96.00 (3.4)
WRed	50.16 (7.3)	53.28 (6.8)	55.41 (7.5)	52.78 (7.4)	55.85 (6.4)
WWhite	50.96 (2.7)	51.74 (2.3)	53.55 (1.8)	51.98 (2.5)	53.25 (1.9)
Blood	77.00 (2.0)	78.34 (1.6)	78.48 (1.6)	78.21 (1.9)	78.74 (1.6)
Glass	62.64 (6.7)	64.50 (8.1)	64.98 (8.1)	64.98 (7.2)	64.98 (8.4)
Sonar	78.45 (8.9)	84.64 (5.8)	83.21 (4.5)	81.33 (7.5)	80.83 (6.6)
ImSeg	92.51 (1.4)	92.51 (1.4)	92.51 (1.4)	92.51 (1.4)	92.51 (1.4)
Vehicle	65.37 (4.2)	65.37 (4.2)	65.37 (4.2)	65.37 (4.2)	65.37 (4.2)
VertCol	81.61 (6.8)	83.23 (6.6)	83.23 (6.6)	83.23 (6.8)	83.23 (7.3)
Wave	81.06 (2.1)	81.08 (2.1)	81.06 (2.1)	81.06 (2.1)	81.06 (2.1)
PBlocks	94.10 (0.7)	94.72 (0.8)	94.41 (0.5)	94.65 (0.9)	94.50 (0.7)

2.3 Considerações Finais

A agregação de classificadores e agrupadores tem como motivação as diversas vantagens inerentes a cada um desses modelos (ACHARYA *et al.*, 2011; CAI; CHEN; ZHANG, 2009; GAO *et al.*, 2009). Em especial, modelos para aprendizado não supervisionado podem aumentar a precisão e a capacidade de generalização de classificadores. Baseando-se nisso, o algoritmo C³E, proposto por Acharya *et al.* (2011), visa obter uma classificação consolidada dos dados a partir

da combinação de dois modelos: um modelo resultante da agregação de classificadores (OZA; TUMER, 2008; KUNCHEVA, 2004) e outro resultante da agregação de agrupadores (GHOSH; ACHARYA, 2011; GHAEMI *et al.*, 2009). Na Seção 2.1.2, foi apresentada uma simplificação deste algoritmo por meio do uso da função de Perda Quadrática (C³E-SL).

O C³E-SL evita a necessidade de informação, por parte do usuário, do multiplicador de Lagrange (λ). Entretanto, outros parâmetros, como α , e também o número de iterações (I) ainda são necessários. Experimentos reportados na Seção 2.2 indicaram que estes parâmetros têm grande influência no resultado final do algoritmo. Considerando que em (ACHARYA *et al.*, 2011) não foi definido nenhum critério sistemático para o ajuste de tais parâmetros, para tornar o C³E-SL menos dependente do usuário, investigou-se formas de se estimar automaticamente (a partir dos dados) os valores de α e I .

O próximo capítulo aborda os principais conceitos sobre meta-heurísticas de busca, as quais são particularmente adequadas para o problema de otimização paramétrica (e.g., otimização de parâmetros em métodos de aprendizado de máquina). Este estudo resultou na concepção de um novo algoritmo evolutivo, o qual tem sido usado para a calibração automática do algoritmo C³E-SL (COLETTA *et al.*, 2015a; COLETTA *et al.*, 2014).

META-HEURÍSTICAS PARA COMBINAR CLASSIFICADORES E AGRUPADORES

O algoritmo C^3E (ACHARYA *et al.*, 2014; ACHARYA *et al.*, 2011) tem apresentado importantes resultados empíricos, sobretudo em sua versão simplificada (C^3E -SL), conforme reportado na Seção 2.2. Entretanto, o C^3E -SL depende explicitamente da informação do parâmetro α , que controla a importância do agregador de classificadores em relação ao agregador de agrupadores, e também do número de iterações, I , do algoritmo. Para melhorar os resultados do C^3E -SL, estes parâmetros são comumente ajustados de maneira manual (pelo usuário/especialista de domínio). Contudo, os valores mais adequados para estes parâmetros variam significativamente de acordo com o problema tratado. Dada esta dificuldade, buscou-se alternativas para se ajustar α e I automaticamente (a partir dos dados), visto que, nos trabalhos que introduziram o algoritmo C^3E (ACHARYA *et al.*, 2014; ACHARYA *et al.*, 2011), nenhuma abordagem sistemática para a estimação desses parâmetros foi investigada. Neste capítulo, portanto, estuda-se a otimização paramétrica do C^3E -SL usando-se meta-heurísticas de busca.

Meta-heurísticas consistem de algoritmos estocásticos que se utilizam de aleatoriedade e busca local para tentar encontrar uma solução ótima global (YANG, 2010). Tais algoritmos exploram (de forma guiada) o espaço de busca sem assumir que este seja convexo ou contínuo — sendo, portanto, uma boa alternativa para problemas não convexos e com condições de diferenciabilidade não garantidas. Vários estudos têm mostrado que meta-heurísticas são capazes de evitar mínimos locais, encontrando boas soluções em tempo computacional razoável para muitos problemas complexos de otimização — incluindo a estimação de valores de parâmetros requeridos por aplicações de aprendizado de máquina (EIBEN; SMITH, 2008; BÄCK; FOGEL; MICHALEWICZ, 1997; FLEMING; PURSHOUSE, 2002; FREITAS, 2003). Dada uma função objetivo (também chamada de *função de fitness*) que se deseja minimizar (ou maximizar) com relação a um conjunto de parâmetros, meta-heurísticas são particularmente úteis na identificação dos valores de parâmetros para os quais a função é minimizada (ou maximizada).

A metodologia de pesquisa adotada envolveu dois passos: (i) avaliar experimentalmente meta-heurísticas existentes no problema específico de otimização de parâmetros do C³E-SL (Seções 3.1 e 3.2); e (ii) a partir da análise dos resultados experimentais obtidos, propor aperfeiçoamentos na meta-heurística que se mostrou mais promissora em (i), derivando-se um algoritmo particularmente adaptado para o problema enfrentado. Esse novo algoritmo, o qual se mostrou útil para o problema em questão, é descrito e avaliado, respectivamente, nas Seções 3.3 e 3.4.

3.1 Meta-heurísticas para Otimização Paramétrica

Nas últimas décadas, diversas meta-heurísticas inspiradas na natureza têm sido desenvolvidas, destacando-se os Algoritmos Evolutivos (AEs) (BäCK; HOFFMEISTER; SCHWEFEL, 1991; BäCK; SCHWEFEL, 1993), os algoritmos concebidos a partir das leis da termodinâmica (KIRKPATRICK; GELATT; VECCHI, 1983), bem como os algoritmos baseados em inteligência de enxames (KENNEDY; EBERHART, 1995) e computação memética (CARAFFINI *et al.*, 2013; CARAFFINI; NERI; PICINALI, 2014). Esses algoritmos têm sido aplicados com sucesso em inúmeros problemas práticos. Por exemplo, AEs têm sido muito úteis na mineração de dados (LUNA *et al.*, 2014; MENÉNDEZ; BARRERO; CAMACHO, 2014; REYES; MORELL; VENTURA, 2014; CAMPOMANES-ÁLVAREZ; CORDÓN; DAMAS, 2013; PEDRINO *et al.*, 2013), bem como na construção de redes *wireless* (MOLINA-GARCÍA *et al.*, 2014), em processos distribuídos (JIA; WANG; FAN, 2014) e segurança contra terremotos (HEJAZI *et al.*, 2013). Semelhantemente, técnicas de inteligência de enxames têm sido usadas na otimização de caminhos para rotas de evacuação (FORCAEL *et al.*, 2014), transporte de cargas (ZENG *et al.*, 2014) e na classificação de sinais (HSU, 2013). Informações mais detalhadas sobre estas meta-heurísticas e suas aplicações podem ser encontradas em (DOWSLAND; THOMPSON, 2012; DAS; SUGANTHAN, 2011; CLERC, 2010; YANG, 2010; FLEMING; PURSHOUSE, 2002; FREITAS, 2002).

Embora meta-heurísticas têm sido utilizadas com sucesso em dezenas de aplicações, até mesmo na otimização de parâmetros requeridos por ferramentas de aprendizado de máquina, tipicamente esses algoritmos de busca dependem também de *parâmetros de controle* que precisam ser informados pelo usuário. Geralmente, argumenta-se que meta-heurísticas são robustas a variação de seus parâmetros de controle, os quais muitas vezes já possuem seus valores (“ótimos”) afixados (pelos autores do algoritmo), ou então tais parâmetros são relativamente fáceis de se ajustar. No entanto, vários estudos têm mostrado que meta-heurísticas bem tradicionais são, de fato, bem sensíveis a variação de seus parâmetros de controle (MALLIPEDDI *et al.*, 2011; WANG; CAI; ZHANG, 2011; QIN; HUANG; SUGANTHAN, 2009; DONG; WANG; CHEN, 2008; BREST *et al.*, 2006; LIANG *et al.*, 2006; HANSEN; OSTERMEIER, 2001; SHI; EBERHART, 2001; INGBER *et al.*, 1996). Sob esta perspectiva, esforços consideráveis têm sido investidos no desenvolvimento de meta-heurísticas que possam se autoadaptar, ajustando automaticamente seus parâmetros de controle durante a sua execução. Esta ideia torna o algoritmo

menos dependente do usuário (ou do especialista de domínio). Em outras palavras, torna o algoritmo mais prático e de fácil utilização — evitando o tempo adicional gasto com análises e investigações empíricas voltadas para o ajuste fino de parâmetros. Como exemplo, o *Self-adaptive Differential Evolution* (SaDE) (QIN; HUANG; SUGANTHAN, 2009) é uma extensão autoadaptativa bem difundida do algoritmo *Differential Evolution* (DE) (PRICE, 1996). A partir desse contexto, as meta-heurísticas estudadas para otimizar os parâmetros do C³E-SL foram:

(1+1)-ES: Introduzido por Rechenberg (1973) e Schwefel (1981), este algoritmo é inspirado no princípio da evolução biológica e faz parte das “Estratégias Evolutivas” concebidas para problemas de otimização de parâmetros. Neste algoritmo, um único indivíduo (solução candidata para um determinado problema) gera um único descendente por meio de uma mutação baseada numa distribuição Gaussiana. Então, o melhor dentre eles serve como ancestral para as gerações seguintes, portanto contribuindo para a evolução das soluções. Tipicamente, dois parâmetros devem ser informados para se regular a taxa de convergência (c_d e c_i), os quais são baseados na regra “1/5-success” de Rechenberg (1973). Empiricamente, $c_d = 0.82$ e $c_i = 1.22$ têm se mostrado bons valores para vários problemas (SCHWEFEL, 1981).

CMA-ES: Formalizado por Hansen e Ostermeier (2001), é uma extensão das “Estratégias Evolutivas” introduzidas por Schwefel (1981), as quais relacionam a cada indivíduo, além de um vetor solução e um desvio padrão, também uma matriz de covariâncias. Esta matriz captura a relação de interdependência entre as variáveis do problema, e sua adaptação corresponde à evolução das soluções. As covariâncias são obtidas empiricamente via amostragem de indivíduos. O CMA-ES exige explicitamente apenas a informação do número de indivíduos da população (P). Experimentos práticos tem mostrado que uma população com poucos indivíduos favorece a rápida convergência. Seguindo estudos prévios, neste trabalho foi adotado $P = 20$ (HANSEN; OSTERMEIER, 2001; HANSEN, 2006).

Differential Evolution (DE): Proposto por Price (1996), este algoritmo evolui uma população de indivíduos (soluções) visando, em geral, otimizar parâmetros de funções. Realiza mutação ao selecionar dois indivíduos aleatoriamente, $\mathbf{z}_\beta$ e $\mathbf{z}_\gamma$, e mais um outro indivíduo aleatório, $\mathbf{z}_\theta$, ou, o *melhor* da população corrente ($\mathbf{z}_{best}$), para se conceber um descendente $\mathbf{v}^{t+1}$ a partir de uma variação baseada na Equação (3.1):

$$\mathbf{v}^{t+1} = \mathbf{z}_{\theta \text{ ou } best}^t + F \cdot (\mathbf{z}_\beta^t - \mathbf{z}_\gamma^t), \quad (3.1)$$

na qual F é um parâmetro real que controla a amplitude da diferença vetorial. É possível aumentar a diversidade da população multiplicando-se mais uma diferença entre outros dois indivíduos aleatórios no segundo termo a direita da Equação (3.1) (PRICE; STORN; LAMPINEN, 2005). A partir disso, o operador de cruzamento seleciona (aleatoriamente)

um outro indivíduo (alvo), $\mathbf{z}_a^t$, e utiliza o indivíduo (doador), $\mathbf{v}^{t+1}$, em um procedimento denominado cruzamento binomial, o qual combina “código genético” para se construir um indivíduo experimental, $\mathbf{u}^{t+1}$, de acordo com a regra:

$$u_i^{t+1} = \begin{cases} v_i^{t+1} & \text{se } r_i \leq Cr \text{ ou } i = L \\ z_i^t & \text{se } r_i > Cr \text{ e } i \neq L \end{cases}, \quad (3.2)$$

na qual i é a posição de um dado *gene* (elemento do vetor), $r_i \in [0,1]$ é um valor aleatório obtido a partir de uma distribuição uniforme, $Cr \in [0,1]$ é uma constante definida pelo usuário e L é um número aleatório entre 1 e o número total de genes. Quando $Cr = 1$ todos os genes (elementos) do vetor experimental virão do vetor doador $\mathbf{v}^{t+1}$; quando $Cr = 0$, todos os genes virão do vetor alvo ($\mathbf{z}_a^t$). Seleciona-se para a próxima geração o indivíduo mais promissor entre $\mathbf{u}^{t+1}$ e $\mathbf{z}_a^t$. Esta decisão leva em conta a função de *fitness* projetada e usada para medir a qualidade dos indivíduos. Considerando a combinação das diferentes estratégias de mutação e cruzamento, a evolução diferencial pode assumir diferentes versões — *e.g.*, *DE/best/2/bin* (STORN; PRICE, 1997). Normalmente, estas versões necessitam da informação de $F \in [0.4, 1.2]$ e $Cr \in [0.8, 1]$ (STORN, 1996; STORN; PRICE, 1997; RONKKONEN; KUKKONEN; PRICE, 2005), sendo inicialmente recomendado $F = 0.5$ e $Cr = 0.9$ (STORN; PRICE, 1997; PRICE, 1996). O número de indivíduos adotado nesta tese foi $P = 20$.

Particle Swarm Optimization (PSO): É um algoritmo de otimização inspirado em comportamento social (de pássaros, peixes, etc.) proposto por Kennedy e Eberhart (1995). Como os algoritmos evolutivos, o PSO inicia a exploração do espaço de busca a partir de uma população aleatória de soluções candidatas, mas, não emprega operadores de cruzamento ou mutação. Ao invés disso, relaciona a cada indivíduo (partícula) uma velocidade para percorrer o espaço. Cada partícula, durante seu percurso, é atualizada com a informação sobre a melhor posição que encontrou (*pbest*), bem como a melhor posição encontrada considerando todas as partículas (*gbest*). A cada passo, cada partícula tem sua velocidade atualizada, de forma a acelerar seu percurso em direção as melhores posições (mínimos locais). As velocidades em cada dimensão são limitadas por um valor máximo, o qual influencia na precisão com que as regiões próximas às soluções correntes serão exploradas (valores altos priorizam a busca global, enquanto que valores menores enfatizam a busca local). Nesta pesquisa, fixou-se a velocidade máxima para cada dimensão do espaço como sendo $V_\alpha = 0.3$ e $V_l = 15$ (SHI; EBERHART, 1998). A população foi formada por 20 partículas e os componentes cognitivo e social assumiram o valor 2. O fator de inércia foi ajustado para diminuir gradualmente de 0.9 para 0.4 durante as primeiras 1.500 iterações (BIRGE, 2003; SHI; EBERHART, 1998).

Simulated Annealing (SA): Foi idealizado por Kirkpatrick, Gelatt e Vecchi (1983) com a finalidade de resolver problemas de otimização se inspirando em processos térmicos. O

algoritmo começa a busca a partir de uma solução inicial qualquer. Em cada iteração, um “vizinho” s' da solução corrente (s) é gerado aleatoriamente. A variação da função objetivo é testada por $\Delta = f(s') - f(s)$. Em problemas de minimização, se $\Delta < 0$, então s' passa a ser a nova solução corrente; caso contrário, s' é aceito com uma probabilidade $e^{-\Delta/T}$, na qual T é uma “temperatura”. Esse processo é repetido até que T seja tão pequeno que nenhuma nova solução possa ser aceita — nesse momento é dito que o sistema está estável (usualmente, a cada iteração faz-se $T = \alpha T$ — α é uma taxa de resfriamento). A ideia é que a probabilidade de aceitar novas soluções que degradam o valor da função objetivo decresça com a temperatura. Assim, inicialmente a tendência de melhorar o resultado é baixa, entretanto, o espaço de busca é melhor explorado globalmente. Com a redução da temperatura, a tendência é de que novas soluções só são aceitas se melhoram o resultado, de maneira que no final do processo ocorra a convergência para um mínimo local — que pode ser alcançado quando o resfriamento ocorre suficientemente lento (recomenda-se $\alpha \leq 0.95$ (LAARHOVEN; AARTS, 1987; KIRKPATRICK; GELATT; VECCHI, 1983)). Adotou-se neste trabalho $\alpha = 0.95$.

SaDE: É uma versão autoadaptativa da evolução diferencial (DE), a qual amostra valores aleatórios para seus parâmetros de controle (F e Cr) a partir das distribuições normais $\mathcal{N}(0.5, 0.3)$ e $\mathcal{N}(CRm, 0.1)$, respectivamente. Inicialmente, CRm é ajustado para 0.5, mas é adaptado a cada 25 gerações (QIN; HUANG; SUGANTHAN, 2009). Similar adaptação ocorre a cada 50 gerações para selecionar a estratégia de geração de vetores experimentais — as quais podem ser $DE/rand/1/bin$ ou $DE/current-to-best/1/bin$. Embora o SaDE tenha alguns parâmetros a serem ajustados pelo usuário, os autores afirmam que o algoritmo é robusto, especialmente, ao se levar em conta a sua sensibilidade aos parâmetros fundamentais, F e Cr (QIN; HUANG; SUGANTHAN, 2009). Adotou-se população $P = 20$.

RIS: Consiste de um algoritmo memético que gradualmente perturba uma única solução por meio de dois componentes, a saber (CARAFFINI *et al.*, 2013): uma busca local determinística que explora a vizinhança da solução e uma amostragem estocástica que produz soluções aleatórias para serem recombinadas (herança). Tal recombinação se dá por meio de cruzamento exponencial (PRICE; STORN; LAMPINEN, 2006). Nesta pesquisa, os parâmetros requeridos foram definidos como em (CARAFFINI *et al.*, 2013). Ou seja, o fator de herança foi configurado como sendo $\alpha_e = 0.5$, o raio da busca inicial foi $\rho = 0.4$ e o critério de parada foi fixado em $\varepsilon = 10^{-6}$.

As meta-heurísticas descritas acima foram estudadas e utilizadas para se calibrar o algoritmo C³E-SL, conforme abordado nos experimentos a seguir.

3.2 Avaliação Empírica das Meta-heurísticas

Esta seção apresenta experimentos para a calibração do algoritmo C³E-SL a partir das meta-heurísticas descritas na Seção 3.1. O objetivo foi tornar o C³E-SL mais prático, suprimindo-se a necessidade de informação (por parte do usuário/especialista de domínio) dos parâmetros α e I . Como mostrado na Figura 4, a estimação destes parâmetros envolve um difícil problema de otimização *multimodal* (com diferentes máximos locais). Meta-heurísticas são particularmente úteis na exploração eficiente de espaços de busca tal como o ilustrado nesta figura. Em outras palavras, a busca guiada/informada desempenhada por estes algoritmos pode auxiliar na construção de melhores classificadores.

Figura 4 – Acurácias de classificação (%) do algoritmo C³E-SL ao se variar os valores dos parâmetros α e I na base de dados *Wine Red Quality* (FRANK; ASUNCION, 2010).

3.2.1 Configuração Experimental

A calibração do C³E-SL consiste da busca de um par de valores ótimos, $\langle \alpha^*, I^* \rangle$, que possa fornecer as melhores acurácias de classificação possíveis para um dado problema. Essencialmente, valores *ótimos* para os parâmetros α e I podem ser estimados por meio de um procedimento de *validação cruzada*, no qual usualmente os dados são divididos em três conjuntos: o conjunto de treinamento, o conjunto de validação e o conjunto de teste. Em experimentos controlados, os novos dados (ou *conjunto alvo*) a serem classificados são frequentemente referidos como sendo o conjunto de teste ou de validação. Estes dois termos têm sido usados indistintamente na literatura, algumas vezes causando confusão. Nesta pesquisa, presume-se que

o conjunto alvo, ou conjunto de teste, não foi usado no processo de construção do agregador de classificadores, nem mesmo na otimização de qualquer parâmetro do algoritmo — i.e., trata-se de um conjunto independente de dados que não contribuíram para a formação do classificador resultante. Portanto, e como usual (WITTEN; FRANK, 2005), somente os conjuntos de treinamento e de validação foram usados para treinar e otimizar os parâmetros do C³E-SL.

A Figura 5 resume os dois principais passos da abordagem de otimização proposta. Inicialmente, um agregador de classificadores é induzido a partir de um conjunto de treinamento. Então, meta-heurísticas são usadas para se estimar α^* e I^* (“ótimos”) a partir de um conjunto de validação, no qual um agregador de agrupadores foi também induzido. Os valores resultantes dos parâmetros são finalmente utilizados para avaliar o classificador em um conjunto de teste, no qual, novamente e como requerido pelo C³E-SL, um agregador de agrupadores (similar ao produzido para o conjunto de validação) deve ser construído. Note que nesta etapa o agregador de classificadores é retreinado usando-se todas as instâncias rotuladas disponíveis (i.e., todos os dados dos conjuntos de treinamento e de validação). Dado que em experimentos controlados (como este) os rótulos de classes de todas as instâncias são conhecidos, o procedimento ilustrado na Figura 5 pode ser repetido várias vezes a fim de se computar estatísticas de interesse que ajudarão a avaliar a abordagem. Para isso, adotou-se o procedimento de validação cruzada de 5 pastas (WITTEN; FRANK, 2005), no qual cada pasta engloba 20% dos dados — este foi exatamente o tamanho do conjunto de teste (conjunto alvo).

Uma maneira fácil (mas tipicamente inviável) de se encontrar o par de valores ótimos $\langle \alpha^*, I^* \rangle$ é através da busca em grade (BERGSTRA; BENGIO, 2012). Esta busca é simplesmente uma busca exaustiva que avalia todas as possíveis combinações de parâmetros, sendo que cada parâmetro assume valores contidos em um subconjunto predefinido. Para o problema em

Figura 5 – Esquema de calibração do algoritmo C³E-SL via validação cruzada. Os experimentos realizados assumiram que o conjunto alvo foi formado por 20% dos dados — os 80% restantes foram divididos em duas partes iguais dando origem aos conjuntos de treinamento e de validação.

questão, o número de iterações (I) é naturalmente discreto, enquanto que o parâmetro α assume valores reais e, portanto, necessita ser discretizado antes da execução da busca. Assim, para ambos os parâmetros, os limites inferiores e superiores foram definidos *a priori*. Em particular, utilizou-se todas as diferentes combinações de $\alpha = \{0, 0.001, 0.002, \dots, 1\}$ e $I = \{1, 2, 3, \dots, 50\}$. Portanto, para o problema em questão, realizar uma busca em grade significa rodar 50050 vezes o algoritmo C³E-SL para, assim, se encontrar o par de valores ótimos, $\langle \alpha^*, I^* \rangle$.

A Equação (3.3) denota a busca pelos valores ótimos dos parâmetros α e I . Em particular, “ClassErr” representa a função de *fitness* que retorna o erro de classificação do C³E-SL, o qual deseja-se minimizar, para um determinado par de valores dos parâmetros.

$$\langle \alpha^*, I^* \rangle = \arg \min_{\alpha, I} \text{ClassErr} \left[\text{C}^3\text{E-SL}(\langle \alpha, I \rangle) \right]. \quad (3.3)$$

Objetivando-se encontrar o par $\langle \alpha^*, I^* \rangle$, as meta-heurísticas avaliadas foram guiadas pela função de *fitness* “ClassErr” na Equação (3.3). Um limite de tempo de execução, com base no tempo requerido pela busca em grade, foi fixado para todas as meta-heurísticas. Mais especificamente, para cada base de dados, o tempo de execução da busca em grade foi anotado e então 50% deste tempo foi dado como tempo limite para a execução das meta-heurísticas¹. O Algoritmo 3 sumariza a abordagem de otimização paramétrica proposta. A partir dela, as meta-heurísticas foram executadas e comparadas entre si. Além disso, comparações que levaram em conta também a busca em grade permitiram avaliar o custo-benefício entre desempenho (taxa de classificação correta) e tempo computacional. Em outras palavras, espera-se que as meta-heurísticas possam fornecer resultados muito próximos da busca em grade, mas em metade do tempo exigido por esta técnica.

Como ilustrado na Figura 1 (da Seção 2.1), o algoritmo C³E-SL refina resultados de classificadores a partir da informação de agrupamentos. Nos experimentos desta seção, o componente supervisionado agregou dois classificadores bem conhecidos: o *Naive Bayes* (NB) e a Árvore de Decisão (*Decision Tree* — DT) (WITTEN; FRANK, 2005). Já o componente não supervisionado agregou duas partições de dados distintas. Estas partições foram induzidas pelo tradicional algoritmo de agrupamento K-Médias (KM) (JAIN, 2010). Uma destas partições assumiu k grupos, enquanto que a outra foi gerada com $2k$ grupos. A variante k nos dois agrupamentos foi estimada automaticamente a partir dos dados usando-se um índice de validade relativo conhecido como Silhueta Simplificada (CAMPELLO; HRUSCHKA, 2006; HRUSCHKA; CAMPELLO; CASTRO, 2006). A matriz de similaridades que concilia os agrupamentos foi construída seguindo o Passo II da Seção 2.1 — i.e., a matriz S foi formada a partir de médias das coocorrências de duas instâncias distintas em um mesmo grupo levando-se em conta todas as partições induzidas.

¹ Todas as meta-heurísticas em estudo foram implementadas em Matlab usando apenas os comandos necessários. Para cada meta-heurística, a inicialização da população foi feita dentro de uma região comum (de α e I) tentando garantir, então, uma comparação mais eficiente e uniforme. O mesmo computador (i7, 3.2 GHz, 12 Gb RAM), rodando apenas o sistema operacional, foi usado para todos os experimentos controlados.

Algoritmo 3: Abordagem para a calibração do C³E-SL**Entrada:** MH $\leftarrow$ escolha uma meta-heurística.**Saída:** $\langle \alpha^*, I^* \rangle$.**1 Executar** MH:

2 Selecionar aleatoriamente soluções iniciais, $\langle \alpha, I \rangle$, tal que $\alpha = \{0, 0.001, \dots, 1\}$ e $I = \{1, 2, \dots, 50\}$. Assim, uma população contendo um ou mais indivíduos (soluções) pode ser inicializada;

3 Executar o C³E-SL (Algoritmo 2) para cada indivíduo da população corrente (usando a solução codificada $\langle \alpha, I \rangle$ como entrada para o algoritmo) — a taxa de classificação incorreta, como expressado na Equação (3.3), é usada para determinar a qualidade dos indivíduos;

4 Aplicar os operadores próprios de MH para se obter novos indivíduos e, assim, formar uma nova população (para mais detalhes veja a Seção 3.1);

5 Retornar ao Passo 3;

6 até tempo limite ser atingido;

7 Selecionar o melhor indivíduo (que fornece a menor taxa de classificação incorreta) para ser o par de valores $\langle \alpha^*, I^* \rangle$.

Vinte bases de dados reais obtidas a partir do repositório de dados da UCI (FRANK; ASUNCION, 2010) foram empregadas nos diferentes experimentos reportados a seguir. As principais características destas bases estão sumarizadas na Tabela 4.

3.2.2 Resultados e Análises

Inicialmente, comparou-se o algoritmo baseado em perda quadrática (C³E-SL) com o C³E original introduzido em (ACHARYA *et al.*, 2011). Seguindo as configurações experimentais

Tabela 4 – Principais características das bases de dados usadas.

ID	Base de Dados	Inst.	Atrib.	Classes
<i>Wisc</i>	Breast Cancer Wisconsin (Original)	683	9	2
<i>Pima</i>	Pima Indians Diabetes	768	8	2
<i>Yeast</i>	Yeast	205	20	4
<i>Spam</i>	Spambase	4601	57	2
<i>Iono</i>	Ionosphere	351	33	2
<i>Iris</i>	Iris	150	4	3
<i>WRed</i>	Wine Red Quality	1599	12	6
<i>Blood</i>	Blood Transfusion Service Center	748	4	2
<i>Glass</i>	Glass Identification	214	9	7
<i>Sonar</i>	Con. Bench (Sonar, Mines vs. Rocks)	208	60	2
<i>VertCol</i>	Vertebral Column	310	6	3
<i>Seeds</i>	Seeds	210	7	3
<i>Ecoli</i>	Ecoli	336	7	8
<i>ILPD</i>	Indian Liver Patient Dataset	579	10	2
<i>LungC</i>	Lung Cancer	27	52	3
<i>Soybean</i>	Soybean (Small)	47	35	4
<i>Heart</i>	Statlog Heart	270	13	2
<i>WDBC</i>	Breast Cancer Wisconsin (Diag.)	569	30	2
<i>Derma</i>	Dermatology	358	33	6
<i>Contra</i>	Contraceptive Method Choice	1473	9	3

definidas na seção anterior, executou-se uma busca em grade para a calibração de ambos os algoritmos. Esta abordagem é ineficiente (e inviável para problemas do mundo real), mas é experimentalmente conveniente, pois fornece uma boa base comparativa para os resultados alcançados pelas meta-heurísticas.

A Tabela 5 apresenta as acurácias (taxas de classificação correta média) das duas versões do C³E juntamente com os resultados produzidos pelo agregador de classificadores (NB+DT). Na maioria das vezes, ambas as versões do algoritmo obtiveram melhores resultados do que o agregador (NB+DT). Além disso, os resultados do C³E-SL mostraram-se comparáveis àqueles produzidos pelo C³E original. Contudo, note que o C³E original desempenha um processo de otimização mais complexo e custoso computacionalmente.

A Tabela 6 reporta os resultados do C³E-SL calibrado pelas diferentes meta-heurísticas, as quais foram rodadas a partir do Algoritmo 3 usando-se metade do tempo requerido pela busca em grade. Para facilitar as comparações, os resultados da busca em grade (apresentados na Tabela 5) foram novamente reproduzidos nesta tabela. Em algumas bases de dados, como esperado, a busca em grade (que é computacionalmente intensiva) proveu os melhores resultados. Contudo, as meta-heurísticas se destacaram em vários casos. Mais especificamente, os algoritmos DE e SaDE alcançaram maiores acurácias do que a busca em grade em 55% das bases de dados, enquanto que os algoritmos (1+1)-ES, CMA-ES e PSO mostraram-se melhores em 50% delas. Já o algoritmo SA produziu melhores resultados em relação à busca em grade em 45% das bases. O algoritmo RIS, por sua vez, foi melhor apenas em 20% das bases de dados.

Tabela 5 – Acurácias (%) do agregador de classificadores (NB+DT) e dos algoritmos C³E e C³E-SL (desvios padrões estão entre parênteses) — os melhores resultados estão destacados em negrito.

Base de Dados	NB+DT	C ³ E*	C ³ E-SL
<i>Wisc</i>	95.60 (2.8)	96.48 (1.2)	96.48 (1.2)
<i>Pima</i>	76.18 (3.1)	77.09 (3.6)	75.92 (2.7)
<i>Yeast</i>	95.61 (3.2)	96.59 (2.8)	97.56 (1.7)
<i>Spam</i>	81.20 (1.2)	92.20 (1.2)	92.81 (1.1)
<i>Iono</i>	88.34 (4.3)	85.20 (11.2)	87.75 (7.1)
<i>Iris</i>	95.33 (3.0)	96.00 (2.8)	96.00 (2.8)
<i>WRed</i>	55.91 (4.5)	57.60 (8.7)	57.48 (9.1)
<i>Blood</i>	75.80 (1.9)	76.47 (1.5)	76.60 (1.7)
<i>Glass</i>	64.97 (6.7)	62.65 (9.9)	64.97 (6.7)
<i>Sonar</i>	73.60 (8.7)	71.20 (7.6)	67.38 (9.4)
<i>VertCol</i>	82.26 (5.9)	59.33 (15.2)	75.16 (16.1)
<i>Seeds</i>	88.57 (7.4)	90.00 (3.1)	91.90 (3.2)
<i>Ecoli</i>	84.83 (3.8)	84.83 (4.5)	84.53 (4.8)
<i>ILPD</i>	62.69 (3.4)	66.84 (9.4)	60.98 (8.2)
<i>LungC</i>	52.68 (10.9)	52.68 (10.9)	52.68 (10.9)
<i>Soybean</i>	80.92 (6.6)	80.92 (6.6)	80.92 (6.6)
<i>Heart</i>	80.74 (2.6)	81.85 (2.3)	81.57 (1.7)
<i>WDBC</i>	93.50 (0.8)	92.97 (0.7)	93.41 (0.5)
<i>Derma</i>	95.25 (1.2)	94.49 (1.8)	90.92 (6.0)
<i>Contra</i>	51.26 (1.0)	50.00 (1.5)	50.90 (1.0)

*Seguindo (ACHARYA *et al.*, 2011), adotou-se $\lambda = 1$.

Tabela 6 – Acurácias (%) do C³E-SL com parâmetros otimizados pelas meta-heurísticas estudadas — desvios padrões estão entre parênteses. Os melhores resultados entre todas as meta-heurísticas estão em negrito. Note que as meta-heurísticas rodaram em apenas 50% do tempo requerido pela busca em grade.

Base de Dados	Busca em Grade	(1+1)-ES	CMA-ES	DE	SA	PSO	SaDE	RIS
<i>Wisc</i>	96.48 (1.2)	96.48 (1.2)	96.04 (1.1)	96.77 (1.5)	95.90 (0.8)	96.19 (1.0)	96.85 (0.5)	96.19 (0.5)
<i>Pima</i>	75.92 (2.7)	74.23 (2.9)	74.74 (2.8)	75.00 (2.6)	75.53 (3.0)	75.39 (2.6)	74.15 (1.5)	70.90 (2.4)
<i>Yeast</i>	97.56 (1.7)	97.07 (2.7)	97.07 (2.0)	97.56 (1.7)	97.07 (2.7)	96.59 (2.8)	96.59 (2.7)	96.59 (2.7)
<i>Spam</i>	92.81 (1.1)	92.72 (1.1)	92.72 (1.1)	92.72 (1.1)	92.74 (1.1)	92.76 (1.1)	91.49 (0.6)	91.09 (0.7)
<i>Iono</i>	87.75 (7.1)	89.20 (5.3)	89.20 (5.3)	89.20 (5.3)	88.62 (4.4)	89.48 (5.1)	87.39 (4.6)	76.91 (11.8)
<i>Iris</i>	96.00 (2.8)	90.00 (13.3)	95.33 (3.0)	95.33 (3.0)	96.00 (2.8)	95.33 (3.0)	96.00 (2.8)	94.17 (2.8)
<i>WRed</i>	57.48 (9.1)	55.85 (10.1)	55.35 (10.1)	55.41 (9.8)	55.29 (9.7)	52.16 (11.1)	43.62 (6.6)	44.50 (4.4)
<i>Blood</i>	76.60 (1.7)	76.47 (0.9)	76.07 (2.2)	75.80 (1.9)	76.87 (1.0)	76.07 (2.2)	76.87 (0.7)	76.40 (0.3)
<i>Glass</i>	64.97 (6.7)	57.53 (9.0)	64.97 (6.7)	64.97 (6.7)	57.53 (9.0)	62.65 (9.9)	51.64 (9.2)	51.76 (7.4)
<i>Sonar</i>	67.38 (9.4)	66.34 (11.3)	68.26 (11.3)	66.83 (13.0)	64.91 (11.6)	68.26 (11.3)	68.75 (7.3)	62.85 (8.2)
<i>VertCol</i>	75.16 (16.1)	74.84 (15.9)	74.84 (15.9)	74.84 (15.9)	75.48 (16.0)	74.84 (15.9)	78.95 (1.9)	73.06 (5.2)
<i>Seeds</i>	91.90 (3.2)	87.14 (6.4)	89.52 (2.1)	89.05 (2.1)	86.67 (6.2)	88.57 (7.4)	91.43 (2.7)	90.00 (1.9)
<i>Ecoli</i>	84.53 (4.8)	85.43 (5.4)	84.83 (4.5)	84.24 (5.3)	84.83 (4.5)	84.53 (4.8)	83.41 (2.6)	81.18 (2.1)
<i>ILPD</i>	60.98 (8.2)	62.01 (9.7)	59.07 (9.6)	61.32 (8.6)	60.46 (8.5)	62.18 (8.4)	64.12 (6.2)	64.42 (6.6)
<i>LungC</i>	52.68 (10.9)	60.13 (8.5)	61.04 (7.3)	59.22 (10.0)	58.31 (6.5)	57.36 (6.3)	58.35 (7.7)	55.54 (12.3)
<i>Soybean</i>	80.92 (6.6)	89.90 (5.7)	86.74 (4.9)	87.27 (5.0)	90.46 (6.1)	89.40 (7.7)	87.27 (7.7)	89.90 (4.7)
<i>Heart</i>	81.57 (1.7)	82.87 (2.3)	82.13 (1.5)	82.41 (2.6)	81.76 (1.3)	82.31 (2.4)	82.41 (2.2)	82.69 (1.7)
<i>WDBC</i>	93.41 (0.5)	93.89 (0.6)	93.98 (0.5)	93.94 (0.6)	93.01 (0.4)	93.89 (0.5)	93.94 (0.6)	92.84 (0.4)
<i>Derma</i>	90.92 (6.0)	94.97 (1.4)	95.46 (1.2)	95.67 (0.9)	94.27 (1.2)	95.39 (1.1)	95.25 (1.4)	87.02 (10.1)
<i>Contra</i>	50.90 (1.0)	51.65 (1.6)	51.54 (1.1)	51.53 (1.0)	50.68 (1.1)	51.56 (1.1)	50.61 (1.2)	48.44 (0.2)
Média Geral	78.80 (5.1)	78.94 (5.8)	79.45 (4.7)	79.45 (4.9)	78.82 (4.9)	79.25 (5.3)	78.45 (3.5)	76.32 (4.3)
Vitórias / Empates*		10	10	11	9	10	11	4

*Contagem em relação à busca em grade.

Considerando comparações apenas entre as meta-heurísticas, observa-se que o algoritmo RIS obteve melhor acurácia apenas em uma base de dados. Os algoritmos CMA-ES e PSO foram os melhores em duas bases de dados, enquanto que o DE em três e o SA em quatro. Já os algoritmos (1+1)-ES e SaDE produziram os melhores resultados, respectivamente, em cinco e seis bases de dados. Embora ainda possa se dizer que o desempenho das diferentes meta-heurísticas possam ser comparáveis entre si, uma análise mais cuidadosa sugere que os algoritmos DE, SaDE, (1+1)-ES, CMA-ES e PSO proporcionaram os melhores resultados. Em outras palavras, desses algoritmos, o DE, CMA-ES e PSO alcançaram as melhores acurácias na média geral, enquanto que os algoritmos SaDE e CMA-ES mostraram-se mais estáveis (observe os desvios entre parênteses na penúltima linha da Tabela 6). Notando-se estes resultados, pode-se dar preferência pelo uso do CMA-ES ou SaDE (porque são algoritmos com menor dependência de parâmetros) ou o DE (devido à simplicidade de implementação e eficiência).

Na prática, escolher um algoritmo (dentre várias alternativas) para uma aplicação em particular não é uma tarefa simples. Para o problema tratado, propriedades convenientes das meta-heurísticas poderiam ser também consideradas no momento da escolha da abordagem mais adequada. Em resumo, queremos responder a seguinte questão: Qual das meta-heurísticas estudadas é a preferível para se otimizar os parâmetros do algoritmo C³E-SL? Para responder esta questão pode-se levar em conta dois aspectos, a saber: o quão *amigável* ao usuário uma meta-heurística é e o seu desempenho com relação à acurácia de classificação (note que o tempo de execução foi fixado para todas elas). Em um primeiro momento, pode-se argumentar que “ser amigável” é um critério bem subjetivo. Basicamente, este aspecto pode ser entendido como

sendo a facilidade de se manusear o algoritmo, e esta facilidade de manuseio está diretamente relacionada à quantidade de parâmetros que a meta-heurística possui. Deste modo, assume-se que, quanto mais parâmetros configuráveis um algoritmo possui, menos amigável ele é. Contudo, vale destacar que algumas meta-heurísticas são naturalmente mais robustas à variação de valores de seus parâmetros do que outras. Como pode ser observado na Seção 3.1, os algoritmos PSO e SaDE possuem vários parâmetros configuráveis, embora o segundo algoritmo seja tipicamente reconhecido pela sua robustez a variações (QIN; HUANG; SUGANTHAN, 2009). De fato, os algoritmos SaDE e CMA-ES adaptam seus parâmetros de controle durante o processo evolutivo, sendo considerados alternativas mais versáteis e de fácil uso. Portanto, pode-se sugerir que o PSO seja possivelmente a meta-heurística menos indicada para o problema de calibração do C³E-SL. As outras meta-heurísticas — (1+1)-ES, DE, SA e RIS — têm poucos parâmetros para serem configurados, mas também estão sujeitas à análises de sensibilidade paramétrica (a exemplo da discussão elaborada acima envolvendo os algoritmos PSO, SaDE e CMA-ES).

A Tabela 7 apresenta a classificação das meta-heurísticas segundo a soma dos ranks do teste de Friedman calculado a partir das acurácias reportadas na Tabela 6. Não foram identificadas diferenças estatisticamente significativas entre as meta-heurísticas, embora os ranks sejam discriminativos da qualidade de cada uma delas. Essencialmente, o teste de Friedman é um teste de significância estatística não paramétrico equivalente à Análise de Variância (ANOVA). Testes de significância estatística podem ser usados com o objetivo de prover algum indício sobre a validade e não aleatoriedade dos resultados obtidos. Mais precisamente, executou-se testes estatísticos seguindo a abordagem descrita em (DEMSAR, 2006). Em resumo, a abordagem adotada é adequada para se comparar múltiplos algoritmos em múltiplas bases de dados usando-se o teste de Friedman seguido do Teste post-hoc de Nemenyi. Estas análises mostraram também que não há diferença estatística significativa quando compara-se os resultados da busca em grade com os obtidos pelas diferentes meta-heurísticas — exceto para o algoritmo RIS (o qual se mostrou inferior com nível de significância $\alpha = 5\%$). A partir destes resultados, pode-se concluir que os algoritmos (1+1)-ES, CMA-ES, DE, SA, PSO e SaDE podem prover, em menor tempo de execução, classificadores tão precisos quanto aqueles produzidos pela busca em grade. Este fato é particularmente relevante para aplicações reais de mineração de dados, nas quais grandes massas de dados estão disponíveis e requerem processamento rápido.

Tabela 7 – Classificação das meta-heurísticas de acordo com a soma dos ranks do teste de Friedman computado a partir das acurácias reportadas na Tabela 6.

Classificação	Meta-heurística
1	DE
2	(1+1)-ES
3	CMA-ES
4	PSO
5	SaDE
6	SA
7	RIS

De acordo com o critério de acurácia explorado na Tabela 7, o algoritmo evolutivo DE é o mais vantajoso entre todos os estudados. Entretanto, conforme já discutido, é esperado que possa haver algum equilíbrio entre facilidade de uso e desempenho (de classificação). Neste contexto, os algoritmos SaDE e CMA-ES poderiam ser indicados também — embora o DE possua apenas dois parâmetros de controle. As Figuras 6–9 mostram os valores mínimos, médios e máximos para α^* e I^* ao se utilizar os algoritmos CMA-ES e DE. Note que há uma alta variância dos valores para diferentes bases de dados (e também para diferentes pastas da validação cruzada). Isto sugere que a calibração do C³E-SL é altamente dependente dos dados. Observa-se também que o C³E-SL, com parâmetros otimizados pelo CMA-ES, produziu iguais ou melhores acurácias que aquelas obtidas pelo agregador de classificadores (NB+DT) em 75% das bases de dados. Os algoritmos DE e SaDE também foram melhores que a agregação NB+DT em, respectivamente,

Figura 6 – Valores mínimos, máximos e médios para α^* (CMA-ES).

Figura 7 – Valores mínimos, máximos e médios para α^* (DE).

Figura 8 – Valores mínimos, máximos e médios para I^* (CMA-ES).

Figura 9 – Valores mínimos, máximos e médios para I^* (DE).

70% e 60% das bases de dados. Portanto, pode-se concluir que o C³E-SL com parâmetros otimizados a partir do CMA-ES, DE ou SaDE pode produzir melhores classificadores em um tempo computacional significativamente menor que uma busca exaustiva. Além disso, empregar os algoritmos CMA-ES, DE ou SaDE na tarefa de calibração garante uma certa facilidade de uso da aplicação, tornando a combinação de classificadores e agrupadores mais versátil e prática.

O algoritmo DE, além de possuir apenas dois parâmetros de controle, se destacou em desempenho, oferecendo em geral um bom custo-benefício. Visto que é um algoritmo simples e com possibilidade de melhorias, decidiu-se investigar uma versão mais robusta deste algoritmo para a calibração do C³E-SL. Esta nova versão do DE é descrita na próxima seção.

3.3 Um Novo Algoritmo para Calibrar o C³E-SL

A computação evolutiva envolve técnicas e métodos que se inspiram no princípio da evolução de Darwin (DARWIN; BEER, 1998) para resolver diferentes tipos de problemas, os quais, em geral, são solucionados por meio de Algoritmos Evolutivos (AEs) (EIBEN; SMITH, 2008; BEYER; SCHWEFEL, 2002; WHITLEY, 2001; BÄCK; FOGEL; MICHALEWICZ, 1997). A exemplo dos AEs apresentados na seção anterior, estes algoritmos, por meio de princípios biológicos (como *seleção natural* e *herança genética*), evoluem (até um dado critério de convergência) uma população de indivíduos que são soluções candidatas para um determinado problema. Neste processo, os indivíduos mais aptos (segundo uma função de *fitness*) têm maiores chances de se reproduzir e sobreviver, estando suscetíveis as modificações promovidas por operadores baseados na genética, tal como cruzamento (*crossover*) e mutação (*mutation*).

De maneira mais formal, AEs utilizam-se de um componente aleatório para evitar mínimos locais e acelerar a sua convergência, muitas vezes obtendo-se soluções ótimas ou próximas de ótimas para diversos problemas (EIBEN; SMITH, 2003; MIETTINEN, 1999; MITCHELL, 1996). Dado que não se conhece (*a priori*) o espaço de busca do problema (geralmente apenas os limites podem ser definidos), inicialmente há uma tendência do algoritmo em explorá-lo globalmente. Com a evolução da busca (e reconhecimento de regiões mais promissoras), cada vez mais passam a ser priorizados locais mais específicos, nos quais potencialmente residem as soluções ótimas, ou próximas das ótimas. Por isso AEs são considerados meta-heurísticas potencialmente efetivas para problemas NP-difíceis, sendo capazes de obter boas soluções para tais problemas em tempo razoável (YU; GEN, 2010; FREITAS, 2002; BÄCK; FOGEL; MICHALEWICZ, 1997). Em particular, estes algoritmos lidam muito bem com problemas de otimização de parâmetros e, em geral, problemas de mineração de dados (os quais muitas vezes são NP-difíceis) — revisões mais detalhadas de AEs para classificação e agrupamento podem ser encontradas em (FREITAS, 2003; HRUSCHKA *et al.*, 2009).

Conforme reportado nas análises experimentais da Seção 3.2, o algoritmo evolutivo *Differential Evolution* (DE) (PRICE, 1996) apresenta, apesar da simplicidade, grande potencial prático. De fato, inúmeros artigos da literatura mostram que o DE alcança bons resultados em várias aplicações (DAS; SUGANTHAN, 2011; PRICE; STORN; LAMPINEN, 2005; STORN; PRICE, 1997). Sob esta perspectiva, objetivou-se aproveitar-se das vantagens do DE, adequando este algoritmo para o problema de calibração do C³E-SL para se obter uma combinação de classificadores e agrupadores que seja livre de parâmetros considerados críticos.

O algoritmo DE (conforme descrito na Seção 3.1) possui três parâmetros de controle que devem ser definidos pelo usuário: o número de indivíduos da população (P) e as constantes de escalonamento (F) e de *crossover* (Cr). Embora em (PRICE, 1996; STORN; PRICE, 1997) seja sugerido que $F = 0.5$ e $Cr = 0.9$ possam ser boas tentativas (iniciais) para se configurar o algoritmo, valores ótimos para estes parâmetros variam significativamente de problema para

problema. Se o usuário optar por um ajuste fino dos valores, buscando melhorar o desempenho do algoritmo, tem sido indicado que $F \in [0.4, 1]$ e $Cr \in [0.8, 1]$ são faixas de valores mais promissoras (STORN, 1996; STORN; PRICE, 1997; RONKKONEN; KUKKONEN; PRICE, 2005). Neste contexto, estudos recentes têm mostrado que o desempenho do DE depende da escolha correta de seus parâmetros de controle (RONKKONEN; KUKKONEN; PRICE, 2005; GAMPERLE; MÜLLER; KOUMOUTSAKOS, 2002; LIU; LAMPINEN, 2002). Valores inapropriados para F e Cr podem levar à lenta convergência e/ou baixo desempenho do algoritmo. A partir desta dificuldade, vários trabalhos têm apresentado diferentes técnicas para se ajustar dinamicamente os parâmetros do DE (DAS; SUGANTHAN, 2011; QIN; HUANG; SUGANTHAN, 2009; BREST *et al.*, 2006; DAS; KONAR; CHAKRABORTY, 2005; LIU; LAMPINEN, 2005). Tais técnicas envolvem abordagens determinísticas, adaptativas ou autoadaptativas para o controle especialmente de F e Cr (EIBEN; SMITH, 2008)². Seguindo este raciocínio, um algoritmo mais robusto a variação de seus parâmetros iniciais é introduzido a seguir. Este algoritmo, chamado de *Dynamic DE* (D²E), é uma extensão da evolução diferencial capaz de adaptar seus próprios parâmetros durante o processo evolutivo para otimizar os parâmetros α e I no C³E-SL.

O algoritmo D²E adota a estratégia conhecida como *DE/best/2/bin* (PRICE, 1996), na qual a Equação (3.1), proveniente do DE original (na Seção 3.1), é substituída pela Equação (3.4). Ou seja, mais uma diferença entre outros dois indivíduos aleatórios é considerada na geração do indivíduo (vetor) doador:

$$\mathbf{v}^{t+1} = \mathbf{z}_{best}^t + F \cdot (\mathbf{z}_\beta^t - \mathbf{z}_\gamma^t + \mathbf{z}_\rho^t - \mathbf{z}_\tau^t). \quad (3.4)$$

Essencialmente, o D²E estende o algoritmo DE realizando a amostragem de seus parâmetros, F e Cr , quando, em duas gerações consecutivas, não ocorrerem mudanças no *fitness* médio da população corrente. Para isso, o algoritmo desenvolvido utiliza a seguinte regra:

$$\langle F, Cr \rangle^{t+1} = \begin{cases} \langle d_1, d_2 \rangle, & \text{se } (\bar{f}_t = \bar{f}_{t-1}); \\ \langle F, Cr \rangle^t, & \text{caso contrário.} \end{cases} \quad (3.5)$$

na qual $d_1 \in [0, 1]$ e $d_2 \in [0, 1]$ são valores aleatoriamente amostrados a partir de uma distribuição uniforme, $\bar{f}_t$ e $\bar{f}_{t-1}$ representam o *fitness* médio da população nas gerações t e $t - 1$, respectivamente.

O algoritmo se inicia com parâmetros de controle fornecidos pelo usuário (e.g., $P = 20$, $F = 0.5$ e $Cr = 0.9$). O D²E, então, permite que F e Cr assumam novos valores caso não ocorram melhorias em duas iterações subsequentes. Esta ideia evita mínimos locais (se valores suficientemente altos de F e/ou Cr forem escolhidos), ou pode reforçar a busca local (com

² O tamanho da população e o número de gerações requeridas para a convergência estão de certo modo relacionados. Em um ponto de vista prático pode-se considerar que, quanto mais indivíduos menor será o número de gerações para a convergência.

valores suficientemente baixos de F e/ou Cr). Esta dinâmica minimiza o impacto dos valores iniciais (definidos pelo usuário) tanto para F , como para Cr . Em outras palavras, esta simples adaptação permite uma boa relação entre exploração global e local, tornando o algoritmo mais robusto aos valores iniciais de seus parâmetros de controle.

Resultados experimentais que mostram que o D²E pode eficientemente otimizar o algoritmo C³E-SL, fornecendo classificadores competitivos em uma variedade de bases de dados, estão sumarizados a seguir.

3.4 Avaliação Empírica do Algoritmo D²E

Esta seção apresenta os resultados do algoritmo evolutivo D²E (*Dynamic Differential Evolution*) introduzido na Seção 3.3 e concebido especialmente para a calibração do C³E-SL. Este algoritmo foi comparado com o algoritmo DE original (PRICE, 1996; STORN; PRICE, 1997) e também com suas versões que são consideradas estado da arte. Utilizou-se da mesma abordagem para otimização paramétrica descrita na Seção 3.2, ou seja, as diferentes versões do DE foram avaliadas através do Algoritmo 3 (que rodou em um tempo máximo correspondente à metade do tempo requerido pela busca em grade). Adotou-se o procedimento de validação cruzada de 5 pastas (conforme ilustrado na Figura 5).

Inicialmente, investigou-se a sensibilidade paramétrica da evolução diferencial variando-se os valores dos parâmetros de controle F e Cr . A Tabela 8 mostra as acurácias (taxa de classificação correta média) alcançadas pelo DE a partir de diferentes combinações de $F = \{0.25, 0.5, 0.75, 1\}$ e $Cr = \{0, 0.25, 0.5, 0.75, 1\}$, juntamente com sugestões da literatura (PRICE, 1996; STORN; PRICE, 1997), i.e., $F = 0.5$ e $Cr = 0.9$ (na última coluna). A estratégia *DE/best/2/bin* foi usada para minimizar o erro de classificação do C³E-SL, baseando-se na Equação (3.3). Como esperado, as melhores configurações de F e Cr não foram sempre as mesmas (para todas as bases de dados), mas uma observação mais criteriosa sugere que, como apontado por Ronkkoen, Kukkonen e Price (2005), resultados mais interessantes são produzidos com $F \in [0.4, 1]$ e

Tabela 8 – Acurácias (%) do C³E-SL otimizado pelo DE a partir de diferentes combinações de F e Cr (os quais permaneceram fixos durante todo o processo). Os melhores resultados estão destacados em negrito — LIT refere-se aos valores de parâmetros sugeridos pela literatura (PRICE, 1996; STORN; PRICE, 1997), i.e., $F = 0.5$ e $Cr = 0.9$.

F	0.25					0.50					0.75					1.00					LIT
Cr	0.00	0.25	0.50	0.75	1.00	0.00	0.25	0.50	0.75	1.00	0.00	0.25	0.50	0.75	1.00	0.00	0.25	0.50	0.75	1.00	
Wisc	95.8	96.5	95.8	96.2	95.9	95.8	96.0	96.2	96.0	96.5	95.9	96.0	96.2	96.5	96.5	95.8	96.0	96.2	95.9	96.6	96.8
Pima	71.6	76.4	74.8	72.7	73.5	74.1	76.4	73.7	75.4	75.0	72.7	76.4	75.3	75.3	76.4	74.2	74.5	76.0	75.5	75.0	75.0
Yeast	95.6	95.6	96.6	96.6	95.1	96.6	94.6	98.0	95.6	96.6	96.1	97.6	97.6	97.6	97.6	94.6	97.6	97.6	96.1	97.6	97.6
Iono	86.9	81.5	85.5	86.6	82.6	84.9	85.2	82.0	86.9	87.2	83.2	86.0	86.3	87.2	85.2	82.3	85.2	86.9	86.3	85.7	89.2
Iris	95.3	96.0	94.7	96.0	96.0	90.0	90.0	95.3	90.0	96.0	90.0	96.0	90.0	96.0	95.3	90.0	96.0	96.0	95.3	96.0	95.3
Blood	76.3	75.7	76.6	76.3	76.3	76.3	76.2	76.6	75.4	75.8	76.3	75.4	75.4	76.5	76.2	75.5	76.2	75.4	76.3	76.2	75.8
Seeds	89.0	89.0	89.5	88.6	87.1	89.0	88.6	89.5	88.6	89.0	89.5	89.0	89.0	88.6	89.0	88.6	89.5	89.5	89.0	91.0	89.1
Ecoli	83.6	79.8	83.3	84.8	83.6	84.2	83.0	85.1	85.1	83.9	77.4	84.2	83.0	84.2	83.6	83.3	84.5	85.1	83.0	83.9	84.2
Ilpd	61.3	61.7	61.3	60.6	61.3	61.0	60.5	61.3	61.3	61.7	60.5	62.2	60.8	61.1	60.5	61.3	61.3	62.2	61.1	61.7	61.3
Glass	59.4	61.7	65.0	63.6	61.2	56.6	60.8	62.2	62.6	63.6	59.4	60.8	59.8	62.6	63.6	62.6	60.8	65.0	65.0	64.5	65.0

Tabela 9 – Acurácias (%) do C³E-SL otimizado pelo D²E com diferentes combinações de valores iniciais de F e Cr (os quais foram dinamicamente atualizados). Os melhores resultados estão destacados em negrito — LIT refere-se aos valores de parâmetros sugeridos pela literatura (PRICE, 1996; STORN; PRICE, 1997), i.e., $F = 0.5$ e $Cr = 0.9$. A última linha apresenta o número de vezes em que o D²E foi melhor/igual ao DE a partir das acurácias da Tabela 8.

F	0.25					0.50					0.75					1.00					LIT
Cr	0.00	0.25	0.50	0.75	1.00	0.00	0.25	0.50	0.75	1.00	0.00	0.25	0.50	0.75	1.00	0.00	0.25	0.50	0.75	1.00	
Wisc	95.8	96.5	96.2	95.9	96.5	96.2	96.8	96.8	96.0	96.3	96.5	96.0	95.8	96.2	96.2	96.3	96.0	96.3	96.5	96.8	96.5
Pima	73.8	76.3	74.0	72.9	74.6	75.9	74.9	72.7	75.1	74.9	75.0	74.5	75.8	75.0	76.3	74.6	75.9	76.2	75.3	76.2	76.3
Yeast	97.1	97.1	97.6	95.6	97.1	97.6	97.6	98.1	97.6	96.1	95.6	98.1	95.6	97.1	96.1	97.6	97.1	98.1	97.1	97.1	97.1
Iono	86.6	87.2	87.2	86.0	84.3	87.2	86.3	87.2	86.3	86.3	86.3	86.3	82.9	86.0	86.3	81.8	86.3	82.6	84.3	85.7	89.2
Iris	96.0	96.0	90.0	96.0	96.0	90.0	96.0	90.0	96.0	96.0	96.0	90.0	96.0	96.0	96.0	96.0	95.3	95.3	96.0	90.0	96.0
Blood	75.4	76.1	76.2	76.2	76.2	75.3	76.5	76.2	76.1	76.7	76.5	75.4	76.1	75.8	75.8	76.2	75.4	76.5	75.4	75.4	76.1
Seeds	86.7	89.5	89.1	89.1	87.1	89.1	89.1	91.0	89.1	89.1	89.5	86.7	86.7	89.1	89.1	89.5	89.1	89.5	89.1	89.1	89.1
Ecoli	85.1	83.6	84.2	83.6	84.5	84.2	84.5	84.2	84.8	84.8	83.6	85.1	84.5	85.1	83.9	83.3	83.9	83.3	83.9	83.6	83.9
Ilpd	60.3	62.0	61.3	61.1	60.5	60.8	60.5	61.7	61.3	61.3	61.1	61.7	61.5	60.8	61.1	61.1	60.5	61.3	61.7	62.5	60.5
Glass	64.0	64.5	63.1	63.6	61.3	62.2	59.8	61.7	63.1	61.3	63.6	62.2	62.2	63.1	63.1	64.5	60.8	61.3	62.2	63.6	59.8
≥	6	9	5	5	8	8	8	5	7	4	9	6	6	4	5	7	4	5	6	4	5

$Cr \in [0.8, 1]$. A Tabela 9 apresenta as acurácias obtidas pelo D²E rodado nas mesmas condições experimentais de seu predecessor. Embora diferentes valores de F e Cr foram inicialmente fixados, tais valores foram dinamicamente ajustados durante o processo do D²E, i.e., de acordo com a regra adotada — Equação (3.5). Na tabela, a última linha indica o número de bases de dados nas quais o D²E alcançou iguais ou maiores acurácias de classificação em relação ao DE — considerando os mesmos valores (iniciais) de F e Cr .

Visando obter garantias sobre a validade e não aleatoriedade dos resultados, executou-se testes estatísticos seguindo a abordagem introduzida em (DEMSAR, 2006). Com isso, constatou-se que não há diferença estatística quando se compara os resultados da busca em grade, do DE e D²E. Portanto, pode-se concluir que os algoritmos DE e D²E produziram classificadores tão precisos quanto aqueles obtidos pela busca em grade, mas utilizando-se de metade do tempo requerido por esta abordagem exaustiva. Considerando-se apenas os algoritmos baseados em evolução diferencial, o D²E é preferível porque agrega as propriedades vantajosas do DE, enquanto oferece robustez aos valores iniciais de seus parâmetros de controle. Como resultado, com valores iniciais $F = Cr = 0.25$, o D²E alcançou acurácias iguais ou superiores ao DE em 90% dos casos — este mesmo percentual é obtido quando se usa $F = 0.75$ e $Cr = 0.0$ (veja novamente a última linha da Tabela 9).

A Figura 10 ilustra, para cada base de dados, as proporções de acurácias maiores ou iguais do D²E sobre o DE nas 21 combinações de F e Cr presentes nas Tabelas 8 e 9. A partir desta comparação pareada, nota-se que o D²E produziu os melhores resultados na maioria das vezes, com destaque para os resultados nas bases de dados *Wisconsin*, *Iris*, *Seeds* e *Ecoli*. Já a Figura 11 destaca os desvios padrões dos algoritmos para as 21 diferentes acurácias das Tabelas 8 e 9 — valores específicos estão apresentados na Tabela 10. Estes números indicam que, na maioria das bases de dados, o D²E mostrou-se mais robusto a variações de seus parâmetros de controle, provendo (como resultado) melhores e mais estáveis acurácias de classificação.

As Figuras 12–17 ilustram o processo evolutivo dos algoritmos DE e D²E, o qual objeti-

Figura 10 – Proporções de acurácias maiores ou iguais do D^2E sobre o DE a partir de 21 diferentes configurações dos parâmetros de controle (F e Cr).

Figura 11 – Acurácias médias com desvios padrões do DE e D^2E a partir de 21 diferentes configurações dos parâmetros de controle (F e Cr).

vou minimizar o erro de classificação do C^3E -SL nas bases de dados *Wisconsin* e *Blood*. Estas figuras, obtidas a partir de uma pasta particular do procedimento de validação cruzada, são bem típicas dos resultados obtidos. Naturalmente para o DE, os valores de F e Cr permaneceram fixos, respectivamente, em 0.5 e 0.75, enquanto que para o D^2E , estes valores foram dinamicamente adaptados. As Figuras 12 e 15 mostram que estes ajustes dinâmicos, a partir da décima quinta geração, contribuíram para a rápida diminuição das taxas de erro. Este comportamento ocorreu na maioria das bases de dados. Levando-se em conta a base de dados *Wisconsin*, a partir da quarta geração, pode-se notar que novos valores (menores) para F e Cr ajudaram a melhorar a qualidade das soluções — veja as Figuras 12–14. Os resultados para *Blood* (Figuras 15–17) expressam a mesma ideia, além de sugerir que, dependendo do problema, a otimização se torna

Tabela 10 – Desvios padrões do DE e D²E a partir das 21 combinações distintas de valores dos parâmetros de controle (F e Cr). Menores valores estão destacados em negrito.

Base de Dados	DE	D ² E
<i>Wisc</i>	0.31	0.30
<i>Pima</i>	1.36	1.07
<i>Yeast</i>	1.07	0.78
<i>Iono</i>	2.00	1.74
<i>Iris</i>	2.66	2.59
<i>Blood</i>	0.42	0.44
<i>Seeds</i>	0.69	1.09
<i>Ecoli</i>	1.80	0.58
<i>Ilpd</i>	0.49	0.57
<i>Glass</i>	2.27	1.39

uma tarefa difícil — entretanto, adaptando-se F e Cr há maiores chances de se alcançar melhores resultados.

Por fim, o algoritmo D²E foi comparado com três variantes do DE consideradas estado da arte: o *Self-adaptive* DE — SaDE (QIN; HUANG; SUGANTHAN, 2009), o jDE (BREST *et al.*, 2006) e o *Composite* DE — CoDE (WANG; CAI; ZHANG, 2011). Estes algoritmos, assim como o D²E, adaptam seus parâmetros de controle (e a estratégia de geração de vetores experimentais) durante o processo evolutivo. As configurações usadas para cada um destes algoritmos foram as recomendadas pela literatura (WANG; CAI; ZHANG, 2011; QIN; HUANG; SUGANTHAN, 2009; BREST *et al.*, 2006). As configurações experimentais foram as mesmas já adotadas aqui.

A Tabela 11 reporta as acurácias obtidas pelo C³E-SL otimizado por cada uma das variantes do DE — os resultados do D²E foram obtidos usando-se $F = Cr = 0.25$. As últimas três linhas resumizam o número de bases de dados nas quais o resultado do D²E foi superior, igual ou inferior aos correspondentes algoritmos. De acordo com estes resultados, pode-se concluir que o D²E mostrou-se competitivo com os três algoritmos considerados estado da arte (não foram identificadas diferenças estatisticamente significativas entre eles).

Conforme já discutido na Seção 3.1, o algoritmo SaDE (QIN; HUANG; SUGANTHAN, 2009) amostra valores aleatórios para F e Cr a partir de distribuições normais, além de selecionar estratégias específicas para a geração do vetor experimental. Estas adaptações ocorrem a partir de um determinado número de gerações. Embora este algoritmo tenha uma porção de parâmetros configuráveis, os autores afirmam que ele tende a ser menos sensível a variação de seus valores. Esta afirmação também é válida para as outras variantes do DE. O algoritmo jDE, em particular, implementa a estratégia $DE/rand/1/bin$ e ajusta F e Cr a cada iteração com probabilidades $\tau_1 = \tau_2 = 0.1$ a partir de distribuições uniformes. Inicialmente, determina-se $F = 0.5$ e $Cr = 0.9$ (BREST *et al.*, 2006). O algoritmo CoDE, por sua vez, combina de uma maneira aleatória três diferentes estratégias: $DE/rand/1/bin$, $DE/rand/2/bin$ e $DE/current-to-rand/1$ com três configurações de parâmetros — $[F = 1.0, Cr = 0.1]$, $[F = 1.0, Cr = 0.9]$ e $[F = 0.8, Cr = 0.2]$. Estas estratégias e valores são tipicamente fixados seguindo (WANG; CAI; ZHANG, 2011).

Figura 12 – Classificação incorreta média – *Wisconsin*. Figura 13 – Valores de F – *Wisconsin*. Figura 14 – Valores de Cr – *Wisconsin*.

Figura 15 – Classificação incorreta média – *Blood*. Figura 16 – Valores de F – *Blood*. Figura 17 – Valores de Cr – *Blood*.

Tabela 11 – Acurácias (%) do C³E-SL otimizado pelas variantes do DE (desvios padrões entre parênteses). Vitórias/Empates/Derrotas denota quantos valores alcançados pelo D²E são superiores, iguais ou inferiores ao algoritmo da coluna correspondente — os melhores resultados estão destacados em negrito.

Bases de dados	SaDE	jDE	CoDE	D ² E
<i>Wisc</i>	96.85 (0.5)	96.82 (0.4)	96.82 (0.6)	96.48 (1.2)
<i>Pima</i>	74.15 (1.5)	74.06 (1.6)	74.06 (1.6)	76.31 (2.7)
<i>Yeast</i>	96.59 (2.7)	96.71 (2.8)	96.71 (2.1)	97.07 (2.0)
<i>Iono</i>	87.39 (4.6)	86.68 (3.0)	88.53 (3.2)	87.17 (9.3)
<i>Iris</i>	96.00 (2.8)	95.67 (3.1)	96.00 (2.8)	96.00 (2.8)
<i>Blood</i>	76.87 (0.7)	77.01 (0.7)	77.01 (0.7)	76.07 (0.5)
<i>Seeds</i>	91.43 (2.7)	90.83 (2.3)	87.14 (2.2)	89.52 (2.1)
<i>Ecoli</i>	83.41 (2.6)	83.56 (2.8)	83.33 (2.8)	83.64 (5.4)
<i>Ilpd</i>	64.12 (6.2)	62.00 (5.1)	67.23 (2.3)	62.01 (7.2)
<i>Glass</i>	51.64 (9.2)	51.99 (7.8)	59.78 (8.8)	64.51 (6.9)
Vitórias	4	7	5	-
Empates	1	0	1	-
Derrotas	5	3	4	-

Entretanto, os autores afirmam que outras configurações podem ser usadas, as quais podem ser baseadas em estudos prévios do problema que se está tratando. A partir deste ponto de vista, os algoritmos avaliados podem também ser passíveis de ajuste fino (ou seja, ajustes manuais em seus parâmetros configuráveis, caso se faça necessário). Visto que o algoritmo D²E requer apenas os valores iniciais de F e Cr , este se apresenta como uma alternativa mais simples (com menor número de parâmetros configuráveis). Apesar da simplicidade, os experimentos desta seção comprovam que o D²E tem grande potencial e aplicabilidade, sobretudo para a tarefa de otimização do algoritmo C³E-SL.

3.5 Considerações Finais

O algoritmo C³E-SL combina resultados de classificadores e agrupadores para produzir predições mais refinadas dos dados. Para isso, o C³E-SL necessita da informação do parâmetro α , que controla a importância dos classificadores em relação aos agrupadores, e também do número de iterações do algoritmo (I). Levando-se em conta que em (ACHARYA *et al.*, 2011) não foi explorado nenhum critério sistemático para a estimação de valores para estes parâmetros, uma abordagem para a calibração (automática, a partir dos dados) do C³E-SL foi proposta. Esta abordagem emprega meta-heurísticas amplamente conhecidas e voltadas para a otimização paramétrica. Por exemplo, os Algoritmos Evolutivos (AEs) (EIBEN; SMITH, 2008; HANSEN; OSTERMEIER, 2001; PRICE, 1996; RECHENBERG, 1973), PSO (*Particle Swarm Optimization*) (KENNEDY; EBERHART, 1995), SA (*Simulated Annealing*) (KIRKPATRICK; GELATT; VECCHI, 1983) e um algoritmo baseado em computação memética (CARAFFINI *et al.*, 2013; CARAFFINI; NERI; PICINALI, 2014).

Os principais resultados do estudo realizado foram documentados em um artigo publicado

em conferência internacional (COLETTA *et al.*, 2013). Neste artigo, assim como abordado na Seção 3.1, as meta-heurísticas tiveram como objetivo otimizar os parâmetros α e I através da minimização do erro de classificação do C³E-SL. Este trabalho foi estendido e publicado em um periódico internacional (COLETTA *et al.*, 2015b). Este artigo oferece um estudo mais detalhado do comportamento do algoritmo C³E-SL, além de conter experimentos adicionais que reforçam a suposição inicial de que a combinação de classificadores e agrupadores possa gerar modelos de classificação com maior capacidade de generalização.

A partir das investigações provenientes das meta-heurísticas, um novo algoritmo evolutivo mais robusto para otimizar a combinação de classificadores e agrupadores foi proposto. Este algoritmo, denominado *Dynamic DE* (D²E), é uma extensão da evolução diferencial (PRICE, 1996; STORN; PRICE, 1997) capaz de adaptar seus parâmetros de controle durante o processo evolutivo. Análises experimentais que indicam que o D²E pode eficientemente calibrar o algoritmo C³E-SL, produzindo classificadores competitivos para uma variedade de problemas, estão sumarizadas em (COLETTA *et al.*, 2015a).

Em resumo, os artigos publicados a partir dos estudos descritos no presente capítulo são:

- COLETTA, L. F. S.; HRUSCHKA, E. R.; ACHARYA, A.; GHOSH, J., *Towards the Use of Metaheuristics for Optimizing the Combination of Classifier and Cluster Ensembles*. In: 1st BRICS Countries Congress on Computational Intelligence (BRICS-CCI), 2013, Porto de Galinhas. Proceedings of the 1st BRICS Countries Congress on Computational Intelligence (BRICS-CCI), p. 483-488, 2013 — (COLETTA *et al.*, 2013).
- COLETTA, L. F. S.; HRUSCHKA, E. R.; ACHARYA, A.; GHOSH, J., *Using metaheuristics to optimize the combination of classifier and cluster ensembles*. Integrated Computer-Aided Engineering, v. 22, n. 3, p. 229-242, 2015 — (COLETTA *et al.*, 2015b).
- COLETTA, L. F. S.; HRUSCHKA, E. R.; ACHARYA, A.; GHOSH, J., *A Differential Evolution Algorithm to Optimise the Combination of Classifier and Cluster Ensembles*, International Journal of Bio-Inspired Computation, v. 7, n. 2, p. 111-124, 2015 — (COLETTA *et al.*, 2015a).

O algoritmo proposto neste capítulo e seus resultados experimentais oferecem uma base para o desenvolvimento de novas variantes do algoritmo C³E-SL. Note que o trabalho descrito aqui cumpre o primeiro objetivo desta pesquisa, o qual envolveu a concepção de abordagens evolutivas para facilitar a combinação de classificadores e agrupadores. Esta contribuição abre espaço para o uso do algoritmo C³E-SL em aplicações práticas. Os próximos capítulos abordam problemas de classificação de dados pouco explorados. Por exemplo, a maioria dos classificadores assume que os dados de treinamento e os dados do conjunto alvo provêm de uma mesma distribuição de probabilidades, além de não considerarem o surgimento de novas classes não observadas durante o treinamento do modelo (LAMPERT; NICKISCH; HARMELING, 2014;

SCHEIRER *et al.*, 2013; MASUD *et al.*, 2010; SONG; RAJASEKARAN, 2008; JIA *et al.*, 2005). Tais dificuldades podem emergir especialmente quando não se dispõe de uma amostragem adequada de dados de treinamento e/ou se enfrenta a escassez de dados rotulados (cenário comum de aprendizado *semisupervisionado* (GOLDBERG, 2010; CHAPELLE; SCHÖLKOPF; ZIEN, 2006)). Nesses casos, será mostrado que o C^3E -SL (otimizado por meio do D^2E) é útil e promissor.

APRENDIZADO SEMISSUPERVISIONADO COMBINANDO CLASSIFICADORES E AGRUPADORES

No aprendizado supervisionado, considera-se que o conjunto de treinamento contém apenas objetos rotulados. Deste modo, os objetos podem ser vistos como tuplas $\langle \mathbf{x}_i^T, C_\ell^i \rangle$ que relacionam a ℓ -ésima classe com o i -ésimo objeto do conjunto de treinamento $\mathcal{X}^T$ (conforme formalizado no Capítulo 1). A tarefa de classificação, então, tem como objetivo realizar a predição da classe de novos objetos (não rotulados). Em outras palavras, busca-se inferir uma função f de tal forma que $\{\langle \mathbf{x}_i^T, C_\ell^i \rangle, i = 1, \dots, N; \ell = 1, \dots, c\} \mapsto f$, a qual espera-se ser apropriada para rotular objetos não observados. O algoritmo C³E-SL (apresentado na Seção 2.1.2) tem sido usado para induzir (refinar) modelos supervisionados. Entretanto, na prática, pode-se encontrar obstáculos para este tipo de aprendizagem. Em muitas aplicações reais, embora seja relativamente fácil adquirir dados não rotulados, o esforço de rotulação é tipicamente caro. Este fato motivou o surgimento do que se convencionou a chamar de aprendizado *semissupervisionado* (GOLDBERG, 2010; ZHU; GOLDBERG, 2009; CHAPELLE; SCHÖLKOPF; ZIEN, 2006).

No aprendizado semissupervisionado, presume-se que o conjunto de treinamento possui, além de objetos rotulados, também objetos que não estão rotulados (sendo estes últimos comumente em maior quantidade). Este tipo de aprendizado pode ser definido como *indutivo* ou *transdutivo*:

Caso Indutivo: busca-se prever a classe de novos objetos, como é feito normalmente em aprendizado supervisionado, porém assumindo-se que se dispõe também dos objetos não rotulados $\{\mathbf{x}_i^T, i = N + 1, \dots, U\}$ no conjunto de treinamento. O problema consiste, então, em inferir uma função f de tal forma que $\{\langle \mathbf{x}_i^T, C_\ell^i \rangle, i = 1, \dots, N; \ell = 1, \dots, c\} \cup \{\mathbf{x}_i^T, i = N + 1, \dots, U\} \mapsto f$.

Caso Transdutivo: objetiva-se apenas prever a classe dos objetos não rotulados no próprio conjunto de treinamento de tal forma que $\{ \langle \mathbf{x}_i^T, C_\ell^i \rangle, i = 1, \dots, N; \ell = 1, \dots, c \} \cup \{ \mathbf{x}_i^T, i = N + 1, \dots, U \} \mapsto \{ C_\ell^i, i = N + 1, \dots, U \}$.

A partir da formalização apresentada, este capítulo introduz uma extensão do algoritmo C³E-SL para o contexto semissupervisionado indutivo. Como uma prova de conceito, esta nova abordagem foi aplicada na área de análise de sentimentos (LIU, 2012; TABOADA *et al.*, 2011; LIU, 2010; PANG; LEE, 2008). O objetivo da análise de sentimentos é a identificação de opiniões, emoções e atitudes contidas em fontes de texto, tais como textos provenientes de avaliações de produtos (TURNER, 2002; HU; LIU, 2004a; HU; LIU, 2004b), *blogs* (HE *et al.*, 2008; MELVILLE; GRYC; LAWRENCE, 2009), notícias (BALAHUR *et al.*, 2010) e redes sociais, como o *Twitter* (SPERIOSU *et al.*, 2011; AGARWAL *et al.*, 2011; ZHANG *et al.*, 2011; MOHAMMAD; KIRITCHENKO; ZHU, 2013). *Twitter*, especificamente, é um *microblogging* bem popular, no qual os usuários postam mensagens curtas (com até 140 caracteres) chamadas de *tweets*. Este ambiente representa uma das maiores e mais dinâmicas bases de dados com conteúdo gerado por usuários (RITTER *et al.*, 2011; SPERIOSU *et al.*, 2011). Os *tweets* podem expressar ideias sobre diversos tópicos, como opiniões de consumidores a respeito de marcas/produtos (JANSEN *et al.*, 2009), a aceitação/rejeição de políticos (DIAKOPOULOS; SHAMMA, 2010), manifestações de *bullying* (XU *et al.*, 2012), além de eventos que possam gerar insegurança (CHEONG; LEE, 2011).

A análise de sentimentos em *tweets* é uma tarefa bastante desafiadora porque, além do alto custo de se obter dados rotulados, usualmente enfrenta-se as seguintes dificuldades: (i) alta frequência de erros ortográficos e gírias nas mensagens, visto que os usuários normalmente postam textos (com vocabulários específicos) usando diferentes dispositivos eletrônicos (e.g., telefones celulares e *tablets*); e (ii) grande variedade de tópicos — ao contrário de outras fontes, as quais geralmente são limitadas por um contexto bem particular. De maneira geral, a análise de sentimentos em *tweets* é reconhecida como um problema de classificação. Ou seja, analogamente ao que é feito com grandes documentos textuais, os *tweets* podem ser atrelados a diferentes rótulos (de classe) para se expressar uma opinião (MONTORO; MARTÍNEZ-BARCO; BALAHUR, 2012; EKMAN, 1982). Nesta tese, presume-se que *tweets* possam expressar uma opinião *positiva*, *negativa* ou *neutra* sobre diferentes tópicos, a exemplo de estudos similares na literatura (LIU, 2012; LIU, 2006; LIU, 2010; AGARWAL; BHATTACHARYYA, 2005).

A abordagem semissupervisionada proposta aqui estende o estudo de Coletta *et al.* (2014), no qual a combinação de classificadores e agrupadores foi usada para classificar *tweets* de maneira supervisionada. Esta extensão, descrita a partir da Seção 4.1, utiliza o algoritmo C³E-SL para combinar máquinas de vetores de suporte (*Support Vector Machines* — SVMs (CORTES; VAPNIK, 1995; BOSER; GUYON; VAPNIK, 1992)) com a informação de similaridades entre *tweets* de um conjunto alvo que se deseja classificar. Neste contexto, o C³E-SL se torna parte de um processo iterativo de autotreinamento (*self-training* (ZHU; GOLDBERG, 2009)), no

qual predições mais confiáveis aprimoram o modelo de classificação final. Visto que esta ideia atua com uma quantidade relativamente pequena de dados rotulados, o classificador gerado é especialmente útil para a classificação de *tweets*. Em particular, o experimento pedagógico da Seção 4.1.2 mostra como a informação não supervisionada do C³E-SL auxilia também na aprendizagem semissupervisionada. Experimentos comparativos mais elaborados são reportados e discutidos a partir da Seção 4.2.

4.1 Abordagens para Aprendizado Semissupervisionado

Aprendizado semissupervisionado situa-se em um meio termo entre aprendizado supervisionado e não supervisionado, uma vez que se aproveita tanto de dados rotulados quanto não rotulados para induzir modelos preditivos (GOLDBERG, 2010). Neste contexto, Nigam *et al.* (2000) projetaram um modelo generativo que utiliza o algoritmo *Expectation-Maximization* (EM) (DEMPSTER; LAIRD; RUBIN, 1977) para gerar misturas de Gaussianas a partir de dados não rotulados. Estas distribuições compensam a falta de conhecimento *a priori* e ajudam a melhorar a capacidade de generalização do modelo. Semelhantemente, modelos discriminativos têm sido também estendidos para o contexto semissupervisionado. Por exemplo, e como uma extensão das máquinas de vetores de suporte (CORTES; VAPNIK, 1995; BOSER; GUYON; VAPNIK, 1992), *Semi-Supervised SVMs* (S3VMs) encontram uma fronteira de máxima margem usando dados rotulados e não rotulados (XU *et al.*, 2008; SINDHWANI; KEERTHI, 2006; JOACHIMS, 1999b; JOACHIMS, 1999a).

A teoria dos grafos também tem ajudado a modelar aprendizado semissupervisionado. Vários estudos da literatura têm representado objetos rotulados e não rotulados como sendo vértices de um grafo (YONG *et al.*, 2014; GOLDBERG; ZHU, 2006; ZHU, 2005; BELKIN; MATVEEVA; NIYOGI, 2004; ZHOU *et al.*, 2004; ZHU *et al.*, 2003; JAAKKOLA; SZUMMER, 2002; BLUM; CHAWLA, 2001). Arestas (com pesos) são, então, usadas para definir as similaridades entre pares de vértices. A partir desta representação, algoritmos basicamente propagam a informação de vértices rotulados para os demais vértices (assumindo-se suavidade na propagação) até que algum critério de convergência seja alcançado. Espera-se que, neste processo, dois objetos *similares* possam (até certo ponto) compartilhar o mesmo rótulo de classe. Infelizmente, a maioria dos algoritmos baseados em grafos possuem ordem de complexidade computacional cúbica, tornando a aplicação limitada a pequenas bases de dados (ZHU, 2005).

Modelos semissupervisionados também são reproduzidos como um processo iterativo que pode ter como base componentes supervisionados. Blum e Mitchell (1998), por exemplo, utilizam dois componentes supervisionados que “ensinam” cada um o outro de forma iterativa. Estes componentes operam em diferentes espaços de atributos, fazendo uma analogia as diferentes *visões* que se podem ter dos dados. Os objetos com maior confiança nas predições de cada um dos componentes são incorporados no conjunto de dados rotulados (o qual, então, é usado para

retreinar os componentes). Este processo, denominado de *Co-training*, tem sido estudado (e melhorado) por inúmeros autores (LIU *et al.*, 2013b; WANG; ZHOU, 2010; WAN, 2009; ZHOU; LI, 2005). Com ideia similar, algoritmos de autotreinamento — *self-training* (SCUDDER, 1965) — usam predições de um único componente supervisionado para melhorar o modelo. A aprendizagem se inicia com um classificador treinado a partir de (poucos) dados rotulados. A cada iteração, após a predição de novos dados, promove-se para o conjunto de dados rotulados os objetos para os quais há grande confiança sobre as estimativas dos rótulos de classe. A partir disso, o classificador é reconstruído e usado para classificar novos dados numa próxima iteração. Esta prática funciona bem se os objetos promovidos estão atrelados aos rótulos de classes corretas (ZHU; GOLDBERG, 2009). Além disso, *self-training* se destaca por ser uma abordagem bem flexível, na qual praticamente qualquer algoritmo (supervisionado) pode ser usado como componente. Muitos estudos, particularmente para a análise de sentimentos, têm sido conduzidos a partir de *self-training* (BECKER *et al.*, 2013; HE; ZHOU, 2011; CRAMMER; KULESZA; DREDZE, 2009; QIU *et al.*, 2009).

A seguir, uma nova abordagem semissupervisionada é descrita, a qual explora estruturas intrínsecas dos dados para refinar modelos discriminativos com o algoritmo C³E-SL em um procedimento de *self-training*.

4.1.1 C³E-SL Semissupervisionado

O algoritmo C³E-SL, conforme abordado na Seção 2.1.2, recebe distribuições de probabilidades de classes que serão refinadas em um processo de otimização. Tipicamente, estas distribuições são resultantes de uma agregação de classificadores (COLETTA *et al.*, 2015b; COLETTA *et al.*, 2014; ACHARYA *et al.*, 2014; ACHARYA *et al.*, 2011), mas o uso de um único classificador operando de maneira independente também é possível. Mais detalhadamente, a versão semissupervisionada do C³E-SL refina resultados de um classificador SVM. Neste ponto, cumpre observar que o SVM possui grande capacidade de generalização, além de alcançar melhores resultados do que algoritmos comumente usados na área de análise de sentimentos — como o *Naive Bayes* e a Regressão Logística (NAKOV *et al.*, 2013). Seguindo Mohammad, Kiritchenko e Zhu (2013), adotou-se uma SVM com um *kernel* linear e parâmetro $C = 0.005$.

Além das distribuições de probabilidades de classes para objetos de um conjunto alvo, o C³E-SL requer também uma matriz de similaridades construída a partir deste conjunto. Em (ACHARYA *et al.*, 2014; ACHARYA *et al.*, 2011; COLETTA *et al.*, 2014) esta matriz provém de agregadores de agrupadores. Já na versão semissupervisionada do algoritmo, a matriz é obtida diretamente dos dados brutos seguindo os três passos a seguir:

Passo I - Geração de uma matriz de distâncias (D): formada pelo cômputo das distâncias Euclidianas entre pares de objetos presentes no conjunto alvo;

Passo II - Normalização da matriz D: para conter somente valores entre 0 e 1. A Equação

(4.1) é usada para calcular a distância normalizada, $\overline{d}_{ij}$, para os objetos i e j , sendo d_{ij} a distância Euclidiana calculada no Passo I. Os termos d_{min} e d_{max} referem-se, respectivamente, às distâncias mínimas e máximas observadas.

$$\overline{d}_{ij} = \frac{d_{ij} - d_{min}}{d_{max} - d_{min}} \quad (4.1)$$

Passo III - Obtenção da matriz de similaridades (S): a matriz **D** normalizada pode ser convertida em uma matriz de similaridades, **S**, calculando-se para cada um de seus elementos:

$$s_{ij} = 1 - \overline{d}_{ij} . \quad (4.2)$$

Tendo definidas as entradas do C³E-SL, a abordagem semissupervisionada do algoritmo é sumarizada nas Figuras 18–20. Em particular, a Figura 18 ilustra a calibração do C³E-SL a partir de um procedimento de validação cruzada que estima os parâmetros α e I do algoritmo. Ou seja, assim como sugerido na Seção 3.2, dividiu-se os objetos rotulados em um conjunto de treinamento e outro de validação. Um classificador SVM é induzido a partir do conjunto de treinamento e, então, o algoritmo D²E (apresentado na Seção 3.3) é usado para estimar os valores “ótimos” dos parâmetros no conjunto de validação (no qual uma matriz de similaridades foi produzida usando-se os Passos I–III). Os valores de α^* e I^* (“ótimos”) são, então, fixados e utilizados para a classificação de dados não rotulados nos conjuntos de teste (alvo). Esta etapa é apresentada na Figura 19. Como resultado, obtém-se um conjunto de vetores, $\{\mathbf{y}_j\}_{j=1}^n$, que são estimativas refinadas das distribuições de probabilidades de classes dos objetos do conjunto alvo. A partir destas estimativas, alguns objetos podem ser promovidos para o conjunto de dados rotulados como parte da prática de *self-training* (ZHU; GOLDBERG, 2009). Portanto, em resumo, o C³E-SL é empregado em uma estrutura algorítmica para desempenhar *self-training*. Assim, assume-se que o modelo de classificação seja realimentado com novos dados a partir de um processo iterativo que aproveita a informação não supervisionada para aumentar a confiança das predições. A Figura 20 apresenta o esquema completo do C³E-SL semissupervisionado.

4.1.2 Exemplo Pedagógico

Esta seção apresenta um experimento instrutivo para elucidar o funcionamento do C³E-SL semissupervisionado. A Tabela 12 descreve quatro *tweets* provenientes da base de dados *Twitter2014* (ROSENTHAL *et al.*, 2014). Na Figura 21, estes *tweets* são representados pelo número de léxicos positivos e negativos — estes atributos, como será visto a seguir, são atributos usados para se construir a matriz de similaridades (**S**). Mais especificamente, para t_1 há dois léxicos positivos: “wins” e “earned”, e não há léxicos negativos. Para t_2 há dois léxicos positivos: “better” e “better”, mas há dois léxicos negativos também: “sad” e “can’t”. Para t_3 há apenas um léxico positivo, “ready”, e não há negativos. Para t_4 , finalmente, há um léxico positivo (“decent”) e um negativo (“not”). Estes termos estão destacados na Tabela 12. Conforme

Figura 18 – Calibração do C³E-SL — parâmetros α e I são estimados em um conjunto de validação por meio do algoritmo D²E (COLETTA *et al.*, 2015a).

Figura 19 – Os valores “ótimos” para α e I são fixados e usados no C³E-SL para refinar resultados do SVM em um conjunto alvo (com dados não rotulados que se deseja predizer).

Figura 20 – Esquema do C³E-SL semissupervisionado.

ilustrado em (a) da Figura 21, o *tweet* t_1 pertence à classe *positiva*, enquanto que t_2 , t_3 e t_4 pertencem à classe *neutra*.

Modelos de classificação foram induzidos a partir de um conjunto de dados rotulados que representou 5% da base de dados *SemEval2013* (NAKOV *et al.*, 2013). Em (b) e (c) da Figura 21 nota-se que o classificador SVM (operando individualmente) não foi capaz de prever corretamente os rótulos de classes dos quatro *tweets* em destaque. Em (c), em particular, avaliou-

Tabela 12 – Exemplo de *tweets* da base de dados *Twitter2014* (ROSENTHAL *et al.*, 2014).

ID	Texto
t_1	@LUFCSOCCER Gators soccer wins 12th SEC title: The three points earned put Florida on top with 33 ...
t_2	I hope my better half is feeling better today! I will be so sad if we can't climb Mount Snowdon tomorrow! Was so looking forward to it!
t_3	Getting ready for Knollwood P.S. bazaar, tonight and tomorrow!
t_4	@SeanJohnGerard @gmurph25 @tompcotter It's decent but not +3 standard yet, teeing it up at MJ on Monday. How'd you play at Lough Erne?

Figura 21 – Espaço bidimensional formado pela contagem dos léxicos positivos e negativos presentes nos *tweets* da Tabela 12. Em (a) são apresentadas as classes reais destes *tweets*. Em (b) e (c) são apresentadas as classificações com o uso do SVM (independente e como componente na abordagem de *self-training*). Em (d), por fim, é apresentada a predição feita pela abordagem do C³E-SL semissupervisionado.

se o SVM como componente supervisionado em um procedimento de autotreinamento. Por outro lado, ao se utilizar a abordagem semissupervisionada do C³E-SL (esquematizada na Figura 20) todos os *tweets* foram corretamente classificados — observe (a) e (d) na Figura 21. Neste caso, a informação não supervisionada codificada pela matriz de similaridades ajudou na predição correta destes *tweets*, pois o C³E-SL conseguiu refinar (e corrigir) classificações prévias feitas pelo SVM ao explorar estruturas intrínsecas dos dados em um espaço de atributos formado por léxicos positivos e negativos. A Tabela 13 apresenta parte da matriz de similaridades usada pelo

Tabela 13 – Valores de similaridades entre *tweets* contidos na matriz de similaridades usada pelo C³E-SL. Em destaque estão os valores máximos calculados para os *tweets* t_1 , t_{20} e t_{1853} .

	t_1	t_2	t_3	t_4	...	t_{20}	...	t_{50}	...	t_{125}	...	t_{1853}
t_1	1.0	0.5	0.5	0.0	...	1.0	...	0.3	...	0.0	...	1.0
t_2	0.5	1.0	0.2	0.3	...	0.5	...	1.0	...	0.2	...	0.0
t_3	0.5	0.2	1.0	0.5	...	0.5	...	0.5	...	0.8	...	0.4
t_4	0.0	0.3	0.5	1.0	...	0.2	...	0.6	...	0.7	...	0.2
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
t_{20}	1.0	0.5	0.5	0.2	...	1.0	...	0.2	...	0.3	...	1.0
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
t_{50}	0.3	1.0	0.5	0.3	...	0.2	...	1.0	...	0.3	...	0.2
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
t_{125}	0.0	0.2	0.8	0.7	...	0.3	...	0.3	...	1.0	...	0.2
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
t_{1853}	1.0	0.0	0.3	0.1	...	1.0	...	0.2	...	0.2	...	1.0

C³E-SL. Nesta matriz, o *tweet* t_1 tem similaridade máxima com relação aos *tweets* t_{20} e t_{1853} — pois possuem a mesma contagem de léxicos positivos e negativos. Os *tweets* t_{20} e t_{1853} pertencem à classe *positiva* e, assim, somaram evidências para que, durante o processo de otimização do C³E-SL, t_1 pudesse ser classificado como sendo da classe *positiva* também. Portanto, este é um exemplo que mostra que o refinamento produzido pelo C³E-SL ajuda a melhorar classificações e, em especial, aumenta as chances de realimentar o conjunto de dados rotulados com objetos (*tweets*) corretos a partir de autotreinamento.

4.2 Avaliação Empírica na Classificação de Tweets

Na análise de sentimentos em *tweets*, a obtenção de dados rotulados depende de um processo custoso de anotação manual que muitas vezes necessita de conhecimento de domínio (e de especialistas). Isso limita bastante a quantidade de *tweets* rotulados oferecidos para classificadores. Os experimentos se basearam nestes cenários para explorar as capacidades do C³E-SL semissupervisionado (introduzido na Seção 4.1.1). Análises empíricas foram conduzidas de maneira comparativa a partir de bases de dados representativas da área.

A Tabela 14 apresenta as características das bases de dados usadas nos experimentos. Estas bases de dados são recomendadas pelo *International Workshop on Semantic Evaluation* (SemEval)¹, cujos organizadores sugerem particularmente o uso da base de dados *SemEval2013* como fonte de dados rotulados para o treinamento dos modelos (NAKOV *et al.*, 2013; ROSENTHAL *et al.*, 2014). Os modelos induzidos foram, então, avaliados em cinco conjuntos de teste (alvo), os quais, na realidade, são bases de dados amplamente conhecidas da área: *LiveJournal*, *SMS2013*, *Twitter2013*, *Twitter2014* e *TwitterSarcasm2014*. As bases de dados *LiveJournal* e *SMS2013* foram usadas com o objetivo de investigar como classificadores treinados

¹ <<http://en.wikipedia.org/wiki/SemEval>>

Tabela 14 – Distribuição das classes *positiva*, *negativa* e *neutra* nas seis bases de dados utilizadas. Os classificadores foram treinados com *tweets* rotulados da base de dados *SemEval2013* (NAKOV *et al.*, 2013). Os resultados reportados foram obtidos a partir de classificações em cinco conjuntos de teste — *LiveJournal*, *SMS2013*, *Twitter2013*, *Twitter2014* e *TwitterSarcasm2014*.

Base de Dados	Classe Positiva	Classe Negativa	Classe Neutra	Total
Dados rotulados para a aprendizagem dos modelos				
<i>SemEval2013</i>	4,215 (37%)	1,807 (15%)	5,325 (48%)	11,338
Novos dados para classificação (conjuntos de teste)				
<i>LiveJournal</i>	427 (37%)	304 (27%)	411 (36%)	1,142
<i>SMS2013</i>	492 (23%)	394 (19%)	1,207 (58%)	2,093
<i>Twitter2013</i>	1,572 (41%)	601 (16%)	1,640 (43%)	3,813
<i>Twitter2014</i>	982 (53%)	202 (11%)	669 (36%)	1,853
<i>TwitterSarcasm2014</i>	33 (38%)	40 (47%)	13 (15%)	86

a partir de mensagens de *Twitter* se comportam em outras fontes de dados (i.e., de *weblogs* e mensagens de celular). Já as bases de dados *Twitter2013*, *Twitter2014* e *TwitterSarcasm2014* foram obtidas pela identificação de entidades nomeadas extraídas de milhões de *tweets* coletados em um período entre Janeiro de 2012 e Janeiro de 2013 através da API pública do *Twitter*. A partir do reconhecimento de temas populares relacionados às entidades nomeadas (os quais são frequentemente mencionados em associação com uma data específica), os *tweets* foram manualmente rotulados. A base de dados *Twitter2013* possui três temas em comum com os temas presentes no conjunto usado para treinamento (i.e., *SemEval2013*). As bases de dados *Twitter2014* e *TwitterSarcasm2014* foram formadas mais recentemente e não contêm necessariamente temas presentes em *SemEval2013* — a última base de dados, em particular, foi coletada através da hashtag *#sarcasm* com o intuito de determinar como o sarcasmo afeta a polaridade dos *tweets*. Todas as bases de dados foram rotuladas por meio de anotadores cadastrados no *Amazon Mechanical Turk*².

A forma de representação de *tweets* não é consistentemente semelhante nos diferentes trabalhos encontrados na literatura. Por exemplo, muitas abordagens têm usado apenas atributos gerados a partir de *N*-gramas (BAUGH, 2013), enquanto que outras incluem também informações de *emoticons* (LIU *et al.*, 2013b; LIU *et al.*, 2013a). Alguns trabalhos mais recentes também consideram um espaço de atributos mais complexo derivado de *part-of-speech tags*, léxicos e *hashtags* (BECKER *et al.*, 2013; XIANG; ZHOU, 2014b; ZHAO; LAN; ZHU, 2014). O conjunto de atributos adotado nesta pesquisa foi inspirado no trabalho de Mohammad, Kiritchenko e Zhu (2013), que conquistou o primeiro lugar na competição promovida pelo evento SemEval 2013 (NAKOV *et al.*, 2013). Esta representação também alcançou os melhores resultados durante o SemEval 2014 — nas bases de dados *LiveJournal*, *SMS2013* e *TwitterSarcasm2014* (ROSENTHAL *et al.*, 2014). O conjunto de atributos adotado foi formado por:

(i) *N*-gramas: um *N*-grama é uma sequência contínua de *N* itens (palavras) presentes em uma

² <<https://www.mturk.com>>

fonte textual ou falada. Foram considerados unigramas, bigramas e trigramas, os quais são subsequências formadas, respectivamente, por uma, duas e três palavras. Este modelo de linguagem é baseado em *bag-of-words*, no qual um atributo representa a frequência com que um N -grama aparece na mensagem considerada (PANG; LEE; VAITHYANATHAN, 2002).

- (ii) *Léxicos*: geram atributos que representam a contagem de certas palavras consideradas positivas ou negativas de acordo com o método descrito em (MOHAMMAD; KIRITCHENKO; ZHU, 2013);
- (iii) *Negação*: indica o número de contextos negados. Segundo Pang, Lee e Vaithyanathan (2002), um contexto negado é um segmento de mensagem que se inicia com uma palavra de negação (e.g., “no”, “shouldn’t”) e termina com uma pontuação (vírgula, ponto, dois pontos, ponto e vírgula, ponto de exclamação ou ponto de interrogação). Um contexto negado afeta os atributos formados por N -gramas e léxicos. Assim, adiciona-se o sufixo “NEG” em cada palavra (e.g., “good” torna-se “good_NEG”). Uma lista de palavras de negação foi adotada a partir do tutorial produzido por Christopher Potts³;
- (iv) *Part-of-speech*: usando-se o aplicativo Ark-twitter NLP (OWOPUTI *et al.*, 2013) extraiu-se *tags* de *part-of-speech* (e.g., *emoticons*, *hashtags*, adjetivos, advérbios, dentre outros). O número de ocorrências de cada *tag* é representado na forma de atributo;
- (v) *Estilo de Escrita*: atributos foram derivados a partir da presença de três ou mais caracteres repetidos em palavras, da sequência de três ou mais pontuações e do número de palavras com todas as letras em maiúsculo;
- (vi) *Atributos do Microblogging*: considerou-se atributos que contam o número total de *hashtags* de sentimentos no texto de léxicos e *emoticons* (HU; LIU, 2004a; THELWALL *et al.*, 2010; MOHAMMAD; KIRITCHENKO; ZHU, 2013).

Para a extensão semissupervisionada do C³E-SL, o componente supervisionado (SVM) foi treinado no espaço de atributos formado pelos itens (i)–(vi), enquanto que os atributos responsáveis pela criação da matriz de similaridades correspondem aos léxicos e contextos negados — ou seja, os itens (i)–(iii). Tal configuração visa minimizar os efeitos negativos da *maldição da dimensionalidade* e tornar o componente não supervisionado mais simples e interpretável.

A avaliação experimental levou em conta o cenário de aprendizado semissupervisionado indutivo. Uma proporção p de *tweets* (rotulados) foi aleatoriamente amostrada a partir da base de dados *SemEval2013* (NAKOV *et al.*, 2013) (veja a Tabela 14) — manteve-se o balanço entre as classes *positiva*, *negativa* e *neutra*. Os *tweets* contidos na importância sobressalente ($1 - p$)

³ <<http://sentiment.christopherpotts.net/lingstruc.html>>

foram usados na fase de aprendizagem das abordagens. Seguindo a filosofia do *self-training*, a cada iteração, 100 *tweets* foram incorporados no conjunto de dados rotulados (mantendo-se a proporção das classes). Este procedimento foi repetido até todos os *tweets* da base de dados *SemEval2013* serem rotulados e promovidos. Os modelos resultantes foram usados para classificar *tweets* nas bases de dados de teste (*LiveJournal*, *SMS2013*, *Twitter2013*, *Twitter2014* e *TwitterSarcasm2014*). Os resultados para cada uma destas bases de dados foram anotados e são reportados a seguir. Adotou-se como medida de desempenho o F-score para as classes *positivas* e *negativas*, bem como o F-score geral dado por $\overline{F1} = (F1_{Pos} + F1_{Neg})/2$ (BECKER *et al.*, 2013; BAUGH, 2013; ZHAO; LAN; ZHU, 2014; XIANG; ZHOU, 2014b).

4.2.1 Análise Comparativa

O C³E-SL semissupervisionado foi comparado com uma abordagem similar, a qual usa o classificador SVM como componente do autotreinamento (*self-training*). O conjunto inicial de *tweets* rotulados foi composto por 5% da base de dados *SemEval2013* (realizou-se amostragem aleatória estratificada). As médias e desvios padrões referentes aos resultados nos conjuntos de teste estão sumarizados nas Tabelas 15–19 (para obter estas estatísticas de interesse, repetiu-se o experimento 10 vezes para cada base de dados). Como referência, são reportados também os resultados para o SVM atuando de maneira independente (sem *self-training*). Os resultados deste SVM independente, denominado de SVM supervisionado, correspondem às médias de resultados de classificação computados para todas as máquinas de vetores de suporte induzidas a partir de conjuntos de treinamento de mesmo tamanho daqueles formados em cada iteração das abordagens de *self-training*. Assim, enquanto o SVM semissupervisionado (baseado em autotreinamento) possui interdependência das predições em diferentes iterações, cada rodada do classificador SVM supervisionado independe de suas outras execuções (por selecionar aleatoriamente conjuntos de treinamento em cada uma delas). Esta configuração favorece a observação das vantagens de uso do *self-training* no contexto das máquinas de vetores de suporte.

As Tabelas 15–19 mostram que as abordagens iterativas oferecem resultados tipicamente melhores que o SVM operando de maneira independente. De fato, o processo desempenhado pelo *self-training* gera um conjunto de treinamento mais confiável e capaz de oferecer melhores resultados, sobretudo com poucos dados rotulados (cenário típico de aprendizado semissupervisionado). A versão semissupervisionada do C³E-SL, em particular e comparativamente às abordagens que usam apenas o SVM, alcançou, na maioria das vezes, os melhores resultados. Mais especificamente, em todas as bases de dados, o C³E-SL semissupervisionado produziu os maiores valores de $\overline{F1}$ (desempenho geral considerando as 10 execuções), além de ter sido relativamente mais estável, gerando os menores desvios padrões para *SMS2013*, *Twitter2014* e *TwitterSarcasm2014*. Isto indica que o C³E-SL semissupervisionado é mais robusto e tem menos dependência dos dados rotulados. Em contrapartida, algumas vezes o desempenho das abordagens iterativas (semissupervisionadas) foi inferior ao alcançado pelo SVM supervisionado

Tabela 15 – F-scores (%) obtidos pelas abordagens semissupervisionadas e o SVM supervisionado na base de dados *LiveJournal* (MIURA *et al.*, 2014). Melhores resultados estão em negrito.

Execuções	SVM Supervisionado (Independente)			SVM Semissupervisionado			C ³ E-SL Semissupervisionado		
	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$
#1	61.10	52.31	56.71	68.40	53.20	60.80	70.07	54.81	62.44
#2	67.19	50.31	58.75	67.71	50.75	59.23	68.59	54.90	61.75
#3	50.30	46.05	48.18	62.97	50.66	56.82	63.47	46.88	55.18
#4	56.66	40.47	48.56	67.42	46.54	56.98	68.05	47.59	57.82
#5	66.85	60.91	63.88	67.51	57.89	62.70	68.38	62.36	65.37
#6	62.19	41.47	51.83	66.12	43.27	54.69	67.24	46.02	56.63
#7	67.56	50.42	58.99	66.46	54.41	60.43	68.23	55.60	61.92
#8	68.38	35.53	51.96	66.03	47.64	56.83	67.72	50.53	59.13
#9	62.64	38.41	50.53	63.76	37.96	50.86	66.88	44.94	55.91
#10	63.56	37.41	50.48	63.76	51.21	57.48	66.32	44.34	55.33
$\overline{F1}$	62.64	45.33	53.99	66.01	49.35	57.68	67.50	50.80	59.15
Desv. Pad.	05.65	08.08	05.26	01.90	05.75	03.37	01.74	05.90	03.54

Tabela 16 – F-scores (%) obtidos pelas abordagens semissupervisionadas e o SVM supervisionado na base de dados *SMS2013* (NAKOV *et al.*, 2013). Melhores resultados estão em negrito.

Execuções	SVM Supervisionado (Independente)			SVM Semissupervisionado			C ³ E-SL Semissupervisionado		
	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$
#1	49.91	49.31	49.61	53.98	47.58	50.78	56.52	51.72	54.12
#2	54.92	47.09	51.01	60.00	40.68	50.34	59.94	49.16	54.55
#3	49.64	41.89	45.77	52.05	39.66	45.85	56.26	43.43	49.84
#4	51.98	39.56	45.77	53.93	42.87	48.40	57.96	46.11	52.04
#5	53.38	45.58	49.48	55.08	41.80	48.44	56.78	48.36	52.58
#6	48.67	39.59	44.13	49.31	38.56	43.94	54.05	42.38	48.21
#7	55.27	47.19	51.23	53.58	43.07	48.32	58.66	51.16	54.90
#8	54.53	37.04	45.78	53.47	41.33	47.40	58.26	47.18	52.72
#9	48.67	38.27	43.47	46.89	34.94	40.92	52.99	41.69	47.34
#10	47.92	39.12	43.52	49.01	44.66	46.83	49.61	46.64	48.12
$\overline{F1}$	51.49	42.46	46.98	52.73	41.52	47.12	56.10	46.78	51.44
Desv. Pad.	02.86	04.42	03.06	03.69	03.45	02.96	03.09	03.47	02.84

Tabela 17 – F-scores (%) obtidos pelas abordagens semissupervisionadas e o SVM supervisionado na base de dados *Twitter2013* (NAKOV *et al.*, 2013). Melhores resultados estão em negrito.

Execuções	SVM Supervisionado (Independente)			SVM Semissupervisionado			C ³ E-SL Semissupervisionado		
	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$
#1	60.50	45.66	53.08	64.98	45.83	55.41	63.99	48.44	56.21
#2	61.18	42.41	51.79	65.13	45.66	55.39	65.30	41.29	53.30
#3	57.55	42.62	50.08	62.21	41.91	52.06	67.95	41.30	54.63
#4	60.42	44.89	52.65	64.80	47.43	56.11	65.66	44.32	54.99
#5	61.43	44.97	53.20	64.74	42.73	53.73	65.32	39.02	52.17
#6	62.30	43.91	53.11	64.53	43.81	54.17	67.49	44.91	56.20
#7	60.80	43.13	51.97	63.73	44.70	54.21	68.62	45.64	57.13
#8	62.59	35.39	48.99	64.74	45.61	55.17	64.32	45.42	54.86
#9	60.00	41.91	50.96	62.16	41.27	51.72	68.62	42.27	55.44
#10	63.85	42.42	53.14	64.29	46.00	55.14	68.98	43.58	56.28
$\overline{F1}$	61.06	42.73	51.90	64.13	44.49	54.31	66.62	43.63	55.12
Desv. Pad.	01.70	02.88	01.47	01.10	02.00	01.46	01.90	02.71	01.49

(e.g., execuções #1 e #6 na Tabela 19). Pressupõe-se que, nestes casos, os dados rotulados representaram razoavelmente bem as distribuições das classes, enquanto que dados não rotulados podem naturalmente favorecer superajuste (*overfitting*). Neste contexto particular, dados não rotulados nem sempre são úteis para se alavancar a classificação — entretanto, como visto, na média geral vantagens significativas foram observadas ao se usar a informação não supervisionada por meio do algoritmo C³E-SL.

Tabela 18 – F-scores (%) obtidos pelas abordagens semissupervisionadas e o SVM supervisionado na base de dados *Twitter2014* (MIURA *et al.*, 2014). Melhores resultados estão em negrito.

Execuções	SVM Supervisionado (Independente)			SVM Semissupervisionado			C ³ E-SL Semissupervisionado		
	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$
#1	57.33	43.72	50.52	64.63	47.03	55.83	63.69	45.00	54.35
#2	60.84	42.13	51.49	64.92	44.33	54.62	64.76	44.80	54.79
#3	57.52	44.92	51.22	63.21	41.87	52.54	63.77	44.02	53.89
#4	58.73	44.32	51.53	62.43	46.31	54.37	64.40	42.66	53.54
#5	61.58	45.96	53.77	64.40	45.23	54.81	64.12	42.32	53.22
#6	60.64	40.87	50.76	64.12	43.26	53.69	65.30	45.01	55.15
#7	60.42	43.14	51.78	63.30	43.41	53.35	64.29	43.70	53.99
#8	61.59	40.12	50.85	64.25	45.15	54.70	65.39	47.96	56.68
#9	58.88	40.58	49.73	61.66	40.72	51.19	62.80	43.74	53.27
#10	62.73	45.58	54.16	63.03	45.23	54.13	64.23	47.15	55.69
$\overline{F1}$	60.02	43.14	51.58	63.60	44.25	53.92	64.28	44.64	54.46
Desv. Pad.	01.82	02.12	01.39	01.05	01.96	01.31	0.77	01.79	01.12

Tabela 19 – F-scores (%) obtidos pelas abordagens semissupervisionadas e o SVM supervisionado na base de dados *TwitterSarcasm2014* (MIURA *et al.*, 2014). Melhores resultados estão em negrito.

Execuções	SVM Supervisionado (Independente)			SVM Semissupervisionado			C ³ E-SL Semissupervisionado		
	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$
#1	50.00	25.53	37.77	50.63	13.64	32.13	50.75	21.74	36.24
#2	49.32	13.64	31.48	52.05	24.49	38.27	54.55	25.53	40.03
#3	46.15	17.39	31.77	50.63	13.64	32.13	50.00	21.28	35.64
#4	44.44	09.30	26.87	48.72	09.30	29.01	51.85	17.78	34.81
#5	55.70	33.96	44.83	52.63	24.49	38.56	56.47	34.62	45.54
#6	49.28	25.53	37.40	48.10	17.39	32.75	51.28	09.30	30.29
#7	58.54	13.64	36.09	55.42	25.53	40.48	56.41	21.74	39.07
#8	54.76	09.30	32.03	52.50	25.00	38.75	62.37	09.52	35.94
#9	49.35	17.78	33.56	51.85	21.28	36.56	53.16	17.78	35.47
#10	43.84	09.30	26.57	52.38	17.78	35.08	49.35	17.78	33.56
$\overline{F1}$	50.14	17.54	33.84	51.49	19.25	35.37	53.62	19.70	36.66
Desv. Pad.	04.86	08.38	05.46	02.10	05.77	03.74	03.96	0.73	0.41

4.2.2 Impactos da Quantidade de Dados Rotulados Disponíveis

Os experimentos da seção anterior foram conduzidos em um cenário que adotou 5% dos dados rotulados da base *SemEval2013*. Esta seção investiga os impactos da variação deste percentual, i.e., do tamanho do conjunto de dados rotulados que se dispõe inicialmente para gerar um modelo de classificação. Conforme já destacado, uma das principais limitações práticas da tarefa de classificação é a escassez de dados rotulados. Portanto, espera-se que as abordagens semissupervisionadas possam se comportar bem em cenários de significativa escassez de dados rotulados. Considerando o compromisso entre quantidade de dados rotulados e desempenho de classificação, as discussões aqui podem ajudar na escolha da alternativa mais vantajosa para o problema que se tem em mãos. As mesmas abordagens comparadas na seção anterior foram usadas — ou seja, levou-se em conta os resultados do C³E-SL semissupervisionado, de sua contraparte, a qual utiliza *self-training* com SVM, e também do SVM supervisionado operando de maneira independente (sem *self-training*).

As Figuras 22–26 apresentam os resultados gerais ($\overline{F1}$) para cada abordagem, considerando 10 execuções independentes a partir de uma certa quantidade (inicial) de dados rotulados. As quantidades usadas foram de 5%, 10%, 20%, 30% e 40% de dados rotulados amostrados da

base *SemEval2013*. Como esperado, os resultados melhoraram com o aumento da quantidade de dados rotulados disponíveis. Comparativamente, os resultados do C³E-SL semissupervisionado foram sensivelmente melhores, com destaque para seu desempenho nas bases de dados *LiveJournal*, *SMS2013*, *Twitter2014* e *TwitterSarcasm2014*. Na base de dados *Twitter2013*, as abordagens semissupervisionadas alcançaram resultados modestos em relação ao classificador SVM. Entretanto, convém destacar que, em todas as bases de dados, para quantidades relativamente pequenas de dados rotulados, as abordagens semissupervisionadas foram as mais vantajosas. A Tabela 20 exibe os valores numéricos do experimento conduzido, além de apresentar o desempenho de abordagens supervisionadas que utilizaram o conjunto de dados completo.

Figura 22 – $\overline{F1}$ (%) para diferentes percentuais de dados rotulados — *LiveJournal*.

Figura 23 – $\overline{F1}$ (%) para diferentes percentuais de dados rotulados — *SMS2013*.

Figura 24 – $\overline{F1}$ (%) para diferentes percentuais de dados rotulados — *Twitter2013*.

Figura 25 – $\overline{F1}$ (%) para diferentes percentuais de dados rotulados — *Twitter2014*.

Figura 26 – $\overline{F1}$ (%) para diferentes percentuais de dados rotulados — *TwitterSarcasm2014*.

A Tabela 21, que foca apenas nos resultados alcançados pelas abordagens semissupervisionadas, reporta a proporção de F-scores mais altos e menores desvios padrões encontrados na Tabela 20. Pode-se constatar, portanto, que o *self-training* baseado no C³E-SL mostra-se mais promissor do que se utilizar apenas o SVM como componente do autotreinamento.

4.3 Considerações Finais

Em (COLETTA *et al.*, 2014) combinou-se classificadores e agrupadores para produzir melhores modelos supervisionados para a classificação de *tweets*. Contudo, abordagens semissupervisionadas têm se mostrado alternativas mais promissoras neste contexto (BECKER *et al.*, 2013; HE; ZHOU, 2011; CRAMMER; KULESZA; DREDZE, 2009; QIU *et al.*, 2009). A partir desta perspectiva, uma abordagem semissupervisionada do algoritmo C³E-SL foi introduzida. Esta abordagem combina máquinas de vetores de suporte — *Support Vector Machines* (SVMs) (CORTES; VAPNIK, 1995; BOSER; GUYON; VAPNIK, 1992) — com a informação de similaridades entre objetos para se gerar classificações mais refinadas. Baseando-se em um procedimento de autotreinamento (ZHU; GOLDBERG, 2009), previsões mais confiáveis aprimoram o modelo de forma iterativa. Como é requerida uma quantidade pequena de dados rotulados, o C³E-SL semissupervisionado é particularmente útil para a classificação de *tweets*.

Tabela 20 – F-scores para diferentes percentuais de dados rotulados para abordagens semissupervisionadas e também supervisionadas — incluindo os melhores resultados da literatura ao se utilizar o conjunto completo de dados rotulados. Os melhores resultados estão em negrito.

LiveJournal																		
Abordagem	5%			10%			20%			30%			40%			100%		
	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$
C ³ E-SL Semissuper.	67.50 ±1.74	50.80 ±5.90	59.15 ±3.54	69.10 ±0.86	58.27 ±2.86	63.69 ±1.82	69.87 ±0.67	62.25 ±4.30	66.06 ±2.23	70.21 ±0.66	64.61 ±2.59	67.41 ±1.49	70.81 ±0.65	66.21 ±2.06	68.51 ±1.14	—	—	—
SVM Semissuper.	66.01 ±1.89	49.35 ±5.75	57.68 ±3.37	68.09 ±0.79	57.77 ±2.77	62.93 ±1.61	69.13 ±0.69	61.24 ±4.08	65.19 ±2.10	69.54 ±0.66	62.49 ±3.13	66.01 ±1.80	70.22 ±0.84	64.71 ±2.18	67.47 ±1.22	—	—	—
SVM Super.	62.64 ±5.64	45.33 ±8.08	53.98 ±5.26	66.54 ±3.21	55.35 ±5.58	60.95 ±3.39	68.33 ±1.57	59.69 ±5.64	64.01 ±2.90	69.01 ±0.87	60.82 ±5.38	64.92 ±2.87	69.21 ±1.18	63.34 ±3.05	66.27 ±1.71	68.84	61.75	67.34
Literatura	Abordagem supervisionada proposta por Zhu, Kiritchenko e Mohammad (2014)																	74.84
SMS2013																		
Abordagem	5%			10%			20%			30%			40%			100%		
	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$
C ³ E-SL Semissuper.	56.10 ±3.09	46.78 ±3.47	51.44 ±2.84	59.97 ±1.10	52.89 ±1.23	56.43 ±1.08	62.06 ±1.10	55.51 ±1.00	58.79 ±0.70	63.34 ±0.81	56.38 ±0.83	59.86 ±0.59	63.87 ±0.49	57.67 ±0.91	60.76 ±0.44	—	—	—
SVM Semissuper.	52.73 ±3.69	41.52 ±3.44	47.12 ±2.96	56.72 ±2.22	49.20 ±1.01	52.96 ±1.32	60.50 ±0.81	52.16 ±0.99	56.33 ±0.81	60.60 ±0.87	53.73 ±1.22	57.16 ±0.94	61.67 ±0.63	55.41 ±0.78	58.54 ±0.52	—	—	—
SVM Super.	51.48 ±2.86	42.46 ±4.42	46.97 ±3.05	56.74 ±2.12	50.85 ±1.29	53.80 ±1.53	60.35 ±1.56	53.77 ±1.06	57.06 ±1.07	61.56 ±1.15	55.15 ±1.28	58.36 ±1.02	62.78 ±0.62	56.72 ±1.04	59.74 ±0.54	59.31	51.69	55.50
Literatura	Abordagem supervisionada proposta por Zhu, Kiritchenko e Mohammad (2014)																	70.28
Twitter2013																		
Abordagem	5%			10%			20%			30%			40%			100%		
	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$
C ³ E-SL Semissuper.	66.62 ±1.90	43.62 ±2.71	55.12 ±1.49	66.10 ±0.65	45.58 ±1.60	55.84 ±0.94	66.63 ±0.56	48.06 ±1.64	57.35 ±0.79	67.38 ±0.41	48.80 ±0.88	58.09 ±0.53	67.86 ±0.32	49.26 ±0.76	58.56 ±0.44	—	—	—
SVM Semissuper.	64.13 ±1.09	44.49 ±1.99	54.31 ±1.46	66.38 ±0.39	46.90 ±1.54	56.64 ±1.41	66.78 ±0.41	47.67 ±0.50	57.23 ±0.28	67.13 ±0.47	48.33 ±1.06	57.73 ±0.42	67.68 ±0.26	49.15 ±0.76	58.42 ±0.37	—	—	—
SVM Super.	61.06 ±1.70	42.73 ±2.87	51.89 ±1.46	64.80 ±0.79	46.13 ±1.69	55.46 ±1.01	66.39 ±0.50	48.29 ±1.37	57.34 ±0.77	66.83 ±0.65	48.88 ±0.83	57.85 ±0.51	67.50 ±0.29	49.40 ±0.83	58.45 ±0.43	70.70	61.41	66.05
Literatura	Abordagem supervisionada proposta por Miura et al. (2014)																	72.12
Twitter2014																		
Abordagem	5%			10%			20%			30%			40%			100%		
	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$
C ³ E-SL Semissuper.	64.28 ±0.77	44.64 ±1.80	54.46 ±1.12	65.88 ±0.91	46.26 ±1.57	56.07 ±1.03	66.73 ±0.79	48.45 ±2.12	57.59 ±1.08	67.80 ±0.42	48.69 ±1.06	58.25 ±0.59	68.18 ±0.56	49.57 ±0.71	58.88 ±0.51	—	—	—
SVM Semissuper.	63.59 ±1.04	44.25 ±1.95	53.92 ±1.31	66.01 ±0.79	46.26 ±2.05	56.13 ±1.34	66.71 ±0.73	47.33 ±1.78	57.02 ±1.08	67.31 ±0.52	47.40 ±1.37	57.35 ±0.82	68.06 ±0.35	48.33 ±1.54	58.19 ±0.90	—	—	—
SVM Super.	60.02 ±1.82	43.13 ±2.12	51.58 ±1.38	65.17 ±1.25	45.29 ±2.28	55.23 ±1.50	66.44 ±0.79	47.61 ±1.97	57.12 ±0.98	67.57 ±0.75	48.30 ±1.39	57.93 ±0.75	68.14 ±0.44	49.26 ±1.08	58.70 ±0.64	71.87	55.51	63.69
Literatura	Abordagem supervisionada proposta por Miura et al. (2014)																	70.96
TwitterSarcasm2014																		
Abordagem	5%			10%			20%			30%			40%			100%		
	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$	$F1_{Pos}$	$F1_{Neg}$	$\overline{F1}$
C ³ E-SL Semissuper.	53.62 ±3.96	19.71 ±7.38	36.66 ±4.12	54.85 ±2.89	23.13 ±4.65	38.98 ±2.40	54.51 ±1.71	23.46 ±4.97	38.99 ±2.42	55.75 ±2.43	25.86 ±2.73	40.81 ±1.54	58.19 ±1.73	23.93 ±2.53	41.06 ±0.97	—	—	—
SVM Semissuper.	51.49 ±2.09	19.25 ±5.76	35.37 ±3.74	52.85 ±1.55	23.99 ±5.24	38.42 ±3.03	53.49 ±1.79	23.62 ±4.33	38.55 ±2.29	54.21 ±1.41	21.62 ±4.68	37.91 ±2.71	54.64 ±1.02	22.06 ±3.34	38.35 ±1.72	—	—	—
SVM Super.	50.13 ±4.86	17.53 ±8.37	33.83 ±5.46	50.40 ±3.26	20.28 ±6.46	35.34 ±4.18	52.20 ±2.63	21.12 ±4.23	36.86 ±2.35	54.66 ±2.23	24.31 ±3.05	39.48 ±1.97	53.95 ±1.45	23.96 ±2.02	38.96 ±1.04	57.15	25.00	41.08
Literatura	Abordagem supervisionada proposta por Zhu, Kiritchenko e Mohammad (2014)																	58.16

Os experimentos envolveram seis bases de dados amplamente conhecidas da área de análise de sentimentos em *tweets* ([MOHAMMAD; KIRITCHENKO; ZHU, 2013](#); [LIU, 2012](#); [LIU, 2010](#); [SPERIOSU et al., 2011](#); [ZHANG et al., 2011](#)). O C³E-SL semissupervisionado foi comparado com uma abordagem similar de *self-training*, a qual usa o classificador SVM como componente supervisionado. Adicionalmente, máquinas de vetores de suporte operando de forma independente (sem *self-training*) foram também testadas. O C³E-SL semissupervisionado obteve os melhores resultados na maioria dos cenários investigados, apresentando-se como

Tabela 21 – Proporção de maiores F-scores e menores desvios padrões ao se comparar apenas os resultados da Tabela 20 para o C³E-SL semissupervisionado e sua contraparte, a qual emprega *self-training* com SVM. Os melhores resultados estão em negrito.

Base de Dados	Maiores F-scores		Menores Desvios-Padrões	
	SVM Semissuper.	C ³ E-SL Semissuper.	SVM Semissuper.	C ³ E-SL Semissuper.
<i>LiveJournal</i>	0%	100%	47%	53%
<i>SMS2013</i>	0%	100%	36%	64%
<i>Twitter2013</i>	42%	58%	86%	14%
<i>Twitter2014</i>	13%	87%	29%	71%
<i>TwitterSarcasm2014</i>	14%	86%	54%	46%

uma alternativa estável e promissora para a aplicação analisada. Estes resultados e discussões foram reportadas no artigo “*Using Unsupervised Information to Improve Semi-supervised Tweet Sentiment Classification*”, submetido para um periódico internacional especializado. Este artigo está atualmente em fase de revisão.

Em resumo, os artigos produzidos nesta pesquisa e que estudam aplicações do C³E-SL na classificação de *tweets* são:

- COLETTA, L. F. S.; SILVA, N. F. F.; HRUSCHKA, E. R.; HRUSCHKA JR., E. R., *Combining Classification and Clustering for Tweet Sentiment Analysis*. In: Brazilian Conference on Intelligent Systems (BRACIS), 2014, São Carlos. Proceedings on Brazilian Conference on Intelligent Systems, p. 210-215, 2014 — (COLETTA *et al.*, 2014).
- SILVA, N. F. F.; COLETTA, L. F. S.; HRUSCHKA, E. R.; HRUSCHKA JR., E. R., *Using Unsupervised Information to Improve Semi-supervised Tweet Sentiment Classification*, 2016 — submetido para um periódico internacional.
- SILVA, N. F. F.; COLETTA, L. F. S.; HRUSCHKA, E. R., *A Survey and Comparative Study of Tweet Sentiment Analysis via Semi-Supervised Learning*, 2016 — submetido para um periódico internacional.

Como tópico emergente, o uso de aprendizado semissupervisionado para a análise de sentimento em *tweets* motiva estudos em diferentes linhas. Para a abordagem proposta, a qual emprega o algoritmo C³E-SL, a escolha dos melhores atributos, tanto para o componente supervisionado quanto para o não supervisionado, requer maiores investigações. Por exemplo, na presença de sarcasmo e ironia, existem atributos que são reconhecidamente mais promissores (CARVALHO *et al.*, 2009; GONZALEZ-IBANEZ; MURESAN; WACHOLDER, 2011; VANIN *et al.*, 2013; FREITAS *et al.*, 2014). Neste sentido, um mecanismo mais dinâmico seria selecionar durante o aprendizado os atributos que são mais adequados, como um procedimento intrínseco ao processo iterativo desempenhado pelo C³E-SL semissupervisionado. Além disso, métodos semissupervisionados ainda dependem significativamente de uma amostragem *ideal* dos dados. Em outras palavras, em cenários com alto custo de rotulamento, deseja-se selecionar alguns

(poucos) objetos para rotular que sejam, de fato, significativos para o propósito de treinamento do modelo, a exemplo das contribuições de [Hajmohammadi et al. \(2015\)](#) e [Smailović et al. \(2014\)](#). Nestes trabalhos, os autores utilizam-se de aprendizado ativo — *active learning* ([SETTLES, 2012](#)) — para maximizar a representação dos dados em pequenas amostras de treinamento.

A ideia de *active learning* ([SETTLES, 2012](#)) é explorada mais profundamente no próximo capítulo desta tese, no qual é proposto um novo algoritmo para a detecção e aprendizagem de novas classes não observadas durante o treinamento do modelo. Para isso, foi estudado um mecanismo que aproveita a informação de grupos presentes nos dados para se selecionar amostras representativas em procedimentos de aprendizado ativo. Os experimentos envolveram a área de visão computacional, na qual a presença de classes desconhecidas é bem comum.

COMBINANDO CLASSIFICADORES E AGRUPADORES PARA APRENDER NOVAS CLASSES

A combinação de classificadores e agrupadores é útil para superar algumas das dificuldades encontradas na classificação de dados (COLETTA *et al.*, 2015b; ACHARYA *et al.*, 2014). Essencialmente, algoritmos de classificação tradicionais necessitam de um conjunto de treinamento representativo para produzir modelos que possam discriminar apropriadamente classes de objetos (BISHOP, 2007). Entretanto, conforme já discutido no capítulo anterior, o processo de obtenção de um conjunto de dados rotulados suficientemente bom para treinamento é uma tarefa muito custosa (TUIA *et al.*, 2011; GUILLAUMIN; VERBEEK; SCHMID, 2010; LAMPERT; NICKISCH; HARMELING, 2009; FOODY; MATHUR, 2004). Por conta disto, em aplicações do mundo real, apenas um conjunto limitado de objetos rotulados está disponível. No Capítulo 4, esta dificuldade foi abordada com base na análise semissupervisionada de sentimentos (LIU, 2012), na qual o algoritmo C³E-SL (Seção 2.1.2) mostrou-se apropriado. Particularmente, além de atuar bem com poucos dados rotulados, o C³E-SL torna a classificação mais “consciente” das possíveis diferenças entre as distribuições dos dados de treinamento e dos dados do conjunto alvo, sendo também capaz de lidar com o surgimento de novos *conceitos* (classes) (COLETTA *et al.*, 2015b; ACHARYA *et al.*, 2014). Esta última propriedade é o foco deste capítulo.

Algoritmos de classificação fundamentalmente assumem um único conjunto finito (e previamente conhecido) de classes. Como resultado, os modelos produzidos são incapazes de operar com novos *conceitos* (ou classes) que possam surgir (SCHEIRER *et al.*, 2013; TORRALBA; EFROS, 2011; HAND, 2006; DUIN; PEKALSKA, 2005). Em outras palavras, estes classificadores são altamente dependentes do conhecimento *a priori* e não conseguem se adaptar as mudanças nos dados (MASUD *et al.*, 2011; MASUD *et al.*, 2010). A Figura 27 apresenta um exemplo intuitivo, no qual um classificador de imagens *ideal* foi induzido por um conjunto de trei-

Figura 27 – Um classificador de imagens hipotético e ideal treinado para reconhecer apenas cavalos, pessoas e elefantes comete erros significativos na presença de uma nova classe, a qual representa dinossauros.

namento com instâncias de três classes visuais conhecidas: “*Cavalos*”, “*Pessoas*” e “*Elefantes*”. Ao inferir os rótulos de classes de imagens em um conjunto alvo balanceado que inclui objetos de uma nova classe (“*Dinossauros*”), este classificador não é capaz de atingir acurácia maior que 75%. Naturalmente, como o classificador não aprendeu a classe “*Dinossauros*” durante o seu treinamento, irá predizer dinossauros como sendo cavalos, pessoas ou elefantes, portanto cometendo erros significativos.

Classificadores capazes de detectar novas classes são úteis em uma grande variedade de problemas (MARKOU; SINGH, 2003a; MARKOU; SINGH, 2003b; HODGE; AUSTIN, 2004). Em especial, para problemas de detecção de intrusão, de falhas e de fraudes, pois nestes cenários podem ocorrer eventos não esperados que necessitam ser tratados de maneira inteligente (KHAN; AWAD; THURASINGHAM, 2007; CRUPI; GUGLIELMINO; MILAZZO, 2004; WANG *et al.*, 2003). Da mesma forma, em redes de sensores e de computadores, transações financeiras, pesquisas *Web* e reconhecimento de imagens podem surgir padrões não previstos que necessitam ser identificados e considerados pelo modelo (LAMPERT; NICKISCH; HARMELING, 2009; MADABHUSHI *et al.*, 2005). Em aplicações de fluxo contínuo de dados (*data streams*) algoritmos de agrupamento têm sido usados para a identificação de mudanças no conceito dos dados (*concept drift*) e, mais especificamente, na evolução de conceito (*concept evolution*), quando a natureza dos dados evolui ocasionando o surgimento de novas classes não previstas no conjunto de treinamento (MASUD *et al.*, 2013; MASUD *et al.*, 2011; KUNCHEVA, 2008).

Neste capítulo, um novo algoritmo para a detecção e aprendizagem de novas classes é proposto. Dado que algoritmos de agrupamento detectam similaridades entre objetos, é possível detectar novas classes a partir de objetos que se concentram em *regiões* muito diferentes daquelas

onde se encontram as classes conhecidas pelo modelo corrente. Portanto, neste trabalho presume-se que um objeto possa ser representante de uma nova classe quando este possui grande incerteza sobre o seu rótulo de classe, além de estar inserido em uma região de alta densidade de dados. Estas informações são manipuladas por uma extensão do algoritmo C^3E -SL, introduzida na Seção 5.1.1, a qual utiliza aprendizado ativo, ou *active learning* (SETTLES, 2012), para adaptar o modelo de classificação para reconhecer novas classes. Essa área de pesquisa aborda algoritmos de aprendizado de máquina que possam alcançar melhor desempenho com pouco treinamento através de uma escolha mais inteligente dos dados rotulados. O algoritmo proposto foi investigado na classificação de imagens, que é uma aplicação particularmente importante para tais algoritmos. Experimentos reportados a partir da Seção 5.1.3 mostram que o classificador resultante é capaz de aprender novas classes em poucas iterações, portanto poupando esforços de rotulamento.

5.1 Aprendizagem de Novas Classes de Imagens

Muita pesquisa tem sido dedicada ao desenvolvimento de técnicas de aprendizado de máquina para a classificação de imagens (SZELISKI, 2010; BISHOP, 2007), cujo objetivo é o reconhecimento de uma classe específica (tais como cavalos, carros, montanhas, etc.) no conteúdo visual de imagens. No entanto, apesar dos avanços, a diferença entre a visão computacional e a visão humana ainda é bem significativa (PRINCE; PRINCE, 2012). Técnicas computacionais são apenas abstrações grosseiras da natural capacidade biológica de aprender conceitos com o tempo. Depois do treinamento, a partir de um conjunto de dados visuais, os classificadores vigentes normalmente encontram dificuldade para prever novos dados que possam ter alguma mudança na distribuição (ou que representem algum conceito novo). Contrastando com esse cenário, a visão humana, auxiliada por uma vasta memória visual formada por experiências prévias e conhecimento contextual, pode instantaneamente reconhecer uma infinidade de classes de imagens de diferentes domínios (ULLMAN, 2000; BIEDERMAN, 1987).

Para melhorar o desempenho dos classificadores, alguns autores têm se preocupado em obter um conjunto de treinamento representativo (TUIA *et al.*, 2011; FOODY; MATHUR, 2004). Outros pesquisadores se aproveitam de informações suplementares encontradas em palavras-chaves e rótulos associados às imagens (LUO *et al.*, 2013; JING; ZHANG; NG, 2012; GUILLAUMIN; VERBEEK; SCHMID, 2010). Estas investigações enfatizam que limitações no processo de amostragem, bem como as pressuposições intrinsecamente assumidas pelos modelos, podem dificultar o aprendizado. Por exemplo, modelos discriminativos, como as máquinas de vetores de suporte — *Support Vector Machines* (SVMs) (ZHOU *et al.*, 2010; WANG; ZHOU, 2010; CHEN; NASRABADI; TRAN, 2013), possuem um grande poder de generalização e são muito úteis na classificação de imagens, mas estas (por si só) não são capazes de reconhecer novas classes que possam surgir. A partir desta perspectiva, Scheirer *et al.* (2013) têm usado o termo “*cenário aberto*” para designar um problema de classificação mais realístico, no qual há conhecimento incompleto do mundo durante o treinamento e classes desconhecidas podem surgir

com o tempo. Em contraste, os autores também delinearam uma situação mais restritiva que remete aos classificadores comuns, pois estes atuam em um “*cenário fechado*” (único e finito) de classes conhecidas. Problemas envolvendo classes desconhecidas foram investigados usando-se extensões de um classificador SVM, as quais utilizam um plano adicional para otimizar o risco do “*cenário aberto*”. O plano adicional ajuda a modelar os limites de decisão para as regiões de interesse (onde se localizam as instâncias da classe positiva), delimitando, assim, regiões onde se encontram as classes negativas e desconhecidas. Esta ideia foi estendida para acomodar classificadores não lineares em problemas multiclasse (SCHEIRER; JAIN; BOULT, 2014), além de ser usada em outros estudos (COSTA *et al.*, 2014; SCHERREIK; RIGLING, 2014; WILBER *et al.*, 2013).

O surgimento de novas classes tem também chamado a atenção de outros pesquisadores da área de visão computacional. Em (JUN; GHOSH, 2013) foi proposto um modelo semissupervisionado adaptativo que permite a detecção de padrões desconhecidos em imagens hiperespectrais. Ao focar em técnicas não supervisionadas, Lee e Grauman (2012) introduziram um algoritmo que captura interações entre objetos dentro das imagens de forma que coocorrências possam identificar novas categorias. Por outro lado, Bart e Ullman (2005) usaram a adaptação de atributos para produzir generalizações cruzadas capazes de aprender uma nova classe a partir de uma única instância. De maneira semelhante, em (LAMPERT; NICKISCH; HARMELING, 2014; LAMPERT; NICKISCH; HARMELING, 2009) foi empregado transferência de conhecimento para reconhecer novas classes de imagens. Inspirados por estes trabalhos, vários autores têm alcançado importantes resultados (ROHRBACH; STARK; SCHIELE, 2011; YU; GEN, 2010; PALATUCCI *et al.*, 2009).

Em uma perspectiva mais ampla, vários pesquisadores têm também investigado abordagens para aprender automaticamente conceitos dissimilares daqueles já conhecidos (CRUPI; GUGLIELMINO; MILAZZO, 2004; MARKOU; SINGH, 2003a; MARKOU; SINGH, 2003b; DIEHL; HAMPSHIRE J.B., 2002; MAŁDZIUK; SHASTRI, 2002). Em aplicações de fluxo contínuo de dados (*data streams*), agrupadores têm sido usados para identificar mudanças na distribuição dos dados (*concept-drift*) e o surgimento de novas classes (*concept-evolution*) (MASUD *et al.*, 2013; MASUD *et al.*, 2011; KUNCHEVA, 2008). Nesta tese, a informação de grupos também é usada para se detectar novas classes. Através de um processo iterativo de aprendizado ativo (*active learning* (SETTLES, 2012)), objetos mais prováveis de pertencerem a novas classes são incorporados no modelo. Este mecanismo opera em problemas multiclasse, mas se difere da abordagem introduzida por Scheirer, Jain e Boulton (2014), a qual identifica classes desconhecidas com o intuito de mantê-las separadas das classes de interesse. Mais detalhadamente, o algoritmo proposto na Seção 5.1.1 é capaz de detectar e aprender novas classes para predizê-las em dados futuros. Ao contrário dos trabalhos de Lampert, Nickisch e Harmeling (2014), Lampert, Nickisch e Harmeling (2009), nenhum “*link*” entre classes conhecidas e desconhecidas é necessário. Ou seja, o algoritmo proposto não utiliza transferência de conhecimento, mas é uma alternativa capaz de preencher algumas das lacunas destacadas na Seção 7.1 de Lampert, Nickisch e Harmeling

(2014) — em particular, a que se refere à necessidade de classificadores de imagens serem capazes de se adaptar para incorporar novas classes que possam aparecer.

5.1.1 Uma Versão Iterativa do C³E-SL Baseada em Active Learning

A extensão do algoritmo C³E-SL para detectar e aprender novas classes baseia-se em um processo iterativo, no qual o classificador pode se autoadaptar por meio de aprendizado ativo. Em cada iteração do algoritmo, uma busca especializada explora objetos com rótulos de classe incertos e que estão localizados em regiões de alta densidade no espaço de atributos. Estes objetos podem denotar alguma *novidade* desencadeada por mudanças na distribuição dos dados, ou serem representativos de novas classes não observadas no treinamento previamente realizado.

Preliminarmente, deseja-se encontrar objetos rotulados que tenham maior incerteza com relação ao seu rótulo de classe. Dada a predição de objetos de um conjunto alvo em particular, rótulos incertos podem ser capturados por meio da medida de entropia (ZHU, 1997; WEL; GAAG; GORTE, 1998). Especificamente, e tendo disponível os vetores c -dimensionais, $\{\mathbf{y}_i\}_{i=1}^n$, resultantes da otimização do C³E-SL, a *entropia de classificação* para um objeto i pode ser computada pela Equação (5.1):

$$e_i = \frac{-\sum_{j=1}^c y_{ij} \log_2 y_{ij}}{\log_2 c}. \quad (5.1)$$

Presume-se que novas classes possam ser detectadas ao se levar em conta a entropia de classificação dos objetos e a informação sobre a densidade em torno deles. Essencialmente, a densidade pode ser estimada a partir de matrizes de similaridades (como aquelas processadas pelo C³E-SL). Seja $\mathbf{H}^i$ o conjunto dos h -vizinhos mais próximos¹ de um dado objeto i , a densidade em torno de i pode ser computada a partir da Equação (5.2):

$$d_i = \frac{1}{h} \sum_{j \in \mathbf{H}^i} s_{ij}, \quad (5.2)$$

na qual os valores de s_{ij} correspondem às entradas de uma matriz de similaridades ($\mathbf{S}$).

Objetos candidatos a pertencerem a novas classes, além de terem alta entropia de classificação (com respeito as classes já conhecidas), devem se situar em regiões de alta densidade (de acordo com a informação de grupos nos dados). Portanto, um candidato promissor é um objeto i para o qual $(e_i \times d_i)$ é maximizado. Esta noção é capturada pelo Algoritmo 4 que realiza uma seleção de objetos baseada em entropia e densidade — *Entropy and Density-based Selection* (EDS). Mais detalhadamente, o Algoritmo 4 (EDS) recebe como entrada os vetores de entropia $\mathbf{e} = \{e_1, e_2, \dots, e_n\}$ e de densidade $\mathbf{d} = \{d_1, d_2, \dots, d_n\}$ e também um valor escalar, P_2 , que representa o número de candidatos a serem selecionados (portanto, as entropias e densidades

¹ Adotou-se $h = 5$.

Algoritmo 4: Seleção baseada em entropia e densidade (EDS)

Entrada: $\mathbf{e}, \mathbf{d}, P_2$.
Saída: $\Omega = \{\omega_j\}_{j=1}^{P_2}$.

```

1 Para  $j \leftarrow 1$  até  $P_2$  faça
2   Se  $j = 1$  então
3      $\omega_j = \arg \max_{1 \leq i \leq n} [e_i \cdot d_i];$ 
4   senão
5      $\omega_j = \arg \max_{1 \leq i \leq n} [e_i \cdot d_i \cdot (1 - \frac{1}{j-1} \sum_{r \in \Omega} s_{ir})];$ 
6    $e_{\omega_j} = -\infty$  (para selecionar instâncias distintas);
```

para os n objetos do conjunto alvo devem ser computadas de antemão). Note que, no Passo 5 do algoritmo, deseja-se selecionar P_2 objetos que sejam tão dissimilares quanto possível entre si (de tal forma que possam contribuir significativamente para a melhoria do modelo ao minimizar redundâncias). A saída do algoritmo — Ω — armazena os índices dos P_2 objetos no conjunto alvo ($\mathcal{X}$) que são mais propensos a serem representantes de uma nova classe.

A partir de um procedimento de aprendizado ativo (*active learning*), os P_2 objetos selecionados pelo Algoritmo EDS podem ser manualmente rotulados por um especialista de domínio de maneira que novas classes possam ser descobertas. Estes objetos são, então, incorporados no conjunto de treinamento e o modelo é retreinado. Ou seja, o algoritmo C³E-SL pode ser rodado novamente para se inferir a classe de objetos remanescentes em $\mathcal{X}$ — assumindo-se que o componente supervisionado foi reconstruído a partir do conjunto de treinamento atualizado² e a matriz de similaridades $\mathbf{S}$ foi reduzida pela eliminação das entradas correspondentes aos P_2 objetos selecionados. O Algoritmo 5 aplica estes passos de forma iterativa. Por simplicidade, o classificador resultante da combinação dos Algoritmos 5 (IC) e 4 (EDS) é chamado de IC-EDS.

5.1.2 Configurando o C³E-SL e seus Componentes

O classificador IC-EDS introduzido na seção anterior faz uso do algoritmo C³E-SL, o qual necessita da informação, por parte do usuário, de dois parâmetros (α e I). Uma alternativa mais prática é otimizar automaticamente estes parâmetros a partir dos próprios dados, conforme descrito na Seção 3.2 deste documento. Para a classificação de imagens, entretanto, usa-se um classificador SVM treinado com metade dos objetos rotulados disponíveis. Então, o algoritmo para evolução diferencial dinâmica (D²E) — apresentado na Seção 3.3 — estima os valores ótimos dos parâmetros (α^* e I^*) através da minimização do erro do C³E-SL em um conjunto de validação. Este conjunto contém a outra metade dos objetos rotulados, no qual, como requerido pelo algoritmo, um agregador de agrupadores foi induzido. Os valores estimados para α^* e I^* são fixados e usados para classificar objetos em um conjunto alvo, no qual um agregador de agrupadores similar ao construído no conjunto de validação é também necessário. Observe que,

² Alternativamente, classificadores incrementais podem ser usados para diminuir o tempo computacional.

Algoritmo 5: Classificador iterativo (IC) para aprendizagem de novas classes

Entrada: $\{\pi_i\}_{i=1}^n, \mathbf{S}_{n \times n}$.
Saída: $\{\mathbf{y}_i\}_{i=1}^q$.

- 1 Executar o C³E-SL a partir de $\{\pi_i\}_{i=1}^n$ e $\mathbf{S}_{n \times n}$ para prover $\{\mathbf{y}_i\}_{i=1}^n$ — *o agregador de classificadores foi construído a partir de P_1 instâncias rotuladas*;
- 2 $P_{tot} = 0$;
- 3 **Executar**
- 4 Obter os rótulos para as P_2 instâncias selecionadas pelo Algoritmo 4 — *e e d são obtidos via Equações (5.1) e (5.2), respectivamente*;
- 5 $P_{tot} = P_{tot} + P_2$;
- 6 $q = n - P_{tot}$;
- 7 Construir um agregador de classificadores a partir de $(P_{tot} + P_1)$ instâncias rotuladas para se obter $\{\pi'_i\}_{i=1}^q$;
- 8 Atualizar a matriz $\mathbf{S}'_{q \times q}$ (removendo as linhas e colunas relacionadas às P_{tot} instâncias rotuladas);
- 9 Executar o C³E-SL a partir de $\{\pi'_i\}_{i=1}^q$ e $\mathbf{S}'_{q \times q}$ para prover $\{\mathbf{y}_i\}_{i=1}^q$;
- 10 **até** um certo número de iterações ser alcançado;

para a classificação dos objetos no conjunto alvo, um novo classificador SVM deve ser gerado a partir de todos os objetos rotulados disponíveis.

Os parâmetros C e γ do SVM não linear usado pelo C³E-SL foram estimados por meio de busca em grade no conjunto de treinamento conforme indicado em (HSU *et al.*, 2010). Além disso, para obter os componentes do agregador de agrupadores, quatro conjuntos de agrupamentos (partições) foram criados. Cada conjunto levou em conta um subconjunto particular de atributos³. Adotou-se o procedimento de múltiplas rodadas ordenadas (CAMPELLO; HRUSCHKA; ALVES, 2009), no qual o K-Médias (JAIN, 2010) é executado 20 vezes para cada número de grupo $k = \{k_{min}, k_{min}+1, k_{min}+2, \dots, k_{max}\}$, $k_{min} = 2c$ e $k_{max} = \sqrt{n}$. Portanto, cada conjunto de agrupamentos se constituiu de várias partições de dados geradas a partir de diferentes inicializações e com diferentes números de grupos em um espaço particular de atributos. Para se construir a matriz de similaridades, cada conjunto contribuiu com a sua melhor partição de acordo com o critério da Silhueta (CAMPELLO; HRUSCHKA, 2006; HRUSCHKA; CAMPELLO; CASTRO, 2006).

5.1.3 Exemplo Pedagógico

Esta seção ilustra o funcionamento do algoritmo IC-EDS operando na presença de uma nova classe (desconhecida). Adotou-se uma base de dados pequena contendo apenas três classes visuais rotuladas como “Duck”, “Cat” e “Piggy”, as quais foram obtidas a partir da base de dados Coil-20⁴. A Figura 28 apresenta instâncias destas classes. Note que os conjuntos (iniciais) para treinamento foram formados apenas por instâncias das classes “Duck” e “Cat” (14 objetos

³ Subconjuntos de atributos que representam 20% do conjunto original de atributos foram amostrados aleatoriamente (sem reposição).

⁴ <<http://www.cs.columbia.edu/CAVE/software/softlib/coil-20.php>>

de cada classe). Ou seja, o algoritmo IC-EDS teve (inicialmente) a capacidade de reconhecer apenas patos e gatos, mas o algoritmo também deveria ser capaz de detectar e aprender a classe intencionalmente omitida “*Piggy*” ao classificar imagens em um conjunto alvo. A classe “*Piggy*” é, portanto, uma nova classe (desconhecida), a qual espera-se que o algoritmo possa reconhecer. Foram utilizados conjuntos alvo contendo 57 imagens não rotuladas de cada uma das classes disponíveis: “*Duck*”, “*Cat*” e “*Piggy*”.

Figura 28 – Imagens da base de dados pedagógica usada. Apenas exemplos de patos e gatos estavam disponíveis na base de treinamento para construir o classificador. O IC-EDS deveria, então, ser capaz de descobrir a nova classe “*Piggy*” presente inicialmente apenas nos conjuntos alvo.

A Figura 29 mostra, para cada iteração do algoritmo IC-EDS, as proporções (por classe) das imagens selecionadas para rotulamento de acordo com o Algoritmo 4 — estes resultados referem-se às médias de cinco execuções do algoritmo IC-EDS (cada execução considerou um conjunto alvo distinto). Na primeira iteração do algoritmo, 56% das imagens selecionadas pertenciam à classe deixada de fora do conjunto de treinamento (“*Piggy*”). Na segunda iteração, esta proporção alcançou 84%. Como exemplo, uma das imagens selecionadas da classe “*Piggy*” apresentou densidade igual a 0.95 e incerteza igual a 0.96 — estes valores foram computados a partir das classificações refinadas pelo C³E-SL: $p(C=Duck|x_{33}) = 0.45$, $p(C=Cat|x_{33}) = 0.21$ e $p(C=Piggy|x_{33}) = 0.34$. Portanto, o algoritmo IC-EDS detectou a classe desconhecida, “*Piggy*”, já nas primeiras iterações. A partir da terceira iteração, as proporções para cada classe tornaram-se mais similares. Mais detalhadamente, quando padrões visuais de “porquinhos” eram totalmente desconhecidos, o IC-EDS foi capaz de identificar este padrão particular por ser (significativamente) dissimilar daqueles já conhecidos. A seguir, após reconhecer a classe “*Piggy*” em seu modelo, não haviam novidades expressivas para serem capturadas. Convém destacar que

Figura 29 – Proporções (por classe) das imagens selecionadas nas três primeiras iterações do algoritmo IC-EDS.

poucas imagens são frequentemente suficientes para descrever cada uma das classes desta base de dados.

Computando-se as distâncias Euclidianas entre os *centroides* das instâncias selecionadas pelo IC-EDS e as instâncias com os rótulos de classe reais, notou-se que as instâncias selecionadas foram mais similares às classes “Cat” e “Piggy”. Este fato é observado na Figura 29, a qual indica que poucas instâncias da classe “Duck” foram selecionadas pelo algoritmo. Contudo, instâncias das classes “Duck” e “Piggy” são as mais similares entre si. Mesmo assim, o IC-EDS selecionou adequadamente imagens de “porquinhos” na maioria das vezes. A Figura 30 mostra uma das matrizes de similaridades construída para o problema. As classes “Duck”, “Cat” e “Piggy” foram (quase que na totalidade) corretamente separadas pelo agregador de agrupadores, o qual auxiliou nas classificações realizadas pelo C³E-SL e também na detecção de objetos de uma nova classe (a partir do Algoritmo 4).

A Tabela 22 mostra as matrizes de confusão produzidas depois de cada iteração do algoritmo IC-EDS (exceto a matriz mais à esquerda, que foi construída a partir de resultados do C³E-SL em conjuntos de treinamento iniciais). O IC-EDS classificou corretamente todas as imagens dos conjuntos alvo em apenas 3 iterações. Em particular, a classe “Piggy”, desconhecida *a priori*, foi detectada adequadamente e aprendida pelo modelo de classificação, o qual foi capaz de classificar corretamente todas as suas instâncias. Estes resultados oferecem indícios de que o algoritmo para detecção e aprendizagem de novas classes funciona bem, além de ser aplicável na prática.

Figura 30 – A matriz de similaridades gerada pelos agrupamentos identificou três grupos (regiões mais escuras) e ajudou no refinamento das classificações do SVM, bem como na detecção da nova classe (“Piggy”). Os grupos a partir da esquerda inferior até a direita superior englobam as instâncias, respectivamente, das classes “Duck”, “Cat” e “Piggy”.

		Predições											
		C ³ E-SL			Iteração 1			Iteração 2			Iteração 3		
		<i>Duck</i>	<i>Cat</i>	<i>Piggy</i>	<i>Duck</i>	<i>Cat</i>	<i>Piggy</i>	<i>Duck</i>	<i>Cat</i>	<i>Piggy</i>	<i>Duck</i>	<i>Cat</i>	<i>Piggy</i>
		<i>Duck</i>	<i>Cat</i>	<i>Piggy</i>	<i>Duck</i>	<i>Cat</i>	<i>Piggy</i>	<i>Duck</i>	<i>Cat</i>	<i>Piggy</i>	<i>Duck</i>	<i>Cat</i>	<i>Piggy</i>
Classes Reais	<i>Duck</i>	57	0	0	56	0	0	56	0	0	56	0	0
	<i>Cat</i>	2	56	0	2	54	0	1	55	0	0	53	0
	<i>Piggy</i>	52	5	0	30	0	25	3	0	48	0	0	49

Tabela 22 – Matrizes de confusão produzidas com resultados do C³E-SL em conjuntos de treinamento iniciais (contendo, portanto, apenas as classes “*Duck*” e “*Cat*”) e com resultados das três primeiras iterações do IC-EDS — em cada iteração, 5 imagens selecionadas foram manualmente rotuladas e incorporadas no modelo de classificação.

5.2 Avaliação Empírica

Nesta seção, o algoritmo IC-EDS, introduzido na Seção 5.1.1, é comparado com um algoritmo que substitui a sua busca especializada (Algoritmo 4) por uma seleção aleatória de instâncias. Este algoritmo, usado como *baseline* para comparação, é chamado de IC-RS (IC-*Random Selection*) e simplesmente seleciona de maneira aleatória instâncias para serem rotuladas no Passo 4 do Algoritmo 5. Em teoria, por executar uma busca mais inteligente por novos conceitos/classes, o algoritmo IC-EDS deve ser superior ao IC-RS. A comparação destes algoritmos é relevante para a validação da seleção de instâncias proposta.

Os experimentos foram baseados no procedimento de *holdout* (WITTEN; FRANK, 2005), no qual as bases de dados são randomicamente divididas em conjuntos de treinamento e conjuntos alvo (de teste). Os dados dos conjuntos de teste são independentes, i.e., não foram usados na construção nem mesmo na otimização dos classificadores. Mais precisamente, um certo percentual de imagens rotuladas (20% da base de dados) foi usado para treinar e validar os classificadores e o restante (conjunto alvo) foi usado para testá-los. Para simular o aparecimento de novas classes, instâncias de uma classe em particular foram deixadas de fora do conjunto de treinamento inicial, enquanto que os conjuntos de teste sempre contiveram instâncias representantes de todas as classes do problema. Para aumentar a confiança nos resultados, os algoritmos foram rodados cinco vezes por classe deixada de fora, levando-se em conta conjuntos de treinamento distintos obtidos por amostragem aleatória estratificada (sem reposição).

Três bases de dados bem populares da área de classificação de imagens foram usadas:

- A base de dados *Coil-20* consiste de 1440 imagens em escalas de cinza de tamanho 128×128 com um fundo preto para 20 diferentes objetos. Cada objeto foi fotografado em 72 diferentes ângulos de visão (NENE *et al.*, 1996);
- A base de dados *Corel-1000*⁵ contém 1000 fotografias coloridas divididas em 10 categorias

balanceadas, as quais envolvem povos africanos, praias, construções, ônibus, elefantes, flores, cavalos, montanhas e comidas (WANG; LI; WIEDERHOLD, 2001);

- A base de dados *Supermarket Produce* compreende 2633 imagens de frutas/vegetais divididas em 15 diferentes categorias (o número de instâncias disponíveis por categoria aparecem entre parênteses): batata agata (201), batata asterix (182), cajú (210), pêssego diamante (211), maçã fuji (212), maçã verde (155), melão (145), kiwi (171), nectarina (247), cebola (75), laranja (103), ameixa (264), pêra (159), limão taiti (106) e melância (192). As instâncias de cada classe variam de acordo com a luminosidade, posição e a presença de recortes, oclusões e sombras (ROCHA *et al.*, 2010).

A base de dados mais comportada, *Coil-20*, foi usada primeiro e, em seguida, a base de dados *Corel-1000*, a qual possui imagens naturais e com certa sobreposição de classes. Por fim, utilizou-se a base de dados *Supermarket Produce*, que contém classes desbalanceadas e certa confusão entre classes específicas. A Figura 31 oferece exemplos para todas as classes destas bases de dados, as quais podem ser descritas pela cor global e/ou atributos visuais de textura. Para simplificar a análise, e também por este estudo não ser focado na extração de atributos (*features*), escolheu-se o uso de descritores globais, que são rápidos para computar e podem ser efetivos em muitos cenários (PENATTI; VALLE; TORRES, 2012). Embora o uso dessas características pode acarretar em maior confusão entre certas classes que precisariam de métodos mais complexos, gerando limites inferiores para os erros de classificação, sua vantagem reside principalmente em não inserir parâmetros adicionais para a classificação de imagens. Primeiramente, as imagens foram preprocessadas usando-se um método de quantização, o qual produziu imagens de 8-bits com menos níveis de cores (PONTI; PICON, 2015). Então, os seguintes extratores foram usados:

- **Border Interior Classification (BIC)**: para a extração de cores que possam codificar informação estrutural, i.e., a distribuição espacial das cores existentes na imagem. A representação gerada envolve a distribuição de cores da imagem a partir de dois histogramas: um para *pixels* de borda e outro para *pixels* de interior. Um *pixel* é classificado como “de borda” se ao mínimo um de seus vizinhos tem uma cor diferente, caso contrário tal *pixel* é classificado como “de interior” (STEHLING; NASCIMENTO; FALCAO, 2002). O tamanho do vetor final tem dimensões $C \times 2$, no qual C é o número de níveis de cores na imagem;
- **Haralick-6**: baseando-se em matrizes de coocorrência de níveis de cinza. Primeiro, calcula-se uma matriz de coocorrência de tamanho $C \times C$ a partir de uma relação fixa de $\Delta(x,y) = (1,0)$ entre pares de *pixels*. Então, são extraídos 6 atributos desta matriz: entropia, homogeneidade, contraste, correlação, probabilidade máxima e uniformidade (HARALICK; SHANMUGAN; DINSTEN, 1973).

O conjunto de atributos para cada base de dados foi computado da seguinte forma:

Coil-20

Corel-1000

Supermarket Produce

Figura 31 – Instâncias de cada classe das três bases de dados usadas nos experimentos.

- *Coil-20*: as imagens foram quantizadas em 8 níveis de cores. As regiões correspondentes ao fundo foram ignoradas, resultando em 15 atributos BIC. A partir do extrator *Haralick-6* outros 6 atributos foram computados e, então, as imagens foram representadas por um conjunto final de 21 atributos;
- *Corel-1000* e *Supermarket Produce*: as imagens foram quantizadas em 64 níveis de cores e somente os atributos BIC foram extraídos, resultando em um espaço de atributos de 128 dimensões.

5.2.1 Base de Dados Coil-20

A Figura 32 mostra as proporções (por classe) das imagens selecionadas depois de 10 iterações dos algoritmos IC-EDS e IC-RS quando instâncias da classe 13 (“Piggy”) foram

Figura 32 – Proporções, por classe, de instâncias selecionadas após 10 iterações quando a classe 13 (“Piggy”) foi intencionalmente omitida com o intuito de ser descoberta pelos algoritmos.

deixadas de fora dos conjuntos de treinamento. Diferentemente do experimento ilustrativo apresentado na Seção 5.1.3, este problema é mais desafiador, exigindo que os algoritmos sejam capazes de distinguir uma nova classe, “Piggy”, em um conjunto de vinte classes, das quais dezenove são conhecidas pelos modelos de classificação. Como esperado, o algoritmo IC-RS selecionou uma proporção mais balanceada de instâncias de cada classe (variando-se entre 3.2% e 6.8%), enquanto que o algoritmo IC-EDS selecionou de maneira mais precisa instâncias da classe desconhecida. Especificamente, instâncias das classes 2, 3, 10 e 13 foram selecionadas com maior frequência pelo IC-EDS. Em particular, 8.4% das instâncias escolhidas pelo algoritmo pertenciam à classe omitida “Piggy”. Como resultado, usando-se o IC-EDS, a taxa de erro da classe “Piggy” diminuiu para 0.13, enquanto que a taxa de erro final ao usar o algoritmo IC-RS foi de 0.74. A Figura 33 mostra as taxas de erro para ambos os algoritmos de acordo com o número de instâncias selecionadas (5 imagens foram selecionadas e manualmente rotuladas em cada iteração).

Quando as instâncias de classe 2 (“Wooden Part I”) foram deixadas de fora dos

Figura 33 – Erro da classe 13, “Piggy”.

Figura 34 – Erro da classe 2, “Wooden Part I”.

conjuntos de treinamento, o algoritmo IC-EDS também detectou a classe apropriadamente. Depois de 10 iterações, 11.6% das instâncias selecionadas pelo IC-EDS eram representantes da classe 2, enquanto que, para a mesma classe, este percentual foi de 6.0% ao usar o IC-RS. No entanto, como mostra a Figura 34, os algoritmos desempenharam um comportamento similar. Em particular, as taxas de erro foram de aproximadamente 0.75 após 50 imagens terem sido selecionadas e rotuladas. Portanto, embora o IC-EDS tenha se apresentado ligeiramente melhor que o seu algoritmo de base, ambos geraram resultados finais ruins neste caso.

Levando em conta a omissão da classe 4 (“Cat”), 22.4% das instâncias selecionadas pelo IC-EDS foram desta classe. Como resultado, depois de 10 iterações, a taxa de erro diminuiu para 0.04. Em contraste, usando-se o algoritmo IC-RS, somente 4% das imagens selecionadas foram da classe “Cat” e sua taxa de erro foi reduzida, então, para apenas 0.68. A Figura 35 mostra as curvas de erro para ambos os algoritmos — note que o IC-EDS reduziu o erro da classe para menos de 0.1 ao rotular apenas 20 instâncias (i.e., depois de 4 iterações do algoritmo).

Figura 35 – Erro da classe 4, “Cat”.

Economizar esforços de rotulamento é uma propriedade bastante apreciada para classificadores de imagens. Sob esta perspectiva, as análises realizadas também focaram no número de instâncias que devem ser rotuladas por cada algoritmo para se alcançar taxas específicas de erro. As Figuras 36 e 37 mostram o número requerido de imagens rotuladas para produzir uma certa taxa de erro para determinadas classes omitidas — “Wooden Part I” e “Cat”, respectivamente. Observe que o IC-EDS necessitou de menos imagens rotuladas para obter o mesmo (ou menor) erro alcançado pelo IC-RS. Portanto, o algoritmo IC-EDS claramente mostra-se como uma alternativa vantajosa neste caso.

A Figura 38 ilustra o F-score global computado a partir da média de vinte F-scores, cada um correspondente a uma classe deixada de fora. Como esperado, o algoritmo IC-EDS produziu, em geral, melhores resultados que o algoritmo IC-RS. A Tabela 23 reporta o percentual de imagens selecionadas, F-scores e taxas de erros para cada classe omitida após 5 iterações dos algoritmos (i.e., quando 25 instâncias foram manualmente rotuladas). Nota-se, portanto, que o IC-EDS detectou e aprendeu com sucesso a maioria das classes. Mais detalhadamente, em

Figura 36 – Número de instâncias rotuladas para alcançar uma certa taxa de erro para a classe omitida “Wooden Part I”.

Figura 37 – Número de instâncias rotuladas para alcançar uma certa taxa de erro para a classe omitida “Cat”.

Figura 38 – F-scores globais obtidos por cada algoritmo na base de dados *Coil-20*.

90% dos casos, o IC-EDS selecionou mais instâncias da classe deixada de fora que o algoritmo IC-RS, além de ter provido as menores taxas de erro em 75% das omissões de classes.

5.2.2 Base de Dados Corel-1000

Com o uso da base de dados *Corel-1000*, notou-se que a seleção correta de instâncias da classe nova não implica necessariamente na redução de seu erro de classificação. A Figura 39 apresenta as proporções de imagens selecionadas quando a classe 10 (“Food”) foi deixada de fora. De acordo com esta figura, bem como as taxas de erro na Figura 40, apesar do algoritmo IC-EDS ter sido capaz de detectar mais imagens da classe “Food” que o algoritmo IC-RS, este último apresentou resultados ligeiramente melhores a partir da sétima iteração (i.e., depois de 30 imagens serem rotuladas). Para clarificar esta discussão, dois pontos devem ser levantados: (i) as imagens selecionadas pelo EDS (Algoritmo 4) que são provenientes de classes já conhecidas podem, de certa forma, atrapalhar o aprendizado de novas classes desconhecidas. Como mostrado na Figura 39, as imagens de classes 1 (“Africans”), 3 (“Buildings”) e 4 (“Buses”) foram mais frequentes que aquelas da classe omitida 10. Estas classes podem apresentar as mesmas características da

Tabela 23 – Imagens selecionadas (%), F-scores e taxas de erro para cada classe deixada de fora da base de dados *Coil-20*. Os algoritmos executaram 5 iterações (25 imagens foram selecionadas e manualmente rotuladas). Desvios padrões estão entre parênteses e os melhores resultados estão destacados em negrito.

Classe Omitida	Seleção (%)		F-score		Taxa de Erro	
	IC-RS	IC-EDS	IC-RS	IC-EDS	IC-RS	IC-EDS
1 (“Duck”)	4.0 (4.90)	13.6 (11.17)	0.08 (0.11)	0.71 (0.23)	0.95 (0.06)	0.41 (0.29)
2 (“Wooden Part 1”)	6.4 (6.07)	14.4 (8.76)	0.23 (0.14)	0.26 (0.19)	0.86 (0.08)	0.84 (0.13)
3 (“Car 1”)	4.8 (3.35)	8.0 (5.66)	0.34 (0.21)	0.05 (0.09)	0.77 (0.14)	0.97 (0.05)
4 (“Cat”)	2.4 (3.58)	38.4 (21.09)	0.17 (0.31)	0.93 (0.06)	0.87 (0.24)	0.08 (0.12)
5 (“Anacin”)	3.2 (4.38)	4.0 (2.83)	0.12 (0.15)	0.20 (0.16)	0.93 (0.09)	0.88 (0.10)
6 (“Car 2”)	4.8 (3.35)	2.4 (3.58)	0.49 (0.28)	0.06 (0.08)	0.62 (0.23)	0.97 (0.05)
7 (“Wooden Part 2”)	7.2 (3.35)	16.0 (16.73)	0.24 (0.18)	0.74 (0.15)	0.85 (0.12)	0.39 (0.22)
8 (“Talc”)	4.0 (2.83)	24.8 (25.52)	0.19 (0.30)	0.52 (0.24)	0.85 (0.25)	0.61 (0.23)
9 (“Tylenol”)	2.4 (2.19)	19.2 (8.67)	0.21 (0.22)	0.56 (0.38)	0.87 (0.14)	0.54 (0.34)
10 (“Vaseline”)	5.6 (2.19)	31.2 (18.63)	0.20 (0.19)	0.86 (0.11)	0.87 (0.12)	0.23 (0.17)
11 (“Wooden Part 3”)	4.8 (7.16)	10.4 (8.29)	0.22 (0.31)	0.49 (0.33)	0.84 (0.22)	0.62 (0.30)
12 (“Cup 1”)	5.6 (4.56)	22.4 (20.12)	0.63 (0.38)	0.95 (0.08)	0.45 (0.40)	0.09 (0.13)
13 (“Piggy”)	2.4 (2.19)	9.6 (9.21)	0.32 (0.30)	0.32 (0.25)	0.78 (0.21)	0.79 (0.18)
14 (“Pot 1”)	4.8 (3.35)	18.4 (15.39)	0.19 (0.28)	0.83 (0.13)	0.87 (0.20)	0.27 (0.20)
15 (“Half Covered Pot”)	3.2 (3.35)	11.2 (14.25)	0.49 (0.48)	1.00 (0.00)	0.55 (0.47)	0.00 (0.00)
16 (“Pot 2”)	4.8 (3.35)	11.2 (20.67)	0.77 (0.44)	0.98 (0.03)	0.21 (0.44)	0.01 (0.01)
17 (“Uncovered Pot”)	2.4 (3.58)	12.0 (12.33)	0.29 (0.43)	0.44 (0.40)	0.73 (0.43)	0.65 (0.32)
18 (“Cup 2”)	4.0 (2.83)	12.0 (13.27)	0.77 (0.44)	0.73 (0.43)	0.25 (0.43)	0.30 (0.45)
19 (“Car 3”)	3.2 (1.79)	11.2 (13.08)	0.13 (0.22)	0.36 (0.32)	0.91 (0.14)	0.74 (0.25)
20 (“Cream Cheese”)	8.0 (2.83)	1.6 (3.58)	0.60 (0.55)	0.20 (0.45)	0.40 (0.55)	0.80 (0.45)

classe 10, o que pode levar a dificuldades de classificação. Em particular, erros de classificação podem ocorrer quando a nova classe (a ser aprendida) se sobrepõe com as classes já conhecidas; (ii) os desempenhos dos algoritmos avaliados tendem a ser mais similares com o aumento do número de iterações. Como enfatizado na Seção 5.1.3, o IC-EDS tem se mostrado melhor em encontrar imagens que representam uma nova classe já nas suas primeiras iterações. A seguir, as classes de imagens selecionadas tornam-se mais equiprováveis (como é feito pelo IC-RS). A Figura 40 mostra que o IC-EDS, de fato, produz uma diminuição acentuada do erro com poucas iterações (i.e., com um total de 25 imagens rotuladas). Este comportamento é mais evidente na

Figura 39 – Proporções, por classe, de instâncias selecionadas após 10 iterações quando a classe 10 (“Food”) foi intencionalmente omitida com o intuito de ser descoberta pelos algoritmos.

Figura 40 – Erro da classe 10, “Food”.

Figura 41 – Erro da classe 3, “Buildings”.

Figura 41, a qual ilustra curvas de erro quando a classe 3 (“Buildings”) foi deixada de fora dos conjuntos de treinamento.

Ao focar nos custos de rotulamento, as Figuras 42 e 43 indicam que o algoritmo IC-EDS mostrou-se mais eficiente que o IC-RS. Em particular, estas figuras mostram o número de imagens rotuladas necessárias para se alcançar uma certa taxa de erro quando as classes “Food” e “Buildings” foram omitidas. Para esta última classe, por exemplo, o algoritmo IC-EDS obteve um erro menor que 0.6 com apenas 25 instâncias rotuladas, enquanto que, para desempenho similar, o IC-RS necessitou do rotulamento de 40 instâncias.

A Figura 44 mostra os F-scores globais produzidos pela média de F-scores independentes computados para cada classe omitida. Embora os resultados indiquem que a base de dados *Corel-1000* seja desafiadora no contexto examinado, resultados interessantes foram produzidos, em particular com poucas imagens rotuladas. A Tabela 24 sumariza o percentual de imagens selecionadas, F-scores e taxas de erro para cada classe deixada de fora quando 25 imagens foram manualmente rotuladas (i.e., depois de 5 iterações dos algoritmos). Os melhores resultados do

Figura 42 – Número de instâncias rotuladas para alcançar uma certa taxa de erro para a classe omitida “Food”.

Figura 43 – Número de instâncias rotuladas para alcançar uma certa taxa de erro para a classe omitida “Buildings”.

algoritmo IC-EDS foram obtidos quando as classes 1 (“Africans”), 2 (“Beach”), 3 (“Buildings”), 5 (“Dinosaurs”) e 10 (“Food”) foram deliberadamente omitidas. Em resumo, o IC-EDS proveu menores taxas de erros que o IC-RS em 50% das classes omitidas, e em 70% dos casos, selecionou mais imagens que o IC-RS.

Figura 44 – F-scores globais obtidos por cada algoritmo na base de dados *Corel-1000*.

Tabela 24 – Imagens selecionadas (%), F-scores e taxas de erro para cada classe deixada de fora da base de dados *Corel-1000*. Os algoritmos executaram 5 iterações (25 imagens foram selecionadas e manualmente rotuladas). Desvios padrões estão entre parênteses e os melhores resultados estão destacados em negrito.

Classe Omitida	Seleção (%)		F-score		Taxa de Erro	
	IC-RS	IC-EDS	IC-RS	IC-EDS	IC-RS	IC-EDS
1 (“Africans”)	11.2 (4.38)	36.0 (11.66)	0.27 (0.23)	0.40 (0.16)	0.81 (0.18)	0.64 (0.11)
2 (“Beach”)	9.6 (6.07)	17.6 (7.80)	0.59 (0.22)	0.60 (0.18)	0.49 (0.31)	0.41 (0.23)
3 (“Buildings”)	16.0 (6.32)	28.8 (14.53)	0.37 (0.26)	0.62 (0.13)	0.73 (0.21)	0.51 (0.14)
4 (“Buses”)	12.8 (6.57)	14.4 (12.52)	0.14 (0.21)	0.05 (0.05)	0.88 (0.19)	0.97 (0.02)
5 (“Dinosaurs”)	13.6 (4.56)	17.6 (6.69)	0.77 (0.43)	0.94 (0.04)	0.21 (0.44)	0.04 (0.05)
6 (“Elephants”)	8.0 (8.00)	8.0 (8.00)	0.22 (0.16)	0.06 (0.07)	0.86 (0.11)	0.97 (0.03)
7 (“Flowers”)	8.8 (6.57)	28.0 (4.00)	0.64 (0.09)	0.35 (0.31)	0.50 (0.10)	0.75 (0.23)
8 (“Horses”)	8.8 (3.35)	3.2 (1.79)	0.14 (0.15)	0.01 (0.03)	0.92 (0.09)	0.99 (0.01)
9 (“Mountains”)	10.4 (5.37)	4.8 (7.16)	0.12 (0.12)	0.03 (0.05)	0.93 (0.07)	0.98 (0.03)
10 (“Food”)	7.2 (1.79)	9.6 (4.56)	0.13 (0.18)	0.15 (0.21)	0.91 (0.12)	0.86 (0.13)

5.2.3 Base de Dados *Supermarket Produce*

Na base de dados *Supermarket Produce*, o algoritmo IC-EDS produziu os melhores resultados na maioria das classes omitidas. Por exemplo, a Figura 45 ilustra as proporções de imagens selecionadas quando a classe 9 (“Nectarine”) foi deixada de fora. Enquanto o algoritmo IC-RS manteve proporções entre 3.2% e 12%, o IC-EDS encontrou “nectarinas” em 31.2% das imagens selecionadas. Como resultado, o IC-EDS realizou uma rápida redução do erro da classe — veja a Figura 46, a qual ilustra as curvas de erro para cada algoritmo. Comportamento similar foi observado quando as classes 4, 5, 6, 7, 12, 13 e 15 foram deixadas de fora dos conjuntos de treinamento. O desempenho satisfatório do IC-EDS também é visível na Figura 47, a qual exhibe

Figura 45 – Proporções, por classe, de instâncias selecionadas após 10 iterações quando a classe 9 (“*Nectarine*”) foi intencionalmente omitida com o intuito de ser descoberta pelos algoritmos.

o F-score global computado a partir de quinze F-scores independentes, cada qual correspondente a uma classe que foi omitida.

As Figuras 48 e 49 ilustram o esforço de rotulamento para se alcançar um certo erro quando as classes “*Nectarine*” e “*Watermelon*” foram omitidas. Para a classe “*Nectarine*”, rotulando-se 25 imagens (i.e., depois de 5 iterações do algoritmo), o IC-EDS obteve uma taxa de erro inferior a 0.7, enquanto que o IC-RS requereu 45 instâncias rotuladas para produzir resultado semelhante. Para a classe “*Watermelon*”, rotulando-se 10 instâncias, o IC-EDS alcançou uma taxa de erro menor que 0.2, enquanto que o IC-RS necessitou de 50 instâncias rotuladas para produzir um desempenho similar.

Os resultados gerais estão sumarizados na Tabela 25, a qual reporta as proporções de imagens selecionadas, F-scores e taxas de erro para cada algoritmo quando uma classe, em

Figura 46 – Erro da classe 9, “*Nectarine*”.

Figura 47 – F-scores globais para a base de dados *Supermarket Produce*.

Figura 48 – Número de instâncias rotuladas para alcançar uma certa taxa de erro para a classe omitida “Nectarine”.

Figura 49 – Número de instâncias rotuladas para alcançar uma certa taxa de erro para a classe omitida “Watermelon”.

particular, foi deixada de fora dos conjuntos de treinamento iniciais. Como esperado, o algoritmo IC-EDS descobriu a classe omitida de forma mais precisa que o IC-RS, sendo que em 87% dos casos, o IC-EDS selecionou mais instâncias que o IC-RS. Como consequência, o IC-EDS produziu menores taxas de erro em 73% das omissões de classes.

5.3 Considerações Finais

Este capítulo introduziu um classificador flexível, denominado IC-EDS, capaz de lidar com o surgimento de novas classes de imagens ao longo do tempo. Seu mecanismo, que tem como base o algoritmo C³E-SL, usa a informação de grupos para detectar novas classes, as quais são aprendidas via *active learning*. O modelo de classificação de imagens resultante pode ser visto como um passo significativo na direção de classificadores que sejam “conscientes” de novos conceitos/classes que possam aparecer com o tempo.

Os resultados experimentais indicaram que o IC-EDS é capaz de detectar e aprender novas classes em poucas iterações. Além disso, este algoritmo tem se mostrado vantajoso na prática, visto que realiza uma seleção inteligente e parcimoniosa de instâncias para rotulamento. Estas

Tabela 25 – Imagens selecionadas (%), F-scores e taxas de erros para cada classe deixada de fora da base de dados *Supermarket Produce*. Os algoritmos executaram 5 iterações (25 imagens foram selecionadas e manualmente rotuladas). Desvios padrões estão entre parênteses e os melhores resultados estão destacados em negrito.

Classe Omitida	Seleção (%)		F-score		Taxa de Erro	
	IC-RS	IC-EDS	IC-RS	IC-EDS	IC-RS	IC-EDS
1 (“Agata Potato”)	10.4 (4.56)	17.6 (7.80)	0.42 (0.15)	0.36 (0.21)	0.72 (0.14)	0.76 (0.15)
2 (“Asterix Potato”)	8.0 (6.32)	20.0 (14.14)	0.43 (0.39)	0.54 (0.02)	0.64 (0.39)	0.63 (0.02)
3 (“Cashew”)	8.8 (3.35)	6.4 (4.56)	0.76 (0.12)	0.37 (0.39)	0.36 (0.16)	0.70 (0.33)
4 (“Diamond Peach”)	4.0 (5.66)	13.6 (9.21)	0.19 (0.30)	0.37 (0.35)	0.86 (0.24)	0.72 (0.30)
5 (“Fuji Apple”)	5.6 (6.07)	21.6 (9.21)	0.29 (0.20)	0.56 (0.20)	0.82 (0.13)	0.58 (0.19)
6 (“Granny-Smith Apple”)	4.0 (4.90)	20.0 (12.00)	0.27 (0.37)	0.45 (0.29)	0.79 (0.29)	0.65 (0.30)
7 (“Honeydew Melon”)	1.6 (2.19)	12.0 (6.32)	0.21 (0.22)	0.31 (0.34)	0.87 (0.15)	0.77 (0.26)
8 (“Kiwi”)	6.4 (8.29)	9.6 (4.56)	0.23 (0.24)	0.15 (0.18)	0.84 (0.19)	0.90 (0.12)
9 (“Nectarine”)	10.4 (6.07)	37.6 (13.45)	0.13 (0.15)	0.50 (0.19)	0.92 (0.10)	0.64 (0.17)
10 (“Onion”)	3.2 (5.22)	2.4 (3.58)	0.01 (0.03)	0.11 (0.26)	0.99 (0.02)	0.92 (0.18)
11 (“Orange”)	4.0 (4.00)	12.0 (4.00)	0.35 (0.32)	0.35 (0.48)	0.75 (0.23)	0.68 (0.43)
12 (“Plum”)	9.6 (6.07)	28.0 (7.48)	0.51 (0.25)	0.79 (0.10)	0.63 (0.23)	0.33 (0.15)
13 (“Williams Pear”)	4.8 (3.35)	28.0 (11.66)	0.11 (0.15)	0.58 (0.18)	0.94 (0.09)	0.55 (0.21)
14 (“Taiti Lime”)	3.2 (5.22)	5.6 (6.07)	0.36 (0.50)	0.28 (0.37)	0.66 (0.46)	0.78 (0.34)
15 (“Watermelon”)	6.4 (3.58)	18.4 (15.65)	0.64 (0.46)	0.97 (0.01)	0.40 (0.47)	0.05 (0.02)

propriedades são bastante relevantes, especialmente na classificação de imagens. O algoritmo IC-EDS foi formalizado em um artigo científico para periódico internacional, o qual está em fase de revisão:

- COLETTA, L. F. S.; PONTI, M; HRUSCHKA, E. R.; ACHARYA, A.; GHOSH, J., *Combining Clustering and Active Learning for the Detection and Learning of New Image Classes*, 2016 — submetido para um periódico internacional de destaque.

CONCLUSÕES

Nesta tese, diferentes abordagens para a combinação de classificadores e agrupadores foram estudadas. O ponto de partida foi o algoritmo C^3E (*Consensus between Classification and Clustering Ensembles*) ([ACHARYA et al., 2014](#); [ACHARYA et al., 2011](#)). Este algoritmo recebe como entradas estimativas de distribuições de probabilidades de classes para objetos de um conjunto alvo, bem como uma matriz de similaridades entre esses objetos. Esta matriz é comumente produzida por agregadores de agrupadores de dados, enquanto que as distribuições de probabilidades de classes são obtidas a partir de agregadores de classificadores induzidos por um conjunto de treinamento. O C^3E produz como saída uma classificação consolidada dos objetos do conjunto alvo por meio de refinamentos nas distribuições de probabilidades de classes. A ideia intuitiva é que a informação de grupos (codificada pela matriz de similaridades) possa prover restrições complementares que auxiliem na classificação dos dados. A partir dessa noção, foram desenvolvidos algoritmos que superem algumas das dificuldades práticas da classificação de dados — especialmente quando se dispõe de poucos dados rotulados e/ou uma amostragem deficiente dos dados de treinamento. Em outras palavras, os estudos e experimentos conduzidos confirmaram a hipótese desta pesquisa de que variantes do algoritmo C^3E possam se beneficiar da combinação de aprendizado supervisionado e não supervisionado para gerarem melhores resultados de classificação. Em particular, os algoritmos concebidos fornecem classificadores mais “conscientes” das possíveis diferenças entre as distribuições dos dados de treinamento e dos dados do conjunto alvo, sendo também capazes de lidar com o surgimento de novas classes não observadas durante o treinamento do modelo.

6.1 Principais Contribuições

A combinação de classificadores e agrupadores foi investigada com foco em um caso particular do algoritmo C^3E ([ACHARYA et al., 2011](#)), o qual tem como base uma função de Perda Quadrática (C^3E -SL). Este algoritmo é mais simples. Seu custo computacional é $O(c \cdot n^2)$,

no qual c e n representam, respectivamente, o número de classes e o número de objetos no conjunto alvo. Com o intuito de suprimir a necessidade de informação de parâmetros requeridos pelo C³E-SL, uma abordagem baseada em meta-heurísticas de busca foi proposta (COLETTA *et al.*, 2015b; COLETTA *et al.*, 2013). Resultados provenientes desse estudo permitiram a concepção de um novo algoritmo evolutivo (COLETTA *et al.*, 2015a). Este algoritmo, chamado de *Dynamic DE* (D²E), é uma extensão do algoritmo *Differential Evolution* (DE) (PRICE, 1996; STORN; PRICE, 1997) capaz de adaptar seus próprios parâmetros de controle durante o processo evolutivo, o que torna a combinação de classificadores e agrupadores desempenhada pelo C³E-SL livre de parâmetros considerados críticos. Convém notar que o algoritmo original, C³E (ACHARYA *et al.*, 2011), exige a informação de três parâmetros (λ , α e I), enquanto que o C³E-SL requer apenas α e I , os quais, então, tem sido automaticamente estimados pelo algoritmo D²E. Durante estas investigações observou-se que valores ótimos de α são tipicamente baixos (inferiores a 0.5), e para I raramente é maior que 10. Portanto, pode-se dizer que o C³E-SL é bastante eficiente em termos do número de iterações requerido, apresentando-se, em alguns casos, com sendo relativamente robusto aos valores de α e I ¹. Além disso, observando-se com maior atenção a função objetivo (2.16) do algoritmo, e também os experimentos conduzidos nesta etapa, sugere-se a existência de uma correlação negativa entre os valores promissores de α e I (a exemplo da Figura 4, a qual é representativa da maioria dos resultados para diferentes bases de dados).

A abordagem com uso do algoritmo D²E foi a base para o desenvolvimento de variantes do C³E-SL que sejam mais práticas e flexíveis². Os algoritmos concebidos neste trabalho tiraram proveito, portanto, da combinação de classificadores e agrupadores para superar algumas das dificuldades da tarefa de classificação. Presumindo-se cenários semissupervisionados, uma versão do C³E-SL baseada em autotreinamento — *self-training* (ZHU; GOLDBERG, 2009) — foi proposta. Experimentos na análise de sentimentos em *tweets* mostraram que o uso de informação não supervisionada é útil para a descoberta de estruturas intrínsecas dos dados, melhorando a classificação especialmente quando se dispõe de uma quantidade limitada de dados rotulados (COLETTA *et al.*, 2014). Para lidar com o surgimento de novas classes não observadas nos conjuntos de treinamento disponíveis, uma extensão do algoritmo C³E-SL baseada em aprendizado ativo — *active learning* (SETTLES, 2012) — foi introduzida. Experimentos na área de classificação de imagens mostraram que o classificador resultante é menos dependente do conhecimento *a priori*, além de ser capaz de apropriadamente se adaptar para reconhecer novas classes presentes em novos dados que se deseja classificar.

Em resumo, os algoritmos desenvolvidos cumpriram, com êxito, os objetivos desta tese sendo capazes de fornecer classificadores com propriedades bastante apreciadas e com grande

¹ Para as bases de dados de imagens (*Coil-20*, *Corel-1000* e *Supermarket Produce*) notou-se que resultados competitivos são alcançados fixando-se $\alpha = 0.01$ e $I = 1$.

² Note também que, tanto no algoritmo de otimização (C³E-SL) quanto na abordagem evolutiva para otimização paramétrica (com uso do D²E), a paralelização de seus componentes pode ser naturalmente projetável.

potencial prático, tais como (COLETTA *et al.*, 2015a; COLETTA *et al.*, 2015b; ACHARYA *et al.*, 2014; COLETTA *et al.*, 2014; COLETTA *et al.*, 2013; ACHARYA *et al.*, 2011):

- Menor dependência do usuário/especialista de domínio a partir do uso de abordagens que são livres de parâmetros considerados críticos, além de serem mais inteligentes com relação à escolha de dados para treinamento;
- Ampla abrangência de aplicação, pois as propriedades dos algoritmos são especialmente úteis em cenários com poucos dados rotulados (aprendizado semissupervisionado), bem como em cenários suscetíveis à mudanças na distribuição dos dados e o surgimento de novas classes;
- Versatilidade da ferramenta, pois o algoritmo C³E-SL aceita o uso de (praticamente) qualquer componente nos agregadores, independente de atribuições rígidas ou suaves de classes e/ou grupos;
- Suporte à computação distribuída e restrições de privacidade, pois os dados de treinamento não necessitam estar disponíveis após a construção do agregador de classificadores, favorecendo aplicações em que o acesso aos atributos é proibitivo.

6.2 Publicações Geradas e Artigos Submetidos

As contribuições obtidas com o desenvolvimento desta tese de doutorado foram divulgadas por meio de publicações em conferências e periódicos internacionais de destaque. A seguir, é apresentada uma lista completa dos artigos publicados e submetidos:

- COLETTA, L. F. S.; HRUSCHKA, E. R.; ACHARYA, A.; GHOSH, J., *Towards the Use of Metaheuristics for Optimizing the Combination of Classifier and Cluster Ensembles*. In: 1st BRICS Countries Congress on Computational Intelligence (BRICS-CCI), 2013, Porto de Galinhas. Proceedings of the 1st BRICS Countries Congress on Computational Intelligence (BRICS-CCI), p. 483-488, 2013 — (COLETTA *et al.*, 2013).
- COLETTA, L. F. S.; SILVA, N. F. F.; HRUSCHKA, E. R.; HRUSCHKA JR., E. R., *Combining Classification and Clustering for Tweet Sentiment Analysis*. In: Brazilian Conference on Intelligent Systems (BRACIS), 2014, São Carlos. Proceedings on Brazilian Conference on Intelligent Systems, p. 210-215, 2014 — (COLETTA *et al.*, 2014).
- COLETTA, L. F. S.; HRUSCHKA, E. R.; ACHARYA, A.; GHOSH, J., *A Differential Evolution Algorithm to Optimise the Combination of Classifier and Cluster Ensembles*, International Journal of Bio-Inspired Computation, v. 7, n. 2, p. 111-124, 2015 — (COLETTA *et al.*, 2015a).

- COLETTA, L. F. S.; HRUSCHKA, E. R.; ACHARYA, A.; GHOSH, J., *Using Metaheuristics to Optimize the Combination of Classifier and Cluster Ensembles*. Integrated Computer-Aided Engineering, v. 22, n. 3, p. 229-242, 2015 — (COLETTA *et al.*, 2015b).
- SILVA, N. F. F.; COLETTA, L. F. S.; HRUSCHKA, E. R.; HRUSCHKA JR., E. R., *Using Unsupervised Information to Improve Semi-supervised Tweet Sentiment Classification*, 2016 — submetido para periódico internacional e em revisão.
- SILVA, N. F. F.; COLETTA, L. F. S.; HRUSCHKA, E. R., *A Survey and Comparative Study of Tweet Sentiment Analysis via Semi-Supervised Learning*, 2016 — submetido para periódico internacional e em revisão.
- COLETTA, L. F. S.; PONTI, M; HRUSCHKA, E. R.; ACHARYA, A.; GHOSH, J., *Combining Clustering and Active Learning for the Detection and Learning of New Image Classes*, 2016 — submetido para periódico internacional e em revisão.

6.3 Limitações e Trabalhos Futuros

Considerando a combinação de classificadores e agrupadores realizada pelo C³E-SL, diversos aspectos podem ser melhor estudados em trabalhos futuros. O impacto da quantidade e dos tipos de classificadores usados no agregador, bem como maneiras eficientes de se obter matrizes de similaridades para o propósito de consenso são tópicos que merecem maior investigação. Em uma perspectiva mais geral, a importância relativa de cada um dos componentes dos agregadores poderia ser capturada na forma de pesos que, então, poderiam ser ajustados de acordo com a aplicação. Especificamente, pode-se explorar formas de se produzir entradas para o C³E-SL que contenham um certo ganho de informação e não sejam redundantes a exemplo das estratégias adotadas para se obter componentes diversificados em agregadores (NALDI; CARVALHO; CAMPELLO, 2013; WINDEATT, 2005; KUNCHEVA; HADJITODOROV, 2004; KUNCHEVA; WHITAKER, 2003). Estas medidas poderiam diminuir as chances do C³E-SL de fornecer resultados ruins. Em outras palavras, o refinamento de classificações a partir de informação não supervisionada nem sempre oferece melhorias nos resultados. Isto é uma limitação do algoritmo. Testes para obter matrizes de similaridades promissoras (evitando a deterioração dos resultados) indicam que tais matrizes devam apresentar informações complementares àquelas providas do componente supervisionado. Nesse contexto, um mecanismo mais inteligente para se gerar matrizes de similaridades poderia ser estudado levando-se em conta a indução de agrupamentos a partir de diferentes “visões” dos dados, com ênfase para a seleção de conjuntos de atributos que sejam relevantes (os quais, por exemplo, poderiam ser aqueles que melhor discriminam as classes). A partir disso, é importante conduzir pesquisas sobre medidas de desempenho úteis para a avaliação das partições de dados e a matriz de similaridades, bem como estudar o uso de *scores* na combinação (SCHEIRER *et al.*, 2010).

Ao se utilizar a versão semissupervisionada do C³E-SL, estudos para considerar a informação não supervisionada, em conjunto com as probabilidades de classes, como um critério mais refinado de seleção de objetos se mostra bem interessante, a exemplo de índices mais sofisticados como em (SCHEIRER *et al.*, 2011). Neste contexto, a maior dificuldade reside na obtenção de informação não supervisionada complementar que possa, de fato, alavancar o modelo semissupervisionado evitando-se a incorporação, no conjunto de treinamento, de objetos com rótulos incorretos. Esta questão não foi investigada por completo nesta pesquisa e merece atenção, uma vez que a promoção de objetos com rótulos incorretos é a principal limitação do autotreinamento. Portanto, monitorar e estudar a seleção de objetos a serem promovidos, bem como conceber melhores critérios de seleção a partir do comportamento do algoritmo seriam significativamente úteis.

Focando-se na abordagem para a descoberta de novas classes, explorar os impactos do valor de h na Equação 5.2, bem como estudar diferentes critérios de densidade e entropia, são outros tópicos relevantes. Em particular, a seleção de objetos para rotulamento feita pelo Algoritmo 4 poderia levar em conta também a distância dos objetos presentes no conjunto de treinamento inicial. Sob esta perspectiva, e visto que o ponto mais desafiador da abordagem é a estratégia de seleção de objetos, outros mecanismos de seleção poderiam ser derivados e comparados com o proposto neste trabalho.

Finalmente, áreas de pesquisas nas quais acredita-se que as abordagens concebidas nesta tese também possam ser úteis são, por exemplo, as que envolvem fluxos contínuos de dados (*data streams*) e a classificação hierárquica e multirrótulo dos dados. Tais campos de estudo podem ser desafiadores e interessantes para o desenvolvimento de mais avanços na combinação de classificadores e agrupadores. Além disso, no contexto de “*Big Data*”, experimentos que envolvam grandes volumes de dados poderiam ser conduzidos a partir do uso de bases de dados de repositórios públicos. Tais investigações têm grande apelo prático e envolvem dados reais sobre o clima, a saúde e a economia mundial, bem como provenientes de diversos ramos da biologia e genética, além de pesquisas espaciais, comportamento em redes sociais e informações da Web³.

³ Grandes volumes de dados públicos estão disponíveis em endereços tais como <<http://www.kdnuggets.com/datasets/>>, <<http://aws.amazon.com/datasets/>> e <<http://www.gapminder.org/data/>>.

REFERÊNCIAS

- ACHARYA, A.; HRUSCHKA, E.; GHOSH, J.; ACHARYYA, S. C³E: A framework for combining ensembles of classifiers and clusterers. In: **Multiple Classifier Systems**. [S.l.]: Springer Berlin Heidelberg, 2011, (Lecture Notes in Computer Science, v. 6713). p. 269–278. ISBN 978-3-642-21556-8. Citado 25 vezes nas páginas 15, 19, 34, 35, 36, 39, 41, 42, 43, 44, 45, 46, 47, 48, 49, 53, 54, 55, 63, 64, 77, 84, 121, 122 e 123.
- ACHARYA, A.; HRUSCHKA, E. R.; GHOSH, J.; ACHARYYA, S. An optimization framework for combining ensembles of classifiers and clusterers with applications to nontransductive semisupervised learning and transfer learning. **ACM Trans. Knowl. Discov. Data**, ACM, New York, NY, USA, v. 9, n. 1, p. 1:1–1:35, ago. 2014. ISSN 1556-4681. Citado 8 vezes nas páginas 34, 39, 40, 55, 84, 99, 121 e 123.
- AGARWAL, A.; BHATTACHARYYA, P. Sentiment analysis: A new approach for effective use of linguistic knowledge and exploiting similarities in a set of documents to be classified. In: **Proceedings of the International Conference on Natural Language Processing (ICON)**. [S.l.: s.n.], 2005. Citado na página 82.
- AGARWAL, A.; XIE, B.; VOVSHA, I.; RAMBOW, O.; PASSONNEAU, R. Sentiment analysis of twitter data. In: **Proceedings of the Workshop on Languages in Social Media**. Stroudsburg, PA, USA: Association for Computational Linguistics, 2011. (LSM '11), p. 30–38. ISBN 978-1-932432-96-1. Citado na página 82.
- AHA, D. W.; KIBLER, D.; ALBERT, M. K. Instance-based learning algorithms. In: **Machine Learning**. Hingham, MA, USA: Kluwer Academic Publishers, 1991. v. 6, p. 37–66. Citado na página 29.
- ALPAYDIN, E. **Introduction to Machine Learning**. 2nd. ed. [S.l.]: The MIT Press, 2010. ISBN 026201243X, 9780262012430. Citado na página 29.
- BÄCK, T.; FOGEL, D. B.; MICHALEWICZ, Z. **Handbook of Evolutionary Computation**. Bristol, UK: IOP Publishing, 1997. ISBN 0750303921. Citado 2 vezes nas páginas 55 e 69.
- BÄCK, T.; HOFFMEISTER, F.; SCHWEFEL, H. P. A survey of evolution strategies. In: **Proceedings of the Fourth International Conference on Genetic Algorithms**. [S.l.: s.n.], 1991. p. 2–9. Citado na página 56.
- BÄCK, T.; SCHWEFEL, H.-P. An overview of evolutionary algorithms for parameter optimization. **Evol. Comput.**, MIT Press, Cambridge, MA, USA, v. 1, p. 1–23, 1993. ISSN 1063-6560. Citado na página 56.
- BAI, R.; CHAI, T.; MA, E. A novel approach for extraction of fuzzy rules using the neuro-fuzzy network and its application in the blending process of raw slurry. In: **Advances in Neural Networks, ISSN 2007**. [S.l.]: Springer Berlin/Heidelberg, 2007, (Lecture Notes in Computer Science, v. 4492). p. 362–370. Citado na página 40.

- BALAHUR, A.; STEINBERGER, R.; KABADJOV, M.; ZAVARELLA, V.; GOOT, E. van der; HALKIA, M.; POULIQUEN, B.; BELYAEVA, J. Sentiment analysis in the news. In: **Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)**. Valletta, Malta: European Language Resources Association (ELRA), 2010. ISBN 2-9517408-6-7. Citado na página [82](#).
- BANERJEE, A.; MERUGU, S.; DHILLON, I. S.; GHOSH, J. Clustering with bregman divergences. **J. Mach. Learn. Res.**, JMLR.org, v. 6, p. 1705–1749, dez. 2005. ISSN 1532-4435. Citado 2 vezes nas páginas [43](#) e [44](#).
- BART, E.; ULLMAN, S. Cross-generalization: learning novel classes from a single example by feature replacement. In: **Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on**. [S.l.: s.n.], 2005. v. 1, p. 672–679 vol. 1. ISSN 1063-6919. Citado na página [102](#).
- BASU, S.; BILENKO, M.; MOONEY, R. J. A probabilistic framework for semi-supervised clustering. In: **Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining**. New York, NY, USA: ACM, 2004. (KDD '04), p. 59–68. ISBN 1-58113-888-1. Citado na página [39](#).
- BASU, S.; DAVIDSON, I.; WAGSTAFF, K. **Constrained Clustering: Advances in Algorithms, Theory, and Applications**. [S.l.]: Chapman & Hall/CRC, 2008. Edição 1. ISBN 1584889969, 9781584889960. Citado 2 vezes nas páginas [34](#) e [39](#).
- BAUER, E.; KOHAVI, R. An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. In: **Machine Learning**. Hingham, MA, USA: Kluwer Academic Publishers, 1999. v. 36, p. 105–139. Citado na página [31](#).
- BAUGH, W. bwbaugh: Hierarchical sentiment analysis with partial self-training. In: **Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)**. Atlanta, Georgia, USA: Association for Computational Linguistics, 2013. p. 539–542. Citado 2 vezes nas páginas [89](#) e [91](#).
- BECKER, L.; ERHART, G.; SKIBA, D.; MATULA, V. AVAYA: Sentiment analysis on twitter with self-training and polarity lexicon expansion. In: **Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)**. Atlanta, Georgia, USA: Association for Computational Linguistics, 2013. p. 333–340. Citado 4 vezes nas páginas [84](#), [89](#), [91](#) e [95](#).
- BELKIN, M.; MATVEEVA, I.; NIYOGI, P. Regularization and semi-supervised learning on large graphs. In: SPRINGER. **COLT**. [S.l.], 2004. v. 3120, p. 624–638. Citado na página [83](#).
- BERGSTRA, J.; BENGIO, Y. Random search for hyper-parameter optimization. **J. Mach. Learn. Res.**, JMLR.org, v. 13, p. 281–305, 2012. ISSN 1532-4435. Citado na página [61](#).
- BEYER, H.-G.; SCHWEFEL, H.-P. Evolution strategies - a comprehensive introduction. **Natural Computing**, Springer Netherlands, v. 1, p. 3–52, 2002. ISSN 1567-7818. Citado na página [69](#).
- BEZDEK, J. C. **Pattern Recognition with Fuzzy Objective Function Algorithms**. Norwell, MA, USA: Kluwer Academic Publishers, 1981. ISBN 0306406713. Citado 2 vezes nas páginas [30](#) e [40](#).

BIEDERMAN, I. Recognition-by-components: a theory of human image understanding. **Psychological Review**, American Psychological Association, v. 94, n. 2, p. 115, 1987. Citado na página 101.

BIRGE, B. PSOT - a particle swarm optimization toolbox for use with matlab. In: **Swarm Intelligence Symposium, 2003. SIS '03. Proceedings of the 2003 IEEE**. [S.l.: s.n.], 2003. p. 182 – 186. Citado na página 58.

BISHOP, C. M. **Pattern Recognition and Machine Learning - Information Science and Statistics**. [S.l.]: Springer, 2007. Edição 1. ISBN 0387310738. Citado 6 vezes nas páginas 29, 30, 31, 35, 99 e 101.

BLUM, A.; CHAWLA, S. Learning from labeled and unlabeled data using graph mincuts. In: **Proceedings of the Eighteenth International Conference on Machine Learning (ICML 2001), Williams College, Williamstown, MA, USA, June 28 - July 1, 2001**. [S.l.: s.n.], 2001. p. 19–26. Citado na página 83.

BLUM, A.; MITCHELL, T. Combining labeled and unlabeled data with co-training. In: **Proceedings of the eleventh annual conference on Computational learning theory**. New York, NY, USA: ACM, 1998. (COLT' 98), p. 92–100. ISBN 1-58113-057-0. Citado na página 83.

BOSER, B. E.; GUYON, I. M.; VAPNIK, V. N. A training algorithm for optimal margin classifiers. In: **ACM. Proceedings of the fifth annual workshop on Computational learning theory**. [S.l.], 1992. p. 144–152. Citado 3 vezes nas páginas 82, 83 e 95.

BOUSQUET, O.; BOUCHERON, S.; LUGOSI, G. Introduction to Statistical Learning Theory. In: **Advanced Lectures on Machine Learning**. [S.l.: s.n.], 2004. p. 169–207. Citado na página 39.

BREIMAN, L. Bagging predictors. In: **Machine Learning**. Hingham, MA, USA: Kluwer Academic Publishers, 1996. v. 24, p. 123–140. Citado na página 31.

BREST, J.; GREINER, S.; BOSKOVIC, B.; MERNIK, M.; ZUMER, V. Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems. **Evolutionary Computation, IEEE Transactions on**, v. 10, n. 6, p. 646–657, 2006. ISSN 1089-778X. Citado 3 vezes nas páginas 56, 70 e 74.

CAI, W.; CHEN, S.; ZHANG, D. Robust fuzzy relational classifier incorporating the soft class labels. **Pattern Recognition Letters**, Elsevier Science Inc., New York, NY, USA, v. 28, p. 2250–2263, December 2007. ISSN 0167-8655. Citado 2 vezes nas páginas 34 e 40.

_____. A simultaneous learning framework for clustering and classification. **Pattern Recognition Letters**, v. 42, n. 7, p. 1248–1259, 2009. Citado 4 vezes nas páginas 34, 39, 40 e 53.

_____. A multiobjective simultaneous learning framework for clustering and classification. **Neural Networks, IEEE Transactions**, v. 21, n. 2, p. 185–200, 2010. ISSN 1045-9227. Citado 3 vezes nas páginas 34, 39 e 41.

CAMPELLO, R.; HRUSCHKA, E. A fuzzy extension of the silhouette width criterion for cluster analysis. **Fuzzy Sets and Systems**, v. 157, n. 21, p. 2858–2875, 2006. Citado 3 vezes nas páginas 52, 62 e 105.

- CAMPELLO, R. J.; HRUSCHKA, E. R.; ALVES, V. S. On the efficiency of evolutionary fuzzy clustering. **Journal of Heuristics**, Kluwer Academic Publishers, Hingham, MA, USA, v. 15, p. 43–75, February 2009. ISSN 1381-1231. Citado na página 105.
- CAMPOMANES-ÁLVAREZ, B. R.; CORDÓN, O.; DAMAS, S. Evolutionary multi-objective optimization for mesh simplification of 3D open models. **Integrated Computer-Aided Engineering**, IOS Press, v. 20, n. 4, p. 375–390, 2013. Citado na página 56.
- CARAFFINI, F.; NERI, F.; PASSOW, B.; IACCA, G. Re-sampled inheritance search: high performance despite the simplicity. **Soft Computing**, Springer Berlin Heidelberg, v. 17, n. 12, p. 2235–2256, 2013. ISSN 1432-7643. Citado 3 vezes nas páginas 56, 59 e 77.
- CARAFFINI, F.; NERI, F.; PICINALI, L. An analysis on separability for memetic computing automatic design. **Information Sciences**, v. 265, n. 0, p. 1–22, 2014. ISSN 0020-0255. Citado 2 vezes nas páginas 56 e 77.
- CARVALHO, P.; SARMENTO, L.; SILVA, M. J.; OLIVEIRA, E. de. Clues for detecting irony in user-generated contents: Oh...!! it's "so easy";-). In: **Proceedings of the 1st International CIKM Workshop on Topic-sentiment Analysis for Mass Opinion**. New York, NY, USA: ACM, 2009. (TSA '09), p. 53–56. Citado na página 97.
- CASTRO, L. D.; HRUSCHKA, E.; CAMPELLO, R. An evolutionary clustering technique with local search to design rbf neural network classifiers. In: **Neural Networks, 2004. Proceedings. 2004 IEEE International Joint Conference on**. [S.l.: s.n.], 2004. v. 3, p. 2083 – 2088 vol.3. ISSN 1098-7576. Citado 2 vezes nas páginas 34 e 40.
- CHAPELLE, O.; SCHÖLKOPF, B.; ZIEN, A. (Ed.). **Semi-Supervised Learning**. Cambridge, MA: MIT Press, 2006. Citado 3 vezes nas páginas 34, 79 e 81.
- CHEN, S.; GUO, G.; CHEN, L. Semi-supervised classification based on clustering ensembles. In: **Proceedings of the International Conference on Artificial Intelligence and Computational Intelligence**. Berlin, Heidelberg: Springer-Verlag, 2009. (AICI '09), p. 629–638. ISBN 978-3-642-05252-1. Citado na página 34.
- CHEN, Y.; NASRABADI, N. M.; TRAN, T. D. Hyperspectral image classification via kernel sparse representation. **Geoscience and Remote Sensing, IEEE Transactions on**, IEEE, v. 51, n. 1, p. 217–231, 2013. Citado na página 101.
- CHEONG, M.; LEE, V. C. A microblogging-based approach to terrorism informatics: Exploration and chronicling civilian sentiment and response to terrorism events via twitter. **Information Systems Frontiers**, Kluwer Academic Publishers, Hingham, MA, USA, v. 13, n. 1, p. 45–59, mar. 2011. ISSN 1387-3326. Citado na página 82.
- CLEMEN, R. T. Combining forecasts: A review and annotated bibliography. **International Journal of Forecasting**, v. 5, n. 4, p. 559 – 583, 1989. ISSN 0169-2070. Citado na página 31.
- CLERC, M. **Particle swarm optimization**. [S.l.]: John Wiley & Sons, 2010. Citado na página 56.
- COLETTA, L.; HRUSCHKA, E.; ACHARYA, A.; GHOSH, J. Towards the use of metaheuristics for optimizing the combination of classifier and cluster ensembles. In: **Computational Intelligence and 11th Brazilian Congress on Computational Intelligence (BRICS-CCI CBIC), 2013 BRICS Congress on**. [S.l.: s.n.], 2013. p. 483–488. Citado 3 vezes nas páginas 78, 122 e 123.

COLETTA, L.; SILVA, N. da; HRUSCHKA, E.; HRUSCHKA, E. Combining classification and clustering for tweet sentiment analysis. In: **Intelligent Systems (BRACIS), 2014 Brazilian Conference on**. [S.l.: s.n.], 2014. p. 210–215. Citado 7 vezes nas páginas 54, 82, 84, 95, 97, 122 e 123.

COLETTA, L.; VENDRAMIN, L.; HRUSCHKA, E.; CAMPELLO, R.; PEDRYCZ, W. Collaborative fuzzy clustering algorithms: Some refinements and design guidelines. **IEEE Transactions on Fuzzy Systems**, v. 20, n. 3, p. 444–462, 2012. ISSN 1063-6706. Citado na página 33.

COLETTA, L. F.; HRUSCHKA, E. R.; ACHARYA, A.; GHOSH, J. A differential evolution algorithm to optimise the combination of classifier and cluster ensembles. **International Journal of Bio-Inspired Computation**, Inderscience Publishers (IEL), v. 7, n. 2, p. 111–124, 2015. Citado 6 vezes nas páginas 15, 54, 78, 86, 122 e 123.

_____. Using metaheuristics to optimize the combination of classifier and cluster ensembles. **Integrated Computer-Aided Engineering**, IOS Press, v. 22, n. 3, p. 229–242, 2015. Citado 6 vezes nas páginas 78, 84, 99, 122, 123 e 124.

CORTES, C.; VAPNIK, V. Support-vector networks. **Machine learning**, Springer, v. 20, n. 3, p. 273–297, 1995. Citado 3 vezes nas páginas 82, 83 e 95.

COSTA, F. de O.; SILVA, E.; ECKMANN, M.; SCHEIRER, W. J.; ROCHA, A. Open set source camera attribution and device linking. **Pattern Recognition Letters**, v. 39, n. 0, p. 92 – 101, 2014. ISSN 0167-8655. Advances in Pattern Recognition and Computer Vision. Citado na página 102.

CRAMMER, K.; KULESZA, A.; DREDZE, M. Adaptive regularization of weight vectors. In: **NIPS**. [S.l.: s.n.], 2009. p. 414–422. Citado 2 vezes nas páginas 84 e 95.

CRUPI, V.; GUGLIELMINO, E.; MILAZZO, G. Neural-network-based system for novel fault detection in rotating machinery. **Journal of Vibration and Control**, SAGE Publications, v. 10, n. 8, p. 1137–1150, 2004. Citado 2 vezes nas páginas 100 e 102.

DARWIN, C.; BEER, G. **The Origin of Species**. [S.l.]: Oxford University Press, 1998. (Oxford World's Classics). ISBN 9780192834386. Citado na página 69.

DAS, S.; KONAR, A.; CHAKRABORTY, U. K. Two improved differential evolution schemes for faster global search. In: **Proceedings of the 2005 conference on Genetic and evolutionary computation**. [S.l.]: ACM, 2005. (GECCO '05), p. 991–998. ISBN 1-59593-010-8. Citado na página 70.

DAS, S.; SUGANTHAN, P. Differential evolution: A survey of the state-of-the-art. **Evolutionary Computation, IEEE Transactions on**, v. 15, n. 1, p. 4–31, 2011. ISSN 1089-778X. Citado 3 vezes nas páginas 56, 69 e 70.

DEMPSTER, A. P.; LAIRD, N. M.; RUBIN, D. B. Maximum likelihood from incomplete data via the em algorithm. **Journal of the Royal Statistical Society, Series B**, v. 39, n. 1, p. 1–38, 1977. Citado 2 vezes nas páginas 30 e 83.

DEMSAR, J. Statistical comparisons of classifiers over multiple data sets. **J. Mach. Learn. Res.**, JMLR.org, v. 7, p. 1–30, December 2006. ISSN 1532-4435. Citado 2 vezes nas páginas 66 e 72.

DIAKOPOULOS, N. A.; SHAMMA, D. A. Characterizing debate performance via aggregated twitter sentiment. In: **Proceedings of the SIGCHI Conference on Human Factors in Computing Systems**. New York, NY, USA: ACM, 2010. (CHI '10), p. 1195–1198. ISBN 978-1-60558-929-9. Citado na página 82.

DIEHL, C.; HAMPSHIRE J.B., I. Real-time object classification and novelty detection for collaborative video surveillance. In: **Neural Networks, 2002. IJCNN '02. Proceedings of the 2002 International Joint Conference on**. [S.l.: s.n.], 2002. v. 3, p. 2620–2625. ISSN 1098-7576. Citado na página 102.

DIETTERICH, T. G. Ensemble methods in machine learning. In: **International Workshop on Multiple Classifier Systems**. [S.l.]: Springer-Verlag, 2000. p. 1–15. Citado na página 31.

DIMITRIADOU, E.; WEINGESSEL, A.; HORNIK, K. Voting-merging: An ensemble method for clustering. In: DORFFNER, G.; BISCHOF, H.; HORNIK, K. (Ed.). **Artificial Neural Networks, ICANN 2001**. [S.l.]: Springer Berlin / Heidelberg, 2001, (Lecture Notes in Computer Science, v. 2130). p. 217–224. Citado na página 33.

DONG, C.; WANG, G.; CHEN, Z. The inertia weight self-adapting in pso. In: **Intelligent Control and Automation, 2008. WCICA 2008. 7th World Congress on**. [S.l.: s.n.], 2008. p. 5313–5316. Citado na página 56.

DOWSLAND, K.; THOMPSON, J. Simulated annealing. In: ROZENBERG, G.; BÄCK, T.; KOK, J. (Ed.). **Handbook of Natural Computing**. [S.l.]: Springer Berlin Heidelberg, 2012. p. 1623–1655. ISBN 978-3-540-92909-3. Citado na página 56.

DUIN, R.; PEKALSKA, E. Open issues in pattern recognition. In: **Computer Recognition Systems**. [S.l.]: Springer Berlin Heidelberg, 2005, (Advances in Soft Computing, v. 30). p. 27–42. Citado na página 99.

DUVAL-POO, M. A.; SOSA-GARCIA, J.; GUERRA-GANDON, A.; VEGA-PONS, S.; RUIZ-SHULCLOPER, J. A new classifier combination scheme using clustering ensemble. In: **Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications**. [S.l.]: Springer Berlin Heidelberg, 2012, (Lecture Notes in Computer Science, v. 7441). p. 154–161. ISBN 978-3-642-33274-6. Citado 2 vezes nas páginas 34 e 39.

EFRON, B.; TIBSHIRANI, R. J. **An Introduction to the Bootstrap**. New York: Chapman & Hall, 1993. Citado na página 31.

EIBEN, A. E.; SMITH, J. E. **Introduction to Evolutionary Computing**. [S.l.]: SpringerVerlag, 2003. ISBN 3540401849. Citado na página 69.

_____. **Introduction to Evolutionary Computing (Natural Computing Series)**. [S.l.]: Springer, 2008. Hardcover. ISBN 3540401849. Citado 4 vezes nas páginas 55, 69, 70 e 77.

EKMAN, P. **Emotion in the Human Face**. [S.l.]: Cambridge University Press, 1982. Citado na página 82.

EVERITT, B. S.; LANDAU, S.; LEESE, M. **Cluster Analysis**. [S.l.]: Wiley Publishing, 2001. Edição 4. ISBN 0340761199, 9780340761199. Citado na página 30.

FAYYAD, U. M.; PIATETSKY-SHAPIO, G.; SMYTH, P.; UTHURUSAMY, R. Preface. In: **Advances in knowledge discovery and data mining**. Menlo Park, CA, USA: American Association for Artificial Intelligence, 1996. p. xiii–xiv. ISBN 0-262-56097-6. Citado na página 34.

FLEMING, P.; PURSHOUSE, R. Evolutionary algorithms in control systems engineering: a survey. **Control Engineering Practice**, v. 10, n. 11, p. 1223–1241, 2002. ISSN 0967-0661. Citado 2 vezes nas páginas 55 e 56.

FOODY, G. M.; MATHUR, A. Toward intelligent training of supervised image classifications: directing training data acquisition for {SVM} classification. **Remote Sensing of Environment**, v. 93, n. 1?2, p. 107 – 117, 2004. ISSN 0034-4257. Citado 3 vezes nas páginas 35, 99 e 101.

FORCAEL, E.; GONZÁLEZ, V.; OROZCO, F.; VARGAS, S.; PANTOJA, A.; MOSCOSO, P. Ant colony optimization model for tsunamis evacuation routes. **Computer-Aided Civil and Infrastructure Engineering**, Wiley Online Library, v. 29, p. 723–737, 2014. Citado na página 56.

FRANK, A.; ASUNCION, A. **UCI Machine Learning Repository**. 2010. University of California, School of Information and Computer Science, Irvine, CA. Disponível em: <http://archive.ics.uci.edu/ml>. Citado 5 vezes nas páginas 15, 49, 51, 60 e 63.

FRED, A.; JAIN, A. Combining multiple clusterings using evidence accumulation. **Pattern Analysis and Machine Intelligence, IEEE Transactions on**, v. 27, n. 6, p. 835 – 850, jun 2005. ISSN 0162-8828. Citado na página 33.

FREITAS, A. A. **Data Mining and Knowledge Discovery with Evolutionary Algorithms**. New York, -USA: Springer-Verlag, 2002. ISBN 978-3-540-43331-6. Citado 2 vezes nas páginas 56 e 69.

_____. A survey of evolutionary algorithms for data mining and knowledge discovery. In: _____. New York, NY, USA: Springer-Verlag New York, Inc., 2003. p. 819–845. ISBN 3-540-43330-9. Citado 2 vezes nas páginas 55 e 69.

FREITAS, L. A. de; VANIN, A. A.; HOGETOP, D. N.; BOCHERNITSAN, M. N.; VIEIRA, R. Pathways for irony detection in tweets. In: **Proceedings of the 29th Annual ACM Symposium on Applied Computing**. New York, NY, USA: ACM, 2014. (SAC '14), p. 628–633. ISBN 978-1-4503-2469-4. Citado na página 97.

FREUND, Y.; SCHAPIRE, R. E. A decision-theoretic generalization of on-line learning and an application to boosting,. **Journal of Computer and System Sciences**, v. 55, n. 1, p. 119 – 139, 1997. ISSN 0022-0000. Citado na página 31.

FROSSYNIOTIS, D. S.; PATERITSAS, C.; STAFYLOPATIS, A. A multi-clustering fusion scheme for data partitioning. **Int. J. Neural Syst.**, v. 15, n. 5, p. 391–401, 2005. Citado na página 33.

GAMPERLE, R.; MÜLLER, S. D.; KOUMOUTSAKOS, P. A parameter study for differential evolution. In: **WSEAS Int. Conf. on Advances in Intelligent Systems, Fuzzy Systems, Evolutionary Computation**. [S.l.]: Press, 2002. p. 293–298. Citado na página 70.

GAN, H.; SANG, N.; HUANG, R.; TONG, X.; DAN, Z. Using clustering analysis to improve semi-supervised classification. **Neurocomputing**, v. 101, n. 0, p. 290–298, 2013. ISSN 0925-2312. Citado 2 vezes nas páginas 34 e 39.

- GAO, J.; FAN, W.; JIANG, J.; HAN, J. Knowledge transfer via multiple model local structure mapping. In: **Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining**. New York, NY, USA: ACM, 2008. (KDD '08), p. 283–291. ISBN 978-1-60558-193-4. Citado na página 35.
- GAO, J.; LIANG, F.; FAN, W.; SUN, Y.; HAN, J. A graph-based consensus maximization approach for combining multiple supervised and unsupervised models. **Knowledge and Data Engineering, IEEE Transactions on**, v. 25, n. 1, p. 15–28, 2013. ISSN 1041-4347. Citado 2 vezes nas páginas 34 e 39.
- GAO, J.; LIANGY, F.; FANZ, W.; SUNY, Y.; HAN, J. Graph-based Consensus Maximization among Multiple Supervised and Unsupervised Models. In: **23rd Annual Conference on Neural Information Processing Systems**. [S.l.: s.n.], 2009. Citado 5 vezes nas páginas 34, 39, 41, 49 e 53.
- GHAEMI, R.; SULAIMAN, N.; IBRAHIM, H.; MUSTAPHA, N. A Survey: Clustering Ensembles Techniques. **Proceedings of World Academy of Science, Engineering and Technology**, v. 38, February 2009. ISSN 2070-3740. Citado 3 vezes nas páginas 33, 34 e 54.
- GHOSH, J.; ACHARYA, A. Cluster ensembles. **Wiley Interdisc. Rew.: Data Mining and Knowledge Discovery**, v. 1, n. 4, p. 305–315, 2011. Citado 3 vezes nas páginas 32, 33 e 54.
- GIONIS, A.; MANNILA, H.; TSAPARAS, P. Clustering aggregation. In: **ACM Trans. Knowl. Discov. Data**. New York, NY, USA: ACM, 2007. v. 1. Citado na página 34.
- GOLDBERG, A. **New directions in semi-supervised learning**. Tese (Doutorado) — University of Wisconsin - Madison, 2010. Citado 3 vezes nas páginas 79, 81 e 83.
- GOLDBERG, A. B.; ZHU, X. Seeing stars when there aren't many stars: Graph-based semi-supervised learning for sentiment categorization. In: **Proceedings of the First Workshop on Graph Based Methods for Natural Language Processing**. Stroudsburg, PA, USA: Association for Computational Linguistics, 2006. (TextGraphs-1), p. 45–52. Citado na página 83.
- GONZALEZ-IBANEZ, R.; MURESAN, S.; WACHOLDER, N. Identifying sarcasm in twitter: A closer look. In: **Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers - Volume 2**. Stroudsburg, PA, USA: Association for Computational Linguistics, 2011. (HLT '11), p. 581–586. Citado na página 97.
- GUILLAUMIN, M.; VERBEEK, J.; SCHMID, C. Multimodal semi-supervised learning for image classification. In: **Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on**. [S.l.: s.n.], 2010. p. 902–909. ISSN 1063-6919. Citado 2 vezes nas páginas 99 e 101.
- HAJMOHAMMADI, M. S.; IBRAHIM, R.; SELAMAT, A.; FUJITA, H. Combination of active learning and self-training for cross-lingual sentiment classification with density analysis of unlabelled samples. **Information Sciences**, v. 317, n. 0, p. 67 – 77, 2015. Citado na página 98.
- HAN, J.; KAMBER, M. **Data Mining. Concepts and Techniques**. [S.l.]: Morgan Kaufmann, 2006. Edição 2. ISBN 1558609016. Citado 2 vezes nas páginas 29 e 34.
- HAND, D. J. Classifier technology and the illusion of progress. **Statistical Science**, The Institute of Mathematical Statistics, v. 21, n. 1, p. 1–14, 2006. Citado na página 99.

HAND, D. J.; MANNILA, H.; SMYTH, P. **Principles of Data Mining (Adaptive Computation and Machine Learning)**. [S.l.]: The MIT Press, 2001. Hardcover. ISBN 026208290X. Citado na página 30.

HANSEN, L.; SALAMON, P. Neural network ensembles. **Pattern Analysis and Machine Intelligence, IEEE Transactions on**, v. 12, n. 10, p. 993 –1001, out. 1990. ISSN 0162-8828. Citado na página 31.

HANSEN, N. The cma evolution strategy: A comparing review. In: LOZANO, J.; LARRAÑAGA, P.; INZA, I.; BENGOTXEA, E. (Ed.). **Towards a New Evolutionary Computation**. [S.l.]: Springer Berlin / Heidelberg, 2006, (Studies in Fuzziness and Soft Computing, v. 192). p. 75–102. ISBN 978-3-540-29006-3. Citado na página 57.

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. Citado 3 vezes nas páginas 56, 57 e 77.

HARALICK, R.; SHANMUGAN, K.; DINSTEN, I. Textural features for image classification. **IEEE Transactions on Systems, Man and Cybernetics**, SMC-3, n. 6, p. 610–621, 1973. Citado na página 109.

HAYKIN, S. **Neural Networks: A Comprehensive Foundation**. Upper Saddle River, NJ: Prentice Hall, 1999. Edição 2. Citado na página 29.

HE, B.; MACDONALD, C.; HE, J.; OUNIS, I. An effective statistical approach to blog post opinion retrieval. In: **Proceedings of the 17th ACM conference on Information and knowledge management**. New York, NY, USA: ACM, 2008. (CIKM '08), p. 1063–1072. ISBN 978-1-59593-991-3. Citado na página 82.

HE, Y.; ZHOU, D. Self-training from labeled features for sentiment analysis. **Information Processing & Management**, v. 47, n. 4, p. 606 – 616, 2011. ISSN 0306-4573. Citado 2 vezes nas páginas 84 e 95.

HEJAZI, F.; TOLOUE, I.; JAAFAR, M.; NOORZAEI, J. Optimization of earthquake energy dissipation system by genetic algorithm. **Computer-Aided Civil and Infrastructure Engineering**, Wiley Online Library, v. 28, n. 10, p. 796–810, 2013. Citado na página 56.

HODGE, V.; AUSTIN, J. A survey of outlier detection methodologies. **Artificial Intelligence Review**, Kluwer Academic Publishers, v. 22, n. 2, p. 85–126, 2004. ISSN 0269-2821. Citado na página 100.

HOLMES, D.; JAIN, L. Introduction to bayesian networks. In: **Innovations in Bayesian Networks**. [S.l.]: Springer Berlin / Heidelberg, 2008, (Studies in Computational Intelligence, v. 156). p. 1–5. Citado na página 29.

HORE, P.; HALL, L.; GOLDFOF, D. A scalable framework for cluster ensembles. **Pattern Recognition**, v. 42, n. 5, p. 676–688, 2009. Citado 2 vezes nas páginas 33 e 34.

HRUSCHKA, E.; CAMPELLO, R.; FREITAS, A.; CARVALHO, A. de. A survey of evolutionary algorithms for clustering. **Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on**, v. 39, n. 2, p. 133 –155, 2009. ISSN 1094-6977. Citado na página 69.

HRUSCHKA, E. R.; CAMPELLO, R. J.; CASTRO, L. N. D. Evolving clusters in gene-expression data. **Information Sciences**, v. 176, n. 13, p. 1898–1927, jul. 2006. ISSN 0020-0255. Citado 3 vezes nas páginas 52, 62 e 105.

HSU, C.-W.; CHANG, C.-C.; LIN, C.-J. *et al.* A practical guide to support vector classification, 2003. 2010. Disponível em: <<http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf>>. Citado na página 105.

HSU, W.-Y. Application of quantum-behaved particle swarm optimization to motor imagery EEG classification. **International Journal of Neural Systems**, v. 23, n. 06, p. 1350026, 2013. Citado na página 56.

HU, M.; LIU, B. Mining and summarizing customer reviews. In: **Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining**. New York, NY, USA: ACM, 2004. (KDD '04), p. 168–177. Citado 2 vezes nas páginas 82 e 90.

_____. Mining opinion features in customer reviews. In: **Proceedings of the 19th national conference on Artificial intelligence**. [S.l.]: AAAI Press, 2004. (AAAI'04), p. 755–760. ISBN 0-262-51183-5. Citado na página 82.

INGBER, L. *et al.* Adaptive simulated annealing (ASA): Lessons learned. **Control and cybernetics**, POLISH ACADEMY OF SCIENCES, v. 25, p. 33–54, 1996. Citado na página 56.

JAAKKOLA, M. S. T.; SZUMMER, M. Partially labeled classification with markov random walks. **Advances in neural information processing systems (NIPS)**, v. 14, p. 945–952, 2002. Citado na página 83.

JAIN, A.; DUIN, R.; MAO, J. Statistical pattern recognition: a review. **Pattern Analysis and Machine Intelligence, IEEE Transactions on**, v. 22, n. 1, p. 4–37, jan 2000. ISSN 0162-8828. Citado na página 33.

JAIN, A.; MURTY, M.; FLYNN, P. Data clustering: A review. **ACM Computing Surveys**, v. 31, p. 264–323, 1999. Citado na página 33.

JAIN, A. K. Data clustering: 50 years beyond k-means. **Pattern Recognition Letters**, v. 31, n. 8, p. 651–666, 2010. ISSN 0167-8655. Citado 2 vezes nas páginas 62 e 105.

JAIN, A. K.; DUBES, R. C. **Algorithms for clustering data**. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1988. ISBN 0-13-022278-X. Citado na página 30.

JANSEN, B. J.; ZHANG, M.; SOBEL, K.; CHOWDURY, A. Twitter power: Tweets as electronic word of mouth. **J. Am. Soc. Inf. Sci. Technol.**, John Wiley & Sons, Inc., New York, NY, USA, v. 60, n. 11, p. 2169–2188, nov 2009. ISSN 1532-2882. Citado na página 82.

JIA, H.; MURPHEY, Y. L.; GUTCHESS, D.; CHANG, T.-S. Identifying knowledge domain and incremental new class learning in svm. In: **Neural Networks, 2005. IJCNN '05. Proceedings. 2005 IEEE International Joint Conference on**. [S.l.: s.n.], 2005. v. 5, p. 2742–2747. Citado 3 vezes nas páginas 35, 78 e 79.

JIA, L.; WANG, Y.; FAN, L. Multiobjective bilevel optimization for production-distribution planning problems using hybrid genetic algorithm. **Integrated Computer-Aided Engineering**, IOS Press, v. 21, n. 1, p. 77–90, 2014. Citado na página 56.

JING, L.; ZHANG, C.; NG, M. SNMFCA: Supervised NMF-based image classification and annotation. **Image Processing, IEEE Transactions on**, v. 21, n. 11, p. 4508–4521, Nov 2012. ISSN 1057-7149. Citado na página 101.

JOACHIMS, T. Making large-scale SVM learning practical. In: SCHÖLKOPF, B.; BURGESS, C.; SMOLA, A. (Ed.). **Advances in Kernel Methods - Support Vector Learning**. Cambridge, MA: MIT Press, 1999. cap. 11, p. 169–184. Citado na página 83.

_____. Transductive inference for text classification using support vector machines. In: **Proceedings of the Sixteenth International Conference on Machine Learning**. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1999. (ICML '99), p. 200–209. ISBN 1-55860-612-2. Citado na página 83.

JUN, G.; GHOSH, J. Semisupervised Learning of Hyperspectral Data With Unknown Land-Cover Classes. **IEEE Transactions on Geoscience and Remote Sensing**, v. 51, n. 1, p. 273–282, 2013. ISSN 0196-2892. Citado na página 102.

KAUFMAN, L.; ROUSSEEUW, P. **Finding Groups in Data: An Introduction to Cluster Analysis**. New York: Wiley Interscience, 1990. Citado na página 30.

KENNEDY, J.; EBERHART, R. Particle swarm optimization. In: **Neural Networks, 1995. Proceedings., IEEE International Conference on**. [S.l.: s.n.], 1995. v. 4, p. 1942–1948. Citado 3 vezes nas páginas 56, 58 e 77.

KHAN, L.; AWAD, M.; THURASINGHAM, B. A new intrusion detection system using support vector machines and hierarchical clustering. **The VLDB Journal**, Springer-Verlag New York, Inc., Secaucus, NJ, USA, v. 16, n. 4, p. 507–521, 2007. ISSN 1066-8888. Citado na página 100.

KIM, S.-W.; OOMMEN, B. Enhancing prototype reduction schemes with lvq3-type algorithms. **Pattern Recognition**, v. 36, n. 5, p. 1083–1093, 2003. ISSN 0031-3203. Citado na página 34.

KIRKPATRICK, S.; GELATT, C. D.; VECCHI, M. P. Optimization by Simulated Annealing. **Science, Number 4598, 13 May 1983**, v. 220, 4598, p. 671–680, 1983. Citado 4 vezes nas páginas 56, 58, 59 e 77.

KLÖSGEN, W.; ZYTKOW, J. M. Knowledge discovery in database terminology. In: **Advances in Knowledge Discovery and Data Mining**. [S.l.: s.n.], 1996. p. 573–592. Citado na página 34.

KOLLIOS, G.; GUNOPULOS, D.; KOUDAS, N.; BERCHTOLD, S. Efficient biased sampling for approximate clustering and outlier detection in large data sets. **Knowledge and Data Engineering, IEEE Transactions on**, v. 15, n. 5, p. 1170–1187, oct. 2003. ISSN 1041-4347. Citado na página 40.

KULIS, B.; BASU, S.; DHILLON, I.; MOONEY, R. Semi-supervised graph clustering: a kernel approach. In: **Proceedings of the 22nd international conference on Machine learning**. New York, NY, USA: ACM, 2005. (ICML '05), p. 457–464. ISBN 1-59593-180-5. Citado na página 39.

KUNCHEVA, L.; HADJITODOROV, S. Using diversity in cluster ensembles. In: **Systems, Man and Cybernetics, 2004 IEEE International Conference on**. [S.l.: s.n.], 2004. v. 2, p. 1214–1219 vol.2. ISSN 1062-922X. Citado na página 124.

KUNCHEVA, L.; WHITAKER, C. Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. **Machine Learning**, Kluwer Academic Publishers, v. 51, n. 2, p. 181–207, 2003. ISSN 0885-6125. Citado na página 124.

KUNCHEVA, L. I. **Combining Pattern Classifiers: Methods and Algorithms**. [S.l.]: Wiley-Interscience, 2004. ISBN 0471210781. Citado 3 vezes nas páginas 31, 34 e 54.

_____. Classifier ensembles for detecting concept change in streaming data: Overview and perspectives. In: **Proceedings of the 2nd Workshop SUEMA**. [S.l.: s.n.], 2008. v. 2008, p. 5–10. Citado 2 vezes nas páginas 100 e 102.

LAARHOVEN, P. J. M.; AARTS, E. H. L. (Ed.). **Simulated annealing: theory and applications**. Norwell, MA, USA: Kluwer Academic Publishers, 1987. ISBN 9-027-72513-6. Citado na página 59.

LAMPERT, C.; NICKISCH, H.; HARMELING, S. Learning to detect unseen object classes by between-class attribute transfer. In: **Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on**. [S.l.: s.n.], 2009. p. 951–958. ISSN 1063-6919. Citado 3 vezes nas páginas 99, 100 e 102.

_____. Attribute-based classification for zero-shot visual object categorization. **Pattern Analysis and Machine Intelligence, IEEE Transactions on**, v. 36, n. 3, p. 453–465, March 2014. ISSN 0162-8828. Citado 5 vezes nas páginas 35, 78, 79, 102 e 103.

LEE, Y. J.; GRAUMAN, K. Object-Graphs for Context-Aware Visual Category Discovery. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 34, n. 2, p. 346–358, 2012. ISSN 0162-8828. Citado na página 102.

LI, X.; YE, N. A supervised clustering and classification algorithm for mining data with mixed variables. **Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on**, v. 36, n. 2, p. 396–406, 2006. ISSN 1083-4427. Citado na página 40.

LIANG, J.; QIN, A.; SUGANTHAN, P.; BASKAR, S. Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. **Evolutionary Computation, IEEE Transactions on**, v. 10, n. 3, p. 281–295, 2006. ISSN 1089-778X. Citado na página 56.

LIU, B. **Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data**. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006. (Data-Centric Systems and Applications). ISBN 3540378812. Citado na página 82.

_____. **Sentiment analysis and subjectivity**. [S.l.]: Taylor and Francis Group, Boca, 2010. Citado 2 vezes nas páginas 82 e 96.

_____. **Sentiment Analysis and Opinion Mining**. [S.l.]: Morgan & Claypool Publishers, 2012. (Synthesis Lectures on Human Language Technologies). Citado 3 vezes nas páginas 82, 96 e 99.

LIU, J.; LAMPINEN, J. On setting the control parameter of the differential evolution method. In: **Proc. 8th int. conf. soft computing (MENDEL 2002)**. [S.l.: s.n.], 2002. p. 11–18. Citado na página 70.

_____. A fuzzy adaptive differential evolution algorithm. **Soft Comput.**, Springer-Verlag, Berlin, Heidelberg, v. 9, n. 6, p. 448–462, jun. 2005. ISSN 1432-7643. Citado na página 70.

LIU, S.; LI, F.; LI, F.; CHENG, X.; SHEN, H. Adaptive co-training SVM for sentiment classification on tweets. In: **Proceedings of the 22Nd ACM International Conference on Conference on Information Knowledge Management**. New York, NY, USA: ACM, 2013. (CIKM '13), p. 2079–2088. ISBN 978-1-4503-2263-8. Citado na página [89](#).

LIU, S.; ZHU, W.; XU, N.; LI, F.; CHENG, X.-q.; LIU, Y.; WANG, Y. Co-training and visualizing sentiment evolvement for tweet events. In: **Proceedings of the 22Nd International Conference on World Wide Web Companion**. Republic and Canton of Geneva, Switzerland: International World Wide Web Conferences Steering Committee, 2013. (WWW '13 Companion), p. 105–106. ISBN 978-1-4503-2038-2. Citado 2 vezes nas páginas [84](#) e [89](#).

LU, D.; WENG, Q. A survey of image classification methods and techniques for improving classification performance. **Int. J. Remote Sens.**, Taylor & Francis, Inc., Bristol, PA, USA, v. 28, n. 5, p. 823–870, jan. 2007. ISSN 0143-1161. Citado na página [33](#).

LUNA, J. M.; ROMERO, J. R.; ROMERO, C.; VENTURA, S. Reducing gaps in quantitative association rules: A genetic programming free-parameter algorithm. **Integrated Computer-Aided Engineering**, IOS Press, v. 21, n. 4, p. 321–337, 2014. Citado na página [56](#).

LUO, Y.; TAO, D.; XU, C.; XU, C.; LIU, H.; WEN, Y. Multiview vector-valued manifold regularization for multilabel image classification. **Neural Networks and Learning Systems, IEEE Transactions on**, v. 24, n. 5, p. 709–722, May 2013. ISSN 2162-237X. Citado na página [101](#).

MACQUEEN, J. B. Some methods for classification and analysis of multivariate observations. In: CAM, L. M. L.; NEYMAN, J. (Ed.). **Proc. of the fifth Berkeley Symposium on Mathematical Statistics and Probability**. [S.l.]: University of California Press, 1967. v. 1, p. 281–297. Citado 2 vezes nas páginas [30](#) e [40](#).

MADABHUSHI, A.; SHI, J.; ROSEN, M.; TOMASZEWSKI, J. E.; FELDMAN, M. D. Graph embedding to improve supervised classification and novel class detection: application to prostate cancer. In: **Proceedings of the 8th international conference on Medical Image Computing and Computer-Assisted Intervention - Volume Part I**. Berlin, Heidelberg: Springer-Verlag, 2005. (MICCAI'05), p. 729–737. ISBN 3-540-29327-2, 978-3-540-29327-9. Citado na página [100](#).

MALLIPEDDI, R.; SUGANTHAN, P.; PAN, Q.; TASGETIREN, M. Differential evolution algorithm with ensemble of parameters and mutation strategies. **Applied Soft Computing**, v. 11, n. 2, p. 1679 – 1696, 2011. ISSN 1568-4946. The Impact of Soft Computing for the Progress of Artificial Intelligence. Citado na página [56](#).

MAÑDZIUK, J.; SHASTRI, L. Incremental class learning approach and its application to handwritten digit recognition. **Inf. Sci. Inf. Comput. Sci.**, Elsevier Science Inc., New York, NY, USA, v. 141, n. 3-4, p. 193–217, abr. 2002. ISSN 0020-0255. Citado na página [102](#).

MARKOU, M.; SINGH, S. Novelty detection: a review-part 1: statistical approaches. **Signal processing**, Elsevier, v. 83, n. 12, p. 2481–2497, 2003. Citado 2 vezes nas páginas [100](#) e [102](#).

_____. Novelty detection: a review-part 2: neural network based approaches. **Signal processing**, Elsevier, v. 83, n. 12, p. 2499–2521, 2003. Citado 2 vezes nas páginas [100](#) e [102](#).

MASUD, M.; CHEN, Q.; KHAN, L.; AGGARWAL, C.; GAO, J.; HAN, J.; THURASINGHAM, B. Addressing concept-evolution in concept-drifting data streams. In: **Data Mining (ICDM), 2010 IEEE 10th International Conference on**. [S.l.: s.n.], 2010. p. 929–934. ISSN 1550-4786. Citado 4 vezes nas páginas 35, 40, 78 e 79.

MASUD, M.; CHEN, Q.; KHAN, L.; AGGARWAL, C.; GAO, J.; HAN, J.; SRIVASTAVA, A.; OZA, N. Classification and adaptive novel class detection of feature-evolving data streams. **Knowledge and Data Engineering, IEEE Transactions on**, v. 25, n. 7, p. 1484–1497, July 2013. ISSN 1041-4347. Citado 3 vezes nas páginas 40, 100 e 102.

MASUD, M.; GAO, J.; KHAN, L.; HAN, J.; THURASINGHAM, B. Classification and novel class detection in data streams with active mining. In: **Advances in Knowledge Discovery and Data Mining**. [S.l.]: Springer Berlin/Heidelberg, 2010, (Lecture Notes in Computer Science, v. 6119). p. 311–324. Citado 3 vezes nas páginas 35, 40 e 99.

_____. Classification and novel class detection in concept-drifting data streams under time constraints. **Knowledge and Data Engineering, IEEE Transactions on**, v. 23, n. 6, p. 859–874, june 2011. ISSN 1041-4347. Citado 4 vezes nas páginas 35, 99, 100 e 102.

MELVILLE, P.; GRYC, W.; LAWRENCE, R. D. Sentiment analysis of blogs by combining lexical knowledge with text classification. In: **Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining**. New York, NY, USA: ACM, 2009. (KDD '09), p. 1275–1284. ISBN 978-1-60558-495-9. Citado na página 82.

MENÉNDEZ, H. D.; BARRERO, D. F.; CAMACHO, D. A genetic graph-based approach for partitional clustering. **International Journal of Neural Systems**, v. 24, n. 03, p. 1430008, 2014. Citado na página 56.

MIETTINEN, K. (Ed.). **Evolutionary Algorithms in Engineering and Computer Science: Recent Advances in Genetic Algorithms, Evolution Strategies, Evolutionary Programming**, GE. New York, NY, USA: John Wiley & Sons, Inc., 1999. ISBN 0471999024. Citado na página 69.

MILLER, D.; BROWNING, J. A mixture model and em-based algorithm for class discovery, robust classification, and outlier rejection in mixed labeled/unlabeled data sets. **Pattern Analysis and Machine Intelligence, IEEE Transactions on**, v. 25, n. 11, p. 1468 – 1483, nov. 2003. ISSN 0162-8828. Citado na página 40.

MITCHELL, M. **An introduction to genetic algorithms**. Cambridge, MA, USA: MIT Press, 1996. ISBN 0-262-13316-4. Citado 2 vezes nas páginas 29 e 69.

MIURA, Y.; SAKAKI, S.; HATTORI, K.; OHKUMA, T. Teamx: A sentiment analyzer with enhanced lexicon mapping and weighting scheme for unbalanced data. In: **Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)**. Dublin, Ireland: Association for Computational Linguistics and Dublin City University, 2014. p. 628–632. Citado 4 vezes nas páginas 22, 92, 93 e 96.

MOHAMMAD, S.; KIRITCHENKO, S.; ZHU, X. Nrc-canada: Building the state-of-the-art in sentiment analysis of tweets. In: **Proceedings of the seventh international workshop on Semantic Evaluation Exercises (SemEval-2013)**. Atlanta, Georgia, USA: [s.n.], 2013. Citado 5 vezes nas páginas 82, 84, 89, 90 e 96.

MOLINA-GARCÍA, M.; CALLE-SÁNCHEZ, J.; GONZÁLEZ-MERINO, C.; FERNÁNDEZ-DURÁN, A.; ALONSO, J. I. Design of in-building wireless networks deployments using evolutionary algorithms. **Integrated Computer-Aided Engineering**, IOS Press, v. 21, n. 4, p. 367–385, 2014. Citado na página 56.

MONTOYO, A.; MARTÍNEZ-BARCO, P.; BALAHUR, A. Subjectivity and sentiment analysis: An overview of the current state of the area and envisaged developments. **Decision Support Systems**, v. 53, n. 4, p. 675 – 679, 2012. ISSN 0167-9236. Citado na página 82.

NAKOV, P.; ROSENTHAL, S.; KOZAREVA, Z.; STOYANOV, V.; RITTER, A.; WILSON, T. Semeval-2013 task 2: Sentiment analysis in twitter. In: **Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)**. Atlanta, Georgia, USA: Association for Computational Linguistics, 2013. p. 312–320. Citado 7 vezes nas páginas 22, 84, 86, 88, 89, 90 e 92.

NALDI, M. C.; CARVALHO, A.; CAMPELLO, R. J. Cluster ensemble selection based on relative validity indexes. **Data Mining and Knowledge Discovery**, Springer, v. 27, n. 2, p. 259–289, 2013. Citado na página 124.

NENE, S. A.; NAYAR, S. K.; MURASE, H. *et al.* **Columbia Object Image Library (COIL-20)**. [S.l.], 1996. Disponível em: <<http://www.cs.columbia.edu/CAVE/software/softlib/coil-20.php>>. Citado na página 108.

NGUYEN, N.; CARUANA, R. Consensus clusterings. **ICDM '07: Proceedings of the 2007 Seventh IEEE International Conference on Data Mining**, v. 1, 2007. Citado na página 32.

NIGAM, K.; MCCALLUM, A.; THRUN, S.; MITCHELL, T. Text classification from labeled and unlabeled documents using em. **Machine Learning**, Kluwer Academic Publishers, v. 39, n. 2-3, p. 103–134, 2000. ISSN 0885-6125. Citado na página 83.

OWOPUTI, O.; DYER, C.; GIMPEL, K.; SCHNEIDER, N.; SMITH, N. A. Improved part-of-speech tagging for online conversational text with word clusters. In: **In Proceedings of NAACL**. [S.l.: s.n.], 2013. Citado na página 90.

OZA, N. C.; TUMER, K. Classifier ensembles: Select real-world applications. **Information Fusion**, v. 9, n. 1, p. 4–20, 2008. ISSN 1566-2535. Special Issue on Applications of Ensemble Methods. Citado 3 vezes nas páginas 31, 34 e 54.

PALATUCCI, M.; POMERLEAU, D.; HINTON, G. E.; MITCHELL, T. M. Zero-shot learning with semantic output codes. In: BENGIO, Y.; SCHUURMANS, D.; LAFFERTY, J.; WILLIAMS, C.; CULOTTA, A. (Ed.). **Advances in Neural Information Processing Systems 22**. [S.l.]: Curran Associates, Inc., 2009. p. 1410–1418. Citado na página 102.

PAN, S. J.; YANG, Q. A survey on transfer learning. **IEEE Transactions on Knowledge and Data Engineering**, IEEE Computer Society, Los Alamitos, CA, USA, v. 22, p. 1345–1359, 2010. ISSN 1041-4347. Citado na página 35.

PANG, B.; LEE, L. Opinion mining and sentiment analysis. **Foundations and trends in information retrieval**, Now Publishers Inc., v. 2, n. 1-2, p. 1–135, 2008. Citado na página 82.

PANG, B.; LEE, L.; VAITHYANATHAN, S. Thumbs up? sentiment classification using machine learning techniques. In: **Proceedings of EMNLP**. [S.l.: s.n.], 2002. p. 79–86. Citado na página 90.

PAPADIMITRIOU, S.; MAVROUDI, S.; VLADUTU, L.; BEZERIANOS, A. Ischemia detection with a self-organizing map supplemented by supervised learning. **Neural Networks, IEEE Transactions on**, v. 12, n. 3, p. 503–515, maio 2001. ISSN 1045-9227. Citado na página 39.

PEDRINO, E. C.; RODA, V. O.; KATO, E. R. R.; SAITO, J. H.; TRONCO, M. L.; TSUNAKI, R. H.; JR, O. M.; NICOLETTI, M. C. A genetic programming based system for the automatic construction of image filters. **Integrated Computer-Aided Engineering**, IOS Press, v. 20, n. 3, p. 275–287, 2013. Citado na página 56.

PEDRYCZ, W.; VUKOVICH, G. Fuzzy clustering with supervision. **Pattern Recognition**, v. 37, n. 7, p. 1339–1349, 2004. ISSN 0031-3203. Citado na página 39.

PENATTI, O.; VALLE, E.; TORRES, R. Comparative study of global color and texture descriptors for web image retrieval. **Journal of Visual Communication and Image Representation**, v. 23, n. 2, p. 359–380, 2012. Citado na página 109.

PERRONE, M. P.; COOPER, L. N. When networks disagree: Ensemble methods for hybrid neural networks. In: . [S.l.]: Chapman and Hall, 1993. p. 126–142. Citado na página 31.

PONTI, M.; PICON, C. Color description of low resolution images using fast bitwise quantization and border-interior classification. In: IEEE. **Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on**. Brisbane, Australia, 2015. Citado na página 109.

PRICE, K. Differential evolution: a fast and simple numerical optimizer. In: **Fuzzy Information Processing Society, 1996. NAFIPS. 1996 Biennial Conference of the North American**. [S.l.: s.n.], 1996. p. 524–527. Citado 10 vezes nas páginas 21, 57, 58, 69, 70, 71, 72, 77, 78 e 122.

PRICE, K.; STORN, R. M.; LAMPINEN, J. A. **Differential Evolution: A Practical Approach to Global Optimization (Natural Computing Series)**. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2005. ISBN 3540209506. Citado 2 vezes nas páginas 57 e 69.

_____. **Differential evolution: a practical approach to global optimization**. [S.l.]: Springer Science & Business Media, 2006. Citado na página 59.

PRINCE, S. J.; PRINCE, S. J. D. **Computer vision: models, learning, and inference**. [S.l.]: Cambridge University Press, 2012. Citado na página 101.

PUNERA, K.; GHOSH, J. Consensus-based ensembles of soft clusterings. **Appl. Artif. Intell.**, Taylor & Francis, Inc., Bristol, PA, USA, v. 22, p. 780–810, August 2008. ISSN 0883-9514. Citado na página 42.

QIAN, Q.; CHEN, S.; CAI, W. Simultaneous clustering and classification over cluster structure representation. **Pattern Recognition**, v. 45, n. 6, p. 2227–2236, 2012. ISSN 0031-3203. Citado 2 vezes nas páginas 34 e 41.

QIN, A. K.; HUANG, V. L.; SUGANTHAN, P. Differential evolution algorithm with strategy adaptation for global numerical optimization. **Evolutionary Computation, IEEE Transactions on**, v. 13, n. 2, p. 398–417, 2009. ISSN 1089-778X. Citado 6 vezes nas páginas 56, 57, 59, 66, 70 e 74.

QIU, L.; ZHANG, W.; HU, C.; ZHAO, K. Selc: A self-supervised model for sentiment classification. In: **Proceedings of the 18th ACM Conference on Information and Knowledge Management**. New York, NY, USA: ACM, 2009. (CIKM '09), p. 929–936. ISBN 978-1-60558-512-3. Citado 2 vezes nas páginas 84 e 95.

QUINLAN, J. R. **C4.5: programs for machine learning**. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1993. ISBN 1-55860-238-0. Citado na página 29.

RAHMAN, A.; VERMA, B. Cluster-based ensemble of classifiers. **Expert Systems**, v. 30, n. 3, p. 270–282, 2013. ISSN 1468-0394. Citado 2 vezes nas páginas 34 e 39.

RECHENBERG, I. **Evolutionsstrategie: Optimierung Technischer Systeme nach Prinzipien der Biologischen Evolution**. Stuttgart-Bad Cannstatt: Frommann-Holzboog Verlag, 1973. (Problemata, 15). Citado 2 vezes nas páginas 57 e 77.

REYES, O.; MORELL, C.; VENTURA, S. Evolutionary feature weighting to improve the performance of multi-label lazy algorithms. **Integrated Computer-Aided Engineering**, IOS Press, v. 21, n. 4, p. 339–354, 2014. Citado na página 56.

RITTER, A.; CLARK, S.; MAUSAM; ETZIONI, O. Named entity recognition in tweets: An experimental study. In: **Proceedings of the Conference on Empirical Methods in Natural Language Processing**. Stroudsburg, PA, USA: Association for Computational Linguistics, 2011. (EMNLP '11), p. 1524–1534. Citado na página 82.

ROCHA, A.; HAUAGGE, D. C.; WAINER, J.; GOLDENSTEIN, S. Automatic fruit and vegetable classification from images. **Computers and Electronics in Agriculture**, v. 70, n. 1, p. 96 – 104, 2010. ISSN 0168-1699. Citado na página 109.

ROHRBACH, M.; STARK, M.; SCHIELE, B. Evaluating knowledge transfer and zero-shot learning in a large-scale setting. In: **Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on**. [S.l.: s.n.], 2011. p. 1641–1648. ISSN 1063-6919. Citado na página 102.

RONKKONEN, J.; KUKKONEN, S.; PRICE, K. V. Real-parameter optimization with differential evolution. In: **IEEE. Evolutionary Computation, 2005. The 2005 IEEE Congress on**. [S.l.], 2005. v. 1, p. 506–513. Citado 3 vezes nas páginas 58, 70 e 71.

ROSENTHAL, S.; NAKOV, P.; RITTER, A.; STOYANOV, V. SemEval-2014 Task 9: Sentiment Analysis in Twitter. In: NAKOV, P.; ZESCH, T. (Ed.). **Proceedings of the 8th International Workshop on Semantic Evaluation**. Dublin, Ireland: [s.n.], 2014. (SemEval 14). Citado 5 vezes nas páginas 22, 85, 87, 88 e 89.

SCHEIRER, W.; JAIN, L.; BOULT, T. Probability models for open set recognition. **Pattern Analysis and Machine Intelligence, IEEE Transactions on**, v. 36, n. 11, p. 2317–2324, Nov 2014. ISSN 0162-8828. Citado na página 102.

SCHEIRER, W.; ROCHA, A.; MICHEALS, R.; BOULT, T. Robust fusion: Extreme value theory for recognition score normalization. In: **Computer Vision–ECCV 2010**. [S.l.]: Springer, 2010. p. 481–495. Citado na página 124.

SCHEIRER, W.; ROCHA, A. de R.; SAPKOTA, A.; BOULT, T. Toward open set recognition. **Pattern Analysis and Machine Intelligence, IEEE Transactions on**, v. 35, n. 7, p. 1757–1772, July 2013. ISSN 0162-8828. Citado 5 vezes nas páginas 35, 78, 79, 99 e 101.

- SCHEIRER, W. J.; ROCHA, A.; MICHEALS, R. J.; BOULT, T. E. Meta-recognition: The theory and practice of recognition score analysis. **Pattern Analysis and Machine Intelligence, IEEE Transactions on**, IEEE, v. 33, n. 8, p. 1689–1695, 2011. Citado na página [125](#).
- SCHERREIK, M.; RIGLING, B. Ladar ATR via probabilistic open set techniques. In: . [S.l.: s.n.], 2014. v. 9090, p. 90900A–90900A–9. Citado na página [102](#).
- SCHWEFEL, H. P. **Numerical Optimization of Computer Models**. Chichester: John Wiley & Sons, Inc., 1981. Tradução de sua dissertação de 1975. ISBN 0471099880. Citado na página [57](#).
- SCUDDER, H. J. Probability of error of some adaptive pattern-recognition machines. **Information Theory, IEEE Transactions on**, IEEE, v. 11, n. 3, p. 363–371, 1965. Citado na página [84](#).
- SETNES, M.; BABUSKA, R. Fuzzy relational classifier trained by fuzzy clustering. **Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on**, v. 29, n. 5, p. 619–625, 1999. ISSN 1083-4419. Citado na página [40](#).
- _____. _____. **Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on**, v. 29, n. 5, p. 619–625, out. 2002. ISSN 1083-4419. Citado na página [34](#).
- SETTLES, B. Active Learning Literature Survey. In: **Relatório Técnico N. 1648**. University of Wisconsin–Madison: [s.n.], 2009. Citado na página [37](#).
- _____. Active learning. **Synthesis Lectures on Artificial Intelligence and Machine Learning**, Morgan & Claypool Publishers, v. 6, n. 1, p. 1–114, 2012. Citado 4 vezes nas páginas [98](#), [101](#), [102](#) e [122](#).
- SHI, Y.; EBERHART, R. Fuzzy adaptive particle swarm optimization. In: **Evolutionary Computation, 2001. Proceedings of the 2001 Congress on**. [S.l.: s.n.], 2001. v. 1, p. 101–106 vol. 1. Citado na página [56](#).
- SHI, Y.; EBERHART, R. C. Parameter selection in particle swarm optimization. In: **Proceedings of the 7th International Conference on Evolutionary Programming VII**. London, UK: Springer-Verlag, 1998. (EP '98), p. 591–600. ISBN 3-540-64891-7. Citado na página [58](#).
- SILVER, D.; BENNETT, K. Guest editor's introduction: special issue on inductive transfer learning. **Machine Learning**, Springer Netherlands, v. 73, p. 215–220, 2008. ISSN 0885-6125. Citado na página [35](#).
- SINDHWANI, V.; KEERTHI, S. S. Large scale semi-supervised linear svms. In: **ACM. Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval**. [S.l.], 2006. p. 477–484. Citado na página [83](#).
- SMAILOVIĆ, J.; GRČAR, M.; LAVRAČ, N.; ŽNIDARŠIČ, M. Stream-based active learning for sentiment analysis in the financial domain. **Information Sciences**, Elsevier, v. 285, p. 181–203, 2014. Citado na página [98](#).
- SNEATH, P. The application of computers to taxonomy. **Journal of general microbiology**, v. 17, p. 201–226, 1957. Citado na página [30](#).
- SOARES, R. G. F.; CHEN, H.; YAO, X. Semisupervised classification with cluster regularization. **Neural Networks and Learning Systems, IEEE Transactions on**, v. 23, n. 11, p. 1779–1792, 2012. ISSN 2162-237X. Citado 2 vezes nas páginas [34](#) e [39](#).

SOKAL, R. R.; MICHENER, C. D. A statistical method for evaluating systematic relationships. **University of Kansas Scientific Bulletin**, v. 28, p. 1409–1438, 1958. Citado na página [30](#).

SONG, M.; RAJASEKARAN, S. A novel scheme for the detection of an unknown class in classification. In: STAHLBOCK, R.; CRONE, S. F.; LESSMANN, S. (Ed.). **Proceedings of The 2008 International Conference on Data Mining, DMIN 2008, July 14-17, 2008, Las Vegas, USA, 2 Volumes**. [S.l.]: CSREA Press, 2008. p. 10–13. ISBN 1-60132-062-0. Citado 3 vezes nas páginas [35](#), [78](#) e [79](#).

SORENSEN, T. A method of establishing groups of equal amplitude in plant sociology based on similarity of species and its application to analyses of the vegetation on Danish commons. **Biologiske Skrifter**, v. 5, p. 35–43, 1948. Citado na página [30](#).

SPERIOSU, M.; SUDAN, N.; UPADHYAY, S.; BALDRIDGE, J. Twitter polarity classification with label propagation over lexical links and the follower graph. In: **Proceedings of the First Workshop on Unsupervised Learning in NLP**. Stroudsburg, PA, USA: Association for Computational Linguistics, 2011. (EMNLP '11), p. 53–63. ISBN 978-1-937284-13-8. Citado 2 vezes nas páginas [82](#) e [96](#).

STEHLLING, R. O.; NASCIMENTO, M. A.; FALCAO, A. X. A compact and efficient image retrieval approach based on border/interior pixel classification. In: **CIKM'02 - 11th ACM Int. Conf. Information Knowledge Management**. [S.l.: s.n.], 2002. p. 102–109. Citado na página [109](#).

STORN, R. On the usage of differential evolution for function optimization. In: **Fuzzy Information Processing Society, 1996. NAFIPS., 1996 Biennial Conference of the North American**. [S.l.: s.n.], 1996. p. 519–523. Citado 2 vezes nas páginas [58](#) e [70](#).

STORN, R.; PRICE, K. Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces. **Jornal of Global Optimization**, Kluwer Academic Publishers, Hingham, MA, USA, v. 11, n. 4, p. 341–359, 1997. ISSN 0925-5001. Citado 8 vezes nas páginas [21](#), [58](#), [69](#), [70](#), [71](#), [72](#), [78](#) e [122](#).

STREHL, A.; GHOSH, J. Cluster Ensembles – A Knowledge Reuse Framework for Combining Multiple Partitions. **Journal of Machine Learning Research**, v. 3, p. 583–617, 2002. Citado 3 vezes nas páginas [32](#), [33](#) e [34](#).

SZELISKI, R. **Computer vision: algorithms and applications**. [S.l.]: Springer, 2010. Citado na página [101](#).

TABOADA, M.; BROOKE, J.; TOFILOSKI, M.; VOLL, K.; STEDE, M. Lexicon-based methods for sentiment analysis. **Computational linguistics**, MIT Press, v. 37, n. 2, p. 267–307, 2011. Citado na página [82](#).

TANG, Y.; PAN, W.; LI, H.; XU, Y. Fuzzy naive bayes classifier based on fuzzy clustering. In: **Systems, Man and Cybernetics, 2002 IEEE International Conference on**. [S.l.: s.n.], 2002. v. 5, p. 6 pp. vol.5–. ISSN 1062-922X. Citado na página [34](#).

THELWALL, M.; BUCKLEY, K.; PALTOGLOU, G.; CAI, D.; KAPPAS, A. Sentiment in short strength detection informal text. **J. Am. Soc. Inf. Sci. Technol.**, John Wiley & Sons, Inc., New York, NY, USA, v. 61, n. 12, p. 2544–2558, dez. 2010. ISSN 1532-2882. Citado na página [90](#).

TIMM, H. Fuzzy cluster analysis of classified data. In: **IFSA World Congress and 20th NA-FIPS International Conference, 2001. Joint 9th**. [S.l.: s.n.], 2001. v. 3, p. 1431–1436 vol.3. Citado na página 39.

TOPCHY, A.; JAIN, A.; PUNCH, W. A mixture model for clustering ensembles. In: **Proc. SIAM Data Mining**. [S.l.: s.n.], 2004. p. 379–390. Citado na página 33.

_____. Clustering ensembles: models of consensus and weak partitions. **Pattern Analysis and Machine Intelligence, IEEE Transactions on**, v. 27, n. 12, p. 1866–1881, 2005. ISSN 0162-8828. Citado 2 vezes nas páginas 32 e 33.

TORRALBA, A.; EFROS, A. Unbiased look at dataset bias. In: **Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on**. [S.l.: s.n.], 2011. p. 1521–1528. ISSN 1063-6919. Citado na página 99.

TUIA, D.; VOLPI, M.; COPA, L.; KANEVSKI, M.; MUNOZ-MARI, J. A survey of active learning algorithms for supervised remote sensing image classification. **Selected Topics in Signal Processing, IEEE Journal of**, v. 5, n. 3, p. 606–617, June 2011. ISSN 1932-4553. Citado 3 vezes nas páginas 35, 99 e 101.

TURNERY, P. D. Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews. In: **Proceedings of the 40th Annual Meeting on Association for Computational Linguistics**. Stroudsburg, PA, USA: Association for Computational Linguistics, 2002. (ACL 02), p. 417–424. Citado na página 82.

ULLMAN, S. **High-level vision: Object recognition and visual cognition**. [S.l.]: MIT press, 2000. Citado na página 101.

VANIN, A. A.; FREITAS, L. A.; VIEIRA, R.; BOCHERNITSAN, M. Some clues on irony detection in tweets. In: **Proceedings of the 22Nd International Conference on World Wide Web Companion**. Republic and Canton of Geneva, Switzerland: International World Wide Web Conferences Steering Committee, 2013. (WWW '13 Companion), p. 635–636. ISBN 978-1-4503-2038-2. Citado na página 97.

VENDRAMIN, L.; CAMPELLO, R.; COLETTA, L.; HRUSCHKA, E. Distributed fuzzy clustering with automatic detection of the number of clusters. In: **Inter. Symp. on Distrib. Comp and AI DCAI'11**. [S.l.: s.n.], 2011. v. 91, p. 133–140. ISBN 978-3-642-19933-2. Citado na página 33.

VENDRAMIN, L.; CAMPELLO, R. J. G. B.; HRUSCHKA, E. R. Relative clustering validity criteria: A comparative overview. **Statistical Analysis and Data Mining**, v. 3, n. 4, p. 209–235, 2010. Citado na página 33.

VERIKAS, A.; GELZINIS, A.; BACAUSKIENE, M. Mining data with random forests: A survey and results of new tests. **Pattern Recognition**, v. 44, n. 2, p. 330–349, 2011. ISSN 0031-3203. Citado na página 31.

VERMA, B.; RAHMAN, A. Cluster-oriented ensemble classifier: Impact of multicluster characterization on ensemble classifier learning. **Knowledge and Data Engineering, IEEE Transactions on**, v. 24, n. 4, p. 605–618, 2012. ISSN 1041-4347. Citado 2 vezes nas páginas 34 e 39.

WAN, X. Co-training for cross-lingual sentiment classification. In: **Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1 - Volume 1**. Stroudsburg, PA, USA: Association for Computational Linguistics, 2009. (ACL '09), p. 235–243. ISBN 978-1-932432-45-9. Citado na página 84.

WANG, H.; FAN, W.; YU, P. S.; HAN, J. Mining concept-drifting data streams using ensemble classifiers. In: **Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining**. New York, NY, USA: ACM, 2003. p. 226–235. ISBN 1-58113-737-0. Citado na página 100.

WANG, J.; LI, J.; WIEDERHOLD, G. Simplicity: semantics-sensitive integrated matching for picture libraries. **Pattern Analysis and Machine Intelligence, IEEE Transactions on**, v. 23, n. 9, p. 947–963, Sep 2001. ISSN 0162-8828. Citado na página 109.

WANG, W.; ZHOU, Z.-H. A new analysis of co-training. In: **Proceedings of the 27th International Conference on Machine Learning (ICML-10)**. [S.l.: s.n.], 2010. p. 1135–1142. Citado 2 vezes nas páginas 84 e 101.

WANG, Y.; CAI, Z.; ZHANG, Q. Differential evolution with composite trial vector generation strategies and control parameters. **Evolutionary Computation, IEEE Transactions on**, v. 15, n. 1, p. 55–66, Feb 2011. ISSN 1089-778X. Citado 2 vezes nas páginas 56 e 74.

WEL, F. J. Van der; GAAG, L. C. Van der; GORTE, B. G. Visual exploration of uncertainty in remote-sensing classification. **Computers & Geosciences**, Elsevier, v. 24, n. 4, p. 335–343, 1998. Citado na página 103.

WHITLEY, D. An overview of evolutionary algorithms: practical issues and common pitfalls. **Information and Software Technology**, v. 43, n. 14, p. 817 – 831, 2001. ISSN 0950-5849. Citado na página 69.

WILBER, M.; SCHEIRER, W.; LEITNER, P.; HEFLIN, B.; ZOTT, J.; REINKE, D.; DELANEY, D.; BOULT, T. Animal recognition in the mojave desert: Vision tools for field biologists. In: **Applications of Computer Vision (WACV), 2013 IEEE Workshop on**. [S.l.: s.n.], 2013. p. 206–213. ISSN 1550-5790. Citado na página 102.

WINDEATT, T. Diversity measures for multiple classifier system analysis and design. **Information Fusion**, v. 6, n. 1, p. 21 – 36, 2005. ISSN 1566-2535. Diversity in Multiple Classifier Systems. Citado na página 124.

WITTEN, I. H.; FRANK, E. **Data mining: practical machine learning tools and techniques**. 2. ed. [S.l.]: Morgan Kaufmann, 2005. (Morgan Kaufmann Series in Data Management Systems). ISBN 0-12-088407-0. Citado 8 vezes nas páginas 29, 30, 34, 49, 52, 61, 62 e 108.

WOLPERT, D. H. Stacked generalization. **Neural Networks**, v. 5, p. 241–259, 1992. Citado na página 31.

WOLPERT, D. H.; MACREADY, W. G. **No free lunch theorems for search**. 1995. Relatório Técnico SFI-TR-95-02-010, Santa Fe Institute. Citado na página 30.

XIANG, B.; ZHOU, L. Improving twitter sentiment analysis with topic-based mixture modeling and semi-supervised training. In: **Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)**. Baltimore, Maryland: Association for Computational Linguistics, 2014. p. 434–439. Citado na página 34.

- _____. _____. In: **Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)**. Baltimore, Maryland: Association for Computational Linguistics, 2014. p. 434–439. Citado 2 vezes nas páginas 89 e 91.
- XU, J.-M.; JUN, K.-S.; ZHU, X.; BELLMORE, A. Learning from bullying traces in social media. In: **HLT-NAACL**. [S.l.]: The Association for Computational Linguistics, 2012. p. 656–666. ISBN 978-1-937284-20-6. Citado na página 82.
- XU, R.; WUNSCH, D. **Clustering**. [S.l.]: Wiley-IEEE Press, 2009. ISBN 9780470276808. Citado 2 vezes nas páginas 30 e 31.
- XU, Z.; JIN, R.; ZHU, J.; KING, I.; LYU, M. Efficient convex relaxation for transductive support vector machine. In: **Advances in neural information processing systems**. [S.l.: s.n.], 2008. p. 1641–1648. Citado na página 83.
- YANG, X.-S. **Nature-inspired metaheuristic algorithms**. 2nd. ed. [S.l.]: Luniver Press, 2010. ISBN 1905986289, 9781905986286. Citado 2 vezes nas páginas 55 e 56.
- YANG, Z. R. A novel radial basis function neural network for discriminant analysis. **Neural Networks, IEEE Transactions on**, v. 17, n. 3, p. 604–612, May 2006. ISSN 1045-9227. Citado na página 34.
- YONG, Q.; ZONG-YI, X.; LI-MIN, J.; YING-YING, W. Study on interpretable fuzzy classification system based on neural networks. In: **ICCAS-SICE, 2009**. [S.l.: s.n.], 2009. p. 5318–5321. Citado na página 40.
- YONG, R.; NOBUHIRO, K.; YOSHINAGA, N.; KITSUREGAWA, M. Sentiment classification in under-resourced languages using graph-based semi-supervised learning methods. **IEICE TRANSACTIONS on Information and Systems**, The Institute of Electronics, Information and Communication Engineers, v. 97, n. 4, p. 790–797, 2014. Citado na página 83.
- YU, X.; GEN, M. **Introduction to evolutionary algorithms**. [S.l.]: Springer Science & Business Media, 2010. Citado 2 vezes nas páginas 69 e 102.
- ZENG, Z.; XU, J.; WU, S.; SHEN, M. Antithetic method-based particle swarm optimization for a queuing network problem with fuzzy data in concrete transportation systems. **Computer-Aided Civil and Infrastructure Engineering**, Wiley Online Library, v. 29, n. 10, p. 771–800, 2014. Citado na página 56.
- ZHANG, L.; GHOSH, R.; DEKHIL, M.; HSU, M.; LIU, B. Combining lexicon-based and learning-based methods for twitter sentiment analysis. In: . [S.l.]: HP Laboratories, 2011. Technical Report. Citado 2 vezes nas páginas 82 e 96.
- ZHANG, S.; NEAGU, D.; BALESCU, C. Refinement of clustering solutions using a multi-label voting algorithm for neuro-fuzzy ensembles. In: WANG, L.; CHEN, K.; ONG, Y. (Ed.). **Advances in Natural Computation**. [S.l.]: Springer Berlin Heidelberg, 2005, (Lecture Notes in Computer Science, v. 3612). p. 1300–1303. ISBN 978-3-540-28320-1. Citado na página 40.
- ZHAO, J.; LAN, M.; ZHU, T. Ecnu: Expression- and message-level sentiment orientation classification in twitter using multiple effective features. In: **Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)**. Dublin, Ireland: Association for Computational Linguistics and Dublin City University, 2014. p. 259–264. Citado 2 vezes nas páginas 89 e 91.

ZHOU, D.; BOUSQUET, O.; LAL, T. N.; WESTON, J.; SCHÖLKOPF, B. Learning with local and global consistency. **Advances in neural information processing systems**, v. 16, n. 16, p. 321–328, 2004. Citado na página [83](#).

ZHOU, X.; YU, K.; ZHANG, T.; HUANG, T. Image classification using super-vector coding of local image descriptors. In: DANIILIDIS, K.; MARAGOS, P.; PARAGIOS, N. (Ed.). **Computer Vision - ECCV 2010**. [S.l.]: Springer Berlin Heidelberg, 2010, (Lecture Notes in Computer Science, v. 6315). p. 141–154. ISBN 978-3-642-15554-3. Citado na página [101](#).

ZHOU, Z.-H.; LI, M. Tri-training: Exploiting unlabeled data using three classifiers. **Knowledge and Data Engineering, IEEE Transactions on**, IEEE, v. 17, n. 11, p. 1529–1541, 2005. Citado na página [84](#).

ZHU, A.-X. Measuring uncertainty in class assignment for natural resource maps under fuzzy logic. **Photogrammetric engineering and remote sensing**, [Falls Church, Va.] American Society of Photogrammetry., v. 63, n. 10, p. 1195–1201, 1997. Citado na página [103](#).

ZHU, X. **Semi-supervised Learning with Graphs**. Tese (Doutorado), Pittsburgh, PA, USA, 2005. AAI3179046. Citado na página [83](#).

ZHU, X. **Semi-Supervised Learning Literature Survey**. 2008. Citado na página [39](#).

ZHU, X.; GHAMRANI, Z.; LAFFERTY, J. *et al.* Semi-supervised learning using gaussian fields and harmonic functions. In: **ICML**. [S.l.: s.n.], 2003. v. 3, p. 912–919. Citado na página [83](#).

ZHU, X.; GOLDBERG, A. B. **Introduction to Semi-Supervised Learning**. [S.l.]: Morgan & Claypool Publishers, 2009. (Synthesis Lectures on Artificial Intelligence and Machine Learning). Citado 7 vezes nas páginas [37](#), [81](#), [82](#), [84](#), [85](#), [95](#) e [122](#).

ZHU, X.; KIRITCHENKO, S.; MOHAMMAD, S. Nrc-canada-2014: Recent improvements in the sentiment analysis of tweets. In: **Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)**. Dublin, Ireland: Association for Computational Linguistics and Dublin City University, 2014. p. 443–447. Citado na página [96](#).