Classificação de fluxo de dados não estacionários com aplicação em sensores identificadores de insetos

Vinícius Mourão Alves de Souza

Vinícius Mourão Alves de Souza

Classificação de fluxo de dados não estacionários com aplicação em sensores identificadores de insetos

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. Gustavo Enrique de Almeida Prado Alves Batista
Coorientador: Prof. Dr. João Manuel Portela da Gama

USP – São Carlos
Junho 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)

Souza, Vinícius Mourão Alves de
S719c Classificação de fluxo de dados não estacionários com aplicação em sensores identificadores de insetos / Vinícius Mourão Alves de Souza; orientador Gustavo Enrique de Almeida Prado Alves Batista; co-orientador João Manuel Portela da Gama. -- São Carlos, 2016.
208 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. Fluxo de dados. I. Batista, Gustavo Enrique de Almeida Prado Alves, orient. II. Gama, João Manuel Portela da, co-orient. III. Título.

Vinícius Mourão Alves de Souza

Classification of non-stationary data streams with application in sensors for insect identification

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. Gustavo Enrique de Almeida Prado Alves Batista
Coadvisor: Prof. Dr. João Manuel Portela da Gama

USP – São Carlos
June 2016

Agradecimentos

Algumas poucas decisões importantes da minha vida eu tomei sem pensar muito sobre as possíveis consequências. Sair de Maringá para fazer doutorado na USP em São Carlos, foi uma dessas decisões. Talvez por isso eu não tenha imaginado nem mesmo um décimo de todas as coisas que aconteceram comigo ao longo destes anos. Como ninguém constrói, conquista ou vive bons momentos sozinho, sou grato ao incentivo, apoio, compreensão e amizade de muitas pessoas que estiveram presentes durante o desenvolvimento deste projeto. Nesta seção, busco agradecer a estas pessoas.

Ao amor mútuo e incentivo dos meus pais Nilson Evelázio de Souza e Ângela Maria Alves Souza, que agora fecham um ciclo na formação acadêmica de seus três filhos com o título de Doutor. Também agradeço aos meus irmãos Rafael e Nilson, pela inspiração e amizade. Ao longo de toda a minha formação pessoal e profissional, estas quatro pessoas foram os principais modelos em que busquei me espelhar, ainda que adaptações sejam sempre necessárias ao longo do tempo. Afinal, viver nada mais é do que resistir e se adaptar.

À Thainara Granero de Melo por toda a sua compreensão, paciência, incentivo, carinho e amor. Foram tantas fases nesses anos, desde os mil quilômetros na estrada a cada duas semanas para ficarmos juntos somente dois dias, a sua mudança para São Carlos, a distância em linha reta de mais de oito mil quilômetros que nos distanciou por vários meses durante o meu estágio no exterior, até as minhas eternas dúvidas, inquietações, questionamentos e paúra sobre o meu futuro profissional que esquentaram os seus ouvidos nos últimos meses de finalização desta tese e atestam a sua imensurável paciência, compreensão e parceria. Os meus agradecimentos também se estendem ao acolhimento, valorização e apoio de seus pais Israel e Vera e seus irmãos Thalita e Thiago.

Ao orientador deste trabalho, Prof. Gustavo Batista. Seus ensinamentos valiosos não só em aprendizado de máquina e mineração de dados, mas profissionais e éticos, eu levarei por toda a minha vida e buscarei propagar para as pessoas ao meu redor. Agradeço por sua confiança em meu trabalho, pela oportunidade de desenvolver esta tese sob a sua exemplar orientação, supervisão e experiência, e por ter me concedido tantas oportunidades.

Ao Prof. João Gama, pela coorientação deste trabalho e oportunidade de realizar meu estágio de pesquisa na Universidade do Porto sob a sua experiente supervisão. Aos colegas do LIAAD, Aljaž Osojnik, Conceição Rocha, Fábio Pinto, João Duarte, Márcia Oliveira, Pedro Abreu, Vânia Almeida e Vitor Cerqueira. Em especial, aos amigos César Byron Guevara Maldonado e Wilson Soto, essenciais pela continuidade deste trabalho em tempos nebulosos.

À Profa. Solange Rezende, por gentilmente me indicar ao Prof. Gustavo e pela supervisão inicial nos meus primeiros meses, enquanto ele ainda estava em Riverside. Também agradeço à Profa. Maria Carolina Monard, que junto com a Profa. Solange e Prof. Gustavo, conseguem propiciar um ambiente como o LABIC.

À Profa. Margareth Capurro e à Dra. Helena Rocha, do Instituto de Ciências Biomédicas da Universidade de São Paulo, que gentilmente cederam algumas centenas de espécies de *Aedes* e *Culex*, tão importantes para este trabalho.

Tudo seria mais difícil se não fossem os colegas e amigos do LABIC, que propiciaram momentos de descontração e também de discussões sobre diversas questões técnicas. Agradeço a Antonio Parmezan, Camila Sundermann, Celso de Sousa, Denis Reis, Jorge Valverde, Marcos Domingues, Merley Conrado, Newton Spolaôr, Pedro Shiguihara, Rafael Rossi, Rafael Giusti, Renan Pádua, Roberta Sinoara e Thiago Faleiros. Em especial, aos amigos Everton Cherman e Chris Tibes, Bruno Nogueira e Vanessa Borges, Igor Braga e Bárbara Rauber, André Maletzke e Bárbara Nadai. Também agradeço Diego Furtado Silva, por todas as colaborações, contribuições para esta tese e discussões nos mais variados problemas de pesquisa desde o início deste trabalho. Espero ter contribuído de alguma maneira com a sua formação, assim como você certamente contribuiu com a minha.

Por fim, agradeço à Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP), pelo financiamento deste projeto de pesquisa por meio do Processo 2011/17698-5 (Doutorado) e 2013/23037-7 (BEPE) e à Universidade de São Paulo e Instituto de Ciências Matemáticas e de Computação por permitirem a realização deste trabalho.

Resumo

Diversas aplicações são responsáveis por gerar dados ao longo do tempo de maneira contínua, ordenada e ininterrupta em um ambiente dinâmico, denominados fluxo de dados. Entre possíveis tarefas que podem ser realizadas com estes dados, classificação é uma das mais proeminentes. Devido à natureza não estacionária do ambiente responsável por gerar os dados, as características que descrevem os conceitos das classes do problema de classificação podem se alterar ao longo do tempo. Por isso, classificadores de fluxo de dados requerem constantes atualizações em seus modelos para que a taxa de acerto se mantenha estável ao longo do tempo. Na etapa de atualização, a maior parte das abordagens considera que, após a predição de cada exemplo, o seu rótulo correto é imediatamente disponibilizado sem qualquer atraso de tempo (latência nula). Devido aos altos custos do processo de rotulação, os rótulos corretos nem sempre podem ser obtidos para a maior parte dos dados ou são obtidos após um considerável atraso de tempo. No caso mais desafiador, encontram-se as aplicações em que após a etapa de classificação dos exemplos, os seus respectivos rótulos corretos nunca são disponibilizados para o algoritmo, caso chamado de latência extrema. Neste cenário, não é possível o uso de abordagens tradicionais, sendo necessário o desenvolvimento de novos métodos que sejam capazes de manter um modelo de classificação atualizado mesmo na ausência de dados rotulados. Nesta tese, além de discutir o problema de latência na tarefa de classificação de fluxo de dados não estacionários, negligenciado por boa parte da literatura, também são propostos dois algoritmos denominados SCARGC e MClassification para o cenário de latência extrema. Ambas as propostas se baseiam no uso de técnicas de agrupamento para a adaptação à mudanças de maneira não supervisionada. Os algoritmos propostos são intuitivos, simples e apresentam resultados superiores ou equivalentes a outros algoritmos da literatura em avaliações com dados sintéticos e reais, tanto em termos de acurácia de classificação como em tempo computacional. Além de buscar o avanço no estado-da-arte na área de aprendizado em fluxo de dados, este trabalho também apresenta contribuições para uma importante aplicação tecnológica com impacto social e na saúde pública. Especificamente, explorou-se um sensor óptico para a identificação automática de espécies de

insetos a partir da análise de informações provenientes do batimento de asas dos insetos. Para a descrição dos dados, foi verificado que os coeficientes Mel-cepstrais apresentaram os melhores resultados entre as diferentes técnicas de processamento digital de sinais avaliadas. Este sensor é um exemplo concreto de aplicação responsável por gerar um fluxo de dados em que é necessário realizar classificações em tempo real. Durante a etapa de classificação, este sensor exige a adaptação a possíveis variações em condições ambientais, responsáveis por alterar o comportamento dos insetos ao longo do tempo. Para lidar com este problema, é proposto um Sistema com Múltiplos Classificadores que realiza a seleção dinâmica do classificador mais adequado de acordo com características de cada exemplo de teste. Em avaliações com mudanças pouco significativas nas condições ambientais, foi possível obter uma acurácia de classificação próxima de 90%, no cenário com múltiplas classes e, cerca de 95% para a identificação da espécie *Aedes aegypti*, considerando o treinamento com uma única classe. No cenário com mudanças significativas nos dados, foi possível obter 91% de acurácia em um problema com 5 classes e 96% para a classificação de insetos vetores de importantes doenças como dengue e zika vírus.

Palavras-chave: Fluxo de dados, classificação, latência, identificação automática de insetos, sensor óptico.

Abstract

Many applications are able to generate data continuously over time in an ordered and uninterrupted way in a dynamic environment, called data streams. Among possible tasks that can be performed with these data, classification is one of the most prominent. Due to non-stationarity of the environment that generates the data, the features that describe the concepts of the classes can change over time. Thus, the classifiers that deal with data streams require constants updates in their classification models to maintain a stable accuracy over time. In the update phase, most of the approaches assume that after the classification of each example from the stream, their actual class label is available without any time delay (zero latency). Given the high label costs, it is more reasonable to consider that this delay could vary for the most portion of the data. In the more challenging case, there are applications with extreme latency, wherein after the classification of the examples, their actual class labels are never available to the algorithm. In this scenario, it is not possible to use traditional approaches. Thus, there is the need of new methods that are able to maintain a classification model updated in the absence of labeled data. In this thesis, besides to discuss the problem of latency to obtain actual labels in data stream classification problems, neglected by most of the works, we also propose two new algorithms to deal with extreme latency, called SCARGC and MClassification. Both algorithms are based on the use of clustering approaches to adapt to changes in an unsupervised way. The proposed algorithms are intuitive, simple and showed superior or equivalent results in terms of accuracy and computation time compared to other approaches from literature in an evaluation on synthetic and real data. In addition to the advance in the state-of-the-art in the stream learning area, this thesis also presents contributions to an important technological application with social and public health impacts. Specifically, it was studied an optical sensor to automatically identify insect species by the means of the analysis of information coming from wing beat of insects. To describe the data, we conclude that the Mel-cepstral coefficients guide to the best results among different evaluated digital signal processing techniques. This sensor is a concrete example of an application that generates a data stream for which it is necessary to perform real-time classification. During the clas-

sification phase, this sensor must adapt their classification model to possible variations in environmental conditions, responsible for changing the behavior of insects. To address this problem, we propose a System with Multiple Classifiers that dynamically selects the most adequate classifier according to characteristics of each test example. In evaluations with minor changes in the environmental conditions, we achieved a classification accuracy close to 90% in a scenario with multiple classes and 95% when identifying *Aedes aegypti* species considering the training phase with only the positive class. In the scenario with considerable changes in the environmental conditions, we achieved 91% of accuracy considering 5 species and 96% to classify vector mosquitoes of important diseases as dengue and zika virus.

Keywords: Data streams, classification, latency, automatic identification of insects, optical sensor.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Algoritmos

Lista de Abreviaturas e Siglas

1CDT	One Class Diagonal Translation
1CHT	One Class Horizontal Translation
1CSurr	One Class Surrounding another Class
1NN	One-Nearest Neighbor
2CDT	Two Classes Diagonal Translation
2CHT	Two Classes Horizontal Translation
4CE1CF	Four Classes Expanding and One Class Fixed
4CR	Four Classes Rotating Separated
4CRE-V1	Four Classes Rotating with Expansion Version 1
4CRE-V2	Four Classes Rotating with Expansion Version 2
5CVT	Five Classes Vertical Translation
ADWIN	Adaptive Windowing
AP	Average Precision
APT	Arbitrary Sub-Populations Tracker
AUC	Area under Curve
C1C	Classificação com uma Classe
CE	Complexidade Estimada
CENS	Chroma Energy Normalized Statistics
CID	Complexity Invariant Distance
COMPOSE	Compacted Object Sample Extraction
CK-1	Campana-Keogh Distance
CVFT	Concept-adapting Very Fast Decision Trees
ED	Euclidean Distance
DDM	Drift Detection Method
DDtools	Data Description Toolbox
DTW	Dynamic Time Warping
EDDM	Early Drift Detection Method
FG-2C-2D	Two Bidimensional Classes as Four Gaussians
gausdd	Gaussian target distribution
GEARS-2C-2D	Two Rotating Gears
GFHF	Gaussian Function Harmonic Field
GLCM	Grey Level Co-occurrence Matrix
GMM	Gaussian Mixture Models

Hz	Hertz
KDE	Kernel Density Estimation
kmeansdd	k-Means data description
kNN	k-Nearest Neighbors
knndd	k-nearest neighbor data description
LBP	Local Binary Pattern
LFC	Linear Frequency Cepstral Coefficients
LLFC	Log Linear Frequency Cepstral Coefficients
LLGC	Learning with Local and Global Consistency
LN	Latência nula
LPC	Linear Predictive Coding Coefficients
lpdd	Linear programming data description
LSF	Line Spectral Frequencies
MAP	Mean Average Precision
MC	Micro-Cluster
MClassification	Micro-Cluster Classification
MFCC	Mel-Frequency Cepstral Coefficients
MG-2C-2D	Two Bidimensional Mulitimodal Gaussian Classes
mogdd	Mixture of Gaussians data description
mstdd	Minimum spanning tree data description
NB	Naive Bayes
NOAA	U.S. National Oceanic and Atmospheric Administration
parzendd	Parzen density estimator
PID	Proporcional Integral Derivativo
PHT	Page-Hinkley Test
RBF	Radial Basis Function
RF	Random Forest
RMS	Root Mean Square
RP	Recurrence Plots
RPCD	Recurrence Patterns Compression Distance
ROC	Receiver Operating Characteristic
SCARGC	Stream Classification Algorithm Guided by Clustering
SFTA	Segmentation-based Fractal Texture Analysis
SMC	Sistemas com Múltiplos Classificadores
STE	Short-Time Energy
svdd	Support vector data description
SVM	Support Vector Machine
TDF	Transformada Discreta de Fourier
TFRP	Texture Features from Recurrence Patterns
UG-2C-2D	Two Bidimensional Unimodal Gaussian Classes
UG-2C-3D	Two 3-dimensional Unimodal Gaussian Classes
UG-2C-5D	Two 5-dimensional Unimodal Gaussian Classes
VFDT	Very Fast Decision Trees
ZCR	Zero Crossing Rate

Capítulo

1

Introdução

1.1 Considerações Iniciais

Fluxo de dados (em inglês, *data streams*) são dados de tamanho potencialmente ilimitado que não permitem o seu armazenamento, gerados continuamente ao longo do tempo em um ambiente dinâmico responsável por alterar as suas características (Gama, 2010). Aplicações reais que geram ou processam fluxo de dados estão cada dia mais populares devido ao crescente uso de sensores e dispositivos portáteis. Por exemplo, pessoas com *smartphones* são capazes de produzir diariamente uma quantidade massiva de dados contínuos que podem ser minerados por algoritmos de aprendizado de máquina para inúmeros fins. Alguns exemplos de aplicações que lidam com fluxo de dados são a análise de tráfico de rede, mineração de dados em portais de notícias, detecção de fraudes em transações bancárias ou em cartões de crédito, monitoramento de áudio e vídeo para fins de segurança, monitoramento de processos industriais, entre outros.

Entre diferentes possíveis tarefas que podem ser realizadas com estes dados, classificação é uma das mais proeminentes e o principal foco de estudo desta tese. Nesta tarefa, os exemplos chegam para um classificador ao longo do tempo de maneira ordenada, uma instância por vez, não necessariamente em intervalos de tempo igualmente espaçados e o objetivo do classificador é predizer um rótulo de classe para cada exemplo recebido em tempo real ou em tempo hábil antes da chegada do próximo exemplo do fluxo. Devido à natureza não estacionária do ambiente responsável por gerar os dados, as características dos dados que descrevem os conceitos das classes do problema podem se alterar ao longo tempo de diferentes maneiras e por diferentes motivos. Este problema é denominado na área de aprendizado de máquina como *mudanças de conceito* ou *concept drifts* (Widmer

e Kubat, 1996). Por exemplo, em um problema de classificação das preferências de um usuário, é bastante provável que tais preferências se alterem ao longo do tempo, seja devido ao surgimento de novos produtos, alterações no comportamento do usuário ou fatores econômicos. Desse modo, classificadores de fluxo de dados requerem constantes atualizações em seus modelos de classificação para que a taxa de acerto se mantenha estável ao longo do tempo em ambientes dinâmicos.

Na etapa de atualização dos modelos de classificação frente às mudanças de conceito, diferentes abordagens podem ser utilizadas. Entre as mais populares, pode-se destacar (Sebastião e Gama, 2009): *i)* abordagens que adaptam o modelo de classificação periodicamente em intervalos regulares de tempo a partir da incorporação de novos dados e o descarte de dados desatualizados sem considerar se a mudança de fato ocorreu e *ii)* abordagens que primeiramente detectam a mudança nos dados a partir do monitoramento de indicadores de desempenho do classificador e, logo após, adaptam o modelo de classificação com a incorporação de novos dados.

Independente da abordagem, ambas consideram que após o processamento de cada exemplo do fluxo de dados, o seu rótulo correto é imediatamente disponibilizado pelo ambiente ou processo responsável pela geração dos dados, sem qualquer atraso de tempo antes da chegada do próximo exemplo. O atraso de tempo entre a classificação de um exemplo por um algoritmo de aprendizado de máquina e a disponibilização de seu respectivo rótulo correto pelo ambiente é referido neste trabalho como *latência* (Marrs et al., 2010). Assim, ao considerar o valor de latência como nulo, é possível monitorar em tempo real indicadores de desempenho do classificador para a identificação de possíveis mudanças e reconstruir o modelo de classificação a partir de dados rotulados recebidos recentemente.

Entretanto, devido a inúmeros fatores como altos custos envolvidos no processo de rotulação, dependência de um especialista para a rotulação, falhas no processo de transmissão dos rótulos, alta taxa de chegada de dados para a requisição de seus rótulos ou mesmo devido às características do processo gerador dos dados, as informações a respeito dos rótulos dos exemplos nem sempre podem ser obtidas para a maior parte dos dados e, em alguns casos, são obtidas após um considerável atraso de tempo em uma série de aplicações reais. No caso mais difícil, encontram-se as aplicações com latência extrema, em que após a etapa de classificação dos exemplos do fluxo, os seus respectivos rótulos corretos nunca são disponibilizados para o algoritmo. Desse modo, neste cenário é impossível monitorar indicadores de desempenho do classificador para a detecção de mudanças ou incorporar dados rotulados recentemente recebidos para a atualização do modelo de classificação, devido à ausência de dados rotulados. Por isso, este cenário exige o desenvolvimento de novos métodos que sejam capazes de manter um modelo de classificação atualizado, mesmo na ausência de dados rotulados.

Embora presente em diferentes aplicações reais, o cenário de latência extrema tem recebido pouca atenção da comunidade de aprendizado em fluxo de dados nos últimos anos,

a qual tipicamente considera um cenário mais otimista. Somente recentemente, o problema de classificação de fluxo de dados não estacionários sob condições de latência extrema foi apontado como um dos principais desafios da área ainda em aberto (Krempl et al., 2014). Conforme será discutido ao longo desta tese, o presente trabalho visa contribuir com soluções nesta direção.

1.2 Motivação

Um exemplo concreto de aplicação responsável por gerar um fluxo de dados não estacionários em que é necessário realizar a tarefa de classificação em tempo real ao longo do tempo, é o sensor identificador de insetos explorado nesta tese. Este sensor utiliza um feixe laser e técnicas de aprendizado de máquina para classificar insetos em suas espécies com base em informações relacionadas às batidas de asas dos insetos que cruzam a luz.

Durante a etapa de classificação, este sensor precisa se adaptar a possíveis variações em condições ambientais, responsáveis por alterar os dados medidos ao longo do tempo. Variações climáticas afetam o comportamento de insetos, uma vez que seu metabolismo é influenciado pela temperatura (Chadwick, 1939; Taylor, 1963; Farnworth, 1972), bem como pela pressão atmosférica (Chadwick e Williams, 1949) e umidade (Mellanby, 1936). Desse modo, esta aplicação exige o uso de métodos capazes de se adaptar a estas mudanças para que o desempenho da tarefa de classificação não apresente degradações ao longo do tempo, devido a variações climáticas. Além disso, não é possível obter o rótulo correto de cada exemplo imediatamente após o seu processamento, como assumido pela maioria dos trabalhos da literatura. Assim, tem-se um problema de classificação de fluxo de dados não estacionários com latência extrema para a obtenção dos rótulos corretos dos dados processados.

Tal sensor é um importante passo para o desenvolvimento de armadilhas inteligentes capazes de atrair e capturar seletivamente espécies de insetos de interesse, rejeitando a captura das demais espécies. Existem diferentes aplicações para armadilhas inteligentes. Em saúde pública, a armadilha pode capturar mosquitos como os pertencentes aos gêneros *Aedes* e *Anopheles*, vetores da dengue e da malária, respectivamente. Na agricultura, a armadilha pode capturar pragas que causam prejuízos como o psilídeo *Diaphorina citri*, vetor das bactérias que causam o *Greening* (Huanglongbing, HLB ou Amarelão), a mais destrutiva doença dos citros (Bové, 2006). Ao mesmo tempo, a armadilha também permite libertar outras espécies de insetos que não são pragas ou vetores de doenças, restringindo o impacto de sua presença no meio-ambiente. Essa é uma importante característica, uma vez que a maior parte dos insetos como as abelhas, vespas, mariposas e borboletas, possui um relevante papel para o equilíbrio ecológico. Estes insetos são fontes de alimentos para outras espécies animais, auxiliam na reprodução de espécies vegetais e na produção agrícola, ao realizarem o processo de polinização e dispersão de sementes, além de serem

responsáveis pela produção de substâncias úteis para o homem como o mel, a cera, a laca e a seda (Scudder, 2009; Waldbauer, 2009). Assim, deseja-se que a armadilha inteligente retire apenas as espécies maléficas de insetos do meio-ambiente.

Devido ao impacto ecônomico e social no Brasil causado pelo mosquito transmissor da dengue, este trabalho tem um interesse especial na classificação da espécie *Aedes aegypti*. A dengue é a mais importante doença viral transmitida por mosquitos (Gubler, 2012). Uma estimativa de 2013, indica 390 milhões de infecções por ano em todo o mundo, das quais 96 milhões com manifestações clínicas (Bhatt et al., 2013). Embora as taxas de mortalidade têm se mantido baixas, as epidemias de dengue possuem um importante impacto econômico, social e de saúde pública nas populações em que ocorrem (Gubler et al., 2014). Por exemplo, as perdas relacionadas com a dengue no Brasil são estimadas em 1,35 bilhão de dólares por ano, em custos médicos e não-médicos, bem como custos indiretos associados a faltas no trabalho (Shepard et al., 2011). O número de casos de dengue está crescendo de maneira alarmante em diversas regiões do mundo. Antes de 1970, somente 9 países tinham presenciado epidemias. Atualmente, a doença é endêmica em mais de 100 países (W.H.O., 2015). Regiões na América, Sudeste Asiático e Pacífico Ocidental são as mais seriamente afetadas. Casos de dengue nessas áreas excederam 1,2 milhão em 2008 e 3 milhões em 2013. Recentemente o número de casos reportados tem aumentado. Em 2013, foram reportados 2,35 milhões de casos de dengue somente nas Américas, dos quais 37.687 foram de dengue hemorrágica. No Brasil, o Ministério da Saúde registrou mais de 1,6 milhão de casos de dengue em 2015, um aumento de 178% em relação a 2014 (Ministério-Saúde, 2016a).

Além da dengue, o mosquito *Aedes aegypti* é também responsável pela transmissão da febre Chikungunya e do vírus Zika, ambas introduzidas no Brasil em 2014 (Vasconcelos, 2015). Embora com sintomas mais brandos do que a dengue, o Ministério da Saúde do governo brasileiro confirmou, em 2015, a relação entre o vírus Zika e a microcefalia (Ministério-Saúde, 2015), uma infecção que provoca má-formação congênita do cérebro de recém-nascidos, impedindo que o mesmo se desenvolva de maneira adequada. A malformação congênita é responsável por 20% das mortes infantis, respondendo por cerca de 8 mil mortes por ano. De acordo com o Ministério da Saúde (Ministério-Saúde, 2016b), até a semana epidemiológica 01/2016, foram registrados 3.530 casos de microcefalia com suspeita de infecção pelo vírus Zika, distribuídos em 21 estados e 724 municípios brasileiros. Destes casos, estão em investigação 46 casos que evoluíram para óbito. Este número representa um incremento de 8,5 vezes o número de casos de microcefalia em relação ao ano anterior (Ministério-Saúde, 2015).

Assim, além de buscar o avanço no estado-da-arte na área de aprendizado em fluxo de dados, este trabalho também visa contribuir com uma importante e promissora aplicação tecnológica para um problema social e de saúde pública para o Brasil e outros países, responsável pelo controle de insetos como os da espécie *Aedes aegypti*.

1.3 Objetivos e Hipóteses

O principal objetivo deste trabalho foi investigar, propor e avaliar métodos de classificação de fluxo de dados com natureza não estacionária, de modo que o classificador seja capaz de se adaptar às mudanças de conceitos a partir do fornecimento de um conjunto inicial de exemplos rotulados para a fase de treinamento.

Na fase de testes, não é considerado que os rótulos corretos estarão disponíveis após ser realizada a classificação (latência extrema), uma suposição comum entre os sistemas de classificação de fluxo de dados supervisionados (Masud et al., 2008). Tais métodos devem ser eficientes em termos de uso de memória e processamento.

Dado o objetivo principal deste trabalho, pode-se declarar a hipótese central:

> *Por meio do uso de soluções baseadas em técnicas de agrupamento, algoritmos de classificação de fluxo de dados podem ser precisos, eficientes e capazes de se adequar às mudanças de conceitos sem a necessidade de que o ambiente forneça ao classificador, os rótulos corretos dos exemplos de teste após a etapa de predição.*

Para que seja possível a adaptação do modelo de classificação sob a restrição de latência extrema, é necessário que sejam consideradas as seguintes condições:

1. As mudanças de conceito devem ser incrementais e responsáveis por gerar um deslocamento das classes no espaço de atributos dos dados. Este deslocamento pode ser, por exemplo, a translação de uma ou mais classes. Esta condição é necessária para que seja possível "rastrear" as mudanças das classes ao longo do tempo, a partir da observação de alterações nos valores dos atributos dos dados.

2. O número de classes é constante ao longo do tempo. Embora algumas aplicações apresentem um número variável ou desconhecido de classes, frequentemente há interesse em somente uma das classes. Assim, o problema pode ser transformado em um problema de classificação binária.

Também pode-se citar como objetivo deste trabalho, a coleta de dados de diferentes espécies e variadas condições ambientais com o sensor identificador de insetos, a exploração e avaliação de diferentes descritores dos dados e algoritmos de classificação com múltiplas classes e com uma única classe. Devido à presença de mudanças de conceitos nos dados obtidos pelo sensor, também se fez necessário a proposta de métodos de classificação capazes de se adaptar às mudanças para que o desempenho do classificador não seja degradado.

Nesta aplicação, embora ocorram mudanças de conceito incrementais nas distribuições dos dados de acordo com a variação de condições ambientais, observou-se que devido à

inatividade de determinadas espécies por um longo período de tempo, também podem ocorrer mudanças abruptas nos dados. De maneira resumida, se houveram mudanças ambientais significativas no intervalo de tempo entre duas ocorrências consecutivas de uma mesma espécie, os dados serão significativamente diferentes, constituindo uma mudança abrupta. Além disso, a inatividade de determinadas espécies durante determinados períodos torna a quantidade de classes do problema não constante ao longo de todo o tempo. Neste caso, por se tratar de uma aplicação sob a condição de latência extrema para a obtenção dos rótulos corretos e com a presença de mudanças que não são exclusivamente do tipo incremental, tem-se a hipótese de que a abordagem de seleção dinâmica de classificadores (Zhu et al., 2004) é adequada para este problema, dada a construção de múltiplos classificadores que contemplam diferentes características dos dados. Assim, é possível selecionar o classificador mais adequado durante a classificação de um exemplo de teste, independente do tipo de mudança ocorrida nos dados.

1.4 Principais Contribuições

As principais contribuições desta tese estão relacionadas à tarefa de classificação de fluxo de dados não estacionários. Especificamente, é abordado o problema de latência para o recebimento dos rótulos corretos dos exemplos processados. Este atraso é uma característica presente na maior parte das aplicações reais que geram ou processam fluxo de dados, mas não é considerada pela maior parte dos algoritmos atuais da literatura. A partir de experimentos com conjuntos de dados reais e sintéticos, em que se considerou diferentes valores de latência, foi possível verificar e discutir como este fator é responsável por alterar significativamente a acurácia de algoritmos estado-da-arte na literatura, sendo necessária a proposta de novas abordagens para o seu uso prático. Desse modo, somente a discussão sobre este assunto já constitui uma contribuição desta tese e espera-se que a comunidade científica considere este importante fator na proposta de novos algoritmos.

No cenário mais díficil de latência extrema, onde há total ausência na disponibilização de exemplos rotulados após a etapa de classificação, foram propostas e avaliadas duas soluções denominadas *Stream Classification Algorithm Guided by Clustering – SCARGC* (Souza et al., 2015c) e *Micro-Cluster Classification – MClassification* (Souza et al., 2015b). Para a avaliação dos algoritmos propostos, foram construídos e compilados um total de 16 conjuntos de dados sintéticos com características e comportamentos específicos que buscou-se avaliar, como grande volume e mudanças de conceito incrementais ao longo do tempo. Estes conjuntos de dados estão publicamente disponibilizados para a comunidade acadêmica e espera-se que estes dados possam contribuir para outros pesquisadores da área na proposta de novas soluções. Além de dados sintéticos, os algoritmos também foram avaliados em 2 conjuntos de dados reais. Em geral, os métodos propostos apresentaram

um bom desempenho em acurácia e tempo computacional, com resultados equivalentes ou superiores aos apresentados na literatura.

Este trabalho também contribuiu na exploração do problema de classificação de insetos por meio de sensores ópticos (Souza et al., 2013a,b), bem como na avaliação de diferentes atributos descritores dos dados provenientes da área de processamento digital de sinais em conjunto com algoritmos de classificação de aprendizado de máquina (Silva et al., 2013c, 2015b). Uma avaliação inicial considerou dados coletados em laboratório sem a presença de mudanças de conceito significativas. Nesta etapa, foi possível constatar o bom desempenho dos atributos *Mel-Frequency Cepstral Coefficients* (MFCC) e *Linear Frequency Cepstral Coefficients* (LFC), bem como dos classificadores Máquina de Vetores de Suporte (SVM) e k-Vizinhos Mais Próximos (kNN) na tarefa de classificação com 5 diferentes espécies de insetos, sendo possível alcançar uma acurácia de 90,33%.

Algoritmos de classificação convencionais com múltiplas classes, tanto em lote como em fluxo de dados, permitem a associação de um exemplo desconhecido a uma de várias possíveis classes previamente definidas. Este procedimento pode ser um problema quando o exemplo a ser classificado não pertence a nenhuma das classes conhecidas. No caso da classificação de insetos, existem milhares de possíveis espécies que podem ser atraídas pela armadilha, não sendo possível coletar dados de todas as espécies e construir um classificador suficientemente abrangente. Por isso, foram avaliados algoritmos que realizam o aprendizado apenas com exemplos positivos (classe de interesse). No caso deste trabalho, considerou-se somente a espécie *Aedes aegypti* como positiva. Nesta avaliação, foi possível atestar que o algoritmo baseado no uso de janelas de Parzen é capaz de obter um nível adequado em relação à taxa de falsos positivos e verdadeiros positivos (AUC = 0,87) na identificação de insetos da espécie *Aedes aegypti*.

Ao considerar um cenário com mudanças de conceito dadas as variações de temperatura e umidade, foram propostas abordagens que realizam a seleção dinâmica de classificadores com base em fatores como horário da ocorrência e temperatura, sendo possível identificar a espécie *Aedes aegypti* (fêmea) com mais de 95% de acurácia. Ainda, uma contribuição deste trabalho foi a construção de um conjunto de dados de insetos com mudanças de conceito. Pelo melhor de nosso conhecimento, este é o primeiro conjunto de dados real com mudanças de conceito induzidas artificialmente em que é possível observar claramente o impacto de fatores externos nos atributos que descrevem os dados.

Ao explorar o problema de identificação automática de insetos, espera-se que este trabalho possa contribuir a médio/longo prazo para mitigar problemas sociais e de saúde pública no Brasil e em diversos países. O sucesso da aplicação do sensor óptico em conjunto com armadilhas inteligentes ou outros dispositivos, pode ser um importante passo no controle de epidemias de dengue, febre amarela, malária, entre outras doenças. No caso do uso desta tecnologia no campo, as informações obtidas pelo sensor permitirão

um uso mais consciente, direcionado e de menor custo, de inseticidas e pesticidas para o controle de insetos pragas e mosquitos transmissores de doenças para os animais.

1.5 Organização do Trabalho

Esta tese está organizada da seguinte maneira:

Capítulo 2: Classificação de Fluxo de Dados

Neste capítulo, são apresentados os conceitos básicos sobre classificação de fluxo de dados e as diferenças entre o tradicional aprendizado em lote (*batch learning*). É também apresentada uma visão geral sobre os diferentes tipos de mudanças de conceito, suas causas e as abordagens utilizadas na literatura para lidar com o problema de classificação de fluxo de dados sob condições de mudanças de conceito ao longo do tempo.

Capítulo 3: Latência de Rótulos

Neste capítulo, é discutido o problema de latência na obtenção de rótulos corretos após a etapa de classificação de fluxo de dados. Embora pouco explorado na literatura, são apresentadas evidências empíricas da influência de diferentes valores de latência em métodos estado-da-arte na classificação de fluxo de dados. Também é apresentado o cenário de latência extrema, de especial interesse para este trabalho, bem como os métodos existentes na literatura que lidam com este problema.

Capítulo 4: Soluções Propostas para a Classificação de Fluxos de Dados Não Estacionários e Latência Extrema

Neste capítulo, são apresentadas duas propostas de algoritmos para lidar com o problema de latência extrema na classificação de fluxo de dados com mudanças de conceito. Especificamente, os algoritmos propostos *Stream Classification Algorithm Guided by Clustering – SCARGC* e *Micro-Cluster Classification – MClassification* são apresentados e avaliados em dados sintéticos de *benchmark* com 16 conjuntos que visam simular de maneira controlada comportamentos específicos dos dados que podem ocorrer em situações reais. Também são apresentados e avaliados dois conjuntos de dados reais com mudanças de conceito incrementais ao longo do tempo.

Capítulo 5: Sensor para a Identificação Automática de Insetos

Neste capítulo, é apresentado o principal objeto de estudo deste trabalho. Além da apresentação do sensor identificador de insetos, é discutido como este sensor pode ser utilizado em conjunto com armadilhas inteligentes capazes de capturar somente espécies

de interesse e auxiliar em diversos problemas sociais e de saúde pública. Também é apresentada uma avaliação de diferentes atributos descritivos dos dados provenientes da área de processamento digital de sinais em conjunto com diferentes classificadores de aprendizado de máquina com múltiplas classes, para a classificação de diferentes espécies de insetos, e com uma única classe, visando a captura somente do mosquito *Aedes aegypti*.

Capítulo 6: Classificação Automática de Insetos em Ambientes Não Estacionários

Neste capítulo, são apresentadas as principais causas de mudanças de conceito nos dados medidos pelo sensor identificador de insetos. Também são apresentadas as etapas de coleta de dados que visam simular um cenário de uso do sensor em campo, com mudanças nas variáveis de temperatura e umidade. O conjunto de dados coletado é analisado e pode-se confirmar a relação entre frequência de batida de asas e temperatura. Também são apresentadas soluções com múltiplos classificadores, em que se realiza a seleção dinâmica do classificador para a adaptação às mudanças de conceito.

Capítulo 7: Conclusões

Neste capítulo, são apresentadas as conclusões finais desta tese, bem como as limitações e os trabalhos futuros.

Apêndice A: Outras Contribuições

Por fim, são apresentadas outras contribuições deste trabalho relacionadas as áreas de séries temporais, processamento digital de sinais e aprendizado ativo. Embora estas contribuições não sejam diretamente relacionadas aos objetivos principais desta pesquisa, surgiram devido à exploração dos problemas de pesquisa e à avaliação de soluções em diferentes domínios.

Capítulo

2

Classificação de Fluxo de Dados

2.1 Considerações Iniciais

Com a crescente popularização de sensores e dispositivos de medições capazes de gerar dados continuamente em grande volume e alta taxa de chegada, o processamento de fluxo de dados tem sido cada vez mais explorado nas áreas de Mineração de Dados e Aprendizado de Máquina. Entre diferentes possíveis tarefas que podem ser realizadas com estes dados, classificação é uma das mais proeminentes. Em diversos cenários, os fluxos de dados alteram suas características ao longo do tempo, sendo necessário o uso de algoritmos adaptativos para que a acurácia de classificação se mantenha estável mesmo com a presença de mudanças nos dados. Neste capítulo é apresentada uma revisão da literatura sobre métodos adaptativos que realizam a classificação de fluxo de dados sob condições de mudanças de conceito ao longo do tempo. Para isso, inicialmente são apresentados os conceitos de fluxo de dados na Seção 2.2 e ambientes não estacionários na Seção 2.3. As principais abordagens utilizadas pelos algoritmos da literatura para lidar com mudanças de conceito são apresentadas na Seção 2.4.

2.2 Fluxo de Dados

Nas últimas décadas, as pesquisas e a prática na área de Aprendizado de Máquina têm focado no aprendizado em *batch* (também conhecido como aprendizado *offline*). Neste tipo de aprendizado, os dados ficam disponíveis de modo integral para o treinamento do algoritmo, o qual realiza, se necessário, múltiplas visitas aos dados para a geração do modelo. Este fato ocorre com a maioria dos algoritmos. Além disso, um único modelo é

gerado e não é mais revisado por novos dados. Essa abordagem se mostrou adequada e eficiente para diferentes aplicações. Entretanto, a maior parte dos algoritmos tradicionais de aprendizado de máquina não é capaz de lidar com dados gerados continuamente em ambientes dinâmicos (Gama, 2010). Nesse tipo de ambiente, é necessário que o algoritmo incorpore de modo incremental novos dados ao modelo de aprendizado. Algoritmos supervisionados como Vizinhos-Mais-Próximos e *Naive-Bayes* são naturalmente incrementais. Por outro lado, algoritmos como Árvores de Decisão e Máquinas de Vetores de Suporte necessitam de mudanças substanciais para que a indução do modelo seja incremental. Também é importante destacar que em muitas aplicações do mundo real, o processo de geração dos dados não é estacionário, de modo que os dados sofrem alterações ao longo do tempo. Por isso, o aprendizado incremental é uma propriedade importante mas não suficiente para a adequação dos algoritmos tradicionais para o aprendizado *online*. Além da incorporação de dados recentes, os algoritmos devem possuir mecanismos rápidos e eficientes para manter o modelo atualizado de acordo com as possíveis mudanças ocorridas nos dados e descartar informações e conceitos que já não refletem o estado atual do problema (Gama, 2010).

Devido às limitações do aprendizado em *batch* apresentadas anteriormente e o constante aumento do número de aplicações capazes de gerar um grande volume de dados continuamente, uma recente área de pesquisa tem ganhado destaque: *Mineração de Fluxos Contínuos de Dados*. Nessa área, os algoritmos de mineração lidam com dados constituídos por uma sequência ordenada e ilimitada de registros ou eventos que chegam ao longo do tempo em grande quantidade e não permitem o armazenamento permanente em memória, denominados fluxos de dados ou *data streams* (Babcock et al., 2002; Gama et al., 2006). Diferentes aplicações reais como monitoramento de tráfego de rede, detecção de fraudes em cartão de crédito, rastro de cliques em páginas *web*, registro de ligações telefônicas e sensores em geral, lidam com fontes geradoras de fluxos de dados.

De acordo com Kirkby (2007), um algoritmo de classificação deve atender a diversos requisitos para lidar com fluxos contínuos de dados, destacando-se principalmente quatro requisitos que os diferenciam do aprendizado em *batch*:

1. **Processar um exemplo por vez e inspecioná-lo somente uma única vez.** Uma importante característica dos fluxos de dados é que os exemplos chegam continuamente um após o outro de maneira ordenada. Tal característica não permite o acesso aleatório aos dados a serem classificados. Assim, os exemplos devem ser classificados de acordo com a ordem de chegada. Um exemplo pode ser inspecionado uma única vez ou descartado. Caso seja descartado, não poderá ser recuperado. Entretanto, o algoritmo pode utilizar uma memória de curta duração para armazenar um subconjunto de exemplos, o qual será eventualmente descartado;

2. **Utilizar uma quantidade limitada de memória.** Uma das principais motivações da abordagem de fluxos de dados é processar uma quantidade de dados maior que a memória disponível. Se não existir um limite definido para a utilização da memória, ela pode ser facilmente esgotada. A memória utilizada pelos algoritmos de classificação de fluxos pode ser dividida em duas categorias: *i)* responsável por armazenar estatísticas dos dados e *ii)* responsável por armazenar o modelo de classificação atual. O modelo deve ser simples para que ocupe pouco espaço em memória. Idealmente, o modelo de um algoritmo altamente eficiente em termos de memória seria composto apenas por estatísticas dos dados sem a necessidade de armazená-los;

3. **Operar em tempo limitado.** O algoritmo deve, preferencialmente, ser capaz de classificar um exemplo antes que o próximo dado do fluxo chegue para ser processado. Caso contrário, irá acarretar perdas de dados devido às limitações de memória que não permitem o armazenamento e processamento futuro do exemplo;

4. **Responder a qualquer momento.** Espera-se que o algoritmo seja capaz de produzir um modelo adequado a partir da observação de qualquer quantidade de dados e não apenas após o processamento completo dos dados. O processo de geração do modelo deve ser o mais eficiente possível. Por isso, sua estrutura deve ser facilmente atualizável para a inclusão ou modificação de um conceito. No melhor caso, o modelo final é diretamente manipulado na memória pelo algoritmo que processa os exemplos, sem a necessidade de recalcular todo o modelo a partir de estatísticas extraídas dos dados.

Na Figura 2.1, é possível observar como estes requisitos interagem em um ciclo de repetição composto por três passos, os quais são tipicamente considerados por algoritmos de classificação de fluxos de dados (Bifet et al., 2009):

1. O algoritmo recebe o próximo exemplo do fluxo a ser processado (Requisito 1);

2. O algoritmo processa o exemplo e atualiza a sua estrutura de dados. Durante a atualização, não deve exceder os limites de memória (Requisito 2) e deve ser o mais rápido possível de modo a terminar esta etapa antes da chegada do próximo exemplo (Requisito 3);

3. O algoritmo está pronto para processar o próximo exemplo. No momento da requisição, deve ser capaz de fornecer um modelo atualizado para predizer a classe dos próximos exemplos não vistos (Requisito 4).

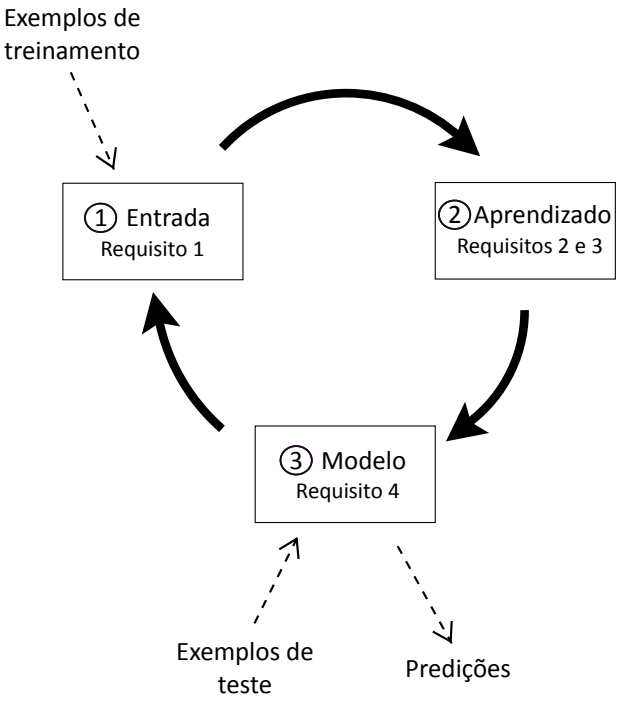


Figura 2.1: Ciclo de classificação no modelo de fluxos de dados (Bifet et al., 2009). Relação entre cada fase do ciclo e os requisitos que devem ser atendidos pelo algoritmo de classificação.

2.3 Ambientes Não Estacionários e Mudanças de Conceito

Em diversas aplicações do mundo real, os dados que descrevem um problema podem passar por modificações ao longo do tempo, devido à natureza não estacionária do ambiente responsável por gerar os dados (Widmer e Kubat, 1996). Mais especificamente em um problema de classificação, as descrições das classes podem variar ao longo do tempo, de modo que os valores dos atributos usados na discriminação das classes no tempo t podem não ser mais úteis no tempo $t + k$. Assim, um classificador induzido a partir de um conjunto de treinamento inicial será ineficiente se não atualizado devido às alterações no problema. Tais alterações na distribuição dos dados em períodos de tempo são denominadas de mudanças de conceito ou *concept drift*. É importante destacar que mudanças sazonais não são consideradas mudanças de conceito. Entretanto, se o período da mudança não é totalmente conhecido, pode-se caracterizar como *drift*.

Considere a chegada de uma sequência de instâncias, uma instância de cada vez, não necessariamente em intervalos de tempo igualmente espaçados. Dado $\vec{x}_t$ um vetor com p valores de atributos preditivos observado no tempo t e y_t um rótulo de classe. Pode-se chamar $\vec{x}_t$ de instância e o par $(\vec{x}_t, y_t)$ de instância rotulada. As instâncias $(\vec{x}_1, \ldots, \vec{x}_t)$ são denominadas *conjunto de dados históricos* e a instância $\vec{x}_{t+1}$ é denominada *instância de teste*. Em cada passo t de tempo, tem-se disponível um conjunto de dados históricos (rotulados) $X^H = \{(\vec{x}_1, y_1), \ldots, (\vec{x}_t, y_t)\}$. Assim que uma instância de teste $\vec{x}_{t+1}$ surge,

a tarefa do classificador é predizer qual o seu rótulo de classe $\hat{y}_{t+1}$. Para isso, se induz um classificador $\mathcal{L}_t$ utilizando como conjunto de treinamento todos os dados ou uma seleção dos dados históricos disponíveis X^H. Esse processo é ilustrado na Figura 2.2, na qual considera-se também que cada instância $\vec{x}_t$ é gerada por uma distribuição S_t. Se todos os dados possuem a mesma distribuição, de modo que $S_1 = S_2 = \ldots = S_{t+1}$, o conceito é estável. Se em qualquer dois instantes de tempo i e j, $S_i \neq S_j$, diz-se que houve uma mudança de conceito, conforme pode ser considerado na Figura 2.2. Estas mudanças podem ocorrer de diferentes maneiras. Por exemplo, as características dos dados de entrada podem mudar ao longo do tempo ou a relação entre os dados e suas respectivas classes pode ser alterada.

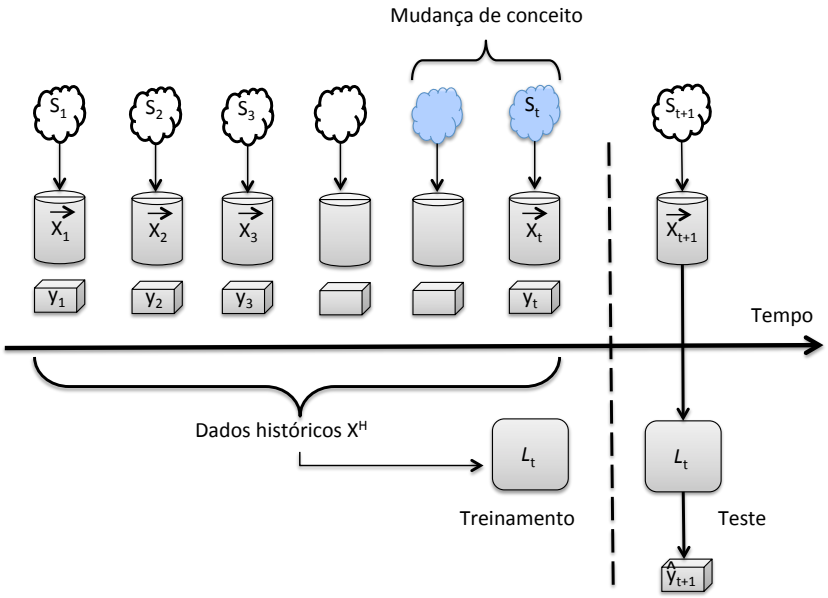


Figura 2.2: Representação do processo de classificação de uma instância X_{t+1} em um fluxo de dados com mudança de conceito (Žliobaitė, 2009b).

Após a atribuição do rótulo $\hat{y}_{t+1}$ para a instância $\vec{x}_{t+1}$, tal exemplo pode passar a compor o conjunto de dados históricos X^H dependendo do algoritmo para formação/atualização do conjunto de treinamento e da memória disponível para armazenamento. Por se tratar de um fluxo de dados, repete-se o processo para as próximas instâncias que chegam para classificação no decorrer do tempo.

2.3.1 Causas de Mudanças

A mudança pode ocorrer em variáveis ocultas que descrevem o problema ou em propriedades intrínsecas das variáveis preditivas observadas. Por exemplo, as preferências dos clientes em um *site* de compras podem mudar ao longo do tempo de acordo com a taxa da inflação, disponibilidade de alternativas em lojas concorrentes, campanhas de

marketing para a venda de determinados produtos ou devido ao dia da semana ou estação do ano. Por isso, frequentemente a causa da mudança de conceito é oculta e previamente desconhecida, o que torna a tarefa de aprendizado mais complexa, já que o algoritmo deve detectar e se adaptar rapidamente às mudanças (Widmer e Kubat, 1996; Tsymbal, 2004).

Independente da presença de mudanças de conceito na distribuição dos dados, em um cenário probabilístico o problema de classificação em aprendizado de máquina pode ser descrito como segue: dada uma instância desconhecida $\vec{x}$ com p valores de atributos descritivos, o objetivo de um classificador ótimo (de menor erro), é atribuir o rótulo de classe y_i de maior probabilidade, dado que $y_i \in Y$ e $Y = \{y_1, y_2, \ldots, y_k\}$ é um conjunto com k possíveis classes do problema. Este classificador é totalmente determinado pelas probabilidades *a priori* $P(y_i)$ e pelas funções de densidade de probabilidade condicionais das amostras das classes $P(\vec{x}|y_i), i = 1, \ldots, k$.

Assim, pode-se definir conceito ou fonte de dados S, o conjunto de probabilidades *a priori* e condicionais das classes:

$$S = \{(P(y_1), P(\vec{x}|y_1)), (P(y_2), P(\vec{x}|y_2)), \ldots, (P(y_k), P(\vec{x}|y_k))\}.$$

Dado que $P(y_i|\vec{x})$ denota a probabilidade do exemplo $\vec{x}$ pertencer à classe y_i (probabilidade *a posteriori*), o teorema de *Bayes* provê um método para a realização do cálculo de modo que $\vec{x}$ pode ser associado à classe y_i de mais alta probabilidade:

$$P(y_i|\vec{x}) = \frac{P(y_i)P(\vec{x}|y_i)}{P(\vec{x})},$$

em que o denominador $P(\vec{x})$ é constante para todas as classes e, por isso, pode ser ignorado.

Conforme apontado por Kelly et al. (1999), a mudança de conceito pode ocorrer de três maneiras diferentes:

1. As probabilidades *a priori* das classes $P(y_i)$ podem mudar ao longo do tempo;
2. As distribuições dos dados (verossimilhança) $P(\vec{x}|y_i)$ podem mudar;
3. As probabilidades condicionais *a posteriori* $P(y_i|\vec{x})$ podem mudar.

Para fins de exemplificação, estas mudanças são ilustradas na Figura 2.3, dado um problema de classificação binária (•, ▲) com dados bidimensionais (Hoens et al., 2012). Na representação, tem-se a disposição dos dados no espaço de atributos no tempo t e uma nova disposição após a ocorrência de uma mudança no tempo $t + 1$. Na Figura 2.3-*a*), nota-se que após a mudança, a classe • passou a ser mais frequente, caracterizando mudanças em $P(y_\bullet)$. Na Figura 2.3-*b*), pode-se observar alterações nos limites da classe •, caracterizando mudanças em $P(\vec{x}|y_\bullet)$. Por fim, nota-se na Figura 2.3-*c*) que a superfície de separação entre as classes • e ▲ se altera, caracterizando mudanças nas probabilidades condicionais *a posteriori* de ambas as classes, $P(y_\bullet|\vec{x})$ e $P(y_\blacktriangle|\vec{x})$.

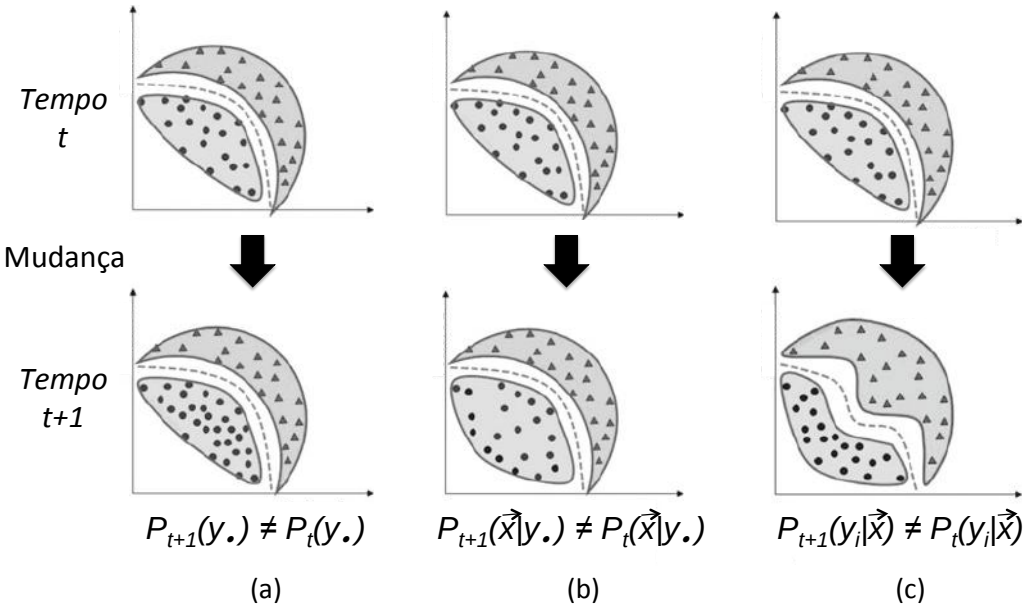
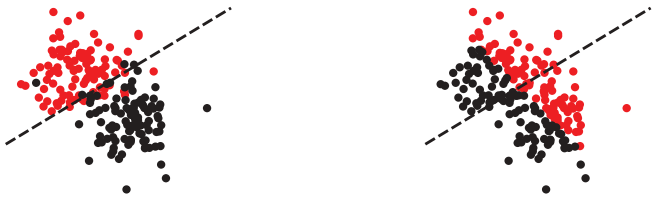


Figura 2.3: Exemplificação da três possíveis mudanças nos dados. Em *a*) são representadas mudanças na probabilidade *a priori* de uma das classes; em *b*) são representadas mudanças na distribuição dos dados de uma das classes e; em *c*) são representadas mudanças nas probabilidades condicionais de ambas as classes. Adaptado de Hoens et al. (2012).

As mudanças nos dados podem ser diferenciadas entre *real* e *virtual* (Widmer e Kubat, 1993). A mudança real ocorre na distribuição condicional da variável de saída a ser predita, $P(y_i|\vec{x})$. A mudança real pode ocorrer com ou sem alterações em $P(\vec{x}|y_i)$ ou $P(y_i)$. Na mudança virtual, embora ocorram alterações em $P(y_i)$ ou $P(\vec{x}|y_i)$, as probabilidades condicionais das classes $P(y_i|\vec{x})$ não se alteram. Entretanto, na prática esta diferenciação entre os tipos de mudanças é pouco importante para os algoritmos de aprendizado, sendo mais importante identificar se determinada mudança é responsável por degradar o desempenho do classificador após as alterações.

A Figura 2.4 ilustra um caso de mudança real, em virtude das alterações em $P(y_i|\vec{x})$. No exemplo, é evidente que a acurácia de um classificador que não se adapta às mudanças irá degradar, com base na incorreta separação entre as classes considerada pelo modelo antigo.

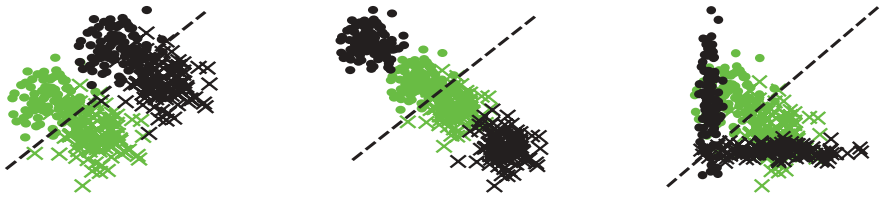
Entretanto, é importante notar que nem sempre as mudanças acarretam a degradação da acurácia do classificador. A Figura 2.5 ilustra este cenário com 3 diferentes exemplos de mudanças considerando um problema com duas classes. Na ilustração, os dados originais das duas classes consideradas são representados na cor verde (•, ×), enquanto os dados após as mudanças são representados na cor preta (•, ×). Embora ocorram mudanças em $P(y_i|\vec{x})$ (*real drift*), a margem de separação das classes inicialmente definida a partir dos dados originais se mantêm válida. Assim, um classificador que não se adapta a estas mudanças ainda é capaz de manter estável a sua acurácia de classificação.



(a) Dados originais

(b) Após uma mudança em $P(y_i|\vec{x})$

Figura 2.4: Exemplo de *real drift* em dados com duas classes. *a*) dados originais e margem de separação entre as classes; *b*) dados após mudanças em $P(y_i|\vec{x})$ e margem de separação entre as classes desatualizada.



(a) Mudança 1 (b) Mudança 2 (c) Mudança 3

Figura 2.5: Três exemplos de mudanças que não degradam a acurácia de um classificador treinado a partir dos dados originais (Kuncheva e Faithfull, 2014). Dados originais são representados na cor verde (•, ×), enquanto os dados após as mudanças são representados na cor preta (•, ×).

2.3.2 Tipos de Mudanças

Existem diferentes possibilidades de mudanças de conceito e sua identificação é importante para que o classificador se adapte e tome uma decisão de como isso será realizado. De modo simplificado, considerando-se apenas duas distribuições S_I e S_{II} que representam os conceitos A e B, temos ao menos quatro diferentes tipos de mudanças conforme apresentado na Figura 2.6, em que o tempo é representado no eixo horizontal e a média das distribuições é apresentada no eixo vertical. Na ilustração, as mudanças representadas são: *a)* Abrupta; *b)* Gradual; *c)* Incremental e *d)* Recorrente. Entretanto, os tipos de mudanças não se limitam a este pequeno conjunto. O trabalho de Minku et al. (2010) aborda outras possibilidades de mudanças levando em consideração características como a previsibilidade, gravidade, velocidade, frequência e recorrência.

Na mudança abrupta (*sudden drift*), um conceito A é repentinamente substituído por um conceito B. Também costuma-se utilizar o termo *concept shift* para este tipo de mudança.

Na mudança gradual (*gradual drift*), os conceitos A e B se misturam no início, de modo que ambos são ativos em um determinado período de tempo. Com o passar do tempo, o número de ocorrências do conceito A diminui sendo substituído por B.

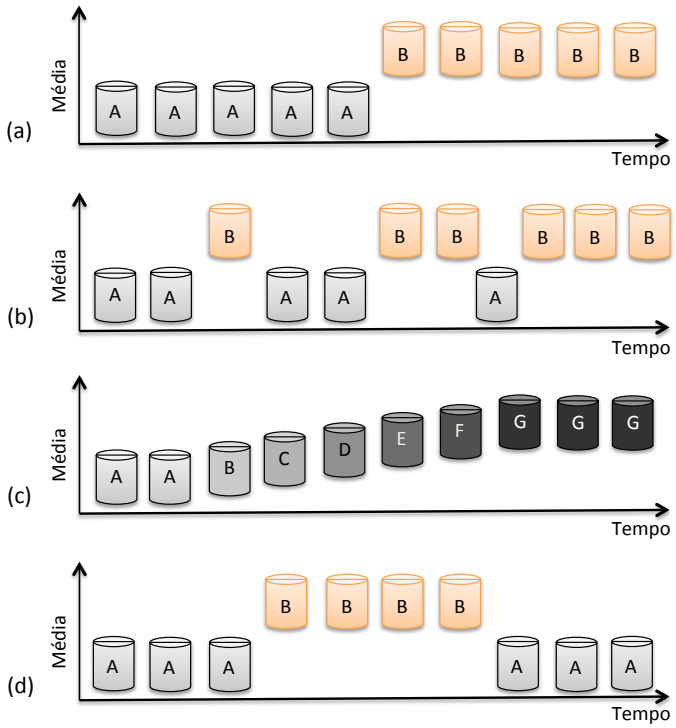


Figura 2.6: Representação de quatro tipos de mudanças de conceitos: (a) Mudança abrupta (*sudden drift*); (b) Mudança gradual (*gradual drift*); (c) Mudança incremental (*incremental drift*) e (d) Mudança recorrente (*reoccurring drift*) (Žliobaitė, 2009b).

A mudança incremental (*incremental drift*) envolve mais de dois conceitos. Entretanto, a diferença entre eles é pequena, sendo notáveis apenas em longos períodos de tempo.

A mudança recorrente (*reoccurring drift*) ocorre quando um conceito anteriormente ativo reaparece após um determinado tempo. Esse tipo de mudança é diferente de uma sazonalidade periódica, pois não é claro quando o conceito irá voltar a ser ativo. Uma consideração especialmente importante para este trabalho é que a sazonalidade periódica não é considerada um problema de mudança de conceito. Entretanto, se a sazonalidade não é totalmente conhecida, se enquadra como mudança de conceito. Um exemplo é a frequência de batidas de asas que caracteriza as espécies de insetos, em que tal valor é associado com a temperatura, entre outras características meteorológicas. Assim, embora se conheça a existência de sazonalidade entre os períodos do ano que ocorrem mudanças na temperatura, os momentos de mudanças não são totalmente conhecidos. Por exemplo, podem existir dias com baixa temperatura no verão, bem como dias com alta temperatura no inverno.

2.4 Abordagens para Lidar com Mudanças de Conceito

Em geral, as abordagens utilizadas para lidar com a classificação de fluxo de dados sob condições de mudanças de conceito podem ser divididas em duas categorias (Sebastião e

Gama, 2009): *i)* abordagens que adaptam o modelo de classificação em intervalos regulares sem considerar se a mudança de fato ocorreu e *ii)* abordagens que primeiramente detectam a mudança e logo após adaptam o modelo de classificação. Estas abordagens são apresentadas nas Subseções 2.4.1 e 2.4.2, respectivamente.

2.4.1 Métodos sem Detecção de Mudanças

Na primeira abordagem, são utilizadas técnicas que envolvem janelas de tempo com tamanho fixo em que o esquecimento dos exemplos mais antigos é abrupto ou são associados pesos aos exemplos de acordo com sua idade ou utilidade, de modo que o esquecimento dos exemplos antigos seja gradual. No uso de técnicas com janelas, a cada passo de tempo o classificador é induzido somente com os exemplos incluídos no intervalo da janela. Essa técnica leva em consideração que informações obtidas a partir dos dados mais recentes do fluxo são consideradas mais relevantes, não sendo necessário observar todo o passado. Isso devido à possibilidade dos dados mais antigos se tornarem desatualizados com o decorrer do tempo e ao tamanho ilimitado do fluxo, o que inviabiliza sua análise completa. Uma das principais dificuldades é definir o tamanho ideal da janela, dado que uma janela pequena pode assegurar uma rápida adaptabilidade do classificador em fases com mudanças de conceito, mas pode afetar o desempenho em fases mais estáveis, já que o classificador passaria por constantes adaptações sem necessidade. Por outro lado, janelas grandes conduzem a bons resultados e a um bom desempenho do classificador em fases estáveis, porém podem produzir um classificador incapaz de reagir rapidamente às mudanças. Diferentes modelos de janelas são apresentas na literatura. Dois exemplos são (Gama, 2010):

- Janelas deslizantes (*sliding windows*). Na abordagem mais simples, a janela deslizante possui um tamanho fixo w definido em termos de número de observações. Esse tipo de janela é similar à estrutura de dados FIFO (*first in, first out*): quando um elemento $\vec{x}_i$ é observado e inserido na janela, o elemento mais antigo $\vec{x}_{i-w}$ é removido. A Figura 2.7-*a)* ilustra uma janela deslizante de tamanho fixo w sendo atualizada com as observações mais recentes. Outra abordagem possível para as janelas deslizantes é definir o seu tamanho em termos de duração de tempo. Assim, uma janela de tamanho t possui uma quantidade variável de elementos que ocorreram dentro do intervalo de tempo t. Por exemplo, a janela possui os últimos eventos ocorridos dentro do intervalo de 2 horas. Assim que o tempo transcorre, a janela é continuamente atualizada. Ainda na Figura 2.7-*a)*, se o eixo horizontal for interpretado como uma linha de tempo, as janelas w são atualizadas em quatro diferentes períodos de tempo;

- Janelas por marcação (*landmark windows*). Na janela por marcação (Gehrke et al., 2001), um ponto considerado relevante do fluxo é fixado como referência (*landmark*)

e se extrai informações dos dados subsequentes de modo que a janela aumenta o seu tamanho ao longo do tempo. A Figura 2.7-*b* ilustra uma janela por marcação em quatro diferentes períodos e com a *landmark* fixada no tempo ou instância $i - j$.

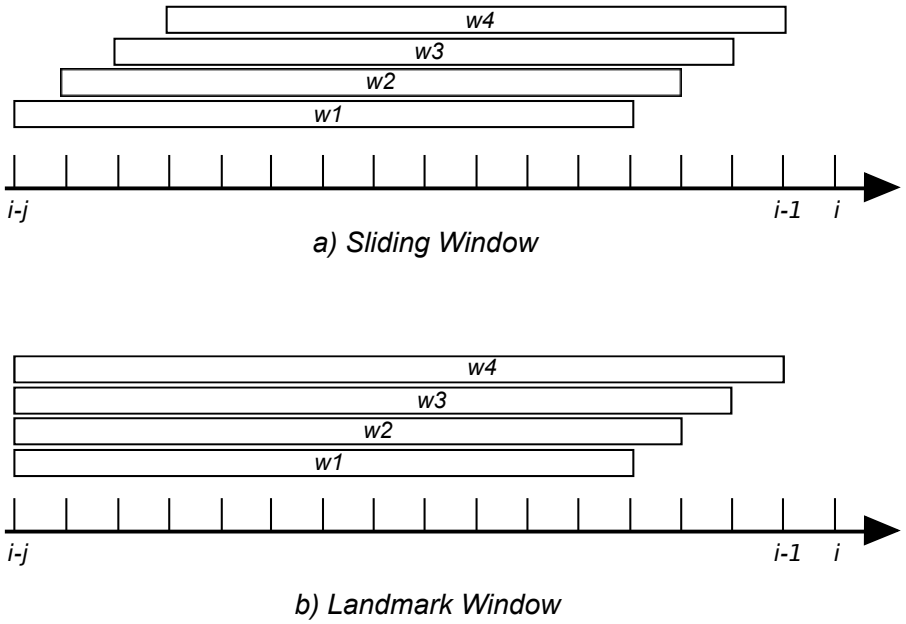


Figura 2.7: Exemplo de (a) janela deslizante (*sliding window*) e (b) janela por marcação (*landmark window*) em quatro diferentes períodos de tempo (Gama, 2010).

Exemplos de trabalhos que utilizam janelas são a família de algoritmos FLORA (Widmer e Kubat, 1996), FRANN (Kubat e Widmer, 1995) e o algoritmo *Time-Windowed Forgetting* – TWF (Salganicoff, 1997).

As técnicas que atribuem pesos aos exemplos se baseiam na simples ideia de que a importância de um exemplo deve diminuir gradualmente com o passar do tempo. Desse modo, pesos são atribuídos aos exemplos de treinamento de acordo com esta heurística. Diferentemente da abordagem com janelas, em que a partir de determinado ponto os dados se tornam irrelevantes e são abruptamente descartados, os dados antigos vão diminuindo sua relevância aos poucos sem a necessidade de descarte. Pode-se atribuir pesos que levam em consideração um esquecimento linear conforme apresentado por Koychev (2000) ou exponencial como apresentado por Klinkenberg (2004).

No caso do esquecimento linear, é considerada uma função f_i para cada exemplo $\vec{x}_i$, dado que $f_i \geq 0$ e $\frac{\sum_{i=1}^{n} f_i}{n} = 1$. A função é definida como $f_i = -\frac{2p}{n-1}(i-1) + 1 + p$, em que $i = \{1, \ldots, n\}$ representa a posição do exemplo no fluxo a partir dos exemplos mais recentes para os mais antigos, n é a quantidade de exemplos observados e $p \in [0, 1]$ é um parâmetro que representa a porcentagem de diminuição do peso para o primeiro exemplo e, consequentemente, a porcentagem de aumento do peso para as instâncias mais recentes.

As técnicas com decaimento exponencial atribuem pesos aos exemplos de acordo com sua idade a partir de uma função de esquecimento $f_\lambda(\vec{x}_i) = exp(-\lambda j)$, em que o exemplo

$\vec{x}_i$ foi processado há j passos anteriores. O parâmetro λ controla a taxa de decaimento. Um valor grande para λ faz com que um peso pequeno seja associado a exemplos antigos, já que estes exemplos são considerados menos informativos para as predições atuais. Por outro lado, se $\lambda = 0$ todos os exemplos possuem o mesmo peso.

2.4.2 Métodos com Detecção de Mudanças

Já nas abordagens com o objetivo de detectar explicitamente as mudanças de conceito, costuma-se utilizar indicadores que são continuamente monitorados ao longo do tempo, a partir da análise de exemplos presentes em janelas de tamanho adaptativo. Exemplos de indicadores são medidas de desempenho como a acurácia, precisão (*precision*) e revocação (*recall*). Também podem ser realizados testes estatísticos para comparar a distribuição dos dados presentes em janelas que consideram diferentes instantes de tempo. Assim que um dos indicadores atinge um determinado limiar, pode-se tomar a decisão de ajustar o modelo de classificação. Uma das vantagens desta abordagem é que os métodos podem proporcionar informações significativas sobre o processo em análise, como a quantificação e os pontos de mudanças.

Os métodos que monitoram a distribuição dos dados utilizam tipicamente uma janela de referência que resume as informações do passado e uma janela deslizante com informações dos dados mais recentes. As distribuições dos dados presentes nas duas janelas são comparadas a partir de métodos estatísticos, com a hipótese nula de que ambas as distribuições são iguais. Se a hipótese nula é rejeitada, considera-se que uma mudança foi identificada a partir do instante de tempo que contempla a janela deslizante com os dados mais recentes. Os testes devem ser realizados separadamente para cada dimensão e classe do problema. Como exemplos de testes estatísticos que podem ser aplicados, pode-se citar os Limites de Chernoff, conforme conduzido por Kifer et al. (2004), e a divergência de Kullback-Leibler para medir a distância entre as distribuições de probabilidade e detectar mudanças, como realizado nos trabalhos de Dasu et al. (2006) e Sebastião e Gama (2007). Ainda, Vorburger e Bernstein (2006) propuseram uma medida baseada em entropia para comparar as distribuições.

De modo geral, os métodos que monitoram medidas de desempenho assumem a premissa de que em processos estacionários o erro de um classificador se mantém constante. Assim, em processos não estacionários o erro aumenta com o decorrer do tempo. Portanto, a identificação da mudança por esses algoritmos se dá a partir da monitoração do erro do classificador. Uma das principais dificuldades é distinguir mudanças efetivas no processo de geração dos dados de perturbações temporárias (ruído). Em aplicações reais, uma dificuldade relevante e abordada neste trabalho é a dificuldade em se obter exemplos rotulados para que a taxa de erro seja monitorada. Os métodos de detecção de mudanças mais tradicionais e utilizados na literatura são: *Page-Hinkley Test* (Page, 1954), *Drift De-*

tection Method (Gama et al., 2004), *Early Drift Detection Method* (Baena-García et al., 2006) e *Adaptive Windowing* (Bifet e Gavalda, 2007). Estes métodos são apresentados a seguir.

Page-Hinkley Test

Um dos métodos mais populares para a detecção de mudanças é a técnica de análise sequencial *Page-Hinkley Test* (PHT) proposto por Page (1954). O PHT foi desenvolvido para detectar mudanças na média de sinais gaussianos e, para isso, utiliza um limiar constante para identificar a ocorrência de mudanças abruptas (Mouss et al., 2004). O teste considera uma variável acumulativa m_T, definida como a diferença acumulada entre os valores observados em um sinal e a sua média até o momento. Dado um sinal gaussiano até então de tamanho T e uma observação x_t ocorrida no tempo t, a variável acumulativa m_T é definida de acordo com a Equação 2.1.

$$m_T = \sum_{t=1}^{T} (x_t - \bar{x}_T - \delta) \tag{2.1}$$

em que δ representa a magnitude das mudanças permitidas no sinal e a média $\bar{x}_T$ é atualizada com o exemplo x_t e pode ser obtida a partir da Equação 2.2.

$$\bar{x}_T = \frac{1}{T} \sum_{t=1}^{T} x_t \tag{2.2}$$

Para detectar as mudanças, são calculados os valores mínimos de m_T a partir de $M_T = min(m_t, t = 1, \ldots, T)$ e monitorada a diferença entre m_T e M_T a partir do teste $PHT = m_T - M_T$. Caso o valor de PHT seja superior a um limiar de detecção ϵ definido pelo usuário, uma mudança na distribuição é identificada. O limiar ϵ tem influência direta na taxa de falsos alarmes. Conforme observado por Sebastião e Gama (2009), ao aumentar ϵ diminui-se a ocorrência de falsos alarmes. Entretanto, faz com que verdadeiras mudanças não sejam detectadas ou sejam detectadas com atraso. Desse modo, o teste requer que o parâmetro referente ao limiar de detecção seja cuidadosamente ajustado para que se alcance uma taxa aceitável entre falsos alarmes e mudanças não detectadas.

Embora o método tenha sido proposto para a análise de sinais, ele pode ser utilizado no contexto de fluxo de dados se considerarmos que um fluxo de dados unidimensional é um sinal. Assim, no caso de fluxo de dados multidimensionais, é necessário aplicar o teste separadamente para cada uma das dimensões do problema. O funcionamento do teste considerando o caso unidimensional é apresentado no Algoritmo 1.

Algoritmo 1: Page-Hinkley Test

Entrada:
Fluxo de dados rotulados X
Limiar de magnitude δ
Limiar de detecção ϵ
Saída:
Indicativo de mudança
Instante de tempo t em que a mudança foi identificada

```
início
    para todo exemplo x_t em X faça
        t ← InstanteTempoAtual;
        x̄_T ← AtualizarMédia(x_t); /* Equação 2.2 */
        m_T ← Atualizar(x_t, x̄_t, δ); /* Equação 2.1 */
        M_T ← min(m_t, t = 1, ..., T);
        PHT ← m_T − M_T;
        se PHT > ϵ então
            Mudança detectada;
            retorna t;
        fim
    fim
fim
```

Drift Detection Method

O *Drift Detection Method* (DDM) realiza o monitoramento da variabilidade da qualidade de um processo ao longo do tempo. No contexto de classificação de fluxo de dados, o método monitora a evolução da taxa de erro de classificadores, que pode se enquadrar em três níveis: *i)* fora de controle; *ii)* sob controle ou *iii)* alerta. Estes níveis podem ser interpretados como o grau da mudança de conceito nos dados e permitem sinalizar as degradações do processo de aprendizado, com alertas ao algoritmo de classificação.

Os alertas de mudança são sinalizados quando se observa um aumento da taxa de erro até um primeiro limiar (*lower threshold*), mas ainda não há evidências para sinalizar uma mudança. Ao entrar no modo de alerta, o algoritmo armazena o tempo t_w referente à observação do exemplo que ultrapassou o primeiro limiar. Se o erro volta a ficar abaixo do primeiro limiar, o modo de alerta é cancelado. Entretanto, se em uma sequência de exemplos do fluxo o erro aumentar até alcançar um segundo limiar (*higher threshold*) no tempo t_d, uma mudança na distribuição dos dados é declarada. Identificada a mudança de conceito, o classificador é retreinado utilizando somente os exemplos processados entre os instantes de tempo t_w e t_d. Para isso, o algoritmo deve armazenar em uma memória de curto prazo os exemplos rotulados recebidos entre o sinal de alerta e a mudança. Os níveis de mudança e sinais de alerta são reiniciados e a taxa de erro volta a ser monitorada nos próximos exemplos.

Early Drift Detection Method

O *Early Drift Detection Method* (EDDM) possui os mesmos mecanismos de alerta de mudança do método DDM. Entretanto, se baseia na ideia de que a distância temporal entre dois erros consecutivos aumenta quando um conceito é estável. Desse modo, o algoritmo EDDM não monitora somente a taxa de erro de classificação para a emissão dos alertas, mas a distância entre os erros em termos de número de exemplos entre dois erros de classificação. Em comparação com o método DDM, tal modificação melhorou o desempenho do método para a detecção de mudanças graduais e mantém os mesmos resultados para a detecção de mudanças abruptas. Por outro lado, o método é mais sensível à presença de ruído nos dados.

Adaptive Windowing

O método ADaptive WINdowing (ADWIN) utiliza uma janela W de tamanho n para armazenar os $\vec{x}_t$ exemplos mais recentes do fluxo de dados e realiza comparações entre distribuições de dados presentes em subjanelas de W com o objetivo de identificar diferenças nas distribuições e, consequentemente, mudanças de conceito. Considerando que $\hat{\mu}_W$ é a média observada dos valores presentes em W, o algoritmo verifica para cada divisão possível de W em duas subjanelas W_1 e W_2, se a diferença entre as médias dos valores nas subjanelas, denotadas por $\hat{\mu}_{W_1}$ e $\hat{\mu}_{W_2}$, é maior do que um limiar dinâmico ϵ_{ADWIN}, definido de acordo com o tamanho das subjanelas e os dados da janela. Assim, se $|\hat{\mu}_{W_1} - \hat{\mu}_{W_2}| \geq \epsilon_{ADWIN}$, uma mudança é detectada, os dados pertencentes à primeira subjanela são descartados e o algoritmo de detecção volta a monitorar novos dados. O descarte dos dados presentes em W_1 ocorre com o objetivo de utilizar apenas os dados mais recentes (W_2) para retreinar o modelo de classificação após a identificação da mudança. Por outro lado, se mudanças não são identificadas nos dados, a janela aumenta de tamanho automaticamente já que cada novo exemplo é armazenado na janela W.

O cálculo do limiar ϵ_{ADWIN} é dado pelo Equação 2.3, onde σ^2_W representa a variância calculada a partir dos dados contidos em W, n_1 e n_2 representam os tamanhos das subjanelas W_1 e W_2 que estão sendo comparadas, e m e δ' são calculados a partir das Equações 2.4 e 2.5, respectivamente. Como as verificações de mudanças consideram diferentes tamanhos de subjanelas, o limiar ϵ_{ADWIN} deve se ajustar dinamicamente aos tamanhos n_1 e n_2 para evitar problemas nos testes realizados.

$$\epsilon_{ADWIN} = \sqrt{\frac{2}{m}.\sigma^2_W.ln\frac{2}{\delta'}} + \frac{2}{3m}.ln\frac{2}{\delta'} \tag{2.3}$$

$$m = \frac{1}{\frac{1}{n_1} + \frac{1}{n_2}} \tag{2.4}$$

$$\delta' = \frac{\delta_{ADWIN}}{n} \tag{2.5}$$

O Algoritmo 2 ilustra em pseudo código o funcionamento do método ADWIN. Pode-se observar na entrada do Algoritmo 2, bem como na Equação 2.5, a presença do parâmetro δ_{ADWIN}. Este parâmetro tem o objetivo de impor um limitante superior para a taxa de falsos positivos detectados pelo método e está diretamente relacionado ao cálculo do limiar ϵ_{ADWIN} utilizado para a detecção de mudanças.

Algoritmo 2: ADaptive WINdowing – ADWIN

Entrada:
Fluxo de dados rotulados X
Limitante $\delta_{ADWIN} \in (0, 1)$
Saída:
Indicativo de mudança
Instante de tempo t em que a mudança foi identificada

```
1  início
2      para todo exemplo x_t em X faça
3          t ← InstanteTempoAtual;
4          W ← Inserir(x_t);
5          para cada possível divisão de W em W_1 e W_2 faça
6              μ̂_{W_1} ← Média(W_1);
7              μ̂_{W_2} ← Média(W_2);
8              ε_ADWIN ← CalcularLimiar(W_1, W_2, δ_ADWIN); /* Equação 2.3 */
9              se |μ̂_{W_1} − μ̂_{W_2}| ≥ ε_ADWIN então
10                 Mudança detectada;
11                 W ← Remover(W_1);
12                 retorna t;
13             fim
14         fim
15     fim
16 fim
```

É interessante notar que o método ADWIN foi originalmente desenvolvido para monitorar dados unidimensionais com valores binários ou contínuos entre o intervalo de valores $[0, 1]$. No caso binário, é possível monitorar os erros/acertos do classificador. No caso de valores contínuos, é possível monitorar tanto a taxa de erro do classificador como a distribuição dos dados, desde que estes valores sejam normalizados entre $[0, 1]$. Caso os dados sejam multidimensionais, é necessário aplicar o teste separadamente para cada uma das d dimensões do problema.

2.5 Considerações Finais

O objetivo deste capítulo foi apresentar uma revisão da literatura sobre classificação de fluxos de dados não estacionários. As principais definições sobre fluxos contínuos de dados e mudanças de conceito foram abordadas, bem como os tipos e as causas das mudanças, e as principais abordagens utilizadas por algoritmos adaptativos para lidar com mudanças de conceito. Basicamente, estes métodos seguem duas abordagens: i) em intervalos regulares de tempo, adaptam o modelo de classificação a partir da incorporação de dados mais recentes e o descarte dos dados mais antigos e potencialmente desatualizados; ii) a partir de indicadores que são continuamente monitorados ao longo do tempo, como a taxa de erro do classificador, identificam a presença de mudanças nos dados para então atualizar o modelo de classificação com os dados mais recentes.

Independente da abordagem utilizada pelos algoritmos para lidar com mudanças, ambas possuem uma suposição otimista que nem sempre pode ser cumprida em cenários reais: a disponibilização dos rótulos corretos referentes aos exemplos processados após a etapa de classificação com pouco ou nenhum tempo de atraso. Devido a inúmeros fatores, como alto custo no processo de rotulação, dependência de um especialista para a rotulação, falhas no processo de transmissão dos rótulos ou mesmo devido a características do processo gerador dos dados, as informações a respeito dos rótulos dos exemplos nem sempre podem ser obtidas. Desse modo, sem estes dados rotulados não é possível monitorar indicadores de desempenho do classificador para a identificação de mudanças e também não é possível utilizar estes dados para a atualização do modelo de classificação. O intervalo de tempo entre a classificação de um exemplo do fluxo e a disponibilização de seu rótulo correto pelo ambiente é denominado de latência. O próximo capítulo aborda os impactos de diferentes valores de latência em métodos de detecção de mudanças e as abordagens existentes para lidar com o caso de latência extrema, em que os dados rotulados nunca são disponibilizados.

Capítulo

3

Latência de Rótulos

3.1 Considerações Iniciais

Neste capítulo, será discutido um problema recorrente nas mais diversas aplicações do mundo real e pouco explorado na literatura de classificação de fluxos de dados não estacionários e que este trabalho possui especial interesse: *latência* para o recebimento de rótulos. Basicamente, a latência está relacionada ao atraso de tempo para a disponibilização dos rótulos corretos pelo ambiente, após a etapa de classificação de cada exemplo do fluxo de dados. Na Seção 3.2, é apresentada uma contextualização sobre o problema e as suas principais definições. Na Seção 3.3, é apresentada uma análise experimental do impacto de diferentes valores de latência em algoritmos estado da arte que utilizam métodos detectores de mudanças para a adaptação. Por fim, na Seção 3.4, são apresentados algoritmos da literatura que lidam com o cenário onde há total ausência de exemplos rotulados após a etapa de classificação, denominado de latência extrema.

3.2 Contextualização e Definições

Conforme discutido no Capítulo 2, a maior parte dos algoritmos da literatura que realiza a tarefa de classificação de fluxo de dados sob condições de mudanças de conceito, considera que após o processamento de cada exemplo do fluxo de dados, o seu rótulo correto é imediatamente disponibilizado pelo ambiente ou processo responsável pela geração dos dados sem qualquer atraso de tempo. Assim, é possível monitorar em tempo real indicadores de desempenho do classificador para a identificação de possíveis mudanças e reconstruir o modelo de classificação a partir de dados rotulados recebidos recentemente.

Esta suposição é perfeitamente factível para uma série de aplicações. Por exemplo, considere a tarefa de predizer por meio de algoritmos de aprendizado de máquina se o valor de uma determinada ação na bolsa de valores irá aumentar ou diminuir na próxima hora. Nesta aplicação, é possível obter o rótulo correto da predição com um atraso de exatamente uma hora após a predição, em condições ideais. Outros exemplos de aplicações reais são a predição do consumo de energia elétrica, predição de vazão de água em bacias hidrográficas, previsão de fenômenos naturais, entre outros. De um modo geral, em tarefas de classificação derivadas de problemas de regressão é possível obter os rótulos corretos de cada exemplo processado antes da chegada do próximo exemplo do fluxo.

Entretanto, devido a inúmeros fatores como altos custos envolvidos no processo de rotulação, dependência de um especialista para a rotulação, falhas no processo de transmissão dos rótulos, ou mesmo devido às características do processo gerador dos dados, as informações a respeito dos rótulos dos exemplos nem sempre podem ser obtidas em uma série de aplicações reais. Para melhor ilustrar esta dificuldade, considere os três exemplos de aplicações a seguir:

1. Um sensor óptico responsável por classificar a espécie de cada inseto que cruza por uma fonte de luz ao longo do tempo. Para realizar esta tarefa de classificação, são utilizadas informações referentes à batida de asas dos insetos obtidas após o seu cruzamento pela luz. Devido às variáveis climáticas como temperatura, umidade e pressão do ar, as características das espécies podem se alterar com o decorrer do tempo, sendo necessário que o algoritmo de classificação se adapte a estas mudanças. Entretanto, nesta aplicação é bastante difícil obter a informação referente ao rótulo da classe correta após a passagem de um inseto pelo sensor. No melhor cenário, é possível obter apenas uma pequena porção dos dados rotulados com o auxílio de um especialista responsável por analisar o sinal gerado. Ainda assim, a disponibilização destas informações para o algoritmo de classificação ocorrerá com um atraso considerável, já que, por questões de tempo e custo com o especialista, é preferível que seja disponibilizado para análise um conjunto de exemplos ao mesmo tempo e não a disponibilização de exemplos individuais ao longo do tempo. Este sensor é melhor discutido ao longo do trabalho no Capítulo 5. Diferentes aplicações que envolvem a classificação de dados provenientes de sensores podem apresentar dificuldades semelhantes;

2. Um sistema de prevenção e detecção de fraudes em transações financeiras, responsável por classificar cada transação em “fraudulenta” ou “legítima”. Este sistema considera informações referentes ao comportamento do cliente como datas e horários de pagamentos de contas, saques, transferências, valores máximos retirados, localização da transação, entre outras informações para classificar cada transação realizada ao longo do tempo. Além de ser um problema de classes desbalanceadas

(a quantidade de transações "legítimas" é potencialmente superior à quantidade de transações "fraudulentas"), nesta aplicação o sistema deve se adaptar automaticamente às mudanças de conceito do problema decorrentes de possíveis alterações no comportamento do usuário. Para a adaptação, o sistema deve considerar um atraso considerável na obtenção dos rótulos corretos das transações, já que transações "fraudulentas" não identificadas pelo sistema dependem da identificação pelo usuário que deve informar a fraude ao banco. Na maior parte dos casos, essa identificação pode levar dias. No caso de transações com cartão de crédito, a identificação pode ocorrer somente após a geração e recebimento da fatura do cartão de crédito. Em alguns casos, o sistema pode nunca receber esta informação caso o cliente não identifique a prática de fraude em sua conta. Aplicações semelhantes, como detecção de intrusão em sistemas ou detecção de falhas em processos industriais, também apresentam um cenário similar;

3. Sistemas de automação de veículos não tripulados. Por exemplo, dado um veículo aéreo não tripulado, inicialmente treinado em um ambiente conhecido em que os seus movimentos podem ser classificados em diferentes classes, o sistema de controle deve periodicamente adaptar o seu modelo para a realização dos movimentos mesmo na presença de mudanças em condições ambientais, como velocidade e direção do vento, altitude, temperatura e pressão atmosférica. Dado que essa adaptação é preferencialmente realizada de maneira não supervisionada ou que um especialista avalia determinados movimentos e fornece tais informações somente após a realização de uma série de movimentos, as informações de rótulos desta aplicação nunca serão disponibilizados ou serão disponibilizados com algum atraso para a maior parte dos dados gerados.

As três aplicações reais apresentadas são apenas exemplificações de situações em que o atraso dos rótulos deve ser considerado, mas é importante destacar que esta questão não se restringe somente a estes problemas. Desse modo, é mais factível que os algoritmos de classificação considerem a possibilidade de atraso na disponibilização dos rótulos corretos após a predição dos exemplos do fluxo. O processo geral considerado pelos algoritmos de classificação de fluxo de dados não estacionários, sejam eles com detectores de mudanças ou não, é apresentado na Figura 3.1. A figura também ilustra em qual ponto do processo ocorre a possibilidade de atraso na disponibilização dos rótulos corretos que pode variar entre $[0, \infty)$.

Este tempo entre o processamento de um exemplo pelo algoritmo de classificação e a disponibilização de seu rótulo correto pelo ambiente ou processo de geração dos dados é denominado *latência* (Marrs et al., 2010). Conforme visto na Figura 3.1, este tempo pode variar de acordo com o problema. Em especial, destacamos três cenários específicos:

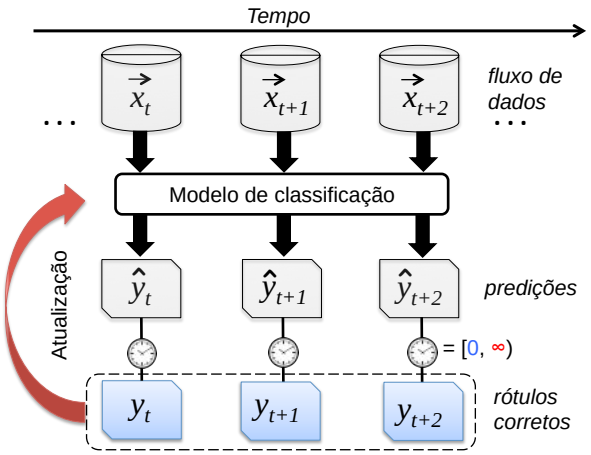


Figura 3.1: Processo geral considerado por algoritmos de classificação de fluxo de dados não estacionários. O tempo entre a predição de um exemplo e a disponibilização de seu rótulo correto pode variar entre $[0, \infty)$.

- **Latência nula:** cenário considerado pela maior parte dos trabalhos da literatura, em que o rótulo de cada exemplo é disponibilizado logo após o seu processamento sem qualquer atraso;
- **Latência extrema:** cenário em que os rótulos corretos dos exemplos processados nunca são disponibilizados para o algoritmo de classificação;
- **Latência intermediária:** cenário em que considera-se algum tempo de atraso t no recebimento dos rótulos corretos de modo que $0 < t < \infty$. A taxa de chegada dos rótulos pode ser fixa ou variável. Quando a taxa é variável, a ordem dos exemplos pode não ser respeitada. A contagem de tempo da taxa de chegada pode ser realizada em termos de número de exemplos (por exemplo, o rótulo correto y_t da instância $\vec{x}_t$ é recebido somente no instante de tempo $t + 5$, onde também recebe-se a instância $\vec{x}_{t+5}$) ou a partir de uma medida explícita de tempo (por exemplo, após t segundos, minutos, horas, dias, etc.). Além disso, existem casos em que o recebimento dos rótulos pode não seguir uma ordem cronológica. Por exemplo, pode-se receber o rótulo correto y_{t+5} do exemplo $\vec{x}_{t+5}$ antes do recebimento do rótulo correspondente ao exemplo $\vec{x}_{t+4}$.

A latência possui papel fundamental no ciclo de vida dos algoritmos de aprendizado *online*, principalmente quando são processados dados não estacionários. Para melhor compreensão, considere que o ciclo de vida destes algoritmos é composto por cinco estágios (Marrs et al., 2010):

1. **Coleta de dados iniciais.** Antes do funcionamento do algoritmo, se faz necessária a etapa de coleta de dados rotulados para a formação de um conjunto de dados de treinamento inicial;

2. **Indução inicial.** Após a coleta de dados iniciais, o algoritmo de aprendizado é aplicado e um primeiro classificador é gerado;

3. **Classificação.** Neste estágio, um novo exemplo $\vec{x_t}$ do fluxo é disponibilizado no tempo t e o classificador atual deve predizer um rótulo $\hat{y}_t$ correspondente a sua classe, formando o par $(\vec{x_t}, \hat{y}_t)$;

4. **Verificação e inclusão do exemplo na base.** No instante de tempo $u > t$, o rótulo correto y_t pode ser obtido. Assim, após um atraso de tempo $(u-t)$, é possível obter o par rotulado $(\vec{x_t}, y_t)$ e incluí-lo na base de treinamento;

5. **Atualização.** Periodicamente ou após o recebimento dos rótulos corretos de uma quantidade considerável de exemplos, o classificador é atualizado a partir do novo conjunto de treinamento.

Desse modo, nota-se que a latência possui impacto direto no processo de adaptação dos algoritmos de classificação. Embora exista um esforço considerável por décadas de pesquisa em explorar problemas relacionados a classificação de fluxo de dados, questões relacionadas a latência ainda foram pouco estudadas na literatura. De acordo com Krempl et al. (2014), a latência é um dos problemas em aberto na área de mineração de fluxo de dados, responsável por afetar o desempenho de classificadores no caso da presença de mudanças de conceito ao longo do tempo. Neste sentido, este trabalho busca contribuir na exploração do problema e na proposta de soluções.

3.3 Impacto da Latência em Métodos de Classificação

Com o objetivo de explicitar o impacto da latência no comportamento de algoritmos de classificação de fluxo de dados não estacionários, foi realizado um experimento que considera um cenário mais realístico onde se verifica a acurácia dos algoritmos dado diferentes valores para o tempo de atraso na disponibilização dos rótulos corretos dos exemplos processados. Especificamente, foi avaliado o comportamento do algoritmo de aprendizado *Naive Bayes* sem detector de mudanças de conceito e em conjunto com os métodos de detecção apresentados no Capítulo 2: DDM, EDDM, ADWIN e *Page Hinkley*. A escolha dessa configuração se deve ao fato de constituir uma abordagem simples e popularmente utilizada na literatura, além de apresentar resultados competitivos que são frequentemente utilizados na comparação de desempenho de novas propostas. Os resultados do algoritmo *Naive Bayes* sem detector de mudanças são utilizados apenas como base de comparação. Embora seja um classificador que absorve incrementalmente novos dados ao modelo de classificação ao longo do tempo, não é um classificador destinado a problemas com mudanças significativas.

Este experimento se baseia no trabalho de Marrs et al. (2010), no qual os autores avaliam o impacto de valores de latência no algoritmo baseado em árvores de decisão CD3 (Gao et al., 2007) em conjuntos de dados artificiais gerados pela ferramenta AutoUniv (Hickey, 2007). Neste estudo, considera-se somente três cenários, chamados de latência nula, latência média e latência alta. Desse modo, o experimento realizado nesta tese visa estender a análise feita por Marrs et al. (2010), bem como confirmar as suas conclusões para um número maior de conjunto de dados e algoritmos.

No sentido de explorar conjuntos de dados que são amplamente avaliados pela literatura em fluxo de dados, o experimento foi conduzido em cinco conjuntos de dados populares e com variados tipos de mudanças de conceito. Uma descrição de cada conjunto é apresentada a seguir:

- *Forest Covertype* (Oza e Russell, 2001; Gama et al., 2003). Este conjunto de dados é composto por 581.012 observações relativas ao tipo de cobertura florestal de uma determinada região. Cada observação é realizada em uma região de 30 metros quadrados e obtida pelo sistema de informação do *US Forest Service*. Os dados são descritos por um conjunto de 54 atributos numéricos que incluem o tipo do solo da região, informações sobre o perfil do terreno (aclive, declive, etc.), distância em relação a fonte de água mais próxima, entre outras informações, para classificar cada região em uma de 7 possíveis classes (*Spruce-Fir, Lodgepole Pine, Ponderosa Pine, Aspen, Douglas-Fir, Krummholz, Cottonwood-Willow*);

- *Electricity* (Harries e Wales, 1999; Gama et al., 2004). Este conjunto de dados contém 45.312 instâncias que descrevem a demanda de uso de energia elétrica no mercado australiano. Neste mercado, os preços são definidos a cada 5 minutos de acordo com a demanda de uso e os custos de fornecimento. Os dados possuem duas classes: *UP*, caso o valor atual tenha aumentado em relação a média móvel de valores nas últimas 24 horas; e *DOWN*, caso o valor atual tenha diminuído;

- *PokerHand* (Cattral et al., 2002). Este conjunto de dados contém 1 milhão de exemplos e 10 atributos numéricos/ordinais que descrevem as cartas presentes em partidas de *poker* (Texas Hold'em) ordenadas ao longo do tempo. Cada exemplo consiste na descrição das 5 cartas (de um total de 52 possíveis) de um jogador da partida. Cada carta é representada por 2 atributos referentes ao naipe e ao seu número. A classe atribuída a cada exemplo refere-se ao ranking obtido dado o conjunto de cartas ou *Poker Hand*. Alguns exemplos são *Royal Flush, Straight Flush*, Quadra, *Full House*, Trinca, entre outras possibilidades;

- *SensorStream* (Zhu, 2010). Conjunto de dados composto por 2.219.803 exemplos, 4 atributos numéricos e 54 classes. Este conjunto de dados contém informações sobre temperatura, umidade, iluminação e voltagem de 54 diferentes sensores utilizados

no laboratório de pesquisa da *Intel Berkeley* durante um período de 2 meses com leituras realizadas a cada 1 à 3 minutos. A tarefa de classificação deste conjunto de dados é identificar um dos 54 sensores (de acordo com o mapa apresentado na Figura 3.2), que apresentam mudanças de conceito ao longo do tempo. Por exemplo, em geral a iluminação durante os horários de trabalho é mais forte do que durante a noite. De mesmo modo, a temperatura da sala de reunião é mais elavada durante reuniões quando há uma quantidade maior de pessoas na sala.

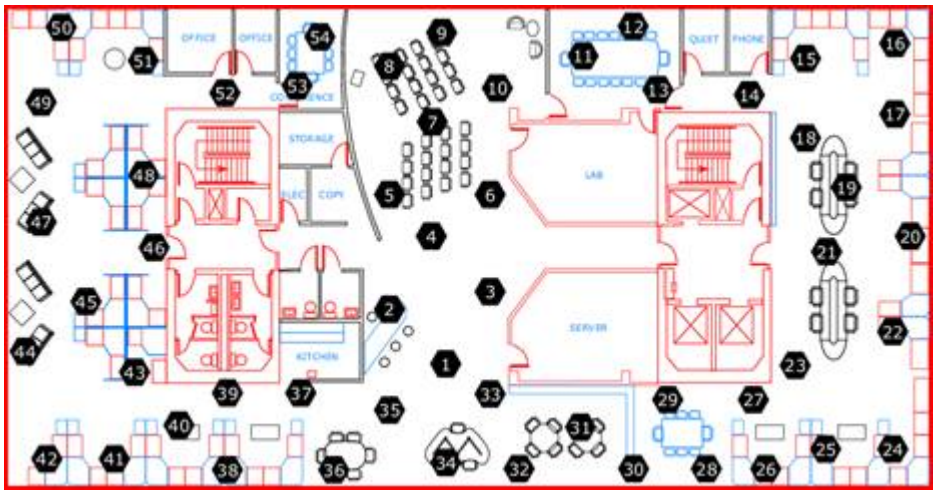


Figura 3.2: Mapa da localização dos 54 sensores dispostos no laboratório de pesquisa da *Intel Berkeley* que contemplam o conjunto de dados *SensorStream* (Zhu, 2010).

- *HyperPlane* (Hulten et al., 2001; Žliobaitė, 2009a). Este é um conjunto de dados sintético com mudanças de conceito graduais composto por 2 classes, 10 atributos numéricos e 10.000 exemplos. Neste conjunto de dados, os exemplos estão dispostos em um hiperplano com 10 dimensões onde a fronteira de separação entre as classes rotaciona gradualmente ao longo do tempo em sentido horário. Especificamente 8 de um total de 10 dimensões sofrem alterações na fronteira de separação das classes a cada 500 exemplos. A Figura 3.3 ilustra as mudanças de conceito neste conjunto de dados em uma simplificação bidimensional dos dados.

Em relação a latência (l), foi avaliado a variação dos valores de l entre 0 e 2000 instâncias. Por exemplo, dado um valor de latência $l = 10$, considera-se que o rótulo y_t do exemplo $\vec{x}_t$ será disponibilizado ao algoritmo de classificação somente no instante de tempo $t+10$ e após o recebimento e predição dos exemplos $\{\vec{x}_{t+1}, \vec{x}_{t+2}, \ldots, \vec{x}_{t+10}\}$. Assim, há um atraso de 10 exemplos para o recebimento do rótulo de cada exemplo. Embora em aplicações reais o tempo de atraso possa variar para cada exemplo, fixou-se um mesmo valor para todas as instâncias do fluxo para facilitar a análise dos resultados.

As acurácias dos classificadores em relação a diferentes valores de latência para os conjuntos de dados *Forest Covertype, Electricity, HyperPlane, PokerHand* e *SensorStream*

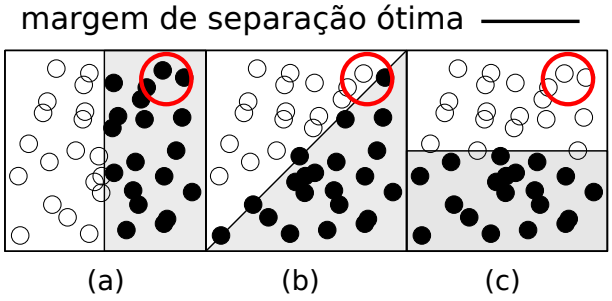
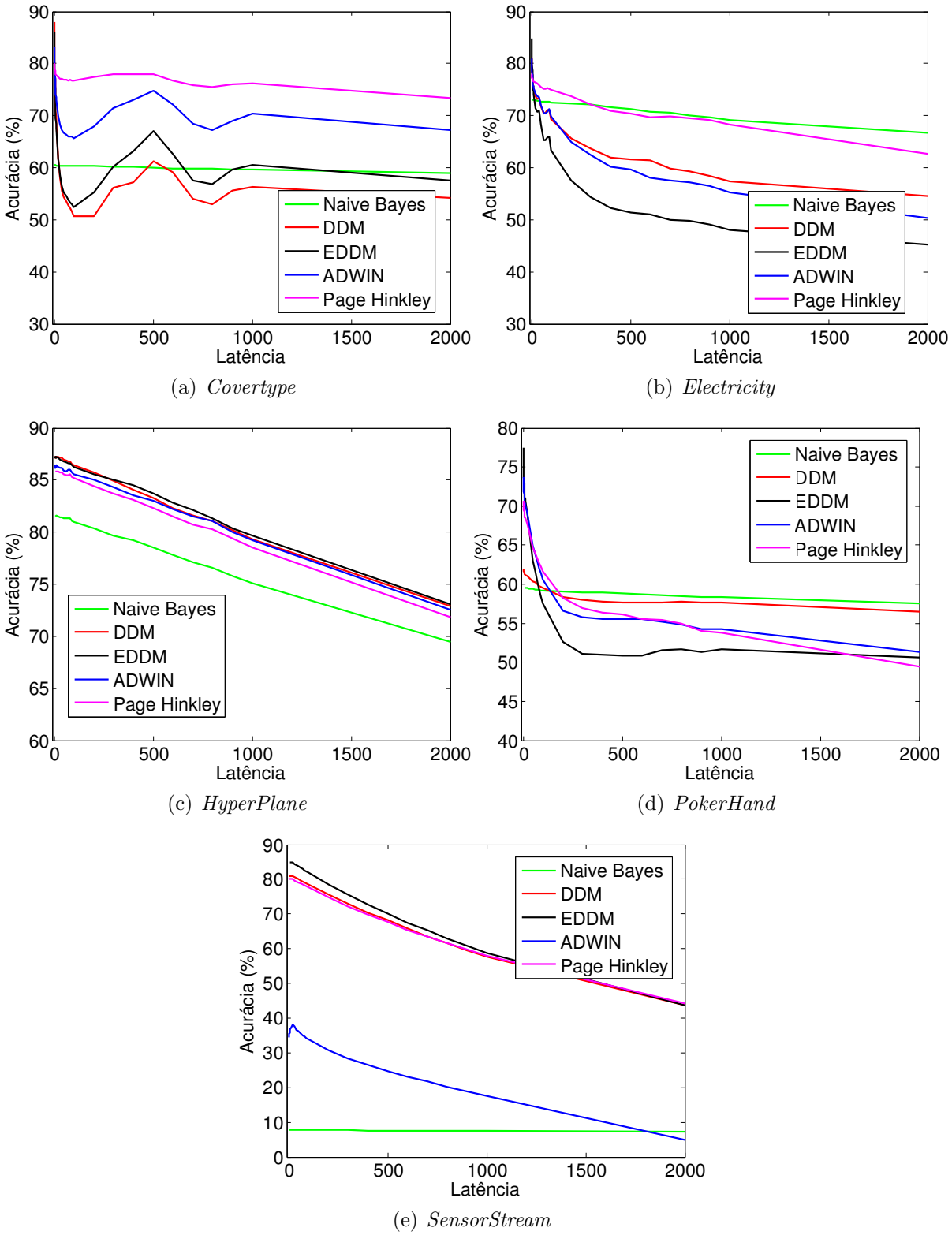


Figura 3.3: Evolução do conjunto de dados sintético *HyperPlane* ao longo do tempo. O conjunto possui duas classes representadas pelos símbolos ∘ e •. Em (a) é apresentada a disposição inicial das classes. Em (b) é apresentado o conjunto após uma rotação de 45^o na fronteira de separação das classes. Por fim, em (c) é apresentado o conjunto após uma rotação de 90^o. Pode-se observar que para uma determinada área fixada no espaço de atributos, representada por um circulo em vermelho, a classe correta desta região se altera ao longo do tempo (Žliobaitė, 2009a).

são apresentadas na Figura 3.4. Dada a variação observada em todos os resultados dos algoritmos que utilizam detectores de mudanças de conceito de acordo com diferentes valores de latência, nota-se a significativa influência do atraso na disponibilização dos rótulos corretos no desempenho dos algoritmos. Além do atraso na detecção de mudanças pelos métodos, após a detecção os algoritmos atualizam o modelo de classificação com dados potencialmente desatualizados, o que justifica os resultados apresentados.

Nota-se na Figura 3.4 que o classificador *Naive Bayes* sem detector de mudanças apresenta os resultados mais estáveis entre todos os algoritmos avaliados. Entretanto, somente nos conjunto de dados *Electricity* e *PokerHand*, os resultados são estáveis e satisfatórios independente do valor de latência considerado. Estes resultados se justificam pelo fato desta abordagem apenas adicionar incrementalmente novos exemplos rotulados com atraso de tempo, mas sem o descarte de exemplos possivelmente desatualizados. Embora ocorram mudanças nestas bases, boa parte dos conceitos antigos ainda são úteis para a tarefa de classificação. Por isso, os algoritmos que detectam mudanças com maior frequência, atualizam o modelo de classificação com dados mais recentes e realizam o descarte de dados mais antigos, apresentam menor acurácia nestes conjuntos de dados.

Para melhor resumir os resultados apresentados na Figura 3.4, é apresentado na Tabela 3.1 um comparativo entre os resultados obtidos por cada algoritmo com a configuração tradicional considerada pelos algoritmos da literatura onde a latência é nula (os rótulos corretos dos exemplos processados são imediatamente disponibilizados ao algoritmo de aprendizado) e o pior resultado obtido pelo mesmo algoritmo dado um valor de latência diferente de zero. Além disso, a Tabela 3.1 permite a consulta com precisão dos valores de acurácia obtidos por cada algoritmo com a configuração de latência nula. Para cada configuração (latência nula ou pior caso) e conjunto de dados, o melhor resultado obtido por um algoritmo está em **negrito** e o pior resultado está sublinhado. Na tabela, o al-



(a) *Covertype* (b) *Electricity*

(c) *HyperPlane* (d) *PokerHand*

(e) *SensorStream*

Figura 3.4: Impacto na acurácia de acordo com a variação da latência no recebimento dos rótulos.

goritmo *Naive Bayes* sem detecção de mudanças é referenciado como *NB* e o método de detecção de mudanças *Page Hinkley* é referenciado como *PHT*.

Tabela 3.1: Comparativo entre os resultados obtidos pelos métodos de classificação com a configuração de latência nula e o pior resultado obtido pelos métodos dado um valor de latência diferente de zero.

	Covertype		*Electricity*		*HyperPlane*		*PokerHand*		*SensorStream*	
	Lat. nula	Pior caso	Lat. nula	Pior caso	Lat. nula	Pior caso	Lat. nula	Pior caso	Lat. nula	Pior caso
NB	60,50	59,00	73,40	**66,70**	81,60	69,50	59,60	**57,50**	7,70	7,30
DDM	**88,00**	50,60	81,20	54,60	**87,20**	72,90	62,00	56,50	80,80	43,60
EDDM	86,10	52,40	**84,80**	45,20	**87,20**	**73,10**	**77,50**	50,60	**84,70**	43,70
ADWIN	83,20	65,70	81,00	50,40	86,40	72,50	73,70	51,30	34,40	5,00
PHT	80,10	**73,30**	78,00	62,70	85,90	71,80	70,70	49,50	80,00	**44,30**

Como o principal objetivo deste experimento é explicitar o impacto da latência nos algoritmos de classificação de um modo geral, não será discutido o comportamento de cada método de acordo com as características de cada conjunto de dados e os valores de latência. A principal conclusão deste experimento é a constatação de que para a maior parte dos métodos e conjuntos de dados, a latência para o recebimento dos rótulos é responsável por alterar significativamente os resultados obtidos. Considere, por exemplo, os resultados obtidos pelo método *Drift Detection Method* (DDM) no conjunto de dados *Covertype*. Quando considerado o cenário ideal de latência nula, o método apresenta acurácia de 88%, sendo o melhor resultado entre os métodos avaliados. Entretanto, se o problema possuísse uma latência $l = 100$, a acurácia seria drasticamente reduzida para $50,60\%$, sendo o pior resultado entre os métodos avaliados. Desse modo, dado o impacto e a recorrência desta característica em uma série de problemas do mundo real, nota-se que a maior parte das propostas de novos métodos de classificação de fluxo de dados não estacionários da literatura, sejam com detectores de mudanças ou não, com o uso de janelas deslizantes ou incrementais, a partir de classificadores isolados ou comitê de classificadores, tem negligenciado este importante fator.

3.4 Latência Extrema

Este trabalho tem um interesse especial no cenário de latência extrema. Neste cenário, os rótulos corretos dos exemplos processados nunca são disponibilizados ao algoritmo de classificação. Desse modo, o algoritmo não é capaz de seguir o procedimento tradicional de monitoramento de indicadores de desempenho ou das distribuições dos dados para a identificação de mudanças. Ainda mais, mesmo que o algoritmo opte por adaptar-se em intervalos regulares de tempo sem que uma mudança seja explicitamente detectada, não há a presença de dados rotulados e recentes para a atualização do modelo de classificação que está potencialmente defasado. O interesse deste trabalho em relação a este cenário se deve a aplicação estudada que envolve a classificação de espécies de insetos por meio

de sensores laser. Embora seja possível obter uma pequena porção de dados rotulados com o auxílio de um especialista, a adaptação do modelo de classificação em ambientes não estacionários de maneira totalmente não supervisionada reduziria consideravelmente os custos envolvidos para o funcionamento do sensor em campo.

Ao considerar o cenário onde há a ausência total na obtenção de rótulos corretos após a classificação dos exemplos, poucos trabalhos da literatura atendem este requisito. Basicamente, foram encontrados dois trabalhos que realizam tal tarefa: *Arbitrary Sub-Populations Tracker* – APT (Krempl, 2011) e *Compacted Object Sample Extraction* – COMPOSE (Dyer et al., 2014). Estes algoritmos são apresentados nas Subseções 3.4.1 e 3.4.2, respectivamente.

É importante ressaltar que para que seja possível a adaptação do algoritmo ao longo do tempo neste cenário, algumas restrições são impostas em relação aos tipos de mudanças de conceito. Em primeiro lugar, as mudanças devem ocorrer somente em atributos observáveis dos dados. Ou seja, mudanças que ocorrem devido a fatores externos em relação aos atributos observáveis do problema e que são responsáveis por alterar a fronteira de separação entre as classes, são difíceis ou impossíveis de serem identificados por qualquer algoritmo que não tem conhecimento de dados rotulados. Por exemplo, no conjunto de dados *HyperPlane* ilustrado na Figura 3.3, as mudanças ocorrem somente na fronteira de separação das classes. A segunda restrição é em relação a velocidade das mudanças. Como o mecanismo de adaptação dos algoritmos neste cenário é incremental, as mudanças não devem ser abruptas. Além disso, por ser um problema de classificação considera-se a presença de um conjunto de dados inicial rotulado utilizado para o treinamento do algoritmo. Este conjunto é fornecido somente na fase inicial do algoritmo e é necessário para definir a disposição inicial das classes no espaço de atributos.

3.4.1 Algoritmo *Arbitrary Sub-Populations Tracker*

O algoritmo *Arbitrary Sub-Populations Tracker* – APT proposto por Krempl (2011) se adapta às mudanças de conceito incrementais ao realizar o rastreamento da movimentação de subpopulações das classes do conjunto de dados com distribuições de probabilidades arbitrárias. Entende-se por população o conjunto de todas as funções geradoras de exemplos no espaço multidimensional. Cada função pertencente a este conjunto é uma subpopulação. O número de subpopulações existente é sempre maior ou igual ao número de classes, havendo ao menos uma subpopulação para cada classe. Assim, diferentes subpopulações podem gerar exemplos de uma mesma classe. Esse conceito permite que uma classe seja a composição de diferentes grupos, cada um com sua própria distribuição de probabilidade, ou seja, formato e densidade.

O funcionamento do algoritmo é dividido em uma fase inicial, executada uma única vez sobre os N exemplos rotulados fornecidos como entrada para o algoritmo, e outra fase

iterativa, executada a cada M exemplos não rotulados processados ao longo do tempo. A fase inicial consiste em encontrar as subpopulações responsáveis por gerar os N exemplos rotulados. Os exemplos pertencentes a cada subpopulação (em conjunto com parâmetros específicos) representam um *kernel*, capaz de estimar uma distribuição de probabilidade para tal subpopulação. Para o caso trivial, no qual é considerado que a quantidade de subpopulações é a mesma que a de classes, cada subpopulação é constituída pelo conjunto de todos os exemplos de uma mesma classe. Para os demais casos, são usadas técnicas de agrupamento de dados a fim de dividir os exemplos em grupos, sendo cada grupo então considerado uma subpopulação. A fase iterativa tem a função de identificar uma relação $1-1$ entre cada um dos M_{t-1} exemplos da iteração anterior (que, no caso da primeira iteração, têm seus rótulos corretos conhecidos) e dos M_t exemplos a serem classificados. Essa relação pressupõe a existência, para cada exemplo da iteração anterior, de um correspondente na iteração seguinte. Uma vez que as correspondências são feitas, o evento a ser classificado herda as características de seu par: classe e subpopulação a que pertence. Para definir as relações em cada iteração, é executado o método *Expectation-Maximization* (Moon, 1996).

Os autores sugerem o uso de validação cruzada para estimar os parâmetros existentes. A falta de capacidade do algoritmo em estimar os parâmetros utilizando a pequena quantidade de dados rotulados fornecidos inicialmente, dificulta seu uso prático. Além disso, o algoritmo exige que o número de eventos em cada subpopulação seja constante ao longo das iterações, ou seja, que a probabilidade *a priori* das classes seja estática. Esta é uma restrição que dificilmente pode ser cumprida em problemas reais. Além disso, a taxa de mudança deve ser constante ao longo do tempo. Por fim, a constante execução do método *Expectation-Maximization* ao longo de suas iterações resulta em um elevado custo para sua execução.

3.4.2 Algoritmo *Compacted Object Sample Extraction*

Para lidar com a classificação de fluxo de dados não estacionários sem o conhecimento dos rótulos corretos ao longo do tempo, o algoritmo *Compacted Object Sample Extraction* – COMPOSE proposto por Dyer et al. (2014) combina técnicas de aprendizado semissupervisionado e geometria computacional para se adaptar incrementalmente às mudanças que ocorrem nos dados ao longo do tempo. O funcionamento do algoritmo pode ser dividido em seis etapas básicas conforme ilustrado na Figura 3.5, as quais são discutidas a seguir.

Antes da etapa de classificação, o algoritmo recebe um conjunto de dados iniciais rotulados $\mathcal{L}^0$. Conforme discutido anteriormente, este conjunto de dados é um pré-requisito dos algoritmos que lidam com latência extrema para que seja possível definir o problema de classificação. Na segunda etapa é recebido um conjunto de dados não rotulados $\mathcal{U}^t$ no

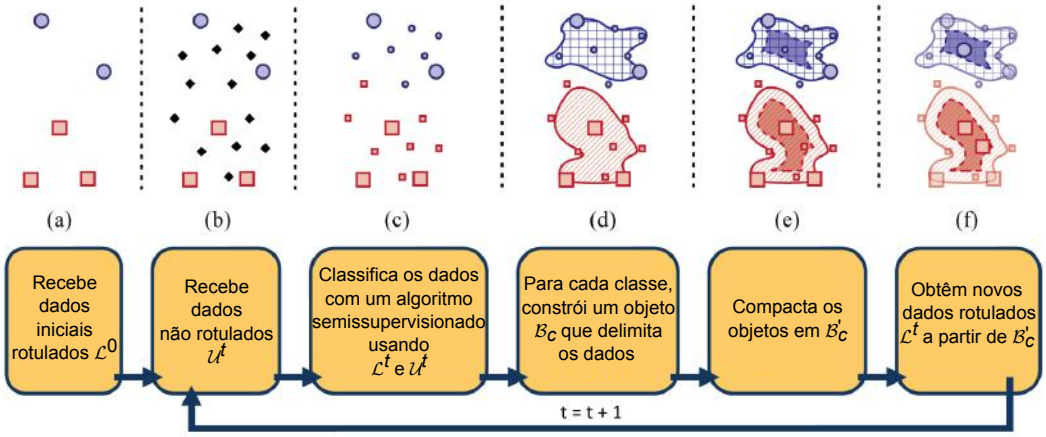


Figura 3.5: Etapas realizadas pelo algoritmo COMPOSE para lidar com fluxos de dados não estacionários sob condições de latência extrema (Dyer et al., 2014).

instante de tempo t, o qual pretende-se predizer os seus respectivos rótulos de classe na próximo etapa. Na etapa de classificação o algoritmo combina os dados rotulados $\mathcal{L}^t$ e não rotulados $\mathcal{U}^t$ em um algoritmo semissupervisionado transdutivo para que sejam calculados os rótulos de $\mathcal{U}^t$. Na próxima etapa é aplicado um algoritmo de fecho convexo para cada uma das classes do problema, considerando que as classes são constituídas por dados rotulados e exemplos classificados na etapa anterior. Esta etapa gera um objeto $\mathcal{B}_C$ que delimita um "envelope" que contorna os dados no espaço de atributos para cada uma das C classes do problema. Estes objetos são chamados de α-shape e visam representar a distribuição condicional das classes. Na próxima etapa, cada objeto $\mathcal{B}_C$ é compactado gerando um outro objeto $\mathcal{B}'_C$ com um envelope interno menor. O objetivo da compactação é extrair exemplos representativos (*core supports*) do centro geométrico de cada objeto. O algoritmo assume que os *core supports* extraídos de $\mathcal{B}'_C$ são exemplos com alta probabilidade de terem sido corretamente classificados. Todo o processo é repetido iterativamente quando novos dados não rotulados chegam, de modo que os *core supports* da iteração anterior passam a ser considerados exemplos rotulados da iteração atual e auxiliam na etapa de classificação por meio de algoritmos transdutivos.

Como os algoritmos transdutivos assumem uma quantidade razoável de exemplos rotulados e não rotulados, o algoritmo COMPOSE considera que um lote de exemplos é recebido a cada instante de tempo. Desse modo, o algoritmo COMPOSE não pode ser diretamente aplicado em problemas onde cada exemplo do fluxo de dados chega individualmente para classificação em diferentes instantes de tempo. Além disso, o algoritmo possui dois importantes parâmetros: α e CP. O primeiro está relacionado ao nível de detalhe do envelope. Para um valor alto de α, o envelope obtido é o fecho convexo. Valores baixos ocasionam formas não convexas ou mesmo desconexas. Considere um problema com duas classes representadas pelos símbolos • e ◊, conforme ilustrado na Figura 3.6.

Na figura também é possível observar os diferentes níveis de detalhamento do envelope dada a variação do parâmetro α para a construção do α-shape da classe $\Diamond$. No caso da ilustração, o valor ótimo é $\alpha = 0.1$. Entretanto, este valor não pode ser generalizado, sendo necessário encontrar um valor razoável para cada problema.

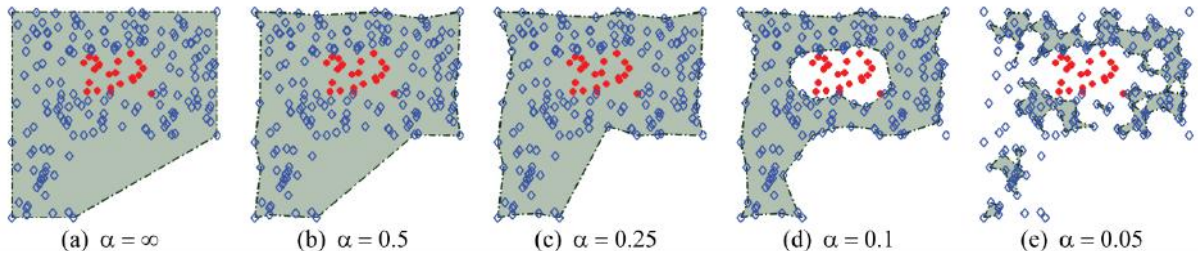


Figura 3.6: Variação do parâmetro α na construção do envelope α-shape considerando os exemplos da classe $\Diamond$ (Dyer et al., 2014).

O segundo parâmetro CP está relacionado ao nível de compactação aplicado ao envelope obtido. Nota-se que ambos os parâmetros influenciam o desempenho do algoritmo, principalmente o parâmetro CP. Caso o nível de compactação do envelope seja muito elevado, exemplos relevantes podem ser descartados. Em contrapartida, a baixa compactação pode ocasionar sobreposição de envelopes referentes a diferentes classes. Além disso, o processo de construção do envelope possui custo computacional exponencial com relação à quantidade de dimensões dos dados, o que se torna um problema ao processar fluxo de dados de alta dimensionalidade. O cálculo do envelope dos dados leva em consideração o uso algoritmo fecho convexo. Para dados com duas ou três dimensões, são conhecidos algoritmos capazes de calcular o fecho convexo em tempo $O(n \log h)$, onde n é a quantidade exemplos e h é a quantidade de dimensões. Para problemas com mais de três dimensões, o tempo para calcular o fecho convexo dos dados é de $O(n^{\lfloor d/2 \rfloor})$ (Chazelle, 1993). Assim, o seu uso se torna impraticável a partir de uma quantidade razoável de dimensões, característica comum em problemas de classificação.

3.5 Considerações Finais

Neste capítulo, foi apresentada uma importante fase no processo de aprendizado *online* de algoritmos que lidam com fluxos de dados não estacionários que é o atraso de tempo na obtenção dos rótulos corretos dos exemplos processados, fornecido pelo processo gerador dos dados. Tal atraso é formalmente denominado *latência* e muitas vezes é negligenciado pelos algoritmos da literatura. Conforme discutido, a maior parte dos algoritmos considera que exemplos rotulados são imediatamente disponibilizados após a sua classificação sem qualquer atraso de tempo. Entretanto, foram apresentados diferentes cenários onde esta suposição otimista não pode ser cumprida. Ainda mais, a partir de um experimento com

conjuntos de dados reais e sintéticos em que se considerou diferentes valores de latência, foi possível verificar e discutir como este fator é responsável por alterar significativamente a acurácia de algoritmos estado-da-arte na literatura, sendo necessário o uso de novas abordagens.

Também foi discutido o caso especial em que há total ausência na disponibilização de exemplos rotulados após a etapa de classificação, denominado *latência extrema*. Neste cenário, há o conhecimento de dois algoritmos que lidam com este problema: APT e COMPOSE. Embora sejam algoritmos pioneiros na resolução deste problema, nota-se que ambos possuem limitações em suas abordagens, principalmente em relação à definição de parâmetros e custo computacional. Visando mitigar estes problemas, são apresentados no Capítulo 4 duas soluções propostas neste trabalho para a classificação de fluxo de dados não estacionários sob condições de latência extrema.

Capítulo

4

Soluções Propostas para a Classificação de Fluxos de Dados Não Estacionários e Latência Extrema

4.1 Considerações Iniciais

O problema de classificação de fluxo de dados não estacionários tem recebido grande atenção de pesquisadores da área de Mineração de Dados e Aprendizado de Máquina nos últimos anos. Isto se deve, principalmente, ao aumento da quantidade de aplicações do mundo real que lidam com dados gerados continuamente em alta velocidade, grande volume e que são suscetíveis à mudanças em suas características ao longo do tempo. Por exemplo, praticamente qualquer pessoa que esteja portando um *smartphone* é capaz de gerar uma enorme quantidade de dados diariamente e muitos destes dados estão relacionados ao comportamento das pessoas que evoluem ao longo do tempo.

Conforme discutido no Capitulo 3, a maior parte dos algoritmos de classificação de fluxo de dados assume um cenário otimista de latência nula que nem sempre pode ser cumprido em aplicações do mundo real. Assim, são poucos os trabalhos da literatura que consideram algum tempo de atraso para a disponibilização (pelo processo gerador dos dados) dos rótulos corretos dos exemplos preditos. No cenário mais difícil e pessimista, denominado *latência extrema*, se encontram os trabalhos que assumem a total ausência na disponibilização destes dados. Infelizmente, estas abordagens possuem limitações relacionadas à definição correta de parâmetros ou um alto custo computacional. Neste sentido, visando mitigar os problemas encontrados nas abordagens estado-da-arte, são apresenta-

das neste capítulo duas propostas de algoritmos desenvolvidos neste trabalho para lidar com a classificação de fluxo de dados não estacionários e latência extrema. É importante destacar que por lidar com o cenário mais pessimista, a concepção destes métodos também é importante para cenários mais simples, em que dados rotulados são disponibilizados após algum tempo de atraso, sendo necessário apenas simples modificações para que estes dados sejam considerados, o que possivelmente melhoraria o desempenho dos algoritmos propostos. Entretanto, estas modificações fogem do escopo desta tese e fazem parte dos futuros trabalhos.

Para avaliar os algoritmos sob determinadas situações de mudanças, foram construídos dados sintéticos de *benchmark* conforme apresentado na Seção 4.2. Os algoritmos também foram avaliados em dados reais com mudanças de conceito incrementais. Estes dados são apresentados na Seção 4.3. As propostas intituladas *Stream Classification Algorithm Guided by Clustering – SCARGC* (Souza et al., 2015c) e *Micro-Cluster Classification – MClassification* (Souza et al., 2015b) são apresentadas nas Seções 4.4 e 4.5, respectivamente. Por fim, a avaliação experimental de ambas as propostas é apresentada na Seção 4.6.

4.2 Construção de Dados Sintéticos de *Benchmark*

O uso de dados sintéticos é interessante para avaliar comportamentos específicos dos dados que podem ocorrer em situações reais. Diferentemente de conjuntos de dados reais, em dados artificiais é possível saber com precisão o tipo e o momento exato em que as mudanças ocorrem. Desse modo, pode-se avaliar a capacidade de adaptação de um algoritmo de classificação sob determinadas situações. Além disso, o uso de um conjunto de dados de *benchmark*, auxilia a avaliação comparativa entre diferentes algoritmos. Para avaliar os métodos propostos neste trabalho foram gerados 10 conjuntos de dados sintéticos com mudanças de conceito incrementais ao longo do tempo. Nestes dados não é considerada a presença de atributos "ocultos" responsáveis por mudanças na distribuição dos dados. Assim, todas as mudanças são visíveis no espaço de atributos dos dados. Ou seja, não ocorrem mudanças isoladamente nas probabilidades condicionais *a posteriori* $P(y_i|\vec{x})$ sem que sejam observadas mudanças nas distribuições $P(\vec{x}|y_i)$ ou $P(\vec{x})$. Por simplificação, nos conjuntos de dados as probabilidades $P(y_i)$ se mantêm constantes ao longo do tempo. Para a ampla divulgação e utilização destes conjuntos de dados por outros pesquisadores interessados, foi criado um *website*[1] onde é possível adquirir os conjuntos de dados de *benchmark* e obter mais informações sobre o comportamento das mudanças ao longo do tempo a partir de uma descrição visual em vídeo.

Uma descrição resumida dos dados é apresentada na Tabela 4.1. A coluna *Mudanças* indica o intervalo em termos de número de exemplos para a ocorrência de mudanças

[1] https://sites.google.com/site/nonstationaryarchive

nos dados. Por exemplo, o conjunto de dados 4CR possui um total de 144.400 exemplos distribuídos em 4 classes que se alteram a cada 400 exemplos. Desse modo, há um total de 361 mudanças durante o período de tempo em que os dados do conjunto 4CR são gerados. Assim, um valor menor representa maior frequência de mudanças ao longo do tempo. Nota-se que os dados possuem diferentes características em termos de quantidade de classes e atributos, número de exemplos e mudanças. Além dos 10 conjuntos de dados sintéticos propostos neste trabalho, também foram utilizados 6 conjuntos de dados propostos por Dyer et al. (2014): FG-2C-2D, UG-2C-2D, UG-2C-3D, UG-2C- 5D, MG-2C-2D e GEARS-2C-2D. Os conjuntos de dados UG-2C-2D, UG-2C-3D e MG-2C-2D foram originalmente propostos por Dyer et al. (2014). Os conjuntos UG-2C-5D e GEARS-2C-2D foram gentilmente cedidos pelos autores do mesmo artigo.

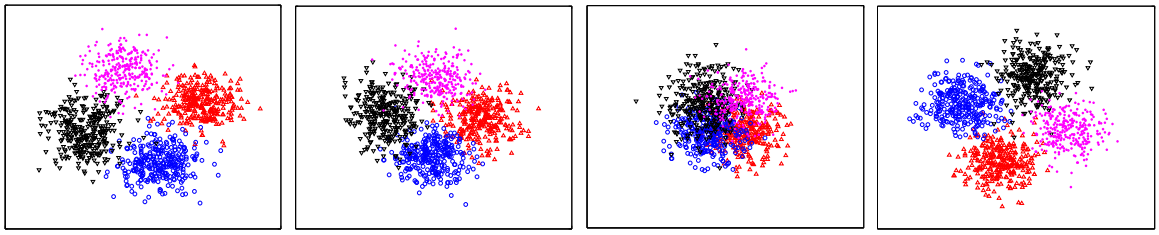
Tabela 4.1: Descrição dos conjuntos de dados sintéticos. Em https://sites.google.com/site/nonstationaryarchive é apresentada uma descrição visual detalhada dos dados.

Nome	**#Classes**	**#Atributos**	**Mudanças**	**Tamanho**
1CDT	2	2	400	16.000
2CDT	2	2	400	16.000
1CHT	2	2	400	16.000
2CHT	2	2	400	16.000
4CR	4	2	400	144.400
4CRE-V1	4	2	1.000	125.000
4CRE-V2	4	2	1.000	183.000
5CVT	5	2	1.000	40.000
1CSurr	2	2	≈600	55.283
4CE1CF	5	2	750	173.250
FG-2C-2D	2	2	2.000	200.000
UG-2C-2D	2	2	1.000	100.000
UG-2C-3D	2	3	2.000	200.000
UG-2C-5D	2	5	2.000	200.000
MG-2C-2D	2	2	2.000	200.000
GEARS-2C-2D	2	2	2.000	200.000

Os nomes dos conjuntos de dados são relacionados com os movimentos no espaço de atributos realizados pelas classes ao longo do tempo. Por exemplo, *One Class Diagonal Translation (1CDT), Two Classes Horizontal Translation (2CHT), Four Classes Rotating (4CR), Four Classes Rotating with Expansion (4CRE), Five Classes Vertical Translation (5CVT), One Class Surrounding another Class (1CSurr), Four Classes Expanding and One Class Fixed (4CE1CF), Two Bidimensional Unimodal Gaussian Classes (UG-2C-2D), Two Bidimensional Mulitimodal Gaussian Classes (MG-2C-2D), Two Rotating Gears (GEARS-2C-2D).*

Para melhor compreensão das mudanças presentes em alguns destes conjuntos de dados, serão apresentadas ilustrações que representam a evolução dos dados ao longo do

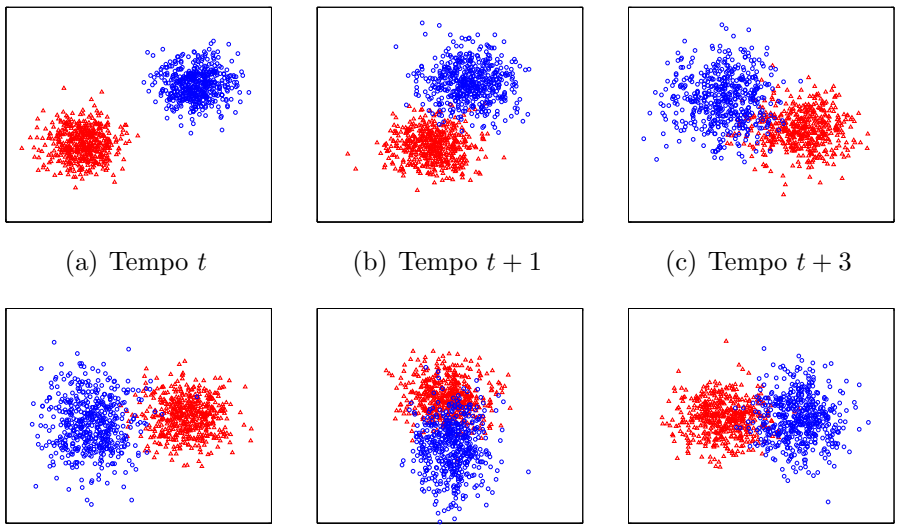
tempo. Por exemplo, na Figura 4.1 é ilustrado o comportamento do conjunto de dados 4CRE-V2 em 4 diferentes instantes de tempo. Neste conjunto de dados, 4 classes (representadas por $\triangledown$, $\bullet$, $\triangle$, $\circ$) giram gradativamente em torno de um eixo central no sentido horário. Além disso, em determinados momentos as classes se posicionam mais próximas entre si de modo que ocorre uma considerável sobreposição dos dados, como evidenciado na Figura 4.1-*c*). No total, as classes realizam 3 rotações completas e retornam para a posição definida inicialmente.



(a) Tempo t (b) Tempo $t+1$ (c) Tempo $t+2$ (d) Tempo $t+3$

Figura 4.1: Quatro diferentes momentos do conjunto de dados 4CRE-V2.

Na Figura 4.2 é ilustrada a movimentação das duas classes do conjunto de dados UG-2C-2D no espaço de atributos bidimensional. Assim como no conjunto de dados 4CRE-V2, as classes do conjunto UG-2C-2D são geradas a partir de uma distribuição Gaussiana e se movimentam em sentidos opostos ao longo do tempo.



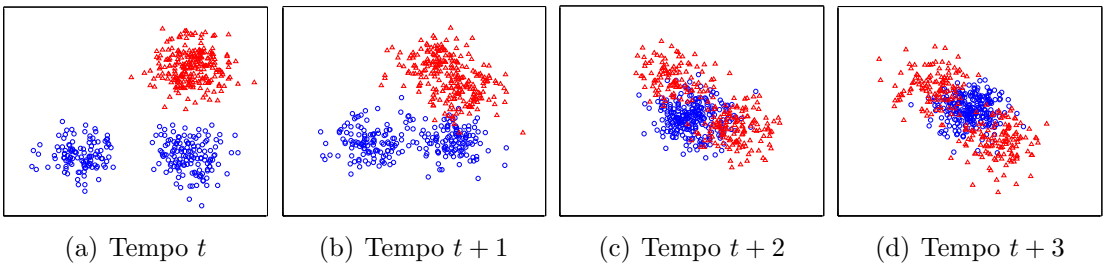
(a) Tempo t (b) Tempo $t+1$ (c) Tempo $t+3$

(d) Tempo $t+4$ (e) Tempo $t+5$ (f) Tempo $t+6$

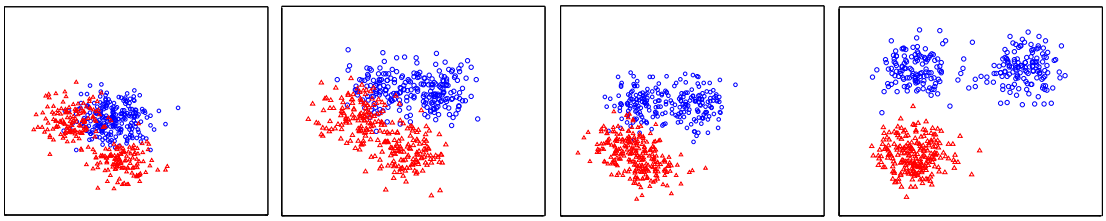
Figura 4.2: Seis diferentes momentos do conjunto de dados UG-2C-2D.

A Figura 4.3 ilustra o comportamento das classes no conjunto de dados MG-2C-2D em 8 diferentes instantes de tempo. Este conjunto de dados é formado por duas classes representadas pelos símbolos $\circ$ e $\triangle$. Inicialmente, os dados da classe $\circ$ estão dispostos

de acordo uma distribuição Gaussiana bimodal, enquanto os dados da classe △ estão dispostos de acordo com uma distribuição Gaussiana unimodal. Ao longo do tempo, a distribuição dos dados da classe △ se torna bimodal e em seguida volta a ser unimodal. Do mesmo modo, a distribuição dos dados da classe ○ se torna unimodal e em seguida volta a ser bimodal. Em determinados instantes de tempo, ambas as classes apresentam distribuições bimodais simultaneamente, como pode ser observado na Figura 4.3-*f*) e Figura 4.3-*g*). De maneira incremental, as duas classes se deslocam em sentidos opostos de modo que os dados da classe △ envolvem os dados da classe ○ com um considerável nível de sobreposição entre as classes, como pode ser observado na Figura 4.3-*c*), *d*) e *e*).



(a) Tempo t (b) Tempo $t+1$ (c) Tempo $t+2$ (d) Tempo $t+3$



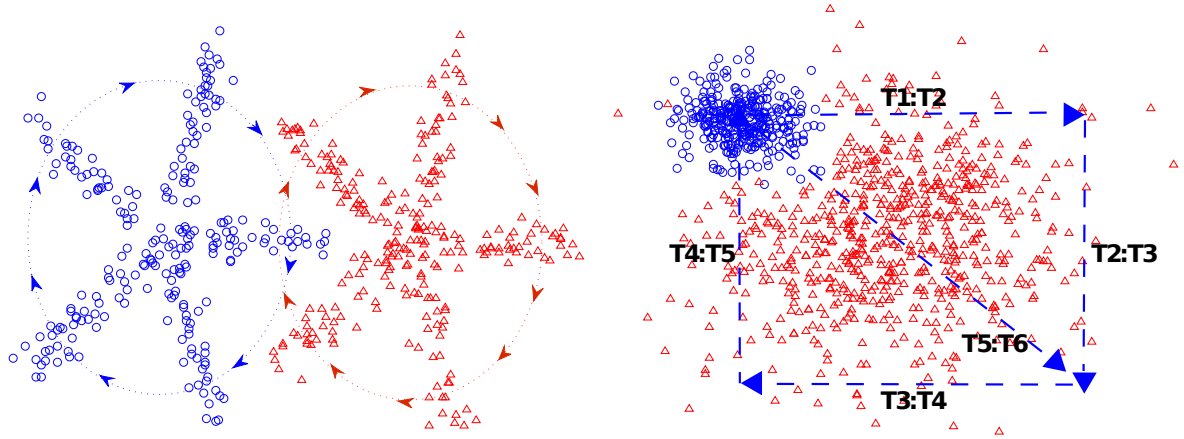
(e) Tempo $t+4$ (f) Tempo $t+5$ (g) Tempo $t+6$ (h) Tempo $t+7$

Figura 4.3: Oito diferentes momentos do conjunto de dados multimodal MG-2C-2D.

A Figura 4.4 ilustra o comportamento ao longo do tempo das classes ○ e △ nos conjuntos de dados GEARS-2C-2D e 1CSurr.

No conjunto de dados GEARS-2C-2D, duas classes com dados dispostos no espaço de atributos bidimensional com o formato de estrelas com 5 pontas giram simultaneamente e de maneira incremental em sentido horário ao longo do tempo. No conjunto de dados 1CSurr, duas classes são geradas a partir de distribuições Gaussianas com diferentes médias e desvios padrão. A distribuição dos dados da classe ○ apresenta menor desvio padrão comparada a distribuição da classe △. Ao longo do tempo, a média da distribuição dos dados da classe ○ se move de acordo com as linhas pontilhadas indicadas na Figura 4.4-*b*). Alguns exemplos da classe △ são removidos para reduzir a sobreposição entre as classes. Um dos propósitos deste conjunto de dados é simular um problema com várias classes transformado em um problema binário e que sofre mudanças ao longo do tempo. Assim, os dados da classe *bigtriangleup* podem ser considerados a união de várias classes.

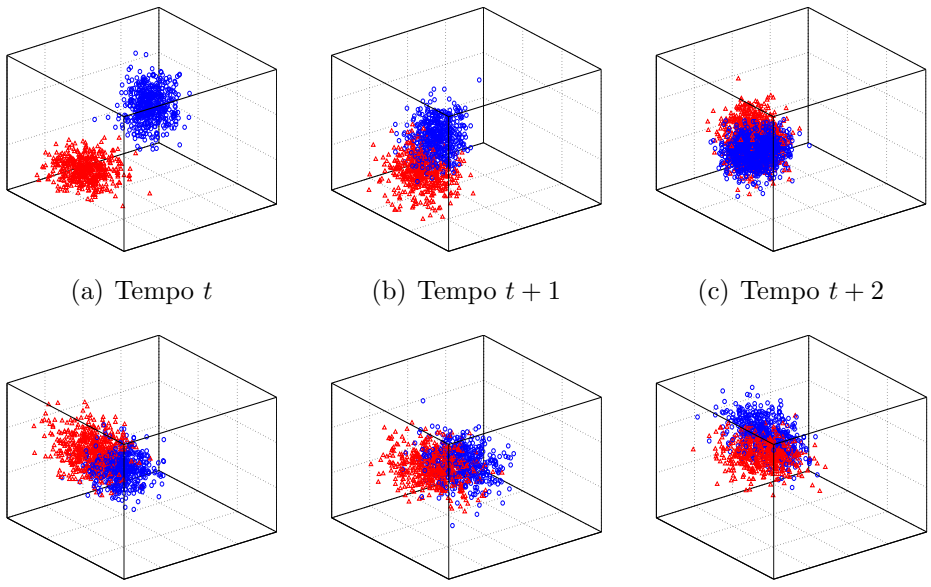
Para ilustrar o caso multidimensional, é apresentado o comportamento do conjunto de dados UG-2C-5D em seis diferentes instantes de tempo na Figura 4.5. Este conjunto



(a) GEARS-2C-2D (b) 1CSurr

Figura 4.4: Comportamento dos conjuntos de dados GEARS-2C-2D e 1CSurr ao longo do tempo.

de dados possui 2 classes e 5 dimensões. Entretanto, na figura são ilustradas somente 3 dimensões para fins de visualização.



(a) Tempo t (b) Tempo $t+1$ (c) Tempo $t+2$

(d) Tempo $t+3$ (e) Tempo $t+4$ (f) Tempo $t+5$

Figura 4.5: Seis diferentes momentos do conjunto de dados UG-2C-5D.

4.3 Conjuntos de Dados Reais

Além dos conjuntos de dados sintéticos desenvolvidos neste trabalho, também foram avaliados dois conjuntos de dados reais. O primeiro é relacionado a tarefa de predição de condições meteorológicas, denominado NOAA e o segundo é relacionado a tarefa de

identificação de usuários de acordo com padrões de digitação, denominado Keystroke. Devido às restrições inerentes ao problema de classificação de fluxo de dados não estacionários sob condicionais de latência extrema, como a presença de mudanças de conceito incrementais e a ocorrência de mudanças somente em atributos observáveis, os conjuntos de dados anteriormente apresentados no Capítulo 2 não cumprem estes requisitos e, por isso, não são utilizados para a avaliar o cenário discutido. Além disso, por ser um problema ainda pouco explorado na literatura, há uma escassez de conjuntos de dados reais disponibilizados para avaliação.

Os dados do conjunto NOAA consistem de medições meteorológicas realizadas pela *U.S. National Oceanic and Atmospheric Administration* durante 50 anos na cidade de Bellevue no estado americano de Nebraska. Este conjunto de dados possui 8 atributos: temperatura, ponto de orvalho, pressão atmosférica ao nível do mar, visibilidade, velocidade média do vento, vento máximo sustentado, temperatura máxima e temperatura mínima. A partir destes atributos, a tarefa de classificação deste conjunto de dados é determinar a partir da medição de um dia se irá chover ou não. No total, o conjunto de dados possui 18.159 exemplos, em que 5.698 são da classe "chove" e 12.461 são da classe "não chove". O conjunto de dados NOAA foi originalmente utilizado para a avaliação de classificadores de fluxo de dados sob condições de latência extrema no trabalho de Dyer et al. (2014) e posteriormente avaliado em Souza et al. (2015c) e Souza et al. (2015b).

Neste trabalho também foi desenvolvido um conjunto de dados real de dinâmica de digitação a partir do conjunto de dados CMU (Killourhy e Maxion, 2010) para a identificação de usuários com base em seu padrão de digitação. No conjunto CMU, 51 usuários digitaram 400 vezes a senha ".tie5Roanl" seguido da tecla *Enter*. Estes dados foram coletados em 8 sessões realizadas em diferentes dias. Assim, cada usuário digitou 50 vezes a senha requerida em 8 dias, totalizando 20.400 exemplos no conjunto de dados. Para identificar um usuário com base em seu padrão de digitação, foram utilizados 10 atributos referentes ao *flight time* de cada tecla pressionada. O atributo *flight time* representa a diferença de tempo entre os instantes em que uma tecla é solta e a seguinte é pressionada. A Figura 4.6 apresenta uma ilustração que representa esta característica no decorrer do fluxo de dados com diferentes usuários.

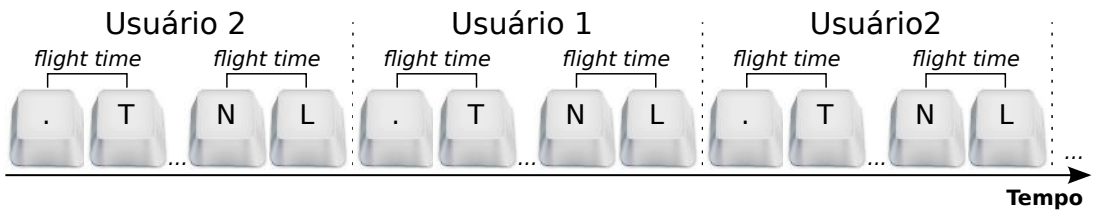


Figura 4.6: Obtenção do atributo *flight time* no fluxo de dados do conjunto Keystroke.

Este é um problema interessante em que a dinâmica de digitação pode ser utilizada como a segunda camada de segurança de um sistema que compartilha uma mesma senha

para um conjunto restrito de usuários. Por exemplo, a senha de superusuário para a administração de um servidor *web*. Entretanto, de acordo com a experiência do usuário no processo de digitação da senha, a sua dinâmica de digitação evolui incrementalmente com o decorrer dos dias. Assim, é necessário que o sistema se adapte a estas mudanças ao longo do tempo.

No conjunto de dados Keystroke construído neste trabalho foram selecionados aleatoriamente 4 usuários e os dados foram organizados de modo a respeitar a ordem cronológica da ocorrência dos exemplos e formar um fluxo de dados. Assim, o conjunto contempla um total de 1.600 exemplos e 4 classes. Este conjunto de dados foi inicialmente avaliado em Souza et al. (2015c) e posteriormente avaliado em Souza et al. (2015b).

4.4 Algoritmo SCARGC

A abordagem geral considerada pelo algoritmo proposto neste trabalho e denominado ***S**tream **C**lassification **A**lgo**r**ithm **G**uided by **C**lustering* – **SCARGC** (Souza et al., 2015c) é realizar sucessivas etapas de agrupamento dos dados ao longo do tempo para se adaptar a possíveis mudanças de conceito em problemas de classificação sob condições de latência extrema. O algoritmo é intuitivo, eficiente e possui baixa complexidade de implementação. Antes da apresentação do funcionamento do algoritmo, considere as seguintes condições impostas pela abordagem:

1. Um conjunto reduzido de dados rotulados é disponibilizado inicialmente antes da execução do algoritmo. O fornecimento destes dados é necessário para que seja possível definir o problema como classificação a partir da definição da quantidade de classes e a sua disposição inicial no espaço de atributos;

2. As mudanças de conceito devem ser incrementais e responsáveis por gerar um deslocamento das classes no espaço de atributos dos dados. Este deslocamento pode ser, por exemplo, a translação de uma ou mais classes. Esta condição é necessária para que seja possível "rastrear" as mudanças das classes ao longo do tempo a partir da observação de alterações nos valores dos atributos dos dados. Devido a sobreposição entre diferentes conceitos ao longo do tempo, a mudança incremental é considerada mais difícil de ser detectada do que a mudança abrupta por métodos tradicionais, mesmo no cenário de latência nula (Dyer et al., 2014);

3. O número de classes é constante ao longo do tempo. Embora algumas aplicações apresentem um número variável ou desconhecido de classes, frequentemente há interesse em somente uma das classes. Assim, o problema pode ser transformado em um problema de classificação binária.

As condições 1 e 2 são inerentes ao problema de latência extrema e também estão presentes nas outras abordagens da literatura como APT e COMPOSE. A condição 3 é inerente a solução proposta neste trabalho e será explorada em trabalhos futuros, já que o problema de evolução do número de classes ao longo do tempo foge do escopo deste trabalho e constitui um problema não trivial. Embora estas condições possam parecer restritivas em um primeiro momento, podem ser consideradas menos otimistas que a condição/suposição de disponibilização pelo processo gerador dos dados de todos os rótulos corretos dos exemplos processados após a etapa de classificação sem qualquer atraso.

Antes da etapa de classificação, o algoritmo SCARGC constrói um modelo de classificação inicial com c classes obtidas a partir dos dados rotulados disponibilizados em um primeiro momento. Estes dados rotulados são agrupados em $k \geq c$ grupos, sendo que k é definido manualmente pelo usuário. Se $k = c$, o algoritmo considera as c classes como grupos iniciais. Se $k > c$, é executado um algoritmo de agrupamento para encontrar os grupos naturais do problema e cada grupo é associado a uma das c classes. Em ambos os casos, o conjunto inicial de k grupos são denotados por $\mathcal{C}^{(0)} = \{\mathcal{C}_1^{(0)}, \mathcal{C}_2^{(0)}, \ldots, \mathcal{C}_k^{(0)}\}$.

Após esta etapa inicial, o algoritmo está pronto para receber novos exemplos não rotulados do fluxo de dados. Ao receber um novo exemplo, duas ações são imediatamente realizadas: a predição do rótulo com base no modelo classificação inicial e o armazenamento do exemplo em um *pool* que representa uma memória de curto prazo. Após o armazenamento de uma quantidade mínima de exemplos de cada classe ou após transcorrido um determinado horizonte de tempo, os exemplos armazenados no *pool* são agrupados em k grupos. Este novo conjunto de grupos é denotado por $\mathcal{C}^{(1)} = \{\mathcal{C}_1^{(1)}, \mathcal{C}_2^{(1)}, \ldots, \mathcal{C}_k^{(1)}\}$. Neste momento, o algoritmo associa cada novo grupo $\mathcal{C}_i^{(1)} \in \mathcal{C}^{(1)}$ a um grupo $\mathcal{C}_j^{(0)} \in \mathcal{C}^{(0)}$ encontrado na etapa anterior. Consequentemente cada grupo $\mathcal{C}_i^{(1)}$ é associado a uma classe c e cada classe c pode ter mais de um grupo associado a ela. Esta associação permite realizar uma nova rotulação dos exemplos armazenados no *pool*. O principal objetivo desta etapa de agrupamento é propagar os conceitos encontrados em dados mais antigos para dados processados mais recentemente.

Os dados recentemente rotulados pela etapa de agrupamento são utilizados para a geração de um modelo de classificação atualizado que substitui o modelo inicial. Este processo de agrupamento seguido de classificação se repete alternadamente por todo o processamento do fluxo de dados. Assim, periodicamente um modelo de classificação atualizado é construído a partir de dados rotulados obtidos na etapa de agrupamento. De modo geral, os rótulos são obtidos por meio da associação dos grupos encontrados na iteração atual $\mathcal{C}^t$ com os rótulos calculados na iteração anterior $\mathcal{C}^{t-1}$.

Devido a necessidade de respostas rápidas impostas em tarefas de classificação de fluxo de dados, optou-se por favorecer a simplicidade e eficiência na implementação do algoritmo SCARGC. Por isso, foi escolhido o algoritmo k-Médias (MacQueen, 1967) para a

realização da etapa de agrupamento de dados. Além de ser um algoritmo de agrupamento eficiente, simples, computacionalmente eficiente e amplamente utilizado na literatura, também possui diversas extensões como X-Médias (Pelleg e Moore, 2000) e *Fuzzy* C-Médias (Dunn, 1973) que podem ser adaptadas para a utilização em conjunto com o algoritmo SCARGC.

Na etapa de agrupamento, os centroides de cada grupo encontrado na iteração anterior são armazenados para a utilização com dois diferentes propósitos na iteração atual. Primeiro, para evitar a instabilidade do algoritmo k-Médias devido a inicialização aleatória, os centroides dos grupos anteriores são utilizados para inicializar o algoritmo de agrupamento. Segundo, os centroides são utilizados como "estatísticas resumidas" dos grupos encontrados na iteração anterior. Assim, a associação entre os grupos $\mathcal{C}^t$ e $\mathcal{C}^{t-1}$ é realizada a partir do cálculo de similaridade entre os centroides. Formalmente, dado os centroides atuais $(q_1, q_2, \ldots, q_k)$ dos grupos $\mathcal{C}^t$ recentemente encontrados e os centroides anteriores $(p_1, p_2, \ldots, p_k)$ dos grupos anteriormente rotulados $\mathcal{C}^{t-1}$, onde q_i e p_i são dados n-dimensionais, cada centroide p_i possui um rótulo y_i e cada centroide q_i necessita de um rótulo $\hat{y}_i$. Este rótulo é obtido a partir da aplicação do algoritmo 1-Vizinho Mais Próximo, de modo que cada novo centroide q_i herda o rótulo do pertencente ao centroide mais próximo calculado na iteração anterior de acordo a distância euclidiana.

Nos experimentos com dados sintéticos realizados neste trabalho, uma nova etapa de agrupamento é iniciada quando o *pool* armazena um total de 300 exemplos e um mínimo de 10 exemplos são preditos para cada classe c pelo classificador. Embora um *pool* com poucos exemplos permita uma adaptação rápida, também aumenta os custos computacionais devido a exaustivas atualizações. Além disso, um *pool* de tamanho muito pequeno pode levar a grupos poucos representativos devido à ausência de dados. Por outro lado, um *pool* com muitos exemplos pode armazenar conceitos provenientes de diferentes instantes de tempo e possivelmente desatualizados.

Com o objetivo de reduzir custos computacionais, o algoritmo não atualiza o modelo de classificação em todas as etapas de agrupamento. Para decidir se o modelo deve ser atualizado ou não, o algoritmo verifica a partir dos últimos exemplos processados se há uma diferença significativa entre os rótulos dados pelo classificador e os rótulos obtidos na etapa de agrupamento. Uma diferença significativa na concordância entre as predições é considerada um indicativo da presença de mudanças de conceito, o que leva a atualização do atual modelo de classificação. Entretanto, é importante destacar que este teste não é um detector de mudanças, mas somente um indicador para evitar atualizações desnecessárias. Entre uma variedade de testes que podem ser aplicados para medir a concordância das atribuições dos rótulos aos exemplos, é sugerido neste trabalho o uso da medida *Kappa* (Cohen, 1960). A medida representa o grau de concordância entre classificadores além do que seria esperado tão somente pelo acaso e é definida de acordo com a Equação 4.1.

$$Kappa = \frac{P_O - P_E}{1 - P_E}, \tag{4.1}$$

onde P_O e P_E são definidas de acordo com a Equação 4.2 e Equação 4.3, respectivamente.

$$P_O = \frac{C}{C + D}, \tag{4.2}$$

sendo que:

- C é o número de concordâncias entre os classificadores;
- D é o número de discordâncias entre os classificadores.

$$P_E = \sum_{i=1}^{n} (p_{i1} * p_{i2}), \tag{4.3}$$

sendo que:

- n é o número de classes;
- i é o índice da classe, de 1 a n;
- p_{i1} é a proporção de ocorrência da classe i para o classificador 1;
- p_{i2} é a proporção de ocorrência da classe i para o classificador 2;

Assim, pode-se considerar que P_O é a concordância relativa observada entre as rotulações atribuídas pelo classificador base e o algoritmo de agrupamento e P_E é a probabilidade de concordância ao acaso entre as rotulações, ou seja, a probabilidade do classificador e do algoritmo de agrupamento concordarem, dado que ambos realizam a rotulação de cada exemplos de modo aleatório. A medida varia entre -1 e 1, sendo que -1 representa máxima discordância e 1 representa concordância perfeita entre os classificadores. Embora um valor acima de 0,8 seja considerado um ótimo nível de concordância, nos experimentos realizados neste trabalho o classificador é atualizado somente quando observado concordância total entre os classificadores ($Kappa = 1$). Assim, por simplicidade a medida é substituída pela simples conferência das atribuições de rótulos dado pelo classificador e o algoritmo de agrupamento, sendo possível remover este parâmetro do algoritmo.

O algoritmo SCARGC permite flexibilidade quanto a escolha do algoritmo de classificação que pode ser aplicado durante o processamento do fluxo de dados. Nos experimentos realizados neste trabalho, foram avaliados o uso do algoritmo 1-Vizinho Mais Próximo (1NN) (Cover e Hart, 1967) e Máquina de Vetores de Suporte (SVM) (Vapnik, 1998).

O algoritmo SCARGC é apresentado em pseudo-código no Algoritmo 3. O SCARGC tem como entrada um conjunto inicial reduzido de dados rotulados ($\mathcal{T}$) e um conjunto de

dados não rotulados $\mathcal{DS}$ o qual busca-se predizer os seus rótulos. O algoritmo também necessita da definição de dois parâmetros: número de grupos k para a etapa de agrupamento e tamanho da memória de curto prazo ou *pool*, definido por θ.

Algoritmo 3: Stream Classification Algorithm Guided by Clustering – SCARGC

Entrada:
Conjunto de treino inicial $\mathcal{T}$
Fluxo de dados não rotulados $\mathcal{DS}$
Tamanho do *pool* θ
Número de grupos k
Saída:
Classificador atualizado Φ
Rótulo y para cada exemplo $x \in \mathcal{DS}$

```
1  início
2      c ← extraiNroClasses(T)
3      Φ ← induzClassificadorInicial(T)
4      se k = c então
5          C^0 ← extraiGruposRotulados(T, k)
6      senão
7          C^0 ← agrupamento(T, k)
8      fim
9      pool ← ∅
10     rotulosPreditos ← ∅
11     t ← 1
12     para todo exemplo x⃗ ∈ DS faça
13         ŷ ← Φ(x⃗)
14         pool ← pool ⋃ {x⃗}
15         rotulosPreditos ← rotulosPreditos ⋃ {ŷ}
16         se |pool| = θ então
17             se contaExemplosPorClasse(rotulosPreditos, c) ⩾ 10 então
18                 C^t ← agrupamento(pool, k)
19                 /* A partir da similaridade entre os centroides dos grupos */
20                 rotulosPropagados ← associaGruposNovosComAntigos(C^t, C^{t−1})
21                 se concordancia(rotulosPropagados, rotulosPreditos) ≠ 1 então
22                     T' ← rotulaDados(pool, rotulosPropagados)
23                     Φ ← atualizaClassificador(T')
24                     pool ← ∅
25                     rotulosPreditos ← ∅
26                     t ← t + 1
27                 fim
28             fim
29         fim
30     fim
31 fim
```

4.5 Algoritmo MClassification

Visando mitigar a dificuldade de definição dos parâmetros impostos pela abordagem SCARGC em relação ao número de grupos (k) e tamanho da memória de curto prazo (θ), este trabalho propõe uma segunda abordagem denominada *Micro-Clusters for Classification* – **MClassification** (Souza et al., 2015b) para lidar com a classificação de fluxo de dados não estacionários sob condições de latência extrema.

Para se adaptar às mudanças nos dados ao longo do tempo, a segunda abordagem proposta neste trabalho utiliza o conceito de *Cluster Feature*, originalmente introduzido por Zhang et al. (1996) no algoritmo de agrupamento para grandes bases de dados BIRCH e posteriormente denominado *Micro-Cluster* (ou microgrupo) no algoritmo de agrupamento de fluxo de dados *CluStream* proposto por Aggarwal et al. (2003). Desse modo, antes de apresentar o funcionamento do algoritmo MClassification, são apresentados os principais conceitos relacionados a representação *Micro-Cluster*.

O objetivo dos *Micro-Clusters* (MC) é fornecer uma representação compacta de um conjunto de exemplos sem a necessidade de armazenar todas as instâncias em memória. A Figura 4.7 ilustra um conjunto de exemplos agrupados em diversos *Micro-Clusters* (em vermelho), de modo que o agrupamento destes *Micro-Clusters* formam dois *Macro-Clusters* ou macrogrupos, assinalados em azul na ilustração.

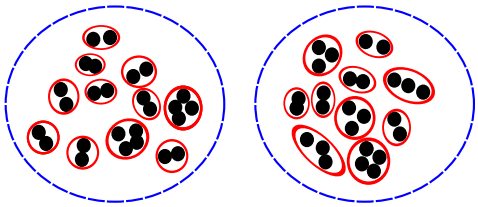


Figura 4.7: Representação de *Micro-Clusters* (vermelho) e *Macro-Clusters* (azul) em um conjunto de exemplos distribuídos no espaço de atributos bidimensional.

Para fornecer esta representação compacta de dados é utilizada a tripla $(N, \overrightarrow{LS}, \overrightarrow{SS})$ para armazenar estatísticas suficientes de um conjunto de exemplos, sendo que:

- N é o número de exemplos em um grupo;
- $\overrightarrow{LS} = \sum_1^N \vec{x_i}$ é a soma linear dos N exemplos de um grupo;
- $\overrightarrow{SS} = \sum_1^N (\vec{x_i})^2$ é a soma quadrada dos exemplos de um grupo.

Desse modo, a partir de um MC que sumariza as informações de um conjunto com N exemplos, é possível calcular medidas como centroide, raio e diâmetro de um MC sem a necessidade de armazenar todos os exemplos a partir das Equações 4.4, 4.5 e 4.6, respectivamente.

$$Centroide = \frac{\overrightarrow{LS}}{N} \tag{4.4}$$

$$Raio = \sqrt{\frac{\overrightarrow{SS}}{N} - \left(\frac{\overrightarrow{LS}}{N}\right)^2} \tag{4.5}$$

$$Diâmetro = \sqrt{\frac{2 * N * \overrightarrow{SS} - 2 * \overrightarrow{LS^2}}{N * (N-1)}} \tag{4.6}$$

Esta representação de dados possui propriedades interessantes que a torna uma escolha natural para problemas de fluxo de dados. As propriedades mais importantes são *incrementalidade* e *aditividade*. Por exemplo, dado um conjunto de exemplos em que as suas estatísticas suficientes estão armazenadas em $MC_A = (N_A, \overrightarrow{LS}_A, \overrightarrow{SS}_A)$, é possível adicionar um novo exemplo $\vec{x}$ em MC_A de maneira incremental somente atualizando as suas estatísticas suficientes da seguinte maneira:

$$\begin{gathered}\overrightarrow{LS}_A \leftarrow \overrightarrow{LS}_A + \vec{x} \\ \overrightarrow{SS}_A \leftarrow \overrightarrow{SS}_A + (\vec{x})^2 \\ N_A \leftarrow N_A + 1\end{gathered}$$

Por sua vez, a propriedade de aditividade considera que dados dois *Micro-Clusters* disjuntos MC_A e MC_B, a união entre estes dois grupos é igual a soma de suas partes. Assim, as estatísticas suficientes de um novo *Micro-Cluster* $MC_C = (N_C, \overrightarrow{LS}_C, \overrightarrow{SS}_C)$ que armazena as informações de $MC_A \cup MC_B$ são computadas do seguinte modo:

$$\begin{gathered}\overrightarrow{LS}_C \leftarrow \overrightarrow{LS}_A + \overrightarrow{LS}_B \\ \overrightarrow{SS}_C \leftarrow \overrightarrow{SS}_A + \overrightarrow{SS}_B \\ N_C \leftarrow N_A + N_B\end{gathered}$$

Embora seja eficiente e apropriada para problemas de fluxo de dados em geral, nota-se que a representação de *Micro-Clusters* tem sido essencialmente utilizada somente em problemas de agrupamento. Neste trabalho é proposto o uso da representação condensada *Micro-Clusters* para a tarefa de classificação de fluxo de dados não estacionários. Para isso, a representação foi modificada para também armazenar informações sobre a classe de um conjunto de pontos. Assim, a representação proposta neste trabalho é composta pela 4-tupla $(N, \overrightarrow{LS}, \overrightarrow{SS}, y)$, onde y é o rótulo correspondente a um conjunto de exemplos. O algoritmo MClassification é apresentado em pseudocódigo no Algoritmo 4 e a ideia geral do algoritmo é apresentada a seguir.

O algoritmo inicia recebendo como entrada um conjunto inicial e reduzido de exemplos rotulados $\mathcal{T}$ de tamanho $|\mathcal{T}|$. A partir deste conjunto de dados são criados m *Micro-Clusters* rotulados de acordo com a memória disponível. Assim, caso $|\mathcal{T}|$ seja menor que

Algoritmo 4: Micro-Clusters for Classification – MClassification

Entrada:
Conjunto de treino inicial $\mathcal{T}$
Fluxo de dados não rotulados $\mathcal{DS}$
Raio máximo do MC r
Quantidade máxima de MC α
Saída:
Rótulo predito $\hat{y}$ para cada exemplo $\overrightarrow{x} \in \mathcal{DS}$

```
início
    MC ← ∅
    para todo exemplo rotulado (x⃗, y) ∈ T faça
        mc ← criaNovoMC(x⃗, y)
        MC ← MC ∪ {mc}
    fim
    dif ← α − |T|
    se dif < 0 então
        MC ← uniãoMCSimilares(MC, dif)
    fim
    para todo exemplo x⃗ ∈ DS faça
        mc ← buscaMCMaisSimilar(MC, x⃗)
        ŷ ← mc.y
        se raio(mc) ≤ r então
            MC ← MC ∪ {mc}
        senão
            MC ← uniãokMCMaisDissimilaresClasse(MC, x⃗, ŷ)
            mc ← criaNovoMC(x⃗, ŷ)
            MC ← MC ∪ {mc}
        fim
    fim
fim
```

a memória disponível para o armazenamento dos exemplos, $m = |\mathcal{T}|$, de modo que cada MC armazena a informação de um único exemplo. Por outro lado, se $|\mathcal{T}|$ for maior que a memória disponível, os exemplos mais similares presentes em $\mathcal{T}$ são condensados em um mesmo MC até que seja possível armazená-los em memória e, consequentemente, $m < |\mathcal{T}|$. Além do parâmetro relacionado a quantidade máxima de MC, o algoritmo também possui o parâmetro r, relacionado ao raio máximo permitido para um MC na fase de classificação. Este parâmetro é melhor discutido a seguir.

Na fase de classificação, cada exemplo $\overrightarrow{x_t}$ do fluxo de dados no instante de tempo t tem o seu rótulo predito $\hat{y}_t$ de acordo com o MC rotulado mais similar de acordo com a distância euclidiana. Neste momento, é verificado se a adição do exemplo $\overrightarrow{x_t}$ no MC mais similar irá fazer com que o raio do MC exceda o tamanho máximo r, definido pelo usuário. Se o raio não exceder o limiar r, o exemplo $\overrightarrow{x_t}$ é adicionado ao MC mais similar

utilizando a propriedade de incrementalidade e suas estatísticas suficientes $(N, \overrightarrow{LS}, \overrightarrow{SS})$ são atualizadas. Assim, cada novo exemplo adicionado ao MC é responsável por alterar sensivelmente o seu centroide, fazendo com que a posição do MC se desloque em direção ao novo conceito da classe, que se altera ao longo do tempo. Por outro lado, se o raio excede o limiar r, um novo MC acompanhado do rótulo predito $\hat{y}_t$ é criado para armazenar o exemplo $\overrightarrow{x_t}$. Além disso, o algoritmo busca pelos dois MCs mais distantes do exemplo $\overrightarrow{x_t}$ pertencentes à classe predita $\hat{y}_t$ e realiza a união destes MCs utilizando a propriedade de aditividade. Assim, os dois MCs mais dissimilares de $\overrightarrow{x_t}$ são unidos em um único MC que é deslocado em direção ao novo conceito da classe. Esta estratégia de manutenção *online Micro-Clusters* é utilizada ao longo de todo o processamento do fluxo de dados com o objetivo de constantemente atualizar o modelo de classificação às mudanças incrementais de conceito nos dados.

4.6 Avaliação Experimental

Nesta seção, serão apresentados os resultados de classificação obtidos pelas abordagens propostas neste trabalho e anteriormente discutidas ao longo deste capítulo: *Stream Classification Algorithm Guided by Clustering* – SCARGC e *Micro-Clusters for Classification* – MClassification. A avaliação experimental leva em consideração os conjunto de dados sintéticos apresentados na Seção 4.2 e os dois conjuntos de dados reais apresentados na Seção 4.3. Em ambos os casos, considera-se que as mudanças de conceito são incrementais ao longo do tempo e visíveis no espaço de atributos dos dados.

4.6.1 Métodos de Comparação

Devido à complexidade de implementação e a indisponibilidade do código fonte dos algoritmos APT e COMPOSE, não foi possível realizar a comparação direta dos resultados obtidos pelas duas abordagens propostas neste trabalho com os outros métodos da literatura em todos os conjuntos de dados sintéticos e reais avaliados neste capítulo. Assim, dados os resultados apresentados em Dyer et al. (2014), foi possível realizar a comparação direta entre as abordagens somente nos conjuntos de dados UG-2C-2D, UG-2C-3D e MG-2C-3D. Por isso, para permitir uma melhor compreensão dos resultados alcançados em todos os conjuntos de dados, foram utilizados 5 métodos de comparação que representam diferentes cenários como o uso de um classificador estático tradicionalmente utilizado na classificação de exemplos em lote e o uso de diferentes classificadores que possuem acesso aos rótulos corretos após a etapa de predição, ou seja, onde a latência é nula. Estes métodos são apresentados a seguir:

- **Estático.** Nesta configuração um classificador é induzido utilizando os primeiros exemplos rotulados do fluxo de dados e nenhuma atualização é realizada ao longo do

tempo. Esta configuração representa a abordagem tradicionalmente utilizada por algoritmos de classificação que não consideram a presença de mudanças de conceito nos dados. Assim, é possível observar se os dados apresentados no início do fluxo de dados são suficientemente representativos para a classificação dos exemplos restantes sem a necessidade de adaptação do classificador. Por ser uma abordagem simples, o seu resultado pode ser considerado um limiar inferior (*baseline*) que deve ser superado por abordagens mais complexas que se adaptam ao longo do tempo. Para todos os conjuntos de dados sintéticos, considerou-se a indução do classificador Estático com os primeiros 300 exemplos rotulados do fluxo de dados;

- **Deslizante com latência nula – Deslizante LN.** Nesta configuração um classificador é inicialmente induzido utilizando os primeiros exemplos rotulados do fluxo de dados e com o decorrer do tempo, cada novo exemplo processado é inserido no conjunto de treinamento acompanhado de seu respectivo rótulo correto e o exemplo mais antigo é descartado. Assim, tem-se um classificador atualizado a cada novo exemplo processado. Em outras palavras, é utilizada uma janela deslizante sobre os exemplos do fluxo de dados considerando-se o cenário de latência nula. Assim como no classificador Estático, são utilizados os 300 primeiros exemplos para a indução inicial e este valor se mantêm constante para o tamanho da janela deslizante durante todo o seu funcionamento;

- **Incremental com latência nula – Incremental LN.** Esta configuração é similar ao classificador Deslizante LN, entretanto não considera o descarte de exemplos antigos. Desse modo, é utilizada uma janela incremental que inclui cada novo exemplo do fluxo de dados acompanhado de seu rótulo correto;

- **Persistente com latência nula – Persistente LN.** Nesta configuração, para cada exemplo do fluxo de dados é predito o mesmo rótulo atribuído ao último exemplo observado. Ou seja, $\hat{y}_t = y_{t-1}$ para todo exemplo $\overrightarrow{x}_t$ do fluxo de dados. Para isso, é necessário que se assuma o cenário de latência nula onde se tem o conhecimento do rótulo correto de cada exemplo processado após a sua predição. Embora simples, o classificador se mostra eficiente em problemas onde há dependência temporal para a ocorrência/geração dos eventos, como é o caso do conjunto de dados *Electricity* em que a tarefa consiste em predizer diariamente o consumo de energia elétrica de uma cidade. Conforme discutido em Žliobaitė et al. (2015), um classificador que considera dependência temporal dos exemplos se mostra adequado a este cenário com resultados superiores a outros classificadores da literatura;

- **Majoritário com latência nula – Majoritário LN.** O classificador Majoritário LN considera a mesma política de inserção incremental de novos exemplos rotulados ao longo do tempo considerada pelo classificador Incremental LN. Entretanto, o

rótulo predito para cada exemplo do fluxo é dado de acordo com a classe majoritária observada no momento da predição. Ou seja, $\hat{y}_t = \arg\max_i P(c = i)$ para todo exemplo $\overrightarrow{x}_t$ do fluxo de dados.

O uso dos métodos de comparação Estático, Deslizante LN e Incremental LN foram originalmente propostos em Souza et al. (2013a, 2015c) para a avaliação de algoritmos de classificação de fluxo de dados sob condições de latência extrema, enquanto os métodos Persistente LN e Majoritário LN foram propostos em Žliobaitė et al. (2015) para a avaliação de classificadores de fluxo de dados que consideram o cenário de latência nula.

4.6.2 Análise dos Resultados

Uma visão geral dos resultados alcançados pelos métodos propostos (SCARGC e MClassification) e pelos métodos de comparação (Estático, Deslizante LN, Incremental LN, Persistente LN e Majoritário LN) é apresentada na Tabela 4.2. Nesta tabela, são reportadas as acurácias médias ao longo do tempo obtidas pelos algoritmos para a classificação de todos os exemplos de cada conjunto de dados. O classificador SCARGC foi inicialmente treinado com somente 50 exemplos rotulados (parâmetro $|\mathcal{T}| = 50$) e com uma memória de curto prazo ou *pool* com capacidade para armazenar 300 exemplos (parâmetro $\theta = 300$). Os resultados apresentados se referem ao desempenho do algoritmo SCARGC com o classificador base 1-Vizinho Mais Próximo e com um valor k ótimo encontrado experimentalmente no reduzido conjunto de dados rotulados de cada base de dados. O classificador MClassification foi inicialmente treinado com 150 exemplos rotulados (parâmetro $|\mathcal{T}| = 150$) e raio máximo do textitMicro-Cluster $r = 0{,}1$. Os classificadores utilizados para comparação que necessitam realizar a etapa de treinamento inicial (Estático, Deslizante LN e Incremental LN) utilizaram 300 exemplos rotulados nesta etapa.

Dados os resultados apresentados na Tabela 4.2, para analisar se há diferenças estatisticamente significativas entre os algoritmos foi realizado o teste de *Friedman* com a hipótese nula de que os algoritmos são equivalentes. O teste de *Friedman* é um teste estatístico não paramétrico que faz um *ranking* dos algoritmos para cada conjunto de dados de acordo com seu desempenho e então compara os *rankings* médios dos algoritmos. Quando a hipótese nula é rejeitada pelo teste de *Friedman*, com confiança de 95%, o pós-teste de *Nemenyi* é utilizado para detectar quais diferenças entre os algoritmos são significativas (Demšar, 2006). De acordo com esse teste, o desempenho de dois algoritmos é estatisticamente diferente sempre que seus correspondentes *rankings* médios diferirem por ao menos um determinado valor de diferença crítica. Os resultados do pós-teste de *Nemenyi* podem ser representados em um diagrama nos quais o *ranking* médio dos algoritmos é apresentado no eixo horizontal. Na Figura 4.8 é apresentado o diagrama de diferença crítica que ilustra o *ranking* obtido pelos algoritmos no teste de acordo com a acurácia observada. Neste diagrama, a diferença crítica (CD) é apresentada acima do eixo hori-

Tabela 4.2: Acurácia média ao longo do tempo obtida pelos métodos de comparação e métodos propostos de acordo com avaliação nos conjuntos de dados de *benchmark*.

Conjunto de dados	Estático	Deslizante LN	Increm. LN	Persist. LN	Majorit. LN	SCARGC	MClassification
1CDT	97,01	99,88	99,23	49,11	47,19	99,75	99,89
1CHT	91,96	99,24	97,48	50,57	47,10	99,25	99,38
1CSurr	65,75	98,52	71,61	53,76	63,45	94,35	85,15
2CDT	54,38	93,47	63,14	49,74	47,29	90,92	95,23
2CHT	54,03	85,44	62,01	50,21	46,45	86,02	87,93
4CE1CF	95,81	97,15	99,76	19,97	19,15	94,08	94,38
4CR	25,06	99,98	39,07	24,88	21,35	99,95	99,98
4CRE-V1	26,17	97,64	40,98	24,99	22,58	97,39	90,63
4CRE-V2	27,11	89,37	32,64	24,71	22,50	91,90	91,59
5CVT	40,72	86,86	63,14	21,96	33,32	90,15	88,40
FG 2C 2D	81,30	93,84	87,50	62,42	75,00	95,16	62,48
GEARS 2C 2D	93,62	99,86	94,97	49,96	48,31	95,89	94,73
MG 2C 2D	49,00	90,40	68,87	49,89	48,55	92,71	80,58
UG 2C 2D	47,28	94,27	65,97	50,18	48,05	95,56	95,28
UG 2C 3D	60,64	92,86	85,03	49,86	48,58	94,77	94,72
UG 2C 5D	68,81	89,91	82,80	50,19	48,56	90,98	91,25
NOAA	66,19	72,01	74,27	68,03	68,62	68,61	67,54
KEYSTROKE	68,69	90,14	92,48	6,48	5,59	88,41	90,41

zontal e os algoritmos que não apresentam resultados estatisticamente significantes entre si, são conectados por uma linha sólida denominada *clique*.

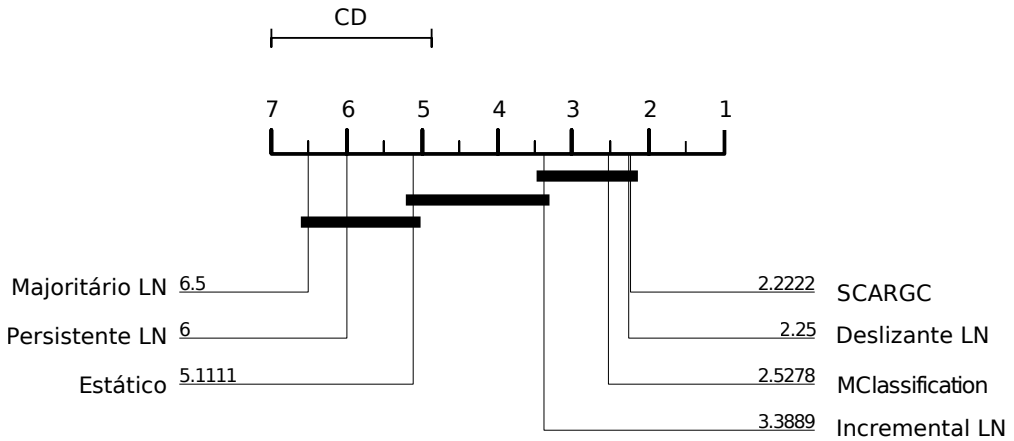


Figura 4.8: Diagrama de diferença crítica obtido a partir do teste de *Friedman* com 95% de significância e pós-teste de *Nemenyi*.

Pode-se observar no diagrama de diferença crítica apresentado na Figura 4.8 que os algoritmos propostos SCARGC e MClassification estão respectivamente na primeira e terceira posição do *ranking* que considera um total de 7 métodos. Também nota-se que os algoritmos SCARGC, MClassification e os métodos de comparação Deslizante LN e Incremental LN, não apresentam diferenças estatisticamente significantes entre si. Este resultado é bastante motivador, dado que os métodos de comparação (Deslizante LN e

Incremental LN) têm o conhecimento de todos os rótulos corretos após o processamento dos exemplos, sem nenhum tempo de atraso. Assim, os algoritmos propostos apresentam resultados similares a métodos que podem ser considerados limiares superiores (*toplines*) para o problema de classificação de fluxo de dados não estacionários. Entre os piores resultados, estão os métodos de comparação Estático, Persistente LN e Majoritário LN. O método Estático é utilizado como limiar inferior (*baseline*), uma vez que não realiza nenhuma atualização em seu modelo de classificação após a indução inicial. Embora em determinadas situações reais os métodos Majoritário LN e Persistente LN se comportem de maneira satisfatória, os resultados obtidos por estes métodos são justificáveis devido as características dos conjuntos de dados sintéticos avaliados neste trabalho, que não possuem dependência temporal entre os exemplos e, na maioria dos casos, apresentam uma quantidade balanceada de exemplos por classe.

Para melhor compreensão do desempenho dos algoritmos, somente o valor médio de acurácia obtido ao longo de todo o processamento do fluxo de dados pode ser pouco informativo. Como os exemplos chegam ao longo do tempo, existe uma ordenação temporal que deve ser considerada e é interessante que os algoritmos também sejam avaliados ao longo do tempo de modo que seja possível observar o seu desempenho em determinados intervalos. Desse modo, serão apresentados a seguir, os resultados de acurácia ao longo do tempo. Com o objetivo de padronizar a apresentação dos resultados, independente da quantidade de exemplos de cada conjunto de dados, os dados foram divididos em 100 partes iguais (quando possível) e calculada a acurácia média obtida em cada uma das partes. Embora os resultados sejam apresentados em partes, é importante destacar que os algoritmos (com exceção dos algoritmos APT e COMPOSE que processam os dados em lotes) retornam uma resposta imediata para cada exemplo do fluxo de dados processado.

Inicialmente, são apresentados os resultados de acurácia obtidos ao longo do tempo pelos métodos propostos (SCARGC e MClassification) e os métodos de comparação (Estático, Deslizante LN, Incremental LN, Persistente LN e Majoritário LN) nos conjuntos de dados 1CSurr, 2CDT, 4CRE-V2 e 5CVT na Figura 4.9. No conjunto de dados 1CSurr, Figura 4.9-*a*), é possível observar que o método de comparação Deslizante LN de fato é um limiar superior, com resultados próximos a 100% durante todo o tempo. Também é possível observar o desempenho similar dos métodos propostos SCARGC e MClassification até aproximadamente 90% dos dados processados, quando ambos apresentam uma queda de desempenho. Nos conjuntos de dados 2CDT, 4CRE-V2 e 5CVT, nota-se que os algoritmos MClassification e SCARGC (isoladamente ou simultaneamente) apresentam resultados superiores ao método de comparação Deslizante LN durante todo o período de tempo. Em todos os conjuntos de dados, observa-se os resultados inferiores dos métodos Majoritário LN e Persistente LN. O método Incremental LN, além de ser pouco eficiente em termos de tempo computacional e uso de memória, já que incorpora todos os exemplos processados no conjunto de treino e utiliza o classificador 1NN na etapa de classificação,

armazena diversos conceitos ao longo tempo e não descarta conceitos desatualizados, o que prejudica o seu desempenho de classificação na maioria dos casos simulados nos dados sintéticos.

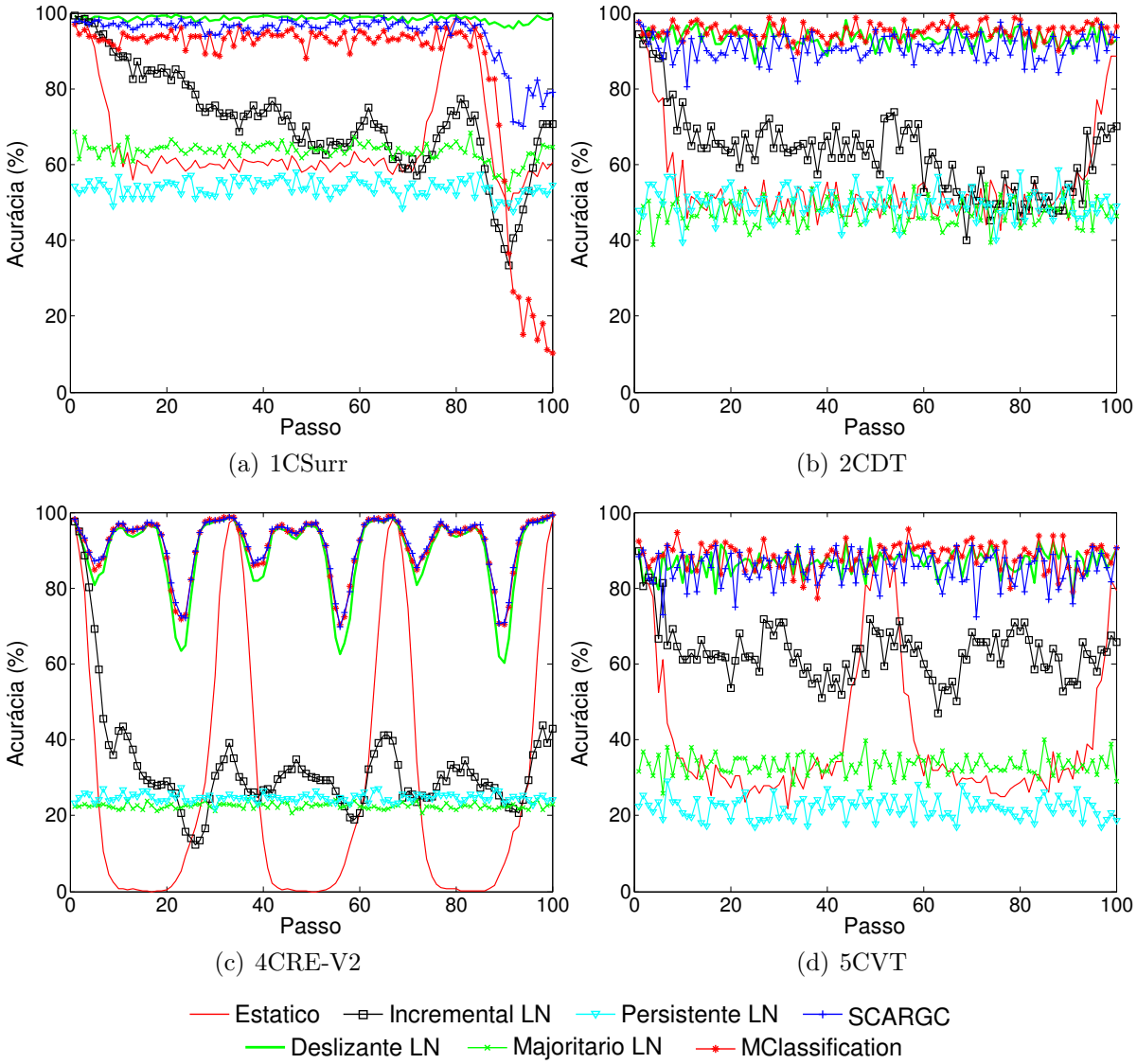


Figura 4.9: Acurácia ao longo do tempo dos algoritmos SCARGC, MClassification e métodos de comparação nos conjuntos de dados 1CSurr, 2CDT, 4CRE-V2 e 5CVT.

Os resultados alcançados nos conjuntos de dados 4CR, FG-2C-2D, GEARS-2C-2D e UG-2C-5D são apresentados na Figura 4.10. Nos conjuntos de dados 4CR (Figura 4.10-*a*)) e UG-2C-5D (Figura 4.10-*d*)), nota-se que os algoritmos SCARGC e MClassification apresentam resultados equivalentes e superam todos os métodos de comparação considerados. No conjunto de dados FG-2C-2D (Figura 4.10-*b*)), o melhor resultado é apresentado pelo algoritmo SCARGC. Por outro lado, neste conjunto de dados o algoritmo MClassification apresenta um dos piores resultados gerais. Basicamente, a partir do instante de tempo 30 pode-se notar uma redução drástica da acurácia de 90% para em torno de 40%.

Entretanto, a partir do instante de tempo 90, o algoritmo volta a apresentar resultados competitivos próximos a 80% de acurácia. Por fim, no conjunto de dados GEARS-2C-2D (Figura 4.10-*c*)) o algoritmo SCARGC apresenta resultados estáveis durante todo o período de tempo, enquanto o algoritmo MClassification apresenta pequenas oscilações, mas sempre com resultados iguais ou superiores ao classificador Estático. Além disso, em alguns momentos os resultados se aproximam de 100%, próximo ao desempenho obtido pelo classificador Deslizante LN.

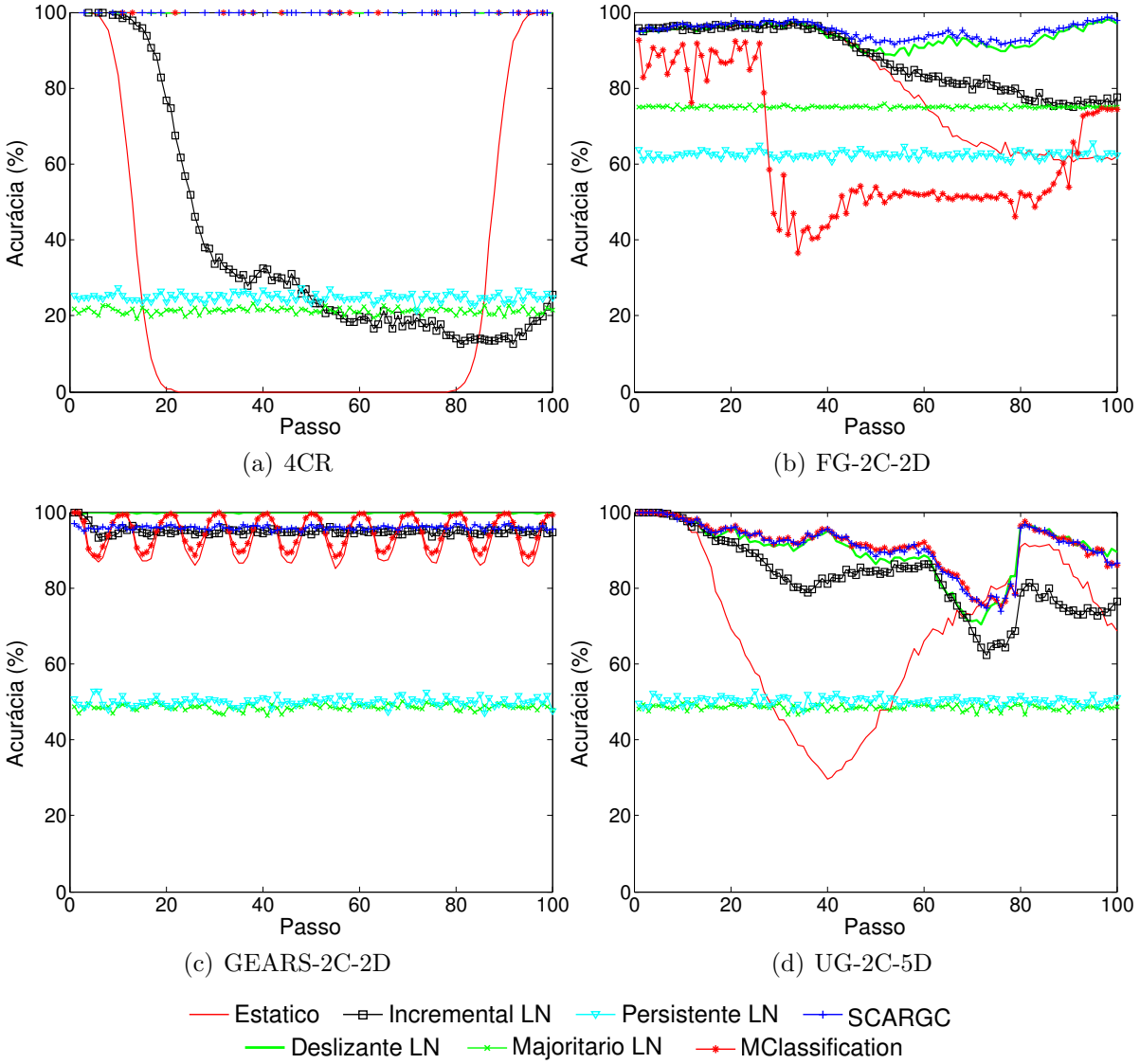


Figura 4.10: Acurácia ao longo do tempo dos algoritmos SCARGC, MClassification e métodos de comparação nos conjuntos de dados 4CR, FG-2C-2D, GEARS-2C-2D e UG-2C-5D.

Conforme anteriormente discutido, tem-se conhecimento do desempenho dos algoritmos APT e COMPOSE somente nos conjuntos de dados UG-2C-2D, UG-2C-3D e MG-2C-2D, relatados em Dyer et al. (2014). Desse modo, é apresentado na Figura 4.11 o desempenho dos algoritmos propostos SCARGC e MClassification em comparação com

os métodos da literatura nestes conjuntos de dados. Para melhor visualização em uma comparação direta, optou-se por omitir o desempenho obtido pelos métodos de comparação. No conjunto de dados UG-2C-2D (Figura 4.11-*a*)), é possível observar que os algoritmos SCARGC e MClassification apresentam resultados similares aos do algoritmo COMPOSE e superiores aos resultados alcançados pelo algoritmo APT. No conjunto de dados UG-2C-3D (Figura 4.11-*b*)), os algoritmos propostos apresentam resultados similares e ambos superam os resultados dos algoritmos COMPOSE e APT. No conjunto de dados MG-2C-2D (Figura 4.11-*c*)), nota-se que o melhor desempenho é apresentado pelo algoritmo SCARGC, seguido do algoritmo da literatura COMPOSE. O algoritmo MClassification apresenta uma queda de desempenho entre os intervalos 30 e 60, com resultados próximos a 65% de acurácia. Entretanto, após este intervalo o algoritmo passa a superar o desempenho do COMPOSE. Além disso, durante quase todo o decorrer do tempo, o algoritmo MClassification supera os resultados apresentados pelo APT.

Os resultados obtidos pelos métodos nos conjuntos de dados reais NOAA e KEYSTROKE são apresentados na Figura 4.12. No conjunto de dados de predição de tempo NOAA, nota-se na Figura 4.12-*a*) que o melhor resultado é obtido pelo algoritmo Incremental LN durante todo o período de tempo. Assim, tem-se que neste problema a presença de conceitos antigos é benéfico ao modelo de classificação. Ainda que, o método Deslizante LN apresente o segundo melhor desempenho entre os métodos avaliados. Embora não apresentem os melhores resultados, deve-se considerar que os algoritmos SCARGC e MClassification apresentam resultados sensivelmente superiores ao classificador Estático. No conjunto de dados de padrões de digitação KEYSTROKE, também nota-se o desempenho superior do método Incremental LN, conforme observado na Figura 4.12-*b*). Entretanto, os algoritmos SCARGC e MClassification apresentam resultados competitivos em relação método Incremental LN e Deslizante LN. Além disso, os resultados alcançados pelos algoritmos propostos são consideravelmente superiores ao método Estático durante todas as 7 sessões de avaliação do problema.

4.6.3 Influência dos Parâmetros

Nesta seção será discutida a influência dos parâmetros no desempenho de classificação das propostas apresentadas neste capítulo, SCARGC e MClassification.

O algoritmo SCARGC possui três parâmetros ou entradas que devem ser fornecidas pelo usuário: *i*) dados iniciais rotulados ($\mathcal{T}$); *ii*) tamanho da memória de curto prazo (θ) que é utilizada para armazenar os exemplos do fluxo que serão posteriormente agrupados; e *iii*) número de grupos (k) a ser considerado na etapa de agrupamento. Para a discussão do impacto da quantidade de dados iniciais rotulados e do tamanho da memória de curto prazo nos resultados do algoritmo SCARGC, foi considerado a variação destes valores nos conjuntos de dados 2CDT, 4CRE-V2 e MG-2C-2D. Os resultados obtidos nestes conjuntos

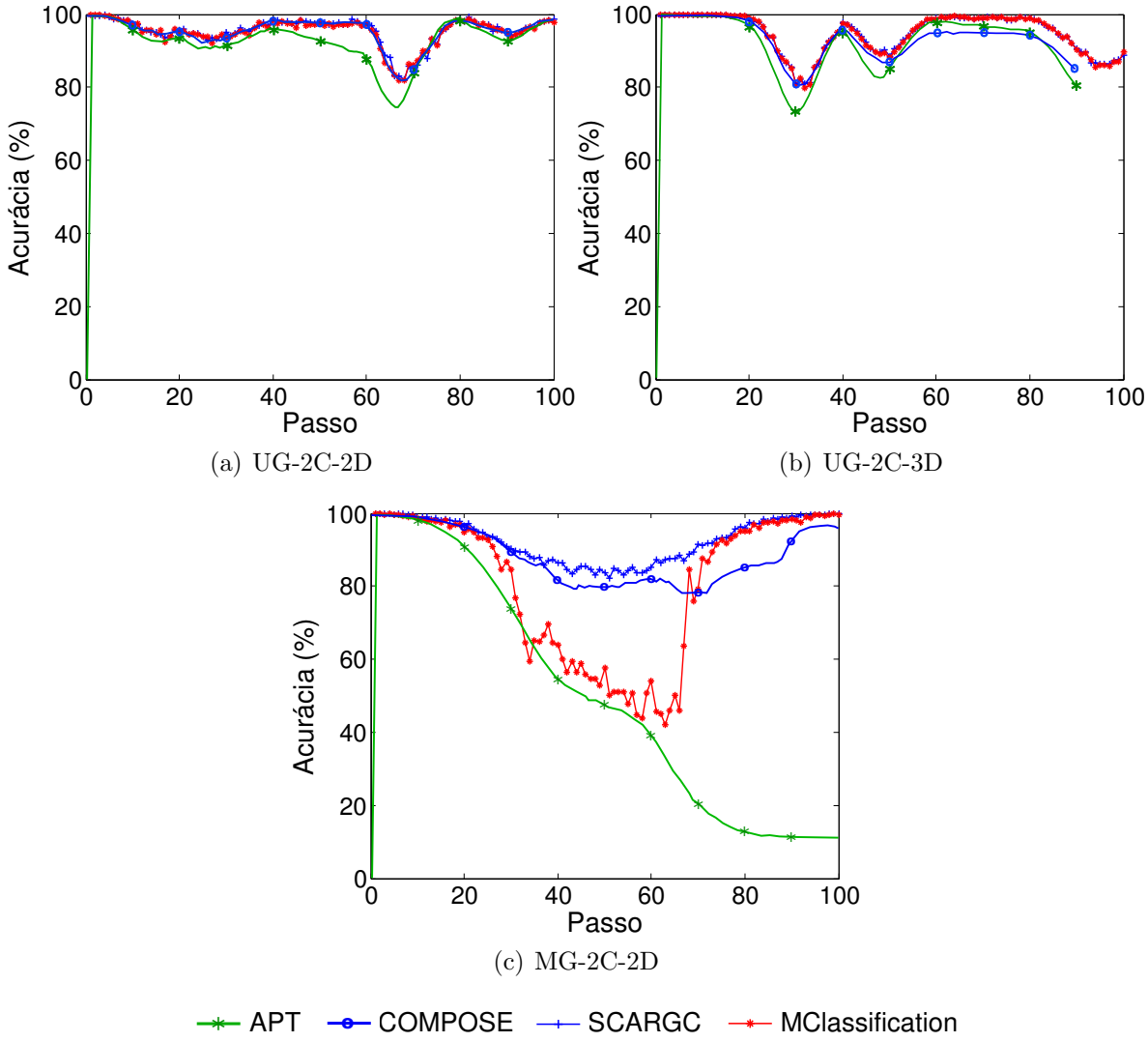
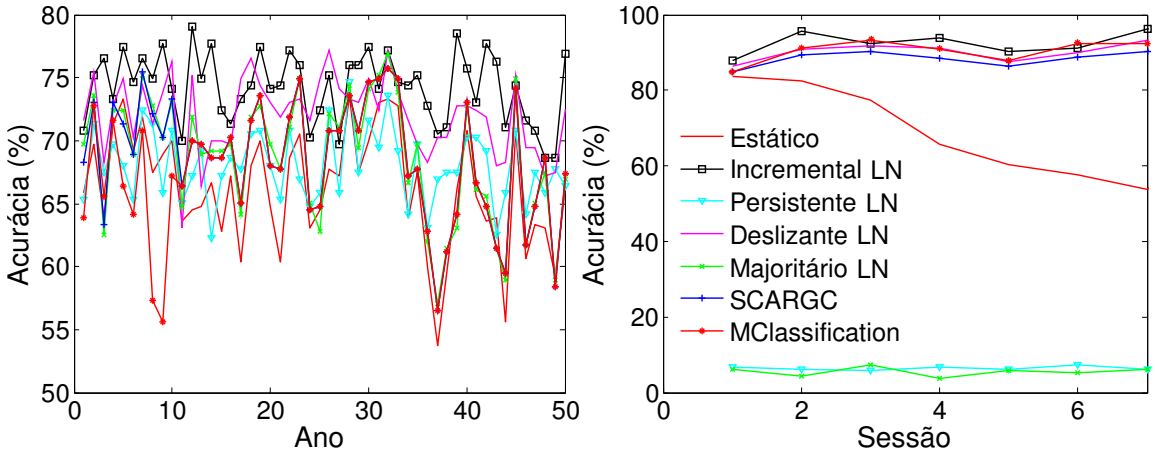


Figura 4.11: Acurácia ao longo do tempo dos algoritmos SCARGC, MClassification, APT e COMPOSE nos conjuntos de dados UG-2C-2D, UG-2C-3D e MG-2C-2D.

de dados são apresentados na Tabela 4.3, Tabela 4.4 e Tabela 4.5, respectivamente, com valores de θ de 200 até 1000 exemplos para o tamanho da memória e $|\mathcal{T}|$ de 50 até 1200 exemplos para o tamanho do conjunto de dados rotulados inicial. Para cada conjunto de dados, os resultados que superam a configuração Deslizante LN são apresentados em negrito, enquanto os resultados que são superados pela configuração Estático são destacados com fundo em cinza. Todos os outros resultados apresentam um valor intermediário entre as configurações Estático e Deslizante LN.

Na Tabela 4.3 é possível observar que considerando o conjunto de dados 2CDT, o algoritmo SCARGC apresenta resultados abaixo da configuração Estático (54, 38%) somente quando o tamanho da memória de curto prazo é maior que 500 exemplos. Estes resultados são esperados, dado que neste conjunto de dados as mudanças ocorrem a cada 400 exemplos. Assim, o uso de uma memória razoavelmente maior acarreta no acumulo



(a) NOAA (b) KEYSTROKE

Figura 4.12: Acurácia ao longo do tempo dos algoritmos SCARGC, MClassification e métodos de comparação nos conjuntos de dados reais NOAA e KEYSTROKE.

Tabela 4.3: Acurácia média obtida pelo algoritmo SCARGC no conjunto de dados 2CDT considerando a variação dos parâmetros θ – tamanho máximo da memória de curto prazo e $|\mathcal{T}|$ – quantidade de exemplos do conjunto de dados inicial.

θ								
1000	59,31	59,19	48,41	48,38	47,84	47,27	57,74	46,09
800	63,31	62,86	50,42	50,30	63,34	62,41	48,31	48,37
600	78,36	78,34	70,71	52,88	71,71	69,92	51,40	70,78
500	83,18	83,16	83,01	82,75	82,79	82,95	82,26	82,19
400	87,73	87,37	86,72	88,05	86,49	87,90	86,34	87,63
300	90,93	90,92	91,27	90,76	90,99	91,08	90,71	90,74
200	**93,51**	**93,60**	93,41	93,33	93,31	93,26	93,23	93,16
$\lvert\mathcal{T}\rvert$	**50**	**100**	**200**	**400**	**600**	**800**	**1000**	**1200**

de um alto número de exemplos desatualizados. Por outro lado, o algoritmo apresenta pouca variação em relação a diferentes valores para o parâmetro $|\mathcal{T}|$.

Na Tabela 4.4 e Tabela 4.5, é possível observar que na maioria dos casos o algoritmo SCARGC apresenta resultados que superam a configuração Deslizante LN, independente dos valores dos parâmetros. Em geral, pode-se indicar que é mais seguro a escolha de um valor pequeno para o tamanho da memória. Entretanto, este tamanho deve ser suficientemente grande para que seja possível obter diferentes grupos na etapa de agrupamento. Em relação ao parâmetro referente aos dados iniciais, nota-se que a quantidade de dados não é responsável por influenciar significativamente os resultados.

Em relação ao parâmetro k do algoritmo SCARGC, referente ao número de grupos utilizado para representar as classes do problema, uma heurística simples é utilizar $k = c$, em que c representa a quantidade de classes. Entretanto, nem sempre é possível realizar o rastreamento da classe a partir do uso de um único grupo. Um exemplo é o conjunto

Tabela 4.4: Acurácia média obtida pelo algoritmo SCARGC no conjunto de dados 4CRE-V2 considerando a variação dos parâmetros θ e $|\mathcal{T}|$.

θ								
1000	**91,38**	**91,40**	**91,39**	**91,30**	**91,31**	**91,40**	**91,40**	**91,35**
800	**91,76**	**91,71**	**91,72**	**91,68**	**91,71**	**91,71**	**91,68**	**91,65**
600	**91,89**	**91,88**	**91,90**	**91,91**	**91,87**	**91,87**	**91,88**	**91,85**
500	**91,95**	**91,92**	**91,94**	**91,99**	**91,90**	**91,93**	**91,89**	**91,90**
400	**91,97**	**91,95**	**91,95**	**91,96**	**91,93**	**91,94**	**91,92**	**91,93**
300	**91,91**	**91,88**	**91,94**	**91,87**	**91,88**	**91,92**	**91,85**	**91,86**
200	**91,65**	**91,70**	**91,73**	**91,72**	**91,72**	**91,71**	**91,70**	**91,69**
$\|\mathcal{T}\|$	**50**	**100**	**200**	**400**	**600**	**800**	**1000**	**1200**

Tabela 4.5: Acurácia média obtida pelo algoritmo SCARGC no conjunto de dados MG-2C-2D considerando a variação dos parâmetros θ e $|\mathcal{T}|$.

θ								
1000	**92,86**	**92,86**	**92,87**	**92,93**	**92,90**	**92,99**	**92,87**	**92,87**
800	**92,89**	**92,94**	**92,87**	**92,91**	**92,91**	**92,88**	**92,84**	**92,88**
600	**92,91**	**92,85**	**92,91**	**92,85**	**92,86**	**92,86**	**92,85**	**92,84**
500	**92,85**	**92,84**	**92,85**	**92,87**	**92,82**	**92,84**	**92,88**	**92,80**
400	**92,81**	**92,87**	**92,78**	**92,85**	**92,75**	**92,84**	**92,75**	**92,82**
300	**92,72**	**92,74**	**92,79**	**92,73**	**92,71**	**92,73**	**92,71**	**92,65**
200	**92,29**	79,66	80,19	80,17	80,15	80,13	80,11	80,09
$\|\mathcal{T}\|$	**50**	**100**	**200**	**400**	**600**	**800**	**1000**	**1200**

de dados MG-2C-2D, anteriormente discutido na Figura 4.3, em que é necessário o uso de dois grupos para representar cada classe.

Nos conjuntos de dados sintéticos avaliados neste trabalho, um ou dois grupos são suficientes para a representação das classes. Nos casos em que esta quantidade de grupos não é suficiente, pode-se utilizar uma grande quantidade de grupos por classe com o objetivo de aproximar o seu formato e realizar o rastreamento das mudanças ao longo do tempo. Por exemplo, considerando o conjunto de dados 1CSurr anteriormente apresentado na Figura 4.4-*b)*, são apresentados os resultados do algoritmo SCARGC em conjunto com o classificador base SVM considerando os valores de $k = \{2, 4, 6, 8, 10\}$ na Figura 4.13. É possível observar que o melhor resultado é obtido com $k = 4$, em que obteve-se uma acurácia média de $94,98\%$. Nota-se que ao final do fluxo de dados, quando a classe $\circ$ está disposta no centro do espaço de atributos e a classe $\triangle$ está distribuída ao redor da outra classe, o algoritmo apresenta uma certa instabilidade com valores diferentes de 4 para o parâmetro k. Entretanto, ainda assim estes resultados superam o obtido pela configuração Estático (65,75%).

Os resultados obtidos a partir da variação dos valores de k para o conjunto de dados GEARS-2C-2D são apresentados na Figura 4.14. Pode-se observar que o melhor resultado é alcançado com $k = 2$. Neste conjunto de dados, a instabilidade apresentada por valores diferentes de 2 para o parâmetro k é justificada pela tendência de "dominância" de uma das classes. Este comportamento é mais evidente quando analisados os resultados com

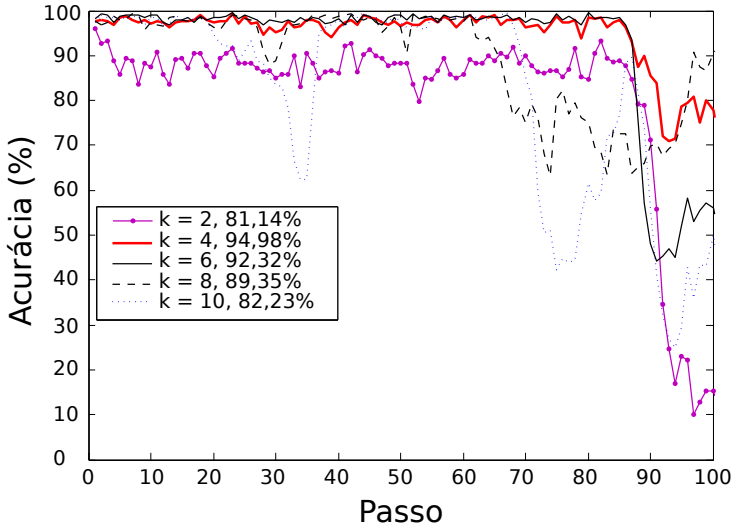


Figura 4.13: Resultados obtidos pelo algoritmo SCARGC (SVM) dada a variação do parâmetro k, considerando o conjunto de dados 1CSurr.

$k = 4$ a partir do passo 40. A partir deste ponto, todos as classificações são atribuídas para uma mesma classe.

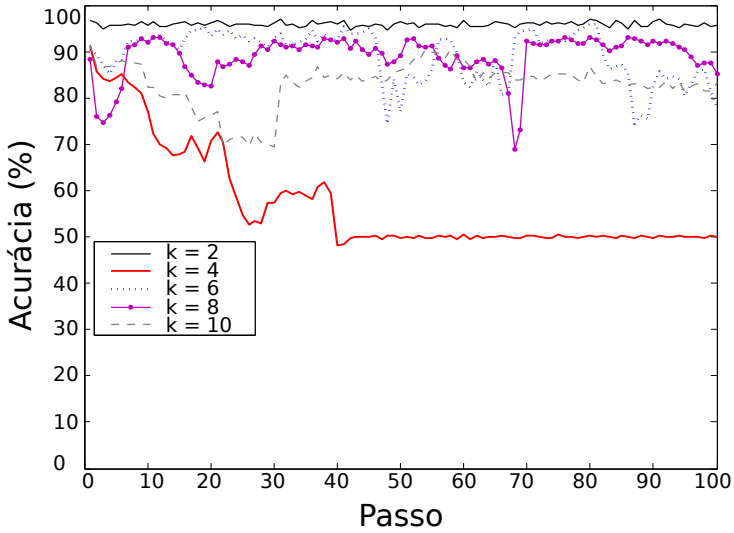


Figura 4.14: Resultados obtidos pelo algoritmo SCARGC (SVM) dada a variação do parâmetro k, considerando o conjunto de dados GEARS-2C-2D.

Em relação ao algoritmo SCARGC, o método proposto MClassification possui um parâmetro a menos, sendo necessário a definição do raio máximo r de cada *Micro-Cluster* e uma quantidade $|\mathcal{T}|$ referente ao número de exemplos rotulados utilizados inicialmente. Assim, foi realizada a avaliação do algoritmo MClassification considerando a variação do parâmetro r partindo de $0,01$ até 1. Em relação ao parâmetro $|\mathcal{T}|$, foi realizada a variação dos valores partindo de 50 até 1200 exemplos. Os resultados obtidos nos

conjuntos de dados 2CDT, UG-2C-2D e 5CVT são apresentados nas Tabelas 4.6, 4.7 e 4.8, respectivamente.

Tabela 4.6: Acurácia média obtida pelo algoritmo MClassification no conjunto de dados 2CDT considerando a variação dos parâmetros θ e r.

r								
1,0	56,55	56,03	55,34	54,92	55,16	54,90	54,99	54,66
0,8	55,89	55,72	55,85	55,12	55,55	55,11	55,25	54,92
0,6	73,71	55,56	55,67	55,56	55,97	55,44	55,63	55,32
0,4	92,35	72,38	57,84	55,43	56,89	56,14	56,18	55,66
0,2	**95,39**	**94,97**	84,29	58,62	57,39	56,76	56,70	56,29
0,1	**95,78**	**95,59**	**94,56**	83,35	64,59	58,52	57,42	56,98
0,01	**95,73**	**95,45**	**94,23**	90,39	81,16	65,50	58,10	56,58
$\lvert\mathcal{T}\rvert$	**50**	**100**	**200**	**400**	**600**	**800**	**1000**	**1200**

Tabela 4.7: Acurácia média obtida pelo algoritmo MClassification no conjunto de dados UG-2C-2D considerando a variação dos parâmetros θ e r.

r								
1,0	47,08	48,47	47,35	46,72	46,71	46,39	46,07	45,81
0,8	44,53	45,60	46,44	46,21	46,15	46,00	46,24	45,50
0,6	87,58	48,25	46,27	43,36	46,13	44,12	47,14	47,22
0,4	83,37	89,04	57,99	53,29	48,62	50,73	47,87	47,75
0,2	**94,90**	**94,56**	**94,54**	68,83	58,57	57,46	56,16	56,06
0,1	**95,03**	**95,22**	**95,28**	**95,35**	**95,34**	93,98	87,85	74,45
0,01	**94,96**	**95,05**	**95,15**	**95,28**	**95,38**	**95,41**	**95,41**	**95,43**
$\lvert\mathcal{T}\rvert$	**50**	**100**	**200**	**400**	**600**	**800**	**1000**	**1200**

Tabela 4.8: Acurácia média obtida pelo algoritmo MClassification no conjunto de dados 5CVT considerando a variação dos parâmetros θ e r.

r								
1,0	38,84	44,09	39,02	43,13	41,74	43,26	44,19	42,02
0,8	37,09	42,79	43,03,	43,02	32,33	38,06	44,25	39,03
0,6	40,83	46,03	39,00	43,15	39,14	43,37	38,64	44,38
0,4	38,05	45,82	45,54	44,24	41,70	40,41	44,53	43,80
0,2	82,60	79,66	63,25	51,55	45,04	46,44	48,43	45,71
0,1	**87,87**	**88,72**	83,98	71,09	57,91	46,24	51,30	49,69
0,01	**86,89**	**87,49**	85,97	78,05	69,02	66,05	49,50	40,97
$\lvert\mathcal{T}\rvert$	**50**	**100**	**200**	**400**	**600**	**800**	**1000**	**1200**

Diferentemente do algoritmo SCARGC, pode-se observar a partir dos resultados obtidos nos conjuntos de dados 2CDT, UG-2C-2D e 5CVT, que a quantidade de dados iniciais rotulados é responsável por impactar o desempenho do algoritmo MClassification. Nota-se que, em geral, o algoritmo não apresenta bom desempenho ao considerar uma quantidade maior de dados. Isto ocorre devido à abordagem utilizada pelo método, que considera a criação de MCs a partir de dados iniciais e realiza a atualização constante de suas localizações ao longo do tempo com um menor número de operações de descarte de

dados. Assim, ao considerar uma quantidade maior de dados, são gerados MCs possivelmente desatualizados, referentes a conceitos anteriores. Como o processo de atualização e descarte de MCs é lento, o algoritmo apresenta um desempenho insatisfatório. Em relação ao parâmetro r, aconselha-se utilizar um valor pequeno entre $0,01$ e $0,2$. Valores acima de $0,2$ podem levar a inclusão de exemplos da classe incorreta no MC, o que consequentemente acarretará na deterioração do desempenho do algoritmo.

4.6.4 Tempo Computacional

Em problemas de classificação de fluxo de dados, a eficiência de tempo é um requisito fundamental para a maior parte das aplicações. Como os dados chegam sequencialmente ao longo do tempo, o algoritmo de classificação deve ser capaz de fornecer uma resposta para cada exemplo em tempo hábil antes da chegada do próximo exemplo do fluxo de dados, para que este exemplo não seja descartado.

Na Tabela 4.9 é apresentado o tempo computacional (em minutos) gasto pelos algoritmos para realizar a classificação de todos os exemplos dos conjuntos de dados sintéticos. Os tempos apresentados na tabela representam a média de 10 execuções de cada algoritmo em um computador com processador de 2,4 GHz e 8 GB de memória. Todos os algoritmos foram implementados em MATLAB. Como o algoritmo SCARGC permite a utilização de qualquer classificador base, é apresentado o seu tempo computacional utilizando os classificadores 1NN e SVM. Assim como os resultados de acurácia, tem-se conhecimento do tempo computacional dos algoritmos APT e COMPOSE somente em relação aos conjuntos de dados UG-2C-2D, UG-2C-3D e MG-2C-2D, conforme os resultados relatados por Dyer et al. (2014).

Nota-se a partir dos resultados apresentados na Tabela 4.9, que a simplicidade do algoritmo SCARGC, proposto neste trabalho, reflete em seu custo computacional. Em especial, ao utilizar o algoritmo SVM como classificador base, pode-se verificar que para qualquer conjunto de dados avaliado, o algoritmo SCARGC não apresentou custo computacional superior a 1 minuto para classificar todos os exemplos. Este é um resultado bastante motivador, considerando que os maiores conjuntos de dados possuem 200 mil exemplos. É importante destacar que parte desta eficiência computacional alcançada com o classificador SVM se deve ao uso da implementação em C do classificador, nomeada SVM^{light} (Joachims, 1999). Ao utilizar o classificador 1NN, o custo de atualização do modelo é drasticamente reduzido, uma vez que os próprios exemplos são utilizados nesta etapa. Entretanto, a fase de classificação é mais custosa quando comparada ao uso do classificador SVM. Desse modo, o algoritmo SCARGC com o classificador base 1NN apresenta um tempo computacional maior em relação a configuração com o algoritmo SVM. Ainda assim, os tempos de execução são consideravelmente inferiores à segunda abordagem proposta neste trabalho, MClassification.

Tabela 4.9: Tempo computacional (em minutos) dos algoritmos propostos para a classificação dos conjuntos de dados sintéticos.

Conjunto de dados	**SCARGC (1NN)**	**SCARGC (SVM)**	**MClassification**	**APT**	**COMPOSE**
1CDT	0,158	0,030	1,240	–	–
2CDT	0,160	0,033	0,883	–	–
1CHT	0,158	0,028	0,918	–	–
2CHT	0,159	0,028	0,891	–	–
4CR	1,446	0,458	8,763	–	–
4CRE-V1	1,261	0,193	5,039	–	–
4CRE-V2	1,857	0,370	8,679	–	–
5CVT	0,408	0,091	1,266	–	–
1CSurr	0,541	0,146	3,156	–	–
4CE1CF	1,782	0,646	10,451	–	–
UG-2C-2D	0,996	0,168	5,143	∼2,5 dias	4,16
UG-2C-3D	1,973	0,346	12,255	∼15 dias	26,66
UG-2C-5D	2,003	0,519	13,739	–	–
FG-2C-2D	1,995	0,337	11,949	–	–
MG-2C-2D	1,961	0,356	10,662	∼14 dias	8,33
GEARS-2C-2D	1,936	0,255	8,470	–	–

Por ser um algoritmo incremental que realiza atualizações a cada novo exemplo, o algoritmo MClassification apresenta um tempo computacional superior ao SCARGC. Entretanto, não possui um parâmetro relativo à definição do número de grupos do problema. Ainda assim, ao comparar os resultados apresentados pelo algoritmo MClassification e COMPOSE nos conjuntos de dados UG-2C-2D, UG-2C-3D e MG-2C-2D, pode-se notar que ambos apresentam tempos computacionais equivalentes. Ainda, pode-se verificar que um número maior de atributos é responsável por um impacto menor no algoritmo MClassification do que no algoritmo COMPOSE, como pode ser visto nos resultados apresentados para o conjunto de dados UG-2C-3D, com 3 atributos. Embora o algoritmo MClassification não seja o mais eficiente entre os algoritmos avaliados, é importante destacar que um tempo computacional de 13,739 minutos (pior caso, conjunto de dados UG-2C-5D) para classificar 200 mil exemplos, representa uma taxa média adequada de $0,00412$ segundo para classificar cada exemplo do fluxo de dados. Por fim, o algoritmo APT se mostrou o menos eficiente em termos de custo computacional entre os avaliados, com tempos de execução que ultrapassam dias.

4.7 Considerações Finais

Neste capítulo, foram apresentadas duas propostas de algoritmos (*Stream Classification Algorithm Guided by Clustering* – SCARGC e *Micro-Cluster Classification* – MClassification) para lidar com um problema recorrente em aplicações reais que está relacionado ao atraso infinito na obtenção dos rótulos corretos após a classificação dos exemplos em

fluxo de dados: *latência extrema*. O problema de latência extrema para o recebimento de rótulos corretos é um problema em aberto na pesquisa de mineração de fluxo de dados não estacionários e foi recentemente apontado como um desafio que merece atenção de pesquisadores da área (Krempl et al., 2014). Entretanto, nota-se que o problema ainda é pouco explorado na literatura e novos algoritmos de classificação ou métodos de detecção de mudanças de conceito são propostos com a suposição otimista de que após a classificação de todo exemplo de um fluxo de dados, o seu rótulo correto é fornecido pelo ambiente ou processo gerador dos dados, sem qualquer atraso de tempo. Tem-se conhecimento de somente duas abordagens na literatura que lidam com o problema até o presente momento, *Arbitrary Sub-Populations Tracker* – APT (Krempl, 2011) e *Compacted Object Sample Extraction* – COMPOSE (Dyer et al., 2014), os quais foram apresentadas e discutidas neste capítulo. Desse modo, acredita-se que as contribuições deste trabalho constituem um importante avanço para a área e espera-se que as limitações presentes nas abordagens motivem outros pesquisadores na proposta de novas abordagens para o problema de latência extrema.

Para a avaliação dos algoritmos propostos, foram construídos e compilados um total de 16 conjuntos de dados sintéticos com características e comportamentos específicos que buscou-se avaliar, como grande volume e mudanças de conceito incrementais ao longo do tempo. Estes conjuntos de dados são publicamente disponibilizados para a comunidade acadêmica e espera-se que estes dados possam contribuir para outros pesquisadores da área. Além de dados sintéticos, os algoritmos também foram avaliados em 2 conjuntos de dados reais. Em geral, os métodos propostos apresentaram um bom desempenho em acurácia e tempo computacional, com resultados equivalentes ou superiores aos apresentados na literatura.

Assim como as abordagens atuais da literatura, os métodos propostos apresentam limitações que serão foco de estudo em trabalhos futuros. Por exemplo, o algoritmo SCARGC é altamente dependente de um algoritmo de agrupamento para realizar a etapa de adaptação a possíveis mudanças de conceito nos dados. Assim, o algoritmo SCARGC herda as limitações destes algoritmos. Nos experimentos realizados neste trabalho, foi utilizado o algoritmo de agrupamento k-Médias, que assume uma distribuição Gaussiana esférica para os dados. Este problema pode ser minimizado a partir do uso de um número maior de grupos para representar uma classe. Além disso, o algoritmo k-Médias apresenta limitações quando os grupos possuem tamanhos e densidades diferentes. Outra limitação do algoritmo SCARGC é relativa ao parâmetro k, que define a quantidade de grupos da classe de um problema. Além de ser constante durante toda a execução do algoritmo, nem sempre é trivial encontrar um valor adequado. Em problemas reais, é comum que o número de grupos referentes a uma classe se altere ao longo do tempo. Em trabalhos futuros pretende-se adaptar o algoritmo para que: i) encontre automaticamente uma quantidade adequada de grupos para representar um problema; ii) seja capaz de realizar comparações

de similaridade entre grupos antigos e grupos atuais afim de encontrar mudanças de conceito, mesmo que a quantidade de grupos seja diferente, e *iii*) sejam utilizadas outras medidas além da similaridade entre os centroides dos grupos.

Visando a remoção do parâmetro k e a restrição da quantidade constante de grupos ao longo do tempo, foi apresentado o algoritmo MClassification. Entretanto, a principal limitação do algoritmo é ser suscetível a presença de valores atípicos (*outliers*) e ruído, que podem influenciar negativamente no comportamento do algoritmo. Como os conjuntos de dados sintéticos propostos não levam em consideração a presença de ruído nos dados, esta limitação não é discutida na avaliação experimental do algoritmo. Em trabalhos futuros, pretende-se explorar novas estratégias para a manutenção *online* dos *Micro-Clusters* para que o algoritmo lide melhor com a presença de valores atípicos e ruído. Uma ideia inicial é atribuição de pesos aos *Micro-Clusters* de acordo com a sua idade ou utilidade e que a criação e deslocamento de *Micro-Clusters* leve em consideração um número maior de exemplos.

Por fim, uma limitação inerente ao problema de classificação de fluxo de dados sob condições de latência extrema, tanto nas abordagens propostas como nos métodos da literatura, é a suposição da ocorrência de mudanças de conceito incrementais. Embora seja muito difícil ou até mesmo impossível lidar com este problema em situações onde as mudanças de conceito sejam abruptas, acredita-se que no futuro com a melhor consolidação deste problema de pesquisa, seja possível realizar a adaptação dos classificadores sob condições de mudanças graduais.

Capítulo

5

Sensor para a Identificação Automática de Insetos

5.1 Considerações Iniciais

Com o desenvolvimento da área de Mineração de Dados, novas e importantes aplicações têm surgido. Um exemplo é o desenvolvimento de sensores inteligentes capazes de coletar informações sobre o ambiente, bem como realizar decisões baseadas nos dados de entrada. Um exemplo concreto é o sensor para a identificação automática de insetos que foi estudado neste trabalho. O desenvolvimento deste sensor se iniciou em 2011 em uma colaboração entre pesquisadores do Instituto de Ciências Matemáticas e de Computação da Universidade de São Paulo (Prof. Gustavo Batista), do Departamento de Ciência da Computação e Engenharia da Universidade da Califórnia, Riverside (Prof. Eamonn Keogh) e a empresa de manejo de pragas agrícolas *ISCA Technologies* (Batista et al., 2011b). Este sensor utiliza um feixe laser e técnicas de Aprendizado de Máquina para classificar insetos de acordo com suas espécies. Tal sensor é um importante passo para o desenvolvimento de armadilhas inteligentes capazes de atrair e capturar seletivamente espécies de insetos de interesse (como as transmissoras de doenças e pragas agrícolas), rejeitando a captura das demais espécies (como abelhas, mariposas, borboletas e outros polinizadores), importantes para o meio ambiente. A apresentação deste sensor será realizada no decorrer deste capítulo.

Uma contextualização sobre o papel dos insetos na vida das pessoas e os métodos atuais de controle serão apresentadas na Seção 5.2. Na Seção 5.3 será apresentado o estado-da-arte sobre a classificação automática de insetos por meio de equipamentos ele-

trônicos. Na Seção 5.4 será apresentado o sensor identificador de insetos estudado neste trabalho e como este sensor poderá ser futuramente utilizado em conjunto com armadilhas inteligentes. Na Seção 5.5 são apresentadas as diferentes representações dos dados obtidos pelo sensor. Os possíveis atributos extraídos dos dados que podem ser utilizados para a caracterização das espécies de insetos são apresentados na Seção 5.6. Os procedimentos realizados para a etapa de coleta de dados utilizando o sensor laser em laboratório e a avaliação experimental sobre o desempenho de algoritmos de Aprendizado de Máquina em conjunto com diferentes atributos descritivos para a classificação de espécies de insetos são apresentadas nas Seções 5.7 e 5.8, respectivamente. Por fim, na Seção 5.9 é discutido o problema de classificação com uma classe em que a etapa de treinamento é realizada somente com uma classe de interesse, bem como a avaliação de diferentes algoritmos desta abordagem para a classificação de insetos da espécie *Aedes aegypti*.

5.2 Papel dos Insetos no Meio-ambiente

A humanidade possui uma estreita relação com insetos. Por exemplo, insetos são vetores de doenças que matam milhões de pessoas por ano e deixam outras dezenas de milhões adoecidas. Estima-se que a dengue, doença transmitida por fêmeas dos mosquitos do gênero *Aedes*, é endêmica em mais de 100 países e afete cerca de 390 milhões de pessoas por ano em todo o mundo, sendo que deste total, aproximadamente 96 milhões de pessoas apresentam manifestações clínicas (W.H.O., 2015). A malária, transmitida por mosquitos do gênero *Anopheles*, afeta aproximadamente 6% da população mundial e estima-se que existam cerca de 200 milhões de casos da doença por ano e aproximadamente 7 milhões de casos letais na última década (W.H.O., 2014).

Na agricultura e pecuária, insetos causam enormes prejuízos financeiros ao atacar colheitas e animais. Frequentemente os insetos estão na raiz do problema chamado insegurança alimentar, referente ao risco de perdas significativas na produção de alimentos (Vreysen e Robinson, 2011). Ao mesmo tempo, insetos possuem importante papel para o equilíbrio ecológico, já que são fontes de alimentos para outras espécies animais, auxiliam na reprodução de espécies vegetais e na produção agrícola ao realizarem o processo de polinização. Estima-se que os insetos sejam responsáveis pela polinização de aproximadamente 80% das espécies empregadas na agricultura, sendo que um terço de todo alimento consumido no mundo é polinizado somente por abelhas (Dixon, 2009). Além disso, insetos têm sido utilizados como bioindicadores de qualidade ambiental, de modo que a sua presença/ausência, distribuição e densidade, permitem definir a qualidade do ecossistema, especialmente em relação a contaminantes do ar, solo e água (Kevan, 1999). De acordo com Wilson (1999):

> "*Se toda a humanidade desaparecesse, o mundo iria se regenerar ao rico estado de equilíbrio existente há dez mil anos. Por outro lado, se todos os insetos desaparecessem, o meio ambiente entraria em colapso e seria um caos.*"

Desse modo, ainda que existam espécies de insetos responsáveis por diversos problemas sociais e econômicos, a humanidade provavelmente não sobreviveria além de poucos meses sem a sua presença dado o importante papel para o equilíbrio do ecossistema. Considerando essa dualidade e a falta de vacinas e medicamentos eficazes para o tratamento de doenças transmitidas por insetos, pesquisadores têm desenvolvido nas últimas décadas um arsenal de métodos químicos, biológicos, mecânicos e educacionais de controle de insetos. Alguns exemplos são (Walker, 2002):

- Uso de redes tratadas com inseticidas;
- Pulverização de inseticidas, pesticidas e larvicidas;
- Introdução de peixes e crustáceos que se alimentam de larvas;
- Dispersão de mosquitos machos estéreis;
- Campanhas educacionais para a redução do *habitat* do inseto, como a remoção de objetos que podem acumular água parada;
- Uso de armadilhas.

Para serem empregados, tais métodos de controle requerem o conhecimento da distribuição espaço-temporal dos insetos. Sem tal conhecimento, o emprego dessas técnicas se torna custoso e ineficiente, além de, no caso do uso de inseticidas, apresentar uma ameaça à vida de outros insetos, pássaros e demais espécies de vida nativa (Colborn et al., 1993; Nakamaru et al., 2002).

Atualmente, estudar a distribuição espaço-temporal de insetos é uma tarefa custosa e demorada. Em linhas gerais, contagens de insetos são realizadas por meio de armadilhas, geralmente adesivas, que são recolhidas periodicamente e analisadas por especialistas que identificam e contam manualmente as espécies de insetos coletadas. Dois exemplares de armadilhas adesivas são apresentadas na Figura 5.1.

Além de ser uma abordagem cara em termos de tempo dos especialistas, o uso de armadilhas como as ilustradas na Figura 5.1, possui um atraso entre o momento de instalação e análise. Tal atraso pode ser de apenas uma semana, mas esse tempo pode representar mais de meia vida de um inseto adulto. Dessa maneira, a doença já pode ter contaminado um grande número de pessoas no momento em que os dados sobre a presença de determinada espécie estejam disponíveis para as autoridades responsáveis pelo controle dos insetos, como agentes comunitários de saúde (Patnaik et al., 2007). Além disso, a

Figura 5.1: Exemplares de armadilhas adesivas para a captura de insetos.

identificação manual das espécies é dificultada devido ao estado de decomposição dos insetos capturados. Existe, portanto, uma necessidade por sensores automáticos e precisos capazes de detectar, classificar e contar, em tempo real, insetos de diferentes espécies.

5.3 Trabalhos Relacionados

A ideia de classificar insetos em suas espécies automaticamente não é recente, tendo ocorrido antes da popularização dos computadores e de equipamentos de gravação de áudio. Em 1945, três pesquisadores da Faculdade de Medicina da Universidade de Cornell, Kahn, Celestin e Offenhauser, gravaram e analisaram o som emitido por insetos transmissores de doenças, dificilmente audíveis ou inaudíveis por humanos devido a sua faixa de frequência (Kahn et al., 1945). Nesse estudo os pesquisadores utilizaram um microfone, um gravador de áudio convencional, um amplificador de sinal e um filtro passa-banda. As gravações foram realizadas em ambiente sem a influência de sons externos e em condições controladas de temperatura e umidade, sendo contempladas quatro espécies de insetos de ambos os sexos: *Anopheles quadrimaculatus, Aedes aegypti, Aedes albopictus* e *Culex pipiens*. Os pesquisadores identificaram cantos de acasalamento, cantos de alerta de perigo e cantos de raiva, além do registro de sons emitidos pelas batidas de asas durante o voo. Em estudo posterior com espécies de mosquitos africanos, além da identificação de características importantes relacionadas a faixa de frequência, intensidade e vibrato, obtidas a partir da análise manual dos espectrogramas dos sinais registrados, Kahn e Offenhauser Jr (1949) expressam em seu trabalho que a evolução dos equipamentos eletrônicos tornariam possíveis observações precisas e rápidas de fenômenos naturais relacionados aos sons de insetos transmissores de doenças, o que também possibilitaria o controle mais eficaz desses

insetos e das doenças que transmitem. Entretanto, não houve progresso significativo nesse sentido no período transcorrido de aproximadamente sessenta anos.

Desde os primeiros estudos de detecção acústica de insetos como os trabalhos de Kahn et al. (1945); Kahn e Offenhauser Jr (1949) ou de trabalhos semelhantes conduzidos ao longo dos anos por Jones (1964); Belton e Costello (1979); Mankin (1994); Campbell et al. (1996), os autores relatam a observação de padrões de faixas de frequências emitidas por insetos de mesma espécie, bem como padrões nas faixas de frequências emitidas por insetos de sexo oposto, sendo possível caracterizar a espécie e sexo. Entretanto, em boa parte destes estudos os insetos foram analisados individualmente e em pequenas amostras. Além disso, devido ao uso de microfones para a captura dos sinais, uma parte dos trabalhos relata a dificuldade de controle do ruído gerado por fatores externos. Outro fator que prejudica as análises é o nível variável do som capturado, já que depende da distância do inseto ao microfone. Desse modo, as abordagens utilizadas por esses autores, bem como as tecnologias aplicadas, tornam tais pesquisas limitadas em termos de tempo para coleta e análise de uma amostra significativa de dados e dificulta a reprodução dos experimentos em um ambiente real. Esses trabalhos foram importantes para a evolução da área de pesquisa, mas devido às suas limitações não serão abordados em destaque neste trabalho. Entretanto, com base nesses estudos pode-se concluir que a frequência de batidas de asas é uma característica importante para a classificação de insetos em suas espécies e a escolha aparentemente óbvia de um sensor, como um microfone nesses casos, pode não ser a melhor solução para o problema.

Com o objetivo de realizar classificações de insetos sem as desvantagens impostas pelo uso de microfones, Moore et al. (1986); Moore (1991, 1998); Caprio et al. (2001) e Moore e Miller (2002) utilizaram sensores ópticos para detecção da frequência de batidas de asas. Uma visão geral sobre estes trabalhos é apresentada a seguir.

Em estudos preliminares, Moore et al. (1986) utilizaram um fotosensor baseado no tacômetro óptico desenvolvido por Unwin e Ellington (1979) para registrar digitalmente a frequência de batidas de asas de duas espécies de insetos: *Aedes aegypti* e *Aedes triseriatus*, tanto do sexo masculino quanto feminino. Os autores relatam a possibilidade de classificar diferentes espécies de insetos a partir do reconhecimento de padrões harmônicos encontrados nos sinais obtidos pelo fotosensor, de maneira análoga ao realizado pelo ouvido humano para distinguir diferentes instrumentos musicais que emitem uma mesma nota musical. Para testar essa ideia, Moore (1991) utilizou os dados obtidos em seu trabalho anterior para treinar uma rede neural artificial. Para o treinamento, foram extraídas dos dados a frequência de batidas de asas e a amplitude das primeiras quatro harmônicas de 75 sinais das duas espécies de ambos os sexos, totalizando 300 exemplos. A rede treinada somente com informações sobre a frequência de batidas de asas foi capaz de identificar a espécie correta e o sexo com taxa de acerto de 84% utilizando 15 sinais

de cada espécie e sexo, totalizando 60 exemplos para o teste. Os autores relatam que a amplitude das harmônicas não contribuíram para a melhora dos resultados.

Posteriormente foi desenvolvido um sensor óptico para o monitoramento de insetos alados com a possibilidade de ser aplicado em campo (Moore, 1998), conforme apresentado na Figura 5.2. Tal sensor é basicamente composto por um recipiente transparente localizado entre um fotosensor ligado a um computador e uma lâmpada halógena ligada a uma fonte de energia regulada. O fotosensor em conjunto com o computador são responsáveis por capturar a variação de luz emitida pela lâmpada halógena dada as passagens dos insetos dentro do recipiente.

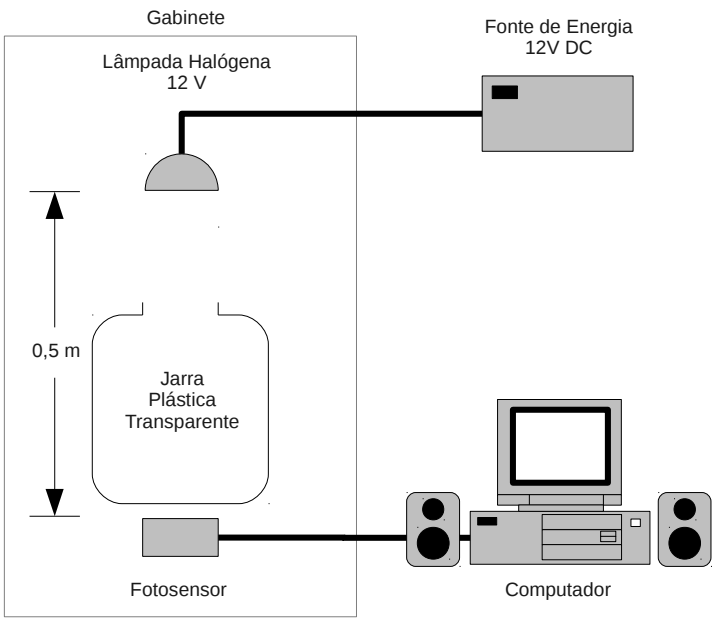


Figura 5.2: Representação do sensor para gravação da frequência de batidas de asas de insetos proposto por Moore (1998); Moore e Miller (2002).

Em Moore e Miller (2002) são relatados experimentos com cinco diferentes espécies de pulgões: *Aphis craccivora, Toxoptera citricida, Aphis nerii, Pentalonia nigronervosa* e *Aphis gossypii*. Os registros dos sinais foram realizados em laboratório com a temperatura de 22°C e variações de no máximo 3°C e possuem duração de uma a três horas. Em vez de utilizar somente os valores da frequência de batidas de asas, foi utilizada a forma de onda das frequências como uma assinatura de cada espécie. A melhor proporção de formas de onda corretamente associadas às espécies foi alcançada com o uso de redes neurais artificiais. Nesse caso, os autores relatam taxa de acerto de 68%, variando de 94% para a espécie *Aphis gossypii* e 41% para *Aphis craccivora*. Segundo os autores, a baixa taxa de acerto geral se deve a grande variância na forma de onda registrada para cada espécie. Tal variação pode ter ocorrido devido ao registro de batidas de asas de pulgões que não estavam voando, mas apenas caminhando sobre o sensor. As diferentes orientações dos

insetos e a distância da fonte de luz também podem ter gerado diferenças nas formas de onda registradas.

De modo semelhante e ainda com o sensor de Moore (1998), Caprio et al. (2001) registraram e analisaram a frequência de batida de asas de três diferentes espécies de insetos do gênero *Anopheles*: *quadrimaculatus, smaragdinus* e *maverlius*. Os autores relatam que a frequência observada nas fêmeas varia entre 320 e 480 batidas por segundo e que não existem diferenças significativas entre as espécies. Para os machos foi observada uma frequência maior, entre 500 e 770 batidas por segundo, sendo que a média da frequência de batidas de asas da espécie *Anopheles maverlius* foi significativamente diferente das outras duas espécies.

5.4 Sensor Laser e Armadilha Inteligente

No decorrer do desenvolvimento deste trabalho, foi explorado um sensor de baixo custo para capturar informações de insetos à distância e classificá-los por meio de algoritmos de Aprendizado de Máquina. Esse sensor utiliza componentes de baixo custo como lasers que podem ser facilmente encontrados em lojas de variedades eletrônicas e fototransistores encontrados em controles remotos utilizados em televisores. É importante que o sensor seja de baixo custo quando produzido em grande quantidade (por volta de R$20,00) para permitir o seu amplo uso em áreas pobres do globo. Um sensor com baixo valor comercial e de troca desestimula o furto, um problema sério para um dispositivo que precisa ser deixado desacompanhado em ambientes externos.

Uma representação esquemática do sensor é apresentada na Figura 5.3. O sensor consiste basicamente de três componentes: *i*) um emissor de luz laser planar similar aos utilizados em serras de madeira para marcar o local de corte; *ii*) um conjunto de fototransistores dispostos lado a lado; e *iii*) um circuito especialmente projetado para filtrar, amplificar e armazenar os sinais capturados pelos fototransistores. Quando um inseto alado cruza o laser, as suas asas ocluem parcialmente a luz causando pequenas variações que são capturadas pelos fototransistores e transmitidas para a placa de circuito. Essas variações ocorrem em um curto espaço de tempo e possuem diferentes magnitudes. Assim, a passagem é representada na forma de uma série temporal e mais especificamente, armazenada como um arquivo de áudio. A partir desse arquivo é possível extrair importantes informações do sinal como a frequência de batida de asas do inseto que cruzou a luz laser.

O sinal gerado é muito similar ao sinal de áudio capturado por um microfone, ainda que os dados sejam obtidos opticamente. Entretanto, o sensor possui uma vantagem importante sobre o áudio capturado por microfones: o sensor é imune a qualquer agente que não atravesse o laser e, portanto, não sofre qualquer interferência externa como a conversa de pessoas, o canto de pássaros e o trânsito de carros e aviões próximos ao local de coleta dos dados.

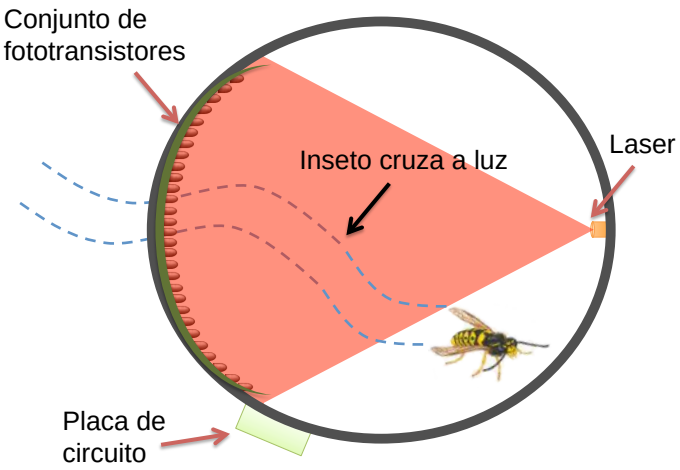


Figura 5.3: Representação esquemática do sensor laser. Uma luz laser planar é direcionada a um conjunto de fototransistores. Quando um inseto alado cruza a luz laser, a variação é registrada pelos fototransistores na forma de uma série temporal e transmitida para uma placa de circuito responsável por filtrar, amplificar e armazenar os sinais gerados.

Os dados coletados pelo sensor são constituídos, em geral, de ruído de fundo com "eventos" ocasionais, resultado dos breves cruzamentos do inseto com o laser. Na Figura 5.4 é exibido um exemplo do dado coletado pelo sensor, obtido pela passagem de um mosquito da espécie *Aedes aegypti*, vetor de doenças como dengue e febre amarela. O ruído de fundo ocorre devido às características físicas do laser, que possui frequentes variações com amplitudes próximas de zero. Nota-se que o sinal gerado pela passagem do inseto possui uma amplitude significativamente maior que a amplitude do ruído de fundo. Dessa maneira, contar o número de insetos que cruzam a luz pode ser considerada uma tarefa relativamente simples, enquanto classificar os sinais de acordo com as possíveis espécies é uma tarefa mais complexa.

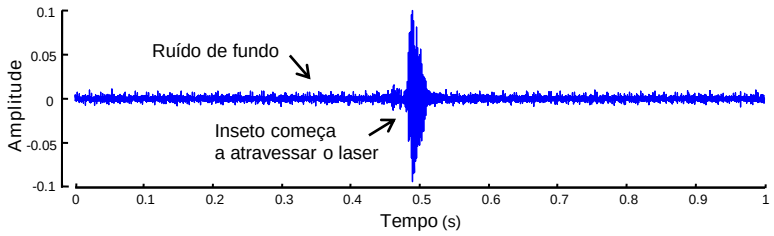


Figura 5.4: Exemplo de dado coletado pelo sensor após a passagem de um inseto da espécie *Aedes aegypti* pelo laser.

O sensor discutido neste trabalho é um importante passo para o desenvolvimento de armadilhas inteligentes capazes de atrair, capturar e classificar automaticamente espécies de insetos. A Figura 5.5 é uma representação esquemática do protótipo da armadilha inteligente que funcionaria em conjunto com o sensor laser para realizar as tarefas de

contagem e classificação de insetos. Na Figura 5.5-*direita* é apresentada uma armadilha tradicional que poderia ser adaptada para trabalhar em conjunto com o sensor. Nas armadilhas tradicionais, as contagens de insetos são realizadas manualmente por um especialista. Portanto, atualmente essas armadilhas são mais ferramentas entomológicas do que um meio efetivo de controlar populações de mosquitos. Por outro lado, a realização automática destas tarefas com o uso da armadilha permitiria uma série de ações, como a construção de mapas de densidade populacional de mosquitos da dengue a partir de informações de diferentes armadilhas instaladas em uma determinada região. Por sua vez, estes mapas permitiriam ações mais efetivas e em tempo hábil (preferencialmente antes da ocorrência de infestações) por parte de órgãos governamentais. Além disso, a armadilha também seria responsável pelo controle populacional de insetos adultos.

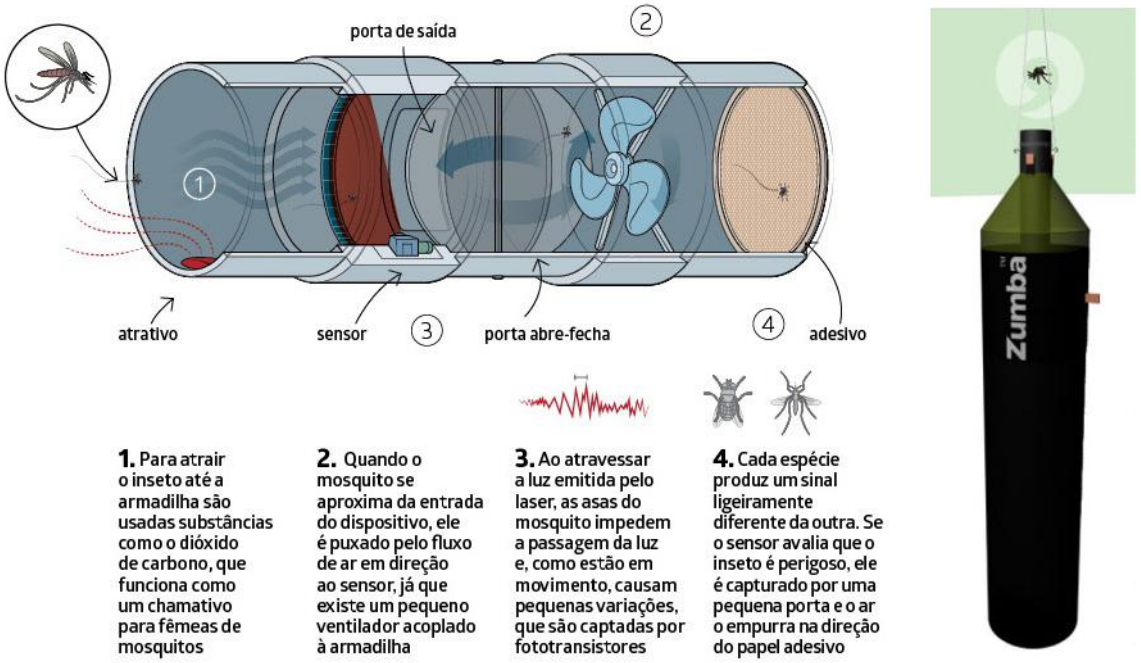


Figura 5.5: Representação esquemática de um protótipo da armadilha inteligente. A esquerda é apresentado o funcionamento da armadilha inteligente em conjunto com o sensor laser e a direita é apresentada uma armadilha tradicional para a captura de insetos e que poderia ser adaptada com o sensor laser. Imagem adaptada de Bertol (2014).

Existem diferentes aplicações para armadilhas inteligentes. Em saúde pública, a armadilha pode capturar mosquitos como os pertencentes aos gêneros *Aedes* e *Anopheles*, vetores da dengue e da malária, respectivamente. Em agricultura e pecuária, a armadilha pode capturar pragas que causam prejuízos como o psilídeo *Diaphorina citri*, vetor da mais preocupante doença dos citros conhecida atualmente (Bové, 2006). Ao mesmo tempo, a armadilha também permite libertar outras espécies de insetos que não são pragas ou vetores de doenças, restringindo o impacto de sua presença no meio-ambiente. Essa é uma importante característica, uma vez que a maior parte dos insetos como as abelhas,

vespas, mariposas e borboletas, possui um relevante papel para o equilíbrio ecológico, já que são fontes de alimentos para outras espécies animais, auxiliam na reprodução de espécies vegetais e na produção agrícola ao realizarem o processo de polinização e dispersão de sementes, além de serem responsáveis pela produção de substâncias úteis para o homem como o mel, a cera, a laca e a seda (Scudder, 2009; Waldbauer, 2009). Assim, deseja-se que a armadilha inteligente retire apenas as espécies maléficas de insetos do meio-ambiente.

5.5 Representação dos Dados

Todo sinal de áudio como o obtido pelo sensor pode ser representado de diversas maneiras. Sua representação primitiva descreve a amplitude de sua forma de onda em cada momento de tempo. Essa representação, conforme ilustrado na Figura 5.4, é chamada temporal. Muitas características do sinal podem ser difíceis de estimar nesta representação. Desse modo, uma representação conveniente para a análise é a espectral, na qual o sinal é transformado do domínio do tempo para o domínio das frequências.

Para analisar as frequências de um sinal complexo, é necessário decompô-lo em uma soma de formas de onda simples como os da família de senos e cossenos, denominados de componentes elementares. Quando um sinal é periódico, essa decomposição se torna uma série de sinais senoidais e cossenoidais de diferentes amplitudes e frequências, denominada Série de Fourier. A Série de Fourier pode ser estendida para sinais não periódicos, como os obtidos pelo sensor classificador de insetos, a partir da Transformada de Fourier.

O cálculo da transformada de um sinal exige o uso de um número infinito de amostras no domínio do tempo e, consequentemente, um número infinito de pontos no domínio da frequência. Como isso é impossível de ser computado, utiliza-se um número finito de pontos no domínio do tempo e define-se uma representação discreta do sinal no domínio da frequência a partir da Transformada Discreta de Fourier (TDF). O método mais utilizado para se calcular a TDF é a Transformada Rápida de Fourier (Cooley e Tukey, 1965). A aplicação desse algoritmo sobre o sinal da Figura 5.4 gera o gráfico de espectro de frequências exibido na Figura 5.6.

O espectro de frequências possui informações importantes que caracterizam um sinal de áudio. No domínio de música, a altura (ou *pitch*), representada pela frequência fundamental, denota a nota musical proferida. Na classificação de insetos alados, tanto por sensores ópticos quanto acústicos, essa característica representa a frequência das batidas de asas do inseto. É possível observar na Figura 5.6 que a frequência de batidas de asas do inseto analisado é estimada em 646Hz a partir da representação espectral. Além disso, a localização e magnitude das harmônicas, chamadas de formantes, também constituem importantes características.

Outra maneira de representar os dados obtidos pelo sensor é a partir da representação cepstral. O cepstro é o resultado da aplicação da Transformada de Fourier sobre o espectro

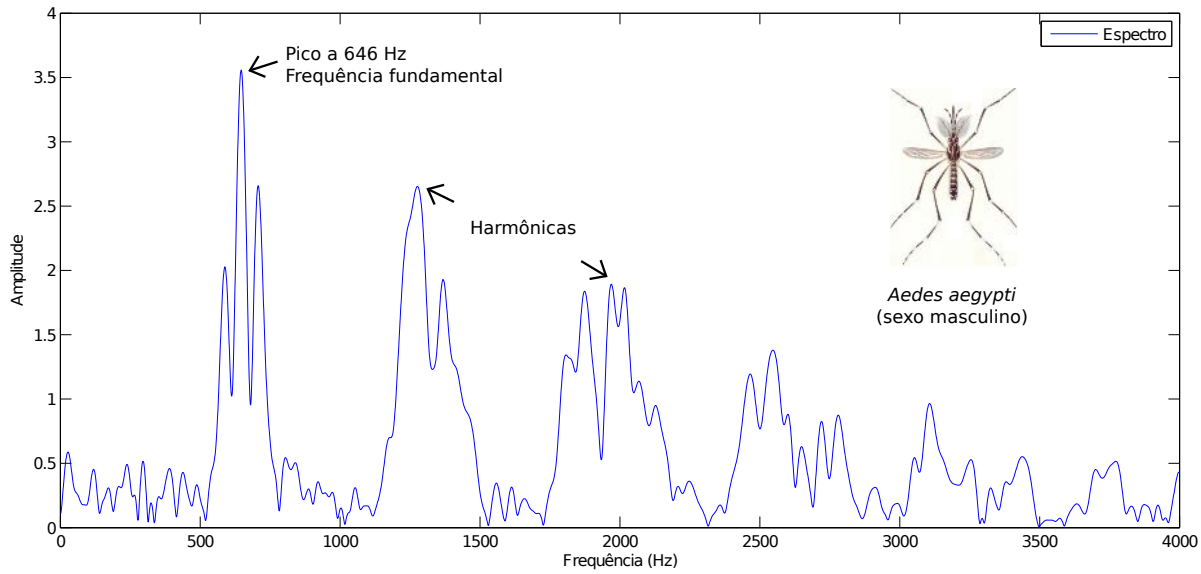


Figura 5.6: Espectro de frequências dada a passagem de um inseto da espécie *Aedes aegypti* pelo sensor.

de um sinal na forma logarítmica. A variável independente do domínio cepstral é denominada de quefrência e, apesar de ser uma medida de tempo, não possui relação direta com a representação temporal do sinal, mas sim ao período, o qual representa o inverso da frequência ($1/f$). Originalmente, o cepstro foi proposto para analisar ecos sísmicos de terremotos e bombas (Bogert et al., 1963). Entretanto, as características cepstrais são também utilizadas na análise de áudio em tarefas como reconhecimento de locutor (Assaleh e Mammone, 1994), reconhecimento de fala (Hur e Kim, 2001), classificação de gênero musical (Lee et al., 2009), reconhecimento de instrumentos musicais (Eronen e Klapuri, 2000), entre outras tarefas.

Na Figura 5.7 é exibido o cepstro referente ao sinal que deu origem à Figura 5.6. Conforme pode ser observado, a frequência fundamental também pode ser estimada a partir da representação cepstral ao analisar a observação de maior amplitude. No exemplo, a maior amplitude se encontra no período de 0,001563s, resultando em uma frequência fundamental estimada em 639,79Hz a partir da análise cepstral.

5.6 Extração de Características

Conforme anteriormente discutido, um dos principais atributos que pode ser extraído dos sinais obtidos pelo sensor é a frequência de batida de asas do inseto responsável por gerar o sinal. Entretanto, somente este atributo não é suficiente para caracterizar as espécies de insetos. Considerando somente a ordem de insetos *Diptera*, estima-se a existência de mais de 240.000 espécies, sendo catalogadas cerca de 120.000 diferentes

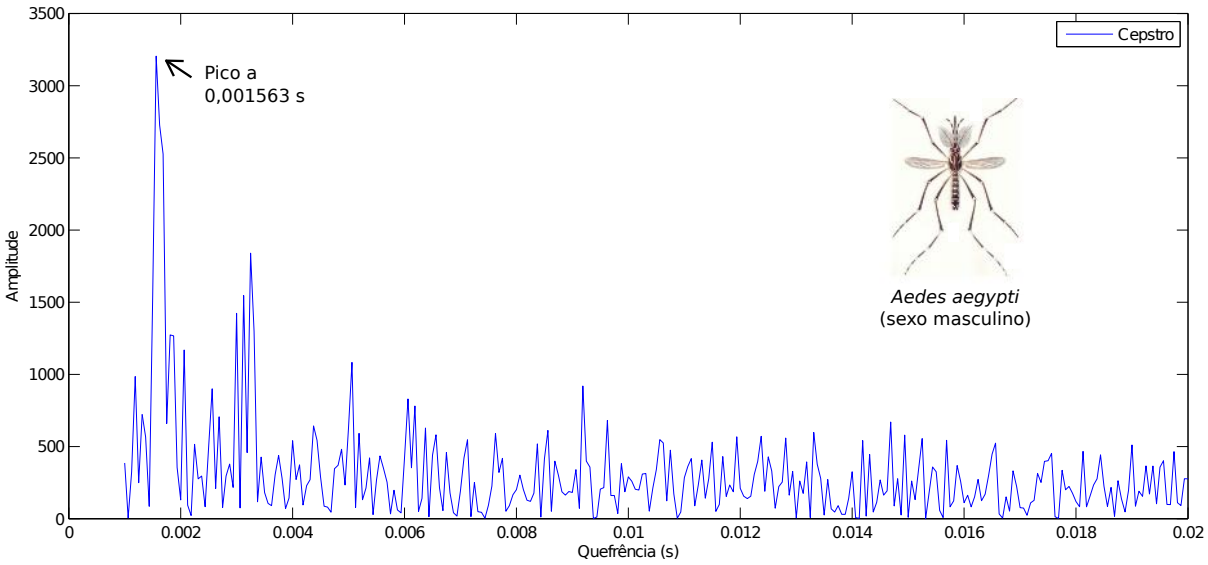


Figura 5.7: Cepstro obtido a partir do sinal coletado pelo sensor dada a passagem de um inseto da espécie *Aedes aegypti*.

espécies (Wiegmann e Yeates, 1996). Se considerarmos que a frequência de batida de asas dos insetos estão tipicamente na faixa de frequência entre 100Hz e 1000Hz, é evidente que a maior parte das espécies irá compartilhar um mesmo valor de frequência de batida de asas. Na matemática, este fenômeno é chamado de *Princípio da casa dos pombos* (Ajtai, 1988): "*se n pombos devem ser postos em m casas, e se $n > m$, então pelo menos uma casa irá conter mais de um pombo*". Para ilustrar este problema, considere o histograma apresentado na Figura 5.8 obtido a partir dos valores de frequência de batida de asas gerados por mosquitos das espécies *Culex stigmatosoma* (fêmea), *Aedes aegypti* (fêmea) e *Culex tarsalis* (macho) (Chen et al., 2014).

A partir da observação da Figura 5.8, é visualmente óbvio que é simples separar a espécie *Cx. stigmatosoma* ♀ da espécie *Cx. tarsalis* ♂ a partir dos dados de frequência de batida de asas, dada a separação entre as distribuições dos dados destas espécies. Entretanto, separar a espécie *Cx. stigmatosoma* ♀ da espécie *Ae. aegypti* ♀ é bastante difícil devido a sobreposição da distribuição dos dados. Este problema se torna maior quando se considera um número maior de espécies. Por este motivo, para a correta classificação de espécies de insetos por algoritmos de Aprendizado de Máquina, é necessário extrair outras características do sinal que sejam capazes de discriminar diferentes espécies. Nesta direção, os atributos explorados neste trabalho e provenientes de diferentes representações do sinal (Silva et al., 2015b) serão apresentados nas próximas subseções. Nas definições apresentadas a seguir, considere um sinal x de tamanho n, onde cada observação é representada por x_i com $1 \leq i \leq n$, e Y refere-se ao espectro de frequência do sinal x composto por N frequências diferentes.

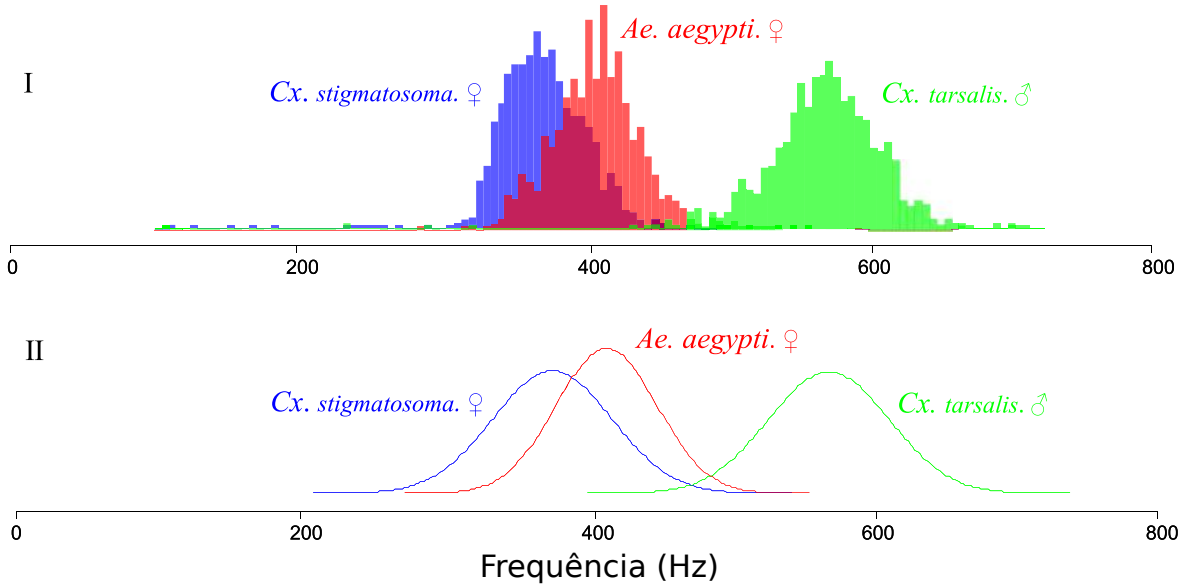


Figura 5.8: *I*) Histograma de frequência de batida de asas referente as espécies *Culex stigmatosoma* (fêmea), *Aedes aegypti* (fêmea) e *Culex tarsalis* (macho) (Chen et al., 2014). Os histogramas foram gerados a partir de mil observações de cada espécie; *II*) Distribuições gaussianas que aproximam os histogramas.

5.6.1 Atributos Extraídos da Representação Temporal

Atributos contidos na representação temporal dos sinais podem ser importantes para a compreensão do comportamento da série no tempo. Algumas medidas simples são a variância e o desvio padrão. Provavelmente, o atributo mais simples que pode ser extraído da representação temporal é a duração do sinal. Essa informação pode ser importante em diversos tipos de sinais de áudio. Por exemplo, na tarefa de classificação por som de tipos de pratos de bateria e o movimento realizado pelo baterista ao tocar o instrumento, a duração de um sinal é essencial para distinguir um prato do tipo *Splash* onde se realiza o "abafamento" do prato após tocá-lo de outro que é atacado normalmente (Souza et al., 2015a). A duração do sinal, em segundos, é calculada pela Equação 5.1.

$$Duração = \frac{n}{taxa\ amostragem} \tag{5.1}$$

Alguns atributos temporais são aproximações de características primitivas que descrevem uma onda sonora, como a amplitude e o período. Por exemplo, dado um sinal x de tamanho n, a amplitude pode ser estimada pela magnitude média do sinal, descrita pela Equação 5.2.

$$Mag_{\mu} = \frac{1}{n}\sum_{i=1}^{n}|x_i| \tag{5.2}$$

O valor quadrático médio (*root mean square* – RMS), também é utilizado como medida de magnitude do sinal. O RMS é definido pela Equação 5.3.

$$RMS = \sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2} \tag{5.3}$$

Uma medida alternativa, ainda relacionada à amplitude, é obtida pelo cálculo do RMS, sem a raiz quadrada. Essa medida, conhecida como *short-time energy* (STE), é descrita pela Equação 5.4.

$$STE = \frac{1}{n}\sum_{i=1}^{n} x_i^2 \tag{5.4}$$

A amplitude do sinal também pode ser estimada a partir de uma medida denominada intervalo, conforme definido na Equação 5.5.

$$Intervalo = \max_{1\leq i\leq n}(x_i) - \min_{1\leq i\leq n}(x_i) \tag{5.5}$$

A concentração da energia do sinal no tempo também pode fornecer informações importantes. O centroide, definido pela Equação 5.6, define o centro de massa do sinal, fornecendo um valor que estima a distribuição da energia no tempo.

$$Centroide = \frac{\sum_{i=1}^{n} i x_i}{\sum_{i=1}^{n} x_i} \tag{5.6}$$

O período (ou o comprimento de onda, distância entre dois vales) pode ser associado à taxa de cruzamento em zero, em inglês *zero-crossing rate* (ZCR). Essa medida, definida pela Equação 5.7, também pode dar estimativas do nível de ruído do sinal. Quanto maior a ZCR, maior o nível de ruído.

$$ZCR = \frac{1}{n-1}\sum_{i=2}^{n} |S(x_i) - S(x_{i-1})| \tag{5.7}$$

$$S(x_i) = \begin{cases} 1, \text{ se } x_i \geq 0 \\ 0, \text{ caso contrário} \end{cases}$$

A medida ZCR também pode ser utilizada como uma estimativa da complexidade do sinal. Outra maneira de estimar a complexidade é a partir de nível de variação no tempo. Intuitivamente, a complexidade estimada deve possuir maior valor quando há muitos picos e vales no sinal. Na Equação 5.8, é definida uma medida de complexidade estimada.

$$CE = \sqrt{\sum_{i=1}^{n-1}(x_i - x_{i+1})^2} \tag{5.8}$$

Por fim, as medidas de obliquidade, Equação 5.9, e curtose, Equação 5.10, podem ser úteis para estimar o formato do sinal. A obliquidade é uma medida de simetria, sendo que valores próximos a 0 indicam simetria, valores negativos indicam maior concentração no lado direito e valores positivos indicam maior concentração no lado esquerdo do sinal. A curtose mede o achatamento da distribuição em relação ao pico de uma distribuição normal. Valores pequenos de curtose indicam uma distribuição mais achatada. Nas Equações 5.9 e 5.10, *Mag* e *DP* referem-se a magnitude e ao desvio padrão do sinal.

$$Obliquidade = \frac{\sum_{i=1}^{n-1}(|x_i| - Mag_\mu)^3}{(n-1)DP^3} \tag{5.9}$$

$$Curtose = \frac{\sum_{i=1}^{n-1}(|x_i| - Mag_\mu)^4}{(n-1)DP^4} \tag{5.10}$$

5.6.2 Atributos Extraídos da Representação Espectral

A partir do espectro do sinal é possível extrair informações importantes para a caracterização de um sinal. No domínio de música, a altura (ou *pitch*), representada pela frequência fundamental, caracteriza a nota musical emitida. Conforme anteriormente discutido, nos dados obtidos pelo sensor laser identificador de insetos, essa característica representa a frequência de batida de asas dos insetos.

A frequência fundamental (F_0) do sinal pode ser obtida de maneira simples, apenas encontrando-se a magnitude máxima no espectro do sinal. Essa abordagem é formalmente definida na Equação 5.11, em que Y refere-se ao espectro do sinal.

$$F_0 = arg\max_i(Y_i) \tag{5.11}$$

Entretanto, algumas perturbações periódicas podem resultar em picos no espectro de frequência que não possuem significado interessante ao sinal. Por exemplo, no contexto de sinais captados opticamente como os discutidos neste trabalho, a presença de uma lâmpada fluorescente no ambiente pode causar um pico na frequência em que a lâmpada pisca. É comum que lâmpadas desse tipo tenham uma frequência de 60Hz, causando variações nas amplitudes nessa frequência e múltiplas dela.

O espectro de um sinal é composto por uma componente principal, na frequência fundamental e componentes harmônicas de menores magnitudes com frequências múltiplas à fundamental. Em um sinal periódico, composto por ondas senoidais, essa afirmação é facilmente verificada. Por outro lado, em sinais complexos essa análise não é tão simples. A primeira dificuldade é o fato de que a frequência fundamental exata é difícil de ser calculada. Outra dificuldade está relacionada ao fato de que pequenas perturbações no sinal podem causar deslocamentos no posicionamento das componentes harmônicas. Para estimar as posições corretas das componentes harmônicas é necessário realizar uma etapa

de análise de harmônicas. Na Figura 5.9 é ilustrada uma abordagem simplificada desta operação, proposta por Park (2004). Em resumo, busca-se os picos de magnitude em frequências próximas às harmônicas teóricas. Os valores encontrados são chamados de harmônicas estimadas.

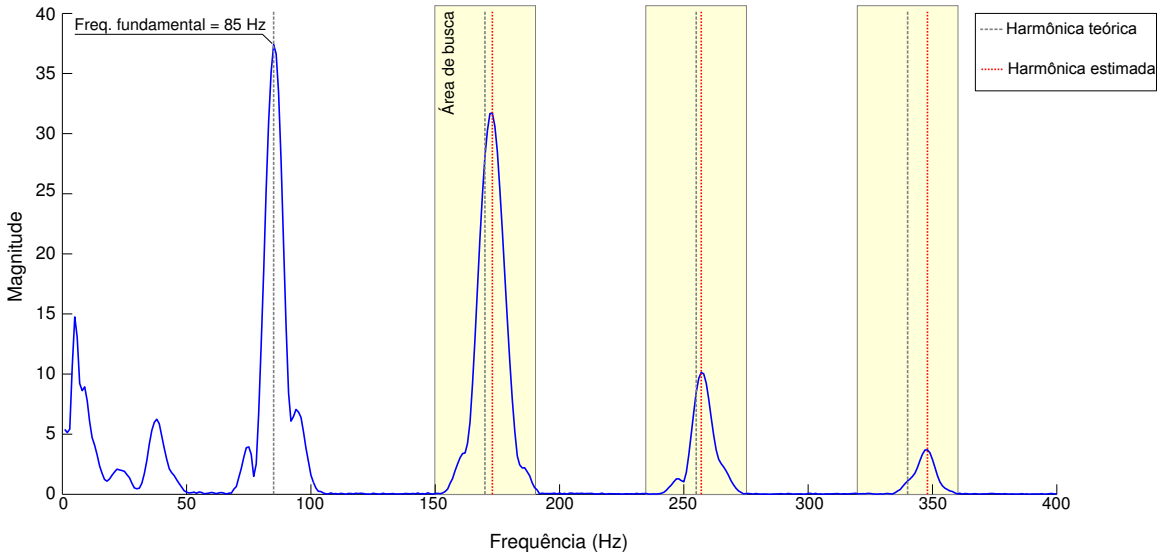


Figura 5.9: Exemplo do procedimento de análise de harmônicas. No exemplo, as frequências relativas às harmônicas teóricas são 170Hz, 255Hz e 340Hz, enquanto as frequências referentes às harmônicas estimadas são 173Hz, 257Hz e 348Hz (Silva, 2014).

A partir da análise de harmônicas, é possível extrair atributos que podem ser utilizados para a caracterização de sinais. Na Equação 5.12, é definida a característica inarmonia, que mede a diferença média das distâncias entre as harmônicas teóricas e as estimadas. Na equação, F_i é a posição estimada para a i-ésima harmônica e N_{harm} é o número de componentes analisadas.

$$Inarm = \sum_{i=1}^{N_{harm}} \frac{|F_i - iF_0|}{iF_0} \tag{5.12}$$

Outros atributos que podem ser calculados a partir da análise de harmônicas são os *tristimulus*. Esses atributos são equivalentes aos atributos de cores utilizados pela visão humana (Pollard e Jansson, 1982). A ideia básica de *tristimulus* na visão humana é que qualquer cor pode ser obtida pela combinação das cores primárias. Em relação à audição humana, qualquer som pode ser distinguido por características relacionadas à frequência fundamental e componentes harmônicas. Essas características são definidas pelas Equações 5.13, 5.14 e 5.15, em que H_k se refere à magnitude da k-ésima harmônica, sendo a de número 0 a frequência fundamental.

$$ts1 = \frac{H_0}{\sum_{k=0}^{N_{harm}} H_k} \tag{5.13}$$

$$ts2 = \frac{\sum_{k=1}^{3} H_k - 1}{\sum_{k=0}^{N_{harm}} H_k} \tag{5.14}$$

$$ts3 = \frac{\sum_{k=4}^{N_{harm}} H_k}{\sum_{k=0}^{N_{harm}} H_k} \tag{5.15}$$

Além da análise de harmônicas, explorar a variação do espectro ao longo das frequências também pode proporcionar informações úteis para a análise de sinais de áudio. Por exemplo, a irregularidade espectral (Krimphoff et al., 1994) revela a variabilidade das frequências vizinhas no espectro, analisando a diferença entre as magnitudes das frequências corrente, anterior e próxima à que se está analisando. Formalmente, a irregularidade espectral é definida pela Equação 5.16.

$$IE = \sum_{i=2}^{N-1} \left(20\log_{10}(Y_i) \frac{20\log_{10}(Y_{i-1}) + 20\log_{10}(Y_i) + 20\log_{10}(Y_{i+1})}{3} \right) \tag{5.16}$$

Outra medida para estimar a variação do espectro é o fluxo espectral. Nesta medida é estimado quão rápido o valor de magnitude das componentes de frequência varia. O fluxo espectral é definido pela Equação 5.17. O valor de q pode ser qualquer valor inteiro. Comumente é utilizado o valor 2 (Wang et al., 2012).

$$Fluxo = [\sum_{i=1}^{N-1} (|Y_i - Y_{i+1}|)^q]^{1/q} \tag{5.17}$$

Outras características espectrais do sinal podem ser obtidas por equações semelhantes às utilizadas para se extrair atributos da representação temporal. Por exemplo, o centroide é uma medida comumente calculada no espectro. Essa medida representa o centro de massa em relação à concentração de energia das componentes de frequência do sinal. Intuitivamente, se há componentes com grandes magnitudes em frequências de baixo valor, o centroide deverá possuir um valor pequeno. Caso a energia do sinal não se concentre em valores de baixa frequência, o valor do centroide será alto. O centroide espectral é definido pela Equação 5.18.

$$CE = \frac{\sum_{i=1}^{N} iY_i}{\sum_{i=1}^{N} Y_i} \tag{5.18}$$

A equação apresentada é muito semelhante à equação que define o centroide temporal. De fato, isso acontece com diversos atributos. Todas as medidas estatísticas e relacionadas à forma utilizadas na representação temporal, podem ser facilmente adaptadas para serem utilizadas na representação de frequências. A variância, desvio padrão, curtose, obliqui-

dade, magnitude média e energia são calculadas de mesmo modo, apenas substituindo a série temporal x pelo espectro de frequência Y.

Assim como o centroide, há outra medida relacionada à concentração das magnitudes nas faixas de frequência. Essa característica, intitulada *Roll-off*, é definida pela Equação 5.19. O valor dessa medida determina a frequência em que a energia do espectro alcança 85% do seu total. Assim, quanto mais a energia se concentra nas baixas frequências, menor o valor do *roll-off*.

$$RollOff = R, \text{ tal que } \sum_{i=1}^{R} Y_i = 0,85 \sum_{i=1}^{N} Y_i \tag{5.19}$$

A forma do espectro também pode ser estimada por uma medida de achatamento, conforme descrito na Equação 5.20. Essa medida é definida pela razão da média geométrica (Mg) pela média aritmética (Ma) dos valores de magnitude do espectro. Valores próximos de 0 indicam sinais aproximadamente periódicos, senoidais.

$$Ach = 10 \log_{10} \left(\frac{Mg}{Ma} \right) = 10 \log_{10} \left(\frac{(\prod_{i=1}^{N} Y_i)^{\frac{1}{N}}}{\frac{1}{N} \sum_{i=1}^{N} Y_i} \right) \tag{5.20}$$

5.6.3 Atributos Extraídos da Representação Cepstral

Além de ser utilizada para estimar a frequência fundamental, conforme discutido na Seção 5.5 (especificamente na Figura 5.7), a partir da representação cepstral também é possível extrair importantes características do sinal denominados de coeficientes cepstrais que podem ser extraídos de diferentes escalas do cepstro. Para melhor aproveitamento desses coeficientes, é comum a utilização de uma escala acusticamente definida, criada a partir de um estudo realizado por Stevens et al. (1937), que relacionou as frequências físicas com as frequências percebidas pelo ouvido humano. Essa escala, intitulada mel, é utilizada para se extrair os chamados coeficientes mel-cepstrais (em inglês *Mel-Frequency Cepstral Coefficients* – MFCC). Para calcular tais coeficientes, tomam-se as magnitudes das componentes de frequência utilizando a escala mel e aplica-se a Transformada Discreta do Cosseno – transformada que utiliza apenas ondas cossenoidais como componentes, muito utilizada para compressão de dados – sobre o logaritmo desses valores. Os MFCC são as amplitudes do cepstro resultante dessa operação. A Equação 5.21 define a escala para a conversão de frequência (f) em mel (m).

$$m = 2595 log_{10} \left(1 + \frac{f}{700} \right) \tag{5.21}$$

Em algumas aplicações, a suposição de que uma escala baseada no sistema auditivo humano é, de fato, a mais adequada pode ser errônea. Por isso, outras escalas podem ser utilizadas. Um exemplo é a escala logarítmica, onde calculam-se os coeficientes denomi-

nados *Log-Linear-Frequency Cepstrum Coefficients* (LLFC). Também é possível realizar a mesma operação sem utilizar qualquer transformação de escala, neste caso, tem-se um conjunto de coeficientes denominados *Linear-Frequency Cepstrum Coefficients* (LFC).

5.6.4 Atributos de Predição Linear

Na predição linear, um sinal x é representado por uma combinação linear de valores anteriormente observados. Assim, um sinal pode ser descrito de acordo com a Equação 5.22, em que k é o índice temporal e L é o número de coeficientes de predição linear (em inglês, *Linear Predictive Coding Coefficients* – LPC) a ser utilizado. Os coeficientes a_l são computados de modo a minimizar o erro de predição $E_k = \hat{x}_k - x_k$, por meio de um método de covariância ou de auto-correlação, em que $\hat{x}_k$ é o valor predito do sinal k_k é o valor observado.

$$\hat{x}_k = \sum_{l=1}^{L} a_l x_{k-l} \tag{5.22}$$

A partir do cálculo dos coeficientes é possível, por exemplo, enviar somente os coeficientes a_l e o erro de predição E_k em uma transmissão de dados, de modo que o volume de dados enviado é consideravelmente menor do que o sinal completo. Para isso, o sinal é comprimido por meio de um filtro de análise, utilizando-se uma função de transmissão que tenta suprimir frequências de maior magnitude. Para receber esse sinal, um filtro de recepção utiliza a função inversa da função de transmissão, voltando a amplificar as frequências suprimidas (Benesty et al., 2008).

A Equação 5.22 pode ser reescrita no domínio de frequência por meio de uma transformada-z (Oppenheim et al., 1989). Desse modo, um curto segmento de sinal se presume ser gerado como a saída de um filtro $H(z) = 1/A(z)$, em que $A(z)$ é o inverso do filtro, conforme exibido na Equação 5.23.

$$H(z) = \frac{1}{A(z)} = \frac{1}{1 - \sum_{i=1}^{p} a_i z^{-i}} \tag{5.23}$$

As *Line Spectral Frequencies* (LSF), introduzidas por Itakura (1975), são um modo alternativo de representar os LPC. Para calcular esses coeficientes, o filtro polinomial inverso é decomposto em dois polinômios $P(z)$ e $Q(z)$. Esses polinômios são descritos na Equação 5.24, em que $P(z)$ é um polinômio simétrico e $Q(z)$ é um polinômio antissimétrico. As raízes de $P(z)$ e $Q(z)$ determinam os coeficientes LSF.

$$\begin{aligned} P(z) &= A(z) + z^{p+1} A(z^{-1}) \\ Q(z) &= A(z) - z^{p+1} A(z^{-1}) \end{aligned} \tag{5.24}$$

As LSF formam um conjunto de características adequadas para as tarefas de quantização e interpolação (Paliwal e Kleijn, 1995). Assim, podem representar o sinal analisado mapeando-o em um pequeno número de coeficientes, e, portanto, consiste de uma representação mais adequada às demais como LPC.

5.7 Procedimentos de Coleta de Dados Estacionários

Para avaliar o poder discriminativo dos diferentes atributos apresentados anteriormente em conjunto com algoritmos de Aprendizado de Máquina na classificação de insetos a partir de dados obtidos pelo sensor laser, preliminarmente foi necessário realizar a tarefa de coleta de dados para conduzir a etapa de avaliação. Assim, nesta seção serão apresentados os procedimentos para a coleta de dados em laboratório e a etapa de pré-processamento dos dados. É importante destacar que neste capítulo será considerado a ausência de mudanças de conceito ao longo do tempo. Fatores externos como temperatura, umidade e pressão atmosférica, responsáveis por alterar o comportamento dos insetos e por gerar mudanças de conceito nos dados, serão considerados posteriormente no Capítulo 6.

O único modo de se obter exemplos rotulados com total confiança a partir dos dados obtidos pelo sensor é depositar uma quantidade de insetos de mesma espécie em uma caixa fechada (insetário) adaptada com o sensor e realizar a coleta separadamente para cada espécie de inseto. Exemplares de insetários são apresentados na Figura 5.10.

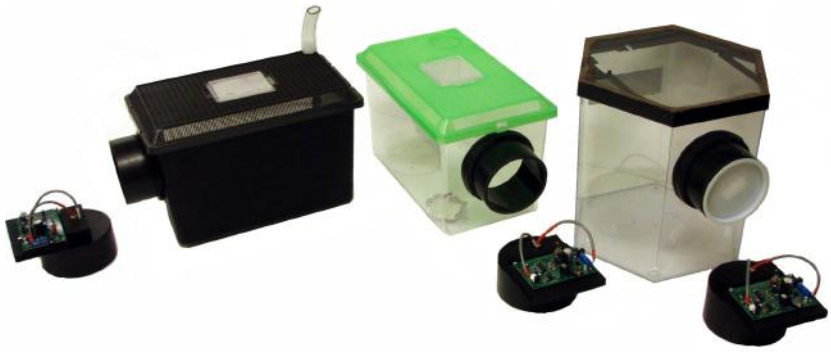


Figura 5.10: Três exemplos de insetários adaptados para coletar dados com o sensor. Ao centro, o insetário padrão para experimentos (Souza et al., 2013a).

Para a coleta dos dados foram utilizadas cinco insetários distintos, sendo que cada um possuía uma única espécie de inseto em seu interior. Assim, foram contempladas cinco diferentes espécies: as moscas *Drosophila melanogaster* e *Musca domestica* e os mosquitos *Culex quinquefasciatus, Culex tarsalis* e *Aedes aegypti*, principais vetores de doenças como filariose, febre do Nilo Ocidental, dengue e febre amarela. Para a simulação de um cenário real em que se obtém um fluxo de dados no qual a sequência de passagens ao longo do

tempo possui todas as cinco possíveis espécies, foram extraídas paralelamente dos cinco insetários as passagens das gravações realizadas a partir de uma mesma data, horário e condições ambientais. As gravações foram realizadas a partir de 7 de novembro de 2011 e possuem duração de aproximadamente 6 dias consecutivos sem interrupções. Condições ambientais como temperatura e umidade do ar foram registradas somente no primeiro e terceiro dia de gravações. Estas condições foram controladas em laboratório para que houvesse pouca variação e não causasse alterações nos dados medidos. A temperatura teve variações entre 20°C e 22°C e a umidade do ar entre 25% e 35%.

Após a gravação por 6 dias consecutivos realizada em cada insetário, foi necessário extrair somente os segmentos de áudios relativos às passagens de insetos pelo laser e associar uma classe a cada passagem. Para isso, aplica-se inicialmente um detector de passagem. O detector utiliza uma janela deslizante sobre os dados e calcula as magnitudes dos componentes do sinal dentro da janela. Então, é utilizada a magnitude máxima dentro de uma faixa de frequências entre 100Hz e 1000Hz (faixa de valores de frequências típicas de batida de asas de insetos) como um valor de confiança para o detector. Dessa maneira, quanto maior for a magnitude, maior a confiança de que o sinal não é um ruído de fundo. Todos os sinais com magnitude acima de um limiar especificado pelo usuário são considerados um evento gerado por um inseto. O funcionamento do detector é ilustrado na Figura 5.11.

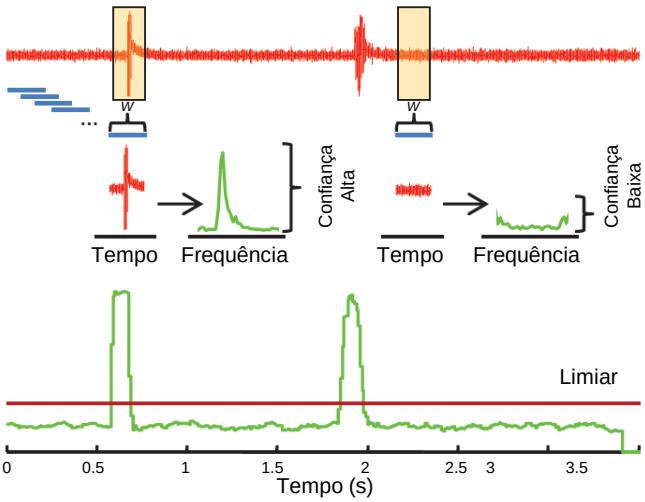


Figura 5.11: Esquema de funcionamento do detector de eventos (Batista et al., 2011a).

Observa-se na ilustração da Figura 5.11 que o detector utiliza uma janela deslizante (W) que corre sobre o sinal capturado pelo sensor. A cada passo, o sinal dentro da janela W é convertido do domínio temporal para o domínio das frequências por meio da TDF e extrai-se a magnitude máxima dos dados da janela. Definido o limiar, é possível estimar o nível de confiança de determinada janela conter a passagem de um inseto.

Durante a identificação de cada evento no áudio gerado pelo sensor, foi associada à passagem a classe correspondente a espécie do inseto que gerou o evento e uma marcação de tempo (*timestamp*) que permitiu sincronizar todas as passagens de modo que a sequência de eventos corresponda às passagens das cinco diferentes espécies como se fossem realizadas em um mesmo sensor ao longo de 6 dias. Desse modo, a partir desta configuração é possível simular a aplicação real de fluxo de dados em um ambiente controlado e obter os rótulos corretos de todas as passagens identificadas pelo sensor para fins de avaliação.

No total foram obtidas 5.325 observações que constituem cinco classes. A distribuição das classes é apresentada na Tabela 5.1.

Tabela 5.1: Distribuição das classes que constituem o conjunto de dados de insetos sem mudanças de conceito.

Classe	**Espécie do inseto**	**Quantidade de exemplos (%)**
Flies	*Musca domestica*	917 (17,22%)
Quinx	*Culex quinquefasciatus*	1285 (24,13%)
Tarsalis	*Culex tarsalis*	1265 (23,76%)
Fruit	*Drosophila melanogaster*	954 (17,91%)
Aedes	*Aedes aegypti*	904 (16,98%)
Total		**5325 (100%)**

Estes dados foram posteriormente utilizados em diferentes trabalhos para a avaliação de algoritmos, estratégias de classificação e atributos descritores dos dados, como Souza et al. (2016), Silva et al. (2015b), Qi et al. (2015), Silva (2014), Souza et al. (2013b), Souza et al. (2013a) e Silva et al. (2013c). A seguir, são apresentados os principais resultados de classificação e conclusões.

5.8 Classificação com Múltiplas Classes

A partir do conjunto de dados coletado com a presença de 5 diferentes espécies de insetos, foi possível avaliar o poder discriminativo dos atributos apresentados na Seção 5.6 com algoritmos de Aprendizado de Máquina. Especificamente, os atributos foram divididos em 7 categorias principais na avaliação experimental:

- **MFCC.** Foram extraídos 40 coeficientes utilizando a técnica *Mel Frequency Cepstral Coefficients*. Embora na literatura sejam tradicionalmente utilizados somente 13 coeficientes em diferentes tarefas de classificação de sinais de áudio, a partir de uma avaliação com a variação do número de coeficientes, foi observado que 40 atributos é um valor mais adequado para a classificação de insetos (Souza et al., 2013a,b);

- **LFC.** Foram extraídos 95 coeficientes utilizando a técnica *Linear Frequency Cepstral Coefficients*;

- **LLFC.** Foram extraídos 70 coeficientes utilizando a técnica *Log Linear Frequency Cepstral Coefficients*;
- **LSF.** Foram extraídos 100 coeficientes utilizando a técnica *Line Spectral Frequencies*;
- **Temporais.** A partir da representação temporal do sinal, foram extraídos 13 atributos: duração, magnitude média, valor quadrático médio, *short-time energy*, intervalo, centroide, simetria, taxa de cruzamento em zero, complexidade estimada, curtose, obliquidade, variância e desvio padrão;
- **Espectrais.** A partir da representação espectral do sinal, foram extraídos 17 atributos: energia, centroide, variância, desvio padrão, simetria, curtose, média, frequência fundamental, inarmonia, *tristimulus 1*, *tristimulus 2*, *tristimulus 3*, irregularidade, irregularidade modificada, fluxo espectral, *roll-off* e achatamento;
- **Frequência de batida de asas.** Este atributo foi extraído tanto da representação espectral como cepstral do sinal. Entretanto, pode-se notar que a frequência de batida de asas obtida a partir da representação cepstral apresentou os melhores resultados. Assim, os resultados sobre este atributo são relativos apenas a extração no domínio cepstral.

A quantidade de coeficientes extraídos por cada uma das técnicas MFCC (40), LFC (95), LLFC (70) e LSF (100) foram obtidas de acordo com a avaliação experimental anteriormente conduzida em Silva et al. (2013c, 2015b) em experimentos de classificação em lote para a identificação de insetos.

Os algoritmos de classificação avaliados foram k-Vizinhos Mais Próximos com o parâmetro $k = 1$ (1NN) e Máquina de Vetores de Suporte (SVM) com o *kernel* RBF (*Radial Basis Function*) e $\gamma = 0,001$, devido a popularidade dos algoritmos e bons resultados obtidos em experimentos anteriores, nos quais foram considerados outros algoritmos como árvores de decisão, floresta aleatória, modelo de misturas Gaussianas, regressão logística e *bayesiano* (Souza et al., 2013a,b; Silva et al., 2013c, 2015b).

Como o conjunto construído possui um fluxo de dados que engloba 6 dias consecutivos de coleta (ou 144 horas), foram avaliadas 4 diferentes configurações para o treino/teste dos classificadores: *i*) dados provenientes das 12 primeiras horas de gravação para treino e os dados obtidos nas 132 horas restantes para o teste; *ii*) 24 horas para o treino e 120 horas para o teste; *iii*) 36 horas para o treino e 108 horas para o teste; e *iv*) 48 horas para o treino e 96 horas para o teste. Os resultados de acurácia obtidos pelos 2 algoritmos e 7 diferentes conjuntos de atributos de acordo com as 4 configurações para o treino/teste dos classificadores são apresentados na Tabela 5.2.

Na Tabela 5.2, o melhor resultado obtido por um algoritmo dada uma configuração de treino/teste é destacado em negrito. Pode-se notar que em todos os casos, os coeficientes

MFCC apresentaram os melhores resultados, sendo adequados para a tarefa de classificação automática de insetos por meio de sensores laser com resultados em torno de 85% a 90% de acurácia. Ainda, o algoritmo SVM apresentou resultados sensivelmente melhores que o algoritmo 1NN.

Tabela 5.2: Resultados obtidos por diferentes atributos descritores na tarefa de classificação de insetos. Foram avaliados os algoritmos de classificação 1-Vizinho Mais Próximo (1NN) e Máquina de Vetores de Suporte (SVM), treinados com dados referentes a 12, 24, 36 e 48 horas.

	12 horas		24 horas		36 horas		48 horas	
Atributo	1NN	SVM	1NN	SVM	1NN	SVM	1NN	SVM
MFCC	**84,92**	**86,91**	**86,96**	**90,33**	**85,93**	**88,05**	**84,99**	**88,62**
LFC	84,38	79,75	84,54	82,48	85,04	86,30	84,04	86,07
LLFC	73,78	40,14	76,82	38,97	76,31	46,05	79,68	43,72
LSF	81,46	74,47	81,80	80,29	81,16	80,49	80,86	81,24
Espectrais	68,18	63,07	71,76	56,46	72,13	57,06	73,16	58,35
Temporais	18,43	22,23	16,20	25,67	16,86	20,64	12,66	22,42
Freq. batida de asas	53,70	66,66	53,28	63,79	53,79	63,04	58,19	63,05

Dados os resultados apresentados na Tabela 5.2, para analisar se há diferenças estatisticamente significativas entre os algoritmos, foi realizado o teste de *Friedman* com o pós-teste de *Nemenyi*. O diagrama de diferença crítica é apresentado na Figura 5.12. Pode-se observar no diagrama que em primeiro lugar no *ranking* está o algoritmo SVM com os coeficientes MFCC, seguido do algoritmo 1NN, também com os coeficientes MFCC. Em seguida, os melhores resultados são apresentados pelos atributos LFC e LSF. Os piores resultados foram obtidos pelos atributos temporais. Entretanto, nota-se que para a maior parte dos conjuntos de atributos e algoritmos de classificação, não há diferenças estatisticamente significativas.

Como os melhores resultados foram obtidos pelo algoritmo de classificação SVM, é apresentado na Figura 5.13 o comportamento deste algoritmo ao longo do tempo, considerando os conjuntos de atributos MFCC, LFC e LSF. Estes resultados levam em consideração o classificador induzido com dados obtidos nas primeiras 24 horas de gravação e avaliado com dados referentes aos 5 dias restantes de gravação. Pode-se verificar uma queda na acurácia de todos os conjuntos de atributos nos dias 3 e 5 (aproximadamente). Essa queda se deve a alta taxa de desbalanceamento das classes nestes períodos. Ainda, na Figura 5.13 é possível observar a superioridade do algoritmo SVM em conjunto com os coeficientes MFCC ao longo do tempo, em relação ao uso de outros coeficientes.

Para a melhor visualização dos erros cometidos pelo classificador SVM com o conjunto de atributos MFCC treinado com dados referentes as primeiras 24 horas do fluxo de dados, a matriz de confusão obtida pelo classificador é apresentada na Tabela 5.3.

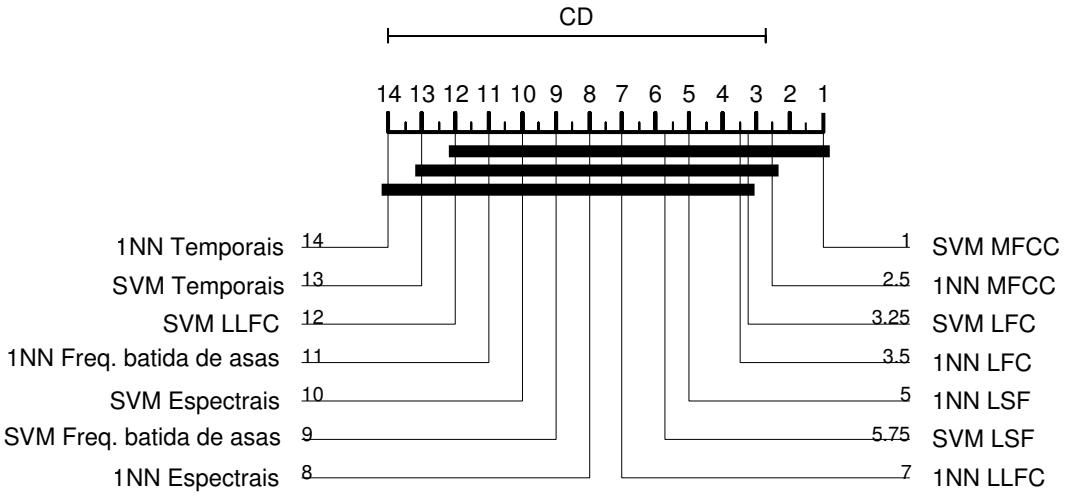


Figura 5.12: Diagrama de diferença crítica considerando as acurácias obtidas pelos classificadores 1NN e SVM com diferentes conjuntos de atributos.

Tabela 5.3: Erros cometidos pelo classificador SVM considerando um problema de classificação com múltiplas classes para a identificação de insetos.

Real	**Predito**				
	Aedes	***Flies***	***Fruit***	***Quinx***	***Tarsalis***
Aedes	662	22	27	42	9
Flies	1	634	22	0	13
Fruit	24	74	725	12	13
Quinx	19	5	9	860	26
Tarsalis	20	40	21	11	951
Acurácia	86,88	94,63	85,50	93,58	91,18

Na Tabela 5.3 também é possível observar a acurácia obtida em cada uma das cinco possíveis classes do problema. Nota-se que o classificador comete um número menor de erros na identificação das moscas da espécie *Musca domestica* (*Flies*) e um número maior de erros para a classificação das moscas *Drosophila melanogaster* (*Fruit*). De um modo geral, devido às semelhanças anatômicas e de frequência de batidas de asas, nota-se que os erros do classificador se concentram em maior proporção entre as três espécies de mosquitos e as duas espécies de moscas.

5.9 Classificação com uma Classe

Conforme anteriormente discutido na Seção 5.4, um dos possíveis usos do sensor identificador de insetos é em conjunto com armadilhas inteligentes para a captura seletiva de espécies de insetos de interesse. Devido à ocorrência frequente de epidemias de dengue em

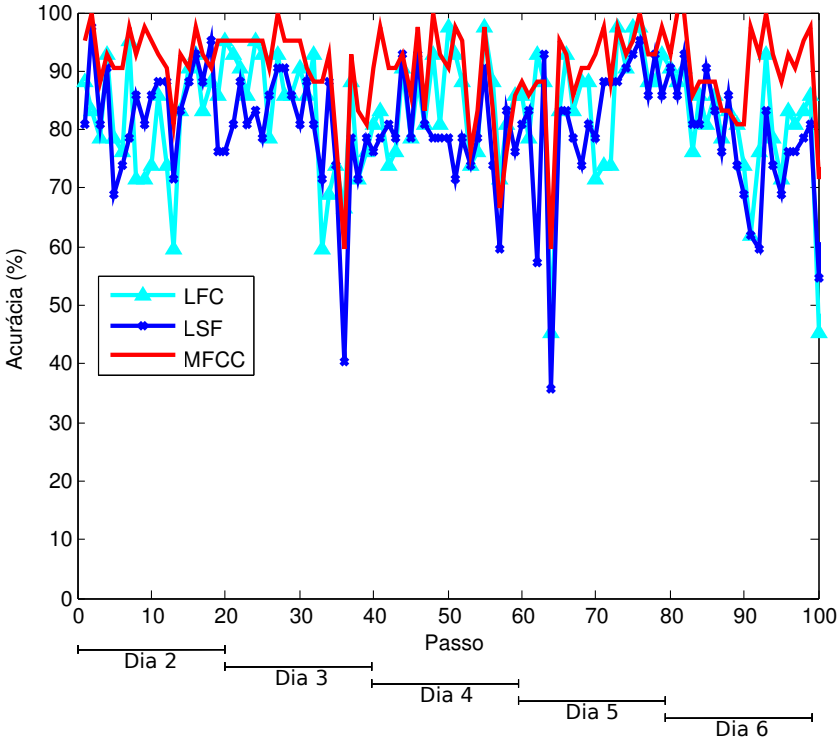


Figura 5.13: Resultados obtidos ao longo do tempo pelo classificador SVM treinado com 24 horas de dados e avaliado nos cinco dias restantes.

diversas regiões do Brasil e por ser o vetor de outras doenças graves como febre amarela, febre Zika e Chikungunya, este trabalho tem um interesse especial na captura de insetos da espécie *Aedes aegypti*. Assim, a armadilha pode ser uma importante ferramenta para a redução dos casos destas doenças ao atrair e capturar mosquitos adultos desta espécie. A captura de mosquitos adultos é importante no controle de epidemias, uma vez que reduz a população de mosquitos fêmeas possivelmente infectada, reduzindo portanto o risco de transmissões. Além disso, a armadilha também é responsável por fornecer informações sobre contagens que podem ser utilizadas para o cálculo de estimativas de densidade da população local dessa espécie de mosquito. Tais estimativas podem auxiliar na redução de custos e aplicação adequada de outros métodos de combate como a pulverização de inseticidas e larvicidas.

Algoritmos de classificação convencionais com múltiplas classes, tanto em lote como em fluxo de dados, permitem a associação de um exemplo desconhecido a uma de várias possíveis classes previamente definidas. Este procedimento pode ser um problema quando o exemplo a ser classificado não pertence a nenhuma das classes conhecidas. No caso da armadilha inteligente, é uma tarefa difícil ou até mesmo impossível o conhecimento de todas as possíveis espécies presentes no local onde a armadilha será utilizada. Para se ter uma dimensão do problema, estima-se que somente a ordem de insetos *Diptera*, possua mais de 240.000 espécies, sendo catalogadas cerca de 120.000 diferentes espécies (Wiegmann e Yeates, 1996). Desse modo, é impossível realizar uma coleta de dados tão

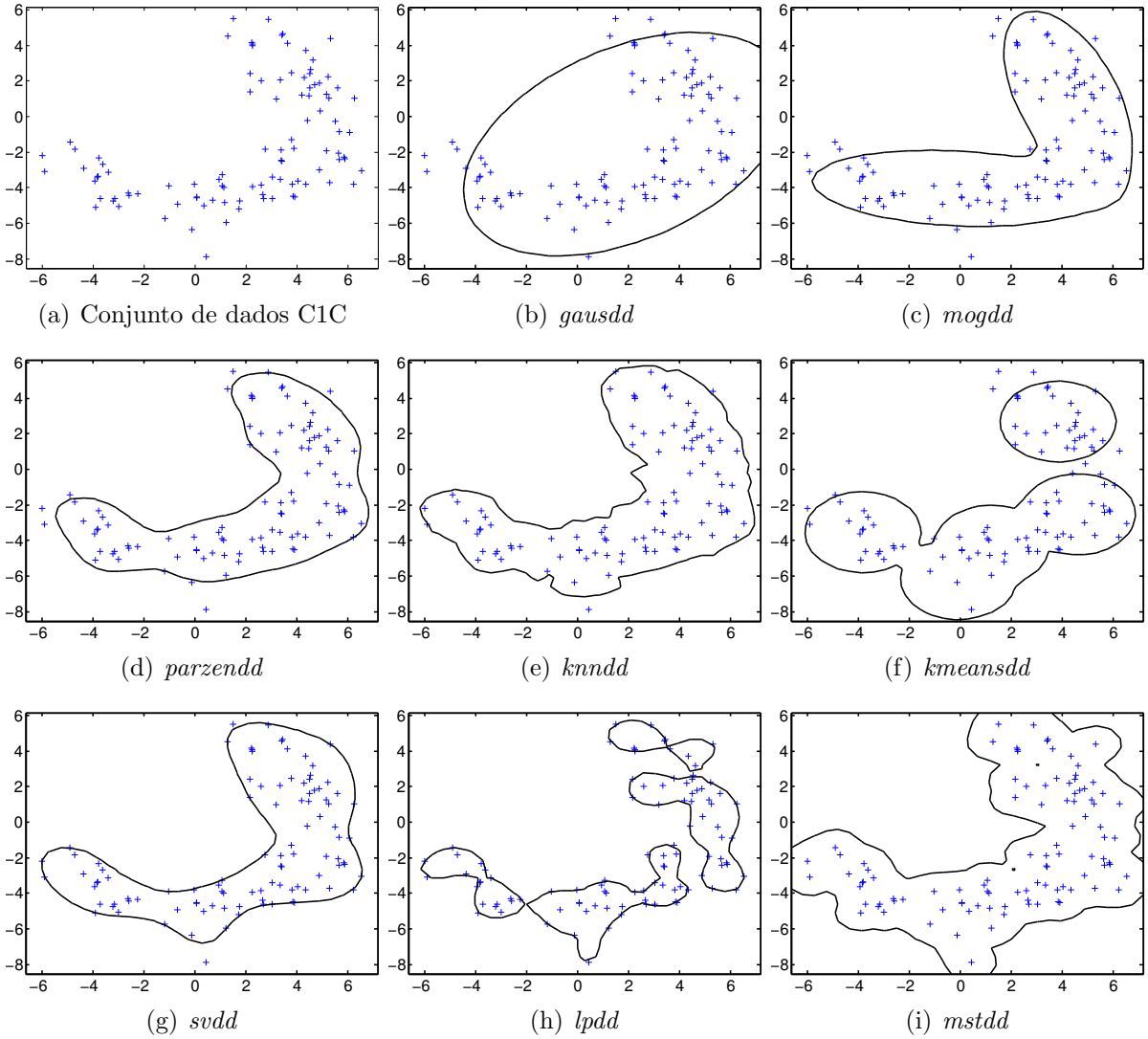
abrangente que contemple todas as possíveis espécies. No caso da captura e contagem pela armadilha de insetos somente da espécie *Aedes aegypti*, esta aplicação é um exemplo de um cenário em Aprendizado de Máquina denominado *classificação com uma classe* (C1C), em inglês *one-class classification* ou *single class classification* (Tax, 2001). Na classificação com uma classe, o aprendizado é realizado apenas com exemplos positivos (classe de interesse) e com nenhum ou poucos exemplos não rotulados que representam a classe negativa. Neste trabalho, será considerado na etapa de treinamento somente exemplos da classe positiva. Portanto, o uso de algoritmos de C1C é adequado para a armadilha na captura e estimativa de densidade populacional de mosquitos da espécie *Aedes aegypti*, de modo que todas as outras espécies sejam ignoradas pela armadilha.

Devido ao fato dos algoritmos de C1C realizarem a etapa de treinamento somente com dados de uma única classe sem a presença de contra exemplos, é um desafio destes algoritmos a definição do posicionamento das fronteiras de decisão. Isto torna o problema de classificação com uma classe mais díficil do que problemas convencionais com múltiplas classes previamente definidas ou binários. Ainda, é esperado que os algoritmos de C1C possuam um grande número de exemplos de treinamento (Tax, 2001), o que nem sempre é possível. Neste sentido, foi realizado neste trabalho uma avaliação experimental considerando diferentes algoritmos de C1C da literatura, com o objetivo de simular o uso do sensor em conjunto com armadilhas inteligentes e avaliar se os algoritmos existentes apresentam um desempenho adequado para o seu uso efetivo.

Nesta avaliação experimental, foi utilizado um pacote que conta com diversos algoritmos de C1C implementados em Matlab, denominado *Data Description toolbox* (DDtools) (Tax, 2015). Mais especificamente, foram avaliados os seguintes algoritmos do pacote: *gausdd (Gaussian target distribution), svdd (support vector data description), parzendd (Parzen density estimator data description), kmeansdd (k-means data description), knndd (k-nearest neighbor data description), lpdd (linear programming data description), mstdd (minimum spanning tree data description)* e *mogdd (mixture of Gaussians data description)*. Na Figura 5.14 é apresentado um exemplo de conjunto de dados artificial com duas dimensões composto somente pela classe positiva. Também pode-se observar nas ilustrações, a fronteira de separação atribuída por cada uma das técnicas avaliadas.

A descrição detalhada de cada uma destas técnicas foge do escopo desta tese e podem ser consultadas em Tax (2001). Basicamente, com dados de somente uma classe do problema, estas técnicas buscam definir uma fronteira de separação da classe positiva em relação a possíveis exemplos que diferem do padrão conhecido. Para isso, diferentes métodos podem ser utilizados como técnicas estatísticas, baseadas em distância, agrupamento ou baseadas em máquinas de vetores de suporte.

Entre as técnicas estatísticas, estão os métodos baseados em Gaussianas e janelas de Parzen, em que na etapa de treinamento realizam a estimação da densidade de probabilidade dos dados e na etapa de teste rejeitam qualquer exemplo que não se aproxime desta



(a) Conjunto de dados C1C (b) *gausdd* (c) *mogdd*

(d) *parzendd* (e) *knndd* (f) *kmeansdd*

(g) *svdd* (h) *lpdd* (i) *mstdd*

Figura 5.14: Exemplo de conjunto de dados artificial de C1C com duas dimensões e margem de separação atribuída por diferentes algoritmos avaliados.

estimativa. As técnicas se diferenciam pela maneira como estimam a distribuição dos dados de treinamento. No caso do uso de Gaussianas, a distribuição dos dados é aproximada com uma única Gaussiana (Bishop, 1995) ou com a mistura de k Gaussianas (Duda e Hart, 1973), o que permite uma modelagem mais flexível dos dados a partir da combinação linear de distribuições normais. A técnica de janelas de Parzen ou *Kernel Density Estimation* (KDE) (Duda e Hart, 1973) se baseia no uso de um histograma contínuo com hipercubos de largura h_n centrados em cada um dos exemplos. A partir do número de exemplos localizados no interior de cada hipercubo e de seu volume, é possível calcular a função densidade de probabilidade dos dados. O grau de generalização é determinado pela largura de h_n.

Na abordagem baseada em distância proposta por Tax e Duin (2000), não é necessário estimar a densidade local dos dados, sendo utilizado apenas a distância de cada exemplo

com o seu vizinho mais próximo. Assim, um exemplo de teste é classificado como positivo se a sua distância com o exemplo de treinamento mais próximo não for superior à distância deste mesmo exemplo de treinamento em relação a seu vizinho mais próximo.

Na abordagem baseada em agrupamento apresentada por Bishop (1995), os dados de treino são descritos por k grupos. No caso do algoritmo k-Means, um exemplo de teste é comparado com os centroides de cada grupo e rejeitado caso a distância seja superior a um limiar definido pelo usuário. Assim, além de definir limiar de distância mínima, também é necessário definir o número de grupos que será utilizado para a descrição dos dados.

Entre as abordagens baseadas em máquinas de vetores de suporte, pode-se citar os algoritmos *v-Support Vector Classifier* (v-SVC) (Schölkopf et al., 2001) e *Support Vector Data Description* (SVDD) (Tax e Duin, 2004). Em específico, neste trabalho é avaliado o algoritmo SVDD, implementado no pacote DDtools. O algoritmo SVDD descreve o domínio dos dados por meio de uma hiperesfera que contêm a maior parte dos exemplos da classe positiva. Esta hiperesfera é caracterizada por um centro e um raio. O volume da esfera é reduzido com a minimização do raio com o objetivo de alcançar uma melhor descrição dos dados.

Na avaliação experimental dos algoritmos de C1C, foi utilizado o mesmo conjunto de dados com 5 espécies de insetos anteriormente apresentado na Tabela 5.1. Dados os resultados obtidos com o uso dos atributos MFCC apresentados na Seção 5.8, estes coeficientes foram considerados adequados como atributos discriminativos dos dados na tarefa de classificação com uma classe.

Na etapa de treinamento dos classificadores, foi considerado somente dados da espécie *Aedes aegypti* (classe de interesse). Especificamente, avaliou-se o desempenho dos algoritmos induzidos com dados coletados durante as primeiras 48 horas do fluxo de dados (ou 347 exemplos). Na etapa de teste, foram considerados os exemplos remanescentes da classe de interesse que não foram utilizados na etapa de treinamento, bem como os dados das outras quatro espécies de insetos. Assim, na etapa de teste foram avaliados a classificação de 557 exemplos remanescentes da espécie *Aedes aegypti* e 4.978 exemplos provenientes das outras espécies de insetos.

Devido à proporção desbalanceada de exemplos da classe de interesse em relação a classe negativa na etapa de avaliação, um algoritmo responsável por predizer a classe negativa para todos os exemplos de teste alcançaria uma acurácia em torno de 90%. Por esta razão e como o objetivo principal da tarefa é a correta detecção de insetos da espécie *Aedes aegypti*, foram utilizadas outras medidas de avaliação além da acurácia de classificação para a avaliação dos algoritmos de C1C. Assim, dada a matriz de confusão apresentada na Tabela 5.4 que representa os erros e acertos de um algoritmo C1C considerando a tarefa avaliada neste trabalho, foram consideradas as medidas de avaliação

definidas nas Equações 5.25, 5.26 e 5.27, em que a espécie *Aedes aegypti* é a classe positiva do problema.

Tabela 5.4: Matriz de confusão de um problema de classificação binário.

Real	**Predito**	
	Ae. aegypti	**¬ *Ae. aegypti***
Ae. aegypti	Verdadeiro Positivo (TP)	Falso Negativo (FN)
¬ *Ae. aegypti*	Falso Positivo (FP)	Verdadeiro Negativo (TN)

$$Precisão(+) = \frac{TP}{TP + FP} \tag{5.25}$$

$$Cobertura(+) = \frac{TP}{TP + FN} \tag{5.26}$$

$$F - Measure(+) = \frac{2 \times (Precisão \times Cobertura)}{Precisão + Cobertura} \tag{5.27}$$

Além das medidas Precisão, Cobertura, F-Measure e Acurácia, também foi utilizada a medida AUC (*Area under Curve*), referente a área abaixo da curva ROC (*Receiver Operating Characteristic*). A curva ROC é uma representação gráfica bidimensional que corresponde a taxa de falsos positivos no eixo x e verdadeiros positivos no eixo y. Assim, no cenário ideal é esperado o valor mínimo de falsos positivos e o valor máximo de verdadeiros positivos, o que consequentemente acarreta um valor de AUC = 1 (Prati et al., 2011).

Os resultados obtidos pelos diferentes algoritmos de C1C considerando diferentes medidas de avaliação são apresentados na Tabela 5.5.

Tabela 5.5: Resultados de algoritmos C1C na tarefa de classificação de insetos *Aedes aegypti*.

Algoritmo	**Precisão**	**Cobertura**	**F-Measure**	**Acurácia**	**AUC**
gausdd	0,45	0,76	0,57	86,96	0,82
mogdd	0,64	0,64	0,64	91,98	0,80
parzendd	0,41	**0,91**	0,56	84,35	**0,87**
knndd	0,43	0,87	0,57	85,42	0,86
kmeansdd	0,41	0,78	0,54	85,05	0,82
svdd	**0,78**	0,73	**0,75**	**94,62**	0,85
lpdd	0,32	0,85	0,46	77,90	0,81
mstdd	0,32	0,89	0,48	78,04	0,83
Baseline	0,10	1,00	0,18	10,98	0,50

Na Tabela 5.5, o melhor resultado obtido por cada medida é destacado em negrito. Os resultados são comparados com um *baseline* que corresponde a um classificador que prediz a classe positiva para todos os exemplos de teste. Assim, este *baseline* apresenta o valor máximo de cobertura mas com baixa precisão na classificação dos exemplos da classe de interesse. É possível observar que para as medidas de avaliação Cobertura e AUC, o algoritmo *parzendd* apresentou os melhores resultados. Por outro lado, considerando as medidas Precisão, F-Measure e Acurácia, o algoritmo *svdd* apresentou os melhores resultados. Para melhor comparar os resultados, são apresentadas as curvas ROC obtidas na avaliação dos diferentes algoritmos de C1C na Figura 5.15.

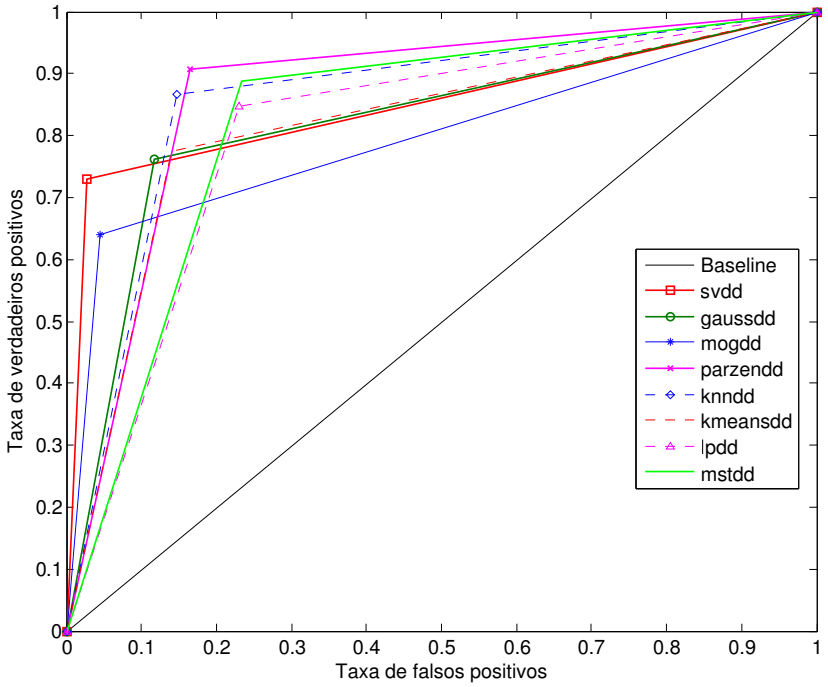


Figura 5.15: Curvas ROC obtidas pelos algoritmos de classificação com uma classe.

A partir dos resultados apresentados, é possível concluir que mesmo com uma quantidade reduzida de exemplos da classe positiva (347 exemplos) e a ausência de contra exemplos da classe negativa, é possível construir classificadores com 1 classe que sejam efetivos para o uso em armadilhas inteligentes resposáveis por capturar insetos da espécie *Aedes aegypti*. Em especial, os algoritmos baseados em janelas de Parzen (*parzendd*), k-Vizinhos Mais Próximos (*knndd*) e *Support Vector Description* (*svdd*) apresentaram os melhores resultados entre os algoritmos avaliados. No melhor caso, o algoritmo *parzendd* atingiu um valor de AUC de 0,87. Para comparar o comportamento destes três algoritmos, na Tabela 5.6 é apresentada uma matriz de confusão com os erros cometidos por cada um dos algoritmos *parzendd, knnd* e *svdd*, respectivamente.

É importante destacar que ainda há margem para a melhora destes resultados, mesmo que sensível, se considerarmos uma quantidade maior de dados da classe positiva, o uso de dados não rotulados que sejam representativos de diferentes classes, a otimização dos

Tabela 5.6: Erros e acertos cometidos pelos algoritmos de classificação com 1 classe janelas de Parzen (*parzendd*), k-Vizinhos Mais Próximos (*knndd*) e *Support Vector Description* (*svdd*), respectivamente.

Real	**Predito**	
	Ae. aegypti	**¬ *Ae. aegypti***
Ae. aegypti	505/483/407	52/74/150
¬ *Ae. aegypti*	727/652/118	3694/3769/4303

parâmetros dos algoritmos e a combinação das respostas de diferentes algoritmos em um esquema de comitê de classificadores, conforme discutido em Tax (2001).

5.10 Considerações Finais

Neste capítulo, foi apresentada uma revisão do estado-da-arte sobre diferentes trabalhos da literatura que visam realizar a classificação automática de insetos por meio de dispositivos eletrônicos. Os primeiros esforços nesta direção datam de 1945 e, nota-se, que mesmo nos dias atuais com a crescente evolução dos equipamentos eletrônicos, a realização desta tarefa ainda não foi atingida com um grau prático satisfatório. Basicamente, o maior problema relaciona-se ao uso de microfones para a gravação do comportamento dos insetos. Como é necessário um alto nível de sensibilidade destes equipamentos para o registro dos sons emitidos pelos insetos, os microfones se tornam extremamente sujeitos a presença de ruído externo, o que impossibilita o seu uso na prática. Desse modo, a gravação por meio óptico é capaz de reduzir drasticamente este problema e tornar a tarefa de classificação mais acessível. Nesta direção, foi apresentado neste trabalho um sensor capaz de realizar o registro de insetos utilizando uma fonte de luz laser.

Além da apresentação do funcionamento do sensor laser, também foi discutido como o sensor pode ser futuramente adaptado em uma armadilha inteligente capaz de atrair e capturar seletivamente espécies de insetos de interesse. Para a correta classificação do sensor por meio de algoritmos de Aprendizado de Máquina e funcionamento da armadilha, é importante a extração de características discriminativas a partir dos dados obtidos pelo sensor. Embora os dados sejam obtidos ópticamente, eles são muito similares a dados de áudio. Por esta razão, foram apresentados neste capítulo, diferentes métodos de extração de características provenientes da área de processamento digital de sinais que podem ser utilizados para a caracterização de espécies de insetos. Também foi conduzida uma avaliação experimental para medir o desempenho dos diferentes métodos de extração de atributos em um conjunto de dados coletado pelo sensor laser identificador de insetos. Em resumo, os coeficientes mel-cepstrais (*Mel-Frequency Cepstral Coefficients* – MFCC) em conjunto com o algoritmo Máquina de Vetores de Suporte (SVM), se mostraram

eficientes na tarefa de classificação em um experimento com 5 diferentes espécies. Neste experimento, no melhor caso o algoritmo foi capaz de realizar a classificação de um fluxo de dados composto por 5.325 observações com uma acurácia média de 90,33% ao longo do tempo.

Por fim, dada a dificuldade do conhecimento e obtenção de dados de todas as possíveis espécies que podem cruzam a luz do sensor óptico em um cenário real fora do laboratório e que podem ser incorretamente classificadas em uma das classes previamente definidas, foram avaliados oito diferentes algoritmos que realizam a etapa de treinamento considerando somente uma classe do problema. Como há um interesse especial deste trabalho em insetos da espécie *Aedes aegypti*, foram conduzidos experimentos que consideraram somente uma pequena porção de dados desta espécie para a etapa de treinamento e dados de cinco diferentes espécies para a etapa de teste. Ainda que a classificação com uma única classe seja mais difícil de ser realizada que a classificação com múltiplas classes e exige uma quantidade maior de dados para treinamento, os experimentos conduzidos e apresentados neste capítulo mostraram que os algoritmos da literatura são adequados para esta aplicação. Em especial, o algoritmo de classificação com uma classe baseado no uso de janelas de Parzen apresentou um resultado de AUC = 0,87 que representa um nível adequada em relação a taxa de falsos positivos e verdadeiros positivos na identificação de insetos da espécie *Aedes aegypti*.

Os resultados obtidos em ambas as avaliações conduzidas neste capítulo são motivadores, dada a importância da tarefa para a solução de problemas de saúde pública, agricultura, pecuária e meio ambiente. Por exemplo, o conhecimento da distribuição populacional de mosquitos transmissores de doenças como dengue, febre amarela e malária, obtido a partir do uso de sensores identificadores de insetos, permitirão ações mais efetivas e em tempo hábil por parte de órgãos governamentais nas regiões com maior incidência do mosquito. No caso da agricultura e pecuária, as informações obtidas pelo sensor permitirão um uso mais consciente, direcionado e de menor custo, de inseticidas e pesticidas para o controle de insetos pragas e mosquitos transmissores de doenças para os animais como a Leishmaniose.

Como o objetivo principal das avaliações conduzidas neste capítulo foi identificar quais as melhores características que descrevem os dados obtidos pelo sensor e os algoritmos mais adequados para o problema discutido, os dados coletados possuem natureza estacionária. Em outras palavras, os dados obtidos em condições controladas de laboratório não apresentam mudanças de conceito ao longo do tempo. Entretanto, a avaliação em um ambiente não-estacionário onde são consideradas mudanças em condições ambientais como temperatura, umidade e pressão atmosférica, responsáveis por alterar o comportamento dos insetos e, consequentemente os dados medidos, será apresentada no Capítulo 6, bem como a proposta de soluções para a tarefa de classificação sob condições de mudanças de conceitos.

Capítulo

6

Classificação Automática de Insetos em Ambientes Não Estacionários

6.1 Considerações Iniciais

No Capítulo 5, foi apresentado o sensor identificador de insetos explorado neste trabalho, bem como uma avaliação experimental sobre o desempenho de diferentes atributos descritivos dos dados obtidos pelo sensor em conjunto com algoritmos de aprendizado de máquina na tarefa de classificação de espécies em um fluxo de dados. Na ocasião, foi considerado um conjunto de dados coletado em um ambiente com variações pouco significativas nas condições meteorológicas. Entretanto, ao utilizar o sensor em campo, é necessário considerar que os dados obtidos podem ser potencialmente diferentes daqueles coletados em laboratório, devido a presença frequente de mudanças significativas em condições ambientais como temperatura, umidade, pressão do ar e luminosidade. Estes fatores são responsáveis por alterar o comportamento dos insetos e, consequentemente, os dados medidos pelo sensor, que passam a ser de natureza não estacionária. Estas alterações são responsáveis por inserir mudanças de conceito nos dados, sendo necessário o uso de métodos capazes de se adaptar a estas mudanças para que o desempenho da tarefa de classificação não apresente degradações ao longo do tempo, conforme anteriormente discutido no Capítulo 2. Além disso, nesta aplicação não é possível obter o rótulo correto de cada exemplo imediatamente após o seu processamento, como assumido pela maioria dos trabalhos da literatura em fluxo de dados. Na verdade, tem-se um problema real de classificação de fluxo de dados não estacionários com latência extrema, conforme discutido no Capítulo 3.

Neste capítulo serão apresentados os principais fatores responsáveis por alterar o comportamento dos insetos e a distribuição dos dados na Seção 6.2. Os procedimentos de coleta de dados não estacionários são apresentados na Seção 6.3. Uma análise das mudanças observadas nos dados é apresentada na Seção 6.4. As soluções propostas neste trabalho para lidar com mudanças de conceito na tarefa de classificação de insetos por meio de sensores são apresentadas e avaliadas na Seção 6.5.

6.2 Fatores Responsáveis por Alterar a Distribuição dos Dados

Diferentes fatores como a idade e sexo dos insetos, características meteorológicas como temperatura e umidade, além da localização no espaço e tempo em que o sensor está localizado durante as classificações, são responsáveis por alterar a distribuição dos dados medidos pelo sensor ao longo do tempo ou por alterar a probabilidade *a priori* da passagem de determinada espécie. Por isso, esses fatores representam um importante papel na tarefa de classificação e devem ser considerados pelo modelo de classificação. Entretanto, dificilmente será possível considerar todas as combinações destes fatores no modelo de classificação.

Esta seção tem o objetivo de apresentar os principais fatores identificados com base na literatura em Entomologia. Entretanto, é possível que características ainda não identificadas ou ocultas aos dados, também influenciem as distribuições de probabilidade dos dados.

6.2.1 Idade e Sexo

A frequência de batidas de asas de uma determinada espécie é diretamente influenciada pelo tempo de vida e o sexo do inseto, já que esses fatores alteram seu comportamento e causam diferenças anatômicas.

Desde muito tempo é sabido que a frequência de batidas de asas de insetos no início de seu ciclo de vida é menor do que a frequência observada em insetos adultos (Chadwick, 1953; Levenbook e Williams, 1956; Farnworth, 1972; Perumpral et al., 1974). A Figura 6.1 ilustra esse comportamento a partir da análise de insetos das espécies *Drosophila funebris* e *Phormia regina*. É possível observar na figura, um crescimento no valor médio de frequência de batidas de asas nos primeiros dias de vida de ambas as espécies. Tais valores são estabilizados após alcançado um patamar de frequência de batidas de asas, aproximadamente, no sétimo dia de vida dos insetos.

Em relação ao sexo dos insetos, diversos estudos apontam que a média de frequência de batida de asas das fêmeas é menor que dos machos quando comparados insetos de uma mesma espécie (Kahn et al., 1945; Kahn e Offenhauser Jr, 1949; Chadwick, 1953; Caprio et al., 2001). Isso se justifica principalmente devido as diferenças anatômicas entre

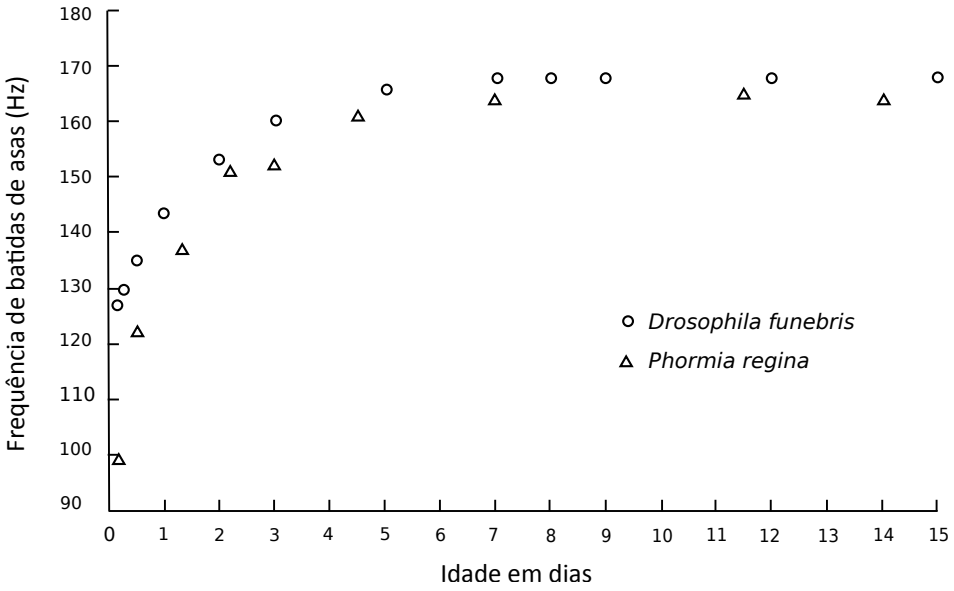
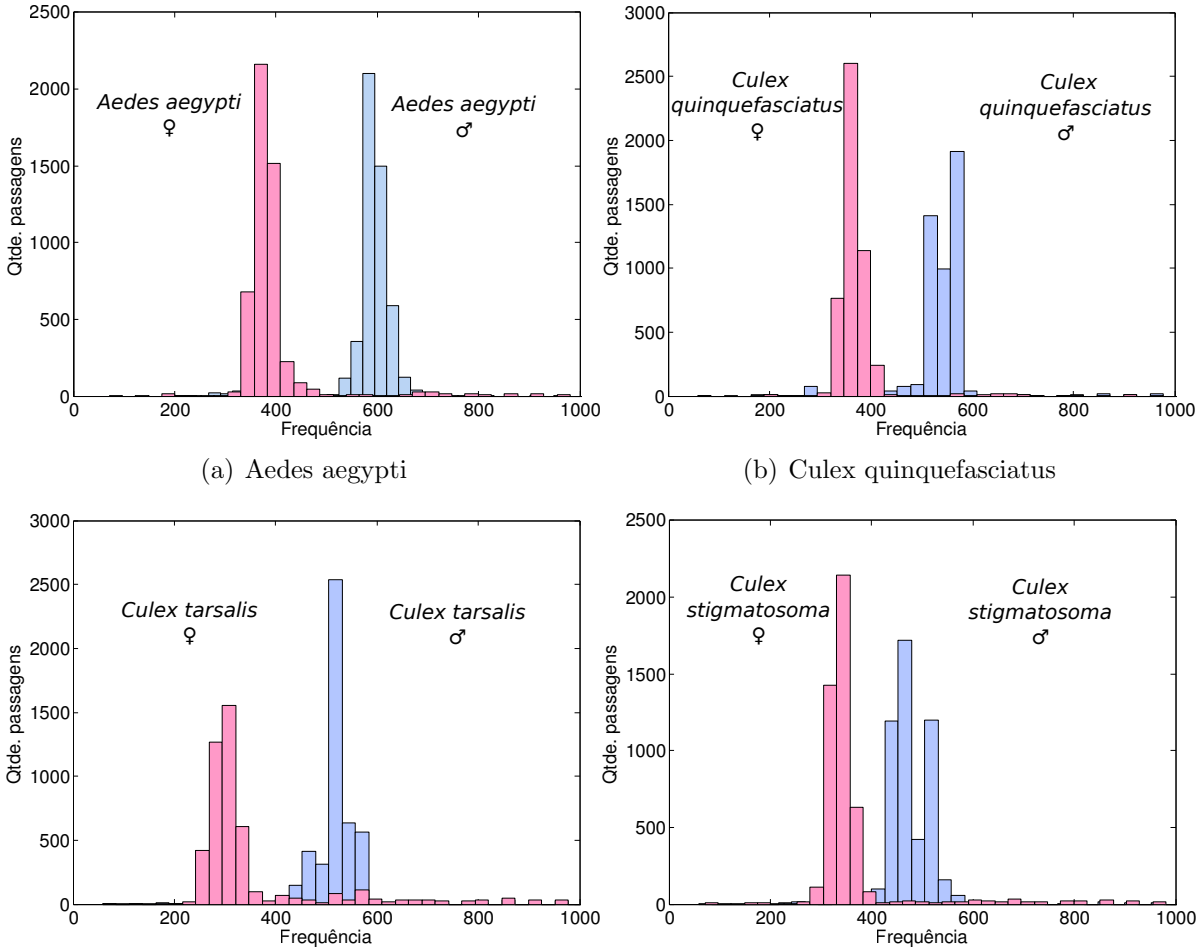


Figura 6.1: Relacionamento entre a frequência de batidas de asas e a idade dos insetos, considerando as espécies *Drosophila funebris* e *Phormia regina*. Cada ponto representa a média de frequência de batidas de asas de 10 a 20 observações de pelo menos 10 moscas de cada espécie (Levenbook e Williams, 1956).

os sexos, como pode ser observado na Figura 6.2, em que são ilustrados dois insetos da espécie *Aedes aegypti* de ambos os sexos, sendo possível observar diferenças no tamanho do corpo e nas antenas do inseto macho em relação à fêmea.

Figura 6.2: Representação de *Aedes aegypti* macho e fêmea. As fêmeas são maiores e os machos possuem uma quantidade maior de flagelo antenal.

Para verificar experimentalmente esta afirmação entre os sexos das espécies, realizou-se a extração da frequência de batida de asas a partir de dados coletados pelo sensor com 4 diferentes espécies de ambos os sexos: *Aedes aegypti, Culex quinquefasciatus, Culex tarsalis* e *Culex stigmatosoma*. Para isso, foi utilizado o conjunto de dados avaliado em Chen et al. (2014), em que a temperatura se manteve constante durante a etapa de coleta dos dados. Um histograma para cada uma das espécies é apresentado na Figura 6.3, onde é possível confirmar que a frequência de batida de asas das fêmeas é menor do que a observada em machos da mesma espécie.



(a) Aedes aegypti

(b) Culex quinquefasciatus

(c) Culex tarsalis

(d) Culex stigmatosoma

Figura 6.3: Histograma da frequência de batida de asas a partir de dados coletados das espécies *Aedes aegypti, Culex quinquefasciatus, Culex tarsalis* e *Culex stigmatosoma* de ambos os sexos. Nota-se que a frequência de batida de asas das fêmeas é menor do que a observada em machos da mesma espécie.

6.2.2 Características Meteorológicas

Temperatura, pressão atmosférica e umidade são variáveis meteorológicas que afetam os insetos por ao menos dois motivos principais. Em primeiro lugar, algumas espécies são mais adaptadas para sobreviver em determinadas condições ambientais. Por exemplo, muitas espécies de mosquitos são originárias de regiões tropicais e subtropicais, onde o clima é normalmente quente e úmido. Portanto, em períodos de clima quente e úmido, a prevalência de insetos dessas espécies pode aumentar.

A segunda razão é que as variações climáticas afetam o comportamento de insetos, uma vez que seu metabolismo é influenciado pela temperatura (Chadwick, 1939; Taylor, 1963; Farnworth, 1972), bem como pela pressão atmosférica (Chadwick e Williams, 1949) e umidade (Mellanby, 1936). Além disso, a temperatura influencia as propriedades aerodinâmicas do ar. Desse modo, é esperado que a frequência de batidas de asas au-

mente proporcionalmente à temperatura. Há pesquisas significativas que sugerem que a temperatura tem efeito linear em uma ampla faixa de valores para a maioria dos insetos (Reed et al., 1941). Esse comportamento pode ser observado no exemplo apresentado na Figura 6.4, em que se verifica esta relação em um experimento conduzido por Reed et al. (1941) com duas subespécies da mosca *Drosophila pseudoobscura*.

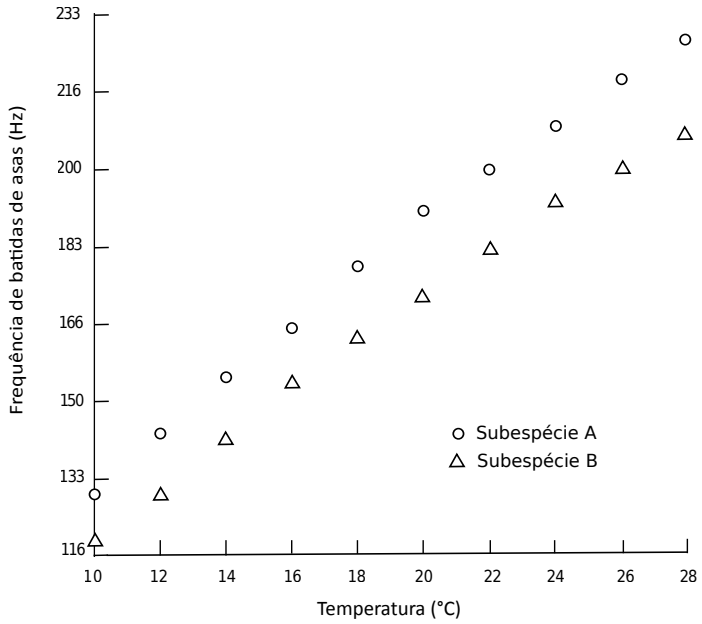


Figura 6.4: Relacionamento entre a frequência de batidas de asas e a temperatura, considerando duas subespécies de insetos da espécie *Drosophila pseudoobscura* (Reed et al., 1941).

Na Figura 6.4 é possível observar que as médias de frequência de batidas de asas de ambas as subespécies da mosca *Drosophila pseudoobscura* aumentam proporcionalmente entre a faixa de 116Hz até 233Hz, considerando-se o aumento de temperatura de 10°C até 28°C.

6.2.3 Ritmo Circadiano

A hora do dia é uma característica interessante para a análise do comportamento dos insetos. Muitos mosquitos são mais ativos durante determinadas horas do dia, o que caracteriza o ritmo circadiano da espécie. Esse fato é conhecido desde a antiguidade e é estudado por quase cem anos (Roubaud, 1918). Insetos podem ser classificados como noturnos (ativos durante a noite) ou diurnos (ativos durante o dia). Os mosquitos podem ser classificados como crepusculares, ou seja, mais ativos durante o amanhecer ou ao anoitecer. Dentro dessa classificação, os insetos podem ser matinais ou vespertinos, o que indica se a espécie é ativa somente no amanhecer ou somente no anoitecer, respectivamente.

6.3 Coleta de Dados com Mudanças Ambientais

A primeira versão do sensor identificador de insetos desenvolvida em 2011 era totalmente analógica e capaz de registrar somente sinais de áudio por meio de gravadores digitais (Batista et al., 2011b). A partir de tais sinais foi possível extrair informações sobre a frequência de batidas de asas de diferentes espécies, além de outros atributos provenientes do sinal e avaliar o uso de algoritmos de aprendizado de máquina na tarefa de classificação, conforme discutido no Capítulo 5.

Com a necessidade de monitorar fatores ambientais responsáveis por alterar o comportamento dos insetos, foi posteriormente desenvolvido em 2015 um sensor totalmente digital para a realização de experimentos com mudanças de conceitos nos dados. Este sensor é capaz de coletar atributos meteorológicos como temperatura, umidade, pressão atmosférica e luminosidade, em conjunto com os dados sobre o voo dos insetos. Este sensor é ilustrado na Figura 6.5.

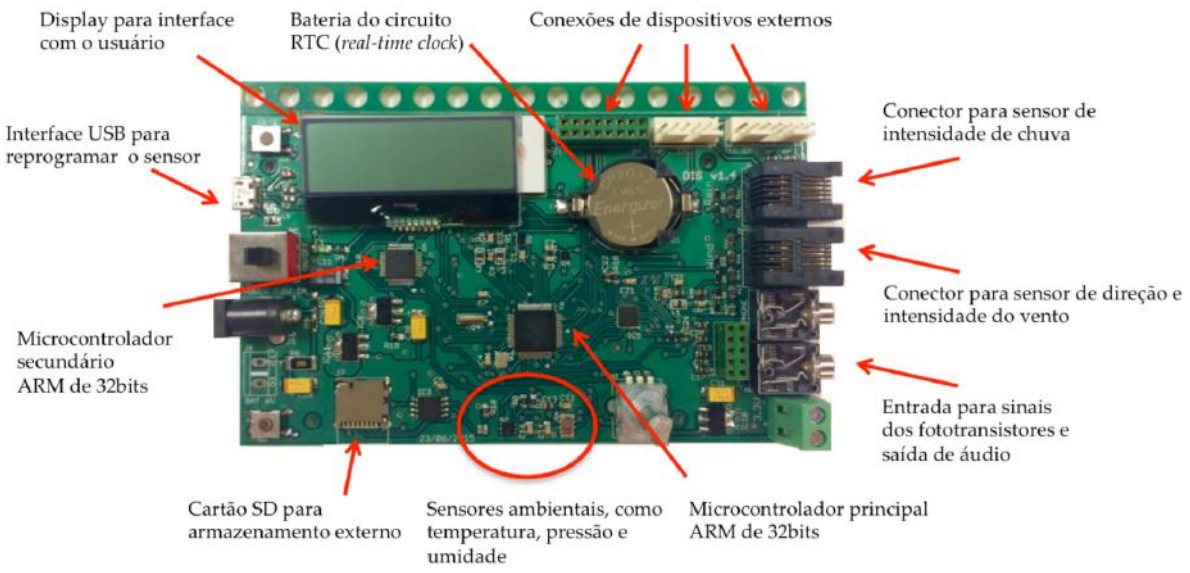


Figura 6.5: Placa de circuito do sensor digital para a coleta de dados. Estão indicadas na imagem os principais componentes de aquisição e processamento de dados da placa.

No experimento com mudanças ambientais realizado neste trabalho, foram consideradas as espécies *Aedes aegypti, Aedes albopictus* e *Culex quinquefasciatus*. Estes insetos foram gentilmente cedidos pelo Departamento de Parasitologia do Instituto de Ciências Biomédicas da Universidade de São Paulo (ICB/USP). Na Figura 6.6 é possível observar os insetos acomodados em recipientes plásticos antes da etapa de coleta de dados, sendo necessária a transferência dos espécimes dos recipientes para os insetários onde foram realizadas as coletas dos dados pelo sensor. Para a transferência dos insetos, foi utilizado um tubo de aspiração adaptado em uma mangueira plástica com um filtro em seu interior.

Figura 6.6: Insetos das espécies *Aedes aegypti, Aedes albopictus* e *Culex quinquefasciatus* acomodados em recipientes plásticos antes da transferência para os insetários onde foi realizada a coleta dos dados com mudanças ambientais.

Para a coleta dos dados, foram utilizados cinco insetários adaptados com o sensor, onde cada insetário continha entre 100 e 200 mosquitos de uma única espécie, para que fosse possível rotular os dados com total confiança. Quando possível, os mosquitos de uma mesma espécie foram separados em sexo e dispostos em diferentes insetários. Durante a coleta dos dados, os cinco insetários registraram dados de diferentes espécies em paralelo, conforme apresentado na Figura 6.7. Desse modo, após a coleta dos dados foi possível sincronizar e ordenar os dados obtidos pelos diferentes sensores a partir do horário de cada passagem registrada, formando um único fluxo de dados com cinco classes.

Figura 6.7: Conjunto de insetários adaptados com o sensor utilizados para coleta de dados com mudanças de temperatura. Diferentes insetários permitem a coleta de dados em paralelo de diferentes espécies separadas por sexo.

Com o objetivo de analisar o impacto da temperatura nos dados registrados pelo sensor, foram coletados dados considerando variações desta variável. Para isso, os insetários foram dispostos em um laboratório munido de aparelhos convencionais de condicionamento e umidificação do ar. Desse modo, a temperatura do ambiente foi variada com o uso de condicionador de ar e a umidade foi controlada com o uso de um aparelho umidificador na tentativa de evitar que a umidade relativa do ar no laboratório ficasse abaixo

de 40%, dado que a umidade ótima para o desenvolvimento dos insetos é entre 40% e 80% (de Carvalho, 1986). Durante a coleta dos dados, utilizou-se a iluminação natural do ambiente para que o fotoperíodo dos insetos fosse respeitado e permitisse observar o ritmo circadiano das diferentes espécies ao longo dos dias. Ainda, buscou-se evitar o trânsito de pessoas no laboratório durante a coleta dos dados, para que a atividade dos insetos não fosse influenciada por fatores externos. Entretanto, foi necessária a entrada diária no laboratório ao menos duas vezes ao dia para a alimentação dos espécimes com algodão embebido em solução açucarada.

6.4 Descrição e Análise dos Dados

Conforme discutido, neste trabalho foram coletados dados de voo de insetos com diferentes alterações ambientais. Nesta seção, serão apresentadas uma descrição e análise do conjunto de dados construído. Pelo melhor de nosso conhecimento, este é o primeiro conjunto de dados real com mudanças de conceito induzidas artificialmente em que é possível observar claramente o impacto de fatores externos nos atributos que descrevem os dados.

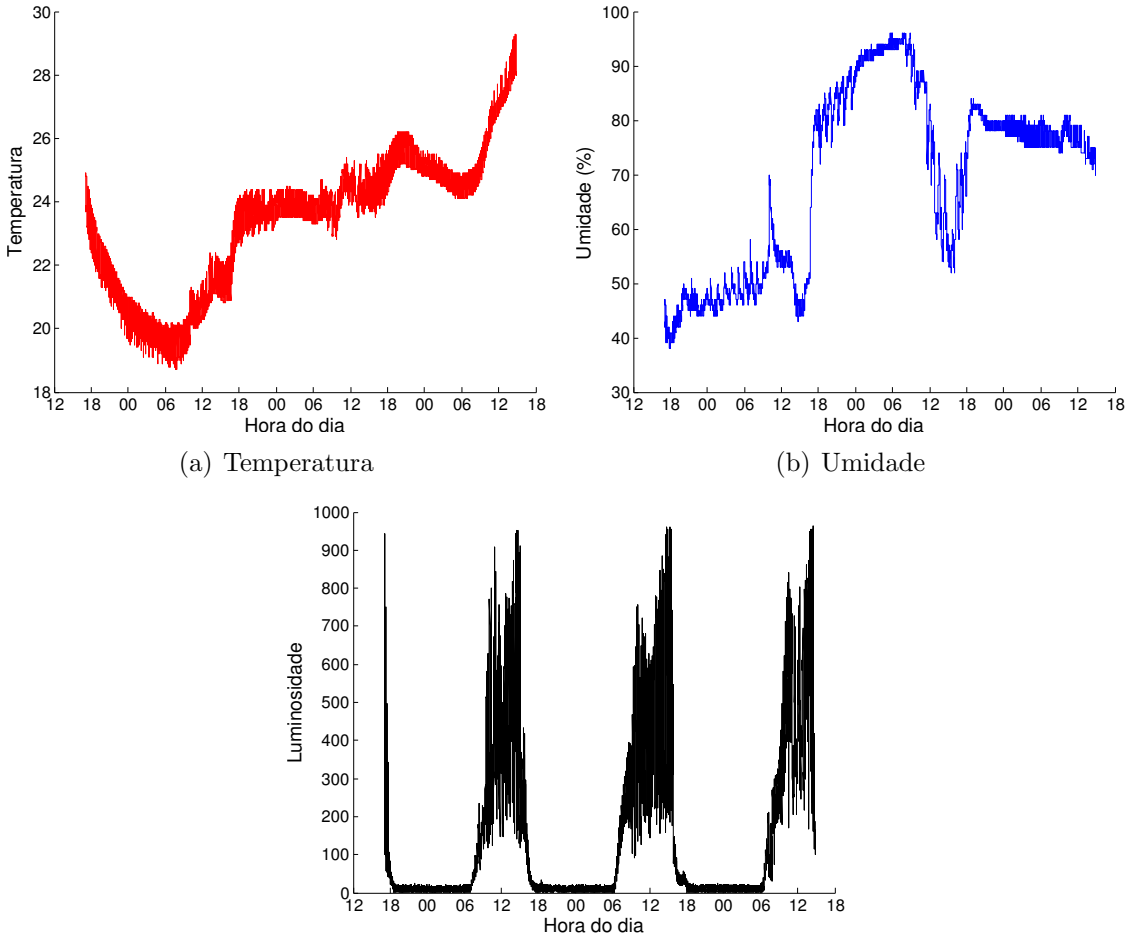
A coleta dos dados foi realizada durante 69 horas e 44 minutos. Neste período, foram registrados um total de 83.339 eventos. Entretanto, devido ao registro incorreto de ruído ou de passagens pouco significativas (por exemplo, pelo cruzamento de somente uma pequena parte do inseto pelo sensor), foi realizada uma etapa de limpeza dos dados. Nesta etapa, considerou-se a remoção dos exemplos que não se enquadraram na faixa de frequência conhecida de batimento de asas de insetos entre 100 e 1000 Hz, e que, possivelmente representariam ruído, totalizando 72.719 exemplos. A distribuição dos dados é apresentada na Tabela 6.1.

Tabela 6.1: Distribuição das classes do conjunto de dados de insetos com variações de temperatura.

Espécie do inseto	**Quantidade de exemplos (%)**
Aedes aegypti (fêmea)	10.962 (15,07)
Aedes aegypti (macho)	20.609 (28,34)
Aedes albopictus (fêmea)	7.234 (9,95)
Aedes albopictus (macho)	11.687 (16,07)
Culex quinquefasciatus (fêmea e macho)	22.227 (30,57)
Total	72.719 (100)

A partir da Tabela 6.1, nota-se que ambas as espécies de *Aedes* foram separadas por sexo durante a coleta dos dados. Em relação a espécie *Culex quinquefasciatus*, os insetos foram fornecidos com ambos os sexos acomodados em um mesmo recipiente, dificultando a separação e transferência para os insetários. Por isso, foram considerados machos e fêmeas da espécie *Culex quinquefasciatus* em uma mesma classe.

Neste conjunto de dados houve uma variação de temperatura de mais 10°C, com mínima de 18, 7°C, média de 23, 94°C e máxima de 29, 3°C. Em relação a umidade relativa do ar, a variação foi entre 38% e 96%, com média em 70, 2%. Também foram registrados valores referentes a luminosidade do ambiente em uma escala entre 0 e 1023, em que 0 representa a total escuridão e 1000 a total claridade do ambiente. Este registro é importante para verificar se o fotoperíodo dos insetos foi respeitado. Além disso, este fator pode influenciar no ritmo circadiano dos insetos. Os registros destas condições ao longo dos dias de coleta são apresentados na Figura 6.8.



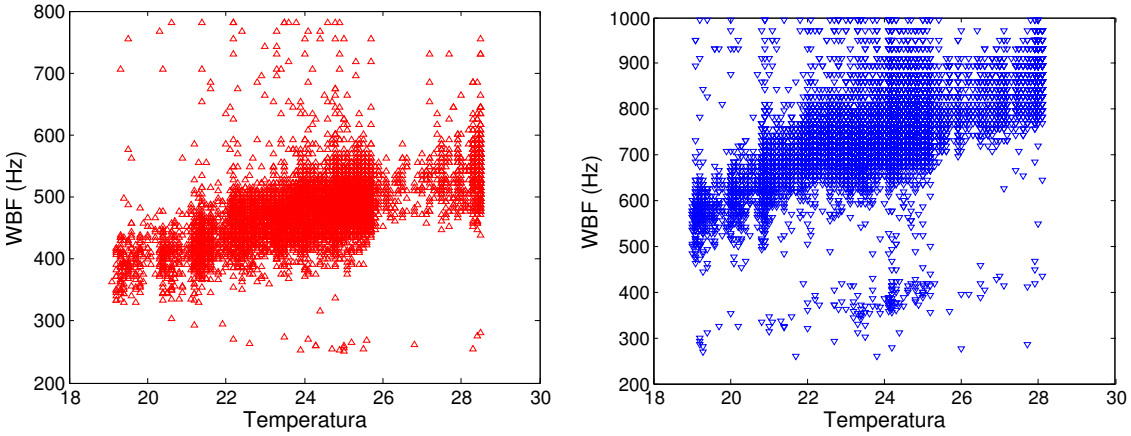
(a) Temperatura

(b) Umidade

(c) Luminosidade

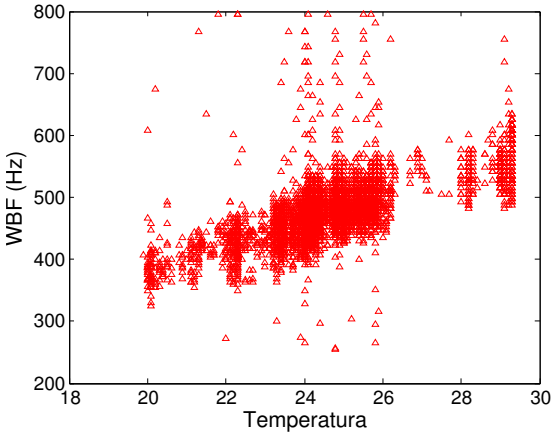
Figura 6.8: Variações observadas ao longo dos dias de coleta nas condições de temperatura, umidade e luminosidade, no conjunto de dados com mudanças ambientais.

Para verificar a relação da frequência de batida de asas dos insetos com a temperatura, foi extraída a frequência fundamental dos sinais a partir da análise cepstral e relacionada com a variação de temperatura observada durante a coleta dos dados. Na Figura 6.9, são apresentados os dados observados para os insetos do gênero *Aedes*. Em todos os casos, pode-se observar a concentração dos eventos em torno de uma média com tendência de aumento linear de ao menos 100 Hz, dada uma variação de 10°C na temperatura.

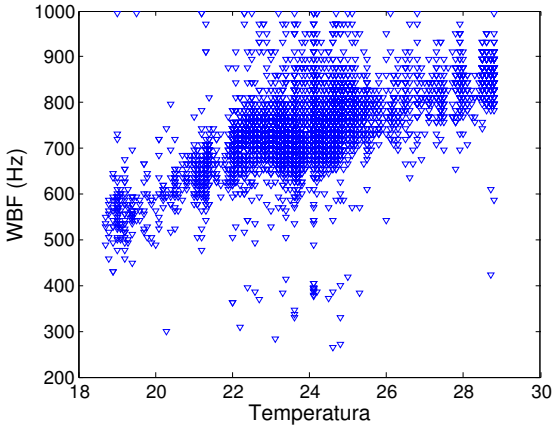


(a) *Ae. aegypti* (fêmea)

(b) *Ae. aegypti* (macho)



(c) *Ae. albopictus* (fêmea)



(d) *Ae. albopictus* (macho)

Figura 6.9: Frequência de batida de asas observada com a mudança na temperatura, considerando insetos do gênero *Aedes*. Cada ponto representa uma observação.

Este comportamento é resumido na Figura 6.10, em que são apresentadas as médias e desvios padrão da frequência de batida de asas dadas diferentes faixas de temperatura.

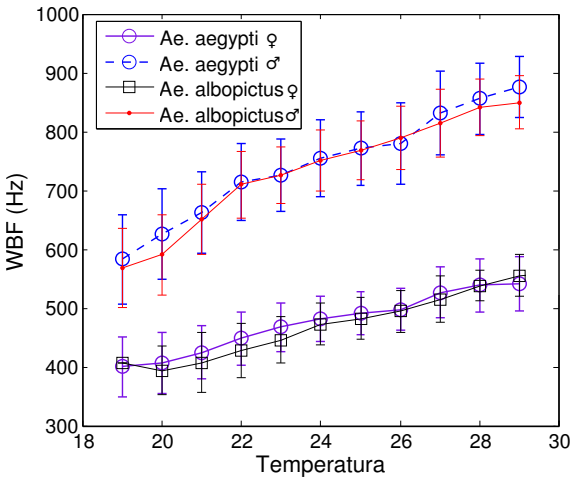


Figura 6.10: Média e desvio padrão da frequência de batida de asas dada diferentes faixas de temperatura para as espécies *Aedes aegypti* e *Aedes albopictus*, de ambos os sexos.

Em ambas as espécies, pode-se observar que os machos sofrem maior influência da temperatura do que as fêmeas. Por exemplo, os machos da espécie *Aedes aegypti* apresentam uma frequência média de batida de asas que varia de 583 Hz aos 19°C e 876 Hz em 29°C, uma diferença de 293 Hz. Por outro lado, as fêmeas apresentam uma frequência média de batida de asas que varia de 400 Hz aos 19°C e 540 Hz em 29°C, uma diferença de 140 Hz. Diferenças similares também são observadas para os mosquitos da espécie *Aedes albopictus*.

Este mesmo padrão comportamental de aumento da frequência de batidas de asas de acordo com a temperatura também é observado para a espécie *Culex quinquefasciatus*, conforme apresentado na Figura 6.11. Entretanto, para esta espécie foram considerados insetos de ambos os sexos em um mesmo insetário durante a coleta dos dados. Assim, também é possível observar na figura a separação dos dados entre o sexo dos mosquitos, de modo que a concentração de pontos na parte inferior do gráfico refere-se às passagens das fêmeas, enquanto a concentração na parte superior refere-se às passagens dos machos. Novamente, é possível constatar que a frequência de batida de asas dos insetos machos sofrem maior influência da temperatura, comparados aos insetos fêmeas.

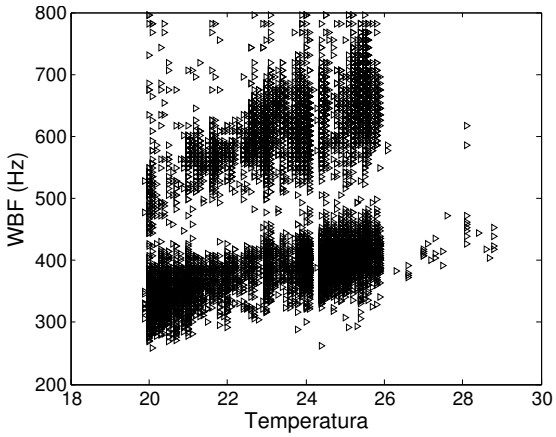


Figura 6.11: Frequência de batida de asas de mosquitos machos e fêmeas da espécie *Culex quinquefasciatus* observada com a mudança na temperatura.

Na Figura 6.12, é possível observar a relação entre a umidade relativa do ar em diferentes faixas de valores e a frequência de batida de asas dos mosquitos do gênero *Aedes*. Neste caso, nota-se um menor impacto desta variável na frequência de batida de asas dos insetos, se comparada à temperatura.

É importante destacar que para uma análise mais precisa sobre a influência isolada destes fatores ambientais no comportamento dos insetos, seria necessário fixar o valor de uma das variáveis, enquanto são avaliados diferentes valores para a segunda variável. Entretanto, temperatura e umidade estão intimamente relacionadas, sendo difícil realizar este tipo de análise com os equipamentos utilizados durante a coleta dos dados. Na tentativa de ilustrar a relação da frequência de batida de asas, tanto com a temperatura

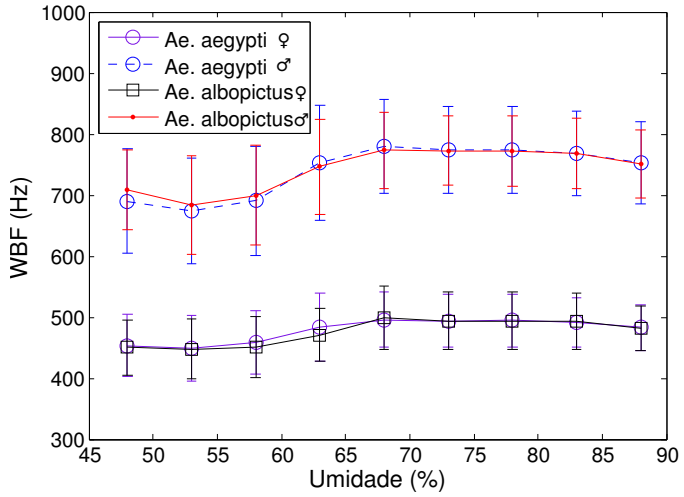


Figura 6.12: Média e desvio padrão da frequência de batida de asas dada diferentes faixas de umidade relativa do ar para as duas espécies *Aedes*, de ambos os sexos.

como umidade, são apresentados na Figura 6.13, os exemplos observados de acordo com a variação nestas duas variáveis para os insetos machos da espécie *Aedes albopictus*.

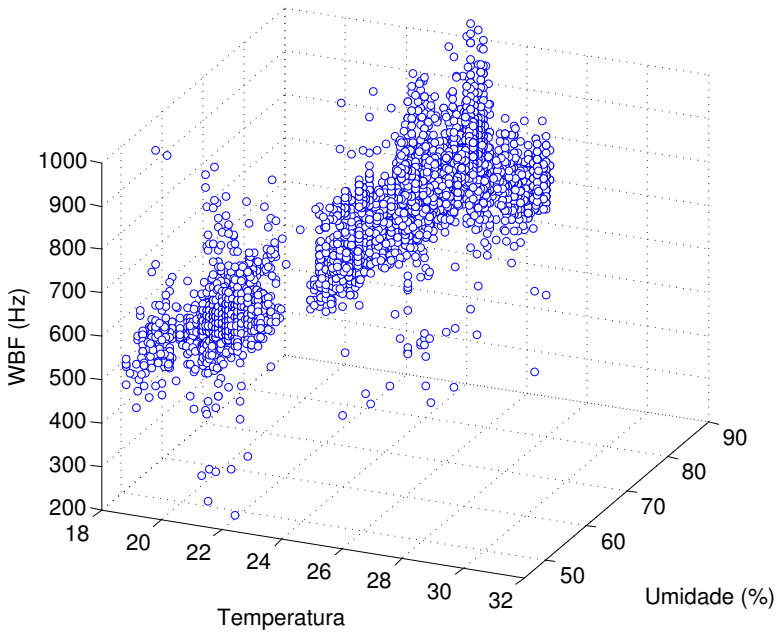
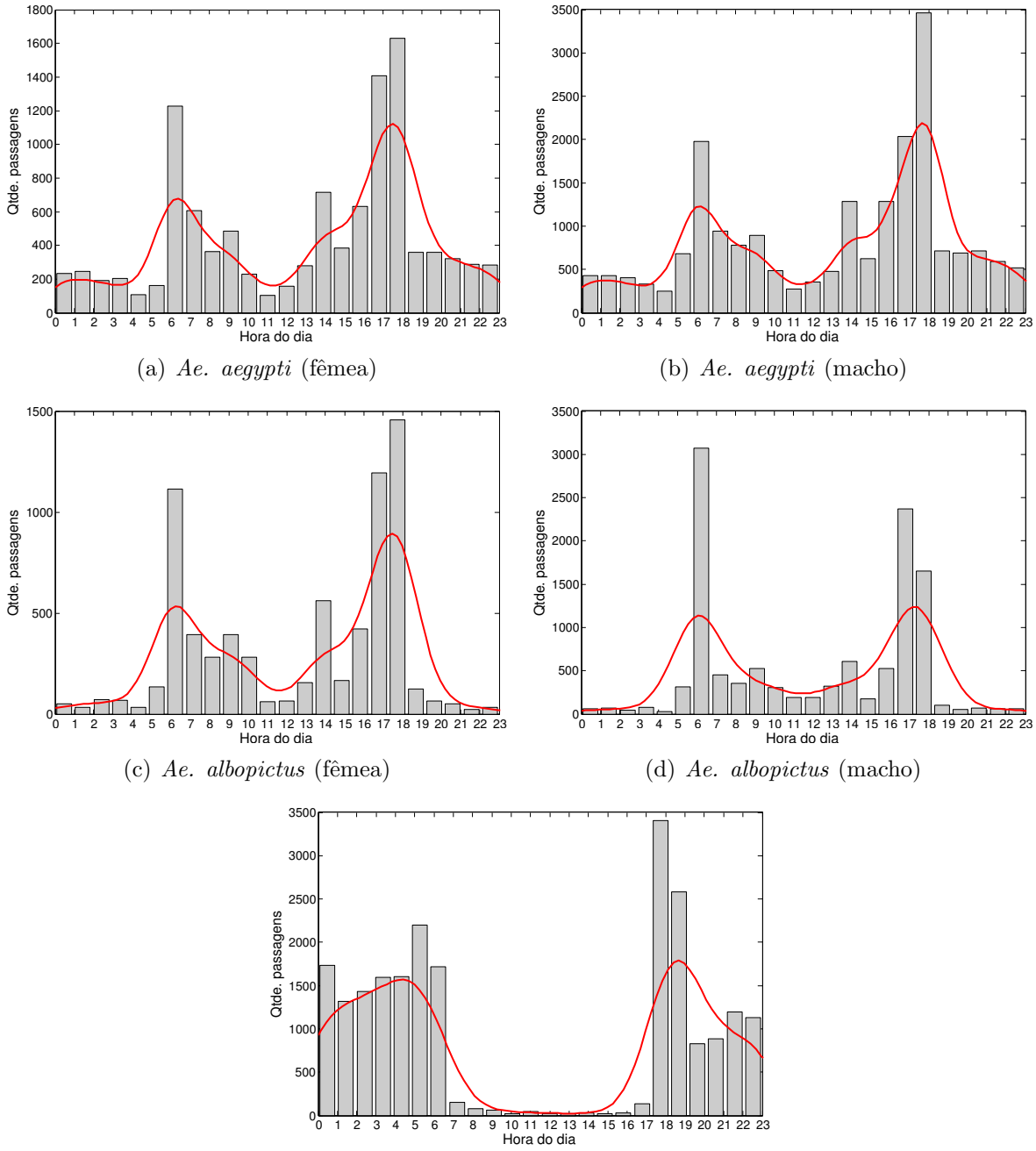


Figura 6.13: Relação entre frequência de batida de asas, umidade e temperatura para os insetos machos da espécie *Aedes albopictus*.

Em relação ao ritmo circadiano, confirmou-se experimentalmente que os insetos avaliados do gênero *Aedes* são diurnos, enquanto os insetos da espécie *Culex quinquefasciatus* são noturnos. Mais especificamente, os insetos *Aedes* apresentam maior atividade entre 6 e 18 horas, enquanto os insetos *Culex* apresentam maior atividade entre 18 e 6 horas. Na Figura 6.14 são apresentados os histogramas que representam as atividades das diferentes espécies ao longo do dia.



(a) *Ae. aegypti* (fêmea)

(b) *Ae. aegypti* (macho)

(c) *Ae. albopictus* (fêmea)

(d) *Ae. albopictus* (macho)

(e) *Culex quinquefasciatus* (macho e fêmea)

Figura 6.14: Ritmo circadiano observado de acordo com a quantidade de passagens de cada espécie pelo sensor no conjunto de dados com variações ambientais.

A única ressalva em relação aos histogramas da Figura 6.14, é quanto ao horário de manutenção dos insetos, que ocorreram em dois horários do dia durante a coleta dos dados: às 9 e 14 horas. Por este motivo, os insetos podem ter sido influenciados a apresentar maior atividade nestes horários. Entretanto, ao aproximar os histogramas a partir de uma curva, como realizado na Figura 6.14, este maior número de atividades é suavizado nestes horários. Desse modo, é mais confiável analisar o ritmo circadiano das espécies a partir das curvas, ao invés de considerar somente os números absolutos de atividades.

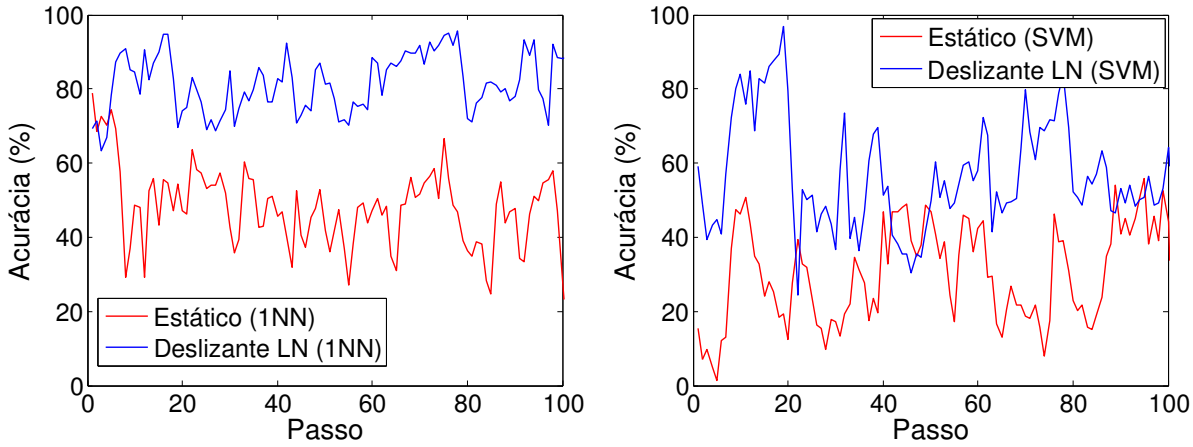
Com o objetivo de verificar experimentalmente a hipótese de que alterações nas condições de temperatura e umidade são responsáveis por reduzir o desempenho de classificadores que não se adaptam à mudanças de conceito no problema de identificação de espécies de insetos, foram avaliados classificadores estáticos e dinâmicos no conjunto de dados com mudanças ambientais.

O classificador estático é treinado uma única vez com uma porção dos dados, enquanto o classificador dinâmico tem o seu conjunto de treinamento constantemente atualizado a cada novo exemplo processado, de modo que o exemplo mais antigo é descartado do conjunto de treinamento para que seja adicionado o exemplo mais recente acompanhado do respectivo rótulo correto. Assim, espera-se que em um ambiente com constantes mudanças como o avaliado, o classificador dinâmico apresente uma taxa de acerto consideravelmente superior ao classificador estático. Por ser apenas um experimento para a verificação da hipótese levantada neste trabalho, será considerado o cenário de latência nula para a atualização do classificador dinâmico. Assim como no Capítulo 4, estas abordagens serão referenciadas como Estático e Deslizante LN (Latência Nula).

Para o treinamento do classificador Estático, foram utilizados dados referentes as primeiras 6 horas da coleta de dados. Este tempo contempla os 11.500 primeiros exemplos ou $15,8\%$ dos dados para o treino e $84,2\%$ dos dados para teste do classificador. Devido a necessidade de constantes atualizações do classificador Deslizante LN e visando a redução do tempo desta operação, optou-se por reduzir o tamanho da janela para o armazenamento dos exemplos de treino. Desse modo, considerou-se uma janela de 1.000 exemplos. Ambos os classificadores, Estático e Deslizante LN, foram treinados com os algoritmos SVM e 1NN, devido aos resultados anteriormente apresentados e discutidos no Capítulo 5. Do mesmo modo, foram utilizados 40 MFCC como atributos para a descrição dos dados, além de mais cinco novos atributos: horário da passagem, temperatura, umidade, luminosidade e frequência de batida de asas.

Uma visão do desempenho destas configurações ao longo tempo é ilustrada na Figura 6.15. Na Figura 6.15-*a* é apresentado o desempenho das configurações Estático e Deslizante LN, com o algoritmo 1NN. Pode-se observar a superioridade do classificador que é atualizado constantemente, ainda que sejam considerados menos exemplos para o treinamento do classificador. Na Figura 6.15-*b* são apresentados os resultados considerando o algoritmo SVM para a indução dos classificadores. Neste caso, embora a configuração Deslizante LN seja superior ao Estático, pode-se observar que, em determinados períodos de tempo, ambas as configurações apresentaram resultados similares. Em geral, o algoritmo 1NN se mostrou superior ao algoritmo SVM na avaliação das configurações Estático e Deslizante LN. Supõe-se que este comportamento se justifica à redução do peso dos atributos horário da passagem, temperatura, umidade, luminosidade e frequência de batida de asas, dada pelo classificador SVM em comparação ao classificador 1NN. Esta suposição é suportada por experimentos de classificação considerando somente os coefici-

entes MFCC, com os algoritmos 1NN e SVM. Neste caso, ambos os algoritmos apresentam desempenho equivalente e, ao incluir os novos atributos, o algoritmo 1NN passa a apresentar resultados superiores, conforme discutido.



(a) 1NN (b) SVM

Figura 6.15: Desempenho das configurações Estático e Deslizante LN com o algoritmos 1NN e SVM no conjunto de dados de insetos com mudanças ambientais.

Um resumo dos resultados de acurácia média ao longo de todo o fluxo obtidos pelas configurações Estático e Deslizante LN, tanto com o algoritmo 1NN como com o SVM, é apresentado na Tabela 6.2.

Tabela 6.2: Resultados de acurácia obtidos pelos classificadores Estático e Deslizante LN com os algoritmos 1NN e SVM no conjunto de dados de insetos com mudanças ambientais. O classificador Estático é treinado uma única vez com instâncias provenientes das primeiras 6 horas de coleta de dados. O classificador Deslizante LN utiliza uma janela com 1.000 exemplos, atualizada a cada novo exemplo processado.

	Algoritmo	
	1NN	**SVM**
Estático	48,09	30,03
Deslizante LN	82,06	56,76

Devido aos resultados alcançados pelo algoritmo 1NN, as soluções propostas na Seção 6.5 consideram este algoritmo para a construção dos classificadores. Além disso, por se tratar de um algoritmo incremental e que não necessita da etapa de indução, é importante que este algoritmo apresente resultados satisfatórios, dado que em trabalhos futuros, pretende-se explorar a atualização destes classificadores. Entretanto, é importante destacar que outros algoritmos podem ser utilizados.

6.5 Sistema com Múltiplos Classificadores

Ao observar os dados com mudanças de temperatura, conforme anteriormente apresentado nas Figuras 6.9 e 6.11, pode-se ter a falsa intuição de que as mudanças de conceito nestes dados são exclusivamente incrementais. Entretanto, nestes gráficos os dados estão ordenados somente pela temperatura, sem qualquer relação temporal entre as ocorrências. Na Figura 6.16, os dados de frequência de batida de asas da espécie *Culex quinquefasciatus* são apresentados temporalmente ordenados, sendo possível uma análise a respeito dos tipos de mudanças de conceito.

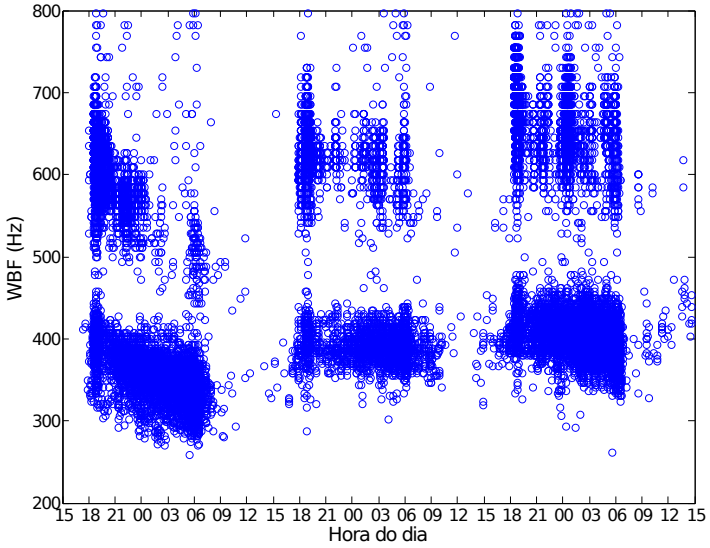


Figura 6.16: Dados da frequência de batida de asas da espécie *Culex quinquefasciatus* temporalmente ordernados.

Pode-se observar na Figura 6.16 que, embora ocorram mudanças incrementais nos dados, devido a inatividade dos insetos em determinados períodos de tempo associada ao ritmo circadiano da espécie, também podem ocorrer mudanças abruptas nos dados. De maneira resumida, se houveram mudanças ambientais significativas no intervalo de tempo entre duas ocorrências consecutivas de uma mesma espécie, os dados serão significativamente diferentes, constituindo uma mudança maior. Em um cenário real de aplicação do sensor, a ocorrência consecutiva de passagens de uma mesma espécie pode levar um considerável período de tempo, de modo que as mudanças sejam mais severas que as observadas na Figura 6.16. Além disso, a inatividade de determinadas espécies durante determinados períodos torna a quantidade classes do problema não constante ao longo de todo o tempo. Devido a estas características dos dados, as abordagens de classificação SCARGC e MClassification, propostas no Capítulo 4, não são adequadas para este problema. Para o bom desempenho destas abordagens, é necessário que as mudanças de conceito sejam exclusivamente incrementais e que o número de classes seja constante.

Desse modo, nesta seção são apresentadas novas soluções para o problema de classificação de insetos em ambientes não estacionários.

Com o objetivo de alcançar um bom desempenho de acurácia de classificação em problemas de fluxo de dados, uma solução intuitiva é dividir o conjunto de dados em vários subconjuntos, construir um classificador com os dados de cada um destes conjuntos e então integrá-los para a tomada de decisão ao classificar um novo exemplo de teste do fluxo (Wang et al., 2003; Kolter e Maloof, 2003). A justificativa do bom desempenho desta solução se deve ao fato de se explorar uma variedade de classificadores construídos a partir de diferentes conjuntos de atributos, diferentes conjuntos de dados ou diferentes algoritmos de aprendizado. Desse modo, é suposto que dado um exemplo de teste, haverá um classificador que seja mais adequado para a geração de uma resposta de classificação para tal exemplo. Esta suposição é suportada pela inviabilidade de construção de um único classificador suficientemente genérico que possa ser utilizado em qualquer situação. Outra justificativa para o uso de múltiplos classificadores para a classificação de fluxo de dados é a eficiência, dado que as etapas de classificação e atualização dos classificadores (caso necessária) podem ser realizadas de maneira mais eficiente com uma quantidade reduzida de dados.

Dada uma quantidade massiva de dados como as presentes em aplicações de fluxo de dados, esta solução pode gerar um grande número de classificadores base, sendo necessário o uso de técnicas de Sistemas com Múltiplos Classificadores (SMC) para integrar tais classificadores (Huang e Suen, 1995; Ali e Pazzani, 1996). Basicamente, as estratégias que podem ser aplicadas para a integração de diferentes classificadores podem ser divididas em duas categorias (Zhu et al., 2004, 2006):

1. Combinação de classificadores. Ao classificar um exemplo de teste, os resultados de todos os classificadores base são combinados para que uma decisão final seja realizada. Nesta estratégia, espera-se que os classificadores do conjunto cometam erros independentes entre si para que se obtenha ganhos em acurácia. Entretanto, em aplicações reais é difícil projetar e treinar um conjunto de classificadores independentes;

2. Seleção de classificadores. Nesta estratégia, todos os classificadores base são avaliados e somente o melhor classificador (de acordo com algum critério de avaliação e/ou escolha) é utilizado para determinar o resultado de classificação de um exemplo de teste.

Como as mudanças de conceito em fluxo de dados ocorrem ao longo tempo e, no caso do sensor identificador de insetos, tem-se o conhecimento de determinadas variáveis responsáveis por alterar os dados, neste trabalho há um interesse especial nas estratégias de seleção dinâmica de classificadores. Neste tipo de estratégia, a escolha do classificador

é realizada durante a etapa de classificação e depende essencialmente do exemplo de teste a ser processado. Assim, pode-se escolher o classificador mais adequado (previamente induzido) de acordo com fatores ambientais e temporais no exato momento da classificação do exemplo.

Dada a análise do conjunto dados de insetos com alterações ambientais construído neste trabalho, é fácil observar que existem duas variáveis principais responsáveis por alterar os dados ao longo do tempo ou a probabilidade de ocorrência de uma determinada classe. Primeiro, a temperatura é responsável por alterar a distribuição dos dados, de modo que, com o seu aumento, a frequência de batida de asas dos insetos passa a ser maior, independente da espécie e com maior impacto para as fêmeas. Em segundo, a probabilidade *a priori* da ocorrência de determinada espécie pode ser ditada pelo horário do evento, dado o ritmo circadiano observado para diferentes espécies. Assim, a construção dos classificadores que integram o Sistema com Múltiplos Classificadores proposto neste trabalho, considera estas duas variáveis.

Para a construção dos múltiplos classificadores, são selecionados determinados exemplos para a indução dos classificadores, enquanto os exemplos restantes são utilizados na etapa de teste do sistema construído, em um esquema similar ao procedimento de avaliação *Holdout*. Uma visão geral deste processo é ilustrado na Figura 6.17.

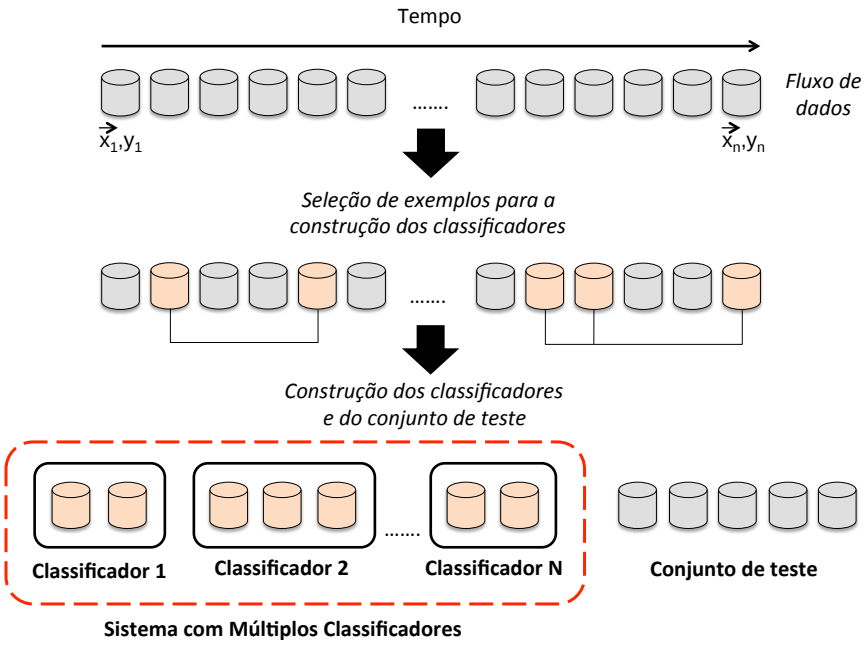


Figura 6.17: Esquema ilustrativo do processo de construção do Sistema com Múltiplos Classificadores (SMC). Uma parcela dos dados é utilizada para a construção dos classificadores, enquanto os exemplos restantes são utilizados para o teste do SMC.

Para construir os N diferentes classificadores, é necessário inicialmente dividir o conjunto de dados em N subconjuntos de acordo com determinadas características que se deseja para os classificadores e, em seguida, selecionar os exemplos mais representativos

destes subconjuntos para a construção dos classificadores. Neste trabalho, para a formação dos subconjuntos, foi considerado o período do dia (dia e noite) e diferentes faixas de temperatura em que os exemplos foram observados. Estas variáveis foram consideradas isoladas e em conjunto. As estratégias utilizadas para a escolha dos exemplos mais representativos de cada subconjunto dos dados são discutidas a seguir.

Dado N subconjuntos de exemplos preliminarmente selecionados do conjunto de dados D, é necessário escolher apenas uma porcentagem reduzida de exemplos (L) para a construção dos classificadores e utilizar o restante dos exemplos (T) para o teste do sistema, de modo que $D = L \cup T$. Para a escolha dos L exemplos, diferentes técnicas podem ser utilizadas. A princípio, a escolha de exemplos mais representativos de cada subconjunto leva à classificadores com melhor poder preditivo. Esta suposição é confirmada neste trabalho a partir da proposta e avaliação de três técnicas para a escolha dos exemplos:

1. Aleatória: os exemplos L são escolhidos aleatoriamente de cada subconjunto dos dados, enquanto os exemplos não selecionados (T) são utilizados para o teste dos classificadores;

2. k-Medoids: é executado o algoritmo de agrupamento k-Medoids com o objetivo de encontrar L grupos. Assim, os exemplos mais próximos do centro de cada grupo encontrado pelo algoritmo (*medoid*), são escolhidos para a formação do conjunto L e construção dos classificadores, enquanto os exemplos não selecionados (T) são utilizados na etapa de teste;

3. k-Médias: é executado o algoritmo de agrupamento k-Médias com o objetivo de encontrar L grupos. Neste caso, cada grupo encontrado possui um exemplo gerado artificialmente que representa a média dos valores dos exemplos do grupo (centroide). Para a formação do conjunto L, são escolhidos todos os centroides calculados pelo algoritmo k-Médias. Como estes exemplos são gerados artificialmente, não são selecionados exemplos de D para a formação do conjunto L. Assim, o conjunto de teste T é composto por todos os exemplos do conjunto de dados D.

Para a comparação direta dos Sistemas com Múltiplos Classificadores propostos neste trabalho com a configuração Estático, anteriormente apresentada, é considerado que o conjunto L possui a mesma quantidade de exemplos da configuração Estático para a construção dos N classificadores do SMC. Especificamente, são utilizados 15% dos exemplos do conjunto de dados ou 11.500 exemplos.

A primeira proposta de SMC considera o uso de somente dois classificadores, selecionados de acordo com o período do dia da ocorrência do evento. Assim, dado um exemplo de teste a ser classificado, o SMC escolhe um dos dois classificadores para realizar a predição com base na árvore de decisão apresentada na Figura 6.18.

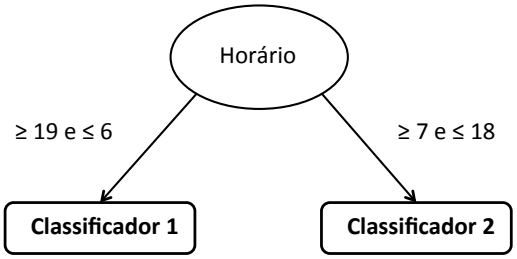


Figura 6.18: Árvore de decisão para a escolha do classificador do SMC de acordo com o horário do exemplo de teste.

Os resultados de acurácia obtidos pelo SMC que considera somente o horário do dia para a escolha do classificador são apresentados na Figura 6.19. Na figura, são apresentados os resultados ao longo do tempo da configuração Estático e do SMC a partir do uso das três diferentes técnicas propostas para a seleção dos exemplos e construção dos classificadores (aleatória, k-Medoids e k-Médias). É possível observar a superioridade dos resultados obtidos pelo SMC em relação à configuração Estático quando utilizadas as técnicas k-Medoids e k-Médias para a seleção dos exemplos.

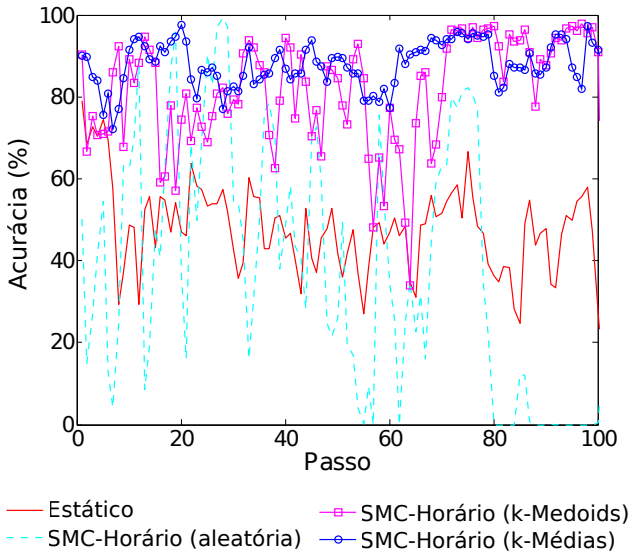


Figura 6.19: Acurácia ao longo do tempo obtida pelo SMC que considera somente o horário do dia para a escolha do classificador na identificação de espécies de insetos em um ambiente com mudanças. Acurácia média do SMC-Horário (aleatória) = $38,02\%$, SMC-Horário (k-Medoids) = $81,73\%$ e SMC-Horário (k-Médias) = $87,90\%$.

A segunda proposta de SMC possui 8 diferentes classificadores e considera diferentes faixas de temperatura para a escolha do classificador durante a etapa de teste. Na Figura 6.20 são apresentadas as faixas de valores consideradas para a escolha do classificador de acordo com o valor de temperatura observado no exemplo de teste. Estes valores foram definidos a partir da análise dos dados.

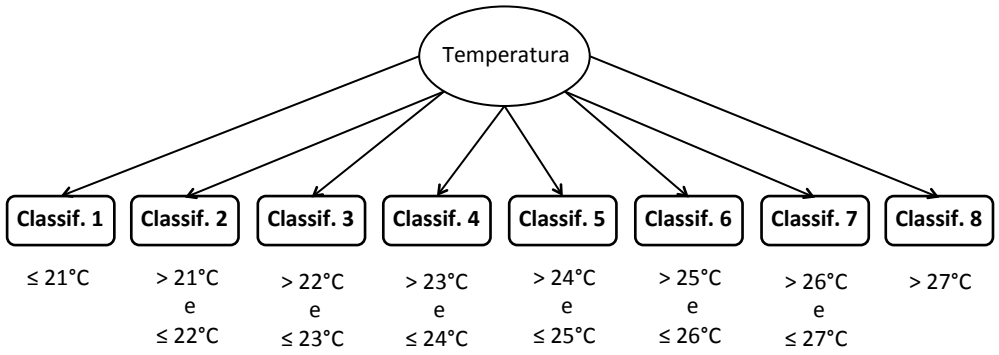


Figura 6.20: Árvore de decisão para a escolha do classificador do SMC de acordo com a temperatura registrada pelo exemplo de teste.

Os resultados de acurácia obtidos pelo SMC que considera somente a temperatura para a escolha do classificador são apresentados na Figura 6.21. Na figura, é possível observar que os resultados são superiores ao SMC que considera somente a hora do dia para a escolha do classificador.

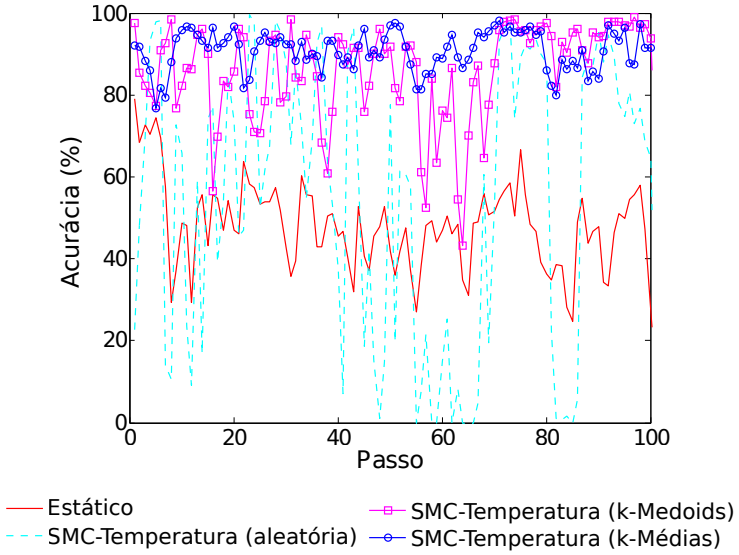


Figura 6.21: Acurácia obtida pelo SMC que considera somente a temperatura para a escolha do classificador. Acurácia média do SMC-Temperatura (aleatória) = $54,69\%$, SMC-Temperatura (k-Medoids) = $85,97\%$ e SMC-Temperatura (k-Médias) = $90,79\%$.

Por último, é proposto um SMC com 14 diferentes classificadores que considera tanto o horário do dia como a temperatura para a escolha do classificador. Desse modo, busca-se unir as vantagens dos dois SMC anteriormente apresentados em uma única proposta. Embora seja possível combinar as respostas isoladas do SMC-Horário e SMC-Temperatura para a tomada de decisão na classificação, seria necessário a definição de um critério de desempate para os casos em que os dois SMC divergissem. Por este motivo, optou-se pela construção de um novo SMC que considera ambas as variáveis.

Para este último SMC, as decisões para a escolha do classificador, dado um exemplo de teste, são apresentadas na Figura 6.22. Neste caso, pode-se observar que são consideradas diferentes faixas de temperatura de acordo com o período do dia (tarde ou noite). Isto se deve a ausência de algumas faixas de valores de temperatura em determinadas horas do dia durante a coleta dos dados.

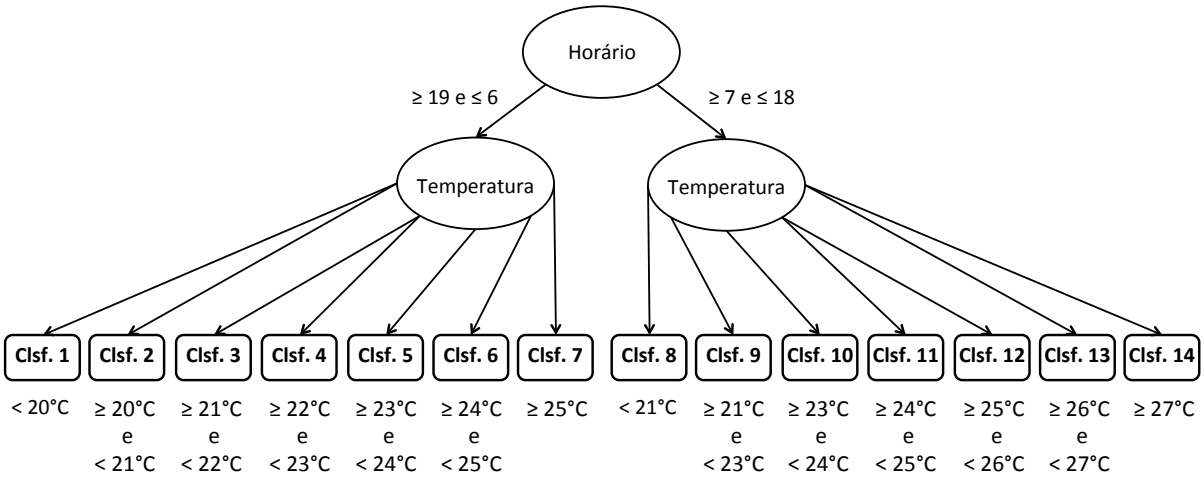


Figura 6.22: Árvore de decisão para a escolha do classificador do SMC de acordo com o horário do dia e temperatura registrados pelo exemplo de teste.

Os resultados de acurácia obtidos pelo SMC que considera tanto o horário do dia como a temperatura para a escolha do classificador são apresentados na Figura 6.23. Nota-se que os resultados superam as duas propostas anteriores.

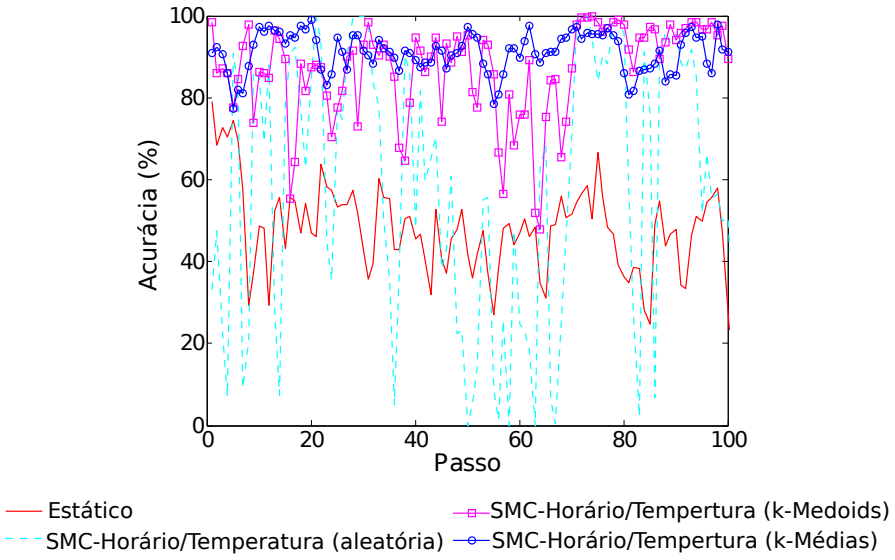


Figura 6.23: Acurácia obtida pelo SMC que considera o horário e temperatura para a escolha do classificador. SMC-Horário-Temperatura (aleatória) = 58, 77%, SMC-Horário-Temperatura (k-Medoids) = 86, 85% e SMC-Horário-Temperatura (k-Médias) = 91, 02%.

A partir da análise dos resultados obtidos pelos três SMC propostos, é possível constatar a importância de uma técnica adequada para a escolha dos exemplos que serão utilizados para a construção dos classificadores. Nas três propostas, a técnica de escolha aleatória se mostrou instável na maior parte do tempo e até mesmo inferior à configuração Estático no caso do SMC que considera somente o horário do dia. Por outro lado, a técnica baseada no algoritmo de agrupamento k-Médias apresentou os melhores resultados nos três Sistemas com Múltiplos Classificadores propostos, sendo a mais adequada dentre as técnicas avaliadas. Os resultados de acurácia média ao longo do tempo dos três SMC são resumidos na Tabela 6.3.

Tabela 6.3: Média de acurácia ao longo do tempo obtida pelas três propostas de Sistemas com Múltiplos Classificadores. Comparado com a configuração Estático ($48,09\%$), o melhor SMC apresenta um resultado significativamente superior de $91,02\%$ de acurácia.

	Seleção dos exemplos		
	Aleatória	k-Medoids	k-Médias
SMC-Horário	38,02	81,73	87,90
SMC-Temperatura	54,69	85,97	90,79
SMC-Horário-Temperatura	58,77	86,85	91,02

Para a melhor visualização da distribuição dos erros cometidos, na Tabela 6.4 é apresentada a matriz de confusão obtida pelo SMC que considera tanto o horário como a temperatura para a escolha do classificador. Serão apresentados somente os valores obtidos pelo SMC com a técnica k-Médias devido aos resultados superiores. Na tabela, também é possível consultar o valor de acurácia obtida para cada uma das classes individualmente.

Tabela 6.4: Matriz de confusão obtida pela execução do SMC que considera tanto o período do dia como a temperatura para a escolha do classificador e a técnica baseada em k-Médias para a seleção de exemplos.

Real	**Predito**				
	Ae. aegypti ♀	***Ae. aegypti*** ♂	***Ae. albopictus*** ♀	***Ae. albopictus*** ♂	***Cx. quinq.*** ♀♂
Ae. aegypti ♀	10194	66	385	32	285
Ae. aegypti ♂	61	18400	17	1874	257
Ae. albopictus ♀	346	31	6653	18	186
Ae. albopictus ♂	46	2202	8	9292	139
Cx. quinq. ♀♂	213	177	106	78	21653
Acurácia (%)	92,99	89,28	91,97	79,51	97,42

A partir da análise dos erros apresentados na Tabela 6.4, pode-se observar a alta eficiência do SMC para a identificação da maior parte das espécies. A menor acurácia do SMC é na identificação das espécies macho de *Ae. albopictus* ($79,51\%$), no qual o sistema

frequentemente atribui incorretamente classificações para a espécie *Ae. aegypti* (macho). Dado que a maior concentração dos erros se localiza entre insetos de mesmo sexo, tais erros não constituem um sério problema prático já que frequentemente se está interessado na captura e contagem de insetos fêmeas.

Dado que as fêmeas tanto da espécie *Ae. aegypti* e *Ae. albopictus* são responsáveis pela transmissão do vírus da dengue, em um cenário prático de uso da armadilha, há o interesse em capturar ambas as espécies do mesmo gênero, com especial interesse para as fêmeas. Assim, este conjunto de dados pode ser transformado em um problema com três classes: *Aedes* (fêmea), *Aedes* (macho) e *Culex quinquefasciatus*. Neste caso, o desempenho dos Sistemas com Múltiplos Classificadores são apresentados na Figura 6.24.

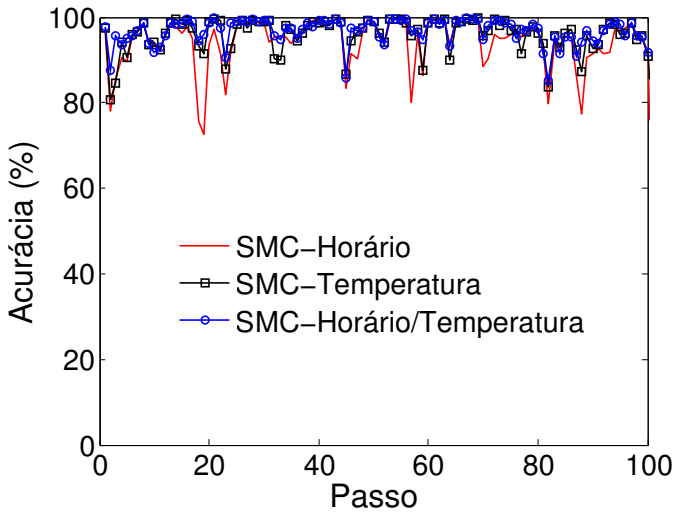


Figura 6.24: Acurácia ao longo do tempo obtida pelos SMC dado um problema com três classes que visa principalmente a identificação do sexo de insetos do gênero *Aedes*. Acurácia média do SMC-Horário = $94,28\%$, SMC-Temperatura = $95,96\%$ e SMC-Horário-Temperatura = $96,72\%$.

Os resultados apresentados na Figura 6.24 se referem as três propostas de SMC com o uso da técnica baseada em agrupamento k-Médias para a seleção dos exemplos. Nos três casos, os SMC apresentaram acurácia média superior a 94% na tarefa de classificação com três classes, em que se busca, principalmente, identificar o sexo de insetos do gênero *Aedes* na presença de machos e fêmeas da espécie *Culex quinquefasciatus*. O SMC que leva em consideração tanto o horário como a temperatura para a escolha do classificador, foi capaz de obter uma acurácia média de $96,72\%$ na tarefa. Os erros deste sistema são apresentados na Tabela 6.5.

A partir dos resultados apresentados na Tabela 6.5, tem-se que é possível realizar a classificação de fêmeas de insetos do gênero *Aedes* com acurácia de $95,77\%$. Este resultado é bastante motivador para o uso prático dos SMC propostos, na armadilha para a captura de insetos transmissores de doenças em um cenário real com mudanças ambientais. É interessante destacar que este resultado foi alcançado mesmo na presença de insetos da

Tabela 6.5: Matriz de confusão obtida pela execução do SMC que considera tanto o período do dia como a temperatura para a escolha do classificador e a técnica baseada em k-Médias para a seleção de exemplos, considerando um problema com três classes.

Real	**Predito**		
	Aedes ♀	***Aedes*** ♂	***Cx. quinq.*** ♀♂
Aedes ♀	14684	144	504
Aedes ♂	157	26655	392
Cx. quinq. ♀♂	447	365	17902
Acurácia (%)	95,77	97,98	95,66

espécie *Culex quinquefasciatus* de ambos os sexos. Esta espécie de mosquito urbano é a mais predominante no Brasil, sendo frequentemente confundido pelas pessoas como do gênero *Aedes* e com alta probabilidade de ser atraído pela armadilha.

6.6 Tempo computacional

Em relação ao tempo computacional da abordagem de Sistema com Múltiplos Classificadores proposta, duas etapas independentes devem ser consideradas: i) o tempo para a seleção dos exemplos de treino que serão utilizados pelos múltiplos classificadores, dadas as estratégias aleatória, k-Medoids e k-Médias e ii) o tempo de classificação dos exemplos de teste.

Na Tabela 6.6 é apresentado o tempo computacional (em minutos) gasto pelos diferentes SMC propostos, tanto para a seleção dos exemplos para a construção dos múltiplos classificadores, como para a classificação dos exemplos de teste. Os tempos apresentados na tabela representam a média de 10 execuções de cada SMC em um computador com processador de 2,4 GHz e 8 GB de memória. Todos os algoritmos foram implementados em MATLAB.

Tabela 6.6: Tempo computacional (em minutos) das diferentes estratégias do Sistema com Múltiplos Classificadores.

Estratégia	**Seleção dos exemplos de treino**	**Classificação dos exemplos de teste**
SMC-Horário (aleatória)	0,002	3,09
SMC-Horário (k-Medoids)	0,619	3,30
SMC-Horário (k-Médias)	23,88	3,66
SMC-Temperatura (aleatória)	0,002	1,81
SMC-Temperatura (k-Medoids)	0,169	1,82
SMC-Temperatura (k-Médias)	4,222	2,16
SMC-Horário-Temperatura (aleatória)	0,001	1,47
SMC-Horário-Temperatura (k-Medoids)	0,109	1,48
SMC-Horário-Temperatura (k-Médias)	2,207	1,76

Pode-se observar nos tempos apresentados na Tabela 6.6, que para a seleção dos exemplos de treino, a abordagem SMC-Temperatura com o algoritmo k-Médias é a que apresenta o pior tempo computacional com aproximadamente 24 minutos. Este tempo se deve ao custo de executar o algoritmo de agrupamento k-Médias em dois grandes subconjuntos de dados com aproximadamente 35 mil exemplos. Além disso, em cada subconjunto são calculados mais de 5 mil grupos. No caso da abordagem que apresenta os melhores resultados em termos de acurácia de classificação (SMC-Horário-Temperatura com o algoritmo k-Médias), este custo é de somente 2,2 minutos. Neste caso, o baixo tempo computacional se deve ao tamanho reduzido dos subconjuntos em que se selecionam os exemplos para o treinamento dos classificadores. De todo modo, é importante destacar que esta etapa é realizada uma única vez para a construção dos classificadores. Assim, mesmo no pior caso em que foram gastos 23 minutos, o tempo computacional não representa um problema.

Em relação a classificação dos exemplos de teste, observou-se um tempo computacional bastante reduzido em todas as abordagens. Este tempo variou entre 1,47 e 3,66 minutos. No caso da abordagem com o melhores resultados de acurácia, o tempo foi de 1,76 minutos. Neste caso, tem-se um tempo médio de 1,5 milissegundo para classificar cada exemplo de teste. Este tempo é suficientemente eficiente para a realização de classificações em tempo real.

6.7 Considerações Finais

Neste capítulo, foi discutido o problema de classificação automática de insetos por meio do sensor óptico em um ambiente com mudanças em variáveis ambientais, como temperatura e umidade. Estes fatores são responsáveis por alterar o comportamento dos insetos, afetando diretamente nas distribuições dos dados. Desse modo, para que a aplicação mantenha a taxa de acerto estável ao longo do tempo em uma situação real de uso, se fez necessária a proposta de abordagens de classificação capazes de se adaptar a mudanças nas condições ambientais.

Para a avaliação das soluções apresentadas, preliminarmente foi necessária a construção de um conjunto de dados de insetos com variações ambientais. Neste conjunto de dados, foram consideradas as espécies *Aedes aegypti, Aedes albopictus* e *Culex quinquefasciatus*. Os mosquitos do gênero *Aedes* foram separados por sexo, formando um fluxo de dados com cinco classes. Durante a etapa de coleta dos dados, variou-se a temperatura por meio de um aparelho condicionador de ar e se controlou a umidade relativa do ar por meio de um aparelho umidificador. A partir da análise dos dados, foi possível observar a influência da temperatura na frequência de batida de asas dos insetos de todas as espécies consideradas. Nota-se que a frequência de batida de asas de todas as espécies aumenta com maiores valores de temperatura. Ainda, observou-se que as fêmeas dos insetos são mais influenciadas por esta variável do que os machos. Pelo melhor de nosso conheci-

mento, este é o primeiro conjunto de dados real com mudanças de conceito induzidas artificialmente em que é possível observar claramente o impacto de fatores externos nos atributos que descrevem os dados.

Foram apresentadas três diferentes propostas de Sistemas com Múltiplos Classificadores para lidar com fluxo de dados não estacionários dado o problema de identificação automática de insetos. Estes sistemas se baseiam na estratégia de seleção dinâmica de classificadores. Neste tipo de estratégia, a escolha do classificador é realizada durante a etapa de classificação e depende essencialmente do exemplo de teste a ser processado. Assim, o classificador mais adequado e previamente induzido pelo sistema, é selecionado no exato momento da classificação do exemplo. Os classificadores que compõem os SMC propostos foram construídos com base nas duas principais variáveis responsáveis por alterar os dados e identificadas na etapa de análise: horário do dia e temperatura. A primeira variável é responsável por alterar a probablidade *a priori* da ocorrência de determinada espécie, dado o ritmo circadiano observado. A segunda variável é responsável por alterar diretamente a distribuição dos dados. Na construção dos SMC, estas duas variáveis foram consideradas isoladas e em conjunto.

A partir do conjunto de dados com cinco classes, foi possível alcançar uma acurácia média de $91,02\%$ ao utilizar o SMC que considera tanto o horário como a temperatura observada no exemplo de teste para a seleção do classificador mais adequado. Dado que as fêmeas tanto da espécie *Ae. aegypti* e *Ae. albopictus* são responsáveis pela transmissão do vírus da dengue, foi considerado um segundo cenário com três classes: *Aedes* (fêmea), *Aedes* (macho) e *Culex quinquefasciatus*. Neste caso, o SMC que leva em consideração tanto o horário como a temperatura para a escolha do classificador, foi capaz de obter uma acurácia média de $96,72\%$ na tarefa de classificação de três classes. Em específico, a acurácia de classificação do *Aedes* (fêmea) foi de $95,77\%$. Estes resultados são bastante motivadores para o uso prático de tal sistema no sensor em conjunto com armadilhas capazes de contar e capturar insetos transmissores de doenças como a dengue.

É importante destacar que tais resultados foram obtidos com o algoritmo 1NN, de modo que outros algoritmos podem apresentar resultados superiores. Entretanto, a escolha do algoritmo 1NN na avaliação das propostas se justifica pela facilidade de atualização dos classificadores do SMC, dado que o algoritmo 1NN é naturalmente incremental. Em trabalhos futuros, pretende-se explorar abordagens de atualização destes classificadores durante a etapa de teste. Por exemplo, pode-se remover exemplos do conjunto de treinamento que raramente são solicitados durante as classificações ou incluir exemplos de teste que apresentam características ainda não consideradas pelo sistema, como determinadas faixas de temperatura.

Capítulo

7

Conclusões

7.1 Considerações Iniciais

Neste capítulo, são apresentadas as conclusões desta tese de doutorado. Na Seção 7.2, são apresentadas as principais contribuições; na Seção 7.3, são discutidas algumas limitações das propostas realizadas neste trabalho; na Seção 7.4 são apresentados direcionamentos para trabalhos futuros; e, por fim, na Seção 7.5, são listadas as publicações geradas durante o período de desenvolvimento desta tese.

7.2 Principais Contribuições

Na tarefa de classificação de fluxo de dados, é necessário realizar constantes atualizações no modelo de classificação para que não ocorram degradações na taxa de acerto do classificador, devido à ocorrência de mudanças de conceito nos dados que descrevem as classes do problema ao longo do tempo. Para isso, a maior parte dos algoritmos da literatura assume que é possível obter o rótulo correto de cada exemplo do fluxo antes da chegada do próximo exemplo. Assim, é possível monitorar indicadores de desempenho do classificador ou comparar se há diferenças significativas entre as distribuições dos dados ao longo do tempo para a detecção explícita de mudanças e, então, utilizar os dados rotulados recebidos mais recentemente para a atualização do modelo. Conforme discutido nesta tese, devido a inúmeros fatores como altos custos envolvidos no processo de rotulação, dependência de um especialista para a rotulação, falhas no processo de transmissão dos rótulos, alta taxa de chegada de dados para a requisição de seus rótulos ou mesmo devido às características do processo gerador dos dados, as informações a respeito dos rótulos dos

exemplos nem sempre podem ser obtidas para a maior parte dos dados, ou são obtidas após um considerável atraso de tempo em uma série de aplicações reais. Este atraso de tempo entre a classificação de um exemplo por um algoritmo de aprendizado de máquina e a disponibilização de seu respectivo rótulo correto pelo ambiente é referido neste trabalho como latência.

Argumenta-se neste trabalho que a restrição de que todos os exemplos tenham os seus rótulos corretos revelados logo após a classificação do exemplo é a principal restrição existente em diversos algoritmos de classificação de fluxo de dados, como *Very Fast Decision Tree* – VFDT (Domingos e Hulten, 2000), *Concept-adapting Very Fast Decision Trees* – CVFT (Hulten et al., 2001) e de métodos de detecção de mudanças como ADWIN e DDM. Com o objetivo de apresentar a importância da latência em algoritmos tradicionais de classificação de fluxo de dados, foi realizada uma avaliação com diferentes algoritmos da literatura ao classificar exemplos de conjuntos de dados amplamente avaliados considerando diferentes valores de latência. A partir desta análise experimental, foi possível constatar o impacto da latência em métodos de classificação de fluxo de dados considerados estado-da-arte. Nota-se que a maior parte das propostas de novos métodos de classificação de fluxo de dados não estacionários, independente da abordagem utilizada, tem negligenciado a latência para o recebimento dos rótulos corretos. Assim, este trabalho busca alertar a necessidade de considerar este importante fator para que os algoritmos de classificação de fluxo de dados possam ser amplamente utilizados na prática em problemas reais.

No cenário de latência extrema onde se inserem aplicações em que há total ausência na disponibilização de exemplos rotulados após a etapa de classificação, foram propostas e avaliadas duas soluções denominadas *Stream Classification Algorithm Guided by Clustering – SCARGC* e *Micro-Cluster Classification – MClassification*. A primeira proposta SCARGC se baseia na ideia de realizar sucessivas etapas de agrupamento dos dados ao longo do tempo para se adaptar a possíveis mudanças de conceito em problemas de classificação sob condições de latência extrema. O algoritmo é intuitivo, eficiente e possui baixa complexidade de implementação. A segunda abordagem proposta neste trabalho utiliza o conceito de *Cluster Feature*, originalmente utilizado pelo algoritmo de agrupamento para grandes bases de dados BIRCH e posteriormente denominado *Micro-Cluster* no algoritmo de agrupamento de fluxo de dados *CluStream*. Embora seja eficiente e apropriada para problemas de fluxo de dados em geral, nota-se que a representação de *Micro-Clusters* tem sido essencialmente utilizada somente em problemas de agrupamento. Assim, neste trabalho é proposto o uso da representação condensada *Micro-Clusters* para a tarefa de classificação de fluxo de dados não estacionários. Para isso, os *Micro-Clusters* passam a armazenar informações sobre a classe a qual pertencem e a cada novo exemplo classificado é realizada uma etapa de atualização na posição dos *Micro-Clusters* representantes das classes do problema. Para a avaliação dos algoritmos propostos, foram construídos

e compilados um total de 16 conjuntos de dados sintéticos com características e comportamentos específicos que buscou-se avaliar, como grande volume e mudanças de conceito incrementais ao longo do tempo. Estes conjuntos de dados estão publicamente disponibilizados para a comunidade acadêmica e espera-se que estes dados possam contribuir para outros pesquisadores da área na proposta de novas soluções. Além de dados sintéticos, os algoritmos também foram avaliados em 2 conjuntos de dados reais. Em geral, os métodos propostos apresentaram um bom desempenho em acurácia e tempo computacional, com resultados equivalentes ou superiores aos apresentados na literatura.

Este trabalho também buscou contribuir com uma importante aplicação real de identificação automática de espécies de insetos por meio de sensores ópticos. Esta aplicação é um exemplo concreto que necessita realizar a classificação de um fluxo de dados não estacionários em tempo real e sob o cenário de latência extrema. Assim, neste trabalho foram exploradas e avaliadas diferentes técnicas de classificação e atributos descritores de dados aplicados na tarefa de classificação de insetos. A partir das avaliações realizadas neste trabalho, foi possível verificar que os atributos baseados na escala *mel*, MFCC, são bons descritores dos dados e em conjunto com algoritmos de classificação de múltiplas classes ou com uma única classe, são capazes de apresentar resultados satisfatórios de classificação que podem ser utilizados na prática. Ao considerar um cenário com dados sob a presença de mudanças de conceito, é proposta uma abordagem de Sistema com Múltiplos Classificadores em que realiza-se a seleção dinâmica do classificador mais adequado de acordo com variáveis observadas em cada exemplo de teste.

Os resultados alcançados na classificação automática de insetos são motivadores, pois permitem o uso efetivo do sensor óptico em conjunto com armadilhas inteligentes capazes de capturar somente insetos de interesse, libertando espécies benéficas ao meio-ambiente. Um exemplo do uso de tais armadilhas é na captura de insetos da espécie *Aedes aegypti*, responsável pela transmissão de diversas doenças como dengue, febre amarela, Zika vírus e febre Chikungunya. Desse modo, o controle das suas populações é considerado assunto de saúde pública.

7.3 Limitações

Embora a principal limitação dos algoritmos tradicionais da literatura tenha sido removida das propostas apresentadas neste trabalho, que é a disponibilização dos rótulos corretos de todos os exemplos do fluxo sem qualquer atraso de tempo, outras limitações são impostas às abordagens SCARGC e MClassification. Uma das limitações da abordagem utilizada pelo algoritmo SCARGC é a dependência de um algoritmo de agrupamento de dados. Desse modo, o algoritmo de classificação proposto consequentemente herda as características das técnicas de agrupamento. Na avaliação realizada neste trabalho, foi utilizado o algoritmo k-Médias devido a sua simplicidade, eficiência computacional e por

ser amplamente utilizado na literatura. Entretanto, o algoritmo k-Médias supõe que os dados apresentam uma distribuição gaussiana esférica, o que nem sempre é observado em problemas reais. Este problema pode ser minimizado a partir do uso de um número maior de grupos para representar uma classe. Além disso, o algoritmo k-Médias apresenta limitações quando os grupos possuem tamanhos e densidades diferentes. Algumas destas limitações podem ser futuramente sanadas a partir da adaptação do algoritmo SCARGC para o uso de diferentes técnicas de agrupamento.

Outra limitação do algoritmo SCARGC é relativa ao parâmetro k, que define a quantidade de grupos utilizados para a descrição dos dados do problema. Além de ser constante durante toda a execução do algoritmo, nem sempre é trivial encontrar um valor adequado. Em problemas reais, é comum que o número de grupos referentes a uma classe se altere ao longo do tempo. Assim, em trabalhos futuros pretende-se adaptar o algoritmo para que: *i*) encontre automaticamente uma quantidade adequada de grupos para representar as classes do problema; *ii*) seja capaz de realizar comparações de similaridade entre grupos antigos e grupos atuais a fim de encontrar mudanças de conceito, mesmo que a quantidade de grupos seja diferente, e *iii*) sejam utilizadas outras medidas além da similaridade entre os centroides dos grupos.

Visando a remoção do parâmetro k e a restrição da quantidade constante de grupos ao longo do tempo no algoritmo SCARGC, foi apresentado o algoritmo MClassification. Entretanto, a principal limitação do algoritmo é ser suscetível à presença de valores atípicos (*outliers*) e ruído, que podem influenciar negativamente no comportamento do algoritmo. Uma ideia inicial em trabalhos futuros é a atribuição de pesos aos *Micro-Clusters* de acordo com a sua idade ou utilidade e que a criação e deslocamento de *Micro-Clusters* leve em consideração um número maior de exemplos.

Por fim, uma limitação inerente ao problema de classificação de fluxo de dados sob condições de latência extrema, tanto nas abordagens propostas como nos métodos da literatura, é a suposição da ocorrência de mudanças de conceito incrementais. Embora seja muito difícil ou até mesmo impossível lidar com este problema em situações onde as mudanças de conceito sejam abruptas, acredita-se que no futuro, com a melhor consolidação deste problema de pesquisa, seja possível realizar a adaptação dos classificadores sob condições de mudanças graduais, mesmo na ausência dos rótulos corretos dos exemplos do fluxo de dados.

7.4 Trabalhos Futuros

Além da exploração de soluções para as limitações anteriormente apresentadas, também observou-se a necessidade de investigar diferentes problemas em trabalhos futuros, conforme discutido a seguir.

Devido à importância da latência para o recebimento dos rótulos corretos em problemas de classificação de fluxo de dados e o seu impacto em métodos estado-da-arte, pretende-se em trabalhos futuros analisar e propor uma nova medida de desempenho para os algoritmos de classificação que leve em consideração esta característica em conjunto com a porcentagem de dados rotulados recebidos. Desse modo, dado por exemplo dois algoritmos de classificação A e B, em que A apresenta uma acurácia média de classificação de 80% e considera o acesso de 100% dos dados rotulados com latência nula e um algoritmo B que apresenta acurácia de 70% mas tem acesso a somente 20% dos dados rotulados sem nenhum atraso enquanto todos os outros exemplos apresentam latência extrema, a medida de avaliação não deve afirmar que o desempenho de A é superior a B com base somente nos resultados de acurácia. Com isso, esta medida pode ser utilizada para orientar na escolha de algoritmos em problemas reais em que há a característica de latência.

Em geral, tanto em problemas de classificação de fluxo de dados como em lote, é considrado que o classificador tem conhecimento de todas as classes do problema com dados rotulados que contemplam todas as possibilidades. Assim, para cada instância $\vec{x}_i$ a ser classificada, é associado um rótulo de classe $y_i \in Y$, de modo que $Y = \{y_1, y_2, \ldots, y_n\}$ é um conjunto fechado com n classes. Entretanto, em muitos problemas não é possível obter exemplos rotulados para cada possível classe, seja pelo custo envolvido no processo de rotulação destes dados, o desconhecimento de todas as possíveis classes, o surgimento de uma nova classe ou a quantidade elevada de classes possíveis que impossibilita a sua incorporação em um modelo de classificação. Nestes casos, é necessário que o algoritmo de classificação considere que o conjunto de possíveis classes Y é um conjunto aberto com a presença de classes desconhecidas. Caso contrário, as instâncias destas classes desconhecidas serão incorretamente associadas às classes existentes, constituindo falsos positivos. Deste modo, pretende-se explorar em trabalhos futuros o problema de classificação de fluxo de dados não estacionários em um cenário em que o conjunto de classes na etapa de teste é "aberto". Este é um cenário que pode ser útil na aplicação de classificação de insetos quando há o interesse em identificar diferentes espécies mas não há o conhecimento de todas as possíveis espécies que podem estar presentes no local onde será utilizado o sensor.

No caso da proposta do Sistema com Múltiplos Classificadores para a identificação de insetos em ambientes não estacionários, é interessante destacar que após a construção dos classificadores que compõem o sistema, estes classificadores não são mais atualizados. Assim, em trabalhos futuros pretende-se explorar abordagens de atualização destes classificadores durante a etapa de teste. Por exemplo, pode-se remover exemplos do conjunto de treinamento que raramente são solicitados durante as classificações ou incluir exemplos de teste que apresentam características ainda não consideradas pelo sistema, como determinadas faixas de temperatura. Destaca-se, ainda, que os resultados apresentados foram

obtidos com o uso do algoritmo 1NN, que é naturalmente incremental. Assim, a etapa de atualização dos classificadores será facilitada e realizada em menor tempo, se comparada à atualização de um algoritmo não incremental.

Ainda em relação ao sensor identificador de insetos, pretende-se explorar técnicas que consideram o baixo consumo de energia. Ao ser utilizado em campo, estes sensores serão alimentados por baterias, sendo necessário o uso de técnicas eficientes em relação ao consumo. Assim, uma das pretensões de trabalho futuro decorrente desta tese é o desenvolvimento tecnológico e científico de armadilhas inteligentes para a captura de insetos com o uso do sensor óptico, de modo que tais armadilhas sejam avaliadas fora do laboratório e que possam ser comercializadas como um eletrônico de consumo para a população.

7.5 Publicações

As publicações geradas durante o desenvolvimento deste trabalho são listadas a seguir.

- **Souza, V. M. A.**; Silva, D. F.; Gama, J.; Batista, G. E. A. P. A. Data Stream Classification Guided by Clustering on Nonstationary Environments and Extreme Verification Latency. *Proceedings of the SIAM International Conference on Data Mining (SDM)*, 2015, p. 873-881. (Souza et al., 2015c)

- **Souza, V. M. A.**; Silva, D. F.; Batista, G. E. A. P. A.; Gama, J. Classification of Evolving Data Streams with Infinitely Delayed Labels. *Proceedings of the IEEE International Conference on Machine Learning and Applications (ICMLA)*, 2015, p. 214-219. (Souza et al., 2015b)

- **Souza, V. M. A.**; Batista, G. E. A. P. A.; Souza-Filho, N. E. Automatic Classification of Drum Sounds with Indefinite Pitch. *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, 2015, p. 1-8. (Souza et al., 2015a)

- Sousa, C. A. R.; **Souza, V. M. A.**; Batista, G. E. A. P. A. An Experimental Analysis on Time Series Transductive Classification on Graphs. *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, 2015, p. 1-8. (De Sousa et al., 2015)

- Qi, Y.; Cinar, G. T.; **Souza, V. M. A.**; Batista, G. E. A. P. A.; Wang, Y.; Principe, J. C. Effective Insect Recognition Using a Stacked Autoencoder with Maximum Correntropy Criterion. *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, 2015, p. 1-7. (Qi et al., 2015)

- Silva, D. F.; **Souza, V. M. A.**; Batista, G. E. A. P. A. Music Shapelets for Fast Cover Song Recognition. *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, 2015, p. 441-447. (Silva et al., 2015a)

- Tibes, C. M. S.; Cherman, E. A.; **Souza, V. M. A.**; Westin, U. M; Zem-Mascarenhas, S. H.; Évora, Y. D. M. Avaliação de um Aplicativo para Apoio à Decisão no Cuidado de Úlceras por Pressão. *Anais do Congresso Internacional de Informática Educativa (TISE)*, 2015, p. 191-199. (Tibes et al., 2015)

- Silva, D. F.; **Souza, V. M. A.**; Ellis, D. P. W.; Keogh, E. J.; Batista, G. E. A. P. A. Exploring Low Cost Laser Sensors to Identify Flying Insect Species. *Journal of Intelligent & Robotic Systems*, v. 1, p. 1-18, 2015. (Silva et al., 2015b)

- **Souza, V. M. A.**; Silva, D. F.; Batista, G. E. A. P. A. Extracting Texture Features for Time Series Classification. *Proceedings of the International Conference on Pattern Recognition (ICPR)*, 2014, p. 1425-1430. (Souza et al., 2014)

- Sousa, C. A. R.; **Souza, V. M. A.**; Batista, G. E. A. P. A. Time Series Transductive Classification on Imbalanced Data Sets: An Experimental Study. *Proceedings of the International Conference on Pattern Recognition (ICPR)*, 2014, p. 3780-3785. (De Sousa et al., 2014)

- Batista, G. E. A. P. A.; Keogh, E. J.; Tataw, O. M.; **Souza, V. M. A.** CID: an efficient complexity-invariant distance for time series. *Data Mining and Knowledge Discovery*, v. 28(3), p. 634-669, 2014. (Batista et al., 2014)

- **Souza, V. M. A.** ; Silva, D. F.; Batista, G. E. A. P. A. Classification of Data Streams Applied to Insect Recognition: Initial Results. *Proceedings of the Brazilian Conference on Intelligent Systems (BRACIS)*, 2013, p. 76-81. (Souza et al., 2013a)

- **Souza, V. M. A.**; Feltrim, V. D. Uma Investigação sobre Algoritmos de Diferentes Abordagens de Aprendizado Supervisionado na Classificação de Papéis Retóricos em Resumos Científicos. *Proceedings of the Brazilian Symposium in Information and Human Language Technology (STIL)*, 2013, p. 30-39. (Souza e Feltrim, 2013)

- **Souza, V. M. A.**; Silva, D. F.; Garcia, P. R. P.; Batista, G. E. A. P. A. Avaliação de Classificadores para o Reconhecimento Automático de Insetos. *Anais do Encontro Nacional de Inteligência Artificial e Computacional (ENIAC)*, 2013. p. 1-12. (Souza et al., 2013b)

- Silva, D. F.; **Souza, V. M. A.**; Batista, G. E. A. P. A.; Keogh, E. J.; Ellis, D. P. W. Applying Machine Learning and Audio Analysis Techniques to Insect Recognition in Intelligent Traps. *Proceedings of the IEEE International Conference on Machine Learning and Applications (ICMLA)*, 2013, p. 99-104. (Silva et al., 2013c)

- Silva, D. F.; **Souza, V. M. A.**; Batista, G. E. A. P. A. Time Series Classification Using Compression Distance of Recurrence Plots. *Proceedings of the IEEE*

International Conference on Data Mining (ICDM), 2013, p. 687-696. (Silva et al., 2013b)

- Domingues, M. A.; Cherman, E. A.; Nogueira, B. M.; Conrado, M. S.; Rossi, R. G.; Padua, R.; Marcacini, R. M.; **Souza, V. M. A.**; Batista, G. E. A. P. A.; Rezende, S. O. A comparative study of algorithms for recommending given names. *Proceedings of the International Conference on Informatics & Applications (ICIA)*, 2013, p. 66-71. (Domingues et al., 2013)

- Silva, D. F.; **Souza, V. M. A.**; Batista, G. E. A. P. A. A comparative study between MFCC and LSF coefficients in automatic recognition of isolated digits pronounced in Portuguese and English. *Acta Scientiarum Technology*, v. 35, p. 621-628, 2013. (Silva et al., 2013a)

- Silva, D. F.; **Souza, V. M. A.**; Batista, G. E. A. P. A.; Giusti, R. Spoken Digit Recognition in Portuguese Using Line Spectral Frequencies. *Proceedings of the Ibero-American Conference on Artificial Intelligence (IBERAMIA)*, 2012, p. 241-250. (Silva et al., 2012)

Apêndice

A

Outras Contribuições

Além de contribuições nas áreas de fluxo de dados e sensores, especificamente nas tarefas de classificação sob condições de latência extrema para o recebimento dos rótulos corretos e a identificação automática de insetos em ambientes estacionários/não estacionários, também foram realizadas outras contribuições nas áreas de séries temporais, processamento digital de sinais e aprendizado ativo durante o desenvolvimento deste projeto de doutorado. Desse modo, o objetivo deste apêndice é apresentar uma visão geral destas contribuições, que embora não sejam diretamente relacionadas aos objetivos principais desta pesquisa, surgiram devido à exploração dos problemas de pesquisa e à avaliação de soluções em diferentes domínios. Por exemplo, os dados obtidos pelo sensor laser são sinais de áudio de curta duração, semelhantes aos dados processados por outras aplicações, como o reconhecimento de dígitos falados e a classificação de instrumentos de percussão. Do mesmo modo, estes sinais de áudio também podem ser interpretados como séries temporais. Assim, explorou-se tanto a classificação de sinais de áudio em diferentes aplicações como abordagens para a classificação de séries temporais. É importante destacar que a apresentação destas contribuições será realizada no apêndice deste documento devido somente a questões de organização do texto e possui a mesma importância/relevância que o restante desta tese.

Na Seção A.1, são apresentadas três aplicações que envolvem a classificação de sinais de áudio por meio de algoritmos de aprendizado de máquina e atributos provenientes da área de processamento digital de sinais. A primeira aplicação envolve a classificação automática de pratos de bateria (instrumento de percussão) e a segunda envolve a classificação de dígitos falados em inglês ou português. A terceira aplicação envolve o reconhecimento de músicas *cover* por meio da adaptação da técnica de *shapelet*, amplamente utilizada na

área de séries temporais. Na Seção A.2, são apresentadas três contribuições em relação à classificação de séries temporais. A primeira contribuição é a proposta de uma medida de distância invariante à complexidade e a segunda contribuição é uma medida baseada em compressão. Por fim, é apresentado o uso de descritores de textura extraídos de gráficos de recorrência para a tarefa de classificação. Na Seção A.3, é apresentada uma avaliação sobre algoritmos baseados em agrupamento e em medidas de centralidade extraídas de grafos para a rotulação de dados de treino por meio de aprendizado ativo para o uso em problemas de classificação.

A.1 Classificação de Sinais de Áudio

A.1.1 Pratos de Bateria

Transcrições musicais em partituras, cifras ou tablaturas são populares entre músicos profissionais, amadores ou iniciantes. Entretanto, a construção destas transcrições requer conhecimento e tempo de pessoas dispostas a realizar esta tarefa para uma determinada música. Por isso, a identificação e classificação automática de particularidades de instrumentos musicais constituem um importante passo para a transcrição musical automática. A tarefa de classificação automática também é interessante para outros profissionais como produtores musicais ou na análise de conteúdo em músicas.

A maior parte dos trabalhos da literatura em aprendizado de máquina e processamento digital de sinais se destina a classificação de instrumentos de corda ou sopro, enquanto somente uma pequena parcela de trabalhos se destina a classificação de instrumentos de percussão (uma ampla revisão é apresentada em Herrera Boyer et al. (2003)). A principal diferença entre instrumentos de percussão e outros tipos de instrumentos é que os instrumentos de percussão produzem sons com altura (*pitch*) indefinida. Assim, não é possível definir com precisão qual a frequência fundamental de um som emitido por uma bateria. Por este motivo, a classificação automática dos sons emitidos por este tipo de instrumento é mais desafiador.

Ao longo dos últimos anos, diversos estudos foram realizados em direção a classificação automática de instrumentos de percussão. Alguns exemplos são Herrera et al. (2002); Paulus (2006); Yoshii et al. (2007); Scholler e Purwins (2011). Entretanto, o objetivo destes trabalhos é distinguir diferentes partes de uma bateria, como bumbo (*bass drum*), caixa (*snare drum*), tons, chimbal (*hi-hat*) e pratos. Embora sejam esforços importantes, nota-se um problema neste tipo de abordagem: existem diferentes tipos de pratos de bateria e todos são classificados em uma única categoria por estes trabalhos.

Visando mitigar a lacuna apresentada por outros trabalhos, em Souza et al. (2015a) é proposta a classificação de pratos de bateria em dois níveis. No primeiro nível realiza-se a classificação do tipo do prato e no segundo nível identifica-se como o som foi gerado de

acordo com a região em que o prato foi atacado ou o movimento realizado para a geração do som emitido. Até a realização deste trabalho, não tínhamos conhecimento de nenhum outro trabalho na literatura que abordou o problema com este nível de especificidade. Assim, o objetivo deste trabalho foi auxiliar um importante passo para a construção de sistemas de transcrição automática de bateria com informações mais detalhadas sobre os pratos.

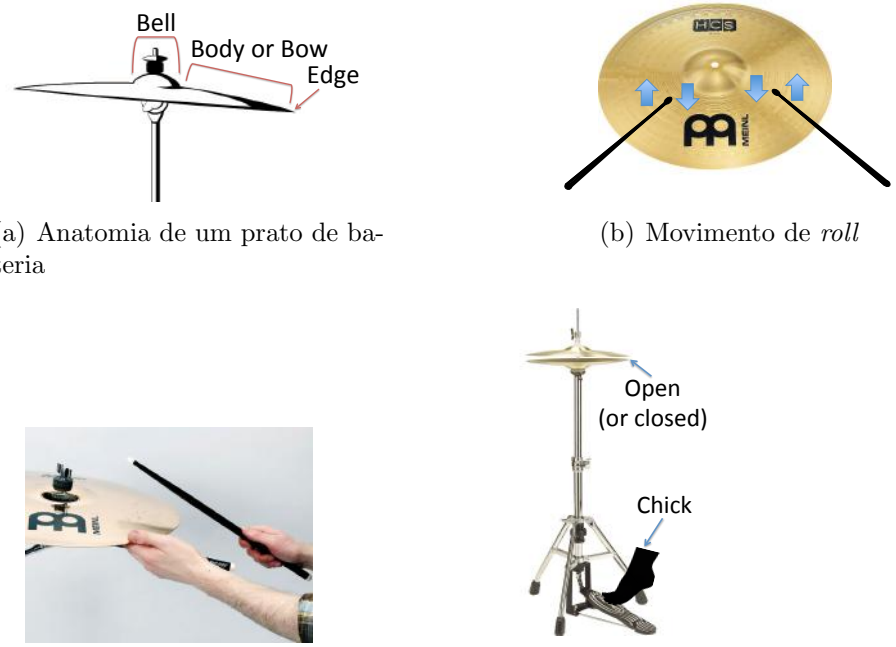
Inicialmente foram coletados 1052 exemplos de sons de pratos de bateria de 18 diferentes séries produzidas pela *Paiste* e obtidos no *website*[1] da empresa. Cada série possui uma especificação diferente em relação a sua construção (tamanho, material, textura, curvatura, peso, etc.), responsável por alterar as características dos sons emitidos. Após a coleta, estes dados foram divididos em cinco categorias principais de acordo com o tipo do prato: *i)* China; *ii)* Crash; *iii)* Hi-hat; *iv)* Ride e *v)* Splash. A Figura A.1 ilustra estes tipos de prato em um *kit* de bateria convencional.

Figura A.1: Exemplo ilustrativo de uma bateria com 5 tipos diferentes de pratos.

A partir das cinco categorias principais, os sons de pratos de bateria foram novamente divididos em outras subcategorias de acordo com a região em que o prato foi atacado ou o movimento realizado pelo baterista para a geração do som. Por exemplo, os pratos do tipo China, Crash, Ride e Splash podem ser atacados na região da borda, chamada de *Edge*. Além desta região, o prato Ride também pode ser atacado em sua região mais central, chamada de *Bell* ou na região entre a borda e o centro, chamada de *Body*. Para o prato do tipo China, também foi considerado o movimento de *Roll*, em que o baterista ataca o prato rapidamente sucessivas vezes na região da borda. Para o prato Splash, foi considerado o movimento de abafamento do prato pela mão após ser atacado, chamado de *Choke*. Para

[1] http://www.paiste.com/

o chimbal (*hi-hat*), foram consideradas três subcategorias: *Chick*, quando o chimbal é acionado pelo pé do baterista; *Closed* quando o chimbal é atacado na região da borda e ambos os pratos estão juntos; e *Open* quando o chimbal é atacado na região da borda e ambos os pratos estão levemente separados. A Figura A.2 ilustra estas subcategorias.



(a) Anatomia de um prato de bateria

(b) Movimento de *roll*

(c) Movimento de *choke*

(d) Chimbal

Figura A.2: Ilustração das subcategorias consideradas de acordo com a região em que o prato é atingido ou o movimento realizado pelo baterista para a geração do som.

A distribuição dos dados de sons de pratos de bateria em cada categoria e subcategoria é apresentada na Tabela A.1.

Tabela A.1: Distribuição dos dados de sons de pratos de bateria.

Categoria	**Subcategoria**	**# Exemplos por subcategoria**	**# Exemplos por categoria (%)**	**Total**
China	Edge	51	99 (9,41)	1052
	Roll	48		
Crash	Edge	152	303 (28,80)	
	Roll	151		
Hi-hat	Chick	75	227 (21,58)	
	Closed	77		
	Open	75		
Ride	Bell	107	344 (32,70)	
	Body	113		
	Edge	124		
Splash	Choke	36	79 (7,51)	
	Edge	43		

A partir deste conjunto de dados[2], foi investigado o desempenho dos coeficientes LSF, MFCC, LFC e atributos provenientes da representação temporal e espectral dos sinais em conjunto com cinco algoritmos de classificação: Naive Bayes (NB), C4.5, Random Forest (RF), *k*-Nearest Neighbor (KNN) e Máquina de Vetores de Suporte (SVM). Para o primeiro nível de classificação em que busca-se classificar os cinco diferentes tipos de pratos de bateria, os resultados são apresentados na Tabela A.2. Os valores apresentados referem-se a acurácia média obtida em um experimento com 10 execuções e validação cruzada ou 10x10-*fold cross-validation*. Os melhores resultados obtidos por cada algoritmo são apresentados sublinhados e os melhores resultados obtidos por um conjunto de atributos são apresentados em **negrito**. Pode-se observar que o melhor resultado é obtido pelo algoritmo SVM em conjunto com os coeficientes LSF, em que obteve-se uma acurácia média de classificação de 96,59% para as 5 classes principais.

Tabela A.2: Resultados de acurácia média na classificação de tipo de prato de bateria.

Atributo	**NB**	**C4.5**	**RF**	**KNN**	**SVM**
LSF	80,90	88,05	93,08	93,59	**96,59**
MFCC	82,68	85,41	90,09	89,33	**94,01**
LFC	73,10	80,41	86,94	85,65	**91,41**
Temporal	70,28	86,51	**89,95**	82,27	85,85
Espectral	66,38	86,48	**89,38**	79,57	87,54

No segundo experimento, buscou-se obter uma classificação mais informativa. Além de classificar o tipo do prato, também foi classificado como o som foi gerado, de acordo com as subcategorias anteriormente apresentadas na Tabela A.1. Os resultados obtidos nesta etapa são apresentados na Tabela A.3.

Tabela A.3: Resultados de acurácia média na classificação de características específicas nos sons de pratos de bateria.

Atributo	**NB**	**C4.5**	**RF**	**KNN**	**SVM**
LSF	78,45	71,53	80,09	74,63	**86,49**
MFCC	65,23	63,90	69,72	56,40	**74,93**
LFC	56,96	57,55	66,53	58,67	**72,91**
Temporal	68,07	82,17	**86,02**	72,93	81,54
Espectral	78,59	80,77	**85,16**	73,18	84,36

Novamente, observa-se na Tabela A.3 que o melhor resultado é obtido com o uso do algoritmo SVM e os coeficientes LSF com acurácia de 86,49%. Entretanto, nota-se uma redução de aproximadamente 10% em relação a primeira etapa de classificação. A partir da audição dos sinais, tem-se a intuição de que características simples obtidas a partir da representação temporal dos dados (como a duração) são essenciais para a classificação do movimento realizado pelo baterista ao atacar um prato. Desse modo, visando melhorar o desempenho de classificação, avaliou-se o comportamento dos classificadores ao

[2]O conjunto de dados construído neste trabalho (tanto o áudio como os atributos extraídos) está livremente disponibilizado no endereço web http://sites.labic.icmc.usp.br/vsouza/paiste_dataset/

combinar os coeficientes LSF e atributos temporais. Os resultados deste experimento são apresentados na Tabela A.4.

Tabela A.4: Resultados de acurácia média na classificação de características específicas nos sons de pratos de bateria considerando o uso dos coeficientes LSF em conjunto com atributos extraídos da representação temporal.

Atributo	**NB**	**C4.5**	**RF**	**KNN**	**SVM**
LSF + Temporal	85,68	87,36	89,36	83,77	**91,54**
Aumento	7,09	5,19	3,34	9,14	5,05

Pode-se observar na Tabela A.4 que o uso dos atributos extraídos da representação temporal do sinal em conjunto com os coeficientes LSF contribuíram significativamente para a melhora do desempenho dos classificadores. Em específico, o algoritmo SVM obteve uma acurácia média de 91,54%, aproximadamente 5% a mais do que o uso somente dos coeficientes LSF, o que representa mais de 53 exemplos corretamente classificados. Um exemplo da matriz de confusão calculada a partir de uma das execuções realizadas pelo classificador (SVM) é apresentado na Tabela A.5. Nesta tabela, também são apresentados os valores das medidas *Precision, Recall* e *F-Measure* para cada uma das 12 classes consideradas.

Tabela A.5: Exemplo de matriz de confusão obtida pelo algoritmo SVM (*kernel* RBF, $c = 3.0$ and $\gamma = 0.01$) com os coeficientes LSF em conjunto com atributos extraídos da representação temporal para a classificação de características específicas nos sons de pratos de bateria.

Real	**Predito**											
	CE	CR	CrE	CrR	HCh	HCl	HOp	RBe	RBo	REd	SC	SE
China-Edge (CE)	44	7										
China-Roll (CR)	12	36										
Crash-Edge (CrE)	1		132	9						10		
Crash-Roll (CrR)			17	132						1	1	
Hi-hat-Chick (HCh)					72	3						
Hi-hat-Closed (HCl)					2	75						
Hi-hat-Open (HOp)							74				1	
Ride-Bell (RBe)								107				
Ride-Body (RBo)									113			
Ride-Edge (REd)			15	1				1	1	106		
Splash-Choke (SC)											34	2
Splash-Edge (SE)	1		4									38
Precision	0,76	0,84	0,79	0,93	0,97	0,96	1,00	0,99	0,99	0,91	0,94	0,95
Recall	0,86	0,75	0,87	0,87	0,96	0,97	0,99	1,00	1,00	0,85	0,94	0,88
F-Measure	0,81	0,79	0,82	0,90	0,97	0,97	0,99	0,99	0,99	0,88	0,94	0,92

Dados os resultados apresentados, tem-se que é possível discriminar cinco diferentes tipos de pratos de bateria com uma acurácia de 96,59% e a maneira em que um prato é atacado (de acordo com 12 diferentes classes) com uma acurácia de 91,54%.

As principais contribuições deste trabalho foram em duas direções: construir e disponibilizar um conjunto de dados de sons de bateria para a comunidade acadêmica interessada

em pesquisar sobre o processamento de sinais musicais com *pitch* indefinido; mostrar para a comunidade acadêmica que os coeficientes LSF e atributos extraídos da representação temporal do sinais são bons descritores para este tipo de dado, superando atributos mais populares na literatura como MFCC.

A.1.2 Dígitos Falados

A classificação de dígitos falados isoladamente é uma aplicação de reconhecimento de fala útil para provedores de serviços de telefonia. Alguns exemplos de serviços que dependem desta etapa são a discagem telefônica por voz, a reserva de passagens aéreas, operações bancárias, cotações de preços, entre outros serviços. Ao utilizar esse tipo de interação, as empresas podem tornar seus serviços de telefonia mais amigáveis quando comparados com a necessidade de digitação de números no teclado do telefone.

Devido à importância da tarefa, diversos trabalhos têm sido propostos na literatura ao longo dos anos para o reconhecimento de dígitos falados em diferentes idiomas como japonês (Kondo et al., 1994), inglês (Abushariah et al., 2010), árabe (Alotaibi, 2003; Hu et al., 2011), hindu (Panwar et al., 2011), bengali (Ghanty et al., 2010), urdu (Azam et al., 2007) e português (Rodrigues e Trancoso, 1999; Bresolin et al., 2008). De acordo com Kopparapu e Rao (2004), esta tarefa não é tão trivial devido as seguintes razões: *i)* dígitos falados são de curta duração, tipicamente alguns segundos de fala e *ii)* alguns dígitos são acusticamente muito semelhantes entre si.

Observou-se que o idioma português foi pouco explorado e que em geral a maior parte dos trabalhos da literatura assume que os coeficientes MFCC são os melhores descritores para os dados. Em Rodrigues e Trancoso (1999) foram utilizados os coeficientes MFCC e modelos ocultos de Markov para realizar a classificação de dígitos falados em português. Em Bresolin et al. (2008) é discutido que atributos baseados na transformada *wavelet* apresentam resultados sensivelmente superiores aos obtidos com os coeficientes MFCC. Entretanto, outros conjuntos de atributos como LSF podem apresentar resultados superiores aos obtidos pelos coeficientes MFCC. Este fato foi discutido em Silva et al. (2012) a partir dos dados anteriormente proposto por Bresolin et al. (2008) e os experimentos deste artigo foram posteriormente estendidos em Silva et al. (2013a) em um novo conjunto de dados com falas em português/inglês coletados pelo nosso grupo de pesquisa no ICMC/USP.

Para avaliar o desempenho do conjunto de atributos LSF, foi utilizada uma estratégia de janelamento com tamanho dinâmico para a extração de atributos. Esta estratégia é interessante por duas razões principais: *i*) o janelamento permite extrair atributos localmente em segmentos do sinal, caracterizando-o no tempo; e *ii*) é uma estratégia simples para obter um vetor de atributos com tamanho fixo a partir de sinais com duração arbitrária. Cada vetor é composto por uma coleção de atributos extraídos dos vários segmentos

do sinal original, obtidos por uma janela deslizante de largura w_s. O valor de w_s é dependente da duração do sinal (s) e o número de janelas que serão utilizadas (N). Além disso, cada janela possui uma sobreposição com a anterior, como pode ser visto na Figura A.3. Tal sobreposição deve ser grande o suficiente para que nenhuma informação seja perdida nas transições do sinal. Comumente, é utilizado uma sobreposição de 50%.

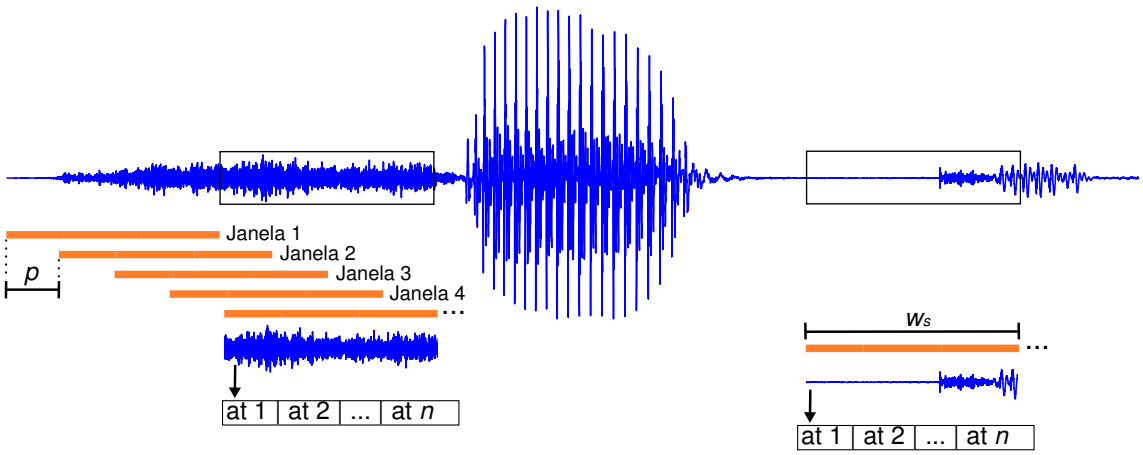


Figura A.3: No janelamento dinâmico, a extração de atributos utiliza uma janela deslizante de largura w_s, proporcional a um número de janelas pré-determinado (N), com tamanho de passo p e, consequentemente, sobreposição de largura $w_s - p$ entre janelas consecutivas. Então, um vetor de atributos $n-$dimensional é extraído de cada janela. Ao fim desse processo, o resultado é um vetor de $n \times N$ atributos que descrevem o sinal (Silva et al., 2012; Silva, 2014).

Nos experimentos realizados, foram utilizadas janelas de largura w_s de acordo com a Equação A.1, em que o é a taxa de sobreposição, com valor no intervalo entre 0 e 1, e e é a largura da janela, desconsiderando a sobreposição entre as janelas consecutivas. O valor de e pode ser obtido pela Equação A.2, em que l_s é o comprimento do sinal e N é o número de janelas a serem utilizadas.

$$w_s = \lceil \frac{e}{1-o} \rceil \tag{A.1}$$

$$e = \lceil \frac{l_s}{N} \rceil \tag{A.2}$$

O conjunto de dados utilizado na avaliação experimental é composto de dígitos falados em português, coletados durante um período de três meses, de 82 homens com idade entre 18 e 42 anos de idade. A taxa de amostragem da gravação é de 22.050Hz. Ao todo, o conjunto de dados possui 216 sequências de 10 dígitos $(0-9)$ cada, totalizando 10 classes e 2.160 exemplos. Dessa maneira, o conjunto de dados possui as classes perfeitamente balanceadas.

Foram comparados 13 MFCC contra 24 e 48 LSF. Esses números de coeficientes foram escolhidos pois 13 MFCC e 12 coeficientes de predição linear são comumente considerados adequados para descrever sinais de áudio (Terasawa et al., 2005; Orság, 2010). Além disso, é comum a utilização de um múltiplo do número de coeficientes de predição linear como o número de coeficientes LSF a ser utilizado.

Ambos conjuntos de atributos foram testados em doze diferentes cenários. Cada cenário é diferente de acordo com as configurações do algoritmo de classificação utilizado. Os algoritmos utilizados e suas respectivas configurações são apresentados na Tabela A.6.

Tabela A.6: Descrição dos cenários de classificação de dígitos falados em português

Cenário	**Indutor/parâmetros**
1-NN	1-Vizinho Mais Próximo
5-NN	5-Vizinhos Mais Próximos ponderado pelo inverso da distância
7-NN	7-Vizinhos Mais Próximos ponderado pelo inverso da distância
9-NN	9-Vizinhos Mais Próximos ponderado pelo inverso da distância
SVM-Poly1	Máquina de Vetores de Suporte com *kernel* polinomial de grau 1
SVM-Poly2	Máquina de Vetores de Suporte com *kernel* polinomial de grau 2
SVM-Poly3	Máquina de Vetores de Suporte com *kernel* polinomial de grau 3
SVM-RBF0,01	Máquina de Vetores de Suporte com *kernel* RBF com γ=0,01
SVM-RBF0,05	Máquina de Vetores de Suporte com *kernel* RBF com γ=0,05
SVM-RBF0,1	Máquina de Vetores de Suporte com *kernel* RBF com γ=0,1
NB	Naïve Bayes
RF	Floresta Aleatória com 15 árvores

Conforme discutido anteriormente, na fase de extração de atributos foi utilizada a estratégia de janelamento dinâmico. Foi estabelecido o número de 25 janelas com uma sobreposição de 75% entre janelas consecutivas. Portanto, cada método de extração de atributos gerou um conjunto de dados cujos exemplos consistem de $25 \times n$ atributos, sendo n o número de atributos extraídos por segmento. Mais precisamente, cada sinal foi transformado em um exemplo com 325, 600 e 1.200 atributos para os 13 MFCC, 24 LSF e 48 LSF, respectivamente.

Os resultados da classificação obtidos a partir de uma estratégia de avaliação de 10×10-*fold cross-validation* são apresentados na Tabela A.7. Os valores são relativos à média e desvio padrão da acurácia obtida na validação cruzada. O melhor resultado de classificação para cada algoritmo é destacado em **negrito**.

Um exemplo da matriz de confusão obtida em uma das execuções da validação cruzada é apresentado na Tabela A.8. Nessa matriz em particular, é apresentado o resultado de uma de dez execuções do classificador SVM-Poly2 sobre o conjunto de dados gerados com a extração de 48 LSF, escolhida aleatoriamente. É possível observar que os erros são bem localizados e totalizam somente 15 casos no total.

Os resultados desses experimentos mostraram que coeficientes LSF podem superar as estratégias que consideram os comumente utilizados 13 MFCC. Mais especificamente, nesse conjunto de dados, o uso de LSF obteve resultados melhores do que o MFCC em todos os cenários. De modo a assegurar maior confiança aos resultados, foi realizado

Tabela A.7: Acurácia e desvio padrão para os três métodos de extração de atributos analisados nos 12 cenários.

Cenário	**13 MFCC**	**24 LSF**	**48 LSF**
1-NN	86,33 (2,15)	92,92 (1,51)	**93,03 (1,64)**
5-NN	89,52 (1,88)	95,57 (1,19)	**95,66 (1,27)**
7-NN	89,61 (1,85)	95,82 (1,32)	**95,98 (1,37)**
9-NN	90,20 (1,82)	**96,13 (1,26)**	95,67 (1,24)
SVM-Poly1	97,96 (0,86)	98,85 (0,69)	**99,30 (0,57)**
SVM-Poly2	97,88 (0,93)	98,77 (0,70)	**99,31 (0,57)**
SVM-Poly3	97,91 (0,90)	98,75 (0,72)	**99,17 (0,63)**
SVM-RBF0,01	93,62 (1,71)	97,93 (0,95)	**98,64 (0,83)**
SVM-RBF0,05	96,88 (1,17)	98,54 (0,83)	**98,70 (0,77)**
SVM-RBF0,1	97,19 (1,04)	**98,32 (0,88)**	98,02 (0,90)
NB	90,63 (1,66)	**94,86 (1,46)**	94,72 (1,35)
RF	91,83 (1,90)	**96,36 (1,23)**	95,89 (1,37)

Tabela A.8: Matriz de confusão de uma execução do algoritmo SVM-Poly2 no conjunto de dados de dígitos, com o conjunto de atributos 48-LSF

Real	**Predito**									
	Zero	Um	Dois	Três	Quatro	Cinco	Seis	Sete	Oito	Nove
Zero	214							3		
Um		216							1	
Dois			214						3	
Três				216						
Quatro					215					1
Cinco						216	1	1		
Seis							215			
Sete	2							212		
Oito			2						212	
Nove					1					215
Acurácia (%)	99,07	100	99,07	100	99,54	100	99,54	98,15	98,15	99,54

o teste estatístico de Wilcoxon. Esse teste é recomendado para comparar populações pareadas. O teste indicou que 24 e 48 LSF de fato obtiveram melhor desempenho que o método de comparação (MFCC) com um grau de confiança de 95%.

Posteriormente, os experimentos realizados em Silva et al. (2012) foram estendidos em Silva et al. (2013a). Especificamente, o estudo anterior foi estendido em termos da metodologia e análise dos dados nos seguintes aspectos:

- Foi utilizado um novo conjunto de dados de dígitos falados em português e inglês, coletado em nosso grupo de pesquisa[3] no ICMC/USP. Com isso, foi possível avaliar a influência da língua nos resultados da classificação, bem como o potencial de extrapolar as técnicas para outros idiomas;

- Foi avaliado um conjunto mais amplo de situações experimentais com diferentes números de coeficientes MFCC e LSF. Assim, proporcionou-se um entendimento mais profundo da influência de tais parâmetros no desempenho da classificação. Pode-se observar, por exemplo, que uma quantidade de coeficientes diferente da

[3]Laboratório de Inteligência Computacional (LABIC) – http://labic.icmc.usp.br/

tradicionalmente utilizada na literatura (13 MFCC e 24 LSF) são capazes de fornecer melhores resultados;

- Foi variada a taxa de amostragem de áudio para simular a faixa de resposta de frequência de redes telefônicas públicas. Dessa maneira, foi possível avaliar a robustez dos métodos em canais de baixa qualidade, como linhas telefônicas padrão.

A.1.3 Reconhecimento de Músicas *Cover*

Música *cover* refere-se a regravação de uma canção previamente gravada por um compositor, realizada em estúdio ou ao vivo. Com a popularidade de *websites* como *YouTube*, houve um significativo aumento no número de músicas *cover* disponibilizadas na internet nos últimos anos. O reconhecimento automático destas versões é interessante por diversos motivos. Por exemplo, a identificação das diferentes versões de uma mesma música pode ser utilizada para estimar um indicativo de sua popularidade ou a dificuldade/facilidade de execução de algum dos instrumentos e que desperta interesse em muitos músicos. Por outro lado, o reconhecimento automático pode ser uma ferramenta importante para sistemas que hospedam conteúdo de áudio como *YouTube, Last.fm* e *SoundCloud* e necessitam evitar problemas legais devido ao envio indevido por seus usuários de versões de músicas protegidas por direitos autorais. Entretanto, conforme observado por Serrà et al. (2010), a identificação de músicas *cover* é dificultada por diferenças entre a música original e a sua versão em características como timbre, tempo, estrutura, arranjo e até mesmo o idioma dos vocais.

Outra dificuldade encontrada por sistemas de reconhecimento automático de músicas *cover*, particularmente por aqueles baseados em comparações por similaridade, é o alto custo de tempo para recuperar as potenciais versões. A dimensão deste problema se agrava se considerarmos que somente o *YouTube* recebe cerca de 300 horas de vídeo acompanhado de áudio por minuto[4] e uma quantidade expressiva destes dados está relacionada a conteúdo musical. Desse modo, os algoritmos que realizam o reconhecimento de músicas *cover* devem ser eficientes no processamento de consultas para que possam ser utilizados em grandes bases de dados.

Em Silva et al. (2015a) nós propomos um algoritmo eficiente para o reconhecimento e recuperação de músicas *cover* baseado na extração de pequenos segmentos representativos das músicas. A principal hipótese deste trabalho é a possibilidade de caracterização/identificação de músicas a partir de pequenos segmentos e utilizar esta informação para buscar por suas versões sem a necessidade de verificar todo o conteúdo das músicas nas comparações por similaridade. Esta hipótese é baseada no sucesso observado em uma técnica similar para a classificação de séries temporais denominada *shapelets* (Ye e Keogh, 2009). Como neste trabalho é proposto uma variação da técnica *shapelet* para o uso no

[4]Estatística obtida em 2015 e consultada no endereço `www.youtube.com/yt/press/statistics.html`

contexto de recuperação de música baseada em conteúdo, denominada *Music Shapelets*, uma visão geral da técnica originalmente proposta para séries temporais é apresentada a seguir.

No contexto de classificação de séries temporais, uma *shapelet* pode ser informalmente definida como uma subsequência de série temporal que seja a mais representativa de sua classe. Assim, dado um conjunto de *shapelets* extraídas dos dados, o algoritmo utiliza estas informações para a construção de um modelo baseado em árvores de decisão que permita a classificação de novos dados de maneira eficiente, sem a necessidade de realizar inúmeras comparações no conjunto de treino. Antes da etapa de classificação, o algoritmo necessita realizar uma fase de treinamento que consiste em três passos principais:

1. Geração de candidatos: dado um conjunto de treinamento S com dados rotulados, são extraídas todas as subsequências de cada série temporal de S. Neste passo é necessário definir o tamanho das subsequências candidatas. No algoritmo original é proposto que sejam extraídas todas as subsequências das séries entre um tamanho mínimo (min_{tam}) e um tamanho máximo (max_{tam});

2. Avaliação da qualidade dos candidatos: a qualidade das subsequências anteriormente extraídas são avaliadas de acordo com o seu poder de separabilidade inter-classes;

3. Geração do modelo de classificação: é construída uma árvore de decisão com base nas *shapelets* candidatas de melhor qualidade.

Em diferentes passos do algoritmo é necessário medir a similaridade entre uma *shapelet* candidata s e uma série temporal x. Como os tamanhos de s e x podem ser potencialmente diferentes, nem sempre é possível realizar a comparação direta entre eles. Desse modo, dado l o tamanho da subsequência candidata, a distância $d(s, x)$ é definida como a menor distância euclidiana entre o candidato s e cada subsequência de x de tamanho l.

A ideia geral da técnica é utilizar as distâncias entre as subsequências candidatas e as séries temporais de treino para construir um modelo de classificação. Para isso, primeiro o algoritmo estima o melhor ganho de informação que pode ser obtido por cada candidato. Isso é feito agrupando-se os exemplos de treino mais próximos da *shapelet* candidata (de acordo com um limiar de distância) daqueles exemplos de treino mais distantes do mesmo candidato. A partir da avaliação de diferentes limiares de distância para um mesmo candidato, é escolhido o limiar responsável por fornecer a melhor separação entre as classes do problema, chamado de *best split point*. Assim, dado um candidato a *shapelet* e o seu melhor limiar de separação, é possível calcular o respectivo ganho de informação.

Por fim, o algoritmo utiliza o ganho de informação para construir uma árvore de decisão. Para cada classe do problema, seleciona-se o candidato a *shapelet* com maior ganho de informação e o utiliza como nó da árvore. Assim, dado um exemplo de teste q a ser classificado, mede-se a distância entre q e os nós da árvore. Se a distância for menor

ou igual ao *best split point* da *shapelet* pertencente ao nó, atribui-se a q a mesma classe associada à *shapelet*. Caso contrário, associa-se o rótulo de outra classe.

Neste trabalho, são propostas diferentes adaptações da técnica original de *shapelet* para a rápida recuperação de música por conteúdo, especificamente para o reconhecimento de músicas *cover*. Estas adaptações são discutidas a seguir.

Janelamento

A abordagem original da técnica de *shapelets* para a extração de subsequências candidatas utiliza janelas deslizantes de diferentes tamanhos. Especificamente, dada uma faixa de valores fornecida pelo usuário, todos os valores são considerados para o tamanho da janela. Este processo é realizado no decorrer de toda série temporal e em todas as séries do conjunto de treinamento. Desse modo, é um processo custoso em termos de tempo de execução. Em especial, este problema é potencializado no contexto de música dado que as músicas possuem uma dimensionalidade consideravelmente superior (principalmente em termos de número de observações) em relação aos dados de séries temporais em que o método foi originalmente proposto. Desse modo, neste trabalho foi utilizado um conjunto reduzido de valores para a definição do tamanho da janela ao invés de realizar a busca em toda uma faixa de valores. Esta decisão é baseada na realização de experimentos preliminares, em que se observou que é possível encontrar bons candidatos a *shapelet* a partir de um conjunto reduzido de valores. Além disso, ao invés de utilizar janelas que se iniciam em cada observação da série temporal, é proposto neste trabalho que seja considerado um deslocamento proporcional ao tamanho da janela para o seu início. Estas pequenas modificações visam reduzir o custo para a geração das subsequências na etapa de treinamento.

Dynamic Time Warping

Para comparar a similaridade entre uma subsequência e toda uma série temporal, é originalmente utilizada a distância euclidiana. Entretanto, a distância euclidiana é sensível a distorções no eixo do tempo, chamadas de *warping*. O uso de uma medida invariante a *warping*, como a *Dynamic Time Warping* (DTW), é comum em aplicações que verificam a similaridade entre músicas devido à diferenças no andamento ou ritmo que podem ocorrer quando uma música é executada por diferentes artistas. Desse modo, neste trabalho foi avaliado o uso tanto da distância euclidiana como da medida DTW para comparar *shapelets* extraídas de músicas. Embora a medida DTW possua complexidade quadrática enquanto a distância euclidiana apresenta complexidade linear, é importante destacar que atualmente existem diversos métodos que buscam acelerar o cálculo da medida DTW e que podem ser utilizados neste trabalho (Rakthanmanon et al., 2012).

Avaliação da Qualidade do *Shapelet*

A técnica de *shapelet* foi originalmente proposta para problemas de classificação em séries temporais. Entretanto, neste trabalho há o interesse em gerar um *ranking* de acordo com a similaridade observada com um dado de consulta. Por exemplo, dada a gravação de uma música em sua versão original, retorna-se 5 outras gravações mais similares e possivelmente *covers* da música original. Desse modo, o ganho de informação não é a melhor escolha para avaliar a qualidade de um candidato a *shapelet*, já que esta medida busca maximizar a separabilidade inter-classes. No problema de recuperação de músicas abordado neste trabalho, há um grande número de classes (cada classe é representada por uma música em especial) com alguns poucos exemplos referentes as versões provenientes da música original. Assim, a avaliação de separabilidade entre as classes é dificultada.

Desse modo, é proposto o uso de um critério baseado em distância. Neste critério é considerado que um bom candidato apresenta um pequeno valor de distância para todas as versões de uma determinada música e um alto valor de distância para qualquer gravação de outra música. Este critério é formalmente definido na Equação A.3.

$$DistDiff(s) = \min_{i=1..n}(dist(s, OtherClass(i))) - \frac{1}{m}\sum_{i=1}^{m} dist(s, SameClass(i)), \quad \text{(A.3)}$$

em que s é um candidato a *shapelet*, $SameClass$ é o conjunto de m versões da música em que o candidato foi extraído, $OtherClass$ é o conjunto com n gravações que não representam versões da música original em que s foi extraído e $dist(s, Set(i))$ é a distância entre o candidato e a i-ésima gravação em Set ($SameClass$ ou $OtherClass$). Para um bom candidato é esperado que a distância seja obtida a partir da observação de um alto valor no primeiro termo da equação e um baixo valor para o segundo termo. Desse modo, quanto maior o valor de *DistDiff*, melhor será a qualidade do candidato a *shapelet*. No caso de empate, o melhor candidato é aquele que fornece o melhor *ranking* médio.

Similaridade entre *Shapelets*

Como há o interesse de separabilidade entre classes no problema de classificação de séries temporais, armazena-se somente um *shapelet* para cada classe do problema. Por outro lado, no problema abordado neste trabalho também há o interesse em representar as diferentes versões de uma mesma música. Desse modo, armazena-se um *shapelet* para cada gravação de música presente conjunto de treino ao invés de armazenar somente um *shapelet* por composição.

A consulta e a recuperação de músicas *cover* são realizadas em dois passos simples. Primeiro o método mede a similaridade entre a *shapelet* da música de consulta e cada

shapelet encontrada na fase de treino. Por fim, o *ranking* é dado pela ordenação crescente das distâncias.

Triplets

Em um cenário real em que a tarefa de recuperação de música é realizada, é altamente provável que uma mesma música possua de uma a três versões autorizadas pelo artista, como a gravação original, uma versão acústica e uma versão ao vivo. Obviamente existem exceções como *remix* ou diversas versões ao vivo. Por isso, quando se extrai as *shapelets* destas composições de uma maneira convencional, o cálculo da qualidade da *shapelet* pode ser prejudicado, dado que existem apenas alguns exemplos para cada classe no conjunto de treino. Além disso, no contexto de música somente um pequeno segmento pode ser pouco informativo para caracterizar uma composição. Esta observação é suportada por outros trabalhos da literatura que lidam com dados de música. Por exemplo, Silla Jr et al. (2008) analisam as gravações musicais em seu início, meio e fim para realizar a tarefa de classificação de gênero.

Visando mitigar estes problemas, neste trabalho utilizou-se três *shapelets* para representar cada gravação musical. Esta tripla de *shapelets* é denominada de *triplets*. A Figura A.4 ilustra o procedimento geral para a extração dos *triplets*.

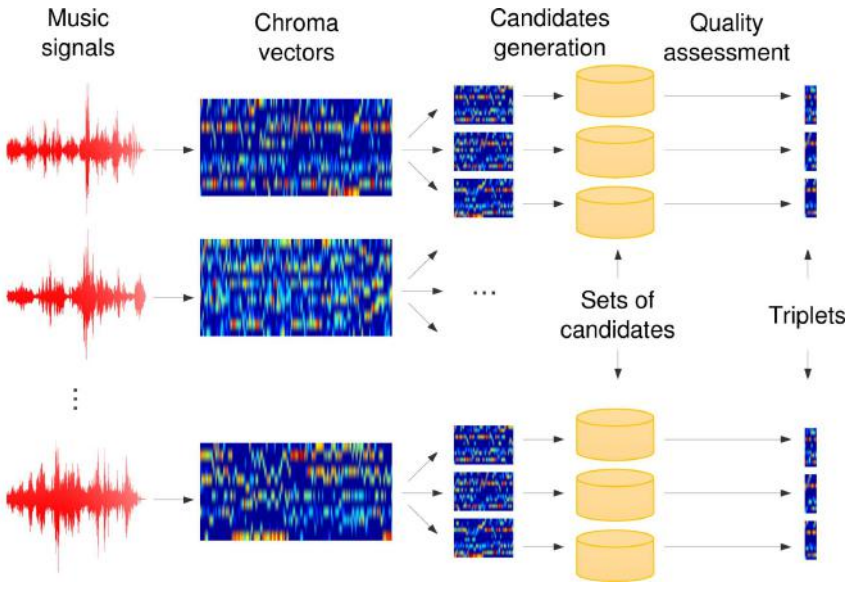


Figura A.4: Procedimento para a geração dos *triplets* (Silva et al., 2015a).

Inicialmente os sinais de música representados no tempo são transformados para uma representação mais informativa em que é possível a extração de atributos descritivos. Neste trabalho, optou-se por utilizar os atributos provenientes da representação *Chroma*. Estes atributos são popularmente utilizados na comparação por similaridade de músicas e constituem uma representação da energia espectral observada na faixa de frequência de

cada um de 20 semitons. O segundo passo é dividir o vetor de atributos *Chroma features* em três partes de mesmo tamanho. Em seguida, são gerados todos os candidatos a *shapelet* para cada parte. Por fim, avalia-se a qualidade das *shapelets* por meio da medida *DistDiff* e selecionam-se somente as 3 subsequências melhores avaliadas para cada música. Na fase de recuperação, utiliza-se a distância média entre a música de consulta e cada uma das *triplets*.

Para avaliar o método proposto *Music Shapelets*, foram utilizados dois conjuntos de dados. O primeiro conjunto denominado *123classical* é composto por músicas clássicas e foi originalmente proposto por Bello (2011). Este conjunto de dados possui 123 gravações de 19 diferentes trabalhos do período clássico e período romântico. Dentre essas gravações, 67 foram tocadas por orquestras, enquanto as 56 demais foram tocadas no piano. Como este conjunto de dados não possui uma divisão natural em dados de treino/teste e possui uma quantidade reduzida de dados, na avaliação experimental foi considerado o uso de amostragem aleatória e estratificada com 1/3 dos dados para treino e o restante para teste. Com este procedimento, o número de exemplos por classe no conjunto de treino varia de 1 a 5.

O segundo conjunto de dados foi construído especificamente para este trabalho e é composto por músicas populares extraídas de vídeos hospedados no *YouTube*. Este conjunto de dados é denominado *YouTube Covers*[5] e possui 50 composições de diferentes gêneros musicais como *reggae, jazz, rock* e *pop* acompanhadas de versões *cover*. Nos experimentos realizados neste trabalho, o conjunto de dados foi dividido em treino e teste. O conjunto de treino possui a gravação original em estúdio de cada composição e uma versão ao vivo. No conjunto de teste há a presença de 5 diferentes versões das composições que incluem versões em diferentes estilos musicais, execuções ao vivo por outros artistas populares, gravações realizadas por fãs, entre outros. Desse modo, este conjunto de dados possui 350 músicas, sendo 100 exemplos de treino e 250 para teste.

Os resultados apresentados neste trabalho consideram dois diferentes cenários de avaliação: *i*) conjunto de teste como consulta e *ii*) conjunto de treino como consulta. Em ambos os cenários realiza-se a extração dos *triplets* da partição de treino. No primeiro cenário simula-se a situação em que deseja-se saber se uma música de consulta (teste) é uma versão *cover* de alguma música previamente conhecida (treino). Em outras palavras, são utilizadas músicas desconhecidas para encontrar músicas similares conhecidas. No segundo cenário é simulada a situação em que o autor de uma das músicas do conjunto de treino deseja sabe se existem versões não autorizadas de sua música em uma base de dados. Desse modo, a música original (dados de treino) é utilizada como consulta para retornar possíveis músicas *cover* do conjunto de teste.

[5]O conjunto de dados *YouTube Covers* é livremente disponibilizado no endereço http://sites.google.com/site/ismir2015shapelets/

O método proposto possui dois parâmetros relacionados à etapa de janelamento, a saber: *i*) tamanho da janela e *ii*) proporção de sobreposição entre janelas consecutivas. Para o primeiro parâmetro, foram avaliados os valores 25, 50 e 75, sendo que os resultados apresentados neste trabalho consideram somente 25 observações para os *triplets*, devido aos bons resultados alcançados e eficiência. Para o segundo parâmetro fixou-se o valor de 2/3 do tamanho da janela como proporção de sobreposição entre as janelas.

Foram utilizadas três medidas para avaliar a tarefa de reconhecimento de músicas *cover*. Estas medidas levam em consideração a posição de músicas relevantes no *ranking* de similaridade retornado pelo método. No contexto deste trabalho, uma música relevante refere-se a uma versão *cover* de determinada composição. Assim, dado um conjunto de n músicas de consulta, um método de recuperação é responsável por retornar um *ranking* r_i $(i = 1, 2, \ldots, n)$ para cada música de consulta. A função $\Omega(r_{i,j})$ retorna o valor 1 se a j-ésima música do *ranking* obtido pela música de consulta i for relevante ou 0, caso contrário.

A primeira medida de avaliação representa o número médio de músicas relevantes retornadas nas primeiras 10 posições do *ranking* (*MNTop10*). Essa medida é formalmente definida de acordo com a Equação A.4.

$$MNTop10 = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{10}\Omega(r_{i,j}) \tag{A.4}$$

A medida *Mean Average Precision* (*MAP*) representa o valor médio de *Average Precision* (*AP*) obtido para cada música de consulta. A medida *AP* é definida de acordo com a Equação A.5.

$$AP(r_i) = \frac{1}{n}\sum_{j=1}^{n}\left[\Omega(r_{i,j})\left(\frac{1}{j}\sum_{k=1}^{j}\Omega(r_{i,k})\right)\right] \tag{A.5}$$

Por fim, é utilizada a medida *MFRank* que representa a média de vezes em que uma música *cover* é corretamente identificada na primeira posição do *ranking*. Esta medida é definida na Equação A.6.

$$MFRank = \frac{1}{n}\sum_{i=1}^{n} fp(r_i), \tag{A.6}$$

em que $fp(r_i)$ é a função que retorna a posição da primeira ocorrência de um objeto relevante no *ranking* r_i.

Para as duas primeiras medidas de avaliação apresentadas, altos valores indicam um melhor desempenho. Para a última medida, um melhor desempenho é obtido a partir de valores menores.

Na Tabela A.9 são apresentados os resultados obtidos no conjunto de dados *123Classical* e na Tabela A.10 são apresentados os resultados obtidos no conjunto de dados *YouTube*

Covers. Em ambos os casos foi considerado os dois cenários propostos na avaliação com os dados do conjunto de teste como consulta (cenário 1) e os dados do conjunto de treino como consulta (cenário 2). Conforme anteriormente discutido, foi avaliado o desempenho dos *Triplets* tanto com o uso da distância euclidiana (ED) como da medida *Dynamic Time Warping* (DTW). O melhor resultado obtido em cada medida é destacado em negrito. O método proposto foi comparado com outros dois métodos da literatura. O primeiro é o alinhamento DTW realizado sobre o vetor de características extraído de toda a música. O segundo método utiliza um algoritmo de sumarização de música para encontrar os segmentos mais significativos das gravações proposto por Cooper e Foote (2002). Este método considera que os trechos mais significativos das músicas são aqueles que mais se repetem durante a sua execução. Para todos os métodos foram utilizados os *Chroma Energy Normalized Statistics* (CENS) como atributos descritivos dos dados.

Tabela A.9: Resultados obtidos no conjunto de dados *123Classical*.

Cenário 1 - Teste como consulta				**Cenário 2 - Treino como consulta**			
	MNTop10	MAP (%)	MFRank		MNTop10	MAP (%)	MFRank
DTW	2,34	97,24	1,12	DTW	4,73	98,92	**1,00**
Sumarização	2,27	93,46	**1,00**	Sumarização	4,44	91,52	1,02
Triplets-DTW	**2,39**	97,24	1,02	Triplets-DTW	**4,78**	**99,41**	**1,00**
Triplets-ED	2,38	**98,05**	**1,00**	Triplets-ED	4,71	97,92	**1,00**

Tabela A.10: Resultados obtidos no conjunto de dados *YouTube Covers*.

Cenário 1 - Teste como consulta				**Cenário 2 - Treino como consulta**			
	MNTop10	MAP (%)	MFRank		MNTop10	MAP (%)	MFRank
DTW	1,14	42,49	11,69	DTW	2,11	39,19	6,58
Sumarização	0,85	32,11	13,82	Sumarização	1,66	29,20	14,46
Triplets-DTW	**1,29**	45,55	**8,45**	Triplets-DTW	2,82	52,87	**4,65**
Triplets-ED	1,26	**47,80**	8,49	Triplets-ED	**2,87**	**54,95**	5,18

Os resultados apresentados mostram que o método proposto é capaz de superar o algoritmo de sumarização e alcança resultados superiores ou iguais ao uso da técnica DTW aplicada em todo o vetor de características. Entretanto, as consultas utilizando a técnica de *triplets* são significativamente mais eficientes do que o uso da DTW por toda música. Embora o método necessite realizar uma fase de treinamento que não é necessária na busca por similaridade utilizando a DTW, esta fase é realizada uma única vez.

Para ilustrar a eficiência do método, na Tabela A.11 é apresentado o tempo gasto em segundos para realizar a tarefa de recuperação utilizando a técnica *Triplets-ED* e a técnica DTW por todo o vetor de características, considerando todo o conjunto de teste de ambos os conjuntos de dados. Pode-se observar que o método proposto é cerca de 15 vezes mais rápido para realizar a recuperação das músicas quando comparado ao método DTW.

Tabela A.11: Tempo total em segundos para calcular a distância entre todas as músicas de consulta (conjunto de treino) e as músicas do conjunto de teste utilizando DTW e *Triplets-ED*.

	Conjunto de dados	
	123Classical	*YouTube Covers*
DTW	2.294	14.124
Triplets-ED	148	928

A.2 Processamento de Séries Temporais

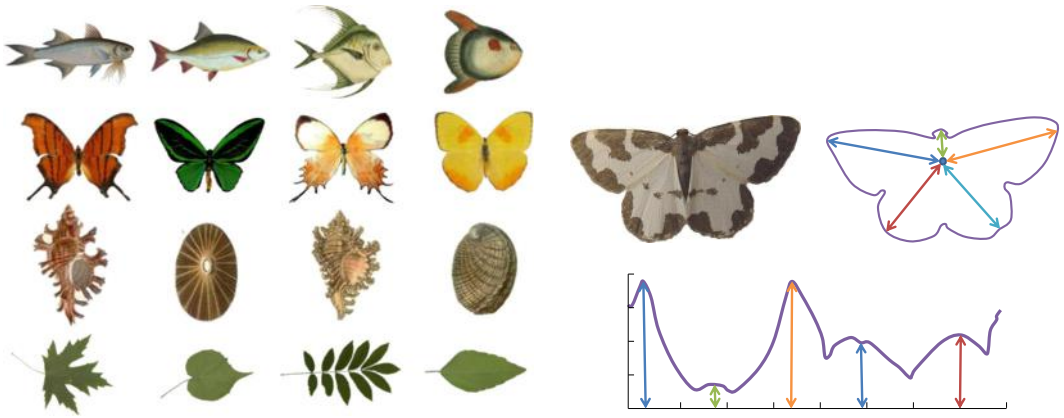
A.2.1 Medida de Distância Invariante a Complexidade

Embora exista uma grande variedade de algoritmos propostos na literatura para as tarefas de classificação, agrupamento e descoberta de *motifs* em séries temporais, diversas pesquisas têm mostrado que os métodos que utilizam o conceito de similaridade definido por meio de uma função de distância fornecem uma maneira simples e intuitiva para a resolução dessas tarefas, mas também tem se mostrado competitivos com outros métodos mais complexos (Ding et al., 2008; Wang et al., 2013).

A popularidade dos métodos baseados em similaridade para processamento de séries temporais tem feito com que diversos pesquisadores proponham variações dos algoritmos clássicos com o objetivo de que a medida de distância se torne invariante a alguma característica dos dados. Um exemplo é a invariância à amplitude, de modo que dados em diferentes unidades (como as temperaturas Celsius e Fahrenheit) possam ser diretamente comparados. Outras formas de invariância incluem invariância à fase (utilizada com dados periódicos), escala local (*warping*), escala uniforme e oclusão (Keogh e Lin, 2005).

Neste trabalho foi observado que diversos problemas apresentam séries temporais com diferentes complexidades e que pares de objetos complexos, mesmo aqueles que são visualmente similares, tendem a ser considerados pelas atuais medidas de distância, mais dissimilares do que pares de objetos simples. Sendo necessário, portanto, o uso de medidas de distâncias invariantes a complexidade para que objetos complexos não sejam incorretamente associados a objetos simples de outra classe. Diferenças de complexidade entre objetos de diferentes classes ocorrem naturalmente na maioria dos domínios de aplicação. Um exemplo visual é apresentado na Figura A.5-*a)*, na qual são apresentadas algumas fotografias de objetos com diferentes complexidades de forma. É possível observar, por exemplo, como uma folha pode ter uma complexidade de forma simples como o formato oval da folha abaixo à direita até a forma complexa e serrilhada da folha de plátano abaixo à esquerda. É interessante destacar que as imagens bidimensionais apresentadas na Figura A.5-*a)* podem ser facilmente transformadas em séries "temporais" unidimensionais a partir de uma mudança de representação que considera as distâncias entre um ponto de referência da imagem (como o seu centro de massa) e os pontos localizados em sua borda,

sendo possível extrair e descrever o formato da imagem (Arica e Vural, 2003), conforme ilustrado na Figura A.5-*b)*.



(a) (b)

Figura A.5: *a)* Exemplo visual de objetos com diferentes complexidades de forma (Batista et al., 2014); *b)* Exemplo de transformação de imagem bidimensional em série "temporal" unidimensional para a extração da forma do objeto (Batista et al., 2010).

Para melhor ilustrar o problema, considere o exemplo artificial apresentado na Figura A.6. Nesta imagem, são apresentadas oito diferentes figuras geométricas e a sua respectiva série temporal (normalizada) obtida a partir da transformação da representação, anteriormente discutida. É possível observar o aumento da complexidade da forma de cada figura geométrica, bem como da série temporal que as representam. Se calcularmos a similaridade a partir da distância euclidiana entre todas as séries temporais, conforme apresentado na Tabela A.12, pode-se observar algumas relações interessantes. Considerando somente figuras com formatos simples, os resultados são intuitivos. Por exemplo, a figura mais similar a estrela com 4 pontas é a estrela com 5 pontas e vice-versa. Ainda, a figura mais similar a estrela com 6 pontas é a figura com 7 pontas. Entretanto, os resultados não são tão claros se considerarmos as figuras de formato mais complexo. Por exemplo, a figura mais similar as estrelas de 32 e 12 pontas é a de 4 pontas (a mais simples). Para a figura de 24 pontas, embora a mais similar seja a estrela com 12 pontas, o segundo e o terceiro objeto mais similares são os que possuem 4 e 5 pontas, respectivamente. Para que estas relações fiquem mais claras, é apresentado na Figura A.7 um dendrograma obtido a partir do agrupamento hierárquico considerando *average link* e distância euclidiana.

Dado este problema de que séries temporais complexas são consideradas mais similares a séries mais simples do que de outras séries também complexas, é proposto neste trabalho a medida *Complexity-Invariant Distance* (CID) para séries temporais (Batista et al., 2014). Intuitivamente, a medida CID mede a extensão da série temporal se ela fosse "esticada" até se tornar um segmento de reta em uma medida denominada Complexidade Estimada (CE). Dessa maneira, séries temporais complexas tendem a resultar em segmentos de reta mais longos do que séries temporais mais simples. A Figura A.8 ilustra

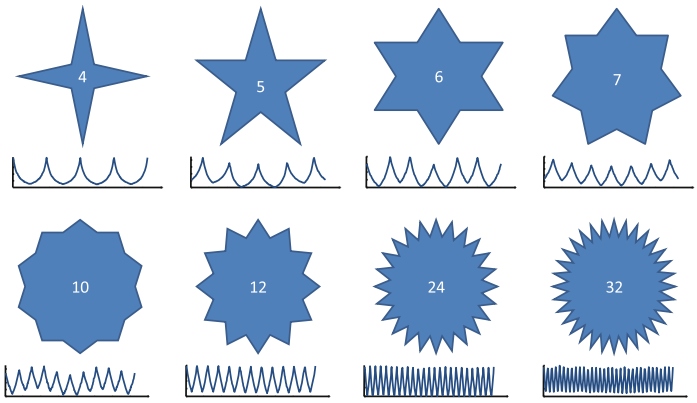


Figura A.6: Figuras geométricas e respectivas séries temporais obtidas a partir do cálculo das distâncias entre o ponto central e os pontos localizados no contorno da figura. A quantidade de pontas de cada figura geométrica está localizada em seu centro. É possível observar o aumento da complexidade da série temporal de cada figura, de acordo com a quantidade de pontas.

Tabela A.12: Matriz de similaridade (distância euclidiana) entre as séries temporais provenientes de figuras geométricas. As séries são identificadas de acordo com a quantidade de pontas.

	4	5	6	7	10	12	24	32
4		1,000	1,122	1,231	1,181	1,048	1,155	1,170
5			1,318	1,068	1,103	1,153	1,165	1,180
6				1,088	1,097	1,103	1,186	1,200
7					1,217	1,199	1,198	1,191
10						1,263	1,195	1,214
12							1,135	1,199
24								1,191
32								

a intuição dessa medida de complexidade. Dada as três séries temporais $T1$, $T2$ e $T3$, pode-se observar que a Complexidade Estimada CE($T3$) > CE($T2$) > CE($T1$).

Dada uma série temporal Q de tamanho n, o cálculo da Complexidade Estimada é dada pela Equação A.7.

$$CE(Q) = \sqrt{\sum_{i=1}^{n-1}(q_i - q_{i+1})^2} \tag{A.7}$$

A medida CID utiliza as informações de complexidade estimada entre duas séries temporais como um fator de correção em conjunto com outras medidas de distância existentes, como distância euclidiana (ED) ou *Dynamic Time Warping* (DTW). Dada a distância euclidiana $ED(Q, C)$ entre as séries Q e C, a medida CID é definida de acordo com a Equação A.8.

$$CID(Q, C) = ED(Q, C) \times CF(Q, C), \tag{A.8}$$

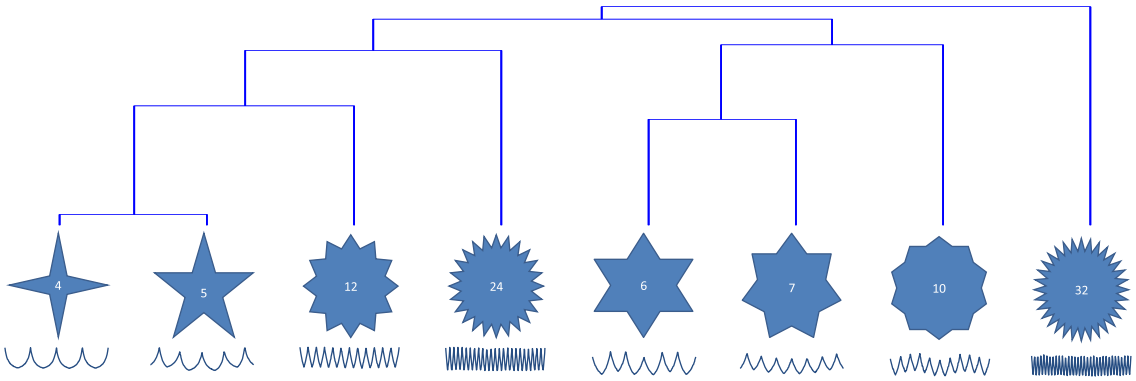


Figura A.7: Dendrograma obtido a partir do agrupamento hierárquico (*average link* e distância euclidiana) das séries temporais provenientes de figuras geométricas.

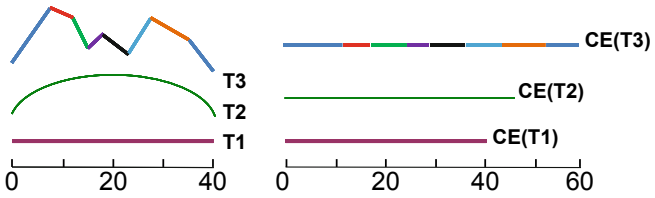


Figura A.8: Complexidade estimada (CE) de três diferentes séries temporais.

em que CF é um fator de correção de complexidade definido de acordo com a Equação A.9.

$$CF(Q,C) = \frac{max(CE(Q), CE(C))}{min(CE(Q), CE(C))}, \tag{A.9}$$

e $CE(T)$ é a Complexidade Estimada da série T, conforme anteriormente discutido.

Apenas como exemplo ilustrativo, a similaridade entre as figuras geométricas anteriormente discutidas foram calculadas utilizando a medida de distância invariante à complexidade (CID) em conjunto com a distância euclidiana. A matriz de similaridade entre os objetos é apresentada na Tabela A.13 e o dendrograma obtido a partir de agrupamento hierárquico utilizando a distância CID e *average linkage* é apresentado na Figura A.9. Nota-se que com a medida proposta, a similaridade entre os objetos se torna muito mais intuitiva e próxima da inspeção visual e que se evita a associação de séries mais complexas com séries simples.

Embora o exemplo apresentado na Figura A.9 seja proveniente de séries temporais artificiais, o problema de complexidade está presente em diversas aplicações do mundo real. Para demonstrar a efetividade da medida em uma variedade de problemas reais, foi realizada uma avaliação da medida CID em problemas de classificação em um conjunto de *benchmark* com 43 conjuntos de dados de séries temporais provenientes de diferentes domínios de aplicação como biologia, medicina, entomologia, engenharia, astronomia, processamento de sinais, entre outros (Chen et al., 2015). Um resumo dos resultados em comparação com a distância euclidiana (ED) é apresentado na Figura A.10. Nesta

Tabela A.13: Matriz de similaridade (distância CID) entre as séries temporais provenientes de figuras geométricas. As séries são identificadas de acordo com a quantidade de pontas.

	4	5	6	7	10	12	24	32
4		1,000	1,061	1,229	1,707	1,839	3,843	5,042
5			1,307	1,138	1,700	2,159	4,135	5,422
6				1,096	1,599	1,953	3,982	5,214
7					1,651	1,977	3,744	4,819
10						1,439	2,580	3,396
12							2,018	2,761
24								1,446
32								

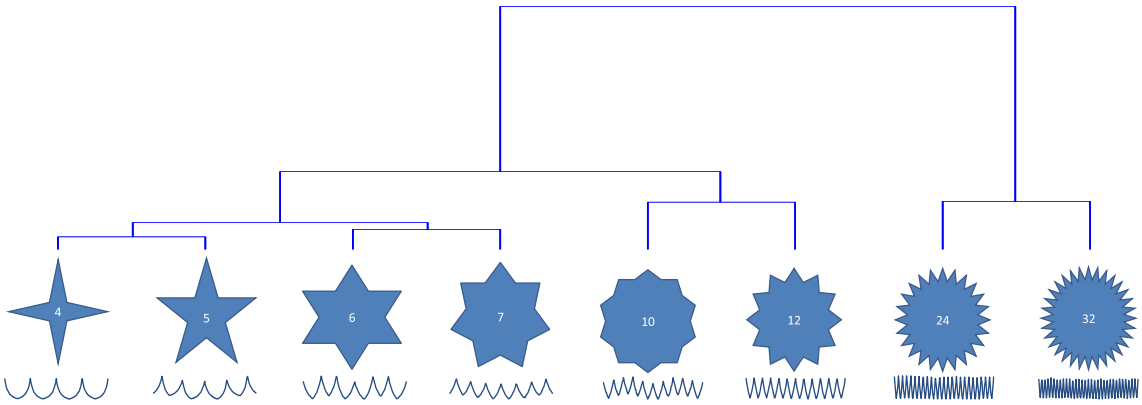


Figura A.9: Dendrograma obtido a partir do agrupamento hierárquico (*average link* e distância CID) das séries temporais provenientes de figuras geométricas.

imagem, cada conjunto de dados é representado por um ponto em que a coordenada no eixo x representa a acurácia obtida pela distância euclidiana e a coordenada no eixo y representa a acurácia obtida pela medida proposta CID. Pode-se observar que a invariância à complexidade melhora os resultados de classificação obtidos pela distância euclidiana em 32 de 43 conjuntos de dados. Em outros 3 conjuntos de dados, os resultados foram os mesmos e em 8 conjuntos de dados a medida CID foi responsável por diminuir os resultados obtidos pela distância euclidiana.

A efetividade da medida CID também foi avaliada em um cenário de agrupamento de dados. Para isso, avaliou-se o desempenho da medida em três paradigmas de agrupamento: hierárquico, particional e espectral. Para o método de agrupamento hierárquico foi utilizado o algoritmo aglomerativo com o critério *average linkage*, que considera a média das distâncias entre os pontos de um agrupamento. Como algoritmo particional, foi utilizado o algoritmo k-Medoids (Kaufman e Rousseeuw, 2009). Por fim, para o agrupamento espectral foi utilizado o algoritmo *normalized spectral clustering* (Ng et al., 2002). A medida foi novamente avaliada nos 43 conjuntos de dados de séries temporais e os resultados são reportados utilizando o Índice Rand, popular na validação de agrupamento de dados (Jain e Dubes, 1988). Os resultados apresentados na Figura A.11 indicam que para

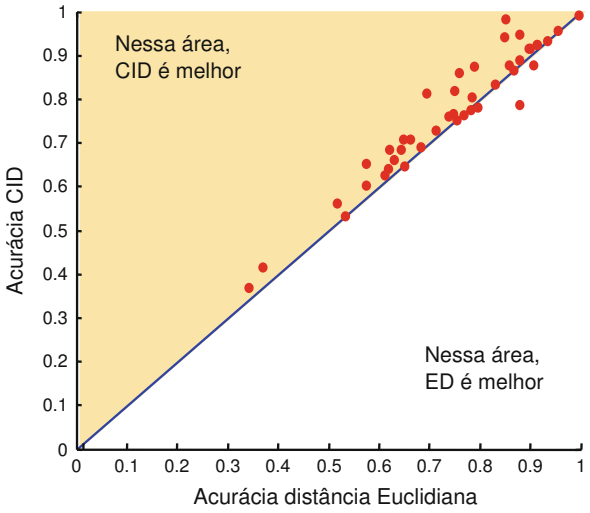


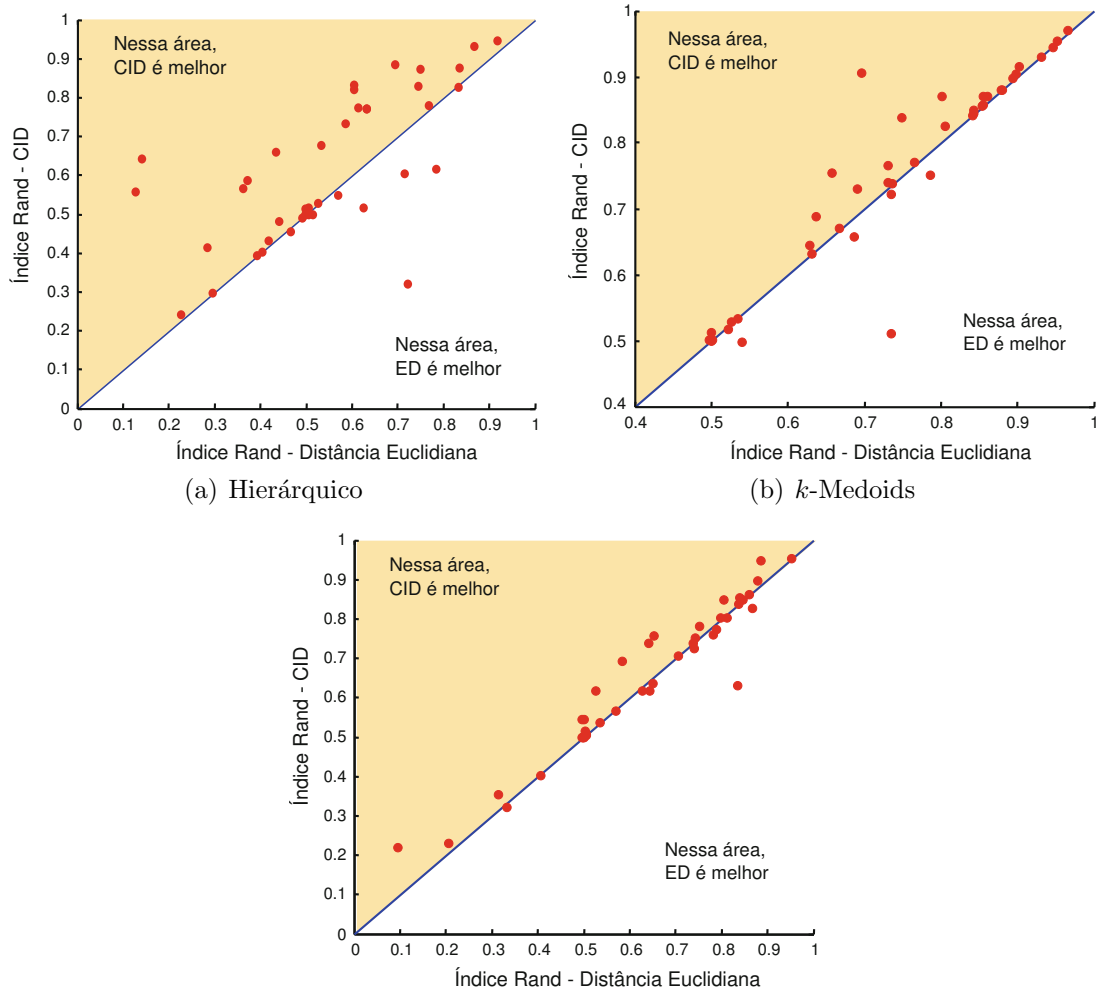
Figura A.10: Comparação dos resultados de acurácia obtidos em 43 conjuntos de dados pela medida de distância invariante à complexidade (CID) e distância euclidiana (ED).

os três métodos de agrupamento, a medida CID é responsável por melhorar os resultados obtidos pela distância euclidiana na maioria dos conjuntos de dados.

A.2.2 Medida de Distância Baseada em Compressão de Dados

Na tarefa de classificação de séries temporais, há trabalhos que evidenciam que o simples algoritmo do vizinho mais próximo em conjunto com uma medida de distância adequada é muito difícil de ser superado mesmo por métodos mais complexos (Ding et al., 2008; Wang et al., 2013). Porém, também foi mostrado que em alguns casos, características importantes das séries temporais não são evidentes em sua representação primitiva no tempo. Por isso, muitos trabalhos realizam uma transformação nos dados, de modo a evidenciar tais características. Alguns exemplos são: a transformação para o domínio das frequências, *wavelets*, análise das componentes principais e autocorrelação (Bagnall et al., 2012).

Em Silva et al. (2013b) nós propomos o uso de gráficos de recorrência (em inglês, *recurrence plots* – RP) para a representação de séries temporais em problemas de classificação. Os gráficos de recorrência (Eckmann et al., 1987) são utilizados em avaliações qualitativas de séries temporais em sistemas dinâmicos, explicitando padrões e mudanças estruturais ocultos nos dados. Gráficos de recorrência consistem em uma ferramenta para categorizar como a similaridade entre subsequências varia no tempo. Neste trabalho, tem-se a hipótese de que esse tipo de informação pode ser útil na classificação de séries temporais, dado que padrões de recorrência são regularidades frequentemente associadas a comportamentos interessantes. Um comportamento recorrente indica a presença de um mecanismo interno responsável por gerar tais padrões. Esse tipo de mecanismo é oposto à geração aleatória de dados, em que uma série temporal não terá tal tipo de padrão. A



(a) Hierárquico (b) k-Medoids

(c) Espectral

Figura A.11: Comparação dos resultados obtidos pela medida CID e distância euclidiana na tarefa de agrupamento de dados pelos algoritmos Hierárquico, k-Medoids e Espectral.

representação explícita de tais regularidades pode revelar os mecanismos que geraram os dados e, portanto, são um recurso potencialmente útil para a caracterização e classificação destes dados.

Os gráficos de recorrência são representações binárias em que cada ponto do gráfico revela se um estado (subsequência da série temporal) é recorrente ou não. Na análise de séries temporais utilizando essa representação, são extraídas informações sobre a quantidade de repetições ocorridas, a duração das repetições, grau de aleatoriedade da série, entre outras informações. Essa abordagem não leva em conta, por exemplo, a localização das recorrências. Em outra direção, neste trabalho é proposto a comparação direta das imagens formadas pelos gráficos de recorrência por meio da medida de distância Campana-Keogh (CK-1) (Campana e Keogh, 2010), dando origem a medida de distância RPCD – *Recurrence Patterns Compression Distance*. A seguir, são apresentadas as definições referentes aos gráficos de recorrência e a medida CK-1 utilizada para comparar as imagens.

Formalmente, um RP pode ser definido de acordo com a Equação A.10, em que N é a quantidade de estados, $\vec{x}_i$ e $\vec{x}_j$ são as subsequências observadas nas posições i e j, respectivamente, $||\cdot||$ é a norma entre as observações (por exemplo, norma euclidiana), ϵ é um limiar para a proximidade e θ a função de *Heaviside*, definida pela Equação A.11.

$$R_{i,j} = \theta(\epsilon - ||\vec{x}_i - \vec{x}_j||), \vec{x}_\cdot \in \Re^m, i, j = 1..N \tag{A.10}$$

$$\theta(z) = \begin{cases} 0, \text{ se } z < 0 \\ 1, \text{ caso contrário} \end{cases} \tag{A.11}$$

Na Equação A.10 é expresso que, caso a trajetória m-dimensional da série temporal no tempo j seja próxima – em termos de uma vizinhança pré-definida – da subsequência observada no momento i, haverá um valor 0 na posição (i, j) da chamada matriz de recorrência. Caso contrário, o valor será 1. Na representação gráfica, uma imagem de $N \times N$ *pixels* é definida de modo que os *pixels* que correspondem as posições da matriz com valores 0 são geralmente pretos e os de valor 1 são brancos. Na Figura A.12, são exibidos exemplos de gráficos de recorrência gerados a partir de séries temporais com diferentes graus de aleatoriedade.

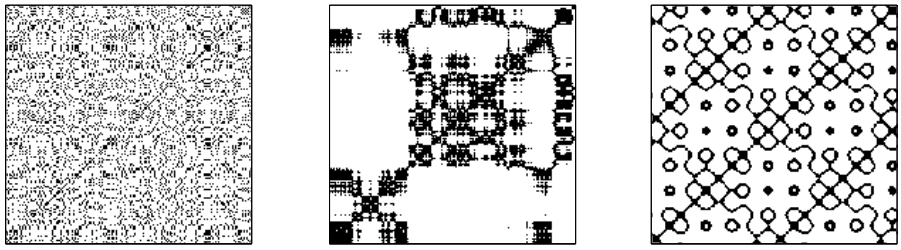


Figura A.12: Alguns exemplos de gráficos de recorrência: ruído totalmente aleatório (esquerda); caminhada aleatória (meio); composição de senos e cossenos (direita).

Embora seja um método simples, ele requer a especificação de um parâmetro de limiar de proximidade que define o tamanho de uma vizinhança em que duas subsequências são consideradas semelhantes. Entretanto, a determinação de um valor adequado para esse parâmetro não é intuitiva. Na prática, esse problema é contornado com o uso de heurísticas para a escolha do limiar. Por exemplo, um limiar de 10% da maior distância observada ou um valor que resulta em uma determinada porcentagem de pontos pretos. No entanto, essas são heurísticas locais que utilizam a informação de um único gráfico de recorrência para definir o valor do limiar. Portanto, é difícil generalizar um limiar que seja consistente com vários gráficos de recorrência.

A fim de eliminar o parâmetro de proximidade, é possível fazer uso da informação de cor. O gráfico de recorrência, nesse caso, deixa de ser uma ferramenta para analisar as recorrências considerando vizinhanças e passa a ser uma ferramenta para analisar o quão

perto cada par de subsequências está no espaço de trajetórias (Iwanski e Bradley, 1998). Para isso, a imagem é gerada com tons de cinza ou outros mapas de cor, de modo que as distâncias são representadas pelas cores. Assim, a imagem é uma representação direta da matriz de distâncias. Essa representação pode ser encontrada com os nomes gráficos de recorrência *unthresholded*, gráficos de distância ou matrizes de auto-similaridade. Na Figura A.13, é exibido um exemplo de gráficos de recorrência com e sem a utilização do limiar de proximidade para uma mesma série temporal.

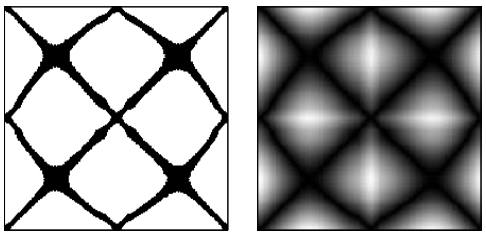


Figura A.13: Exemplo de gráficos de recorrência com limiar de proximidade (esquerda) e sem limiar de proximidade (direita), ambos gerados a partir de uma mesma série temporal.

A definição de RP dada na Equação A.10 possui um segundo parâmetro, m, que define a dimensão da trajetória. Esse parâmetro também é conhecido como dimensão embarcada. Nos experimentos executados neste trabalho, foi escolhido o valor de $m = 1$ para todos os conjuntos de dados. Isso significa que, ao invés de considerar m pontos consecutivos da série para analisar a trajetória, foram consideradas apenas subsequências unitárias. Foi escolhido fixar $m = 1$ uma vez que a estrutura dos gráficos de recorrência parece mudar muito pouco à medida que se varia tal parâmetro. Essa opinião também é suportada por outro trabalho da literatura (Iwanski e Bradley, 1998).

Após a geração da imagem do gráfico de recorrência de diferentes séries temporais, é necessário utilizar uma medida que seja adequada para a comparação entre as imagens para realizar a tarefa de classificação. Para isso, foi utilizada a medida Campana-Keogh (CK-1) (Campana e Keogh, 2010). A CK-1 é uma medida baseada em compressão que utiliza conceito teórico da complexidade de Kolmogorov, que visa quantificar a informação de *strings* ou objetos de maneira absoluta. A complexidade condicional de Kolmogorov $K(x|y)$ de x para y, é definida pelo tamanho do menor programa capaz de computar x dado y como entrada, como uma Máquina de Turing. Enquanto $K(xy)$ representa o tamanho do menor programa capaz de gerar uma saída das entradas x e y concatenadas. Assim, a distância entre duas *strings* x e y é dada pela Equação A.12.

$$d(x, y) = \frac{K(x|y) + K(y|x)}{K(xy)} \tag{A.12}$$

Entretanto, a complexidade de Kolmogorov não é computável na prática e, por isso, diversos pesquisadores têm proposto aproximações para esse valor por meio de algoritmos

de compressão. Nesse sentido, a medida CK-1 utiliza compressão de vídeo (algoritmo MPEG-1) para comparar a similaridade entre duas imagens. Assim, para medir a distância d entre duas imagens x e y, a medida CK-1 é definida de acordo com a Equação A.13.

$$d(x,y) = \frac{C(x|y) + C(y|x)}{C(x|x) + C(y|y)} - 1, \tag{A.13}$$

em que $C(a|b)$ é o tamanho da compressão de vídeo utilizando MPEG-1 composto por dois quadros, a e b, nessa ordem. De acordo com Campana e Keogh (2010), MPEG-1 e a maioria dos algoritmos de codificação de vídeo realizam a compressão para encontrar padrões recorrentes dentro de um quadro (compressão intra-quadro) e/ou entre os quadros (compressão inter-quadros). Desse modo, quando a e b são duas imagens similares, a etapa de compressão inter-quadros deve ser capaz de explorar esse fato para produzir um arquivo de menor tamanho. Assim, quanto maior a taxa de compressão, a similaridade entre as imagens também será maior e o tamanho do arquivo será menor. É importante destacar que devido ao fato do vídeo digital ser uma importante aplicação comercial, diversas pesquisas têm sido realizadas para se alcançar uma alta taxa de compressão na codificação de vídeo, próxima da complexidade condicional de Kolmogorov.

Uma visão geral do funcionamento da medida proposta RPCD em um algoritmo de classificação de séries temporais é apresentada na Figura A.14. Inicialmente um conjunto de treinamento de séries temporais é transformado em um conjunto de imagens obtidas a partir da representação gráfico de recorrências (RP). Na fase de classificação, cada série temporal desconhecida é transformada em uma imagem de gráfico de recorrências e comparada às outras imagens do conjunto de treino por meio da medida de distância CK-1. Ao encontrar a imagem com a menor distância CK-1, pode-se retornar a série temporal correspondente a imagem e, consequentemente, associar a sua classe para a série desconhecida.

A medida RPCD foi avaliada na tarefa de classificação em um conjunto de *benchmark* com 38 conjuntos de dados de séries temporais amplamente utilizados na literatura (Chen et al., 2015). Foram realizadas comparações com as medidas de distância mais populares em séries temporais e consideradas estado-da-arte: distância euclidiana (ED) e *Dynamic Time Warping* (DTW). Para facilitar a visualização dos resultados[6], as taxas de acurácia obtidas neste experimento são representadas graficamente na Figura A.15. Nestes gráficos, cada conjunto de dados é representado por um ponto, em que a coordenada y expressa a acurácia obtida pela RPCD e a coordenada x indica a acurácia obtida pelo método competidor. Assim, pontos acima da diagonal principal referem-se a conjuntos de dados em que a RPCD obteve melhor resultado.

[6] A descrição detalhada do resultado obtido em cada conjunto de dados pode ser consultada no *website* do artigo, disponível no endereço eletrônico `http://sites.labic.icmc.usp.br/dfs/rpcd/`

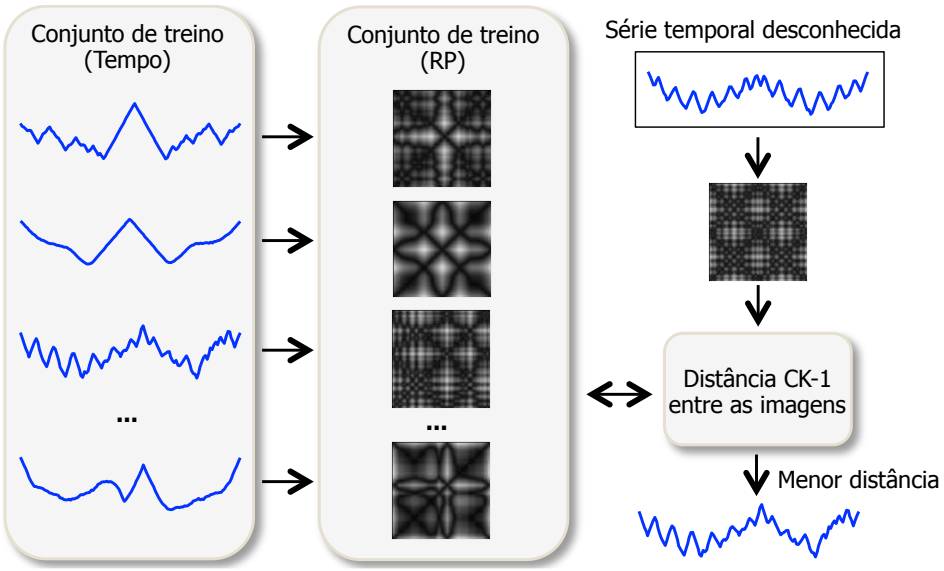


Figura A.14: Visão geral do funcionamento da medida *Recurrence Pattern Compression Distance* (RPCD) em um algoritmo de classificação.

Pode-se observar que a RPCD é muito competitiva com a distância euclidiana e a DTW. Dentre os 38 conjuntos de dados utilizados nos experimentos, a RPCD foi superior à distância euclidiana em 28 $(73,68\%)$ conjuntos de dados e superior à DTW em 20 $(52,63\%)$. Por ser uma medida de distância para busca por similaridade em séries temporais, pode ser aplicada em diferentes tarefas como agrupamento e classificação semissupervisionado.

A.2.3 Classificação Baseada em Atributos de Textura

Em Souza et al. (2014) nós propomos realizar a tarefa de classificação de séries temporais a partir do uso de descritores de textura extraídos das imagens dos gráficos de recorrências gerados a partir das séries no tempo. Neste trabalho, tem-se a hipótese de que as imagens obtidas a partir dos gráficos de recorrências possuem características de textura importantes para a classificação e que podem ser identificadas a partir de métodos que realizam a extração explícita destas informações e que podem não ter sido consideradas pela RPCD ao utilizar a medida CK-1. Desse modo, o algoritmo proposto neste trabalho e denominado *Texture Features from Recurrence Patterns* (TFRP) utiliza o classificador Máquina de Vetores de Suporte (SVM) para realizar a classificação de séries temporais após a extração de informações de textura fornecidas por quatro diferentes métodos: *i) Local Binary Pattern* (LBP), *ii) Grey Level Co-occurrence Matrix* (GLCM), *iii) Gabor filters* e *iv) Segmentation-based Fractal Texture Analysis* (SFTA).

A Figura A.16 apresenta uma visão geral do funcionamento do algoritmo TFRP para a classificação de séries temporais. O primeiro passo do algoritmo é mudar a representação temporal dos dados para os gráficos de recorrências, tanto em dados rotulados como em

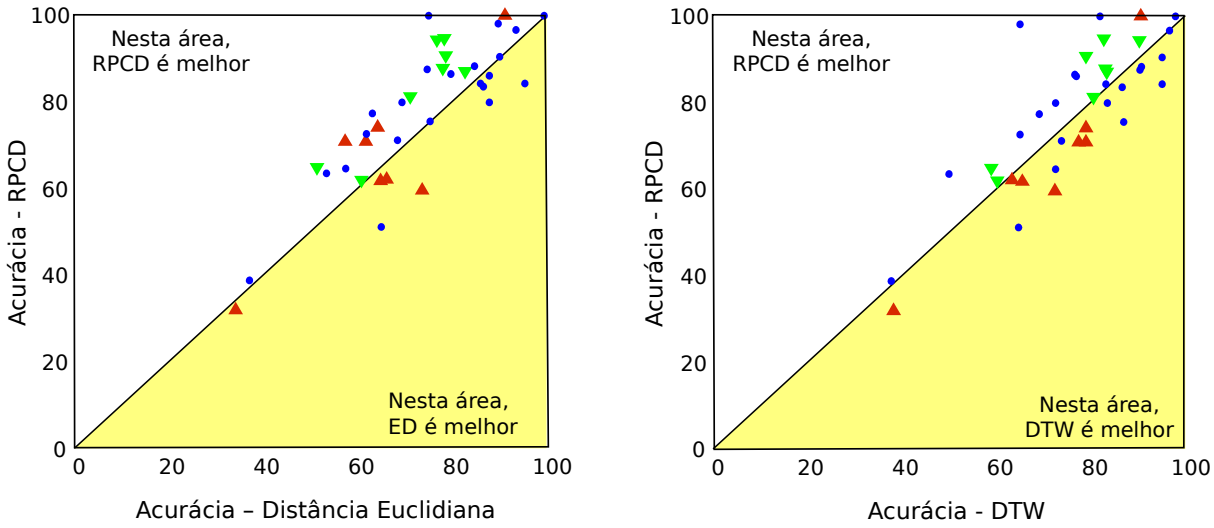


Figura A.15: Representação gráfica dos resultados obtidos pela RPCD contra a distância euclidiana e DTW. Cada ponto representa um conjunto de dados diferente. Os pontos acima da diagonal principal representam os conjuntos de dados em que a RPCD superou distância euclidiana (esquerda) ou DTW (direita). Os símbolos ▼, ▲ e ● representam dados obtidos de contorno de figuras, de movimentos humanos e dos demais tipos de produção de dados, respectivamente

não rotulados. Após a mudança de representação e geração das imagens, é realizada a etapa de extração de características de textura pelos métodos LBP, GLCM, Gabor e SFTA. Por fim, estas características são utilizadas na indução de um classificador SVM que é responsável pela classificação de dados não rotulados. Uma descrição sobre os métodos de extração de características de textura é apresentada a seguir.

Apesar do forte conceito intuitivo atribuído pelas pessoas, a textura é uma propriedade de difícil definição e que carece de consenso para tal formalismo (Ebert et al., 2003). De modo geral, a textura pode ser entendida como os diferentes aspectos de uma superfície. No contexto de imagens, Haralick (1979) define textura a partir de medidas de uniformidade, regularidade, aspereza, densidade, intensidade, entre outros descritores da imagem. Assim, a textura é uma importante característica que pode ser utilizada para classificar e reconhecer objetos ou padrões em imagens a partir de variações locais em valores de *pixels* que se repetem de maneira regular ou aleatória ao longo da imagem. Com o objetivo de caracterizar e quantificar a textura, é necessário o uso de descritores. Neste trabalho, foram utilizados quatro métodos:

Local Binary Pattern (LBP). O LBP é um dos descritores de textura mais populares na literatura de análise e classificação de imagens (Doost e Rahebi, 2012). Este método foi introduzido por Ojala et al. (1996) e utiliza a escala de cinza e um operador LBP em uma vizinhança circular de *pixels* para descrever a textura local de uma imagem. Basicamente, a intensidade de um *pixel* central g_c é comparada com P *pixels* g_p em uma vizinhança circular de acordo com um raio R. Com o objetivo de garantir que os

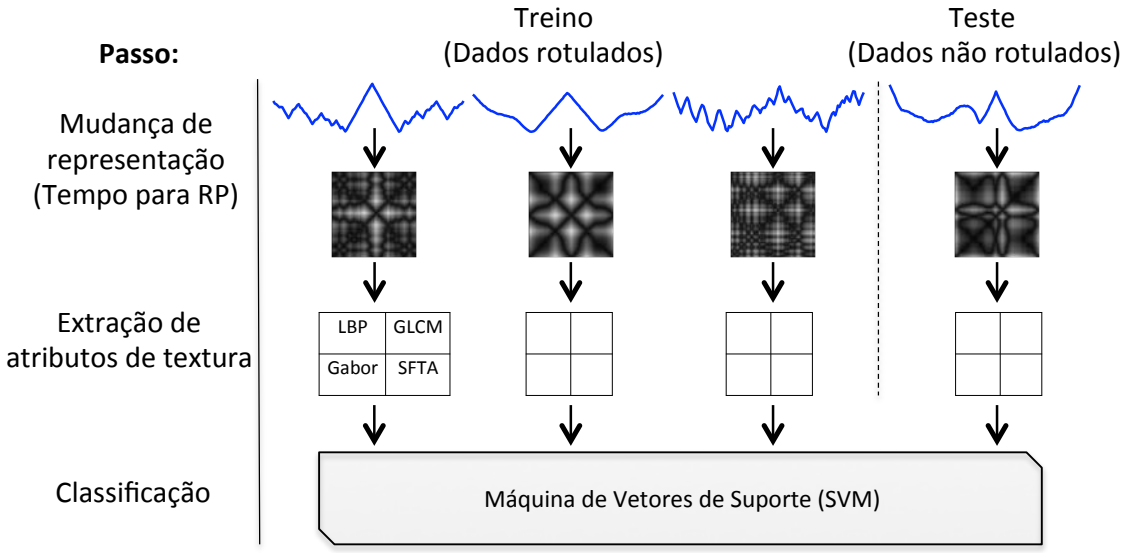


Figura A.16: Visão geral do funcionamento do algoritmo *Texture Features from Recurrence Patterns* (TFRP) para a classificação de séries temporais.

valores sejam independentes de alterações na escala de cinza utilizada por cada imagem, considera-se somente os sinais s dos resultados ao invés dos valores exatos. Em sua versão original, o operador LBP é definido como $LBP_{P,R} = \sum_{p=0}^{P-1} s(g_p - g_c)2^p$. Entretanto, neste trabalho foram avaliadas algumas extensões deste operador por apresentarem melhores resultados, como o operador uniforme $LBP^{u2}_{P,R}$, o operador invariante a rotação $LBP^{ri}_{P,R}$ e o operador uniforme invariante a rotação $LBP^{riu2}_{P,R}$. Nos experimentos realizados, foram considerados uma vizinhança com 8 *pixels* ($P = 8$) e os raios $R = \{1, 2, 3\}$. Uma descrição detalhada sobre estes operadores pode ser encontrada em Ojala et al. (2002).

Grey Level Co-occurrence Matrix (GLCM). As matrizes de coocorrência GLCM ou descritores de Haralick (Haralick et al., 1973), constituem um método estatístico para descrever a textura de uma imagem a partir de estatísticas de segunda ordem extraídas de matrizes que contam as ocorrências de níveis de cinza em uma imagem. O método utiliza a dependência espacial entre *pixels* e extrai estatísticas que consideram a direção, distância e relação entre pares de *pixels*. O método descreve em uma matriz a frequência em que um tom de cinza ocorre em uma determinada região e a sua relação com outro tom de cinza. Nos cálculos para obtenção dessas matrizes, a variação da direção entre *pixels* vizinhos é feita em termos angulares, onde são utilizados quatro direcionamentos: $0°, 45°, 90°$ e $135°$. Neste trabalho foram avaliadas 20 diferentes relações entre os tons de cinza: segundo momento angular, entropia, dissimilaridade, contraste, diferença inversa, correlação, homogeneidade, autocorrelação, *cluster shade, cluster proeminence*, probabilidade máxima, soma dos quadrados, qui-quadrado, soma média, soma da variância, soma da entropia, diferença do vetor de variância, coeficiente de máxima correlação, inverso da diferença normalizado e inverso da diferença do momento normalizado. Estas estatísticas

foram computadas em todas as 4 possíveis direções e com a distância $d = \{1, 2, 3, 4, 5\}$ entre *pixels* vizinhos. Uma descrição detalhada destas estatísticas pode ser encontrada em Haralick et al. (1973); Baraldi e Parmiggiani (1995).

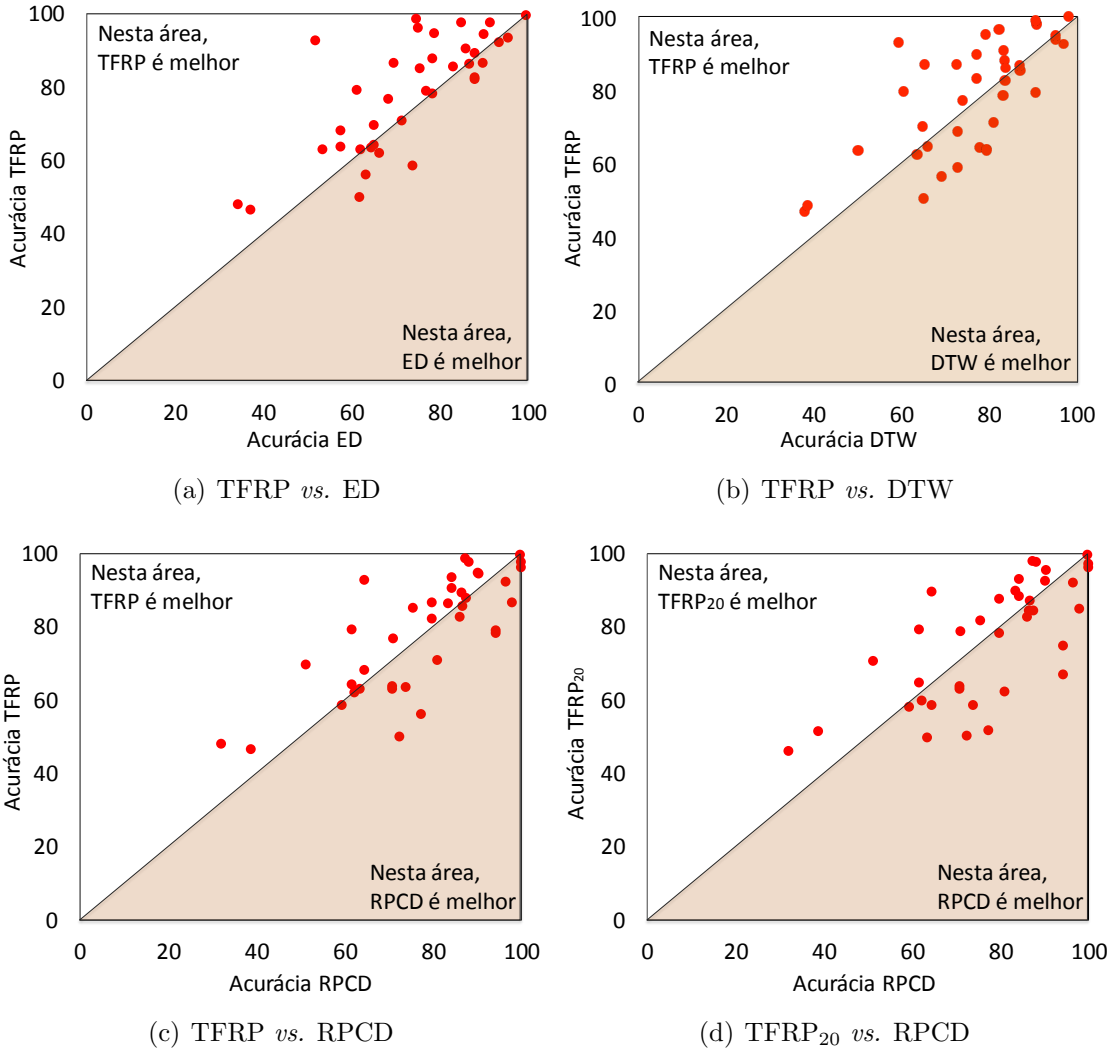
Gabor filters. Os filtros de Gabor (Manjunath e Ma, 1996) são um conjunto de funções senoidais complexas, bidimensionais, moduladas por uma função gaussiana também bidimensional. Essas funções têm como objetivo extrair atributos para caracterizar diferentes tipos de texturas em imagens, descritas pela frequência e orientação já definidas pelas funções senoides. Estes filtros são eficientes no processo de análise de textura a partir de frequências espaciais e simulam algumas características do sistema visual humano. Dada uma imagem, para a extração de atributos de textura é necessário inicialmente realizar a convolução da imagem utilizando a Transformada Wavelet de Gabor com um conjunto de filtros de Gabor de diferentes orientações e frequências espaciais. Assim, cada ponto da imagem resultante representa uma informação sobre a relação espacial entre *pixels* vizinhos. O vetor com atributos de textura é dado pelos resultados de todas as transformadas wavelets (Idrissa e Acheroy, 2002). Neste trabalho considerou-se 5 escalas wavelets e 6 orientações de filtros para a convolução da imagem.

Segmentation-based Fractal Texture Analysis (SFTA). O método SFTA (Costa et al., 2012) consiste em decompor uma imagem de entrada na escala de cinza em um conjunto de imagens binárias. Na decomposição da imagem é utilizada a técnica *Two-Threshold Binary Decomposition* (TTBD), também proposta pelos autores. Para cada imagem binária resultante, é calculada a dimensão fractal nas regiões da borda da imagem. Além disso, são calculados o nível médio de cinza na região e o número de *pixels*. A partir da dimensão fractal é possível mensurar o grau de irregularidade de um objeto e assim, descrever a textura de uma imagem. Esta abordagem se mostrou eficiente em tarefas de classificação de imagens e recuperação de imagens por conteúdo, superando métodos tradicionais como GLCM e *Gabor filters*.

O algoritmo proposto TFRP busca agregar as vantagens dos diferentes métodos de extração de atributos de textura na tarefa de classificação de séries temporais. Por isso, foi considerado um vetor de características com atributos provenientes dos quatro métodos de extração. Além disso, em experimentos preliminares foi verificado que nenhum método utilizado isoladamente foi capaz de propiciar bons resultados. Entretanto, ao considerar todos os métodos, obtêm-se um vetor de características de alta dimensionalidade com 823 atributos no total. Por isso, também foi considerado o uso dos métodos de seleção de atributos *Correlation-based Feature Selection* (CFS) (Hall, 1999) e *ReliefF* (Kononenko, 1994). Para o método *ReliefF* foram considerados os limiares de corte de 5%, 10% e 20% de todos os 823 atributos.

Os resultados obtidos pelo algoritmo TFRP em comparação com a distância euclidiana (ED), *Dynamic Time Warping* (DTW) e *Recurrence Patterns Compression Distance* (RPCD) na tarefa de classificação de 38 conjuntos de dados de séries temporais (Chen

et al., 2015) são apresentados resumidamente na Figura A.17. Nesta imagem também são apresentados os resultados obtidos pelo método TFRP utilizando somente 20% ou 164 do total de 823 atributos. Entre as configurações avaliadas utilizando os métodos de seleção de atributos, o método *ReliefF* com limiar de 20% foi o que apresentou os melhores resultados. Estes resultados estão identificados como TFRP_{20}.



(a) TFRP *vs.* ED

(b) TFRP *vs.* DTW

(c) TFRP *vs.* RPCD

(d) TFRP_{20} *vs.* RPCD

Figura A.17: Representação gráfica dos resultados obtidos pelo método proposto TFRP em comparação com outros métodos na tarefa de classificação de séries temporais.

Em geral, os resultados obtidos pelo algoritmo TFRP são bastante competitivos e superam algoritmos estado-da-arte na classificação de séries temporais como o 1-Vizinho Mais Próximo utilizando a distância euclidiana e a distância DTW. O método também supera sensivelmente os resultados apresentados por nossa proposta anterior, RPCD. Em termos de acurácia média obtida nos 38 conjuntos de dados, temos os seguintes resultados: 73,27% (ED), 76,29% (DTW), 77,57% (RPCD) e 78,71% (TFRP). Em alguns conjuntos de dados específicos é possível observar uma diferença mais significativa. Por exemplo, no

conjunto *OSULeaf* nota-se que o método TFRP apresentou uma acurácia 30% superior ao RPCD e próximo de 20% superior nos conjuntos *Adiac* e *InlineSkate*.

A.3 Aprendizado Ativo para Rotulação de Dados de Treino

Em muitas aplicações reais como as que fazem uso de sensores e dispositivos de medições, uma grande quantidade de dados sequenciais é gerada com um custo bastante reduzido. Entretanto, para que algoritmos de aprendizado de máquina possam realizar a tarefa de classificação destes dados, é necessário previamente que uma porção considerável de dados esteja rotulada para a indução do classificador. Por ser um processo essencialmente manual na maioria dos casos, a rotulação de dados pode ser cara, demorada ou altamente dependente de um especialista de domínio. Desse modo, nestes casos em que dados não rotulados são abundantes mas a aquisição de seus rótulos é cara, métodos de aprendizado ativo podem contribuir na redução do esforço despendido pelo especialista responsável pelo processo de rotulação dos dados. Métodos de aprendizado ativo selecionam para rotulação uma quantidade reduzida de dados que sejam representativos de toda a distribuição, de modo a evitar a análise de todo o conjunto de dados. Portanto, estes métodos permitem que os classificadores obtenham uma acurácia satisfatória mesmo com uma menor quantidade de dados rotulados, minimizando o custo de obtenção de seus rótulos (Settles, 2010).

Em geral, os métodos de aprendizado ativo iniciam com uma pequena quantidade de dados rotulados e selecionam alguns exemplos informativos de um conjunto com dados não rotulados para posteriormente solicitar os seus rótulos a um oráculo que pode ser, por exemplo, um anotador humano (Settles et al., 2008). Diferentes estratégias podem ser utilizadas para a seleção dos exemplos como a amostragem baseada em incerteza, consulta por comitê, mudança esperada do modelo, redução de erro esperada, redução da variância, densidade ponderada, entre outras. Uma ampla revisão destas estratégias pode ser consultada em Settles (2010). É interessante observar que todas estas estratégias dependem de um modelo de classificação para que possam ser utilizadas. Em outras palavras, estas abordagens exigem a existência de um conjunto de exemplos rotulados para a indução de um modelo de classificação inicial que será utilizado para o cálculo de informatividade/importância de exemplos não rotulados e que serão selecionados para rotulação na etapa de aprendizado ativo.

Diferentemente da configuração popularmente utilizada na literatura, em Souza et al. (2016) nós temos interesse no cenário onde o processo de aprendizado ativo é realizado em um conjunto de dados totalmente não rotulado desde o início do processo. Desse modo, não é possível assumir a existência de um classificador inicial que possa fornecer informações sobre exemplos não rotulados com base em algum conhecimento prévio. Embora os métodos de aprendizado ativo sejam amplamente utilizados em diversos domínios

de aplicação, o uso de métodos que consideram somente dados não rotulados não tem recebido muita atenção na literatura. Em uma revisão sistemática realizada por Hu et al. (2010) em mais de 200 artigos que utilizam métodos de aprendizado ativo, é apontado que 94% dos trabalhos utilizam amostragem aleatória ou não especificam a estratégia utilizada para a seleção inicial de exemplos em conjuntos de dados não rotulados. Em uma revisão realizada neste trabalho, nota-se que nos últimos anos o cenário ainda é o mesmo. Acredita-se que esta prática se deve a falta de um estudo mais amplo que mostre que métodos simples podem superar os resultados obtidos pela amostragem aleatória. Dessa maneira, o objetivo da pesquisa realizada em Souza et al. (2016) é apresentar uma revisão sobre diferentes métodos que podem ser utilizados para esta tarefa em conjunto com uma ampla avaliação experimental.

Para um exemplo concreto de aplicação que necessita de técnicas de aprendizado ativo para a rotulação de um conjunto de dados inicial, também conhecido como conjunto de treino ou treinamento, considere o sensor laser identificador de insetos discutido no decorrer desta tese. O sensor é capaz de coletar uma grande quantidade de dados, mas o processo de rotulação de um conjunto de treinamento inicial é caro e demanda de especialistas. Por conveniência, assume-se o uso de dados coletados em laboratório para a construção de um modelo inicial. Entretanto, no uso do sensor em campo, os dados previamente coletados em laboratório podem ser diferentes dos dados observados no meio ambiente, prejudicando a acurácia de classificação. Uma solução prática para este problema é realizar uma etapa de coleta de dados com o sensor *in loco* por um período de tempo e selecionar uma amostra destes dados para a rotulação por um especialista. Assim, é possível induzir um classificador inicial com dados que correspondam exatamente àqueles observados na região em que o sensor será utilizado.

Existem diversos outros exemplos de problemas em que a coleta de dados não rotulados é simples, mas a sua rotulação pode ser cara. Na medicina, procedimentos não invasivos de eletrocardiograma (ECG) e eletroencefalograma (EEG) são relativamente simples e capazes de registrar uma enorme quantidade de dados. Entretanto, a rotulação destes dados é um processo tedioso que exige um clínico treinado. Na indústria, pode-se citar como exemplo as usinas de energia elétrica ou indústrias automotivas. Nestas indústrias, sensores são responsáveis por coletar uma grande quantidade de dados para o monitoramento de processos de fabricação. Entretanto, para realizar a tarefa de detecção de falhas é necessário que anteriormente um especialista tenha analisado os dados para a identificação de características que antecedem uma falha.

Em Souza et al. (2016) é fornecida uma ampla avaliação experimental sobre métodos de aprendizado ativo não supervisionados para a construção de um conjunto de treinamento inicial a partir de dados totalmente não rotulados. Devido à popularidade de dados sequenciais em diferentes aplicações e a facilidade de representar outros tipos de dados como dados sequenciais (por exemplo, os dados de imagem anteriormente discutidos neste

apêndice e apresentados na Figura A.5), a avaliação experimental conduzida neste trabalho considera 24 conjuntos de dados de séries temporais da UCR (Chen et al., 2015). Em relação aos métodos de aprendizado ativo, foram avaliados métodos de amostragem baseados em algoritmos de agrupamento e a partir de medidas de centralidade extraídas de grafos. Estes métodos foram comparados com a amostragem aleatória, popularmente aceito na literatura e a técnica *Farthest-First Traversal* (Baram et al., 2004). A técnica *Farthest-First Traversal* busca selecionar k exemplos mais distantes um do outro a partir da escolha inicial aleatória do primeiro exemplo.

A ideia geral dos algoritmos de aprendizado ativo baseados em agrupamento é realizar a etapa de agrupamento de dados considerando o número de grupos k a ser encontrado como o mesmo número de exemplos que se deseja rotular. Dado que os algoritmos de agrupamento formam grupos baseados na similaridade observada entre diferentes conjuntos de exemplos, espera-se que a estrutura encontrada no agrupamento seja representativa para o processo de rotulação. Mais especificamente, seleciona-se o exemplo mais representativo de cada grupo para ser rotulado. Neste trabalho, foi considerado que o exemplo mais representativo de um grupo é aquele mais próximo do centroide do grupo. Como cada algoritmo de agrupamento possui um viés, foram avaliados 5 diferentes abordagens para o agrupamento dos dados e seleção de exemplos: k-Means, k-Medoids, Modelo de Misturas Gaussianas (GMM), método de agrupamento Hierárquico Aglomerativo e Density Peak (Rodriguez e Laio, 2014).

A segunda abordagem avaliada neste trabalho considera a construção de um grafo a partir de dados não rotulados, em que os nós representam os exemplos e as arestas representam a similaridade entre os exemplos. A extração não supervisionada de propriedades da topologia do grafo permite medir a importância de um nó no grafo e, dessa maneira, realizar a seleção de exemplos representativos para a rotulação. Existem diferentes maneiras de construir um grafo, neste trabalho foi utilizada a estratégia Mutual k-Nearest Neighbor devido aos bons resultados obtidos por essa estratégia na classificação semissupervisionada de séries temporais (De Sousa et al., 2014, 2015). Entre as informações que podem ser extraídas do grafo, foram consideradas três medidas de centralidade, sendo: *i*) Grau, *ii*) *PageRank* e *iii*) *Betweenness*. Desse modo, os nós do grafo são ordenados em ordem decrescente de acordo com os valores obtidos pelas medidas e são selecionados os k primeiros exemplos para a rotulação por um especialista. A escolha destas medidas é baseada no uso por outros trabalhos da literatura (Newman, 2010; Araujo e Zhao, 2013).

Em nosso cenário de avaliação, é disponibilizado somente um conjunto de dados com n exemplos $D^U = \{(\overrightarrow{x_1}, ?), (\overrightarrow{x_2}, ?), \ldots, (\overrightarrow{x_n}, ?)\}$, em que $\overrightarrow{x_i}$ é um vetor d-dimensional e os rótulos de classe y são desconhecidos. O principal objetivo é selecionar uma amostra com $b\%$ dos exemplos não rotulados para ser rotulado por um oráculo e construir um conjunto de treinamento inicial de boa qualidade para a indução de um classificador. Como os conjuntos de dados avaliados neste trabalho possuem divisões naturais em treino e teste,

para cada conjunto de dados foi considerado que os exemplos não rotulados D^U onde será realizada a amostragem são provenientes da partição de treino, enquanto os dados de teste são utilizados para a avaliação da estratégia. A Figura A.18 apresenta uma visão geral da configuração experimental considerada na avaliação deste trabalho.

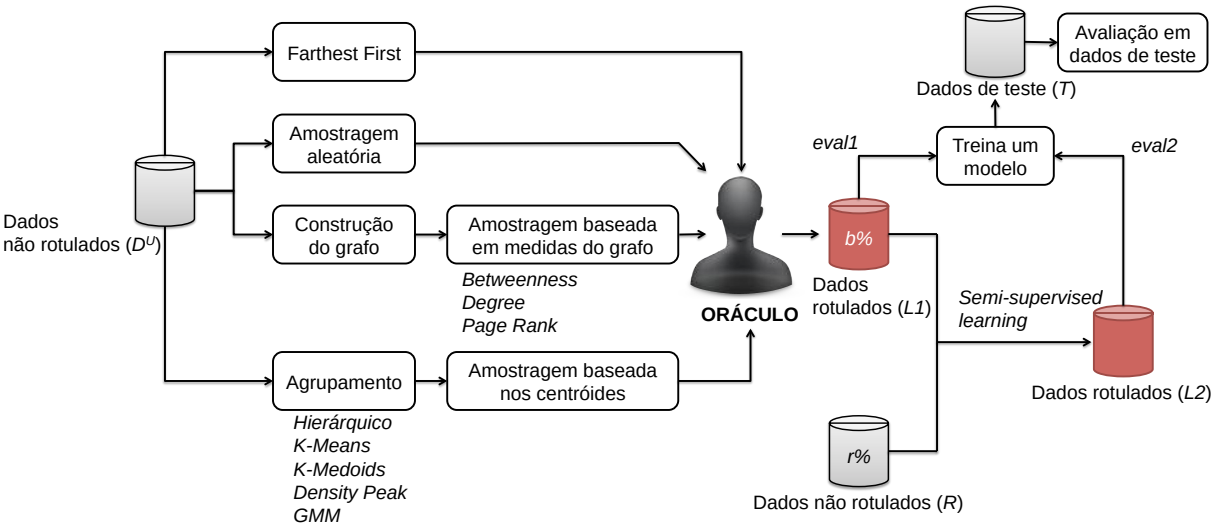


Figura A.18: Visão geral da configuração experimental utilizada na avaliação de diferentes algoritmos para a seleção inicial de exemplos utilizando aprendizado ativo.

A partir do conjunto de exemplos não rotulados D^U, neste trabalho são consideradas quatro possibilidades para a seleção de $b\%$ dos exemplos para ser rotulado por um oráculo, descritas a seguir:

1. Utilizar o algoritmo *Farthest-First Traversal*;
2. Realizar a amostragem aleatóra dos dados;
3. Construir um grafo a partir dos dados não rotulados e extrair as medidas de centralidade do grafo como Grau, *Betweenness* e *PageRank*. Após calcular as medidas, ordenar os exemplos de acordo com os valores obtidos e selecionar os primeiros $b\%$ dos exemplos;
4. Realizar o agrupamento dos dados não rotulados considerando $b\%$ como o número de grupos a ser encontrado pelos algoritmos. Após o processo de agrupamento, selecionar o exemplo mais similar ao centroide de cada grupo encontrado como exemplos representativos da estrutura dos dados.

Na avaliação experimental, variou-se a quantidade de exemplos selecionados pelos algoritmos para ser rotulado de 5% a 95% do tamanho da partição de treinamento de cada conjunto de dados. Assim, para conjuntos de dados de tamanho reduzido como Haptics que possui apenas 155 exemplos de treinamento, a configuração inicial da avaliação

considerou neste caso somente 8 exemplos rotulados pelo oráculo. Por outro lado, em conjuntos de dados maiores como NonInvasiveFetalECG Thorax que possui 1800 exemplos de treinamento, a configuração inicial considerou 90 exemplos rotulados.

Após o processo de amostragem dos exemplos e respectiva rotulação de $b\%$ dos exemplos, tem-se dois conjuntos de exemplos. O primeiro é composto por $b\%$ de exemplos rotulados ($L1$) e o segundo conjunto (R) consiste de $r\%$ dos exemplos remanescentes de D^U que não foram selecionados para o processo de rotulação, de modo que $D^U = L1 \bigcup R$. A partir dos conjuntos $L1$ e R, são realizadas duas diferentes avaliações:

- *eval1:* um classificador é induzido a partir dos dados de $L1$ e o seu desempenho é avaliado em dados não vistos T, provenientes da partição de teste de cada conjunto de dados. Esta avaliação permite verificar se o método de aprendizado ativo foi capaz de selecionar bons exemplos para a indução do classificador. Na verdade, foi avaliado o desempenho do classificador 1-Nearest Neighbor (1NN) com a distância euclidiana. Este algoritmo utiliza a abordagem *lazy* que dispensa a etapa de treinamento, pois os próprios exemplos são utilizados na classificação. O algoritmo 1NN foi selecionado por ser uma abordagem simples e considerado estado-da-arte na literatura de classificação de séries temporais (Ding et al., 2008);

- *eval2:* a partir dos exemplos rotulados pelo oráculo ($L1$), os exemplos remanescentes não rotulados R são classificados utilizando algoritmos semissupervisionado transdutivos. Mais especificamente, foram avaliados os algoritmos *Gaussian Function Harmonic Field* (GFHF) (Zhu et al., 2003), *Learning with Local and Global Consistency* (LLGC) (Zhou et al., 2004) e *Self-Training* (Zhu e Goldberg, 2009). Desse modo, todos os exemplos iniciais D^U são rotulados e constituem o conjunto de dados $L2$. Para analisar se uma quantidade maior de dados rotulados obtidos por meio de transdução pode melhorar a acurácia de classificação, é avaliado o desempenho do classificador 1NN que utiliza os dados do conjunto $L2$ para a classificação dos dados em T.

A Tabela A.14 apresenta a acurácia média obtida pelas diferentes abordagens consideradas neste trabalho, considerando os resultados com a variação entre 5% a 95% dos dados rotulados. Dado um conjunto de dados, o melhor resultado é destacado em negrito. Estes resultados se referem a avaliação *eval1* discutida na Figura A.18. Os resultados indicam que de um total de 24 conjuntos de dados avaliados, os melhores resultados são obtidos em geral pelos métodos baseados em agrupamento de dados utilizando os algoritmos k-Means e Hierárquico. Somente para os conjuntos de dados Haptics e StarLightCurves os melhores resultados foram obtidos pelos algoritmos Density Peak e Farthest First Traversal, respectivamente. Dado que a amostragem aleatória é a mais utilizada na literatura, esta abordagem foi utilizada como comparativo com outros métodos. Pode-se observar que os

algoritmos k-Medoids, k-Means, Hierárquico e GMM apresentam resultados superiores à amostragem aleatória em uma quantidade expressiva de conjuntos de dados.

Tabela A.14: Resultados obtidos pelos métodos de amostragem baseados em agrupamento de dados e medidas de grafos considerando a acurácia média com 5% a 95% de dados rotulados.

Dataset	**Rand.**	**k-Medoids**	**Hierár.**	**k-Means**	**Density Peak**	**GMM**	**Farth. First**	**Betw.**	**Grau**	**Page Rank**
50words	0,547	0,566	**0,582**	0,573	0,529	0,563	0,482	0,519	0,502	0,532
Adiac	0,493	0,514	0,507	**0,517**	0,404	0,513	0,427	0,442	0,413	0,474
ChlorineConc,	0,549	0,549	0,550	**0,552**	0,533	0,551	0,537	0,518	0,530	0,550
Cricket-X	0,462	0,472	**0,499**	0,498	0,486	0,469	0,437	0,430	0,429	0,455
Cricket-Y	0,524	0,547	0,542	**0,557**	0,542	0,532	0,476	0,514	0,488	0,518
Cricket-Z	0,491	0,506	0,508	**0,517**	0,504	0,498	0,453	0,472	0,460	0,488
FaceAll	0,620	0,639	0,660	**0,660**	0,447	0,639	0,557	0,589	0,566	0,608
FacesUCR	0,615	0,634	0,653	**0,657**	0,579	0,625	0,588	0,604	0,574	0,601
Fish	0,685	0,719	0,722	**0,727**	0,694	0,704	0,621	0,678	0,647	0,677
Haptics	0,361	0,366	0,341	0,355	**0,368**	0,361	0,310	0,362	0,351	0,364
MedicalImages	0,614	0,620	**0,640**	0,630	0,603	0,617	0,609	0,552	0,557	0,611
NonInvFetECG1	0,775	0,787	0,794	**0,796**	0,772	0,788	0,711	0,732	0,696	0,777
NonInvFetECG2	0,835	0,847	0,850	**0,852**	0,836	0,842	0,777	0,758	0,721	0,831
OSULeaf	0,454	0,460	**0,477**	0,472	0,432	0,468	0,429	0,433	0,403	0,442
StarLightCurves	0,835	0,827	0,834	0,834	0,8367	0,831	**0,840**	0,806	0,784	0,831
SwedishLeaf	0,691	0,701	0,717	**0,722**	0,653	0,716	0,551	0,628	0,559	0,659
Synt, Control	0,833	0,857	**0,888**	0,882	0,528	0,868	0,801	0,858	0,784	0,840
Two Patterns	0,779	0,798	**0,827**	0,821	0,772	0,796	0,765	0,772	0,742	0,770
uWaveGestLib-X	0,703	0,712	**0,722**	0,719	0,695	0,712	0,648	0,693	0,654	0,702
uWaveGestLib-Y	0,619	0,625	0,630	**0,633**	0,621	0,623	0,603	0,607	0,563	0,627
uWaveGestLib-Z	0,618	0,621	**0,629**	0,628	0,615	0,621	0,578	0,611	0,591	0,619
Wafer	0,989	0,991	**0,994**	0,992	0,984	0,992	0,993	0,961	0,965	0,991
WordsSynonyms	0,511	0,531	**0,553**	0,544	0,482	0,526	0,454	0,516	0,480	0,503
Yoga	0,753	0,767	0,773	**0,776**	0,731	0,771	0,747	0,734	0,697	0,740
Vitórias contra amostragem aleatória		23	22	22	8	22	2	3	0	7

A partir dos resultados apresentados na Tabela A.14, também foi realizado o teste de Friedman com o pós-teste de *Nemenyi* com 95% de confiança para comparação dos resultados. O diagrama de diferença crítica resultante do teste é apresentado na Figura A.19.

Neste momento é possível responder algumas questões importantes que podem ser úteis para outros trabalhos que buscam realizar o processo de aprendizado ativo em um cenário em que os dados são totalmente não rotulados.

- **Questão 1:** Existem alternativas simples e efetivas para realizar a seleção de exemplos para a construção de um conjunto de treinamento inicial e que sejam superiores à amostragem aleatória?

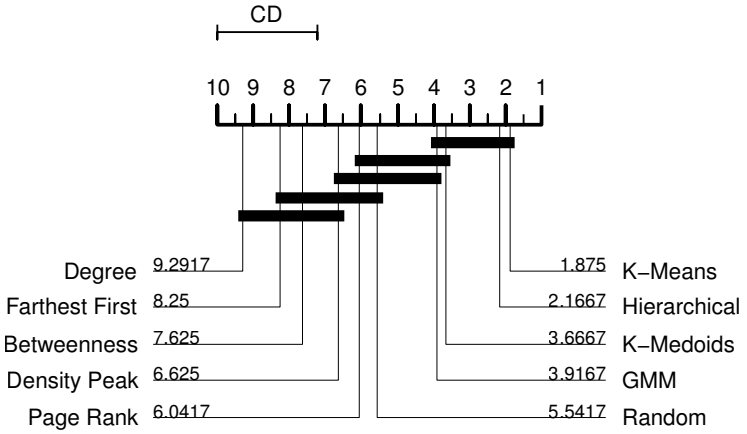


Figura A.19: Diagrama de diferença crítica considerando a média dos resultados de cada algoritmo na tarefa de seleção de exemplos para aprendizado ativo.

- **Resposta:** Sim. Considerando a avaliação experimental conduzida neste trabalho, é possível observar que soluções simples baseadas em algoritmos de agrupamento como k-Means, Hierárquico e k-Medoids são capazes de realizar uma melhor seleção de exemplos em termos de poder preditivo se comparados com a amostragem aleatória. Estes algoritmos são simples e as suas implementações são facilmente encontradas devido a sua popularidade.

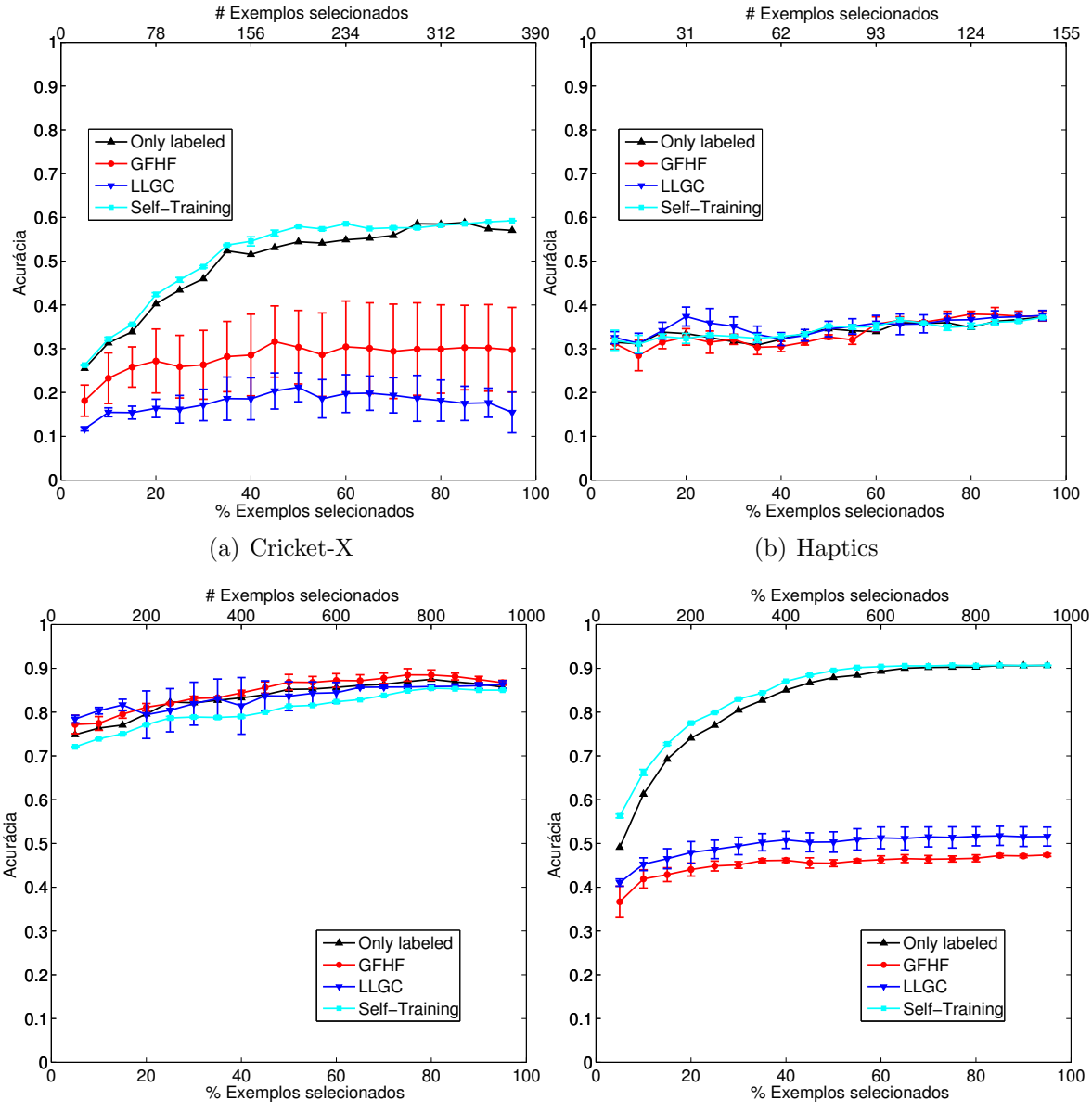
- **Questão 2:** Entre as estratégias avaliadas, qual é a melhor?

- **Resposta:** Os melhores resultados foram obtidos por estratégias baseadas em algoritmos de agrupamento. Em particular, pelos algoritmos k-Means e Hierárquico. Entre estes dois algoritmos, neste trabalho é sugerido o uso do Hierárquico Aglomerativo, por se tratar de um algoritmo determinístico diferentemente do k-Means que eventualmente pode apresentar um resultado inferior em uma de suas execuções. Em geral, a qualidade dos exemplos selecionados por este algoritmo é significativamente superior aos exemplos selecionados aleatoriamente. Entre as medidas de centralidade extraídas de grafos, a medida PageRank foi a que apresentou os melhores resultados. Entretanto, os seus resultados são próximos daqueles obtidos pela amostragem aleatória. Desse modo, não se recomenda o uso de estratégias baseadas em medidas de centralidade extraídas de grafos devido ao seu baixo desempenho.

Na segunda avaliação realizada neste trabalho (*eval2*), foi verificado se o uso de algoritmos de aprendizado semisupervisionado transdutivo para a rotulação de um número maior de exemplos pode contribuir para a formação de um conjunto de treinamento inicial de melhor qualidade. Como nos experimentos realizados com o objetivo de encontrar a melhor abordagem para selecionar exemplos para rotulação pelo oráculo observou-se que o algoritmo de agrupamento Hierárquico Aglomerativo apresentou um dos melhores resultados e tem a vantagem de ser determinístico, na avaliação *eval2* foi considerado somente

os resultados deste algoritmo em conjunto com os algoritmos transdutivos GFHF, LLGC e Self-Training.

De um total de 24 conjuntos de dados avaliados neste trabalho, verificou-se que os algoritmos transdutivos foram responsáveis por melhorar a qualidade do conjunto de treinamento em somente 8 destes conjuntos. Por motivos de brevidade, são apresentados os resultados obtidos somente em quatro conjuntos (Cricket-X, Haptics, StarLightCurves e TwoPatterns) na Figura A.20.



(a) Cricket-X (b) Haptics

(c) StarLightCurves (d) TwoPatterns

Figura A.20: Exemplos de conjuntos de dados em que o uso de algoritmos de aprendizado semisupervisionado pode melhorar a qualidade do conjunto de treinamento rotulado somente pelo oráculo.

Nos resultados apresentados na Figura A.20, o eixo horizontal inferior apresenta a porcentagem de dados rotulados considerada e o eixo horizontal superior indica esta quantidade em valores absolutos. O eixo vertical representa a acurácia obtida ao classificar exemplos de teste T. Nas imagens é possível observar a média e desvio padrão da acurácia obtida pelos algoritmos transdutivos com diferentes configurações de parâmetros. Estes algoritmos são comparados com os resultados obtidos pelo conjunto $L1$ em que os exemplos foram selecionados utilizando a estratégia baseada no agrupamento Hierárquico (*Only labeled*). Pode-se notar que a melhora dos resultados por meio dos algoritmos transdutivos não é significativa. Assim, neste trabalho é respondida uma terceira pergunta, apresentada a seguir.

- **Questão 3:** Exemplos não rotulados, mais especificamente aqueles não selecionados inicialmente pelo algoritmo de aprendizado ativo, podem melhorar o desempenho do algoritmo de classificação por meio de sua incorporação no conjunto de treino utilizando algoritmos de aprendizado semissupervisionado transdutivos?

- **Resposta:** De um total de 24 conjuntos de dados avaliados, nota-se que os algoritmos transdutivos melhoraram apenas sensivelmente os resultados em somente um terço dos conjuntos de dados. Entre os algoritmos avaliados (GFHF, LLGC e Self-Training), o algoritmo Self-Training foi o que mais se destacou. Entretanto, em muitos conjuntos de dados os algoritmos transdutivos foram responsáveis por reduzir a qualidade do conjunto de treinamento inicial. Desse modo, além de representar um custo adicional no processo de rotulação de dados, não foram encontradas evidências de que os algoritmos transdutivos podem melhorar a qualidade do conjunto de treinamento previamente rotulado por um especialista com o auxílio de estratégias de aprendizado ativo para a seleção de exemplos.

É interessante destacar que uma discussão e análise mais abrangente dos resultados são apresentadas em Souza et al. (2016). Além disso, os resultados de todos os algoritmos discutidos nos 24 conjuntos de dados avaliados são apresentados no material complementar[7] que acompanha o artigo.

[7] http://sites.labic.icmc.usp.br/vsouza/SM_IDA.pdf

Referências Bibliográficas

Abushariah, A. A. M., Gunawan, T. S., Khalifa, O. O., e Abushariah, M. A. M. (2010). English digits speech recognition system based on hidden markov models. Em *Proceedings of the International Conference on Computer and Communication Engineering (ICOCOE)*, páginas 1–5. Citado na página 153.

Aggarwal, C. C., Han, J., Wang, J., e Yu, P. S. (2003). A framework for clustering evolving data streams. Em *Proceedings of the International Conference on Very Large Data Bases (VLDB)*, volume 29, páginas 81–92. Citado na página 57.

Ajtai, M. (1988). The complexity of the pigeonhole principle. Em *Proceedings of the Annual Symposium on Foundations of Computer Science*, páginas 346–355. Citado na página 88.

Ali, K. M. e Pazzani, M. J. (1996). Error reduction through learning multiple descriptions. *Machine Learning*, 24(3):173–202. Citado na página 127.

Alotaibi, Y. A. (2003). High performance arabic digits recognizer using neural networks. Em *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, páginas 670–674. Citado na página 153.

Araujo, B. e Zhao, L. (2013). Detecting and labeling representative nodes for network-based semi-supervised learning. Em *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, páginas 1–8. Citado na página 182.

Arica, N. e Vural, F. T. Y. (2003). Bas: a perceptual shape descriptor based on the beam angle statistics. *Pattern Recognition Letters*, 24(9):1627–1639. Citado na página 166.

Assaleh, K. T. e Mammone, R. J. (1994). Robust cepstral features for speaker identification. Em *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume i, páginas 129–132. Citado na página 87.

Azam, S. M., Mansoor, Z. A., Mughal, M. S., e Mohsin, S. (2007). Urdu spoken digits recognition using classified MFCC and backpropgation neural network. Em *Proceedings of Computer Graphics, Imaging and Visualisation*, páginas 414–418. Citado na página 153.

Babcock, B., Babu, S., Datar, M., Motwani, R., e Widom, J. (2002). Models and issues in data stream systems. Em *Proceedings of the ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems (PODS)*, páginas 1–16. Citado na página 12.

Baena-García, M., del Campo-Ávila, J., Fidalgo, R., Bifet, A., Gavaldà, R., e Morales-Bueno, R. (2006). Early drift detection method. Em *Proceedings of the International Workshop on Knowledge Discovery from Data Streams*, páginas 1–10. Citado na página 23.

Bagnall, A., Davis, L. M., Hills, J., e Lines, J. (2012). Transformation based ensembles for time series classification. Em *Proceedings of the SIAM International Conference on Data Mining (SDM)*, volume 12, páginas 307–318. Citado na página 170.

Baraldi, A. e Parmiggiani, F. (1995). An investigation of the textural characteristics associated with gray level cooccurrence matrix statistical parameters. *IEEE Transactions on Geoscience and Remote Sensing*, 33(2):293–304. Citado na página 178.

Baram, Y., El-Yaniv, R., e Luz, K. (2004). Online choice of active learning algorithms. *The Journal of Machine Learning Research*, 5(1):255–291. Citado na página 182.

Batista, G. E. A. P. A., Campana, B., e Keogh, E. (2010). Classification of live moths combining texture, color and shape primitives. Em *Proceedings of the International Conference on Machine Learning and Applications (ICMLA)*, páginas 903–906. Citado na página 166.

Batista, G. E. A. P. A., Hao, Y., Keogh, E., e Neto, A. M. (2011a). Towards automatic classification on flying insects using inexpensive sensors. Em *Proceedings of the International Conference on Machine Learning and Applications (ICMLA)*, volume 1, páginas 364–369. Citado na página 97.

Batista, G. E. A. P. A., Keogh, E., Tataw, O. M., e Souza, V. M. A. (2014). Cid: an efficient complexity-invariant distance for time series. *Data Mining and Knowledge Discovery*, 28(3):634–669. Citado nas páginas 145 e 166.

Batista, G. E. A. P. A., Keogh, E. J., Neto, A. M., e Rowton, E. (2011b). Sigkdd demo: sensors and software to allow computational entomology, an emerging application of data mining. Em *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, páginas 761–764. Citado nas páginas 77 e 116.

Bello, J. P. (2011). Measuring structural similarity in music. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(7):2013–2025. Citado na página 162.

Belton, P. e Costello, R. A. (1979). Flight sounds of the females of some mosquitoes of western canada. *Entomologia Experimentalis et Applicata*, 26(1):105–114. Citado na página 81.

Benesty, J., Chen, J., e Huang, Y. (2008). Linear prediction. Em Benesty, J., Sondhi, M. M., e Huang, Y., editores, *Springer Handbook of Speech Processing*, páginas 121–134. Citado na página 95.

Bertol, E. (2014). Inseticida seletivo. *Revista Galileu*, 277:1–1. Citado na página 85.

Bhatt, S., Gething, P. W., Brady, O. J., Messina, J. P., Farlow, A. W., Moyes, C. L., Drake, J. M., Brownstein, J. S., Hoen, A. G., e Sankoh, O. (2013). The global distribution and burden of dengue. *Nature*, 496(7446):504–507. Citado na página 4.

Bifet, A. e Gavalda, R. (2007). Learning from time-changing data with adaptive windowing. Em *Proceedings of the SIAM International Conference on Data Mining (SDM)*, páginas 443–448. Citado na página 23.

Bifet, A., Holmes, G., Pfahringer, B., Kirkby, R., e Gavalda, R. (2009). New ensemble methods for evolving data streams. Em *Proceedings of the ACM SIGKDD international Conference on Knowledge Discovery and Data Mining (KDD)*, páginas 139–148. Citado nas páginas 13 e 14.

Bishop, C. M. (1995). *Neural networks for pattern recognition*. Oxford university press. Citado nas páginas 104 e 105.

Bogert, B. P., Healy, M. J. R., e Tukey, J. W. (1963). The quefrency alanysis of time series for echoes: Cepstrum, pseudo-autocovariance, cross-cepstrum and saphe cracking. Em *Proceedings of the Symposium on Time Series Analysis*, páginas 209–243. Citado na página 87.

Bové, J. M. (2006). Huanglongbing: a destructive, newly-emerging, century-old disease of citrus. *Journal of plant pathology*, 88(1):7–37. Citado nas páginas 3 e 85.

Bresolin, A. A., Neto, A. D. D., e Alsina, P. J. (2008). Digit recognition using wavelet and SVM in brazilian portuguese. Em *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, páginas 1545–1548. Citado na página 153.

Campana, B. J. L. e Keogh, E. J. (2010). A compression based distance measure for texture. Em *Proceedings of the SIAM International Conference on Data Mining (SDM)*, páginas 850–861. Citado nas páginas 171, 173, e 174.

Campbell, R. H., Martin, S. K., Schneider, I., e Michalson, W. R. (1996). Analysis of mosquito wing beat sound. *The Journal of the Acoustical Society of America*, 100(4):2710–2710. Citado na página 81.

Caprio, M. A., Huang, J. X., Faver, M. K., e Moore, A. (2001). Characterization of male and female wingbeat frequencies in the anopheles quadrimaculatus complex in mississippi. *Journal of the American Mosquito Control Association*, 17(3):186–189. Citado nas páginas 81, 83, e 112.

Cattral, R., Oppacher, F., e Deugo, D. (2002). Evolutionary data mining with automatic rule generalization. *Recent Advances in Computers, Computing and Communications*, 1(1):296–300. Citado na página 34.

Chadwick, L. E. (1939). Some factors which affect the rate of movement of the wings in drosophila. *Physiological Zoology*, 12(2):151–160. Citado nas páginas 3 e 114.

Chadwick, L. E. (1953). The motion of the wings. *Insect physiology*, (1):577–614. Citado na página 112.

Chadwick, L. E. e Williams, C. M. (1949). The effects of atmospheric pressure and composition on the flight of drosophila. *The Biological Bulletin*, 97(2):115–137. Citado nas páginas 3 e 114.

Chazelle, B. (1993). An optimal convex hull algorithm in any fixed dimension. *Discrete & Computational Geometry*, 10(1):377–409. Citado na página 42.

Chen, Y., Keogh, E., Hu, B., Begum, N., Bagnall, A., Mueen, A., e Batista, G. (2015). The ucr time series classification archive. `www.cs.ucr.edu/~eamonn/time_series_data/`. Citado nas páginas 168, 174, 178, e 182.

Chen, Y., Why, A., Batista, G., Mafra-Neto, A., e Keogh, E. (2014). Flying insect classification with inexpensive sensors. *Journal of insect behavior*, 27(5):657–677. Citado nas páginas 88, 89, e 113.

Cohen, J. (1960). A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1):37–46. Citado na página 54.

Colborn, T., von Saal, F. S., e Soto, A. M. (1993). Developmental effects of endocrine-disrupting chemicals in wildlife and humans. *Environmental Health Perspect*, 101(5):378–384. Citado na página 79.

Cooley, J. W. e Tukey, J. W. (1965). An algorithm for the machine calculation of complex Fourier series. *Mathematics of Computation*, 19(90):297–301. Citado na página 86.

Cooper, M. L. e Foote, J. (2002). Automatic music summarization via similarity analysis. Em *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. Citado na página 164.

Costa, A. F., Humpire-Mamani, G., e Traina, A. J. M. (2012). An efficient algorithm for fractal analysis of textures. Em *Proceedings of the Conference on Graphics, Patterns and Images (SIBGRAPI)*, páginas 39–46. Citado na página 178.

Cover, T. M. e Hart, P. E. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27. Citado na página 55.

Dasu, T., Krishnan, S., Venkatasubramanian, S., e Yi, K. (2006). An information-theoretic approach to detecting changes in multi-dimensional data streams. Em *Proceedings of Symposium on the Interface of Statistics, Computing Science, and Applications*. Citado na página 22.

de Carvalho, J. P. (1986). *Introdução à entomologia agrícola*. Fundação Calouste Gulbenkian. Citado na página 118.

De Sousa, C. A. R., Souza, V. M. A., e Batista, G. E. A. P. A. (2014). Time series transductive classification on imbalanced data sets: an experimental study. Em *Proceedings of the International Conference on Pattern Recognition (ICPR)*, páginas 3780–3785. Citado nas páginas 145 e 182.

De Sousa, C. A. R., Souza, V. M. A., e Batista, G. E. A. P. A. (2015). An experimental analysis on time series transductive classification on graphs;. Em *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, páginas 1–8. Citado nas páginas 144 e 182.

Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research*, 7:1–30. Citado na página 62.

Ding, H., Trajcevski, G., Scheuermann, P., Wang, X., e Keogh, E. (2008). Querying and mining of time series data: experimental comparison of representations and distance measures. *Proceedings of the VLDB Endowment*, 1(2):1542–1552. Citado nas páginas 165, 170, e 184.

Dixon, K. W. (2009). Pollination and restoration. *Science*, 325(5940):571–573. Citado na página 78.

Domingos, P. e Hulten, G. (2000). Mining high-speed data streams. Em *Proceedings of the ACM SIGKDD international conference on Knowledge discovery and data mining (KDD)*, páginas 71–80. Citado na página 140.

Domingues, M. A., Cherman, E. A., Nogueira, B. M., Conrado, M. S., Rossi, R. G., Padua, R., Marcacini, R. M., Souza, V. M. A., Batista, G. E. A. P. A., e Rezende, S. O. (2013). A comparative study of algorithms for recommending given names. Em *Proceedings of the International Conference on Informatics & Applications (ICIA)*, páginas 66–71. Citado na página 146.

Doost, H. R. E. e Rahebi, J. (2012). An efficient method for texture classification with local binary pattern based on wavelet transformation. *International Journal of Engineering Science*, 4(12):4881–4885. Citado na página 176.

Duda, R. O. e Hart, P. E. (1973). *Pattern classification and scene analysis*, volume 3. Wiley New York. Citado na página 104.

Dunn, J. C. (1973). A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters. *J. Cybernet*, 3(1):32–57. Citado na página 54.

Dyer, K. B., Capo, R., e Polikar, R. (2014). Compose: A semisupervised learning framework for initially labeled nonstationary streaming data. *IEEE Transactions on Neural Networks and Learning Systems*, 25(1):12–26. Citado nas páginas 39, 40, 41, 42, 47, 51, 52, 60, 66, 73, e 75.

Ebert, D. S., Musgrave, F. K., Peachey, D., Perlin, K., e Worley, S. (2003). *Texturing & Modeling: A Procedural Approach*. Morgan Kaufmann Publisher. Citado na página 176.

Eckmann, J. P., Kamphorst, O. S., e Ruelle, D. (1987). Recurrence plots of dynamical systems. *Europhysics Letters*, 4(9):973–977. Citado na página 170.

Eronen, A. e Klapuri, A. (2000). Musical instrument recognition using cepstral coefficients and temporal features. Em *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 2, páginas 753–756. Citado na página 87.

Farnworth, E. G. (1972). Effects of ambient temperature, humidity, and age on wing-beat frequency of periplaneta species. *Journal of Insect Physiology*, 18(5):827–839. Citado nas páginas 3, 112, e 114.

Gama, J. (2010). *Knowledge Discovery from Data Streams*. Data mining and knowledge discovery. Chapman & Hall/CRC, 1st edição. Citado nas páginas 1, 12, 20, e 21.

Gama, J., Medas, P., Castillo, G., e Rodrigues, P. (2004). Learning with drift detection. Em Bazzan, A. e Labidi, S., editores, *Advances in Artificial Intelligence - SBIA*, volume 3171 of *Lecture Notes in Computer Science*, páginas 286–295. Citado nas páginas 23 e 34.

Gama, J., Rocha, R., e Medas, P. (2003). Accurate decision trees for mining high-speed data streams. Em *Proceedings of the ACM SIGKDD international conference on Knowledge discovery and data mining (KDD)*, páginas 523–528. Citado na página 34.

Gama, J., Rodrigues, P., e Aguilar-Ruiz, J. (2006). An overview on learning from data streams. *New Generation Computing*, 25:1–4. Citado na página 12.

Gao, J., Fan, W., e Han, J. (2007). On appropriate assumptions to mine data streams: Analysis and practice. Em *Proceedings of the International Conference on Data Mining (ICDM)*, páginas 143–152. Citado na página 34.

Gehrke, J., Korn, F., e Srivastava, D. (2001). On computing correlated aggregates over continual data streams. *ACM SIGMOD*, 30(2):13–24. Citado na página 20.

Ghanty, S., Shaikh, S., e Chaki, N. (2010). On recognition of spoken Bengali numerals. Em *Proceedings of the International Conference on Computer Information Systems and Industrial Management Applications (CISIM)*, páginas 54–59. Citado na página 153.

Gubler, D. J. (2012). The economic burden of dengue. *The American journal of tropical medicine and hygiene*, 86(5):743–744. Citado na página 4.

Gubler, D. J., Ooi, E. E., Vasudevan, S., e Farrar, J. (2014). *Dengue and dengue hemorrhagic fever*. CABI. Citado na página 4.

Hall, M. A. (1999). *Correlation-based feature selection for machine learning*. Tese de Doutorado, The University of Waikato. Citado na página 178.

Haralick, R. (1979). Statistical and structural approaches to texture. *Proceedings of the IEEE*, 67(5):786–804. Citado na página 176.

Haralick, R. M., Shanmugam, K., e Dinstein, I. H. (1973). Textural features for image classification. *IEEE Transactions on Systems, Man and Cybernetics*, 1(6):610–621. Citado nas páginas 177 e 178.

Harries, M. e Wales, N. S. (1999). Splice-2 comparative evaluation: Electricity pricing. Relatório técnico, The University of South Wales. Citado na página 34.

Herrera, P., Yeterian, A., e Gouyon, F. (2002). Automatic classification of drum sounds: a comparison of feature selection methods and classification techniques. Em *Proceedings of the International Conference on Music and Artificial Intelligence (ICMAI)*, páginas 69–80. Citado na página 148.

Herrera Boyer, P., Peeters, G., e Dubnov, S. (2003). Automatic classification of musical instrument sounds. *Journal of New Music Research*, 32(1):3–21. Citado na página 148.

Hickey, R. J. (2007). Structure and majority classes in decision tree learning. *Journal of Machine Learning Research*, 8(8):1747–1768. Citado na página 34.

Hoens, T. R., Polikar, R., e Chawla, N. V. (2012). Learning from streaming data with concept drift and imbalance: an overview. *Progress in Artificial Intelligence*, 1(1):89–101. Citado nas páginas 16 e 17.

Hu, R., Mac Namee, B., e Delany, S. J. (2010). Off to a good start: Using clustering to select the initial training set in active learning. Em *Proceedings of the International Florida Artificial Intelligence Research Society Conference (FLAIRS*, páginas 26–31. Citado na página 181.

Hu, X., Zhan, L., Xue, Y., Zhou, W., e Zhang, L. (2011). Spoken arabic digits recognition based on wavelet neural networks. Em *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, páginas 1481–1485. Citado na página 153.

Huang, Y. S. e Suen, C. Y. (1995). A method of combining multiple experts for the recognition of unconstrained handwritten numerals. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(1):90–94. Citado na página 127.

Hulten, G., Spencer, L., e Domingos, P. (2001). Mining time-changing data streams. Em *Proceedings of the ACM SIGKDD international conference on Knowledge discovery and data mining (KDD)*, páginas 97–106. Citado nas páginas 35 e 140.

Hur, H. Y. e Kim, H. S. (2001). Formant weighted cepstral feature for lsp-based speech recognition. Em *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 1, páginas 141–144. Citado na página 87.

Idrissa, M. e Acheroy, M. (2002). Texture classification using gabor filters. *Pattern Recognition Letters*, 23(9):1095–1102. Citado na página 178.

Itakura, F. (1975). Minimum prediction residual principle applied to speech recognition. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 23(1):67–72. Citado na página 95.

Iwanski, J. S. e Bradley, E. (1998). Recurrence plots of experimental data: To embed or not to embed? *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 8(4):861–871. Citado na página 173.

Jain, A. K. e Dubes, R. C. (1988). *Algorithms for clustering data*. Prentice-Hall. Citado na página 169.

Joachims, T. (1999). Making large-scale SVM learning practical. Em Schölkopf, B., Burges, C., e Smola, A., editores, *Advances in Kernel Methods - Support Vector Learning*, capítulo 11, páginas 169–184. MIT Press, Cambridge, MA. Citado na página 73.

Jones, M. D. R. (1964). The automatic recording of mosquito activity. *Journal of Insect Physiology*, 10(2):343–351. Citado na página 81.

Kahn, M. C., Celestin, W., e Offenhauser Jr, W. H. (1945). Recording of sounds produced by certain disease-carrying mosquitoes. *Science*, 101(2622):335–336. Citado nas páginas 80, 81, e 112.

Kahn, M. C. e Offenhauser Jr, W. H. (1949). The identification of certain west african mosquitoes by sound. *The American Journal of Tropical Medicine and Hygiene*, 29(5):827–836. Citado nas páginas 80, 81, e 112.

Kaufman, L. e Rousseeuw, P. J. (2009). *Finding groups in data: an introduction to cluster analysis*, volume 344. John Wiley & Sons. Citado na página 169.

Kelly, M. G., Hand, D. J., e Adams, N. M. (1999). The impact of changing populations on classifier performance. Em *Proceedings of the ACM SIGKDD international conference on Knowledge discovery and data mining (KDD)*, páginas 367–371. Citado na página 16.

Keogh, E. e Lin, J. (2005). Clustering of time-series subsequences is meaningless: implications for previous and future research. *Knowledge and information systems*, 8(2):154–177. Citado na página 165.

Kevan, P. G. (1999). Pollinators as bioindicators of the state of the environment: species, activity and diversity. *Agriculture, Ecosystems & Environment*, 74(3):373–393. Citado na página 78.

Kifer, D., Ben-David, S., e Gehrke, J. (2004). Detecting change in data streams. Em *Proceedings of the International conference on Very large data bases (VLDB)*, volume 30, páginas 180–191. Citado na página 22.

Killourhy, K. e Maxion, R. (2010). Why did my detector do that?! Em *Recent Advances in Intrusion Detection*, páginas 256–276. Citado na página 51.

Kirkby, R. B. (2007). *Improving hoeffding trees*. Tese de Doutorado, The University of Waikato. Citado na página 12.

Klinkenberg, R. (2004). Learning drifting concepts: Example selection vs. example weighting. *Intelligent Data Analysis*, 8(3):281–300. Citado na página 21.

Kolter, J. Z. e Maloof, M. (2003). Dynamic weighted majority: A new ensemble method for tracking concept drift. Em *Proceedings of the IEEE International Conference on Data Mining (ICDM)*, páginas 123–130. Citado na página 127.

Kondo, K., Kamata, H., e Ishida, Y. (1994). Speaker-independent spoken digits recognition using lvq. Em *Proceedings of the IEEE World Congress on Computational Intelligence (WCCI)*, volume 7, páginas 4448–4451. Citado na página 153.

Kononenko, I. (1994). Estimating attributes: analysis and extensions of relief. Em *Proceedings of the European conference on Machine Learning (ECML)*, páginas 171–182. Citado na página 178.

Kopparapu, S. K. e Rao, P. V. S. (2004). Enhancing spoken connected-digit recognition accuracy by error correction codes: A novel scheme. *Sadhana (Academy Proceedings in Engineering Sciences)*, 29(5):559–571. Citado na página 153.

Koychev, I. (2000). Gradual forgetting for adaptation to concept drift. Em *Proceedings of ECAI Workshop on Current Issues in Spatio-Temporal Reasoning*, páginas 101–106. Citado na página 21.

Krempl, G. (2011). The algorithm apt to classify in concurrence of latency and drift. Em *Advances in Intelligent Data Analysis X*, páginas 222–233. Citado nas páginas 39 e 75.

Krempl, G., Žliobaite, I., Brzeziński, D., Hüllermeier, E., Last, M., Lemaire, V., Noack, T., Shaker, A., Sievi, S., Spiliopoulou, M., e Stefanowski, J. (2014). Open challenges for data stream mining research. *ACM SIGKDD Explorations Newsletter*, 16(1):1–10. Citado nas páginas 3, 33, e 75.

Krimphoff, J., McAdams, S., e Winsberg, S. (1994). Caractérisation du timbre des sons complexes. ii: Analyses acoustiques et quantification psychophysique. *Journal de Physique*, 4(5):625–628. Citado na página 93.

Kubat, M. e Widmer, G. (1995). Adapting to drift in continuous domains. Em *Machine Learning: ECML-95: 8th European Conference on Machine Learning, Heraclion, Crete, Greece, April 25-27, 1995. Proceedings*, volume 8, página 307. Citado na página 21.

Kuncheva, L. I. e Faithfull, W. J. (2014). Pca feature extraction for change detection in multidimensional unlabeled data. *IEEE Transactions on Neural Networks and Learning Systems*, 25(1):69–80. Citado na página 18.

Lee, C.-H., Shih, J.-L., Yu, K.-M., e Lin, H.-S. (2009). Automatic music genre classification based on modulation spectral analysis of spectral and cepstral features. *IEEE Transactions on Multimedia*, 11(4):670–682. Citado na página 87.

Levenbook, L. e Williams, C. M. (1956). Mitochondria in the flight muscles of insects. iii. mitochondrial cytochrome c in relation to the aging and wing beat frequency of flies. *The Journal of General Physiology*, 39(5):497–512. Citado nas páginas 112 e 113.

MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. Em *Proceedings of the Berkeley symposium on mathematical statistics and probability*, volume 1, páginas 281–297. Citado na página 53.

Manjunath, B. S. e Ma, W.-Y. (1996). Texture features for browsing and retrieval of image data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(8):837–842. Citado na página 178.

Mankin, R. W. (1994). Acoustical detection of aedes taeniorhynchus swarms and emergence exoduses in remote salt marshes. *Journal of the American Mosquito Control Association - Mosquito News*, 10(2):302–308. Citado na página 81.

Marrs, G. R., Hickey, R. J., e Black, M. M. (2010). The impact of latency on online classification learning with concept drift. Em *Knowledge Science, Engineering and Management*, páginas 459–469. Citado nas páginas 2, 31, 32, e 34.

Masud, M. M., Gao, J., Khan, L., Han, J., e Thuraisingham, B. (2008). A practical approach to classify evolving data streams: Training with limited amount of labeled data. Em *Proceedings of the IEEE International Conference on Data Mining (ICDM)*, páginas 929–934. Citado na página 5.

Mellanby, K. (1936). Humidity and insect metabolism. *Nature*, 138(1):124–125. Citado nas páginas 3 e 114.

Ministério-Saúde (2015). *Boletim Epidemiológico - Monitoramento dos casos de microcefalias no Brasil, até a semana epidemiológica 46, 2015*, volume 46. Citado na página 4.

Ministério-Saúde (2016a). *Boletim Epidemiológico - Monitoramento dos casos de dengue, febre de chikungunya e febre pelo vírus Zika até a Semana Epidemiológica 52, 2015*, volume 47. Citado na página 4.

Ministério-Saúde (2016b). *Boletim Epidemiológico - Monitoramento dos casos de microcefalias no Brasil, até a semana epidemiológica 1, 2016*, volume 1. Citado na página 4.

Minku, L. L., White, A. P., e Yao, X. (2010). The impact of diversity on online ensemble learning in the presence of concept drift. *IEEE Transactions on Knowledge and Data Engineering*, 22(5):730–742. Citado na página 18.

Moon, T. K. (1996). The expectation-maximization algorithm. *IEEE Signal processing magazine*, 13(6):47–60. Citado na página 40.

Moore, A. (1991). Artificial neural network trained to identify mosquitoes in flight. *Journal of insect behavior*, 4(3):391–395. Citado na página 81.

Moore, A. (1998). Development of a data acquisition system for long-term outdoor recording of insect flight activity using a photosensor. Em *Proceedings of the Conference on Aerobiology and Biometeorology*. Citado nas páginas 81, 82, e 83.

Moore, A., Miller, J. R., Tabashnik, B. E., e Gage, S. H. (1986). Automated identification of flying insects by analysis of wingbeat frequencies. *Journal of economic entomology*, 79(6):1703–1706. Citado na página 81.

Moore, A. e Miller, R. H. (2002). Automated identification of optically sensed aphid (homoptera: Aphidae) wingbeat waveforms. *Annals of the Entomological Society of America*, 95(1):1–8. Citado nas páginas 81 e 82.

Mouss, H., Mouss, D., Mouss, N., e Sefouhi, L. (2004). Test of page-hinckley, an approach for fault detection in an agro-alimentary production system. Em *Proceedings of the Asian Control Conference*, volume 2, páginas 815–818. Citado na página 23.

Nakamaru, M., Iwasa, Y., e Nakanishi, J. (2002). Extinction risk to herring gull populations from DDT exposure. *Environmental toxicology and chemistry*, 21(1):195–202. Citado na página 79.

Newman, M. (2010). *Networks: an introduction*. Oxford University Press. Citado na página 182.

Ng, A. Y., Jordan, M. I., e Weiss, Y. (2002). On spectral clustering: Analysis and an algorithm. *Advances in neural information processing systems*, 2(1):849–856. Citado na página 169.

Ojala, T., Pietikäinen, M., e Harwood, D. (1996). A comparative study of texture measures with classification based on featured distributions. *Pattern recognition*, 29(1):51–59. Citado na página 176.

Ojala, T., Pietikainen, M., e Maenpaa, T. (2002). Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):971–987. Citado na página 177.

Oppenheim, A., Schafer, R., e Buck, J. (1989). *Discrete-time signal processing*, volume 2. Prentice Hall. Citado na página 95.

Orság, F. (2010). Speaker dependent coefficients for speaker recognition. *International Journal of Security and Its Applications*, 4(1):31–48. Citado na página 155.

Oza, N. C. e Russell, S. (2001). Experimental comparisons of online and batch versions of bagging and boosting. Em *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, páginas 359–364. Citado na página 34.

Page, E. (1954). Continuous inspection schemes. *Biometrika*, 1(1):100–115. Citado nas páginas 22 e 23.

Paliwal, K. e Kleijn, W. (1995). Quantization of LPC parameters. *Speech Coding and Synthesis*, 1(1):433–466. Citado na página 96.

Panwar, M., Sharma, R. P., Khan, I., e Farooq, O. (2011). Design of wavelet based features for recognition of hindi digits. Em *Proceedings of the International Conference on Multimedia, Signal Processing and Communication Technologies (IMPACT)*, páginas 232–235. Citado na página 153.

Park, T. H. (2004). *Towards Automatic Musical Instrument Timbre Recognition*. Tese de Doutorado, Princeton University. Citado na página 92.

Patnaik, J. L., Juliusson, L., e Vogt, R. L. (2007). Environmental predictors of human west nile virus infections, colorado. *Emerging Infectious Diseases*, 13(11):1788–1790. Citado na página 79.

Paulus, J. (2006). Acoustic modelling of drum sounds with hidden markov models for music transcription. Em *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 5, páginas 241–244. Citado na página 148.

Pelleg, D. e Moore, A. W. (2000). X-means: Extending k-means with efficient estimation of the number of clusters. Em *Proceedings of the International Conference on Machine Learning (ICML)*, páginas 727–734. Citado na página 54.

Perumpral, J. V., Earp, U. F., e Stanley, J. M. (1974). Factors affecting the wingbeat frequency of cabbage loopers. *Transactions of the American Society of Agricultural Engineers*, 17(4–6):726. Citado na página 112.

Pollard, H. e Jansson, E. (1982). A tristimulus method for the specification of musical timbre. *Acta Acustica united with Acustica*, 51(3):162–171. Citado na página 92.

Prati, R. C., Batista, G. E. A. P. A., e Monard, M. C. (2011). A survey on graphical methods for classification predictive performance evaluation. *IEEE Transactions on Knowledge and Data Engineering*, 23(11):1601–1618. Citado na página 106.

Qi, Y., Cinar, G. T., Souza, V. M. A., Batista, G. E. A. P. A., Wang, Y., e Principe, J. C. (2015). Effective insect recognition using a stacked autoencoder with maximum correntropy criterion. Em *International Joint Conference on Neural Networks (IJCNN)*, páginas 1–7. Citado nas páginas 98 e 144.

Rakthanmanon, T., Campana, B., Mueen, A., Batista, G., Westover, B., Zhu, Q., Zakaria, J., e Keogh, E. (2012). Searching and mining trillions of time series subsequences under dynamic time warping. Em *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, páginas 262–270. Citado na página 159.

Reed, S. C., Williams, C. M., e Chadwick, L. E. (1941). Frequency of wing-beat as a character for separating species races and geographic varieties of drosophila. *Genetics*, 27(3):349–361. Citado na página 115.

Rodrigues, F. e Trancoso, I. (1999). Digit recognition using the SPEECHDAT corpus. Em *Proceedings of Conference on Telecommunications*, páginas 1–4. Citado na página 153.

Rodriguez, A. e Laio, A. (2014). Clustering by fast search and find of density peaks. *Science*, 344(6191):1492–1496. Citado na página 182.

Roubaud, E. (1918). Rhythmes physiologiques et vol spontan chez l'anopheles maculipennis. *C. R. Hebdomadaires des Seances de l'Academie des Science*, 167(1):967–969. Citado na página 115.

Salganicoff, M. (1997). Tolerating concept and sampling shift in lazy learning using prediction error context switching. *Artificial Intelligence Review*, 11(1):133–155. Citado na página 21.

Schölkopf, B., Platt, J. C., Shawe-Taylor, J., Smola, A. J., e Williamson, R. C. (2001). Estimating the support of a high-dimensional distribution. *Neural computation*, 13(7):1443–1471. Citado na página 105.

Scholler, S. e Purwins, H. (2011). Sparse approximations for drum sound classification. *IEEE Journal of Selected Topics in Signal Processing*, 5(5):933–940. Citado na página 148.

Scudder, G. G. E. (2009). The importance of insects. *Insect Biodiversity: Science and Society*, 1(1):7–32. Citado nas páginas 4 e 86.

Sebastião, R. e Gama, J. (2007). Change detection in learning histograms from data streams. Em *Progress in artificial intelligence: Proceedings of the Portuguese Conference on Artificial Intelligence*, páginas 112–123. Citado na página 22.

Sebastião, R. e Gama, J. (2009). A study on change detection methods. Em *Proceedings of the Portuguese Conference on Artificial Intelligence (EPIA)*, páginas 353–364. Citado nas páginas 2, 19, e 23.

Serrà, J., Gómez, E., e Herrera, P. (2010). Audio cover song identification and similarity: background, approaches, evaluation, and beyond. Em *Advances in Music Information Retrieval*, páginas 307–332. Citado na página 157.

Settles, B. (2010). Active learning literature survey. *University of Wisconsin, Madison*, 52(55-66):11. Citado na página 180.

Settles, B., Craven, M., e Friedland, L. (2008). Active learning with real annotation costs. Em *Proceedings of the NIPS workshop on cost-sensitive learning*, páginas 1–10. Citado na página 180.

Shepard, D. S., Coudeville, L., Halasa, Y. A., Zambrano, B., e Dayan, G. H. (2011). Economic impact of dengue illness in the americas. *The American journal of tropical medicine and hygiene*, 84(2):200–207. Citado na página 4.

Silla Jr, C. N., Koerich, A. L., e Kaestner, C. A. A. (2008). The latin music database. Em *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, páginas 451–456. Citado na página 161.

Silva, D. F. (2014). Classificação de séries temporais por similaridade e extração de atributos com aplicação na identificação automática de insetos. Dissertação de Mestrado. *Instituto de Ciências Matemáticas e de Computação – Universidade de São Paulo*. Citado nas páginas 92, 98, e 154.

Silva, D. F., Souza, V. M. A., e Batista, G. E. A. P. A. (2013a). A comparative study between mfcc and lsf coefficients in automatic recognition of isolated digits pronounced in portuguese and english-doi: 10.4025/actascitechnol. v35i4. 19825. *Acta Scientiarum. Technology*, 35(4):621–628. Citado nas páginas 146, 153, e 156.

Silva, D. F., Souza, V. M. A., e Batista, G. E. A. P. A. (2013b). Time series classification using compression distance of recurrence plots. Em *Proceedings of the IEEE International Conference on Data Mining (ICDM)*, páginas 687–696. Citado nas páginas 146 e 170.

Silva, D. F., Souza, V. M. A., e Batista, G. E. A. P. A. (2015a). Music shapelets for fast cover song recognition. Em *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, páginas 441–447. Citado nas páginas 144, 157, e 161.

Silva, D. F., Souza, V. M. A., Batista, G. E. A. P. A., e Giusti, R. (2012). Spoken digit recognition in portuguese using line spectral frequencies. Em *Advances in Artificial Intelligence – IBERAMIA 2012*, páginas 241–250. Citado nas páginas 146, 153, 154, e 156.

Silva, D. F., Souza, V. M. A., Batista, G. E. A. P. A., Keogh, E., e Ellis, D. P. W. (2013c). Applying machine learning and audio analysis techniques to insect recognition in intelligent traps. Em *Proceedings of the International Conference on Machine Learning and Applications (ICMLA)*, volume 1, páginas 99–104. Citado nas páginas 7, 98, 99, e 145.

Silva, D. F., Souza, V. M. A., Ellis, D. P. W., Keogh, E., e Batista, G. E. A. P. A. (2015b). Exploring low cost laser sensors to identify flying insect species. *Journal of Intelligent & Robotic Systems*, 1(1):1–18. Citado nas páginas 7, 88, 98, 99, e 145.

Souza, V. M. A., Batista, G. E. A. P. A., e Souza-Filho, N. E. (2015a). Automatic classification of drum sounds with indefinite pitch. Em *International Joint Conference on Neural Networks (IJCNN)*, páginas 1–8. Citado nas páginas 89, 144, e 148.

Souza, V. M. A. e Feltrim, V. D. (2013). Uma investigação sobre algoritmos de diferentes abordagens de aprendizado supervisionado na classificação de papéis retóricos em resumos científicos. Em *Proceedings of the Brazilian Symposium in Information and Human Language Technology (STIL)*, páginas 30–39. Citado na página 145.

Souza, V. M. A., Rossi, R. G., Rezende, S. O., e Batista, G. E. A. P. A. (2016). Unsupervised active learning techniques for labeling training sets. *Submetido para Intelligent Data Analysis Journal*, páginas 1–29. Citado nas páginas 98, 180, 181, e 188.

Souza, V. M. A., Silva, D. F., e Batista, G. E. A. P. A. (2013a). Classification of data streams applied to insect recognition: Initial results. Em *Proceedings of the Brazilian Conference on Intelligent Systems (BRACIS)*, páginas 76–81. Citado nas páginas 7, 62, 96, 98, 99, e 145.

Souza, V. M. A., Silva, D. F., e Batista, G. E. A. P. A. (2014). Extracting texture features for time series classification. Em *Proceedings of the International Conference on Pattern Recognition (ICPR)*, páginas 1425–1430. Citado nas páginas 145 e 175.

Souza, V. M. A., Silva, D. F., e Batista, G. E. A. P. A. (2015b). Classification of evolving data streams with totally delayed labels. Em *Proceedings of the International Conference on Machine Learning & Applications (ICMLA)*, páginas 214–219. Citado nas páginas 6, 46, 51, 52, 57, e 144.

Souza, V. M. A., Silva, D. F., Gama, J., e Batista, G. E. A. P. A. (2015c). Data stream classification guided by clustering on nonstationary environments and extreme verifi-

cation latency. Em *Proceedings of the SIAM International Conference on Data Mining (SDM)*, páginas 873–881. Citado nas páginas 6, 46, 51, 52, 62, e 144.

Souza, V. M. A., Silva, D. F., Garcia, P. R., e Batista, G. E. A. P. A. (2013b). Avaliação de classificadores para o reconhecimento automático de insetos. Em *Proceedings of Encontro Nacional de Inteligência Artificial e Computacional (ENIAC)*, páginas 1–12. Citado nas páginas 7, 98, 99, e 145.

Stevens, S. S., Volkmann, J., e Newman, E. B. (1937). A scale for the measurement of the psychological magnitude pitch. *Journal of the Acoustical Society of America*, 8(3):185–190. Citado na página 94.

Tax, D. M. J. (2001). *One-class classification*. Tese de Doutorado, TU Delft, Delft University of Technology. Citado nas páginas 103 e 108.

Tax, D. M. J. (2015). Ddtools, the data description toolbox for matlab. version 2.1.2. Citado na página 103.

Tax, D. M. J. e Duin, R. P. W. (2000). Data description in subspaces. Em *Proceedings of the International Conference on Pattern Recognition*, volume 2, páginas 672–675. Citado na página 104.

Tax, D. M. J. e Duin, R. P. W. (2004). Support vector data description. *Machine learning*, 54(1):45–66. Citado na página 105.

Taylor, L. R. (1963). Analysis of the effect of temperature on insects in flight. *Journal of Animal Ecology*, 32(1):99–117. Citado nas páginas 3 e 114.

Terasawa, H., Slaney, M., e Berger, J. (2005). The thirteen colors of timbre. Em *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, páginas 323–326. Citado na página 155.

Tibes, C. M. S., Cherman, E. A., Souza, V. M. A., Westin, U. M., Zem-Mascarenhas, S. H., e Evora, Y. D. M. (2015). Avaliação de um aplicativo para apoio à decisão no cuidado de Úlceras por pressão. Em *Anais do Congresso Internacional de Informática Educativa (TISE)*, páginas 191–199. Citado na página 145.

Tsymbal, A. (2004). The problem of concept drift: Definitions and related work. Relatório Técnico TCD-CS-2004-15, Trinity College Dublin. Citado na página 16.

Unwin, D. M. e Ellington, C. P. (1979). An optical tachometer for measurement of the wing-beat frequency of free-flying insects. *Journal of Experimental Biology*, 82(1):377–378. Citado na página 81.

Vapnik, V. (1998). *Statistical learning theory*, volume 1. Wiley New York. Citado na página 55.

Vasconcelos, P. F. C. (2015). Doença pelo vírus zika: um novo problema emergente nas américas? *Revista Pan-Amazônica de Saúde*, 6(2):9–10. Citado na página 4.

Vorburger, P. e Bernstein, A. (2006). Entropy-based concept shift detection. Em *Proceedings of the IEEE International Conference on Data Mining (ICDM)*, páginas 1113–1118. Citado na página 22.

Vreysen, M. J. B. e Robinson, A. S. (2011). Ionising radiation and area-wide management of insect pests to promote sustainable agriculture. a review. *Agronomy for sustainable development*, 31(1):233–250. Citado na página 78.

Waldbauer, G. (2009). *Fireflies, honey, and silk*. University of California Press. Citado nas páginas 4 e 86.

Walker, K. (2002). A review of control methods for african malaria vector. Relatório Técnico 108, Bureau for Global Health. Citado na página 79.

Wang, H., Fan, W., Yu, P., e Han, J. (2003). Mining concept-drifting data streams using ensemble classifiers. Em *Proceedings of the ACM SIGKDD international conference on Knowledge discovery and data mining (KDD)*, páginas 226–235. Citado na página 127.

Wang, W., Yu, X., Wang, Y. H., e Swaminathan, R. (2012). Audio fingerprint based on spectral flux for audio retrieval. Em *Proceedings of the International Conference on Audio, Language and Image Processing (ICALIP)*, páginas 1104–1107. Citado na página 93.

Wang, X., Mueen, A., Ding, H., Trajcevski, G., Scheuermann, P., e Keogh, E. (2013). Experimental comparison of representation methods and distance measures for time series data. *Data Mining and Knowledge Discovery*, 26(2):275–309. Citado nas páginas 165 e 170.

W.H.O. (2014). The world malaria report. Relatório técnico, World Health Organization. Citado na página 78.

W.H.O. (2015). Dengue and severe dengue. Relatório Técnico Fact Sheet 117, World Health Organization. Citado nas páginas 4 e 78.

Widmer, G. e Kubat, M. (1993). Effective learning in dynamic environments by explicit context tracking. Em *Proceedings of the European Conference on Machine Learning (ECML)*, páginas 227–243. Citado na página 17.

Widmer, G. e Kubat, M. (1996). Learning in the presence of concept drift and hidden contexts. *Machine learning*, 23(1):69–101. Citado nas páginas 1, 14, 16, e 21.

Wiegmann, B. e Yeates, D. (1996). Tree of life: Diptera. *The Tree of Life: Diptera*. Citado nas páginas 88 e 102.

Wilson, E. O. (1999). *The diversity of life*. W. W. Norton & Company, 2nd edição. Citado na página 78.

Ye, L. e Keogh, E. (2009). Time series shapelets: a new primitive for data mining. Em *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, páginas 947–956. Citado na página 157.

Yoshii, K., Goto, M., e Okuno, H. G. (2007). Drum sound recognition for polyphonic audio signals by adaptation and matching of spectrogram templates with harmonic structure suppression. *IEEE Transactions on Audio, Speech, and Language Processing*, 15(1):333–345. Citado na página 148.

Zhang, T., Ramakrishnan, R., e Livny, M. (1996). Birch: an efficient data clustering method for very large databases. Em *ACM SIGMOD Record*, volume 25, páginas 103–114. Citado na página 57.

Zhou, D., Bousquet, O., Lal, T. N., Weston, J., e Schölkopf, B. (2004). Learning with local and global consistency. *Advances in neural information processing systems*, 16(16):321–328. Citado na página 184.

Zhu, X. (2010). Stream data mining repository. `www.cse.fau.edu/~xqzhu/stream.html`. Citado nas páginas 34 e 35.

Zhu, X., Ghahramani, Z., e Lafferty, J. (2003). Semi-supervised learning using gaussian fields and harmonic functions. Em *Proceedings of the International Conference on Machine Learning (ICML)*, volume 3, páginas 912–919. Citado na página 184.

Zhu, X. e Goldberg, A. B. (2009). Introduction to semi-supervised learning. *Synthesis lectures on artificial intelligence and machine learning*, 3(1):1–130. Citado na página 184.

Zhu, X., Wu, X., e Yang, Y. (2004). Dynamic classifier selection for effective mining from noisy data streams. Em *Proceedings of the International Conference on Data Mining (ICDM)*, páginas 305–312. Citado nas páginas 6 e 127.

Zhu, X., Wu, X., e Yang, Y. (2006). Effective classification of noisy data streams with attribute-oriented dynamic classifier selection. *Knowledge and Information Systems*, 9(3):339–363. Citado na página 127.

Žliobaitė, I. (2009a). Combining time and space similarity for small size learning under concept drift. Em *Foundations of Intelligent Systems*, páginas 412–421. Citado nas páginas 35 e 36.

Žliobaitė, I. (2009b). Learning under concept drift: an overview. Relatório técnico, Faculty of Mathematics and Informatics of Vilnius University, Lithuania. Citado nas páginas 15 e 19.

Žliobaitė, I., Bifet, A., Read, J., Pfahringer, B., e Holmes, G. (2015). Evaluation methods and decision theory for classification of streaming data with temporal dependence. *Machine Learning*, 98(3):455–482. Citado nas páginas 61 e 62.