

UNIVERSIDADE ESTADUAL PAULISTA
“Júlio de Mesquita Filho”

Pós-Graduação em Ciência da Computação

Ricardo Ricci Lopes

APRENDIZADO DE MÁQUINA BASEADO EM
TENSORES E SUAS APLICAÇÕES PARA FLORESTA
DE CAMINHOS ÓTIMOS

UNESP
2015

Lopes, Ricardo Ricci.

Aprendizado de máquina baseado em tensores e suas aplicações
para floresta de caminhos ótimos / Ricardo Ricci Lopes. -- São José do
Rio Preto, 2015

50 f. : il., tabs.

Orientador: João Paulo Papa

Dissertação (mestrado) – Universidade Estadual Paulista “Júlio de
Mesquita Filho”, Instituto de Biociências, Letras e Ciências Exatas

1. Computação - Matemática. 2. Processamento de imagens
Técnicas digitais. 3. Reconhecimento de padrões. 4. Álgebra tensorial.
5. Aprendizado do computador. 6. Floresta de caminhos ótimos. I. Papa,
João Paulo. II. Universidade Estadual Paulista "Júlio de Mesquita Filho".
Instituto de Biociências, Letras e Ciências Exatas. III. Título.

CDU – 518.72:76

Ficha catalográfica elaborada pela Biblioteca do IBILCE
UNESP - Câmpus de São José do Rio Preto

Ricardo Ricci Lopes

APRENDIZADO DE MÁQUINA BASEADO EM
TENSORES E SUAS APLICAÇÕES PARA FLORESTA
DE CAMINHOS ÓTIMOS

Orientador: Prof. Dr. João Paulo Papa

Dissertação de Mestrado elaborada junto ao Programa de Pós-Graduação em Ciência da Computação - Área de Concentração em Processamento de Imagens e Visão Computacional, como parte dos requisitos para a obtenção do título de Mestre em Ciência da Computação.

UNESP
2015

Ricardo Ricci Lopes

Aprendizado de Máquina Baseado em Tensores e suas Aplicações
para Floresta de Caminhos Ótimos

Dissertação apresentada como parte dos requisitos para obtenção do título de Mestre em Ciência da Computação, junto ao Programa de Pós-Graduação em Ciência da Computação, do Instituto de Biociências, Letras e Ciências Exatas da Universidade Estadual Paulista "Júlio de Mesquita Filho", Campus de São José do Rio Preto.

Comissão Examinadora

Prof. Dr. João Paulo Papa
UNESP – Bauru
Orientador



Prof. Dr. Alexandre Levada
UFSCAR – São Carlos



Prof. Dr. Antonio Carlos Sementille
UNESP – Bauru



São José do Rio Preto
03 de agosto de 2015

Á minha família e namorada, por estarem sempre ao meu lado em todos os momentos durante todos esses anos.

Agradecimentos

Aos meus pais, Antônio Cesar Lopes e Eleusa Paula G. R. Lopes, pelo suporte, confiança e amor incondicional.

Às minhas irmãs, Ana Elisa e Ana Carolina, pelas cobranças (mesmo que indevidas), amor e companhia durante todos esses anos. À toda minha família por estarem sempre presentes.

À minha namorada Juliana, pela dedicação, companheirismo, compreensão e alegrias compartilhadas desde o meu primeiro ano em Bauru.

Ao me orientador João Paulo Papa, por acreditar em mim e sempre me ajudar, principalmente nos últimos meses, que esteve sempre presente (mesmo muito longe). Obrigado por todos ensinamentos e paciência. A ele todo meu respeito e admiração.

Ao meu amigo Tiago, pela ajuda durante tantos anos.

Aos colegas do grupo Recogna e a todos da república Bauruzinho, pelas inúmeras conversas, discussões, risadas e ajuda para resolver qualquer problema.

À UNESP e ao CNPq pela ajuda financeira, laboratórios e excelentes professores durante tantos anos.

A Deus pela força e por permitir-me chegar onde cheguei.

Sumário

1	Introdução	1
1.1	Motivação e justificativa	1
1.2	Objetivos	3
1.3	Organização	3
2	Representação de Dados Baseada em Tensores	5
2.1	Álgebra tensorial	6
2.2	Técnicas baseadas em tensores	8
3	Máquinas de Vetores de Suporte	11
3.1	Representação espaço-vetorial	11
3.2	Representação espaço-tensorial	12
4	Análise de Componentes Principais Multilineares	15
4.1	Fundamentação teórica	15
4.2	Projeções multilineares	16
4.2.1	Projeção vetor-vetor	16
4.2.2	Projeção tensor-tensor	17
4.2.3	Projeção tensor-vetor	18
4.3	MPCA	18
4.4	Extração de características	19
5	Floresta de Caminhos Ótimos	23
5.1	Fundamentação teórica	23
5.2	OPF com grafo completo	23
5.2.1	Treinamento	25
5.2.2	Classificação	26
5.3	OPF com grafo k -nn	26

5.3.1	Fundamentação teórica	27
5.3.2	Treinamento	28
5.3.3	Classificação	30
6	Metodologia	33
7	Resultados e Análise	37
7.1	Faces	37
7.2	Modo de andar	39
7.2.1	Validação cruzada	39
7.2.2	Galeria e provas	39
8	Conclusões	43
8.1	Principais contribuições	43
8.2	Trabalhos futuros	44
8.2.1	Combinação de espaços vetoriais-tensoriais	44
8.2.2	Otimização evolucionista aplicada ao mapeamento vetorial-tensorial	44
	Referências Bibliográficas	47

Lista de Figuras

2.1	(a) Exemplo de mapeamento de um vetor para um tensor de segunda ordem, e (b) ilustração de tensor de terceira ordem em $\Re^{l_1 \times l_2 \times l_3}$	6
2.2	Ilustração de m-modos de um vetor: (a) um tensor $\mathcal{A} \in \Re^{8 \times 6 \times 4}$, (b) vetores do 1-modo, (c) vetores do 2-modo, e (d) vetores do 3-modo.	8
4.1	Representação de uma projeção vetor-vetor.	17
4.2	Representação de uma projeção tensor-tensor.	17
4.3	Representação de uma projeção tensor-vetor, sendo EMP projeções multilineares elementares.	18
4.4	Diagrama para representação do algoritmo MPCA+LDA. . . .	21
5.1	(a) Grafo completo ponderado nas arestas para um determinado conjunto de treinamento. (b) MST do grafo completo. (c) Floresta de caminhos ótimos resultante para f_{max} e dois protótipos (nós contornados). As entradas (x, y) acima dos nós denotam, respectivamente, o custo e o rótulo das instâncias. Os arcos direcionados indicam os nós predecessores no caminho ótimo. (d) Amostra de teste (triângulo) e suas conexões (linhas tracejadas) com os nós de treinamento. (e) O caminho ótimo do protótipo mais fortemente conexo, seu rótulo 2, e o custo de classificação 0.3 é associado à instância de teste. . . .	31
6.1	Metodologia proposta para aplicação de OPF em espaços tensoriais.	33
6.2	Exemplos das bases de dados (a) Gait-based Human ID e (b) AT&T Face datasets.	35

Lista de Tabelas

4.1	Comparação entre subespaços lineares e multilineares.	16
6.1	Conjuntos de amostras diferentes para classificação.	34
7.1	Acurácias obtidas considerando o AT&T Face Dataset.	38
7.2	Acurácias obtidas considerando o experimento com “dados aleatórios” para a base Gait-based Human ID dataset.	40
7.3	Taxa de reconhecimento considerando o experimento da galeria e provas para a base Gait-based Human ID dataset.	41

Lista de abreviaturas e siglas

FIFO	First-in-first-out
LDA	Linear discriminant analysis
MPCA	Multilinear Principal Componente Analysis
MST	Minimum Spanning Tree
OPF	Optimum-Path Forest
PCA	Principal Componente Analysis
RBF	Radial Basis Function
STM	Support Tensor Machines
SVD	Singular Value Decomposition
SVM	Support Vecor Machines
TSM	Tensor Space Model
VSM	Vector Space Model

Resumo

Técnicas de aprendizado de máquina, usualmente, objetivam aprender alguma superfície que separe amostras de classes diferentes por meio de sua representação vetorial. Entretanto, existem muitas aplicações que podem, eventualmente, perder informações essenciais e inerentes da estrutura dos dados em tal representação e, com o crescimento de base de dados com alta dimensionalidade, essas informações se tornam cada vez mais importantes. Os espaços de representação de dados com curvatura, baseados em trabalhos na área da Matemática e Física, têm despertado interesse por parte da comunidade de aprendizado de máquina com o intuito de resolver tal situação. Esses espaços de representação são baseados em tensores, os quais mantêm a estrutura original dos dados, bem como permitem a utilização de variedades em superfícies com curvatura não nula. Esta dissertação de mestrado apresenta uma revisão bibliográfica sobre abordagens de aprendizado de máquina baseadas em tensores, bem como um referencial teórico sobre álgebra multilinear. Também é apresentado um estudo da aplicabilidade do classificador Floresta de Caminhos Ótimos, do inglês *Optimum-Path Forest* - OPF, em espaços tensoriais através da técnica Análise de Componentes Principais Multilineares, bem como a comparação dos resultados obtidos com outras técnicas conhecidas na literatura em contexto de reconhecimento em fotos e vídeos. Também foi demonstrado que o OPF pode obter maior acurácia em algumas situações quando se trabalha com características no espaço tensorial.

Palavras-chave: Tensores, Aprendizado de Máquina, Floresta de Caminhos Ótimos.

Abstract

Machine learning techniques usually learn some decision surface that separates samples from different classes by means of their vectorial representation. However, there exist many applications that might lose important information that are strongly related to the data itself. Additionally, such information has gained importance with the popularity of high-dimensional datasets. As such, works based on Mathematics and Physics, where curvature-based space representations have been used in several applications, have gained attention by the machine learning community. Such representations are based on tensors, which keep the original structure of the data, as well as they allow us to use manifolds in curvature-based spaces. This master's dissertation presents a review of the literature with respect to tensor-based machine learning techniques, as well as a brief review about multilinear algebra. We also evaluate the performance of the Optimum-Path Forest classifier (OPF) in tensor-oriented spaces by means of the Multilinear Principal Component Analysis, as well as its comparison against with other related techniques is also performed. It is shown OPF can benefit from such feature space representation in some situations.

Keywords: Tensor, Machine Learning, Optimum-Path Forest.

Capítulo 1

Introdução

1.1 Motivação e justificativa

A tarefa de representação de dados tem sido um dos problemas mais abordados pela comunidade científica na área de aprendizado de máquina nos últimos anos, dado que grande parte dessas técnicas são baseadas em um modelo de representação do tipo espaço-vetorial (*Vector Space Model* - VSM). Assim sendo, dada uma base de dados $\mathcal{X}$ e um conjunto de rótulos $\mathcal{Y}$, temos que cada amostra $\mathbf{x}_i \in \mathcal{X}$ é representada como sendo um ponto nesse espaço vetorial com N dimensões, ou seja, $\mathbf{x}_i \in \mathbb{R}^n$, $i = 1, 2, \dots, |\mathcal{X}|$. Basicamente, um algoritmo de aprendizado de máquina objetiva, então, encontrar uma função $f : \mathcal{X} \rightarrow \mathcal{Y}$ que leve a uma melhor separação do espaço de características, de tal forma que amostras com o mesmo rótulo pertençam ao mesmo grupo, ou a agrupamentos próximos [1].

Com os avanços em técnicas de armazenamento e criação de coleções de dados, uma grande quantidade de dados multidimensionais tem sido gerado diariamente. Esse novo ambiente digital de informações requer o estudo e desenvolvimento de algoritmos de aprendizado de máquina e extração de informação mais eficientes e, conseqüentemente, eficazes [2]. Imagens geradas por câmeras digitais, por exemplo, possuem milhões de *pixels*, e são inerentemente vetores bidimensionais. Sinais biomédicos multicanais, como é o caso do eletroencefalograma, imagens de ressonância magnética e vídeos, dentre outros, são exemplos de sinais que possuem, já em sua forma nativa, várias dimensões.

Desta forma, um segmento da comunidade de aprendizado de máquina,

em especial, dedica-se ao estudo de representações mais naturais para dados que possuem alta dimensão. Uma dessas representações recebe o nome de **tensor**, o qual é, basicamente, uma estrutura geométrica que possui certas relações com outros tensores, vetores e escalares. Pode-se considerar, por exemplo, uma imagem como sendo um tensor de segunda ordem, também conhecido como uma matriz convencional. Dessa forma, é razoável considerar que *pixels* próximos em determinadas regiões representam informações semelhantes, da mesma maneira que um vídeo é considerado um tensor de terceira ordem (cubo), sendo a terceira dimensão representada pelo tempo [1]. Pode-se dizer então que a representação de dados baseada em tensores preserva a topologia dos dados de acordo com a forma que eles estão representados na natureza.

Nesse contexto, existem vários trabalhos que se dedicam ao estudo de um modelo de representação do tipo espaço-tensorial (*Tensor Space Model* - TSM), o qual possui, basicamente, tensores ao invés de representações vetoriais do conjunto de dados, muito embora estas últimas sejam um caso particular de tensor, como será visto adiante. Assim, dá-se o nome de **álgebra multilinear** à área da Matemática que estuda as propriedades dos tensores, sendo esta área uma extensão da álgebra linear tradicional.

Mesmo recebendo mais destaque, não são muitos os trabalhos que fazem uso de modelos tensoriais na área de reconhecimento de padrões e aprendizado de máquina, muito embora a área de álgebra multilinear (que estuda propriedades de tensores) seja bem conhecida na Matemática. Desta forma, existe ainda uma carência no estudo da viabilidade de utilização de diversos outros classificadores em espaços tensoriais. Recentemente, Papa et al. [20, 21, 22] propuseram um classificador de padrões baseado em grafos chamado Floresta de Caminhos Ótimos (*Optimum-Path Forest* - OPF), o qual modela o problema de reconhecimento de padrões como sendo uma tarefa de particionamento de um grafo em componentes conexos. A ideia consiste em modelar as amostras da base de dados como sendo vértices de um grafo, os quais são conectados por uma relação de adjacência previamente definida. Após a escolha de um conjunto de nós protótipos (nós-chave), os mesmos iniciam um processo de disputa com o intuito de conquistar as demais amostras do conjunto de dados oferecendo caminhos de custo ótimo: quando uma amostra conquista outra, a esta é oferecido o rótulo do nó conquistador. Ao final desse processo, temos então um grafo particionado em componentes chamadas Árvores de Caminhos Ótimos, as quais são enraizadas por nós protótipos.

1.2 Objetivos

Atualmente, existem duas versões do classificador OPF supervisionado: (i) a primeira delas faz uso de um grafo completo como relação de adjacência [21, 22], e escolhe os protótipos nas regiões de fronteira entre as classes; (ii) e a segunda versão, chamada de OPF_k , faz uso de uma relação de adjacência do tipo k -vizinhos mais próximos e estima os protótipos nas regiões com alta densidade de amostras. Entretanto, nenhuma dessas versões do classificador OPF foi ainda avaliada em espaços tensoriais, sendo este o principal objetivo do presente trabalho, bem como contribuir com mais estudos sobre álgebra tensorial aplicada às técnicas de reconhecimento de padrões e aprendizado de máquina. Ademais, um estudo sobre o classificador STM será realizado, bem como a aplicação do classificador SVM também avaliada em espaços tensoriais com o intuito de realizar comparações entre o mesmo e o classificador OPF com características mapeadas em espaços tensoriais.

1.3 Organização

Este documento está organizado da seguinte maneira: o Capítulo 2 apresenta o referencial teórico acerca do modelo de espaço tensorial e técnicas de aprendizado de máquina baseadas em tensores. Já no Capítulo 3 temos a apresentação do tradicional classificador SVM e uma versão para o espaço-tensorial de ordem 2. A Análise de Componentes Principais Multilineares e a técnica Floresta de Caminhos Ótimos serão apresentadas nos Capítulos 4 e 5, respectivamente e a metodologia a ser empregada no presente projeto de pesquisa é apresentada no Capítulo 6. Por fim, os resultados obtidos e as conclusões são descritos nos Capítulos 7 e 8, respectivamente.

Capítulo 2

Representação de Dados Baseada em Tensores

Tradicionalmente, uma amostra é representada por um vetor no chamado espaço-vetorial, o qual pode ser caracterizado por uma pequena ou alta dimensão. Nesse contexto, alguns algoritmos de aprendizado são então aplicados nesse espaço para a classificação dos dados, como de costume. Já no modelo espaço-tensorial, cada amostra $\mathbf{x}_i \in \mathcal{X}$ da base de dados, $i = 1, 2, \dots, |\mathcal{X}|$, é representada como sendo um tensor $\hat{\mathbf{x}}_i \in \mathcal{T}$, onde $\mathcal{T}$ representa esse espaço tensorial. Um exemplo seria a transformação da amostra $\mathbf{x}_i \in \mathbb{R}^n$ em sua representação tensorial de segunda ordem (matriz) $\hat{\mathbf{x}}_i \in \mathbb{R}^{n_1 \times n_2}$, onde $n_1 \times n_2 \approx n$. A Figura 2.1a ilustra exemplos de transformação de um vetor para um tensor de segunda ordem. Na verdade, um escalar é um tensor de ordem 0, ao passo que vetores e matrizes são tensores de ordem 1 e 2, respectivamente. Outro exemplo é o tensor de terceira ordem, conforme ilustrado na Figura 2.1b.

Como pode ser observado, existem várias maneiras de transformar um vetor em um tensor, sendo este um dos principais desafios da área de aprendizado de máquina baseada em tensores. Um outro ponto importante a ser considerado diz respeito à organização dos elementos em um tensor, ou seja, à ordem com a qual os elementos serão dispostos nesse tensor e também a forma que ele foi mapeado. A próxima seção apresenta alguns fundamentos básicos sobre álgebra tensorial.

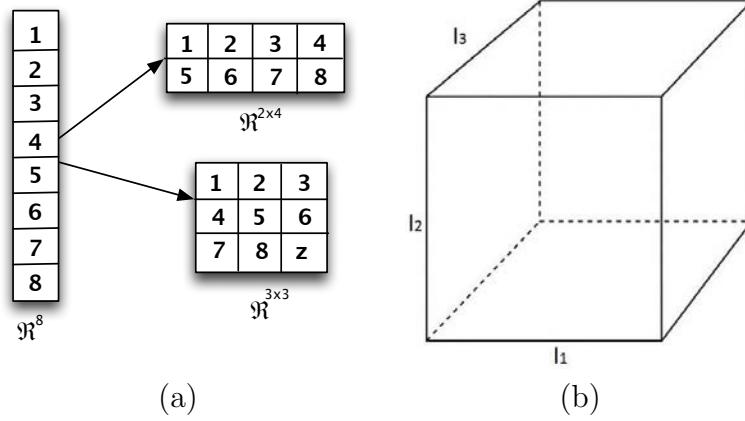


Figura 2.1: (a) Exemplo de mapeamento de um vetor para um tensor de segunda ordem, e (b) ilustração de tensor de terceira ordem em $\mathfrak{R}^{l_1 \times l_2 \times l_3}$.

2.1 Álgebra tensorial

Esta seção apresenta alguns fundamentos sobre a Álgebra Multilinear, a qual é responsável pela manipulação de dados em espaços tensoriais. Temos que um tensor T de ordem l é, na verdade, uma função multilinear¹ real-valorada em l espaços vetoriais dada por:

$$T : \mathfrak{R}^{n_1} \times \mathfrak{R}^{n_2} \times \dots \times \mathfrak{R}^{n_l} \rightarrow \mathfrak{R}. \quad (2.1)$$

O conjunto de todos os l -tensores em $\mathfrak{R}^{n_i}$, $i = 1, 2, \dots, l$, denotado por $\mathcal{T}^l$, é um espaço vetorial regido pelas operações pontuais² de adição e multiplicação por um escalar dadas, respectivamente, por:

$$(T + T')(a_1, a_2, \dots, a_l) = T(a_1, a_2, \dots, a_l) + T'(a_1, a_2, \dots, a_l) \quad (2.2)$$

e

$$(\lambda T)(a_1, a_2, \dots, a_l) = \lambda(T(a_1, a_2, \dots, a_l)), \quad (2.3)$$

¹Uma mapeamento multilinear é uma função de várias variáveis que é linear em cada uma delas separadamente.

²Operações aritméticas pontuais são realizadas para cada elemento da imagem de uma função.

onde $\mathbf{a}_i \in \mathbb{R}_i^n$ e T' denota um tensor de ordem l . Ainda, dados dois tensores $S \in \mathcal{T}^m$ e $T \in \mathcal{T}^l$, o seu produto tensorial, também conhecido como produto de Kronecker, o mapeamento de $S \otimes T$ é dado por:

$$S \otimes T : \mathbb{R}^{n_1} \times \dots \times \mathbb{R}^{n_m} \times \mathbb{R}^{n_{m+1}} \times \dots \times \mathbb{R}^{n_{m+l}} \rightarrow \mathbb{R}, \quad (2.4)$$

sendo que $S \otimes T$ denota então um tensor de ordem $(m + l)$. Já o produto escalar de dois tensores $T_1, T_2 \in \mathbb{R}^{n_1 \times n_2 \dots \times n_l}$ é definido como:

$$\langle T_1, T_2 \rangle = \sum_{a_1} \sum_{a_2} \dots \sum_{a_l} T_1(a_1, a_2, \dots, a_l) \cdot T_2(a_1, a_2, \dots, a_l), \quad (2.5)$$

onde o operador $\cdot$ denota a multiplicação pontual.

Outra importante definição diz respeito à norma de Frobenius, dada por [26]:

$$\| T \|_F = \sqrt{\langle T, T \rangle}, \quad (2.6)$$

a qual é utilizada para calcular a distância $d : \mathcal{T}^l \times \mathcal{T}^l \rightarrow \mathbb{R}$ entre dois tensores, dada por:

$$d(T_1, T_2) = \| T_1 - T_2 \|_F. \quad (2.7)$$

Um espaço tensorial de primeira ordem (espaço onde os vetores tradicionais são representados) é, na verdade, um co-vetor em $\mathbb{R}^{n_1}$, ou seja, $\mathcal{T}^1 = \mathcal{R}^{n_1}$, onde $\mathcal{R}^{n_1}$ é o espaço dual³ de $\mathbb{R}^{n_1}$. Já o espaço tensorial de segunda ordem (espaço onde as matrizes são representadas) é considerado como um produto tensorial de dois espaços tensoriais de primeira ordem, ou seja, $\mathcal{T}^2 = \mathcal{R}^{n_1} \otimes \mathcal{R}^{n_2}$. Essas informações permitem afirmar que todo tensor em $\mathcal{R}^{n_1} \otimes \mathcal{R}^{n_2}$ corresponde, na verdade, à uma matriz com dimensões $n_1 \times n_2$ [13]. Tal afirmativa é de grande importância, pois podemos mapear os tradicionais vetores de características para matrizes, que são então tensores de segunda ordem (Figura 2.1a).

Outra definição importante é o modo, sendo os modos de um tensor as suas diferentes dimensões. Por exemplo, um tensor $\mathcal{A} \in \mathbb{R}^{8 \times 6 \times 4}$ de ordem 3, representado na Figura 2.2a, tem seu 1-modo ilustrado na Figura 2.2b,

³Um espaço dual $\mathcal{V}^*$ de $\mathcal{V}$ é definido como o conjunto de todos os funcionais lineares, ou seja, os mapeamentos lineares de um espaço vetorial para o seu campo de escalares. Um co-vetor pode ser visto, na verdade, como um “vetor nesse espaço dual”.

seu 2-modo ilustrado na Figura 2.2c e seus vetores de 3-modo ilustrado na Figura 2.2d. Na próxima seção serão apresentadas algumas técnicas baseadas em tensores e suas aplicações.

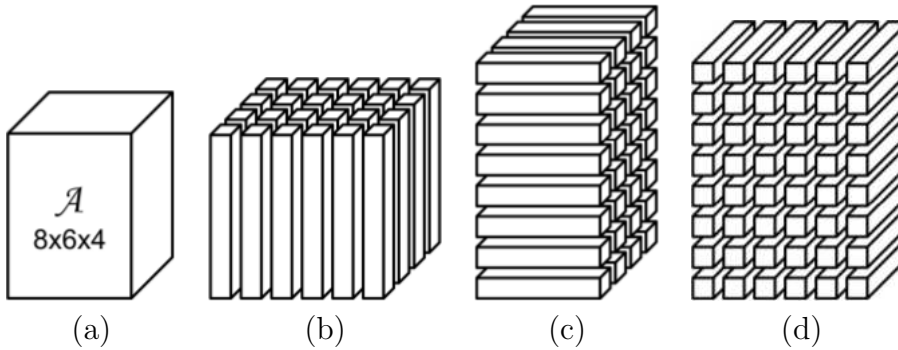


Figura 2.2: Ilustração de m-modos de um vetor: (a) um tensor $\mathcal{A} \in \mathbb{R}^{8 \times 6 \times 4}$, (b) vetores do 1-modo, (c) vetores do 2-modo, e (d) vetores do 3-modo.

2.2 Técnicas baseadas em tensores

Alguns trabalhos têm se dedicado ao estudo de espaços tensoriais em aplicações na área de Visão Computacional. Vasilescu e Terzopoulos [3], por exemplo, utilizaram modelos tensoriais para redução de dimensionalidade visando o reconhecimento de indivíduos por meio da face, sendo que cada imagem era então representada por um tensor de alta ordem. Sua técnica proposta, chamada TensorFace, utilizou uma versão da tradicional decomposição em valores singulares (*Singular Value Decomposition* - SVD) para espaços tensoriais, mostrando bons resultados em situações com certos problemas de iluminação e sombra. Alguns outros trabalhos mostraram como estender técnicas tradicionais como Análise de Componentes Principais e Análise Linear Discriminante [4, 5] para modelos tensoriais. Várias outras abordagens baseadas em tensores podem ser encontradas no contexto de reconhecimento facial humano [6, 7]. Conforme dito anteriormente, pelo fato de uma imagem digital ser, naturalmente, um tensor de ordem 2, ou seja, uma matriz, vários trabalhos envolvendo reconhecimento de pessoas por meio de imagens faciais podem ser encontrados na literatura.

Uma outra área bastante ativa no que diz respeito a modelos baseados em tensores é a de extração de conhecimento em conteúdo visual e textual. Dado

que, geralmente, a tarefa de classificação de textos envolve a criação de um dicionário de palavras para posterior contagem da frequência de ocorrência de cada palavra deste dicionário no texto como sendo a sua representação no espaço de características tradicional, tal problema é usualmente tratado como sendo de alta dimensão. Assim, torna-se um bom candidato para ser representado em espaços tensoriais. Sutskever et al. [8], por exemplo, aplicaram fatorização tensorial e agrupamento Bayesiano para o aprendizado de estruturas relacionais em textos, e Ranzato et al. [9] aplicaram um modelo de arquitetura de aprendizado em profundidade parametrizado por tensores para a extração de conhecimento em imagens naturais.

Yu et al. [10] também aplicaram uma arquitetura de aprendizado em profundidade baseada em tensores, mas no contexto de reconhecimento de fala, sendo que uma extensão desse modelo para análise de sentimentos foi proposta por Socher et al. [11]. O mesmo grupo de pesquisa também aplicou Redes Neurais Tensoriais para a extração de conhecimento em bases com conteúdo textual [11]. Hutchinson et al. [12] propuseram uma arquitetura escalável visando a paralelização de redes neurais em profundidade baseadas em espaços tensoriais.

Cai et al. [13] foram um dos primeiros a abordar a tradicional técnica de Máquinas de Vetores de Suporte (*Support Vector Machines* - SVM) baseadas em tensores: as chamadas *Support Tensor Machines* (STM), as quais foram validadas no contexto de categorização de textos. As amostras foram mapeadas como sendo tensores de segunda ordem, e o problema de identificar um hiperplano separador com máxima margem foi mapeado para um espaço tensorial. Posteriormente, vários outros trabalhos utilizaram STM nas mais variadas aplicações [14, 15, 19], que vão desde segmentação de imagens à tarefas que envolvem regressão de dados. O modelo de representação tensorial para o classificador SVM e sua versão tradicional serão descritos na próxima seção.

Capítulo 3

Máquinas de Vetores de Suporte

Neste capítulo são apresentados conceitos básicos sobre o classificador SVM em suas representações espaço-vetorial e espaço-tensorial. Também é exposto um pseudo-código das *Support Tensor Machines*, propostas por Zhang et al. [14].

3.1 Representação espaço-vetorial

Máquinas de Vetores de Suporte são uma família de algoritmos, propostos, inicialmente, por Vapnik et al. [27]. O algoritmo de treinamento do SVM é baseado na ideia da minimização do risco estrutural, ao invés da minimização do risco empírico, possibilitando novos modelos de treinamento em classificadores polinomiais, redes neurais e funções de base radial.

Considere o SVM em um cenário de classificação binária. Assumindo que temos um conjunto de dados rotulado $D = \{\mathbf{x}_i, y_i\}_{i=1}^m$, onde $y_i \in \{-1, 1\}$, queremos selecionar, entre os infinitos classificadores lineares que separam esses dados, aquele que minimiza os erros de classificação nesse conjunto de dados. Um hiperplano com essa propriedade é aquele definido pela margem máxima entre as duas classes. Este hiperplano pode ser calculado da seguinte maneira:

$$f(x) = \mathbf{w}^T \mathbf{x} + b, \quad (3.1)$$

onde $\mathbf{w}$ é um vetor ortogonal a esse hiperplano. Determinar o melhor hiper-

plano é considerado um problema de otimização e resolvido por técnicas de programação quadrática.

A otimização do problema do SVM é definida em 3.2, onde C é uma constante de regularização, ponderando a minimização dos erros no conjunto de treinamento, ou seja, quanto maior C , menor a quantidade de classificações erradas. A variável de folga ε_i é um escalar não-negativo e indica erros no treinamento para valores de $\varepsilon_i > 1$.

$$\begin{aligned} \min_{\mathbf{w}, b, \varepsilon} \quad & \frac{1}{2} \mathbf{w}^T \mathbf{w} + C \sum_{i=1}^m \varepsilon_i \\ \text{sujeito a} \quad & y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1 - \varepsilon_i \\ & \varepsilon_i \geq 0, \quad i = 1, \dots, m. \end{aligned} \quad (3.2)$$

Dada uma nova amostra $\mathbf{x}$ para classificar, uma classe é determinada de acordo com seu relacionamento com as margens de decisão, sendo essa a função de decisão:

$$g(\mathbf{x}) = \text{sign}(\mathbf{w}^T \mathbf{x} + b). \quad (3.3)$$

3.2 Representação espaço-tensorial

Dado um conjunto de amostras para treinamento $\{\hat{\mathcal{X}}_i, y_i\}, i = 1, \dots, m$, onde $\hat{\mathcal{X}}_i$ é um tensor em um espaço tensorial de ordem 2, $\hat{\mathcal{X}}_i \in \mathcal{R}^{n_1} \otimes \mathcal{R}^{n_2}$ e $y_i \in \{-1, 1\}$ é a classe associada a $\mathcal{X}_i$. Deseja-se encontrar o tensor classificador:

$$f(\hat{\mathcal{X}}) = \mathbf{u}^T \hat{\mathcal{X}} \mathbf{v} + b, \quad (3.4)$$

que separe as duas classes com a maior margem, onde $\mathbf{u}$ e $\mathbf{v}$ são, respectivamente, vetores ortonormais em $\mathcal{R}^{n_1}$ e $\mathcal{R}^{n_2}$.

Sendo o algoritmo do STM uma generalização do SVM, o mesmo pode ser definido pelo seguinte procedimento [14]:

1. **Inicialização:** Seja $\mathbf{u} = (1, \dots, 1)^T$
2. **Cálculo de $\mathbf{v}$:** Seja $\tilde{\mathbf{x}}_i = \hat{\mathcal{X}}_i^T \mathbf{u}$ e $\beta_1 = \|\mathbf{u}\|^2$, $\mathbf{v}$ pode ser computado

pela solução do seguinte problema de otimização:

$$\begin{aligned}
& \min_{v, b, \varepsilon} && \frac{1}{2} \beta_1 \mathbf{v}^T \mathbf{v} + C \sum_{i=1}^m \varepsilon_i \\
& \text{sujeito a} && y_i(\mathbf{v}^T \tilde{\mathbf{x}}_i + b) \geq 1 - \varepsilon_i \\
& && \varepsilon_i \geq 0, \quad i = 1, \dots, m.
\end{aligned} \tag{3.5}$$

O problema de otimização da Equação 3.5 é o mesmo da Equação 3.2 no algoritmo padrão do SVM, ou seja, qualquer outro método computacional para o SVM também pode ser usado.

3. **Cálculo de $\mathbf{u}$:** Após o $\mathbf{v}$ encontrado, seja $\tilde{\mathbf{x}}_i = \hat{\mathbf{x}}_i^T \mathbf{v}$ e $\beta_2 = \|\mathbf{v}\|^2$. O valor de $\mathbf{u}$ pode ser obtido pela resolução do seguinte problema de otimização

$$\begin{aligned}
& \min_{u, b, \varepsilon} && \frac{1}{2} \beta_2 \mathbf{u}^T \mathbf{u} + C \sum_{i=1}^m \varepsilon_i \\
& \text{sujeito a} && y_i(\mathbf{u}^T x_i + b) \geq 1 - \varepsilon_i \\
& && \varepsilon_i \geq 0, \quad i = 1, \dots, m.
\end{aligned} \tag{3.6}$$

Como anteriormente, o problema de otimização de 3.6 é o mesmo da Equação 3.2 no SVM e podemos usar métodos computacionais para resolvê-lo.

4. **Computar iterativamente $\mathbf{u}$ e $\mathbf{v}$:** Pelos passos 2 e 3, podemos iterativamente encontrar $\mathbf{u}$ e $\mathbf{v}$ até que eles convergem.

Capítulo 4

Análise de Componentes Principais Multilineares

Análise de Componentes Principais (*Principal Component Analysis* - PCA) é uma tradicional técnica linear e não supervisionada usada para redução de dimensionalidade. É comum que o PCA redimensione os dados para vetores, o qual demanda alto custo de processamento e gasto de memória. Por exemplo, ao utilizar-se o PCA em um vídeo com dimensionalidade $(180 \times 140 \times 20)$, a vetorização deste vídeo implicaria em um vetor de dimensões (504000×1) , o qual pode demandar capacidade computacional superior a muitas máquinas atuais suportariam.

Baseado no tradicional PCA, Lu et al. [16] propuseram a Análise de Componentes Principais Multilineares (*Multilinear Principal Component Analysis* - MPCA). Serão abordados neste capítulo a projeção de objetos tensoriais e a extração de características desses objetos utilizando o MPCA.

4.1 Fundamentação teórica

A aprendizagem em subespaço multilinear é uma extensão de um subespaço linear, sendo seu objetivo otimizar uma projeção multilinear, dado um conjunto de treinamento, com critérios ótimos. A Tabela 4.1¹ apresenta as principais diferenças entre aprendizagem em subespaços lineares e multilineares.

Dado um conjunto com M amostras tensoriais $\mathcal{X}_1, \mathcal{X}_2, \dots, \mathcal{X}_n$ de ordem N em um espaço tensorial $\mathfrak{R}^{I_1 \otimes I_2 \otimes \dots \otimes I_N}$, o objetivo da aprendizagem multili-

¹São utilizados como critério a capacidade computacional de máquinas comuns.

Subespaço linear	Subespaço multilinear
Redimensiona em vetores	Representações tensoriais naturais
Quebra da estrutura dos dados	Preserva a estrutura dos dados
Grande quantidade de parâmetros	Poucos parâmetros para estimar
Problema com poucas amostras	Menos afetado pelo problema de poucas amostras
Difícil aplicação com muitos dados	Capaz de lidar com muitos dados
Soluções fechadas	Subótimo, solução iterativa

Tabela 4.1: Comparação entre subespaços lineares e multilineares.

near é encontrar uma transformação multilinear (projeção) onde as amostras projetadas satisfaçam um critério ótimo e a dimensionalidade do subespaço projetado seja muito inferior ao espaço tensorial original. O espaço tensorial pode ser representado pelo produto de Kronecker (Equação 2.1) de N vetores $\mathbb{R}^{I_1, I_2, \dots, I_N}$, onde essas características projetadas são utilizadas em um classificador.

É possível encontrar um subespaço tensorial que capture a maioria das variações do objeto tensorial e que seja possível a extração de características para reconhecimento e classificação. A projeção de um tensor $\hat{\mathcal{X}}$, no espaço tensorial citado acima, para um outro tensor $\hat{\mathcal{X}}'$ em um espaço tensorial $\mathbb{R}^{P_1 \otimes P_2 \otimes \dots \otimes P_N}$ de menor dimensionalidade, pode ser obtido pela Equação 4.2, que será apresentada na seção a seguir.

4.2 Projeções multilineares

Um subespaço multilinear é definido através de uma projeção que realize o mapeamento de um espaço para outro em outra dimensionalidade. Nessa seção serão apresentadas três tipos de projeções: vetor-vetor, tensor-tensor e tensor-vetor.

4.2.1 Projeção vetor-vetor

Um vetor $\mathbf{x} \in \mathbb{R}^I$ pode ser projetado linearmente para outro $\mathbf{y} \in \mathbb{R}^P$ usando uma matriz projeção $\mathbf{U} \in \mathbb{R}^{I \times P}$, ou seja:

$$\mathbf{y} = \mathbf{U}^T \mathbf{x} = \mathbf{x} \times \mathbf{U}^T. \quad (4.1)$$

Em aplicações de reconhecimento de padrões tradicionais é comum encontrarmos $P \ll I$, onde é necessário realizar o redimensionamento para um vetor. A Figura 4.1 representa como é feito este tipo de projeção.

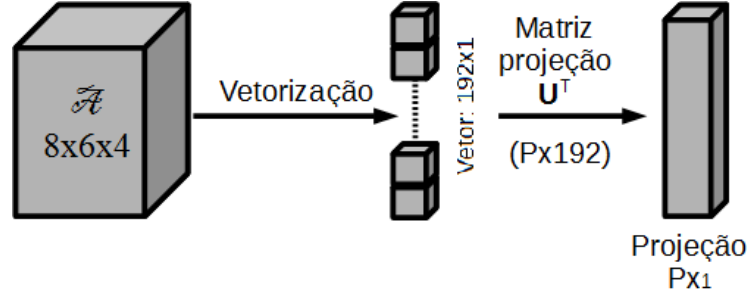


Figura 4.1: Representação de uma projeção vetor-vetor.

4.2.2 Projeção tensor-tensor

Entre as projeções que podem ser realizadas pode-se destacar a projeção tensor-tensor, que nos permite, através do produto de Kronecker, realizar projeções para tensores de mesma ou menor ordem. Um tensor $\mathcal{X}$, pertencente a um espaço tensorial $\mathbb{R}^{I_1} \otimes \mathbb{R}^{I_2} \dots \otimes \mathbb{R}^{I_N}$, projetado a outro tensor $\mathcal{Y}$, em um espaço tensorial $\mathbb{R}^{P_1} \otimes \mathbb{R}^{P_2} \dots \otimes \mathbb{R}^{P_N}$, onde $P_n \leq I_n$ para todo n , então N matrizes projetadas $\{\mathbf{U}^{(n)} \in \mathbb{R}^{I_n \times P_n}, n = 1, \dots, N\}$ são usadas, de modo que

$$\mathcal{Y} = \mathcal{X} \times_1 \mathbf{U}^{(1)T} \times_2 \mathbf{U}^{(2)T} \dots \times_N \mathbf{U}^{(N)T}. \quad (4.2)$$

A Figura 4.2 apresenta como um tensor $\mathcal{A}$ pode ser projetado a outro tensor de menor ordem.

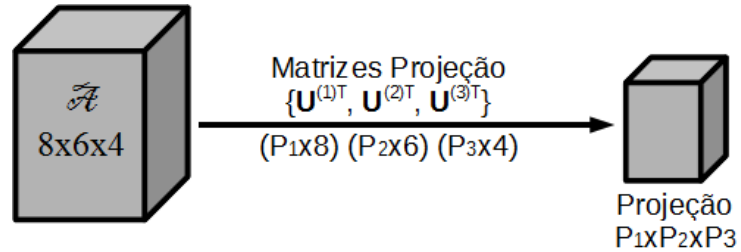


Figura 4.2: Representação de uma projeção tensor-tensor.

4.2.3 Projeção tensor-vetor

A projeção de um objeto no espaço tensorial para o espaço vetorial pode ser visto como multiplas projeções de um tensor para um escalar, como ilustrado na Figura 4.3, onde um tensor $\mathcal{A} \in \mathbb{R}^{8 \times 6 \times 4}$ foi projetado para um vetor $P \times 1$ de P projeções de $\mathcal{A}$ a um escalar.

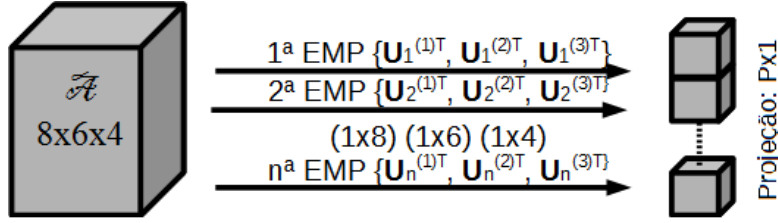


Figura 4.3: Representação de uma projeção tensor-vetor, sendo EMP projeções multilineares elementares.

A projeção multilinear $\{\mathbf{U}^{(1)\top}, \mathbf{U}^{(2)\top}, \dots, \mathbf{U}^{(N)\top}\}$ é chamada de EMP, sendo esta projeção única para cada modo.

4.3 MPCA

Dado um conjunto com M amostras $\{\mathcal{X}_1, \mathcal{X}_2, \dots, \mathcal{X}_M\}$ disponíveis para treinamento, cada objeto tensorial $\mathcal{X}_m \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$ assume um valor no espaço tensorial $\mathbb{R}^{I_1} \otimes \mathbb{R}^{I_2} \otimes \dots \otimes \mathbb{R}^{I_N}$, onde I_N é a dimensionalidade do tensor. O MPCA tem como objetivo encontrar uma projeção multilinear tensor-tensor que realize o mapeamento do espaço tensorial original $\mathbb{R}^{I_1} \otimes \mathbb{R}^{I_2} \otimes \dots \otimes \mathbb{R}^{I_N}$ em um subespaço $\mathbb{R}^{P_1} \otimes \mathbb{R}^{P_2} \otimes \dots \otimes \mathbb{R}^{P_N}$ com $P_n \leq I_n$ para $n = 1, \dots, N$:

$$\mathcal{Y}_m = \mathcal{X}_m \times_1 \tilde{U}^{(1)\top} \times_2 \tilde{U}^{(2)\top} \dots \times_N \tilde{U}^{(N)\top}, m = 1, \dots, M, \quad (4.3)$$

de tal modo que $\{\mathcal{Y}_m \in \mathbb{R}^{P_1} \otimes \mathbb{R}^{P_2} \otimes \dots \otimes \mathbb{R}^{P_N}, m = 1, \dots, M\}$ captura a maior variação do objeto tensorial original. A seguir, o Algoritmo 1 apresenta o algoritmo do MPCA de forma simplificada.

Algoritmo 1 – MPCA

ENTRADA: Conjunto de amostras de tensores $\{\mathcal{X}_m \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}, m = 1, \dots, M\}$ e a dimensão do subespaço tensorial desejado $\{P_n, n = 1, 2, \dots, N\}$.

SAÍDA: As N matrizes projetadas $\{\tilde{U}^{(n)} \in \mathbb{R}^{I_n \times P_n}, n = 1, 2, \dots, N\}$ que maximizem a matriz de dispersão total no espaço projetado.

1. **Pré-processamento:** *Centralizar as amostras de entrada.*
2. **Inicialização:** *Calcular a auto-decomposição.*
3. **Otimização local:** *Calcular projeção $\mathcal{Y}_m$ com $m = 1, \dots, M$.*
4. *Calcular $\Psi y_0 = \sum_{m=1}^M \|\tilde{\mathcal{Y}}\|_F^2$*
5. *Para $k = 1$ até K **Faça***
 6. *Para $n = 1$ até N **Faça***
 7. *└ Define matriz $\tilde{U}^{(n)}$ com maiores P_n autovalores.*
 8. *Calcular $\mathcal{Y}_m$ com $m = 1, \dots, M$ e Ψy_k .*
 9. ***Se** $(\Psi y_k - \Psi y_{k-1}) / \Psi y_{k-1} < \eta$, **Então***
 10. *└ └ Retorna $\{\tilde{U}^{(n)} \in \mathbb{R}^{I_n \times P_n}, n = 1, 2, \dots, N\}$*

No pré-processamento o tensor de entrada é centralizado através da subtração dos tensores médios de entrada $\{\tilde{X}_m = X_m - \bar{X}, m = 1, \dots, M\}$. Posteriormente as matrizes projeção são inicializadas e é feita a otimização local, onde Ψ_y representa o tensor de dispersão de $\mathcal{Y}_m$, através da atualização das matrizes projeção. Esta otimização é realizada até que o resultado convirja ou chegue ao número máximo de K iterações, sendo esta a forma mais comum de encerramento do algoritmo. A convergência se da por η , sendo este um valor muito pequeno definido pelo usuário, por exemplo $\eta = 10^{-6}$.

4.4 Extração de características

Um conjunto de auto-tensores são obtidos com a redução da dimensionalidade ($P_n \leq I_n$) determinados por um Q especificado pelo usuário e, cada entrada do tensor de característica projetado pode ser visto como uma característica (escalar) correspondente a um auto-tensor particular. Alguns ruídos são removidos nessa projeção.

Para o reconhecimento, o MPCA é uma técnica não supervisionada, não levando rótulos de classes em conta. Como resultado, a variação capturada no subespaço tensorial projetado inclui variações entre e intraclasses.

Na tarefa de classificação, uma grande variação entre classes indica boa separabilidade e, conseqüentemente, uma pequena variação entre classes indica uma difícil tarefa na separação das mesmas. A seleção de auto-tensores é feita de acordo com o poder de discriminação de suas classes, o qual é definido pela razão entre a dispersão entre classes e intraclases, dado por:

$$\Gamma_{p_1 p_2 \dots p_N} = \frac{\sum_{c=1}^C N_c [\hat{\tilde{\mathbf{Z}}}_c(p_1, p_2, \dots, p_N) - \hat{\tilde{\mathbf{Z}}}(p_1, p_2, \dots, p_N)]^2}{\sum_{m=1}^M [(\hat{\tilde{\mathbf{Z}}}_m(p_1, p_2, \dots, p_N) - (\hat{\tilde{\mathbf{Z}}}_{cm}(p_1, p_2, \dots, p_N))]^2}, \quad (4.4)$$

onde C é o número de classes, M é o número de amostras de treinamento, N_c é o número de amostras da classes c e c_m é a classe da m -ésima amostra da galeria $\hat{\mathcal{X}}_m$. Já o $\hat{\tilde{\mathbf{Z}}}_m$ é um tensor de características de $\hat{\mathcal{X}}_m$ em um subespaço tensorial projetado, e os tensores $\hat{\tilde{\mathbf{Z}}}$ e $\hat{\tilde{\mathbf{Z}}}_c$ são, respectivamente, os tensores médios de características e característica por classe.

Para seleção de auto-tensores, os dados da projeção de $\hat{\tilde{\mathbf{Z}}}_m$, do conjunto de treinamento, são rearranjados em um vetor de características $\mathbf{z}_m$ e ordenados de acordo com $\Gamma_{p_1, p_2 \dots p_N}$ em ordem decrescente e, só os primeiros H_y componentes mais discriminantes de $\mathbf{z}_m$ são mantidos para classificação, sendo H_y determinado empiricamente ou determinado manualmente. Com esse procedimento, determina-se quais são as características usadas, uma vez que é possível verificar quais são as que permitem boa separabilidade e as que não são tão interessantes.

Em um próximo passo, um tensor de peso $\mathcal{W}$ é formado, definido por $\mathcal{W}(p_1, p_2, \dots, p_N) = \sqrt{\Pi_{n=1}^N \lambda_{p_n}^{(n)}}$, onde $\lambda_{p_n}^{(n)}$ é o auto-valor do p_n -ésimo modo correspondente à matriz projeção $\tilde{\mathbf{U}}^{(n)}$ e Π_n são amostras do tensor. $\mathcal{W}$ é rearranjado em um vetor $\mathbf{w}$, da mesma ordem de $\mathbf{Z}_m$, e só os primeiros H_y componentes de $\mathbf{w}$ são usados como peso.

O vetor de características $\mathbf{z}_m$ pode ser usado diretamente para reconhecimento em um classificador, no entanto pode ser feita uma Análise Linear Discriminante (*Linear discriminant analysis* - LDA), obtendo-se a variação MPCA+LDA, similar ao popular PCA+LDA [18]. Na Figura 4.4 temos o diagrama do algoritmo MPCA+LDA.

O LDA tem como objetivo encontrar uma projeção $\mathbf{V}$ que maximize a razão entre as matrizes de dispersão entre classes $\mathbf{S}_B$ e intraclases $\mathbf{S}_W$, onde $\mathbf{S}_W = \sum_{m=1}^M (\mathbf{z}_m - \bar{\mathbf{z}}_{cm})(\mathbf{z}_m - \bar{\mathbf{z}}_{cm})^T$ e $\mathbf{S}_B = \sum_{c=1}^C N_c (\bar{\mathbf{z}}_c - \mathbf{z})(\bar{\mathbf{z}}_c - \mathbf{z})^T$. A solução $\mathbf{V}_{lda} = \arg \max \mathbf{v} \frac{\mathbf{V}^T \mathbf{S}_B \mathbf{V}}{\mathbf{V}^T \mathbf{S}_W \mathbf{V}} = [\mathbf{v}_1 \mathbf{v}_2 \dots \mathbf{v}_{H_z}]$, onde $\{\mathbf{v}_{hx}, h_z, \dots, H_z\}$ é

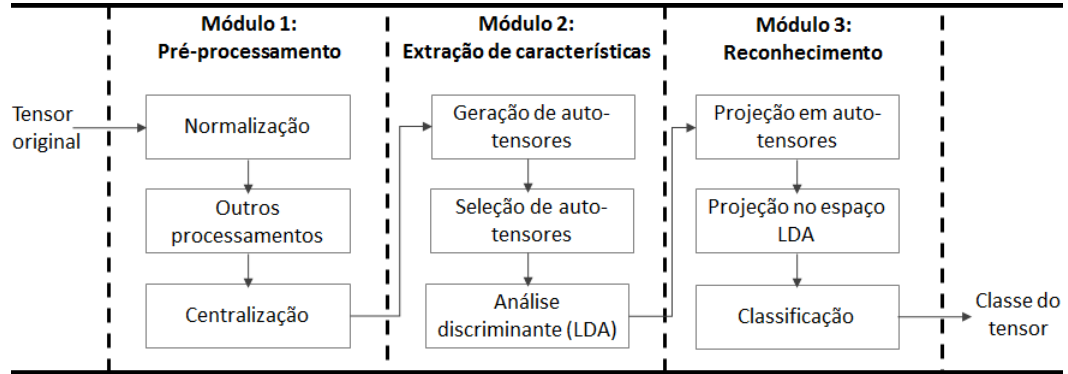


Figura 4.4: Diagrama para representação do algoritmo MPCA+LDA.

um conjunto generalizado de autovetores de $\mathbf{S}_B$ e $\mathbf{S}_W$ correspondentes aos $H_z(\leq C - 1)$ maiores autovetores generalizados $\{\lambda_{hz}, h_z = 1, \dots, H_z\}$. O vetor de características discriminante $\mathbf{z}_m$ é obtido pela equação

$$\mathbf{z}_m = \mathbf{V}_{lda}^T \mathbf{y}_m. \quad (4.5)$$

Como a Análise Linear Discriminante também é uma técnica de redução de dados, com sua aplicação é comum que se trabalhe com menor quantidade de características, visando continuar com as características que melhor discrimine cada classe.

Capítulo 5

Floresta de Caminhos Ótimos

A Floresta de Caminhos Ótimos, sendo um classificador baseado em grafos, visa modelar os dados como sendo vértices conectados por uma relação de adjacência. Neste capítulo é apresentada a fundamentação teórica e as versões com grafo completo e k -nn do OPF apresentadas por Papa et al. [20, 21, 22].

5.1 Fundamentação teórica

Seja D uma base de dados e D_1 e D_2 os conjuntos de treinamento e teste, respectivamente, com $|D_1|$ e $|D_2|$ instâncias, as quais podem ser pontos ou elementos de imagem, tais que $D = D_1 \cup D_2$. Seja $\lambda(s)$ uma função que associa o rótulo correto i , $i = 1, 2, \dots, c$ da classe i a qualquer instância $s \in D_1 \cup D_2$. Seja, também, $S \in D_1$ um conjunto de protótipos de todas as classes (isto é, instâncias importantes que melhor representam as classes). Seja $\vec{v}$ um algoritmo que extrai n atributos (cor, forma e propriedades de textura) de qualquer instância $s \in D_1 \cup D_2$, e retorna um vetor de atributos $\vec{v}(s) \in \mathbb{R}^n$. A distância $d(s, t)$ entre duas instâncias, s e t , é dada pela distância entre seus vetores de características $\vec{v}(s)$ e $\vec{v}(t)$. Nosso problema consiste em usar S , $(\vec{v}, d)$, D_1 e D_2 para projetar um classificador ótimo, o qual pode-se prever o rótulo correto $\lambda(s)$ de qualquer instância $s \in D_2$.

5.2 OPF com grafo completo

Seja (D_1, A) um grafo completo cujos nós são as instâncias em D_1 , onde qualquer par de instâncias define um arco em A (isto é, $A = D_1 \times D_1$) (Fi-

gura 5.1a). Note que os arcos não precisam ser armazenados e o grafo não precisa ser explicitamente representado.

Um caminho é uma sequência de instâncias $\pi = \langle s_1, s_2, \dots, s_k \rangle$, onde $(s_i, s_{i+1}) \in A$ para $1 \leq i \leq k-1$. Um caminho é dito ser trivial se $\pi = \langle s_1 \rangle$. Nós associamos a cada caminho π o custo dado por uma função de custos suave f [23], denotada $f(\pi)$. Dizemos que um caminho π é ótimo se $f(\pi) \leq f(\tau)$ para qualquer caminho τ , onde π e τ terminam na mesma instância s , independente de sua origem. Também denotamos $\pi \cdot \langle s, t \rangle$ a concatenação do caminho π com término em s e o arco (s, t) .

O algoritmo OPF pode ser utilizado com qualquer função de custo suave que pode agrupar instâncias com propriedades similares [23]. Entretanto, o OPF foi projetado usando a função de custo f_{max} , por causa de suas propriedades teóricas para estimar protótipos ótimos [24]:

$$\begin{aligned} f_{max}(\langle s \rangle) &= \begin{cases} 0 & \text{se } s \in S, \\ +\infty & \text{caso contrário} \end{cases} \\ f_{max}(\pi \cdot \langle s, t \rangle) &= \max\{f_{max}(\pi), d(s, t)\}, \end{aligned} \quad (5.1)$$

sendo que $f_{max}(\pi)$ computa a distância máxima entre instâncias adjacentes em π , quando π não é um caminho trivial.

O algoritmo OPF associa um caminho ótimo $P^*(s)$ de S a toda instância $s \in D_1$, formando uma floresta de caminhos ótimos P (uma função sem ciclos, a qual associa a todo $s \in D_1$ seu predecessor $P(s)$ em $P^*(s)$, ou uma marca *nil* quando $s \in S$). Seja $R(s) \in S$ a raiz de $P^*(s)$, a qual pode ser alcançada usando $P(s)$. O algoritmo OPF computa, para cada $s \in D_1$, o custo $V(s)$ de $P^*(s)$, o rótulo $L(s) = \lambda(R(s))$ e o seu predecessor $P(s)$, como implementado pelo Algoritmo 2.

As linhas 1–2 inicializam os mapas e inserem protótipos em Q . O laço principal calcula um caminho ótimo de S para cada instância $s \in D_1$ em uma ordem não decrescente de custos (linhas 3–10). A cada iteração um caminho de custo de ótimo $V(s)$ é obtido em P (linha 4). Empates são resolvidos em Q utilizando a política FIFO (first-in-first-out), ou seja, quando dois caminhos atingem uma determinada instância s com o mesmo custo mínimo, s é associado ao primeiro caminho que o atingiu. O restante das linhas avalia se o caminho que atinge uma instância adjacente t através de s é mais barato que o caminho que termina em t . Caso positivo, atualiza Q , $P(t)$, $L(t)$ e $V(t)$. No final do algoritmo, V armazena o valor do custo do caminho ótimo de S a cada instância $s \in D_1$ de acordo com f_{max} .

Algoritmo 2 – OPF

ENTRADA: Um conjunto de treinamento D_1 λ -rotulado, protótipos $S \subset D_1$ e o par (v, d) para vetor de características e cálculo das distâncias.
 SAÍDA: Floresta de caminhos ótimos P , mapa de valores de custo de caminhos V e mapa de rótulos L .
 AUXILIARES: Fila de prioridades Q , e variável cst .

1. **Para todo** $s \in D_1$, **Faça** $P(s) \leftarrow nil$ e $V(s) \leftarrow +\infty$.
2. **Para todo** $s \in S$, **Faça** $V(s) \leftarrow 0$, $P(s) \leftarrow nil$, $L(s) = \lambda(s)$ e *insira* s em Q .
3. **Enquanto** Q *é não vazia*, **Faça**
4. *Remova de* Q *uma instância* s *tal que* $V(s)$ *é mínimo*.
5. **Para cada** $t \in D_1$ *tal que* $s \neq t$ e $V(t) > V(s)$, **Faça**
6. *Calcule* $cst \leftarrow \max\{V(s), d(s, t)\}$.
7. **Se** $cst < V(t)$, **Então**
8. **Se** $V(t) \neq +\infty$, **Então** *remova* t *de* Q .
9. $P(t) \leftarrow s$, $L(t) \leftarrow L(s)$ e $V(t) \leftarrow cst$.
10. *Insira* t *em* Q .

O algoritmo OPF com grafo completo utilizando f_{max} pode ser entendido como uma transformada de Watershed usando a IFT [25] computada no espaço de características n -dimensional. Além dessa extensão, outras contribuições significativas foram os processos de treinamento e aprendizado, o qual encontra protótipos ótimos (marcadores) na fronteira entre as classes afastando *outliers* (instâncias de uma determinada classe que estão presentes em uma região de outra classe no espaço de atributos) do conjunto de treinamento, aumentando a acurácia do classificador.

5.2.1 Treinamento

Dizemos que S^* é um conjunto ótimo de protótipos quando o Algoritmo 2 propaga os rótulos $L(s) = \lambda(s)$ para todo $s \in D_1$. Desta forma, S^* pode ser encontrado explorando a relação teórica entre a Árvore de Espalhamento Mínima (*Minimum Spanning Tree* - MST) [24] e a árvore de caminhos mínimos para f_{max} . O treinamento consiste essencialmente em encontrar S^* e um classificador OPF enraizado em S^* .

Computando uma MST no grafo completo (D_1, A) , obtemos um grafo conexo acíclico cujos nós são todas as instâncias em D_1 , e os arcos são não direcionados e ponderados (Figura 5.1b). Seus pesos são dados pela distância

d entre os vetores de atributos de instâncias adjacentes. Esta árvore de espalhamento é ótima no sentido em que a soma dos pesos de seus arcos é mínima se comparada a outras árvores de espalhamento no grafo completo. Na MST, cada par de instâncias é conectado por um caminho, o qual é ótimo de acordo com f_{max} . Os protótipos ótimos são os elementos conectados na MST com diferentes rótulos em D_1 , isto é, elementos mais próximos de classes diferentes. Removendo-se os arcos entre classes diferentes, tais instâncias adjacentes tornam-se protótipos em S^* e o Algoritmo 2 pode computar uma floresta de caminhos ótimos sem erros de classificação em D_1 .

5.2.2 Classificação

Para qualquer instância $t \in D_2$, consideramos todos os arcos conectando t com instâncias $s \in D_1$, tornando t como se fosse parte do grafo (ver Figura 5.1c, onde a instância t é representada pelo losango no grafo). Considerando todos os possíveis caminhos entre S^* e t , desejamos encontrar o caminho ótimo $P^*(t)$ de S^* até t com a classe $\lambda(R(t))$ de seu protótipo mais fortemente conexo $R(t) \in S^*$. Este caminho pode ser identificado incrementalmente, avaliando o valor do custo ótimo $V(t)$ como

$$V(t) = \min\{\max\{V(s), d(s, t)\}\}, \forall s \in D_1. \quad (5.2)$$

Seja $s^* \in D_1$ o nó que satisfaz a equação acima (isto é, o predecessor $P(t)$ no caminho ótimo $P^*(t)$). Dado que $L(s^*) = \lambda(R(t))$, a classificação simplesmente associa $L(s^*)$ como a classe de t (Figura 5.1d). Um erro ocorre quando $L(s^*) \neq \lambda(t)$.

5.3 OPF com grafo k -nn

A técnica de classificação supervisionada baseada em floresta de caminhos ótimos apresentada nesta seção modela as instâncias novamente como sendo os nós de um grafo [20]. Entretanto, a relação de adjacência utilizada é a dos k -vizinhos mais próximos e tanto os arcos quanto os nós são ponderados. A diferença básica entre esta abordagem e a previamente apresentada (Seção 5.2), é o fato de a abordagem que faz uso do grafo completo estimar protótipos na fronteira das classes, enquanto a metodologia a ser apresentada aqui estima os protótipos nos pontos de alta concentração de instâncias.

A principal motivação desta representação seria o estudo de outras metodologias para classificadores supervisionados baseados em florestas de caminhos ótimos, utilizando outras funções de custo, relações de adjacência e maneiras diferentes de se encontrar os protótipos. As definições, bem como os algoritmos desta abordagem, serão apresentados nas próximas seções.

5.3.1 Fundamentação teórica

Dado um grafo (D_1, A_k) , o algoritmo OPF com grafo k -nn (OPFkNN) particiona o conjunto D_1 em uma floresta de caminhos ótimos de acordo com uma função de densidade de probabilidade (fdp). Nesta abordagem temos um grafo ponderado nos vértices e nos arcos. A distância $d(s, t)$ entre duas instâncias $s, t \in D_1$ novamente define o peso do arco (s, t) . Os nós são ponderados pelos seus valores de densidade de probabilidade, os quais são computados usando o peso dos arcos, como segue:

$$\rho(s) = \frac{1}{\sqrt{2\pi\sigma^2}|\mathcal{A}(s)|} \sum_{\forall t \in \mathcal{A}(s)} \exp\left(\frac{-d^2(s, t)}{2\sigma^2}\right), \quad (5.3)$$

onde $\sigma = \frac{d_f}{3}$ e d_f é o comprimento do maior arco em (Z, A_k) . A escolha deste parâmetro considera todos os nós para o cálculo da densidade, assumindo que uma função gaussiana cobre a grande maioria das instâncias com $d(s, t) \in [0, 3\sigma]$.

A função de valor de caminho dada pela Equação 5.4 associa a um nó terminal s de cada caminho o mínimo entre os valores de densidade ao longo do mesmo e um valor de densidade inicial.

$$\begin{aligned} f_2(\langle t \rangle) &= \begin{cases} \rho(t) & \text{se } t \in S \\ h(t) & \text{caso contrário} \end{cases} \\ f_2(\pi_s \cdot \langle s, t \rangle) &= \min\{f(\pi_s), \rho(t)\}. \end{aligned} \quad (5.4)$$

O caminho de valor máximo para cada instância s , a partir de um conjunto de elementos protótipos (raízes), particiona o conjunto de treinamento D_1 em uma floresta de caminhos ótimos. As raízes da floresta formam um subconjunto dos máximos da fdp onde, cada raiz, define uma árvore de caminhos ótimos (zona de influência do respectivo máximo) composta pelas suas instâncias mais fortemente conexas, as quais recebem o mesmo rótulo de sua raiz, como pode ser demonstrado pelo Algoritmo 3.

Algoritmo 3 – CLASSIFICADOR SUPERVISIONADO BASEADO EM FLORESTA DE CAMINHOS ÓTIMOS USANDO GRAFO k -NN

ENTRADA: Grafo k -nn (D_1, A_k) , $\lambda(s)$ para todo $s \in D_1$ e função de valor de caminho f_1 .

SAÍDA: Mapa de rótulos L , mapa de valores de caminho V e a floresta de caminhos ótimos P .

AUXILIARES: Fila de prioridade Q e a variável tmp .

1. **Para todo** $s \in D_1$, **Faça** $P(s) \leftarrow nil$, $V(s) \leftarrow \rho(s) - \delta$, $L(s) \leftarrow \lambda(s)$ e *insira* s em Q .
2. **Enquanto** Q *é não vazia*, **Faça**
3. *Remova de* Q *uma instância* s *tal que* $V(s)$ *é máximo*.
4. **Se** $P(s) = nil$, **Então**
5. $V(s) \leftarrow \rho(s)$.
6. **Para cada** $t \in A_k(s)$ e $V(t) < V(s)$, **Faça**
7. $tmp \leftarrow \min\{V(s), \rho(t)\}$.
8. **Se** $tmp > V(t)$, **Então**
9. $L(t) \leftarrow L(s)$, $P(t) \leftarrow s$, $V(t) \leftarrow tmp$.
10. *Atualize posição de* t *em* Q .

Inicialmente (Linha 1), todos os caminhos são triviais com valores $f_1(\langle t \rangle) = \rho(t) - \delta$. Os máximos globais da fdp são os primeiros elementos a serem removidos de Q , os quais são identificados como sendo as raízes da floresta pelo teste $P(s) = nil$ na linha 4, onde atualizamos o seu valor de caminho correto $f_1(\langle t \rangle) = V(s) = \rho(t)$. Cada no s removido de Q oferece um caminho $\pi_s \cdot \langle s, t \rangle$ para cada no t adjacente no laço da Linha 6 até a Linha 10. Se o valor de caminho $f_1(\pi_s \cdot \langle s, t \rangle) = \min\{V(s), \rho(t)\}$ (Linha 7) é melhor que o valor de caminho atual $f_1(\pi_t) = V(t)$ (Linha 8), então π_t é atualizado para $\pi_s \cdot \langle s, t \rangle$ (isto é, $P(s) \leftarrow s$, e o valor do caminho e o rótulo de t são atualizados (Linha 9). Máximo locais da fdp são também identificados como raízes durante a execução do algoritmo, o qual tem como saída o mapa de rótulos L , mapa de valores de caminho V e a floresta de caminhos ótimos P .

5.3.2 Treinamento

Um erro de classificação no conjunto de treinamento ocorre quando $L(t) \neq \lambda(t)$. Assim, definimos o melhor valor de $k^* \in [1, k_{max}]$ como sendo aquele que maximiza a acurácia da classificação no conjunto de treinamento.

É esperado que cada classe seja representada por pelo menos um máximo da fdp e $L(t) = \lambda(t)$ para todo $t \in D_1$. Entretanto, essas propriedades

não podem ser garantidas com a função de valor de caminho f_2 e o melhor valor k^* . Com o intuito de assegurar tais propriedades, primeiro encontramos k^* utilizando f_2 e então executamos o Algoritmo 3 mais uma vez usando a função de valor de caminho f_3 ao contrário de f_2 , dada por

$$\begin{aligned} f_3(\langle t \rangle) &= \begin{cases} \rho(t) & \text{se } t \in \mathcal{S} \\ \rho(t) - \delta & \text{caso contrário} \end{cases} \\ f_3(\pi_s \cdot \langle s, t \rangle) &= \begin{cases} -\infty & \text{se } \lambda(t) \neq \lambda(s) \\ \min\{f_2(\pi_s), \rho(t)\} & \text{caso contrário.} \end{cases} \end{aligned} \quad (5.5)$$

A Equação 5.5 pondera todos os arcos $(s, t) \in A_k$ tais que $\lambda(t) \neq \lambda(s)$ com $d(s, t) = -\infty$, impedindo que tais arcos pertençam a algum caminho ótimo, onde δ é um número real suficientemente pequeno. O processo de treinamento acima é descrito pelo Algoritmo 4.

Algoritmo 4 – TREINAMENTO

ENTRADA: Conjunto de treinamento D_1 , $\lambda(s)$ para todo $s \in D_1$, k_{max} e funções de valor de caminho f_2 e f_3 .
 SAÍDA: Mapa de rótulos L , mapa de valores de caminho V e a floresta de caminhos ótimos P .
 AUXILIARES: Variáveis i , k , k^* , $MaxAcc \leftarrow -\infty$, Acc e vetores FP e FN de tamanho c .

1. Para $k = 1$ até k_{max} **Faça**
2. Crie um grafo (D_1, A_k) ponderado nos nós através da Equação 5.3.
3. Calcule (L, V, P) utilizando o Algoritmo 3 com f_2 .
4. Para cada classe $i = 1, 2, \dots, c$, **Faça**
5. $FP(i) \leftarrow 0$ e $FN(i) \leftarrow 0$.
6. Para cada instância $t \in D_1$, **Faça**
7. Se $L(t) \neq \lambda(t)$, então
8. $FP(L(t)) \leftarrow FP(L(t)) + 1$.
9. $FN(\lambda(t)) \leftarrow FN(\lambda(t)) + 1$.
10. Calcule Acurácia.
11. Se $Acc > MaxAcc$, então
12. $k^* \leftarrow k$ e $MaxAcc \leftarrow Acc$.
13. Destrua o grafo (D_1, A_k) .
14. Crie o grafo (D_1, A_{k^*}) ponderados nos nós através da Equação 5.3.
15. Calcule (L, V, P) utilizando o Algoritmo 3 com f_3 .

5.3.3 Classificação

Uma instância $t \in D_2$ pode ser associada a uma dada classe $i, i = 1, 2, \dots, c$ simplesmente identificando qual raiz (protótipo) oferece o caminho ótimo como se esta instância pertencesse à floresta. Considerando os k -vizinhos mais próximos de t em D_2 , podemos utilizar a Equação 5.3 para computar $\rho(t)$, avaliar os caminhos ótimos $\pi_s \cdot \langle s, t \rangle$ e selecionar o que satisfaz a Equação 5.6:

$$V(t) = \max_{\forall (s,t) \in A_k^*} \{\min\{V(s), \rho(t)\}\}. \quad (5.6)$$

O rótulo de t será o mesmo rótulo do seu predecessor $P(t)$, ou seja, $L(t) = L(P(s))$. Um erro ocorre quando $L(t) \neq \lambda(t)$.

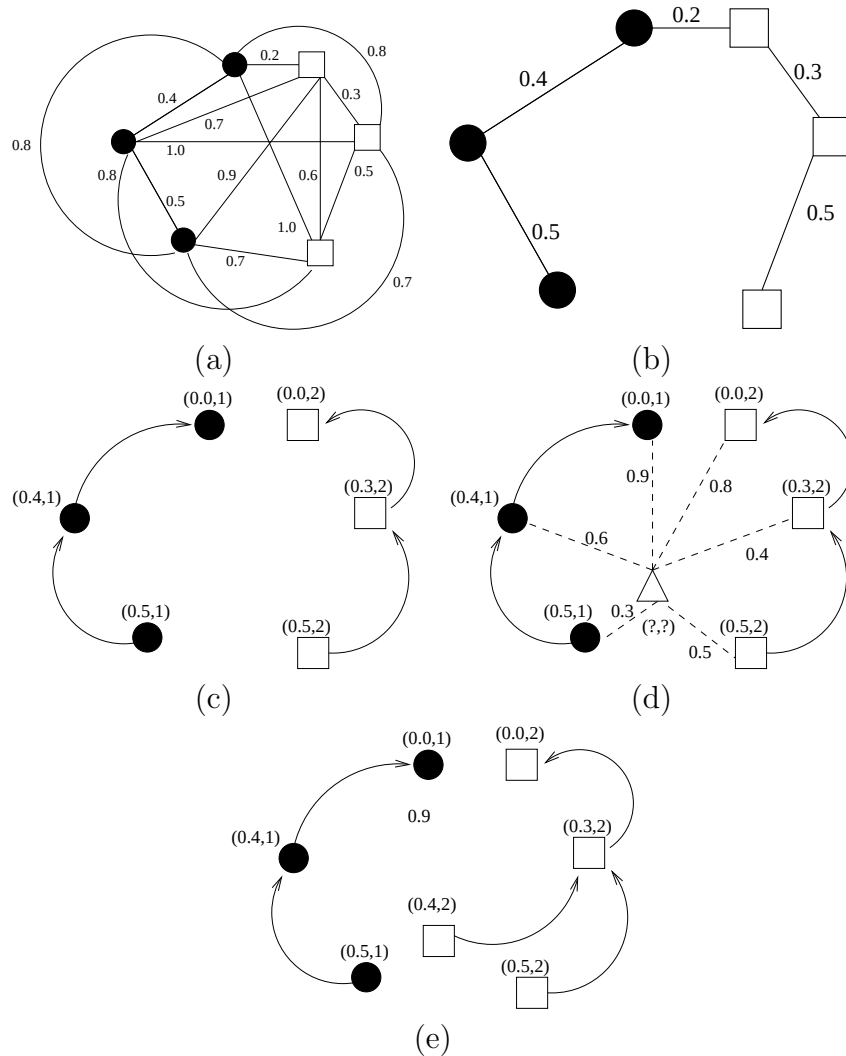


Figura 5.1: (a) Grafo completo ponderado nas arestas para um determinado conjunto de treinamento. (b) MST do grafo completo. (c) Floresta de caminhos ótimos resultante para f_{max} e dois protótipos (nós contornados). As entradas (x, y) acima dos nós denotam, respectivamente, o custo e o rótulo das instâncias. Os arcos direcionados indicam os nós predecessores no caminho ótimo. (d) Amostra de teste (triângulo) e suas conexões (linhas tracejadas) com os nós de treinamento. (e) O caminho ótimo do protótipo mais fortemente conexo, seu rótulo 2, e o custo de classificação 0.3 é associado à instância de teste.

Capítulo 6

Metodologia

A metodologia empregada pela presente dissertação de mestrado será apresentada neste capítulo. Inicialmente, foi realizado um estudo de técnicas para representação de informações vetoriais em espaços tensoriais e, posteriormente, a aplicação do classificador OPF nesse novo espaço. Basicamente, esta etapa consiste no estudo da aplicabilidade do classificador OPF com características extraídas em espaços tensoriais. A Figura 6.1 ilustra essa ideia.

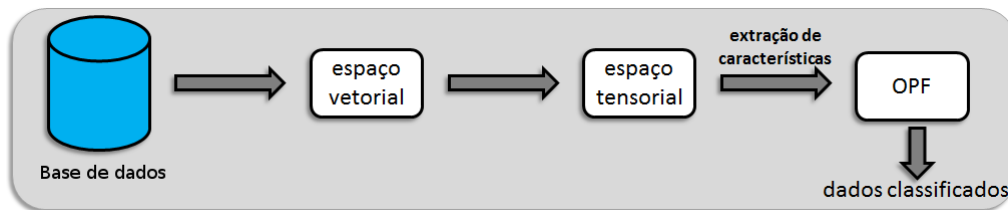


Figura 6.1: Metodologia proposta para aplicação de OPF em espaços tensoriais.

Em uma segunda etapa, foram utilizadas bases de dados que envolvem imagens em sua representação natural, tais como reconhecimento holístico de pessoas por meio de faces¹ e a identificação de pessoas através da silhueta de seu caminhar. Para tal, foram empregadas as seguintes bases de dados:

1. Gait-based Human ID²: esta base de dados consiste em vídeos de 71 sujeitos caminhando em direferentes situações. A base é composta por

¹O reconhecimento holístico de faces utiliza os tons de cinza da imagem como sendo a principal informação para sua representação.

²<http://figment.csee.usf.edu/GaitBaseline/>

sete cenários para classificação, com dificuldade crescente, intitulados “Prb” e para treinamento é utilizado um cenário chamado “Gallery” com 731 amostras. O número de amostras e diferenças nos conjuntos de teste são apresentados na Tabela 6.1.

Conjunto	Classes (sequências)	Diferença da galeria
Galeria	71(737)	-
PrbA	71(727)	Direção
PrbB	41(423)	Sapato
PrbC	41(420)	Direção, Sapato
PrbD	70(682)	Superfície
PrbE	44(435)	Sapato, Superfície
PrbF	70(685)	Visão, Superfície
PrbG	44(424)	Direção, Sapato, Superfície

Tabela 6.1: Conjuntos de amostras diferentes para classificação.

Para classificação, são utilizadas as silhuetas dos indivíduos, sendo estas redimensionadas para 32×22 *pixels* em sequências de 10 quadros. A extração das silhuetas se dá pelo algoritmo base apresentado por Sarkar et al. [17] e modificado por Lu et al. [16], onde é feita a redução da quantidade de quadros, redimensionamento e extração do fundo, uma vez que esta base consiste em vídeos em diversos cenários.

2. AT&T Face Dataset³: anteriormente “ORL Dataset”, compreende em imagens 92×112 de faces de 40 indivíduos.⁴

O cenário da primeira base citada é bastante interessante, pois corresponde à classificação de vídeos, onde cada um deles é representado como um tensor de ordem 3 (Figura 6.2b), sendo a terceira dimensão correspondente ao tempo. Ao invés de serem extraídas características de cada quadro, como geralmente é feito, foram utilizadas sequências de silhuetas em suas representações originais, evitando perder informações.

³<http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html>

⁴Embora esta base de dados não seja considerada nova, a principal idéia deste trabalho não é obter os melhores resultados em reconhecimento de faces, mas sim avaliar se representações baseadas em tensores podem ser mais precisas do que representações no espaço vetorial.

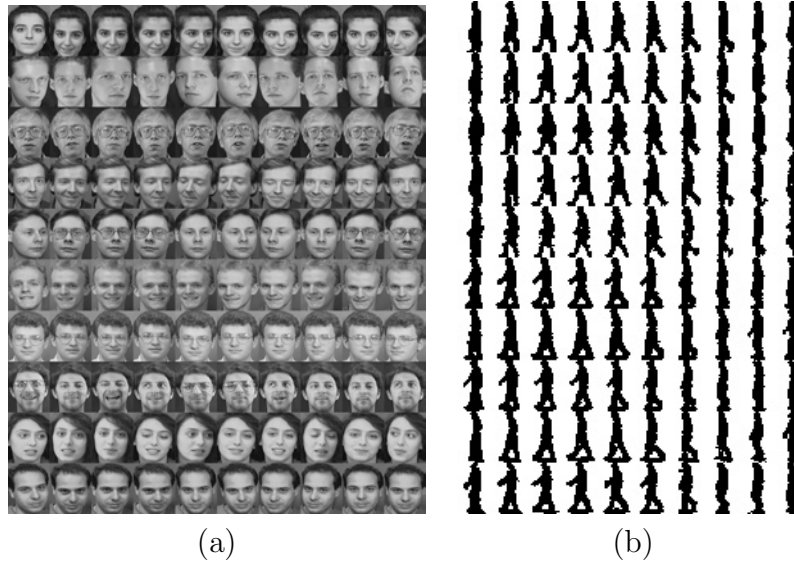


Figura 6.2: Exemplos das bases de dados (a) Gait-based Human ID e (b) AT&T Face datasets.

Um ponto negativo desta abordagem diz respeito à quantidade de informações a serem armazenadas e processadas, dado que o vídeo inteiro agora seria utilizado na etapa de reconhecimento e classificação, sendo inviável o processamento de vídeos longos ou de alta resolução sem o tratamento adequado.

Em outra etapa, foram realizados experimentos usualmente empregados em aplicações que fazem uso de técnicas de reconhecimento de padrões, ou seja, dada uma base de dados, a mesma é particionada em conjuntos de treinamento e teste gerados aleatoriamente para posterior emprego do classificador OPF. Para tal, foi realizado um método de validação cruzada por toda base de dados ($Gal \cup PrbA \cup PrbB \dots \cup PrbG$) para comparação de resultados, sendo o mesmo procedimento aplicado para “AT&T Face Dataset”.

Foram realizadas comparações do classificador OPF em dois cenários diferentes: TSM e VSM. Também foi avaliado o SVM no mesmo contexto, usando funções de base radial (*Radial Basis Function* - RBF), linear, polinomial, e *kernel* sigmóide, bem como o SVM sem mapeamento por *kernel*. Para avaliação do SVM com funções *kernel*, foi utilizada a reconhecida biblioteca LIBSVM [29], e para o SVM sem função de mapeamento por *kernel*

foi utilizado a biblioteca LIBLINEAR [28]. Já para o classificador OPF, foi empregada a biblioteca de código aberto LibOPF [21]. Em relação a extração de características, foi utilizado o *framework* MPCA [16] e, por se tratarem de bases de dados desbalanceadas, o cálculo da acurácia utilizada foi a proposta por Papa et al. [21].

Capítulo 7

Resultados e Análise

Neste capítulo serão apresentados os resultados obtidos a partir dos experimentos apresentados no Capítulo 6. Este capítulo está dividido em duas seções, sendo elas a classificação no contexto de faces e de vídeos, sendo esta última dividida entre experimentos em sete ambientes controlados e a validação cruzada entre todos eles. Finalmente, será feita uma análise dos resultados obtidos e uma breve discussão sobre alguns problemas encontrados.

As técnicas marcadas como “PCA” são utilizadas no modelo espaço vetorial, já as nomeadas com “MPCA” e “MPCA-LDA” denotam características modeladas no espaço tensorial, sendo esta segunda uma aproximação com a realização de Análise Linear Discriminante após o MPCA.

7.1 Faces

Foram empregados modelos espaço tensoriais e vetoriais considerando a tarefa de reconhecimento através de faces. Foram realizados dois experimentos distintos, sendo primeiro dele usando 10% da base de dados como conjunto de treinamento e o segundo usando 50% para treinar o classificador. Uma vez que foi feita a validação cruzada através da realização do experimento por 10 vezes, foi realizada a validação estatística através do teste de Wilcoxon [30], sendo apresentado os resultados na Tabela 7.1 as técnicas similares destacadas em **negrito** segundo o teste.

Os melhores resultados, obtidos com 10% de treinamento, foram apresentados pelo OPF-MPCA. No entanto, o OPF-PCA e algumas variações do

Tabela 7.1: Acurácias obtidas considerando o AT&T Face Dataset.

	10%	50%
OPF-PCA	83.97%±1.34	96.54%±0.74
OPF-PCA-knn	79.34%±9.91	96.59%±1.10
OPF-MPCA	84.52%±0.85	96.49%±0.94
OPF-MPCA-knn	77.28%±11.10	97.00%±0.73
OPF-MPCA-LDA	61.11%±1.52	77.54%±1.34
OPF-MPCA-LDA-knn	58.69%±4.85	76.74%±0.99
SVM-noKernel-PCA	83.93%±0.93	96.28%±0.75
SVM-Linear-PCA	83.97%±1.34	97.90%±0.55
SVM-Poly-PCA	59.95%±2.48	88.49%±1.27
SVM-RBF-PCA	83.97%±1.34	97.46%±1.16
SVM-Sig-PCA	83.62%±1.98	97.79%±0.76
SVM-noKernel-MPCA	83.46%±1.23	96.10%±0.86
SVM-Linear-MPCA	84.17%±0.97	97.85%±0.63
SVM-Poly-MPCA	65.61%±1.32	92.51%±1.20
SVM-RBF-MPCA	50.00%±0.00	94.49%±2.13
SVM-Sig-MPCA	51.14%±0.50	52.03%±0.92
SVM-noKernel-MPCA-LDA	57.95%±1.10	73.72%±1.26
SVM-Linear-MPCA-LDA	61.11%±1.52	79.00%±1.29
SVM-Poly-MPCA-LDA	53.02%±0.44	68.92%±1.66
SVM-RBF-MPCA-LDA	54.39%±1.05	77.10%±1.76
SVM-Sig-MPCA-LDA	51.13%±0.55	62.85%±0.95

SVM obtiveram resultados semelhantes. Considerando o experimento com 50% da base de dados, o SVM alcançou melhores resultados, sendo que as representações baseadas em tensores não tiveram grande importância nesse aspecto, embora também tenham obtido bons resultados.

Uma possível maneira de contornar tal problema seria a extração de características das imagens, e então mapear tais características em um modelo espaço-tensorial, uma vez que foi adotado o reconhecimento holístico para tal, como a intensidade dos *pixels* para propósito de reconhecimento.

7.2 Modo de andar

Nesta seção, serão apresentadas duas formas distintas de experimentos: (i) a validação-cruzada com amostras geradas aleatoriamente; e (ii) os dados dispostos em sete cenários diferentes, com dificuldade crescente e uma galeria pré-definida para treinamento.

7.2.1 Validação cruzada

Para estes experimentos, foram criados conjuntos de treinamento aleatoriamente, sendo eles com 10% e 50% de toda base de dados. A Tabela 7.2 apresenta os resultados considerando os experimentos descritos acima, onde ainda segundo o teste de Wilcoxon, temos as técnicas similares destacadas em negrito.

Considerando este experimento, pode ser observado que o OPF com características extraídas no espaço tensorial obteve melhores resultados do que o OPF tradicional quando foram utilizados 10% da base para propósito de treinamento. No entanto, quando foram utilizados 50% da base para treinamento, somente o OPF com o MPCA superou a tradicional modelagem espaço-vetorial (OPF-PCA), uma vez que o OPF-MPCA-LDA não conseguiu superar o OPF-PCA. Isso pode ter ocorrido devido ao mal mapeamento realizado pelo LDA quando é utilizado um grande conjunto de treinamento. Finalmente, o SVM também se beneficiou das características extraídas no espaço tensorial, alcançando também melhores resultados do que o OPF.

7.2.2 Galeria e provas

A Tabela 7.3 apresenta os resultados obtidos através do treinamento pela *Gallery* e a classificação dos sete conjuntos de teste, onde os conjuntos apresentam dificuldade crescente.

Neste experimento, podemos observar que modelos no espaço tensorial o melhores resultados nos classificadores OPF e SVM. Também podemos notar que o SVM obteve os melhores resultados em 5 de 7 experimentos, no entanto o OPF também se beneficiou do espaço tensorial, uma vez que obteve resultados próximos ao SVM e apresentou melhores resultados nos conjuntos onde se tinha maior dificuldade na classificação (amostras muito diferentes do conjunto de treinamento).

Tabela 7.2: Acurácias obtidas considerando o experimento com “dados aleatórios” para a base Gait-based Human ID dataset.

	10%	50%
OPF-PCA	60.86%±0.41	76.92%±0.50
OPF-PCA-knn	60.94%±0.42	77.49%±0.75
OPF-MPCA	67.77%±0.47	80.55%±0.45
OPF-MPCA-knn	67.93%±0.33	81.50%±0.31
OPF-MPCA-LDA	63.72%±0.47	73.15%±0.42
OPF-MPCA-LDA-knn	64.02%±0.51	73.43%±0.49
SVM-noKernel-PCA	55.68%±0.21	56.07%±0.26
SVM-Linear-PCA	57.79%±0.30	65.03%±0.50
SVM-Poly-PCA	55.27%±0.55	70.25%±0.44
SVM-RBF-PCA	59.06%±0.38	73.81%±0.48
SVM-Sig-PCA	57.62%±0.40	65.04%±0.58
SVM-noKernel-MPCA	72.16%±0.57	86.89%±0.42
SVM-Linear-MPCA	74.06%±0.66	91.42%±0.48
SVM-Poly-MPCA	61.07%±1.04	83.29%±0.39
SVM-RBF-MPCA	73.95%±0.61	91.41%±0.44
SVM-Sig-MPCA	74.01%±0.69	91.38%±0.43
SVM-noKernel-MPCA-LDA	66.55%±0.58	76.24%±0.41
SVM-Linear-MPCA-LDA	67.89%±0.54	79.98%±0.56
SVM-Poly-MPCA-LDA	53.77%±1.61	69.93%±0.83
SVM-RBF-MPCA-LDA	68.57%±0.57	81.68%±0.52
SVM-Sig-MPCA-LDA	67.95%±0.60	79.59%±0.68

Tabela 7.3: Taxa de reconhecimento considerando o experimento da galeria e provas para a base Gait-based Human ID dataset.

	PrbA	PrbB	PrbC	PrbD	PrbE	PrbF	PrbG
OPF-PCA	50.59%	49.59%	49.91%	50.59%	49.65%	50.36%	50.56%
OPF-PCA-knn	50.38%	49.59%	49.91%	50.59%	49.65%	50.30%	50.56%
OPF-MPCA	74.55%	73.73%	63.70%	59.50%	50.00%	54.66%	54.62%
OPF-MPCA-knn	74.70%	73.95%	64.09%	59.50%	50.00%	54.66%	54.62%
OPF-MPCA-LDA	86.39%	50.84%	70.74%	60.66%	49.82%	57.18%	56.67%
OPF-MPCA-LDA-knn	86.03%	50.84%	70.74%	60.58%	49.82%	57.18%	56.56%
SVM-noKernel-PCA	50,64%	49,66%	49,86%	50,55%	50,03%	50,38%	50,09%
SVM-Linear-PCA	51,32%	49,37%	49,93%	50,68%	49,66%	50,67%	51,26%
SVM-Poly-PCA	50,63%	49,52%	49,65%	50,16%	49,79%	50,85%	50,30%
SVM-RBF-PCA	51,32%	49,37%	49,93%	50,68%	49,66%	50,67%	51,27%
SVM-Sig-PCA	51,32%	49,37%	49,93%	50,68%	49,66%	50,67%	51,26%
SVM-noKernel-MPCA	86.22%	79.52%	70.81%	58.81%	50.27%	56.82%	54.51%
SVM-Linear-MPCA	87.20%	78.93%	70.81%	60.68%	50.10%	56.69%	55.86%
SVM-Poly-MPCA	70.28%	67.08%	59.92%	56.31%	49.61%	52.86%	52.61%
SVM-RBF-MPCA	87.06%	79.06%	70.81%	60.68%	50.10%	56.63%	55.87%
SVM-Sig-MPCA	87.20%	78.93%	70.81%	60.68%	50.10%	56.69%	55.86%
SVM-noKernel-MPCA-LDA	84.86%	51.29%	69.34%	59.63%	50.10%	56.17%	54.82%
SVM-Linear-MPCA-LDA	88.52%	50.92%	71.09%	61.58%	50.30%	56.64%	55.26%
SVM-Poly-MPCA-LDA	68.78%	49.91%	55.07%	55.32%	50.16%	53.30%	52.00%
SVM-RBF-MPCA-LDA	88.68%	51.03%	70.99%	61.73%	50.30%	56.69%	55.26%
SVM-Sig-MPCA-LDA	88.20%	50.90%	71.25%	61.51%	50.60%	56.64%	55.73%

Capítulo 8

Conclusões

Técnicas de aprendizado de máquina geralmente trabalham em um espaço de características cujas informações encontram-se representadas de maneira vetorial. Entretanto, existem diversas aplicações que poderiam ser melhor exploradas se utilizássemos os dados em sua representação original. Por meio da álgebra multilinear, modelos de representação baseados em tensores vêm sendo estudados nos últimos anos, com avanços consideráveis com relação à representação tradicional dos dados em sua forma vetorial. Ademais, a representação dos dados em espaços tensoriais permitem explorar espaços com curvatura, diferentemente do tradicional espaço Euclidiano, o qual apresenta curvatura nula.

8.1 Principais contribuições

O presente trabalho deixa como principal indicativo que técnicas de aprendizado de máquina baseadas em tensores ainda não são muito exploradas na literatura, muito embora existam trabalhos que datem de 5 ou 6 anos atrás. Um possível argumento para esta baixa demanda seria o forte referencial matemático necessário para o emprego de tais técnicas, as quais são mais utilizadas na Matemática e Física.

A presente dissertação de mestrado apresentou uma revisão bibliográfica sobre métodos de aprendizado de máquina baseados em tensores, bem como um referencial teórico sobre álgebra multilinear foi também discutido. Ademais, foi apresentada uma versão do tradicional classificador por Máquina de Vetores de Suporte adaptado a espaços tensoriais e Análise de Componen-

tes Principais Multilineares, a qual foi utilizado como base para extração de características e comparação de resultados.

O trabalho em questão visou o estudo da aplicabilidade do classificador OPF em dois contextos: (i) seu emprego direto em características extraídas em espaços tensoriais e em espaços vetoriais, (ii) comparação do classificador OPF com outras técnicas tradicionais na literatura, em espaços tensoriais e vetoriais, em bases de dados que pudessem ser modeladas com tensores de ordem 2 e 3. Nesse último caso, essa aplicação se deu pela aplicação do classificador OPF em contexto de vídeos.

Uma vez que a aplicação do classificador OPF nunca havia sido realizada no contexto de modelagem espaço-tensorial, podemos considerar que este classificador pode se beneficiar dessa representação baseada no espaço tensorial em determinados casos.

8.2 Trabalhos futuros

Como a aplicação de tensores na área de aprendizado de máquina é relativamente nova, ainda existem pontos e técnicas que podem ser beneficiados deste espaço. Neste capítulo serão apresentados dois caminhos que podem ser estudados e desenvolvidos futuramente.

8.2.1 Combinação de espaços vetoriais-tensoriais

O classificador OPF modela os dados como sendo vértices de um grafo, sendo cada vértice deste grafo portador de um vetor de características. Uma possível aplicação do espaço tensorial neste classificador seria a utilização de um mapeamento das características para o espaço tensorial, ou mesmo a substituição de entrada do classificador para alta dimensionalidade, e a utilização deste subespaço para aprendizagem/classificação ou sua combinação com o espaço vetorial.

8.2.2 Otimização evolucionista aplicada ao mapeamento vetorial-tensorial

No futuro, algoritmos podem ser desenvolvidos visando modelar o problema do mapeamento dos dados do espaço vetorial para o espaço tensorial como

sendo uma tarefa de otimização, onde podem ser empregadas técnicas meta-heurísticas para tal finalidade. Como ilustrado na Figura 2.1a, existem diferentes maneiras de mapear um vetor em um tensor, que vão desde a ordenação dos elementos do vetor em sua representação tensorial, como a escolha das dimensões desse novo tensor. Um elemento $\mathbf{x} \in \mathbb{R}^8$ qualquer, por exemplo, pode ser mapeado para um tensor $\hat{\mathcal{X}}$ em $\mathbb{R}^{2 \times 4}$, ou $\mathbb{R}^{4 \times 2}$ ou até mesmo qualquer outra combinação $\mathbb{R}^{n_1 \times n_2}$ tal que $n_1 \times n_2 \approx 8$.

Assim, uma das ideias consiste em utilizar um conjunto de validação com o intuito de encontrar os valores de n_1 e n_2 tal que, após o mapeamento de uma amostra $\mathbf{x} \in \mathbb{R}^n$ em um tensor $\hat{\mathcal{X}} \in \mathbb{R}^{n_1 \times n_2 \approx n}$, os mesmos maximizam a taxa de acerto do classificador nesse conjunto de validação. Para a situação acima descrita, onde temos o mapeamento de um vetor em um tensor de ordem 2, por exemplo, o problema consiste em uma tarefa de otimização de duas variáveis: n_1 e n_2 .

Muito embora qualquer técnica de otimização possa ser utilizada nesse contexto, objetivamos empregar técnicas por meta-heurísticas, tais como algoritmos baseados em inteligência evolucionista e coletiva, dado que o grupo de pesquisa o qual o discente participa vem trabalhando nessa área a um certo tempo. Ademais, não se tem muita notícia acerca de trabalhos que empregam otimização por meta-heurísticas em abordagens de aprendizado de máquina baseadas em tensores.

Referências Bibliográficas

- [1] D. Cai, X. He, and J. Han. Learning with tensor representation. Technical report, University of Illinois at Urbana-Champaign, Department of Computer Science, 2006.
- [2] H. Lu, K. N. Plataniotis, and A. N. Venetsanopoulos. A survey of multilinear subspace learning for tensor data. *Pattern Recognition*, 44(7):1540–1551, 2011.
- [3] M. A. O. Vasilescu, and D. Terzopoulos. Multilinear subspace analysis of image ensembles. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 93–9, 2003.
- [4] J. Ye, R. Janardan, and Q. Li. Two-dimensional linear discriminant analysis. In L.K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 1569–1576. MIT Press, 2005.
- [5] X. He, D. Cai, and P. Niyogi. Tensor subspace analysis. In Y. Weiss, B. Schölkopf, and J.C. Platt, editors, *Advances in Neural Information Processing Systems 18*, pages 499–506. MIT Press, 2006.
- [6] S. J. Wang, C.-G. Zhou, Na Zhang, X.-J. Peng, Y.-H. Chen, and X. Liu. Face recognition using second-order discriminant tensor subspace analysis. *Neurocomputing*, 74(12–13):2142 – 2156, 2011.
- [7] X. Gao and C. Tian. Multi-view face recognition based on tensor subspace analysis and view manifold modeling. *Neurocomputing*, 72(16–18):3742 – 3750, 2009.
- [8] I. Sutskever, J. B. Tenenbaum, and R. Salakhutdinov. Modelling relational data using bayesian clustered tensor factorization. In Y. Bengio,

- D. Schuurmans, J.D. Lafferty, C.K.I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 1821–1828. Curran Associates, Inc., 2009.
- [9] M. Ranzato, A. Krizhevsky, and G. E. Hinton. Factored 3-way restricted boltzmann machines for modeling natural images. In *Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *JMLR Proceedings*, pages 621–628. JMLR.org, 2010.
 - [10] D. Yu, L. Deng, and F. Seide. The deep tensor neural network with applications to large vocabulary speech recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, 21(2):388–396, 2013.
 - [11] R. Socher, A. Perelygin, J. Y. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, 2013.
 - [12] B. Hutchinson, L. Deng, and D. Yu. Tensor deep stacking networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1944–1957, 2013.
 - [13] D. Cai, X. He, J. Wen., J. Han, and W. Ma. Support tensor machines for text categorization. Technical report, University of Illinois at Urbana-Champaign, Department of Computer Science, 2006.
 - [14] Z. Zhang and T. W. S. Chow. Maximum margin multisurface support tensor machines with application to image classification and segmentation. *Expert Systems with Applications*, 39(1):849 – 860, 2012.
 - [15] I. Kotsia, W. Guo, and I. Patras. Higher rank support tensor machines for visual recognition. *Pattern Recognition*, 45(12):4192–4203, 2012.
 - [16] H. Lu, K. N. Plataniotis, and A. N. Venetsanopoulos. MPCA: Multilinear Principal Component Analysis of Tensor Objects, *IEEE Transactions on Neural Networks*, vol. 19, no. 1, pp. 18-39, Jan. 2008.
 - [17] S. Sarkar, P. J. Phillips, Z. Liu, I. Robledo, P. Grother, and K. W. Bowyer. The human ID gait challenge problem: Data sets, performance, and analysis, *IEEE Trans. Pattern Anal. Mach. Intell*, vol. 27, no. 2, pp. 162-177, Feb. 2005.

- [18] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection, *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 7, pp. 711-720, Feb. 1997.
- [19] C. Gao and X. J. Wu. Kernel support tensor regression. *Procedia Engineering*, 29(0):3986–3990, 2012. 2012 International Workshop on Information and Electronics Engineering.
- [20] J. P. Papa, and A. X. Falcão. A new variant of the optimum-path forest classifier. *International Symposium on Visual Computing*, 47(1):935–944, 2008.
- [21] J. P. Papa, A. X. Falcão, and C. T. N. Suzuki. Supervised pattern classification based on optimum-path forest. *International Journal of Imaging Systems and Technology*, 19:120–131, 2009.
- [22] J. P. Papa, A. X. Falcão, V. H. C. Albuquerque, and J. M. R. S. Tavares. Efficient supervised optimum-path forest classification for large datasets. *Pattern Recognition*, 45(1):512–520, 2012.
- [23] A. X. Falcão, J. Stolfi, and J. M. R. A. Lotufo. The image foresting transform: Theory, algorithms, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(1):19–29, 2004.
- [24] C. Allène, J. Y. Audibert, M. Couprie, J. Cousty, and R. Keriven. Some links between min-cuts, optimal spanning forests and watersheds. In *Proc. of the 8th International Symposium on Mathematical Morphology*, pages 253–264, 2007.
- [25] J. M. R. A. Lotufo, and A. X. Falcão. The ordered queue and the optimality of the watershed approaches. In *Proc. of the 8th International Symposium on Mathematical Morphology*, volume 18, pages 341–350. Kluwer, 2000.
- [26] L. D. Lathauwer, B. D. Moor, and J. Vandewalle. A multilinear singular value decomposition. *SIAM Journal on Matrix Analysis and Applications*, 21(4):1253–1278, 2000.
- [27] V. N. Vapnik. The Nature of Statistical Learning Theory. Springer-Verlag, 1995.

- [28] R.-E. Fan, K.-W. Chang, X.-R. Wang, and C.-J. Lin. LIBLINEAR: A library for large linear classification journal of machine learning research. vol. 9, pp. 1871-1874, 2008.
- [29] C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. ACM Transactions on Intelligent Systems and Technology, vol. 2, pp. 27:1-27:27, 2011, software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>.
- [30] F. Wilcoxon. Individual comparisons by ranking methods. Bulletin, vol. 1, no. 6, pp. 80-83, 1945.

Autorizo a reprodução xerográfica para fins de pesquisa.

São José do Rio Preto, 21 / 08 / 15

Ricardo Ricci Lopes
Assinatura