UNIVERSIDADE FEDERAL DE SÃO CARLOS

CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

AVANÇOS RECENTES EM CARACTERIZAÇÃO E CLASSIFICAÇÃO DE IMAGENS DE TEXTURAS: EXPLORANDO TEORIA DA INFORMAÇÃO, APRENDIZADO PROFUNDO E DE VARIEDADES

CÉDRICK BAMBA NSIMBA

ORIENTADOR: PROF. DR. ALEXANDRE LUIS MAGALHÃES LEVADA

São Carlos – SP

Abril/2020

UNIVERSIDADE FEDERAL DE SÃO CARLOS

CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

AVANÇOS RECENTES EM CARACTERIZAÇÃO E CLASSIFICAÇÃO DE IMAGENS DE TEXTURAS: EXPLORANDO TEORIA DA INFORMAÇÃO, APRENDIZADO PROFUNDO E DE VARIEDADES

CÉDRICK BAMBA NSIMBA

Tese apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de São Carlos, como parte dos requisitos para a obtenção do título de Doutor em Ciência da Computação, área de concentração: Processamento de imagens e sinais

Orientador: Prof. Dr. Alexandre Luis Magalhães Levada

São Carlos – SP

Abril/2020

UNIVERSIDADE FEDERAL DE SÃO CARLOS

Centro de Ciências Exatas e de Tecnologia
Programa de Pós-Graduação em Ciência da Computação

Folha de Aprovação

Defesa de Tese de Doutorado do candidato Cédrick Bamba Nsimba, realizada em 30/04/2020.

Comissão Julgadora:

Prof. Dr. Alexandre Luis Magalhães Levada (UFSCar)

Prof. Dr. Nelson Delfino D'Avila Mascarenhas (UFSCar)

Prof. Dr. José Hiroki Saito (UFSCar)

Prof. Dr. Marcelo Andrade da Costa Vieira (EESC/USP)

Prof. Dr. Adilson Gonzaga (USP)

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001.
O Relatório de Defesa assinado pelos membros da Comissão Julgadora encontra-se arquivado junto ao Programa de Pós-Graduação em Ciência da Computação.

Aos meus pais e a todos que acreditaram em mim.

AGRADECIMENTOS

Aos meus pais, irmãos, e a toda minha família que, com muito carinho e apoio, não mediram esforços para que eu chegasse até esta etapa de minha vida.

Ao professor e orientador Alexandre Luis Magalhães Levada pela paciência na orientação e incentivo que tornaram possível a conclusão desta tese de doutorado.

Aos meus amigos e colegas do GAPIS, pelo incentivo e apoio constantes.

À minha avó Ndona Teresa *in memoriam* pelo apoio total, desde a minha infância, e pelo provimento de recursos necessários para realizar a boa parte dos meus estudos por onde eu passei pelo mundo todo até agora.

À Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) pela concessão da bolsa de estudos nos primeiros três anos do curso de doutorado e da bolsa PDSE (Programa de Doutorado-sanduíche no Exterior) de um ano de duração para o estágio no Canadá.

La connaissance s'acquiert par l'expérience, tout le reste n'est que de l'information.

Albert Einstein

RESUMO

A tarefa de extração de características a partir de imagens é uma atividade de grande importância para várias aplicações de visão computacional e de processamento de imagens. Principalmente a caracterização e identificação de texturas é uma questão fundamental nessa área, e permite a promoção de desenvolvimento de uma gama ampla de aplicações interessantes que inclui análise de imagens médicas, recuperação de imagens por conteúdo, reconhecimento de objetos, entre outras. Devido a essa importância, muitas pesquisas vêm sendo realizadas nessa área e, consequentemente, resultaram no desenvolvimento de um conjunto de descritores de texturas. Contudo, em geral, algoritmos considerados como o atual estado da arte apresentam ainda problemas de performance quando aplicados na presença de ruído, uma vez que não se mantém propriedades intrínsecas da imagem sendo analisada. A partir do princípio de que, campos aleatórios e *wavelets* sejam ferramentas matemáticas apropriadas para auxiliar nesse processo, oportunidades se abrem para o desenvolvimento de aplicações bastante interessantes, com potencial de promover um alto nível de acerto na classificação de texturas. Enquanto campos aleatórios modelam bem as propriedades estatísticas das imagens de texturas, *wavelets* fornecem uma ferramenta robusta para decomposição e análise multiresolução das mesmas. Além disso, as medidas baseadas na teoria da informação, como por exemplo, a informação de Fisher e a entropia de Shannon de modelos de campos aleatórios Gaussianos Markovianos obtidas a partir das subbandas de uma árvore de decomposição *wavelet* são capazes de mensurar padrões de textura. Caso a dimensionalidade dos descritores de textura calculados seja alta, é possível utilizar métodos de redução de dimensionalidade não lineares baseados em técnicas de aprendizado de variedades. Nossa percepção é que técnicas de aprendizado de variedades são capazes de selecionar características discriminativas a partir de um espaço de alta dimensionalidade. O objetivo deste trabalho de pesquisa foi propor novas abordagens baseadas no uso de teoria da informação, aprendizado profundo e de variedades para caracterizar e classificar imagens de texturas. Para tal, três novas abordagens foram propostas: (1) concepção de descritor baseado nas matrizes de informação de Fisher de modelos de campos aleatórios Gaussianos Markovianos obtidas a partir das subbandas de uma árvore de decomposição *wavelet*; (2) concepção de descritor baseado na aplicação de métodos de aprendizado de variedades atuando como selecionador de informações de textura relevantes; e (3) concepção de descritor baseado na combinação de aprendizado profundo e de variedades. Resultados experimentais demonstraram que os métodos propostos são competitivos com descritores de estado da

arte relacionados.

Palavras-chave: Classificação de Textura, Campos Aleatórios Gaussianos Markovianos, Teoria da Informação, Informação de Fisher, Transformada *Wavelet*, Aprendizado Profundo e de Variedades

ABSTRACT

The task of extracting features from images is a very important activity for many computer vision and image processing applications. Especially the characterization and identification of textures is a fundamental issue in this area which allows the promotion of the development of a wide range of interesting applications including medical image analysis, content image retrieval, object recognition, among others. Due to this importance, many kinds of research have been carried out in this area and, consequently, resulted in the development of a set of texture descriptors. However, in general, algorithms considered as the state of the art still present performance problems when applied in the presence of noise, since it does not maintain intrinsic properties of the image being analyzed. Assuming that random fields and wavelets are appropriate mathematical tools to assist in this process, opportunities open for the development of very interesting applications, with the potential to promote a high level of texture classification accuracy. While random fields model better the statistical properties of texture images, wavelets provide a robust tool for decomposition and multi-resolution analysis. In addition, information theory-based measurements, such as Fisher information and Shannon entropy of Gaussian Markov random field models obtained from the sub-bands of a *wavelet* decomposition tree are able to measure texture patterns. If the dimensionality of the calculated texture descriptors is high, nonlinear dimensionality reduction methods based on manifold learning techniques can be used. Our perception is that manifold learning techniques are capable of selecting discriminative characteristics from a high dimensionality space. The aim of this research work was to propose new approaches based on the use of information theory, deep and manifold learning to characterize and classify texture images. To this end, three new approaches have been proposed: (1) descriptor design based on Fisher information matrices of Gaussian Markov random field models obtained from the sub-bands of a *wavelet* decomposition tree; (2) descriptor design based on the application of manifold learning methods acting as a selector of relevant texture information; and (3) descriptor design based on the combination of deep and manifold learning. Experimental results demonstrated that the proposed methods are competitive with related state of the art descriptors.

Keywords: Texture Classification, Gaussian Markov Random Field, Information Theory, Fisher Information, Wavelet Transform, Deep and Manifold Learning

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

Capítulo 1

INTRODUÇÃO

Textura é uma característica de qualquer superficie de imagens naturais ou daquelas produzidas artificialmente. A tarefa de extração de características a partir de imagens de textura é uma atividade de grande importância para várias aplicações de visão computacional e de processamento de imagens. Algoritmos considerados como o atual estado da arte são eficazes em caracterizar texturas, no entanto, em geral, apresentam problemas de performance quando aplicadas na presença de ruído, uma vez que não se mantém propriedades intrínsecas da imagem de textura sendo analisada. Este trabalho de pesquisa buscou contribuir com novas abordagens para extração de características a partir de imagens de texturas, procurando preservar tais propriedades intrínsecas a fim de garantir uma melhor performance para a posterior tarefa de classificação.

1.1 Contexto

A textura é uma propriedade onipresente em imagens naturais e em aquelas produzidas artificialmente. Ela constitui um importante elemento visual usado para uma variedade de aplicações nas áreas de processamento de imagens e de visão computacional. A análise de texturas em imagens, por sua vez, é uma fonte ideal para um conjunto diversificado de aplicações oriundas de tais áreas, como por exemplo o reconhecimento de objetos (KHAN; WEIJER; VANRELL, 2009; SANDE; GEVERS; SNOEK, 2010), de pisos (NILSBACK; ZISSERMAN, 2008; QI et al., 2014), de materials (LI; FRITZ, 2012; SHARAN et al., 2013) e de tecidos em imagens médicas (LERSKI et al., 1993; ONG et al., 1996), classificação de texturas (PIETIKäINEN; MäENPää; VIERTOLA, 2002), sensoriamento remoto (JIANG; RICH; BUHL-BROWN, 2015) e muito mais. No entanto, a caracterização e a identificação de texturas não são triviais, seu processo apresenta desafios bastante consideráveis relacionados, por exemplo, às condições nas quais elas foram gravadas. Consequentemente, alterações na geometria de

iluminação e intensidade ou no ponto de visão da câmera pode ter um impacto significativo sobre a aparência de uma imagem de textura (LEUNG; MALIK, 2001).

Parte dessa complexidade encontrada no processo de representação de texturas advém principalmente da falta, na literatura, de uma definição formal (matemática) que possa auxiliar na descrição completa do conceito de textura (MUSGRAVE et al., 1994; W. LAM N. N., 1999). Outra dificuldade nesse processo oriunda de diferentes tipos de texturas existentes, tais como microtextura, macrotextura e textura em que ambos os tipos se fazem presentes, o que acaba dificultando muito a definição de métodos robustos o suficiente para representar e classificar os tipos de informações de textura encontradas em imagens. Em geral, cada tipo de textura possui informações com características próprias, sendo adequadamente representada por um modelo estatístico específico.

Do ponto de vista computacional, a caracterização de texturas é um processo no qual um conjunto de técnicas de processamento de imagens é usado para caracterizar parâmetros de textura em determinadas imagens. Essas técnicas permitem extrair descritores de uma imagem, ou de uma região da mesma, relativos a características que remetem a propriedades intrínsecas como suavidade, rugosidade, regularidade, dentre outras. O método de análise de textura escolhido para extrair características é crucial para o sucesso da fase de classificação. Além disso, a métrica usada na comparação de vetores de características também é crucial. Existem várias abordagens para a extração de características de textura de uma imagem. Isso se deve ao fato de que a maioria das pesquisas na área de classificação de textura concentra-se na parte de extração de características (RANDEN; HUSØY, 1999), com extensas *surveys* e estudos comparativos (CONNERS; HARLOW, 1980), (HARALICK, 1979), (OHANIAN; DUBES, 1992), (REED; DUBUF, 1993), (TUCERYAN; JAIN,), (GOOL; DEWAELE; OOSTERLINCK, 1985), (WESZKA; DYER; ROSENFELD, 1976). Dentre esses métodos, pode-se citar alguns deles: *Gray Level Co-occurrence Matrix* (GLCM) (HARALICK, 1979; HARALICK; SHANMUGAM; DINSTEIN, 1973), *Markov Random Fields* (CROSS; JAIN, 1983), *Fractal Models* (KELLER; CHEN; CROWNOVER, 1989; MANDELBROT; FREEMAN; COMPANY, 1983), *Local Binary Patterns* (PIETIKäINEN et al., 2011; OJALA; PIETIKAINEN; MAENPAA, 2002), *Scale-Invariant Feature Transform* (LOWE, 2004), entre outros.

Em geral, todas essas abordagens focam em informações específicas na imagem. Além disso, a maioria delas escolhe um conjunto limitado de características de textura a partir da informação contextual na forma de *patches* extraídos da imagem local. No entanto, caracterizar texturas por meio de extração de características focando somente em informação local de texturas (*local image patches*) pode impactar negativamente para uma posterior classificação das

mesmas, por exemplo. Uma das simples razões desse fato é que textura é também caracterizada pela sua aparência global, o que representa a repetição e a relação entre padrões locais (LIU; FIEGUTH, 2012).

Outra dificuldade na extração de características de texturas está na alta dimensionalidade do espaço do conjunto de pontos de interesse sob análise. Diversas técnicas têm sido utilizadas nesta tarefa (redução/projeção para um subespaço de dimensão menor), porém mesmo com várias opções, não existe uma estratégia abrangente e, também, as técnicas desenvolvidas são sempre focadas em determinado propósito. Por isso, existem muitas pesquisas e incentivos para o desenvolvimento de novas abordagens abrangentes. Outro sério problema encontrado na classificação de imagens de textura é a presença de ruídos , que distorcem os dados observados. Frequentemente, ruídos são inerentes ao processo de aquisição de imagens reais. Na grande maioria dos casos, os métodos convencionais de classificação e alguns recentes apresentam ainda problemas de performance para classificação de imagens de texturas ruidosas.

Diante do exposto, a proposta desse trabalho consiste em combinar a teoria da informação (informação de Fisher e entropia de Shannon) de modelos de campos aleatórios Gaussianos Markovianos (GMRF) utilizados para modelar as subbandas de uma árvore de decomposição *wavelet*, com o intuito de superar as limitações encontradas nos métodos de classificação de imagens de texturas convencionais. A grande vantagem dos modelos GMRF é que eles permitem, em primeiro lugar, a captura das interações não lineares entre os coeficientes de uma subbanda *wavelet* e, em segundo lugar, a derivar as expressões de forma fechada exatas para duas quantias relevantes no contexto desse trabalho: (1) estimadores para o parâmetro (β) (o parâmetro que controla as interações entre variáveis vizinhas); e (2) matrizes de informação de Fisher esperada de cada subbanda *wavelet*.

1.2 Motivaçao

Durante os últimos anos a noção de informação tem se tornado cada vez mais presente e relevante para qualquer escala da sociedade moderna. Com o advento da tecnologia, o volume de dados produzido nunca foi tão grande, de modo que ser capaz de decodificar os símbolos presentes nesse vasto oceano de dados é um passo essencial para aprender, entender e analisar as regras que governam os mais variados fenômenos complexos que são parte da natureza. Alguns dos maiores desafios em lidar com esse cenário são justamente a mineração, a identificação, o processamento e a classificação de padrões e símbolos presentes nos dados, que são produzidos por uma vasta e heterogênea gama de fenômenos observáveis.

Uma das grandes limitações nesse processo de coleta de informação relevante a partir dos dados baseia-se na adoção da hipótese de que a estrutura dimensional de tais conjuntos pode ser bem representada por um espaço Euclidiano $\mathscr{R}^n$. Pesquisas cada vez mais tem mostrado que essa suposição é bastante fraca, no sentido de que na grande maioria dos casos, as métricas definidas no espaço Euclidiano ambiente (extrínsecas) não são adequadas para mensurar a similaridade entre instâncias dos conjuntos de dados, uma vez que a coleção de tais instâncias define uma variedade (*manifold*) imersa no espaço ambiente. Em outras palavras, a estrutura dimensional intrínseca costuma ser muito menor que a dimensão do espaço ambiente. Uma analogia simples que ilustra esse fato consiste em imaginar a seguinte situação hipotética: suponha que se queira medir distâncias entre pontos em uma bola de futebol. Sabe-se que pequenas distâncias nessa superfície são bem aproximadas por segmentos de reta, mas claramente, no caso geral, é necessário levar em conta a curvatura do espaço. Em termos práticos a melhor opção não seria utilizar uma régua de madeira para mensurar essas distâncias mas sim uma fita métrica flexível (a análise torna-se muito mais precisa se formos capazes de curvar/adaptar o instrumento de medição). Esse é o objetivo da área conhecida como aprendizado de variedades, ou *manifold learning*, que tem como foco principal a utilização de técnicas não paramétricas, pois não assume hipóteses acerca da distribuição dos dados observados.

Numa outra vertente, foca-se em modelos paramétricos para os dados, ou seja, assume-se que as observações são ocorrências de uma ou mais variáveis aleatórias e portanto seguem modelos estatísticos bem definidos, como por exemplo, a distribuição normal. Nesse cenário, cada modelo com seus parâmetros especificados define um ponto no espaço paramétrico. O objetivo então consiste em mensurar distâncias entre diferentes modelos nesses espaços paramétricos, com o intuito de quantificar a similaridade entre variáveis aleatórias, que nesse caso representam padrões não determinísticos. Pesquisas mostram que espaços paramétricos de modelos estatísticos definem variedades Riemannianas onde a métrica natural é dada pela matriz de informação de Fisher, uma importante medida da teoria da informação. Nesse contexto, esta tese de doutourado tem como objetivo principal estudar a evolução de sistemas complexos a partir de deformações na estrutura geométrica de seu espaço paramétrico.

Nesse sentido, esta pesquisa científica visa explorar essas duas áreas, teoria da informação e aprendizado de variedades, no desenvolvimento de ferramentas computacionais para modelagem e análise de padrões provenientes de imagens de texturas.

1.3 Hipótese de pesquisa

O objetivo desta seção consiste em estipular a principal hipótese que originou o tema escolhido desta tese de doutorado. Em outras palavras, pretende-se deixar explíticta qual é a pergunta motivadora deste trabalho, ou seja, quais questões que se desejam investigar com esta tese?

Portanto, a hípotese de pesquisa deste trabalho é *"A incorporação de medidas não Euclidianas na extração de características de imagens de textura, tanto de forma paramétrica, a partir da teoria da informação em modelos de campos aleatórios Markovianos, quanto de forma não paramétrica, a partir de métodos de aprendizado profundo (através de não-linearidades presentes nas redes) e de variedades (através da redução de dimensionalidade não linear), pode ser capaz de melhorar o reconhecimento de tais padrões?"*.

1.4 Objetivos e Justificativas

Tendo em vista os aspectos mencionados na seção anterior, este trabalho busca novas ferramentas, que levem tais aspectos em consideração, para caracterização de texturas visando a extração de informações relevantes e posterior reconhecimento de tais padrões.

De modo geral, esta tese teve como **objetivo geral**, propor novas abordagens para extração de características a partir de imagens de texturas, combinando os conceitos de teoria da informação, aprendizado profundo e de variedades, tendo como prioridade construir descritores de texturas que possam manter e/ou melhorar o poder discriminativo de classificadores de imagens de texturas em geral, e ruidosas, em particular.

Os **objetivos específicos** desta tese foram:

- Propor um novo descritor para imagens de texturas, baseado nas matrizes de informação de Fisher de modelos de campos aleatórios Gaussianos Markovianos obtidas a partir das subbandas de uma árvore de decomposição *wavelet*.
- Comparar o desempenho do descritor proposto frente a métodos clássicos e a abordagens baseadas em aprendizado profundo.
- Aplicar métodos de aprendizado de variedades para a redução de dimensionalidade não linear em problemas de classificação de texturas, verificando se tais algoritmos são mais eficazes do que métodos lineares como a Análise de Componentes Principais (PCA).

- Combinar aprendizado profundo e de variedades na classificação de imagens de textura.

1.5 Contribuições

As maiores contribuições deste projeto de pesquisa estão relacionadas com a incorporação de medidas não Euclidianas na caracterização espacial de padrões de texturas. De um modo geral, foram definidas as seguintes três contribuições:

- Desenvolvimento de um novo descritor de texturas baseado na informação de Fisher em modelos de campos aleatórios Gaussianos Markovianos;
- Combinação de aprendizado profundo e de variedades na extração de características de imagens de textura; e
- Combinação de descritores de texturas convencionais e de aprendizado de variedades na extração de características de imagens de textura.

1.6 Visão Geral

Basicamente, a metodologia proposta para caracterização e classificação de imagens de texturas segue as etapas ilustradas pelo diagrama de blocos da Figura 1.1. Dada uma imagem de textura de entrada, o objetivo é construir um descritor representativo com um alto poder discriminativo para classificá-la em uma das possíveis C classes previamente definidas (amostras pré-rotuladas), caracterizando o modelo de aprendizado supervidsionado. Em uma primeira abordagem Figura 1.1(A), uma combinação das medidas baseadas na teoria da informação, especificamente informação de Fisher e entropia de Shannon, de modelos GMRF aplicados no domínio *wavelet* foi adotada para representar a textura de entrada. O primeiro passo referente à primeira abordagem consiste em decompor a imagen de textura em n subbandas *wavelet* (Figura 1.1(A)(a)). Em seguida, para cada subbanda é gerado um *dataset* com *patches* de dimensão $n \times n$ (onde $n \times n$ é igual ao valor do maior inteiro positivo $\leq$ ordem-vizinhança $+1$) (Figura 1.1(A)(b)). A partir do *dataset* gerado, são calculados os componentes da matriz de informação de Fisher de cada subbanda (FIMatrixCn) e sua entropia de Shannon (ShEntropy) respectiva (Figura 1.1(A)(c)). A etapa final dessa abordagem consiste em criar uma versão normalizada a partir da concatenação das medidas de todas subbandas. Desse modo, o vetor resultante será o vetor de características que represente a imagem de textura sob análise.

Na segunda abordagem Figura 1.1(B), o conceito de redução de dimensionalidade é explorado para classificação de textura, utilizando técnicas de redução de dimensionalidade, tanto lineares (Análise de Componentes Principais (PCA) (DUNTEMAN, 1989)) quanto não lineares (técnicas de Aprendizado de Variedades (*Machine Learning*) (TENENBAUM; SILVA; LANGFORD, 2000; ROWEIS; SAUL, 2000; BELKIN; NIYOGI, 2002a)). A ideia consiste em criar uma abordagem para classificação de textura fortemente baseada na aplicação das técnicas de redução de dimensioalidade. Essa abordagem leva em consideração duas formas distintas de extrair características a partir de imagens de texturas de modo a construir um vetor de características de alta dimensionalidade para servir de entrada para os algoritmos de redução de dimensionalidade adotados no contexto deste trabalho.

De acordo com a Figura 1.1(B)(a), a primeira forma proposta para extração de características consiste em aplicar Histograma de Gradientes Orientados (HOG) (DALAL; TRIGGS, 2005), Matrizes de co-ocorrência (GLCM) versão 1 (em que todos os descritores inicialmente sugeridos por Haralick foram considerados), Matrizes de co-ocorrência versão 2 (em que somente alguns dos descritores inicialmente sugeridos por Haralick foram considerados) (HARALICK, 1979) e Padrões locais binários (LBP) (OJALA; PIETIKAINEN; HARWOOD,) e, em seguida, concatenar seus resultados em um vetor de características final. Figura 1.1(B)(b) mostra a segunda forma proposta na qual características são extraídas em forma de *patches* de 32×32 a partir da imagem de entrada. A concatenação dos atributos desses *patches* forma o vetor de características da imagem sob análise.

Os vetores de características resultantes de ambas as formas de extração propostas são entregues como entrada para quatro algoritmos de redução de dimensionaludade: PCA, *Locally Linear Embedding* (LLE) (ROWEIS; SAUL, 2000), *Isometric Feature Mapping* (ISOMAP) (TENENBAUM; SILVA; LANGFORD, 2000) e *Laplacian Eigenmaps* (Lap.Eig.) (BELKIN; NIYOGI, 2002a) para seleção de atributos e posterior tarefa de classificação utilizando os classificadores clássicos.

A última abordagem proposta neste trabalho Figura 1.1(C) consiste em combinar os conceitos de aprendizado profundo e de variedades para extrair informações relevantes a partir de imagens de texturas a fim de classificá-las. A ideia principal consiste em integrar as técnicas de aprendizado de variedades ao CNN como sendo ferramenta de seleção de atributos com o propósito de reduzir a dimensionalidade do vetor de características gerado pela penúltima camada de uma forma não linear. No contexto deste trabalho, utilizamos a penúltima camada da VGGNet (SIMONYAN; ZISSERMAN, 2014), a FC2, que possui 4096 neurônios, gerando um vetor de 4096 elementos. Como a dimensionalidade é alta, decidimos utilizar métodos de

redução de dimensionalidade não lineares. Nossa percepção é que técnicas de aprendizado de variedades são capazes de selecionar características discriminativas a partir de um espaço de pontos de interesse de alta dimensionalidade.

A implementação dos métodos propostos foi realizada utilizando o ambiente de desenvolvimento ANACONDA (baseado em PYTHON) (ANACONDA, 2019). Experimentos foram realizados em imagens de texturas reais obtidas a partir de diversas bases de dados públicas que servem de referência (*benchmark databases*) para o estudo de padrões de textura. A taxa de acerto ou acurácia global e o coeficiente de *Kappa* de Cohen (COHEN, 1960) foram as duas primeiras métricas aplicadas como forma de avaliação quantitativa nos primeiros experimentos. As duas métricas demonstraram-se adequadas para avaliar o desempenho da classificação, já que todos conjuntos de dados usados nesta tese são balanceados e também a amostragem para a validação cruzada é estratificada (SUN et al., 2007). Neste trabalho, todos os experimentos foram conduzidos utilizando a metodologia de validacao cruzada k-fold estratificadas (BARROW; CRONE, 2016).

Também foram incluídas nos experimentos sensibilidade, precisão e F1-score (SANTAFE; INZA; LOZANO, 2015) como métricas alternativas para validar o método proposto. Enquanto a sensitividade foi utilizada para medir a proporção de instâncias positivas que foram previstas pelo classificador, a precisão, por sua vez, mede a confiabilidade de uma determinada predição. F1-score foi adotado como sendo um compromisso entre a sensitividade e a precisão. Isso significa que ele representa ambas as medidas.

Experimentos foram realizados comparando as abordagens propostas aos seguintes métodos conhecidos na literatura para a classificação de imagens de texturas: HOG, GLCM versão 1, GLCM versão 2, LBP, *Texture CNN* (T-CNN) (ANDREARCZYK; WHELAN, 2016) e *wavelet convolutional neural networks* (*wavelet* CNNs) (FUJIEDA; TAKAYAMA; HACHISUKA, 2017).

1.7 Organização do Trabalho

O restante deste trabalho está organizado da seguinte maneira:

- No Capítulo 2 os principais descritores de estado da arte em classificação de textura relacionados ao presente trabalho de pesquisa são descritos em maiores detalhes.
- No capítulo 3, o referencial teórico necessário para a compreensão da problemática em relação à classificação de texturas, dos conceitos, algoritmos e métodos utilizados para

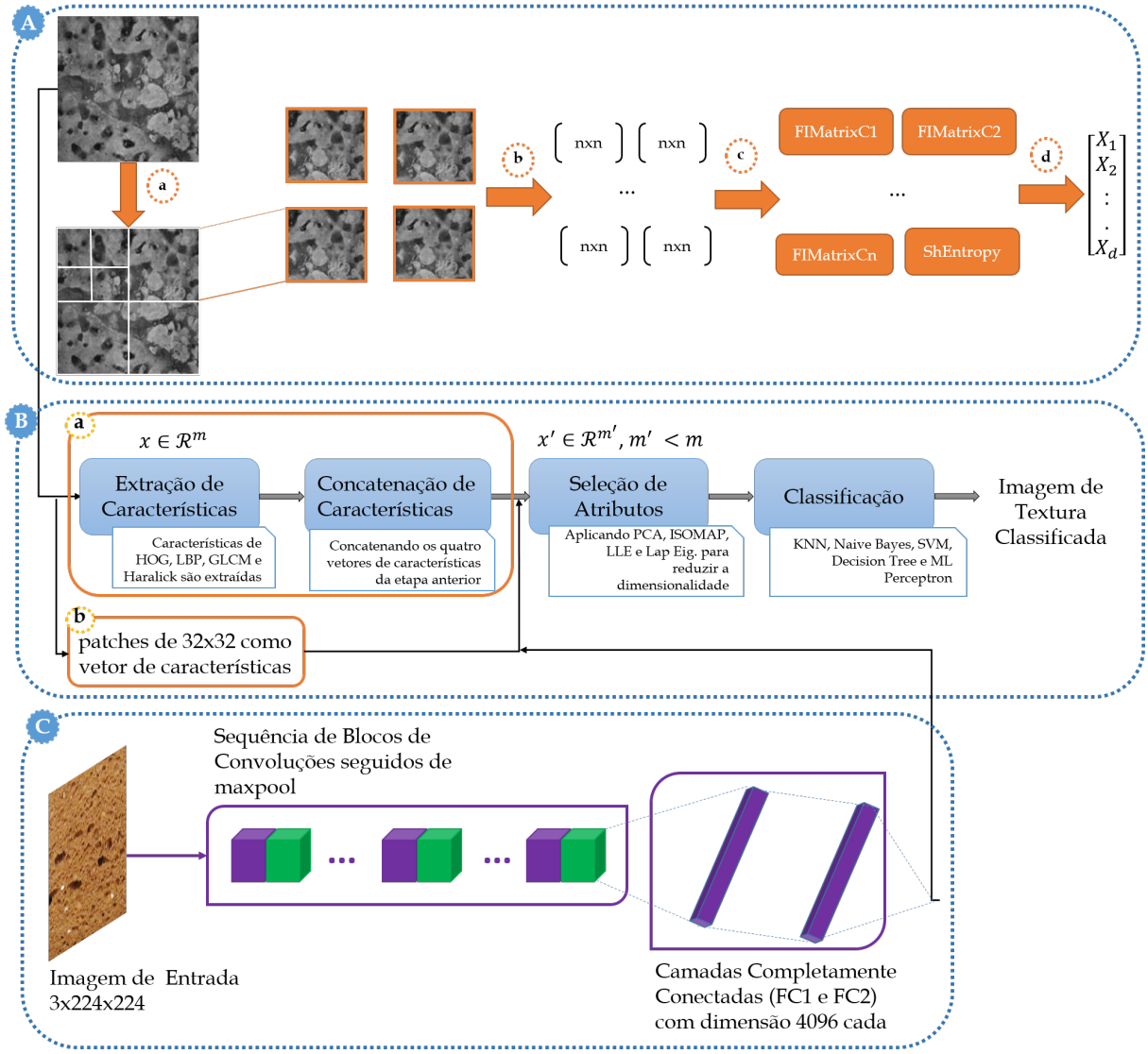


Figura 1.1: Diagrama de blocos do sistema proposto para classificação de imagens de texturas utilizando a combinação de medidas baseadas na teoria da informação de modelos GMRF no domínio *wavelet*, o aprendizado profundo e de variedades.

contorná-la é descrito. A seção 3.1 apresenta as considerações iniciais acerca do capítulo. A seção 3.2 introduz a noção de textura. A seção 3.3 descreve os tipos de descritores de textura utilizados na tarefa de classificação de imagens de texturas. A seção 3.4 apresenta fundamentos da transformada *wavelet*. Também é apresentado o processo da decomposição de uma imagem de textura usando a transformada *wavelet* discreta (DWT). A seção 3.5 introduz o modelo GMRF adotado para modelar os coeficientes da subbanda *wavelet*. As duas medidas, informação de Fisher e entropia de Shannon, baseadas na teoria da informação e utilizadas para a computação dos padrões de texturas são apresentadas na seção 3.6. Os algoritmos de redução de dimensionalidade tanto lineares como não lineares são apresentados nas seções 3.7 e 3.8, respectivamente. Finalmente, a seção 3.9

apresenta os modelos baseados no aprendizado profundo para a classificação de texturas;

- No Capítulo 4 são apresentadas as novas abordagens para a extração de características a partir de imagens de texturas propostas por este trabalho de pesquisa;
- No Capítulo 5 são apresentados os experimentos realizados com imagens de texturas reais e os respectivos resultados obtidos. Também são apresentadas as discussões sobre esta proposta de pesquisa; e
- No Capítulo 6 são apresentadas as conclusões finais sobre este trabalho de pesquisa. Em seguida, são apresentadas as propostas para trabalhos futuros. Finalmente, são apresentados alguns artigos científicos em conferências e revistas da área que o presente projeto de doutorado gerou.

Capítulo 2

REVISÃO DA LITERATURA

Esta seção apresenta o panorama atual da área de análise de texturas, com ênfase na extração de características a partir das imagens de textura. São apresentados alguns descritores de texturas existentes que podem ser considerados o estado da arte em análise de texturas.

2.1 Matrizes de co-ocorrência (GLCM)

No contexto de visão computacional, as estatísticas de primeira ordem referem-se aos momentos estatísticos computados a partir do histograma de uma imagem. O histograma de uma imagem, por sua vez, refere-se à medida de frequência com a qual os tons de cinza aparecem em uma imagem. Em outras palavras, um histograma deve ser visto como a distribuição probabilística dessas frequências. Assim, um momento estatístico seria uma medida ou informação básica da distribuição dos níveis de cinza. Como exemplos de estatísticas de primeira ordem destacam-se a entropia, variância, energia, assimetria e curtose.

O arranjo de pixels produzido por um histograma de níveis de cinza pode oferecer informações importantes para a descrição de texturas (GONZALEZ; WOODS, 2010), no entanto, a falta de levar em consideração a estrutura de dependência espacial dos pixels faz com o que o histograma apresente limitações na descrição de texturas mais complexas. Para contornar esse problema, utilizam-se métodos baseados nas estatísticas de segunda ordem, ou seja, estatísticas dadas por pares de pixels. O mais popular descritor de textura baseado em estatísticas de segunda ordem é a chamada matriz de co-ocorrência (GLCM) (HARALICK, 1979). Uma vez computada, essa matriz carrega todas as informações de dependência espacial entre pixels de uma imagem. De modo sucinto, uma matriz de co-ocorrência expressa a probabilidade $P(i, j \mid d, \theta)$ ou frequência relativa das vezes em que pares de pixels i e j estão entre si a uma distância d e ângulo θ. De

acordo com (HARALICK; SHANMUGAM; DINSTEIN, 1973), as relações entre tons de cinza são caracterizadas por uma função de distâncias e ângulos, e representam toda informação de texturas quando contidas nessas estruturas. Na Figura 2.1, é apresentado esse conceito.

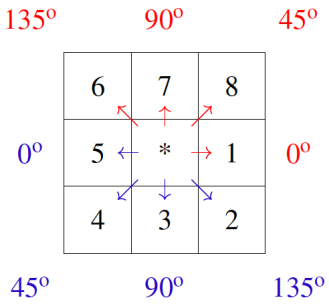


Figura 2.1: Relação espacial considerada para construção das GLCMs. Os pixels 1 e 5 estão a 0 graus (horizontal) do pixel central *; 2 e 6 estão a 135 graus; 3 e 7 a 90 graus; e 4 e 8 a 45 graus.

O processo da criação das matrizes de co-ocorrência está definida na Figura 2.2 em que a matriz de entrada Figura 2.2 (a) representa uma imagem de dimensão 5×5 preenchida com valores de intensidades variando de 0 a 3. Já as GLCMs de saída (Figuras 2.2 (b, c, d, e)) terão cada uma a dimensão 4×4 referente a quantidade de níveis de cinza da imagem correspondente/de entrada. O preenchimento dessas matrizes é feito respeitando a distância d e ângulo θ em que pares de pixels se encontram na matriz de entrada. Como pode ser observado na Figura 2.1, o ângulo considerado é sempre ou 0^0 ou 45^0 ou 90^0 ou 135^0, ou seja, varia a cada 45 graus. Desse modo, preenche-se nas coordenadas (i, j) e (j, i) de cada matriz o número de ocorrências de i com j e j com i.

Após computar as GLCMs, as informações contidas nelas precisam ser quantificadas com objetivo de caracterizar as texturas. Para isso, seis descritores foram propostos na literatura para esse propósito (GEBEJES; HUERTAS, 2013). Dentre eles destacam-se: Segundo Momento Angular (Energia), Contraste, Correlação, Momento da Diferença Inverso, Entropia e Dissimilaridade.

2.2 Padrões locais binários (LBP)

Local Binary Patterns (LBP) é um dos descritores mais populares no contexto de caracterização de texturas e foi inicialmente porposto pelos autores em (OJALA; PIETIKAINEN; HARWOOD,). A ideia chave por atrás do LBP basea-se no processo de binarização para rotular os pixels de uma imagem em tons de cinza.

O processo de binarização é aplicado sobre a vizinhança circular de cada pixel g_c (sendo

2	1	3	3	0
1	0	3	1	2
0	3	2	1	1
3	2	3	0	0
2	1	0	1	3

(a) Imagem 5x5 com 4 níveis de cinza.

	0	1	2	3
0	2	3	0	4
1	3	2	4	3
2	0	4	0	3
3	4	3	3	2

(b) $d = l$ e $\theta = 0$

	0	1	2	3
0	4	3	0	1
1	3	2	2	2
2	0	2	4	0
3	1	2	0	6

(c) $d = 1$ e $\theta = 45$

	0	1	2	3
0	0	5	1	4
1	5	2	3	1
2	1	3	0	4
3	4	1	4	2

(d) $d = 1$ e $\theta = 90$

	0	1	2	3
0	0	1	5	1
1	1	2	0	6
2	5	0	0	1
3	1	6	1	2

(e) $d = 1$ e $\theta = 135$

Figura 2.2: Matrizes de co-ocorrência (b, c, d, e) de (a) calculadas com distância 1 e ângulos 0, 45, 90 e 135 graus, respectivamente.

considerado um pixel central) em uma imagem $I(x,y)$ em níveis de cinza. A saída desse processo consiste em um conjunto de códigos binários que compõe um histograma capaz de representar a distribuição dos padrões de textura locais. A partir do pixel central g_c, seus P vizinhos estão a um raio de distância R e quanto maior o valor de R, maior será P, isto é, mais amostras da vizinhança terão de ser analisadas, como mostra a Figura 2.3. Esse conceito pode ser representado por

$$g_p = I(x_p, y_p), p = 0, 1, ..., P-1, \tag{2.1}$$

e

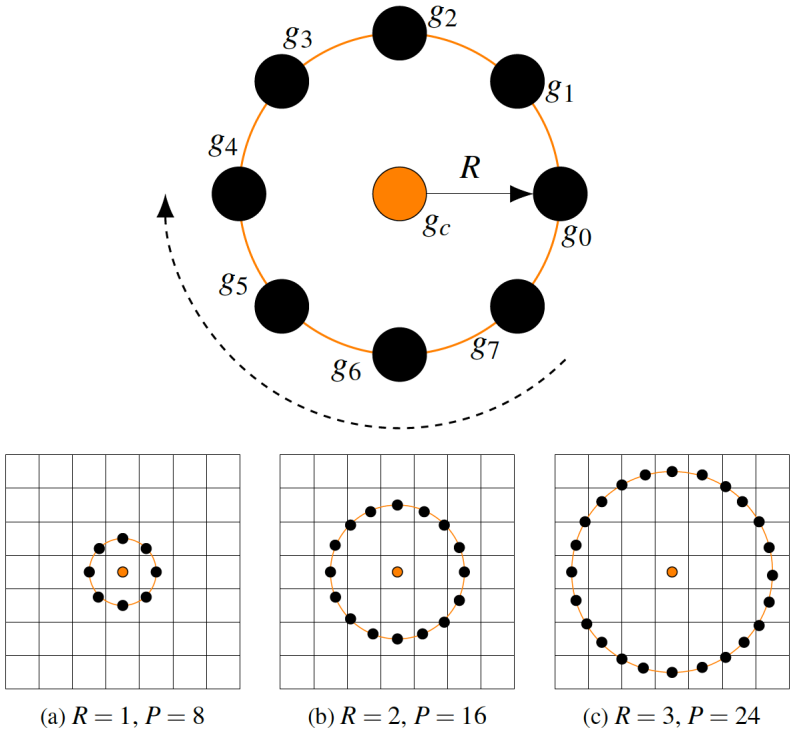
$$\begin{aligned} x_p &= x + Rcos(2\pi p/P), \\ y_p &= y - Rsin(2\pi p/P). \end{aligned} \tag{2.2}$$

Na equação acima, x_p e y_p representam as coordenadas de um determinado ponto na vizinhança de g_c que está a p distância dele.

O mecanismo de binarização por trás da técnica LBP consiste em rotular os vizinhos com

intensidade de cinza maior ou igual ao pixel central com 1 e os que tiverem intensidade de cinza menor ao pixel central com 0. Tais valores, quando apresentados sequencialmente em sentido horário, representam um número em binário que define a textura local.

Logo depois do surgimento da técnica LBP, houveram alguns variantes da técnica cobrindo alguns problemas notórios da mesma. A primeira extensão foi proposta pelos autores em (PIETIKÄINEN et al., 2011) tendo como objetivo resolver o problema relacionado aos pixels presentes nas extremidades da imagem, uma vez que suas respectivas vizinhanças não formam uma circunferência completa. Nesse contexto, a solução sugerida foi limitar as bordas com base no valor de R. Logo, os pixels de uma imagem $N \times M$ partiriam de R (em ambas linhas e colunas) chegando até os limites $N - R$ e $M - R$. Há outras extensões do LBP na literatura propostas por outros autores, uma delas foi sugerida por (LIU et al., 2012).



(a) $R = 1, P = 8$ (b) $R = 2, P = 16$ (c) $R = 3, P = 24$

Figura 2.3: Representação de um padrão de textura no método LBP. A figura superior mostra o pixel central gc rodeado por seus vizinhos gp que, neste caso, vão de 0 a 7. Uma seta tracejada mostra que após passarem pelo processo de binarização, devem ser interpretados no sentido horário. As imagens inferiores, por conseguinte, exemplificam a distribuição da vizinhança de acordo com os parâmetros *R* e *P*. É possível notar que *P* cresce em função de *R*.

2.3 Histograma de Gradientes Orientados (HOG)

O método de extração de características HOG foi introduzido em (DALAL; TRIGGS, 2005) como parte de um algoritmo de 5 passos para detecção de pedestres em imagens. No entanto, também podemos usar descritores HOG para quantificar e representar forma e textura.

Os 5 passos incluem:

Passo 1: Normalização da imagem: Essa etapa de normalização é totalmente opcional, mas em alguns casos, ela pode melhorar o desempenho do descritor HOG. Existem três métodos principais de normalização como descritos a seguir:

- **(a) Normalização pela lei de gama / potência:** Nesse caso, considera-se o $\log(p)$ de cada pixel p na imagem de entrada. No entanto, como os autores em (DALAL; TRIGGS, 2005) demonstraram, essa abordagem talvez seja uma "super correção", em termo inglês *"over-correction"*, e prejudica o desempenho.
- **(b) Normalização pela raiz quadrada:** Aqui, considera-se o $\sqrt{(p)}$ de cada pixel p na imagem de entrada. Por definição, a normalização pela raiz quadrada comprime as intensidades de pixel de entrada muito menos que a normalização pela lei de gama. E, novamente, como os autores em (DALAL; TRIGGS, 2005) demonstraram, a normalização pela raiz quadrada realmente aumenta a acurácia, em vez de prejudicá-la.
- **(c) Normalização pela variância:** Uma forma de normalização um pouco menos usada é a normalização pela variância. Aqui, calcula-se a média μ e o desvio padrão σ da imagem de entrada. Todos os pixels centralizados, subtraindo a média da intensidade do pixel, e então normalizados através da divisão pelo desvio padrão: $p' = (p-\mu)/\sigma$. Os autores em (DALAL; TRIGGS, 2005) não relatam sobre acurácia na normalização pela variância; no entanto, é uma forma de normalização que vale a pena testar.

Passo 2: Cálculo da orientação e magnitude das arestas na imagem: Aqui, calcula-se o gradiente da imagem nas direções x e y. Em seguida, aplica-se uma operação de convolução para obter os gradientes da imagen de entrada usando a seguinte expressão

$$G_x = I * D_x \;\; e \;\; G_y = I * D_y$$

onde I é a imagem de entrada, D_x é o filtro na direção x, e D_y é o filtro na direção y. Em seguida, calcula-se a representação final da magnitude do gradiente da imagem como

$$|G| = \sqrt{G_x^2 + G_y^2}.$$

Por fim, a orientação do gradiente para cada pixel na imagem de entrada pode ser calculada por:

$$\theta = \arctan 2(G_y, G_x).$$

Dado ambos $|G|$ e θ, agora pode-se calcular um histograma de gradientes orientados, em que o *bin* do histograma é baseado em θ e a contribuição ou peso adicionado a um determinado *bin* do histograma é baseado em $|G|$.

Passo 3: Divisão da imagem em células e blocos: Nesta fase, serão definidas duas estruturas, designadas por células e blocos. Estas células são janelas da imagem original, com dimensões de $c \times c$ pixels. Um bloco é visto como uma junção de $n \times n$ células, gerando um bloco com um total de $n \times n \times c \times c$ pixels. É necessário que exista uma sobreposição de algumas células de bloco para bloco tanto na horizontal como na vertical.

Passo 4: Cálculo do histograma de orientação dos gradientes por células, posteriormente agrupados em blocos: Nesta etapa, para cada uma das células da imagem, é preciso construir um histograma de gradientes orientados usando a magnitude do gradiente $|G|$ e orientação θ mencionadas acima.

Passo 5: Concatenação dos histogramas do Passo 4, formando assim o vetor descritor HOG: Finalmente, nesta fase, utilizando os blocos, é aplicada uma equalização por bloco, tornando o descritor menos invariante à iluminação e a sombras. Cada bloco agora pode ser representado por um histograma da orientação do gradiente de cada célula. Finalmente, depois que todos os blocos são normalizados, os histogramas resultantes são concatenados para formar o vetor de características final.

2.4 Descritores baseados nas Redes Neurais Convolucionais (CNNs)

Um grande número de métodos de representação de textura baseados em CNN foram propostos nos últimos anos desde o recorde, resultante da acurácia na classificação de imagens (KRIZHEVSKY; SUTSKEVER; HINTON, 2012a), alcançado em 2012. Uma chave para o sucesso das CNNs é sua capacidade de aprender características de alta qualidade a partir de grandes volumes de dados rotulados. No entanto, treinar uma CNN, equivale a estimar milhões de parâmetros e requer um número muito grande de imagens rotuladas, uma questão que limita a aplicabilidade de CNNs em problemas com dados de treinamento limitados. Uma descoberta importante, a esse respeito, foi que as características extraídas a partir de modelos CNNs pré-treinados em conjuntos de dados muito grandes revelaram ser capazes de transferir bem para muitos outros problemas, incluindo análise de textura, com um mínimo de esforço de adaptação (CIMPOI et al., 2016).

Nesta pesquisa, os métodos de representação de textura baseados nas CNNs serão classificados em três categorias:

(1) usando modelos CNN genéricos pré-treinados: modelos CNN pré-treinados são modelos treinados durante semanas na base imageNet usando uma plataforma computacional robusta equipada de múltiplos GPUs, por exemplo. Uma vez treinados, esses modelos são disponibilizados para o benefício da comunidade para diversos fins, por exemplo *finetuning*. Dentre essa categoria de modelos, destam-se AlexNet (KRIZHEVSKY; SUTSKEVER; HINTON, 2012b), VGGNet (SIMONYAN; ZISSERMAN, 2014), GoogleNet (SZEGEDY et al., 2015), ResNet (HE et al., 2016) e DenseNet (HUANG; LIU; WEINBERGER, 2016). A abordagem mais adotada aqui para a tarefa de classificação de textura consiste em extrair os descritores de textura a partir de uma das camadas totalmente conectadas da rede, por exemplo, utilizam-se os descritores ou de $FC6$ ou de $FC7$ em AlexNet ou de $FC1$ ou de $FC2$ em VGGNet. As camadas totalmente conectadas são pré-treinadas discriminativamente e, pode ser uma vantagem ou uma desvantagem, dependendo se as informações que elas capturaram podem ser transferidas para o domínio de interesse. Os descritores das camadas totalmente conectadas possuem um campo receptivo global e são geralmente vistos como características globais adequadas para classificação com um classificador SVM. Portanto, o descritor extraído de uma das camadas FCs serve de entrada para SVM para classificar a textura de entrada.

(2) usando modelos CNN com ajuste fino (*finetuned*): Ao realizar o *finetuning* de um mo-

delo CNN, todas camadas do modelo pretreinado são mantidas exceto a última camada totalmente conectada que é modificada para ter o mesmo número de classes correspondentes ao número de classes no conjunto de dados atual. A natureza do conjuntos de dados usados no *finetuning* é de extrema importante para aprender características discriminativas. O modelo CNN pré-treinado é capaz de discriminar imagens de diferentes objetos ou classes de cena, mas pode ser menos eficaz em discernir a diferença entre diferentes texturas (tipos de material), pois uma imagem no ImageNet pode conter diferentes tipos de texturas (materiais). O tamanho do conjunto de dados usado no *finetuning* também é importante, já que um conjunto de dados muito pequeno pode ser inadequado para a aprendizagem completa. Como exemplos de alguns modelos *finetuned* para a classificação de imagens de textura, destacam-se: Fisher Vectors CNN (FVCNN)(CIMPOI et al., 2016) e Texture CNN (TCNN) (ANDREARCZYK; WHELAN, 2016).

(3) usando redes convolucionais profundas treinadas do zero (*handcrafted CNNs*): Semelhante à arquitetura da CNN, os autores em (BRUNA; MALLAT, 2013) propuseram ScatNet. A principal diferença em relação à CNN comum, em que filtros convolucionais são aprendidos a partir dos dados, os filtros convolucionais no ScatNet são predeterminados - são simplesmente filtros wavelet, como wavelets de Gabor ou de Haar, e nenhum aprendizado é necessário. Além disso, o ScatNet geralmente não pode ser tão profundo como uma CNN tradicional; os autores em (BRUNA; MALLAT, 2013) sugeriram duas camadas convolucionais, já que a energia da terceira camada responsável pela dispersão dos coeficientes é insignificante. Outro modelo de destaque nessa categoria é PCANet, proposto em (CHAN et al., 2015), uma rede convolucional muito simples baseada em filtros PCA treinados, em vez de filtros de Gabor predefinidos. Comparado com o ScatNet, a extração de caracteristicas no PCANet é muito mais rápida, mas com pouco desempenho na tarefa de classificação de textura.

Capítulo 3

FUNDAMENTAÇÃO TEÓRICA

O objetivo deste capítulo é fornecer uma fundamentação teórica dos métodos e algoritmos utilizados no trabalho. O conteúdo apresentado neste capítulo visa auxiliar o entendimento da proposta deste trabalho de doutorado e descrever o tipo de ferramental utilizado na concepção da solução proposta.

3.1 Considerações Iniciais

O propósito deste capítulo é fornecer um referencial teórico amplo e necessário para a compreensão da problemática em relação à classificação de texturas e também uma descrição detalhada sobre os conceitos, algoritmos e métodos utilizados para contorná-la. A seção 3.2 introduz a noção de textura. A seção 3.3 descreve os tipos de descritores de textura utilizados na tarefa de classificação de imagens de texturas. A seção 3.4 apresenta fundamentos da transformada *wavelet*. Também é apresentado o processo da decomposição de uma imagem de textura usando a transformada *wavelet* discreta (DWT). A seção 3.5 introduz o modelo GMRF adotado para modelar os coeficientes da subbanda *wavelet*. As duas medidas, informação de Fisher e entropia de Shannon, baseadas na teoria da informação e utilizadas para a computação dos padrões de texturas são apresentadas na seção 3.6. Os algoritmos de redução de dimensionalidade tanto lineares como não lineares são apresentados nas seções 3.7 e 3.8, respectivamente. Finalmente, a seção 3.9 apresenta os modelos baseados no aprendizado profundo para a classificação de texturas.

3.2 O que é Textura?

A textura é uma propriedade intrínseca de uma superfície qualquer presente na natureza ou elaborada de forma artificial. Ela constitui um importante elemento visual usado em várias aplicações nas áreas de processamento de imagens e de visão computacional.

Texturas em imagens são as diferenças locais em níveis de intensidade. De modo geral, texturas são padrões visuais complexos compostos de entidades, ou subpadrões, que possuem algumas características como brilho, cor, tamanho, etc (Figura 3.1). Consequentemente, a distribuição das micro-estruturas, também chamadas de textons, em imagens constituem elementos chaves por caracterizar a textura. Algumas propriedades interessantes como a percepção luminosa, uniformidade, densidade, rugosidade, regularidade, linearidade, direcionalidade, frequência, aspereza, aleatoriedade, suavidade e granulação são criadas a partir da combinação/inter-relação dessas micro-estruturas (MATERKA; STRZELECKI, 1998). Vale lembrar que, no contexto computacional, padrões periódicos e ruidosos em imagens são interpretados como textura também, e não apenas, aqueles relacionados à característica de uma superfície física. No entanto, para que isso aconteça, os tais padrões devem possuir, pelo menos, alguma das propriedades citadas acima estruturada pelos textons (MIRMEHDI; XIE; SURI, 2008).

Na Figura 3.1, é apresentada uma textura com padrões locais repetidos (Figura 3.1 (b)), e também, o conceito de padrão local singular (Figura 3.1 (a)). Pode-se notar que a textura serve para representar detalhes em uma imagem.

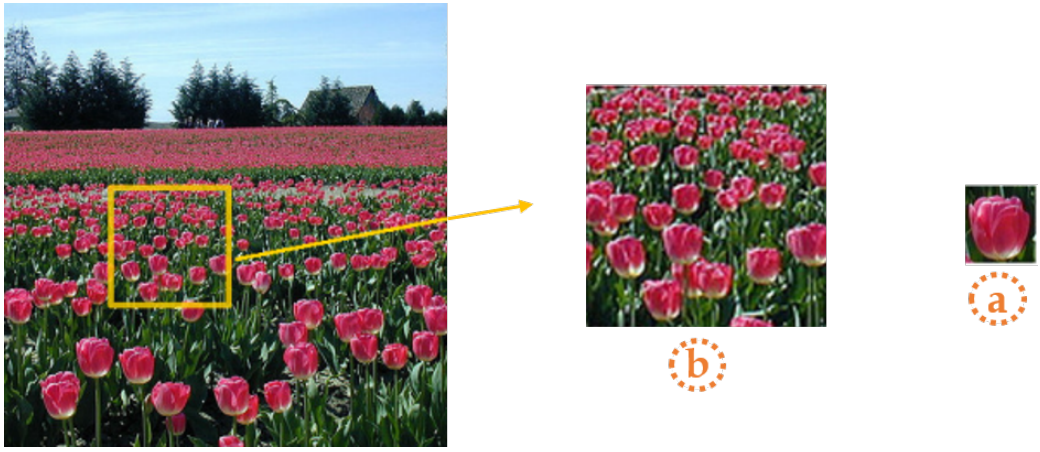


Figura 3.1: Exemplo de textura com padrões locais repetidos (b) e padrão local singular (a)

Existem, principalmente, dois tipos de texturas: microtextura e macrotextura, classificadas de acordo com a regularidade dos micro padrões presentes no material que compõe uma determinada superfície e da possibilidade ou não de conseguir qualificar informações relacionadas à forma. As duas subseções a seguir são reservadas para fornecer detalhes sobre cada uma das

categorias de texturas.

3.2.1 Microtextura

A microtextura é caracterizada por primitivas ou micro padrões que formam o panorama de uma determinada superfície. Esses micro padrões são irregulares, o que dificulta a elaboração de um modelo estrutural que possa ser capaz de descrevê-los adequadamente, uma vez que é difícil obter informações relacionadas à forma. Assim, é comum, adotar uma abordagem estocástica para sua descrição. Ou seja, utilizar-se de um descritor que possa, estatisticamente, sumarizar a relação existente entre os padrões presentes no material que uma superfície é composta.

3.2.2 Macrotextura

O contraste entre sombras e áreas bem iluminadas, e principalmente, a forma dos padrões em texturas da categoria macrotextura criam primitivas com dimensão maior quando comparada às dimensões de microtextura. Além disso, os tais padrões junto com sua forma são bem repetitivos no material que compõe a superfície sendo avaliada.

Na Figura 3.2 são exemplificados os conceitos de micro e macro texturas e um caso específico onde as duas categorias estão presentes no mesmo material, portanto, não no mesmo contexto. Na Figura 3.2 ((a), (b)) são apresentadas texturas do tipo microtextura. Na Figura 3.2 ((c), (d)) são exemplificadas texturas do tipo macrotextura. Por fim, na Figura 3.2 ((e), (f)) encontram-se texturas de ambos os tipos, onde os padrões criados pela forma dos pedaços de madeira e dos tijolos; e as pequenas variações de níveis de cinza presentes na superfície dos pedaços de madeira e dos tijolos caracterizam, respectivamente, a macrotextura e a microtextura.

3.3 Análise de Texturas

A análise de texturas consiste em conjunto de técnicas e métodos responsáveis por processar informações de textura em todo panorama da superfície ou em determinadas regiões de interesse (ROI's) de uma imagem. De modo geral, um dos principais objetivos da análise de texturas é comparar texturas e decidir se elas são iguais ou diferentes, por exemplo. Essa tarefa é evidenciada na Figura 3.3. De acordo com (MATERKA; STRZELECKI, 1998), quatro elementos são componentes desse conjunto, a saber: extração de características, classificação por textura, segmentação por textura e reconstrução de formas através de textura. Este trabalho

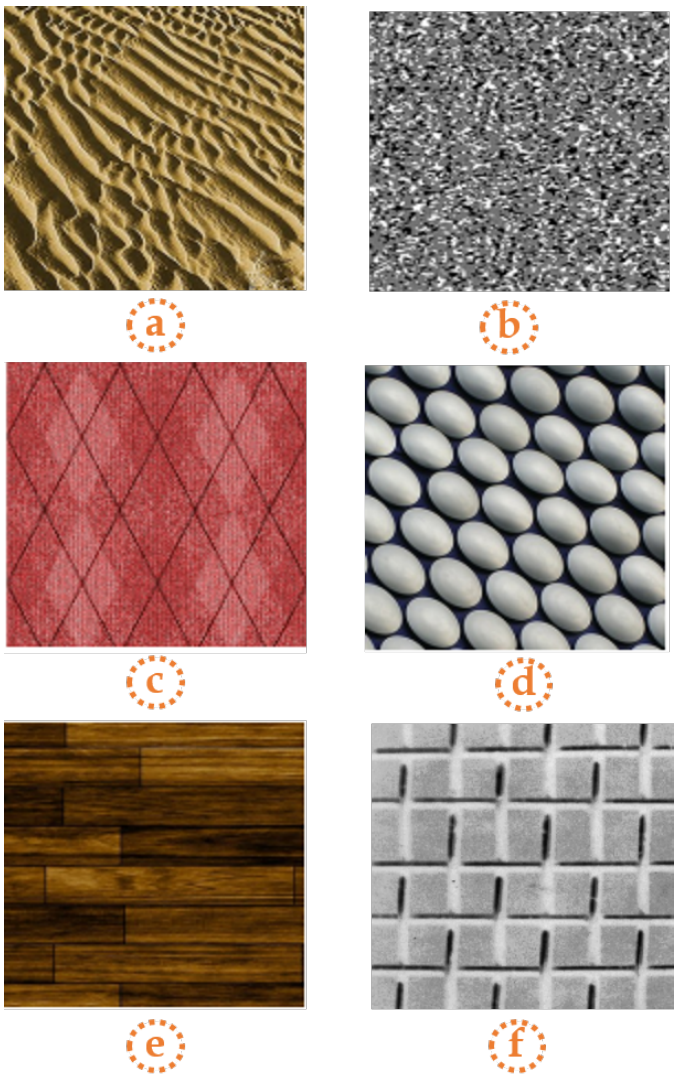


Figura 3.2: Imagens de texturas. (a) e (b) são exemplos de microtextura. (c) e (d) são exemplos de macrotextura, e (e) e (f) apresentam, cada uma, ambos os tipos juntos.

foca, principalmente, nos dois primeiros elementos desse conjunto.

3.3.1 Extração de Características

Qualquer análise de texturas começa pela extração de características em imagens. Em outras palavras, a extração de características é o primeiro passo da análise de texturas. O objetivo principal dessa tarefa consiste em revelar a correlação existente entre padrões visuais em imagens enquanto transforma tal correlação em valores quantitativos. Os descritores de textura são conhecidos como métodos responsáveis por essa função. Existem diferentes tipos de descritores, classificados de acordo com a abordagem adotada por cada um deles para sua elaboração. No entanto, na literatura, é comum encontrar diferentes formas de classificação. A primeira

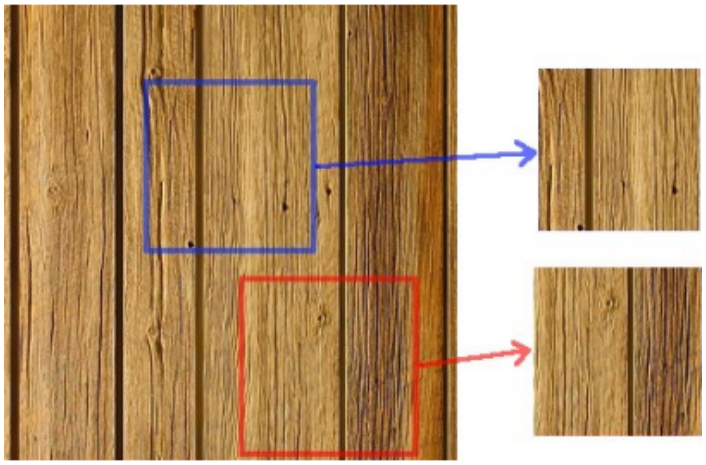


Figura 3.3: Comparação de texturas

classificação foi proposta por (BHARATI; LIU; MACGREGOR, 2004) e (MATERKA; STRZELECKI, 1998) que definem quatro tipos distintos: (1) métodos estatísticos, (2) métodos estruturais, (3) métodos baseados em modelos e (4) métodos baseados em transformadas. A segunda classificação, proposta por (DOUGHERTY, 2009) e (GONZALEZ; WOODS, 2010), os dividem em apenas três: 1) abordagem estatística, (2) abordagem estrutural e (3) abordagem espectral. Por fim, (MäENPää; PIETIKäINEN,), (RAO, 1990) e (TUCERYAN; JAIN,) os retratam em apenas dois grupos: 1) abordagem estatística ou estocástica e (2) abordagem estrutural. Além das abordagens para descritores de textura citadas acima, existem abordagens que, ao longo do tempo e com a extensa pesquisa nessa área, passaram a ser empregadas para descriminar a textura também. Essas abordagens são baseadas em modelos auto regressivos, campos aleatórios Gaussiano Markoviano, wavelets, fractais e outros. Esses modelos fornecem ferramentas mais poderosas para análise de texturas que preserve a invariançia à algumas transformacões. Decorrente disso, (ZHANG; TAN, 2002) sugeriram uma nova categorização relativamente independente com objetivo de apresentar uma classificação mais clara. A seguir são apresentados detalhes de cada um dos métodos da primeira classe de descritores apresentado acima.

Os métodos estatísticos representam a textura por meio de propriedades estatísticas não determinísticas que cercam a distribuição e a correlação existentes entre os níveis de cinza de uma imagem. O reconhecimento de texturas pelo sistema visual humano através das propriedades estatísticas (estatísticas de primeira e segunda ordem) foi investigado por (JULESZ, 1975). Mais tarde, os estudos conduzidos por (JULESZ, 1981) dão, de fato, origem a esse modelo estatístico.

Como visto na seção 2.1, no contexto de visão computacional, as estatísticas de primeira or-

dem referem-se aos momentos estatísticos computados a partir do histograma de uma imagem. O histograma de uma imagem, por sua vez, refere-se à medida de frequência com a qual os tons de cinza aparecem em uma imagem. Em outras palavras, um histograma deve ser visto como a distribuição probabilística dessas frequências. Assim, um momento estatístico seria uma medida ou informação básica da distribuição dos níveis de cinza. Como exemplos de estatísticas de primeira ordem destacam-se a entropia, variância, energia, assimetria e curtose.

O arranjo de pixels produzido por um histograma de níveis de cinza pode oferecer informações importantes para a descrição de texturas (GONZALEZ; WOODS, 2010), no entanto, a falta de levar em consideração a estrutura de dependência espacial dos pixels deixa o histograma a desejar. Para contornar esse problema, utilizam-se métodos baseados nas estatísticas de segunda ordem, ou seja, estatísticas dadas por pares de pixels. O mais popular descritor de textura baseado em estatísticas de segunda ordem é a chamada matriz de co-ocorrência (GLCM) (HARALICK, 1979). Uma vez computada, essa matriz carrega todas as informações de dependência espacial entre pixels de uma imagem. De modo sucinto, uma matriz de co-ocorrência expressa a probabilidade $P(i, j \mid d, \theta)$ ou frequência relativa das vezes em que pares de pixels i e j estão entre si à uma distância d e ângulo θ. De acordo com (HARALICK; SHANMUGAM; DINSTEIN, 1973), as relações entre tons de cinza são caracterizadas por uma função de distâncias e ângulos, e representam toda informação de texturas quando contidas nessas estruturas.

A abordagem estrutural, investigada por (HARALICK, 1979) e (LEVINE, 1985), refere-se a análise de texturas decompondo a imagem em primitivas chamadas de texels (*texture elements*). Além dos textels, é preciso definir as regras de posicionamento dos mesmos para uma descrição completa da textura. Ou seja, os métodos estruturais modelam a textura como sendo um conjunto de texels possuindo um arranjo específico de acordo com determinadas regras de posicionamento. Alguns exemplos evidenciando esses conceitos podem ser vistos na Figura 3.4. A principal vantagem da abordagem estrutural é que ela fornece uma boa descrição simbólica de uma imagem, portanto, ela é mais adequada para tarefas de análise do que de síntese de texturas (MATERKA; STRZELECKI, 1998). Como as técnicas estruturais restringem-se à macrotexturas bem definidas, consequentemente, não apresentam uma boa performance na análise de texturas naturais.

No contexto dos métodos baseados em modelos, a textura é modelada como sendo um modelo matemático probabilístico ou como sendo a combinação linear de conjunto de funções bases. Por isso, dizem-se que são modelos orientados a paradigmas estocásticos e generativos, respectivamente. Para análise de textura nesse contexto, primeiro, os coeficientes desses modelos são estimados e depois usados para caracterizar imagens de textura. A questão chave

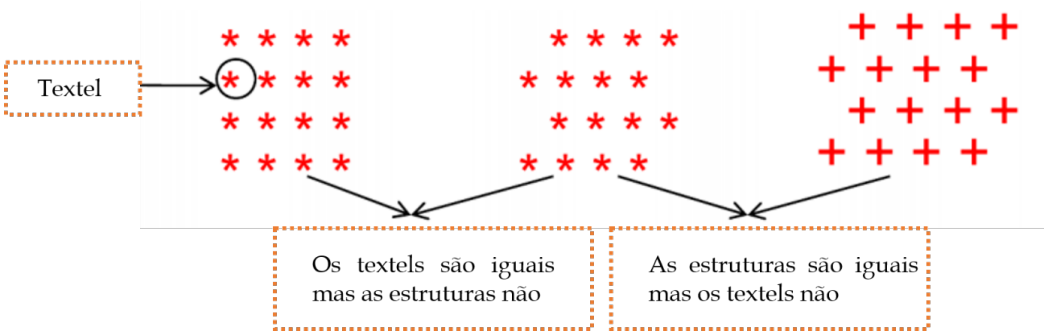


Figura 3.4: Textels e suas estruturas

sempre é como estimar os coeficientes do modelo e como escolher o modelo adequado para determinada textura (ZHANG; TAN, 2002). Os exemplos clássicos desses paradigmas são campos aleatórios Gaussianos Markovianos e fractais, respectivamente. Devido a sua distribuição de probabilidade condicional local, o modelo de campos aleatórios Gaussianos Markovianos é capaz de encapsular dependências espaciais entre um pixel e seus vizinhos. Isto é, a tal distribuição de probabilidade estabelece que o valor de cada pixel possui uma relação direta de dependência com os seus respectivos vizinhos (Zhao et al., 2007). Os fractais baseiam-se nas regras de construção sistemáticas em diferentes escalas e têm mostrado uma boa performance para modelar e representar superfícies naturais, uma vez que padrões provenientes da natureza, na maioria dos casos, apresentam qualidades afins entre si (com algumas variações estatísticas) em diferentes níveis de escala (MANDELBROT, 2007), (TUCERYAN; JAIN,), (JAIN, 1989) e (DOUGHERTY, 2009).

Os descritores baseados em transformadas representam a imagem em espaço cujo sistema de coordenadas possui uma interpretação que esteja estreitamente relacionada com as características (como frequências, por exemplo) da textura sendo analisada (MATERKA; STRZELECKI, 1998). Nessa abordagem, a análise de textura é feita de acordo com as frequências que compõem a imagem. Dentre esses métodos destacam-se transformada de Fourier, filtros de Gabor e transformadas *wavelet*. Para obtenção das informações de textura, os descritores de Fourier utilizam-se do espectro de potências ou energia do sinal. De acordo com (GONZALEZ; WOODS, 2010), esse espectro tem um papel fundamental na caracterização da textura. Ele é capaz de descrever a direcionalidade dos padrões periódicos de uma imagem. Vale notar que métodos baseados nas transformadas de Fourier não apresentam uma boa performance na prática devido à falta de informações de localização espacial, isto é, falta informação que indica em qual instante de tempo determinada frequência aparece exatamente. Os filtros Gabor, por sua vez, proporcionam meios para uma melhor localização espacial. No entanto, a sua utilidade é limitada na prática porque geralmente não há uma única resolução de filtro na

qual se pode localizar uma estrutura espacial em texturas naturais, por exemplo (MATERKA; STRZELECKI, 1998). As transformadas *wavelet*, por sua vez, apresentam uma representação mais completa da imagem utilizando-se da abordagem multiresolução. Essa abordagem permite ter informações sobre as variações do sinal em diferentes escalas e tanto no domínio de espaço como no domínio de frequência. Quando comparadas a filtros de Gabor, pode-se notar que as transformadas *wavelet* apresentam algumas vantagens na análise de texturas. Como por exemplo, variando a resolução espacial (diferentes escalas), isso possibilita a representação de texturas na escala mais adequada. Além disso, existem uma gama ampla de funções *wavelet*, isso abre a possibilidade de escolha de acordo com o domínio de aplicação na qual a tarefa de análise de texturas está sendo executada.

3.3.2 Classificação de Texturas

O objetivo principal de uma tarefa de classificação por texturas consiste em prever em qual classe (duas ou mais classes conhecidas de antemão) uma amostra de textura pertence de acordo com um determinado critério de similaridade. Essa tarefa contém dois processos relacionados à extração de features seguida da classificação de padrões. A classificação de padrões remete diretamente ao uso de algoritmos de aprendizagem de máquinas supervisionados. Isto é, faz-se necessário, em primeiro lugar, ter de antemão um conjunto de amostras para o treinamento. Em seguida, cria-se um mapeamento entre os valores dessas amostras e o rótulo de suas respectivas classes, permitindo a inferência de novas amostras desconhecidas, denominadas amostras de teste. Essas amostras de teste podem ser obtidas de duas formas: a partir de validação cruzada (cross-validation) ou retiradas de amostras não incluídas no processo de treinamento. Segundo (BARROW; CRONE, 2016) , a validação cruzada é uma prática estatística que mede o quanto os resultados de uma estimativa são capazes de generalizar uma base de dados independente.

3.4 A Transformada *Wavelet*

Em geral, a transformada *wavelet* consiste em decompor uma função $f(t) \in L^2(R)$[1] de energia finita em dois componentes mais simples: aproximação e detalhe, utilizando-se diferentes escalas a e deslocamentos b. Os componentes de aproximação e detalhe são obtidos por meio da translação e dilatação de uma única função, denominada *wavelet* mãe $\psi(t)$. O fator de escala a causa tanto a compressão como a dilatação quando for $0 < a < 1$ e $a > 1$, respectivamente.

[1]Considera-se que $L^2(R) = \{f : R \to C | \int_{-\infty}^{\infty} |f(t)|^2 dt < \infty\}$ e é o conjunto de todas as funções com energia finita.

A transformada *wavelet* pode ser representada na forma contínua (conhecida como transformada *wavelet* contínua) bem como na forma discreta (conhecida como transformada *wavelet* discreta). Através de operações de dilatações (isto é, contraindo ou esticando a função *wavelet* por um fator de $1/a$ e translações (isto é, deslocando b ao longo do eixo de tempo), a família de *wavelets* escaladas e transladadas pode ser vista como sendo funções parametrizadas por a e b e obtida por,

$$\psi_{a,b}(t) = \frac{1}{\sqrt{a}}\psi\left(\frac{t-b}{a}\right), \tag{3.1}$$

com $a \in R^+$, $b \in R$ e $a \neq 0$. O propósito de ter o fator $1/\sqrt{a}$ na equação (3.1) é de garantir que a energia da família *wavelet* permaneça a mesma nas demais escalas. De acordo com (RIOUL; VETTERLI, 1991), a transformada *wavelet* contínua de um sinal $f(t)$ é dada pela seguinte expressão,

$$\begin{aligned} \omega_{a,b} = \frac{1}{\sqrt{a}}\int_{-\infty}^{+\infty} f(t)\psi^*\left(\frac{t-b}{a}\right)dt = f * \bar{\psi}_a(b), \\ \text{com } \bar{\psi}_a(b) \text{ definido por,} \\ \bar{\psi}_a(b) = \frac{1}{\sqrt{a}}\psi\left(\frac{-t}{a}\right), \end{aligned} \tag{3.2}$$

em que $\omega_{a,b}$ representa o coeficiente *wavelet* de $f(t)$; $*$ representa o operador convolucional entre pares de funções; e $\psi^*(t)$ o conjugado complexo de $\psi(t)$.

Para que a transformada *wavelet* contínua seja invertível, a função $\psi(t)$ deve satisfazer a condição de admissibilidade $0 < C < \infty$ em que,

$$C = \int_0^{+\infty} \frac{\Psi(u) * X(u)}{u}du = \int_{-\infty}^{0} \frac{\Psi(u) * X(u)}{u}du, \tag{3.3}$$

em que $\Psi(f)$ e $X(f)$ são as transformadas de Fourier de $\psi(t)$ e $\chi(t)$, respectivamente. Vale ressaltar que, nem sempre $\chi(t)$ é uma função *wavelet*. Caso isso seja possível, teremos $C = \int_0^{+\infty} \frac{1}{u}|\Psi(u)|^2 du$. Normalemente C é chamada de constante de admissibilidade. Além disso, a tal condição de admissibilidade implica em $\Psi(0) = 0$, ou seja, a função *wavelet* ψ possui média zero.

Finalmente, nessas condiçoes de admissibilidade, a função $f(t)$ pode ser recuperada a partir de $\omega_{a,b}$ por meio da transformação inversa (WAKIN, 2011),

$$f(t) = \frac{1}{C}\int_0^{+\infty}\int_{-\infty}^{+\infty}\frac{1}{\sqrt{a}}.\omega_{a,b}.\chi\left(\frac{t-b}{a}\right)\frac{db.da}{a^2}, \tag{3.4}$$

sendo que $\chi(t)$ é conhecida como a função dual de $\psi(t)$.

Como visto anteriormente, a transformada *wavelet* contínua pode variar continuamente enquanto transladando e dilatando o sinal. Dessa forma, a sua aplicação em um sinal levará na geração de informações redundantes. Essa redundância pode ser, condicionalmente, benéfica para algumas aplicações como filtragem de ruídos em sinais, e maligna, para aplicações como compressão de imagens. De modo geral, para sistemas digitais e, em específico, para compressão de imagens é preciso executar a transformada *wavelet* no modo discreto. Com isso, diminuíra-se a quantidade de redundância dos coeficientes *wavelet* em diferentes escalas da decomposição, enquanto preserva-se as informações contidas no sinal original. A discretização dos parâmetros de escala a e de translação b é feita de acordo com a discretização logarítmica a seguir

$$\begin{cases} a = a_0^j \\ \\ b = kb_0a_0^j \end{cases} \qquad a_0 > 1, b_0 \neq 0 \text{ and } j,k \in Z$$

onde Z representa o conjunto de inteiros, a_0 é o parâmetro de dilatação fixo e b_0 o fator de translação fixo o qual depende do fator de dilatação. Dessa forma, a família de *wavelets* de interesse correspondente é dada por:

$$\psi_{j,k}(t) = \frac{1}{\sqrt{a_0^j}}\psi\left(\frac{t-kb_0a_0^j}{a_0^j}\right). \tag{3.5}$$

Geralmente, 2 e 1 são os valores adotados para a_0 e b_0, respectivamente (ADDISON, 2002). Com isso, a equação (3.5) torna-se

$$\psi_{j,k}(t) = \frac{1}{\sqrt{2^j}}\psi\left(\frac{t-k2^j}{2^j}\right) = 2^{-j/2}\psi\left(2^{-j}t-k\right). \tag{3.6}$$

A equação (3.6) forma uma base ortonormal para o espaço de funções $L^2(R)$ ou L^2-integráveis. Nesse caso, a transformada *wavelet* é então dada por,

$$\omega_{j,k} = 2^{-j/2}\int_{-\infty}^{+\infty} f(t).\psi\left(2^{-j}t-k\right)dt. \tag{3.7}$$

Os coeficientes $\omega_{j,k}$ são chamados de coeficientes *wavelet* ou de detalhe.

O principal foco da transformada *wavelet* base-se na utilização da análise multiresolução cujo objetivo é capturar propriedades de sinais em diferentes escalas ou resoluções (diferentes níveis de "zoom"). Dessa forma, uma função $f(t)$ pode ser decomposta em duas sequências de subespaços aninhados de detalhes e de aproximações com menor resolução. Nesse sentido, enquanto $V_j \subset V_{j+1}, j \in Z$ são denominados espaços de aproximação, $W_j \subset W_{j+1}, j \in Z$ são denominados espaços de detalhe. Quanto maior o número de j, maior a resolução. Nessa configuração, W_j é um subconjunto de V_{j+1}. Assim, $f_{j+1}(t) \in V_{j+1}$ pode ser aproximada por $f_j(t) \in V_j$ e W_j como

$$\begin{aligned} f_{j+1}(t) &= f_j(t) + W_j(t) \\ f_{j+1}(t) &= f_{j-1}(t) + W_{j-1}(t) + W_j(t) \\ f_{j+1}(t) &= f_{j-2}(t) + W_{j-2}(t) + W_{j-1}(t) + W_j(t). \end{aligned}$$

Como pode ser observado, a aproximação em um dado nível de decomposição é igual à aproximação mais detalhe no nível anterior. Por exemplo, a aproximação no nível cinco é igual à aproximação no nível quatro mais detalhes no nível quatro e a aproximação no nível quatro é igual à aproximação no nível três mais detalhes no nível três. Isso nos permite decompor um sinal em uma aproximação f_1 e uma série de detalhes em diferentes resoluções: $w_1, w_2, w_3, w_4, ..., w_n$ no qual cada detalhe contém um tipo de informação diferente.

Em termos práticos, uma forma eficiente para realizar a transformada *wavelet* discreta é através de filtragens sucessivas do sinal original. Considerando-se dois filtros, um passa-baixa l e outro passa-alta h, a ideia é dividir o espectro de frequência original exatamente ao meio. Os componentes resultantes da filtragem passa-alta contêm altas frequências que fornecem informações com mínimos detalhes. Já os resultantes da filtragem passa-baixa fornecem apenas uma visão global destas frequências.

Em resumo, a transformada *wavelet* discreta pode ser calculada com sequências sucessivas desses dois filtros seguido da dizimação (*downsampling*) pelo fator 2: $\downarrow 2$. Figura 3.5 mostra a decomposição de uma imagem 2D utilizando a transformada *wavelet* discreta. As saídas da Figura 3.5 (a aproximação e os respectivos detalhes) são fornecidas pelas equações (3.8) and (3.9), respectivamente.

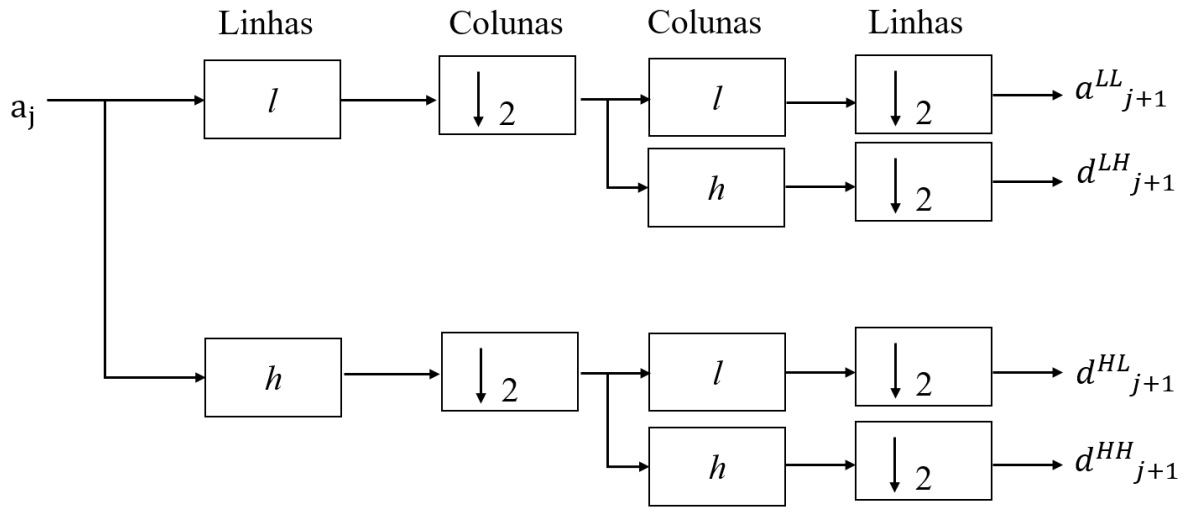


Figura 3.5: Decomposição de uma imagem 2D utilizando a transformada *wavelet* discreta.

$$a_{j+1}[n] = \sum_{k=-\infty}^{+\infty} l[2n-k]a_j[k] \tag{3.8}$$

$$d_{j+1}[n] = \sum_{k=-\infty}^{+\infty} h[2n-k]a_j[k] \tag{3.9}$$

Nas equações (3.8) and (3.9), os a_j são utilizados para a computação dos coeficientes na próxima escala e os d_j, conhecidos como coeficientes *wavelet*, constituem a saída da transformada. É importante ressaltar que, na escala $j+1$, o número de amostras é reduzido por um fator de 2. Além disso, qualquer decomposição *wavelet* de uma imagem 2D envolve quatro sub-bandas: *LL* (Aproximação), *LH* (Detalhe Horizontal), *HL* (Detalhe Vertical) e *HH* (Detalhe Diagonal). A imagem de sub-banda *LL* é usada apenas para a computação da transformada na próxima escala. Existem muitos tipos de transformada *wavelet* , como Haar, Daubechies, Symlets, etc. A decisão de usar uma delas deve levar em consideração a forma como ela se adapta melhor à uma determinada situação.

3.5 Campos Aleatórios Markovianos

3.5.1 Fundamentos

Esta seção tem como objetivo apresentar alguns conceitos fundamentais acerca dos campos aleatórios markovianos. Especificamente, vamos apresentar alguns conceitos sobre cadeias de Markov, noção de vizinhança e cliques e algumas propriedades de Markov.

3.5.1.1 Cadeias de Markov

Processos estocásticos são sequências de variáveis aleatórias independentes e identicamente distribuídas. Como modelos, essas sequências de variáveis nem sempre são interessantes porque apresentam um comportamento mais ou menos iguais. Quando se pensa em ter maior variabilidade, pode-se permitir alguma dependência com o passado, abrindo a possibilidade para determinar equações de recorrência. Essa é uma das características fundamentais de Cadeias de Markov homogêneas e discretas no tempo, já que podem ser representadas por uma equação de recorrência estocástica $X_{n+1} = f(X_n, Z_{n+1})$ onde $\{Z_n\}_{n\geq 1}$ é uma sequência independente e identicamente distribuída, independente do estado inicial X_0.

É importante ressaltar que a noção de dependência probabilística introduzida com o passado é feita somente com o estado imediatamente anterior e essa configuração, por menor que pareça, é suficiente para produzir uma grande variedade de comportamentos.

Como nosso interesse está em sistemas não determinísticos, consideramos $X_n \geq 0$ como variáveis aleatórias definidas sobre um espaço de probabilidades. Neste caso a sequência $\{X_n\}_{n\geq 0}$ de um espaço enumerável Ω com elementos $x_0, x_1, x_2, ..., x_n$ constitui um processo estocástico de tempo discreto com espaço de estado Ω. Assim, se $X_n = x_n$ então o processo é dito estar no estado x_n no tempo n ou visita o estado x_n tempo n. Sabe-se que a maioria dos sistemas assume que dado um estado presente, os estados passados não influenciam o futuro. Esta propriedade é chamada de propriedade de Markov e um sistema que possui essa propriedade é chamado Cadeia de Markov. A seguir, apresentamos uma definição formal da mesma:

Definição 3.5.1 (Cadeia de Markov). Seja $\{X_n\}_{n\geq 0}$ um processo estocástico a tempo discreto com um espaço enumerável Ω. Se para todo inteiro $n \geq 0$ e todo estado $x_0, x_1, x_2, ..., x_n$ em Ω,

$$P(X_n = x_n | X_0 = x_0, ..., X_{n-1} = x_{n-1}) = P(X_n = x_n | X_{n-1} = x_{n-1}) \tag{3.10}$$

e ambos os lados estiverem bem definidos, este processo estocástico é chamado Cadeia de Markov. As probabilidades condicionais $P(X_n = x_n | X_{n-1} = x_{n-1})$ são chamadas probabilidades de transição da cadeia. Caso alguma delas for independente de n então ela é dita estacionária. Para ilustrar esses conceitos, segue exemplo de cadeia de Markov de dois estados x_0 e x_1.

Exemplo 3.5.1 (Cadeia de Markov de dois estados). Considere uma máquina que no começo de um determinado dia ou está quebrada ou está funcionando. Assume que se a máquina está quebrada no começo do $n-$ésimo dia, existe uma probabilidade p de que a máquina esteja funcionando no $(n+1)-$ésimo dia. Além disso, assume que se a máquina está funcionando no

$n-$ésimo dia existe uma probabilidade q de ela estar quebrada no $(n+1)-$ésimo dia. Finalmente, seja $\pi_0(0)$ a probabilidade de que a máquina está quebrada no $0-$ésimo dia. O estado 0 corresponde à máquina quebrada e o estado 1 corresponde à máquina que está operando. Seja X_n a variável aleatória denotando o estado da máquina no tempo n. De acordo com a descrição acima, tem-se que

$$\begin{aligned} P(X_{n+1}=1|X_n=0) &= p, \\ P(X_{n+1}=0|X_n=1) &= q, \\ P(X_0=0|X_n=1) &= \pi_0(0). \end{aligned}$$

Como existem somente dois estados, x_0 e x_1, deduz-se imediatamente o seguinte:

$$\begin{aligned} P(X_{n+1}=0|X_n=0) &= 1-p, \\ P(X_{n+1}=1|X_n=1) &= 1-q, \end{aligned}$$

e a probabilidade $\pi_0(1)$ de estar inicialmente no estado $x_1=1$ é dada por

$$\pi_0(1) = P(X_0=1) = 1-\pi_0(0).$$

A partir destas informações, podemos calcular $P(X_n=0)$ e $P(X_n=1)$. Observa-se que:

$$\begin{aligned} P(X_{n+1}=0) &= P(X_{n+1}=0 \cap X_n=0) + P(X_{n+1}=0 \cap X_n=1) \\ &= P(X_n=0)P(X_{n+1}=0|X_n=0) + P(X_n=1)P(X_{n+1}=0|X_n=1) \\ &= P(X_n=0)(1-p) + P(X_n=1)q \\ &= (1-p)P(X_n=0) + q(1-P(X_n=0)) \\ &= (1-p)P(X_n=0) - qP(X_n=0) + q \\ &= (1-p-q)P(X_n=0) + q \end{aligned}$$

Como $P(X_0=0)=\pi_0(0)$, então

$$P(X_1 = 0) = (1 - p - q)\pi_0(0) + q$$

e

$$\begin{aligned} P(X_2 = 0) &= (1 - p - q)P(X_1 = 0) + q \\ &= (1 - p - q)^2\pi_0(0) + q[1 + (1 - p - q)] \end{aligned}$$

Repetindo o processo n vezes teremos

$$P(X_n = 0) = (1 - p - q)^n\pi_0(0) + q\sum_{j=0}^{n-1}(1 - p - q)^j.$$

Sabemos que

$$\sum_{j=0}^{n-1}(1 - p - q)^j = \frac{1 - (1 - p - q)^n}{p + q}$$

consequentemente temos

$$P(X_n = 0) = \frac{q}{p+q} + (1 - p - q)^n(\pi_0(0) - \frac{q}{p+q}) \tag{3.11}$$

$$P(X_n = 1) = \frac{p}{p+q} + (1 - p - q)^n(\pi_0(1) - \frac{p}{p+q}). \tag{3.12}$$

Sendo $|1 - p - q| < 1$, $0 < p + q| < 2$ e neste caso quando $n \to \infty$ as probabilidades são

$$\begin{aligned} \lim_{n\to\infty} P(X_n = 0) &= \frac{q}{p+q} \\ \lim_{n\to\infty} P(X_n = 1) &= \frac{p}{p+q} \end{aligned}$$

3.5.1.2 Campos Aleatórios

Como visto na seção anterior, um processo Markoviano possui a seguinte propriedade de Markov: a probabilidade de um evento no tempo $n+1$, dado todos os eventos anteriores depende somente do evento no tempo n. Além disso, é preciso ter uma distribuição inicial e sua respectiva probabilidade de transição para que se construa um determinado processo Markoviano. Observa-se, a partir da própria descrição da propriedade de Markov, que não existe nenhuma simetria no tempo, já que ela se refere à probabilidade de um evento futuro dado o passado. Ao mesmo tempo, um processo Markoviano possui uma propriedade mais simétrica: a probabilidade de que um determinado evento aconteça no tempo n dado todos os eventos anteriores e futuros depende somente dos eventos nos tempos $n-1$ e $n+1$. Essa característica mais simétrica permite que a propriedade de Markov seja generalizada imediatamente para qualquer grafo. Essa generalização leva à noção de Campo Aleatório Markoviano definida abaixo.

Definição 3.5.2 (Campo Aleatório). Seja S um conjunto finito, com elementos denotados por s chamados de sítios. E seja Ω um conjunto finito chamado de espaço de fases. Um campo aleatório sobre S com fases em Ω é uma coleção $X = X(S)_{s\in S}$ de variáveis aleatórias $X(S)$ com valores em Ω.

Para que o campo aleatório definido acima seja um campo aleatório Markoviano é preciso introduzir a propriedade de Markov. Para isso, é necessário introduzir uma topologia (sistema de vizinhança) sobre os elementos de S.

Definição 3.5.3 (Vizinhança). Um sistema de vizinhanças sobre S é uma família $N = \{N_s\}_{s\in S}$ de subconjuntos S tal que para todo $s \in S$, temos

(i) $s \notin N_s$,

(ii) $t \in N_s \Rightarrow s \in N_t$

O subconjunto N_s é chamado vizinhança do sítio s. O par (S,N) é chamado grafo ou topologia. No contexto de noção de grafos, S é o conjunto de vértices e N define as arestas: os sítios s e t são ligados por uma aresta se e somente se eles são vizinhos, isto é, $t \in N_s$.

Definição 3.5.4 (Campo Aleatório Markoviano). Dado um grafo não direcionado $G = (V,E)$, um conjunto de variáveis aleatórias $X = X_{vv\in V}$ formam um campo aleatória de Markov com a relação ao grafo G se satisfazerem as propriedades de Markov a seguir:

- **Propriedade de Markov dos pares**
  Quaisquer duas varáveis não adjacentes são condicionalmente independentes, dado todas

as outras variáveis:
$X_u \perp X_v | X_{V\setminus\{u,v\}}\ if\ \{u,v\} \notin E$.

- **Propriedade de Markov local**
  Uma varável é condicionalmente independente de todas as outras variáveis, dados os seus vizinhos:
  $X_u \perp X_{V\setminus cl(u)} | X_{ne(u)}$,
  Onde $ne(u)$ é o conjunto de vizinhos de u e $cl(u) = u \cup ne(u)$ é a vizinhança de u.

- **Propriedade de Markov global**
  Quaisquer dois subconjuntos de variáveis são condicionalmente independentes dado a separação do subconjunto:
  $X_A \perp X_B | X_S$,
  onde cada caminho de um nó em A para um nó em B passa por S.

Agora vamos ver a noção do que é conhecido como *clique*. Dado um grafo arbitrário diz-se que um conjunto de vértices C é um clique se cada par de vértices desse conjunto são conectados por uma aresta, ou seja, são vizinhos.

Definição 3.5.5 (Clique). Qualquer conjunto unitário s é um clique. Um subconjunto $C \subset S$ com mais que um elemento é chamado um clique de um grafo (S,N) se e somente se quaisquer dois sítios de C são vizinhos.

Além do clique, existe também o conceito de *potencial*. Um potencial V é uma maneira de atribuir um número $V_c(x)$ para toda subconfiguração de x. A seguir, é apresentado o conceito do potencial de Gibbs.

Definição 3.5.6 (Potencial de Gibbs). Um potencial de Gibbs sobre Ω^s relativo a um sistema de vizinhanças N é uma coleção $\{V_C\}_{C\subset S}$ de funções $V_C : \Omega^s \to R$ tal que

(i) $V_C \equiv 0$ se C não é um clique,

(ii) para todo $x, x' \in \Omega^s$ e para todo $C \subset S$,
$x(C) = x'(C) \Rightarrow V_C(x) = V_C(x')$.

A função energia $E : \Omega^s \to R$ é dita derivar de um potencial $\{V_C\}_{C\subset S}$ se

$$E(x) = \sum_C V_C(x)$$

Se considerar $P(X = x)$ como sendo a probabilidade de encontrar as variáveis X assumindo o estado (fase ou valor) x, $P(X = x)$ será definida em relação a V_C da maneira seguinte:

$$P(X = x) = \prod_{C \in cl(G)} \phi_C(x_C)$$

onde $\phi_C = V_C$ e $cl(G)$ é o conjunto de cliques do grafo G. Vale lembrar que a função V_C depende somente das fases dos sítios dentro do subconjunto considerado.

3.5.2 O modelo Gaussiano-Markoviano

GMRF (Gaussian Markov Random Fields) é um tipo específico de campo aleatório de Markov em que as variáveis aleatórias possuem uma distribuição normal multivariada. Este tipo de modelo é importante e adequado para representar efeitos aleatórios com estrutura de correlação induzida por grafos. Sabe–se que a matriz de precisão (inversa da matriz de covariâncias) desse modelo é esparsa, ou seja, cheia de zeros, esse fato facilita muito a implementação de métodos de simulação Monte Carlo via cadeias de Markov, por exemplo.

Considere um grafo $G = (V,E)$ e seja Q uma matriz $n \times n$ simétrica e definida positiva tal que o elemento Q_{ij} é igual a zero, se, e somente se, os vértices i e j não estiverem ligados por uma aresta. Com isso, um GMRF pode ser definido formalmente da seguinte maneira:

Definição 3.5.7 (GMRF). Um vetor aleatório $X \in R^n$ é denominado GMRF com respeito ao grafo $G = (V,E)$ com média μ e matriz de precisão $Q > 0$ (definida positiva), se e somente se, a distribuição conjunta do vetor é dada por

$$P(X = x) = (2\pi)^{-\frac{n}{2}} |Q|^{\frac{1}{2}} exp(-\frac{1}{2}(x-\mu)^T Q(x-\mu))$$

onde $Q_{ii} \neq 0 \iff i, j \in E$ para todo $i \neq j$.

Em outras palavras, pode-se dizer que uma distribuição normal multivariada forma um campo aleatório de Markov em relação a um grafo $G = (V,E)$ se as arestas faltantes no grafo considerado correspondem aos zeros na matriz de precisão, isto equivale a

$$X = (X_v)_{v \in V} \sim N(\mu, \sum)$$

de tal forma que

$$(\sum^{-1})_{uv} = 0 \; if \; \{u,v\} \notin E$$

Nota-se que $\sum^{-1} = Q$.

Sabe-se que campos aleatórios Markovianos Gaussianos (GMRFs) são ferramentas importantes para modelagem de dados espaciais. Por essa razão, no contexto desse trabalho, tais modelos são utilizados para a captura da estrutura de dependência espacial subjacente em imagens. Um dos motivos para essa escolha é a tratabilidade matemática que oferece e que possibilita a definição de expressões fechadas para diversos estimadores. No contexto deste trabalho, consideramos a *isotropic pairwise* GMRF, isto é, considera-se uma interação binária no modelo restringindo o número máximo de cliques a 2 e o parâmetro β, cujo o prinncipal papel é controlar as interações entre variáveis vizinhas, é invariante com relação as mudanças nas direções. Na equação (3.13) é apresentada a função densidade condicional local de um GMRF em que $\theta = (\mu, \sigma^2, \beta)$ é o vetor de parâmetros composto pela média, variância e inverso da temperatura, usado para o controle da estrutura de dependência espacial global do sistema. Note que se $\beta = 0$, as variáveis no campo são independentes e, consequentemente, o modelo degenera-se para uma simples densidade Gaussiana, ou seja, não há dependência.

$$p(x_i|\eta_i, \theta) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{ -\frac{1}{2\sigma^2}\left[x_i - \mu - \beta \sum_{j \in \eta_i} (x_j - \mu) \right]^2 \right\} \tag{3.13}$$

O valor da intensidade do pixel na posição i é representado por x_i e η_i denota a sua vizinhança.

3.5.3 Estimação de parâmetros

Os parâmetros do modelo, $\theta = (\mu, \sigma^2, \beta)$, são estimados usando a técnica chamada de máxima pseudo-verossimilhança, através da equações propostas no capítulo 4. Essa abordagem, proposta por (BESAG, 1974), é uma alternativa à estimação máxima verossimilhança que se mostra inviável para estimar os parâmetros de campos aleatórios Markovianos, devido a existência das funções de partição nas distribuições conjuntas de Gibbs. O método máxima pseudo-verossimilhança é baseado no princípio da independência condicional. Brevemente falando, a função pseudo-verossimilhança é definida como produto das funções densidade de

probabilidade condicional local, conhecidas como LCDF's *(Local Conditional Density Functions)* em termo inglês, para todas n variáveis do sistema, modelado como campo aleatório. Vale lembrar que se $\beta = 0$, os estimadores máxima pseudo-verossimilhança de μ e σ^2 tornam-se a média e a variância amostrais, respectivamente.

3.6 Teoria da informação

A teoria da informação é um ramo da matemática cujo principal objetivo é quantificar informações na transmissão, processamento, extração e compressão de dados. Foi reconhecido como uma nova área de pesquisa após o trabalho seminal de Claude Shannon, *"A Mathematical Theory of Communication"*, publicado em 1948. É particularmente relevante no estudo de processos aleatórios como uma maneira de caracterizar e quantificar a noção de incerteza de forma precisa e formal.

Muitos algoritmos de compactação de dados foram desenvolvidos a partir do conceito da entropia de Shannon, que é uma medida para estimar a máxima eficiência possível que um algoritmo de codificação pode atingir. Foi demonstrado que a entropia está diretamente relacionada à compressão teórica máxima para um determinado alfabeto de mensagens. Outra aplicação da teoria da informação é a construçaõ dos códigos de correção e detecção de erros. Basicamente, os métodos de correção de erros adicionam bits extras aos dados, a fim de ajudar a corrigir erros. Desse modo, esses métodos operam na direção oposta à da compressão de dados. Os códigos de detecção de erros, por outro lado, indicam que ocorreu um erro, mas não o corrigem automaticamente.

A criptologia é a ciência da comunicação segura de dados e envolve dois processos principais: a criptoanálise, que é responsável por revelar a mensagem criptografada, e a criptografia, que se preocupa com como as informações são criptografadas. Shannon notou que simples transposições de cifras (aquelas obtidas permutando as letras do alfabeto) não afetam a entropia, porque elas re-rotulam os caracteres de sua fórmula sem alterar suas probabilidades associadas. Ele também percebeu que, na criptografia, a quantidade de incerteza que se pode introduzir na solução não pode ser maior que a incerteza principal, ou seja, que chaves aleatórias devem ser selecionadas para tornar a criptografia mais segura. É por isso que números aleatórios são cruciais para a segurança da informação.

No reconhecimento de padrões e no aprendizado de máquina, o problema de quantificar a semelhança entre diferentes objetos ou agroupamentos é uma tarefa desafiadora, especialmente nos casos em que a distância Euclidiana padrão não seja uma escolha razoável. Muitos traba-

lhos sobre seleção de características adotam divergências estatísticas para escolher o conjunto de características que maximizam alguma medida de separação entre classes. Além disso, na maioria dos sistemas de reconhecimento, geralmente as tarefas de aprendizado são modeladas como problemas de otimização, ou seja, é preciso minimizar uma função de erro para aprender os pesos adequados da rede. Os mínimos quadrados são freqüentemente aplicados no problema de estimativa, no entanto, várias funções de custo baseadas em medidas da teoria da informação têm mostrado resultados promissores.

O processamento digital de sinais e imagens é outra área de pesquisa na qual a teoria da informação traz muitos benefícios. Desde a simples compressão de imagem até os algoritmos de remoção de ruído em imagem, a entropia e as divergências estatísticas relacionadas fornecem uma ferramenta matemática proveitosa. Por exemplo, na remoção de ruído de imagem baseada em *patch*, a estimativa de um dado pixel não ser ruídoso depende dos pesos de similaridade entre um *patch* de referência e *patches* não locais ao longo da imagem. No caso do ruído Gaussiano, a simples distância Euclidiana é a escolha usual para a medida de similaridade. No entanto, foi demonstrado que, para outros tipos de ruído, as divergências baseadas na teoria da informação podem melhorar o desempenho dos filtros na remoção de ruído de imagem.

3.6.1 Entropia

Vamos começar por introduzir a entropia de uma variável aleatória x como sendo o valor esperado da auto-informação:

$$H(p) = -\int p(x)[log\ p(x)]dx = -E\left[log\ p(x)\right] \tag{3.14}$$

onde $p(x)$ é a função densidade de probabilidade (fdp) de x. Supondo que x seja normalmente distribuída como $N(\mu,\sigma^2)$, a fdp $p(x)$ é dada por:

$$p(x;\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} exp\left\{-\frac{1}{2\sigma^2}(x-\mu)^2\right\} \tag{3.15}$$

onde μ representa a média e σ^2 representa a variância de x. Calculando o logaritmo da fdp temos:

$$log\ p(x) = -\frac{1}{2} log\ (2\pi\sigma^2) - \frac{1}{2\sigma^2}(x-\mu)^2 \tag{3.16}$$

Substituindo a equação (3.16) na equação (C.1) resulta em:

$$H(p) = \frac{1}{2} log\ (2\pi\sigma^2) + \frac{1}{2\sigma^2} E[(x-\mu)^2] = \frac{1}{2} log\ (2\pi\sigma^2) + \frac{1}{2} = \frac{1}{2}\left(1 + log\ (2\pi\sigma^2)\right) \quad (3.17)$$

Suponha agora que temos o vetor aleatório $\vec{x} \in R^d$ cuja fdp é um Gaussiano multivariado $N(\vec{\mu}, \Sigma)$, onde $\vec{\mu}$ é o vetor das médias e Σ é a matriz de covariância:

$$p(\vec{x}; \vec{\mu}, \Sigma) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} exp\left\{ -\frac{1}{2}(\vec{x}-\vec{\mu})^T \Sigma^{-1} (\vec{x}-\vec{\mu}) \right\} \quad (3.18)$$

Assim, o logaritmo da fdp é dada por:

$$log\ p(\vec{x}; \vec{\mu}, \Sigma) = -\frac{d}{2} log(2\pi) - \frac{1}{2} log|\Sigma| - \frac{1}{2}(\vec{x}-\vec{\mu})^T \Sigma^{-1} (\vec{x}-\vec{\mu}) \quad (3.19)$$

o que resulta em

$$\begin{aligned}
H(p) &= \frac{d}{2} log(2\pi) + \frac{1}{2} log|\Sigma| + \frac{1}{2} E\left[(\vec{x}-\vec{\mu})^T \Sigma^{-1} (\vec{x}-\vec{\mu})\right] \\
&= \frac{d}{2} log(2\pi) + \frac{1}{2} log|\Sigma| + \frac{1}{2} E\left[Tr\left((\vec{x}-\vec{\mu})^T \Sigma^{-1} (\vec{x}-\vec{\mu})\right)\right] \\
&= \frac{d}{2} log(2\pi) + \frac{1}{2} log|\Sigma| + \frac{1}{2} E\left[Tr\left(\Sigma^{-1} (\vec{x}-\vec{\mu})(\vec{x}-\vec{\mu})^T\right)\right] \\
&= \frac{d}{2} log(2\pi) + \frac{1}{2} log|\Sigma| + \frac{1}{2} Tr\left(\Sigma^{-1} E\left[(\vec{x}-\vec{\mu})(\vec{x}-\vec{\mu})^T\right]\right) \\
&= \frac{d}{2} log(2\pi) + \frac{1}{2} log|\Sigma| + \frac{1}{2} Tr\left(\Sigma^{-1}\Sigma\right) \\
&= \frac{d}{2} log(2\pi) + \frac{1}{2} log|\Sigma| + \frac{1}{2} Tr\,(I) \\
&= \frac{1}{2} log|\Sigma| + \frac{d}{2}\,(1 + log(2\pi))
\end{aligned} \quad (3.20)$$

Observe que, como no caso univariado, a entropia de um vetor aleatório Gaussiano não depende da média.

3.6.2 Informação de Fisher

Desde sua definição, nos trabalhos de Sir Ronald Fisher, o conceito de informação de Fisher está presente de maneira ubíqua em toda a estatística matemática, desempenhando um papel importante em várias aplicações, desde métodos de estimação numérica baseados na iteração de Newton-Raphson até a Definition 3.dos limites inferiores na estimativa imparcial (limite infe-

rior de Cramer-Rao). Mais recentemente, com o desenvolvimento da geometria da informação, foi descoberto outro papel fundamental da informação de Fisher nos modelos estatísticos: ela define as propriedades geométricas intrínsecas do espaço paramétrico de um modelo, caracterizando o tensor métrico da respectiva variedade (*manifold*). Em outras palavras, a matriz de informação de Fisher é a métrica Riemanniana natural da variedade (espaço paramétrico), dado um modelo estatístico.

De um modo geral, a informação de Fisher pode ser pensada como um análogo a entropia, que é freqüentemente usada como uma medida de incerteza, mas é baseada em probabilidade, não em verossimilhança, como é o caso de informação de Fisher. Basicamente, no contexto da teoria da informação, a informação de Fisher mede a quantidade de informação que uma amostra aleatória carrega sobre um parâmetro desconhecido.

Considerando a fdp $p(x;\vec{\theta})$, onde $\vec{\theta} \in R^k$, a matriz da informação de Fisher é definida como:

$$I(\vec{\theta})_{ij} = E\left[\left(\frac{\partial}{\partial\theta_i} log\ p(x;\vec{\theta})\right)\left(\frac{\partial}{\partial\theta_j} log\ p(x;\vec{\theta})\right)\right] = -E\left[\frac{\partial^2}{\partial\theta_i\partial\theta_j} log\ p(x;\vec{\theta})\right] \tag{3.21}$$

Agora mostramos que, sob certas condições de regularidade, a igualdade de informação se mantém. Primeiro, observe que:

$$E\left[\frac{\partial^2}{\partial\theta_i\partial\theta_j} log\ p(x;\vec{\theta})\right] = E\left[\frac{\partial}{\partial\theta_j}\left(\frac{\partial}{\partial\theta_i} log\ p(x;\vec{\theta})\right)\right] = E\left[\frac{\partial}{\partial\theta_j}\left(\frac{1}{p(x;\vec{\theta})}\frac{\partial}{\partial\theta_i}p(x;\vec{\theta})\right)\right] \tag{3.22}$$

Pela regra do produto, temos:

$$E\left[\frac{\partial}{\partial\theta_j}\left(\frac{1}{p(x;\vec{\theta})}\frac{\partial}{\partial\theta_i}p(x;\vec{\theta})\right)\right] = E\left[-\frac{1}{p(x;\vec{\theta})^2}\frac{\partial}{\partial\theta_i}p(x;\vec{\theta})\frac{\partial}{\partial\theta_j}p(x;\vec{\theta}) + \frac{1}{p(x;\vec{\theta})}\frac{\partial^2}{\partial\theta_i\partial\theta_j}p(x;\vec{\theta})\right]$$
$$= -E\left[\left(\frac{1}{p(x;\vec{\theta})}\frac{\partial}{\partial\theta_i}p(x;\vec{\theta})\right)\left(\frac{1}{p(x;\vec{\theta})}\frac{\partial}{\partial\theta_i}p(x;\vec{\theta})\right)\right] + E\left[\frac{1}{p(x;\vec{\theta})}\frac{\partial^2}{\partial\theta_i\partial\theta_j}p(x;\vec{\theta})\right] \tag{3.23}$$

Pela Definition 3.do valor esperado, o segundo termo da equação (3.23), pode ser simplificado a:

$$E\left[\frac{1}{p(x;\vec{\theta})}\frac{\partial^2}{\partial\theta_i\partial\theta_j}p(x;\vec{\theta})\right] = \int p(x;\vec{\theta})\frac{1}{p(x;\vec{\theta})}\frac{\partial^2}{\partial\theta_i\partial\theta_j}p(x;\vec{\theta})dx = \int \frac{\partial^2}{\partial\theta_i\partial\theta_j}p(x;\vec{\theta})dx \tag{3.24}$$

Sob certas condições de regularidade, é possível trocar os operadores de integração e diferenciação:

$$\int \frac{\partial^2}{\partial\theta_i\partial\theta_j}p(x;\vec{\theta})dx = \frac{\partial^2}{\partial\theta_i\partial\theta_j}\int p(x;\vec{\theta})dx = \frac{\partial^2}{\partial\theta_i\partial\theta_j}1 = 0 \tag{3.25}$$

Observe também que:

$$\frac{1}{p(x;\vec{\theta})}\frac{\partial}{\partial\theta_i}p(x;\vec{\theta}) = \frac{\partial}{\partial\theta_i}log\ p(x;\vec{\theta}) \tag{3.26}$$

o que finalmente resulta na igualdade:

$$E\left[\left(\frac{\partial}{\partial\theta_i}log\ p(x;\vec{\theta})\right)\left(\frac{\partial}{\partial\theta_j}log\ p(x;\vec{\theta})\right)\right] = -E\left[\frac{\partial^2}{\partial\theta_i\partial\theta_j}log\ p(x;\vec{\theta})\right] \tag{3.27}$$

Considerando o caso do Gaussiano univariado, onde $\vec{\theta} = (\mu, \sigma^2)$ o logaritmo da verrossimilhança (*log-likelihood*) é:

$$log\ p(x;\vec{\theta}) = -\frac{1}{2}log\ (2\pi) - \frac{1}{2}log\ \sigma^2 - \frac{1}{2\sigma^2}(x-\mu)^2 \tag{3.28}$$

e a primeira e segunda derivada em relação a μ são:

$$\frac{\partial}{\partial\mu}log\ p(x;\vec{\theta}) = \frac{1}{\sigma^2}(x-\mu) \tag{3.29}$$

$$\frac{\partial^2}{\partial\mu^2}log\ p(x;\vec{\theta}) = -\frac{1}{\sigma^2} \tag{3.30}$$

o que resulta em:

$$-E\left[\frac{\partial^2}{\partial\mu^2}log\ p(x;\vec{\theta})\right] = \frac{1}{\sigma^2} \tag{3.31}$$

É evidente ver que as derivadas parciais de segunda ordem são zero, ou seja,:

$$\frac{\partial^2}{\partial\mu\partial\sigma^2} log\ p(x;\vec{\theta}) = \frac{\partial^2}{\partial\sigma^2\partial\mu} log\ p(x;\vec{\theta}) = 0 \tag{3.32}$$

A primeira e segunda derivada em relação a σ^2 são dadas por:

$$\frac{\partial}{\partial\sigma^2} log\ p(x;\vec{\theta}) = -\frac{1}{2\sigma^2} + \frac{1}{2\sigma^4}(x-\mu)^2 \tag{3.33}$$

$$\frac{\partial^2}{\partial(\sigma^2)^2} log\ p(x;\vec{\theta}) = \frac{1}{2\sigma^4} - \frac{1}{\sigma^6}(x-\mu)^2 \tag{3.34}$$

Calculando o valor esperado da segunda derivada resulta em:

$$-E\left[\frac{\partial}{\partial\sigma^2} log\ p(x;\vec{\theta})\right] = \frac{1}{2\sigma^4} - \frac{1}{\sigma^6}E\left[(x-\mu)^2\right] = -\frac{1}{2\sigma^4} + \frac{1}{\sigma^4} = \frac{1}{2\sigma^4} \tag{3.35}$$

Assim, a matriz da informação de Fisher do modelo é :

$$I(\vec{\theta}) = \begin{bmatrix} \frac{1}{\sigma^2} & 0 \\ 0 & \frac{1}{2\sigma^4} \end{bmatrix} \tag{3.36}$$

A geometria da informação tem sido uma área de pesquisa relevante desde os trabalhos pioneiros de Shun'ichi Amari nos anos 80, desenvolvidos por meio da aplicação de métodos teóricos de geometria diferencial ao estudo da estatística matemática. Desde então, essa área foi expandida e explorada com sucesso por pesquisadores de diversas áreas da ciência, da física estatística e mecânica quântica à teoria dos jogos e aprendizado de máquina.

Essencialmente, a geometria da informação pode ser vista como um ramo da teoria da informação que fornece um tratamento robusto e geométrico para a maioria dos modelos paramétricos em estatística matemática. Nesse contexto, é possível investigar como duas variáveis aleatórias independentes e distintas do mesmo modelo paramétrico estão relacionadas em termos de características geométricas intrínsecas. Por exemplo, neste arcabouço (*framework*), é possível medir distâncias entre duas variáveis aleatórias Gaussianas $X \sim N(\mu_x, \sigma_x^2)$ e $Y \sim N(\mu_y, \sigma_y^2)$.

Suponha que $p(x;\vec{\theta})$ seja um modelo estatístico pertencente à família exponencial, em que $\vec{\theta}$ indique o vetor de parâmetros do modelo. Então, a coleção de todos os vetores admissíveis $\vec{\theta}$ define o espaço paramétrico Θ, que provou ser uma variedade Riemanniana. Além disso,

foi demonstrado que, no caso Gaussiano, a variedade é uma superfície com curvatura negativa constante, definindo sua geometria como hiperbólica. Como o espaço paramétrico Θ não é um espaço Euclidiano, segue-se que a variedade é curvada. Assim, para que se torne possivel o cálculo de distâncias e comprimentos de arco em Θ, é necessário expressar um deslocamento infinitesimal ds na variedade de maneira adaptativa ou local.

De modo geral, essa é a razão pela qual uma variedade deve estar equipada com um tensor métrico, que é a estrutura matemática responsável pela Definition 3.de produtos internos nos espaços tangentes locais. Com o tensor métrico, é possível expressar o quadrado de um deslocamento infinitesimal na variedade, ds^2, em função de um deslocamento infinitesimal no espaço tangente, que no caso de uma variedade 2D é dado por um vetor $[du, dv]$. Assumindo uma notação matricial, temos:

$$ds^2 = \begin{bmatrix} du & dv \end{bmatrix} \begin{bmatrix} A & B \\ B & C \end{bmatrix} \begin{bmatrix} du \\ dv \end{bmatrix} = Adu^2 + 2Bdudv + Cdv^2 \tag{3.37}$$

onde a matriz de coeficientes A, B, e C é o tensor métrico. Se o tensor métrico for uma matriz definida positiva, a variedade é conhecida como Riemanniana. Observe que no caso Euclidiano, onde o tensor métrico é a matriz identidade (como o espaço é plano), temos a conhecida relação de Pitágoras $ds^2 = du^2 + dv^2$. A matriz da informação de Fisher é o tensor métrico do espaço paramétrico.

Por exemplo, podemos expressar um deslocamento infinitesimal na variedade das densidades Gaussianas univariadas em termos da matriz da informação de Fisher da seguinte maneira:

$$ds^2 = \begin{bmatrix} d\mu & d\sigma^2 \end{bmatrix} \begin{bmatrix} \frac{1}{\sigma^2} & 0 \\ 0 & \frac{1}{2\sigma^4} \end{bmatrix} \begin{bmatrix} d\mu \\ d\sigma^2 \end{bmatrix} = \frac{1}{\sigma^2} d\mu^2 + \frac{1}{2\sigma^4} (d\sigma^2)^2 \tag{3.38}$$

Observa-se que a partir da equação (3.21) é possível calcular a matriz da informação de Fisher esperada de um modelo de duas maneiras diferentes, mas equivalentes, devido à propriedade de permutabilidade dos operadores de integração e diferenciação. Assim, ao lidar com o modelo GMRF, portanto, essas duas matrizes (uma obtida através da primeira derivada no logaritmo da função verossimilhança e outra através da segunda derivada) não são equivalentes. Eles desempenham papéis distintos e podem ser usados para quantificar diferentes aspectos do comportamento global esperado do sistema. Nesse caso, refere-se à primeira matriz da informação de Fisher como a do tipo-I e à segunda como sendo a do tipo-II.

Dois componentes das matrizes de informação de Fisher são particularmente relevantes para essa pesquisa (os relacionados ao parâmetro de acoplamento β, um do tipo-I e outro do

tipo-II):

1. $I^{(1)}_{\beta\beta}(\vec{\theta})$ quantifica o grau médio de concordância entre um pixel central, x_i, e a configuração definida por seu sistema de vizinhança para um determinado β. Em outras palavras, fornece um grau médio de quanto um pixel central difere de seus vizinhos dentro do GMRF. Como o valor do β determina o comportamento global esperado dos *patches* (se β for grande, é esperado um alto grau de dependência espacial entre as observações e se β estiver próximo de zero, as observações serão independentes), os padrões de configuração mais alinhados com esse comportamento global esperado têm uma pequena contribuição para $I^{(1)}_{\beta\beta}(\vec{\theta})$. Por outro lado, os *patches* que são muito informativos (uma vez que não se espera que existam para esse valor específico de β), têm uma grande contribuição para $I^{(1)}_{\beta\beta}(\vec{\theta})$.

2. $I^{(2)}_{\beta\beta}(\vec{\theta})$ mede o grau de concordância ou dependência entre as observações pertencentes ao mesmo sistema de vizinhança η_i. Em outras palavras, mede a estabilidade de tais *patches*. Portanto, os padrões considerados mais suscetíveis a alterações (não possuem uma configuração "estável") têm uma pequena contribuição para $I^{(2)}_{\beta\beta}(\vec{\theta})$. Por outro lado, *patches* estáveis têm uma grande contribuição para $I^{(2)}_{\beta\beta}(\vec{\theta})$ (é improvável que ocorram alterações em sua estrutura no caso de pequenas perturbações).

No geral, essas medidas estão relacionadas à distribuição de *patches* no campo aleatório, que nesta pesquisa é representada por imagens de textura. Para obter mais detalhes sobre o papel das medidas de informação de Fisher na dinâmica de sistemas complexos não determinísticos, o leitor pode-se referir aos trabalhos de Levada (LEVADA, 2011, 2017), onde simulações de Cadeias Markovianas de Monte-Carlo são realizadas para estudar o comportamento de campos aleatórios Gaussianos numa perspectiva da teoria da informação.

3.7 Redução de Dimensionalidade Linear

Os métodos de redução de dimensionalidade lineares foram desenvolvidos em muitas áreas da ciência, como estatística, otimização, aprendizado de máquina e outras áreas aplicadas por mais de um século, e os mesmos se tornaram ferramentas matemáticas poderosas para analisar dados ruídos e de alta dimensionalidade (CUNNINGHAM; GHAHRAMANI, 2015). Parte do sucesso dos métodos lineares vem do fato de eles terem interpretações geométricas simples e propriedades computacionais tipicamente atraentes (CUNNINGHAM; GHAHRAMANI,

2015). Basicamente, nos métodos lineares, busca-se uma matriz de projeção T que possa mapeiar as amostras de entrada no espaço original (R^m) para um subespaço linear em R^d, com $d < m$. Existem várias maneiras de definir métodos de redução de dimensionalidade linear, mas aqui adotamos a estrutura de otimização introduzida por Cunningham and Ghahramani (CUNNINGHAM; GHAHRAMANI, 2015).

Definição 3.7.1 (Redução de Dimensionalidade Linear). Dado um conjunto de n pontos $X = [\vec{x}_1, \vec{x}_2, ..., \vec{x}_n] \in R^{m \times n}$ tendo cada ponto a dimensionalidade m e dada uma opção de dimensionalidade $d < m$, otimize alguma função objetiva $f_X(.)$ para produzir uma transformação linear $T \in R^{d \times m}$ e chame $Y = TX \in R^{d \times n}$ de conjunto de dados transformados e de menor dimensionalidade.

3.7.1 Análise de Componentes Principais

Análise de Componentes Principais, ou simplesmente PCA, é um método computacional que implementa a transformação de Karhunen-Loève, também conhecida como transformação de Hotteling, uma clássica técnica de estatística multivariada que expande um determinado vetor aleatório $\vec{x} \in R^m$ em autovetores de sua matriz de covariância (JOLLIFFE, 2002). O PCA é o método mais conhecido para compressão de dados e extração de características. O PCA não faz suposições sobre funções densidade de probabilidade, pois todas as informações necessárias ao método podem ser estimadas diretamente dos dados (HASTIE; TIBSHIRANI; FRIEDMAN, 2009). Como o PCA depende exclusivamente da matriz de covariância, ele é um método estatístico de segunda ordem. O PCA é construído de duas maneiras diferentes: 1) maximizando a variância da nova representação compacta Y; 2) minimizando o erro quadrático médio (EQM) entre os dados originais X e a nova representação compacta Y. Do ponto de vista estatístico, o objetivo do PCA é reduzir a redundância entre as variáveis aleatórias que compõem o vetor aleatório $\vec{x} \in R^m$, que é medida pelas correlações entre eles. Nesse sentido, o PCA primeiro decorrelaciona as características e, em seguida, reduz a dimensionalidade encontrando novas características que são combinações lineares das originais.

3.7.2 PCA pela Maximização da Variância

Seja $Z = [T^T, S^T]$ uma base ortonormal para R^m em que $T^T = [\vec{w}_1, \vec{w}_2, ..., \vec{w}_d]$ representa os componentes $d < m$ que desejamos reter durante o processo de redução de dimensionalidade e $S^T = [\vec{w}_{d+1}, \vec{w}_{d+2}, ..., \vec{w}_m]$ são os componentes remanescentes que devem ser descartados. Em outras palavras, T define o subespaço PCA linear e S define o subespaço linear eliminado pelo

processo de redução (YOUNG; CALVERT, 1974).

O problema em questão pode ser resumido como: dado um espaço de características de entrada, queremos encontrar d direções $\vec{w}_j$, for $j = 1,2,...,d$ em que, ao projetar so dados, a variação seja maximizada. Em outras palavras, queremos as direções que maximizem a dispersão dos dados. A questão é: como obter as direções $\vec{w}_j$? Sem perda de generalidade, assumimos que a amostra $X = [\vec{x}_1, \vec{x}_2, ..., \vec{x}_n]$ tem média zero, ou seja , os dados estão centralizados em torno da origem.

Observe que podemos escrever $\vec{x} \in R^m$ como uma expansão na base ortonormal Z como:

$$\vec{x} = \sum_{j=1}^{m} (\vec{x}^T \vec{w}_j)\vec{w}_j = \sum_{j=1}^{m} c_j \vec{w}_j \tag{3.39}$$

onde c_j são os coeficientes da expansão.

Assim, o novo vetor $\vec{y} \in R^d$ pode ser obtido pela transformação $\vec{y} = T\vec{x}$, ou seja:

$$\vec{y}^T = \vec{x}^T T^T = \sum_{j=1}^{m} c_j \vec{w}_j^T [\vec{w}_1, \vec{w}_2, ..., \vec{w}_d] \tag{3.40}$$

Como temos uma base ortonormal, $\vec{w}_i^T \vec{w}_j = 1$ para $i = j$ e $\vec{w}_i^T \vec{w}_j = 0$ para $i \neq j$, isso resulta em:

$$\vec{y}^T = [c_1, c_2, ..., c_d] \tag{3.41}$$

Desta forma, busca-se uma transformação linear T que maximize a variação retida nos dados, ou seja, queremos maximizar as seguintes funções (HYVARINEN; KARHUNEN; OJA, 2001):

$$J_1^{PCA}(T) = E[\|\vec{y}\|^2] = E[\vec{y}^T \vec{y}] = \sum_{j=1}^{d} E[c_j^2] \tag{3.42}$$

Como c_j é a projeção de $\vec{x}$ em $\vec{w}_j$, isto é, $c_j = \vec{x}^T \vec{w}_j$, temos:

$$J_1^{PCA}(T) = \sum_{j=1}^{d} E\left[\vec{w}_j^T \vec{x}\vec{x}^T \vec{w}_j\right] = \sum_{j=1}^{d} \vec{w}_j^T E\left[\vec{x}\vec{x}^T\right] \vec{w}_j = \sum_{j=1}^{d} \vec{w}_j^T \Sigma_x \vec{w}_j \tag{3.43}$$

onde Σ_x representa a matriz de covariância do conjunto de pontos X.

Portanto, temos o seguinte problema de otimização restrita:

$$\max_{\vec{w}_j} \sum_{j=1}^{d} \vec{w}_j^T \Sigma_x \vec{w}_j \qquad \text{sujeito a} \quad \left\|\vec{w}_j\right\| = 1 \quad \text{para} \quad j = 1,2,...,d \tag{3.44}$$

que é resolvido pelos multiplicadores de Lagrange. A função de Lagrange é dada por:

$$J_1^{PCA}(T,\lambda_1,\lambda_2,...,\lambda_d) = \sum_{j=1}^{d} \vec{w}_j^T \Sigma_x \vec{w}_j - \sum_{j=1}^{d} \lambda_j \left(\vec{w}_j^T \vec{w}_j - 1\right) \tag{3.45}$$

Derivando em relação ao $\vec{w}_j$ e igualando o resultado a zero nos dá a condição necessária para o ótimo:

$$\frac{\partial}{\partial \vec{w}_j} J_1^{PCA}(T,\lambda_1,\lambda_2,...,\lambda_d) = \Sigma_x \vec{w}_j - \lambda_j \vec{w}_j = 0 \tag{3.46}$$

o que resulta em equação de autovetores:

$$\Sigma_x \vec{w}_j = \lambda_j \vec{w}_j \tag{3.47}$$

Voltando ao problema de otimização, podemos reescrevê-lo como:

$$\max_{\vec{w}_j} \sum_{j=1}^{d} \vec{w}_j^T \Sigma_x \vec{w}_j = \max_{\vec{w}_j} \sum_{j=1}^{d} \vec{w}_j^T \lambda_j \vec{w}_j = \max_{\vec{w}_j} \sum_{j=1}^{d} \lambda_j \tag{3.48}$$

o que significa que devemos selecionar os k autovetores associados aos k maiores autovalores da matriz de covariância dos dados para compor a base do subespaço PCA linear.

3.7.3 PCA pela Minimização do EQM

Outra propriedade ideal do subespaço PCA é que ele minimiza o erro quadrático médio entre X e Y. Em outras palavras, a aproximação do PCA é a melhor representação em termos de compactação de dados. Suponha que o erro quadrático médio entre um vetor aleatório $\vec{x} \in R^m$ e um vetor aleatório $\vec{y} \in R^d$ seja:

$$J_2^{PCA}(T) = E\left[\left\|\vec{x} - \vec{y}\right\|^2\right] = E\left[\left\|\vec{x} - \sum_{j=1}^{d} (\vec{w}_j^T \vec{x})\vec{w}_j\right\|^2\right] \tag{3.49}$$

Expandindo a norma, temos a seguinte expressão para o erro quadrático médio:

$$J_2^{PCA}(T) = E\left[\left(\vec{x} - \sum_{j=1}^{d}(\vec{w}_j^T\vec{x})\vec{w}_j\right)^T\left(\vec{x} - \sum_{j=1}^{d}(\vec{w}_j^T\vec{x})\vec{w}_j\right)\right] \tag{3.50}$$

Aplicando a propriedade distributiva resulta em:

$$J_2^{PCA}(T) = E\left[\vec{x}^T\vec{x} - \vec{x}^T\left(\sum_{j=1}^{d}(\vec{w}_j^T\vec{x})\vec{w}_j\right) - \left(\sum_{j=1}^{d}(\vec{w}_j^T\vec{x})\vec{w}_j\right)^T\vec{x} + \left(\sum_{j=1}^{d}(\vec{w}_j^T\vec{x})\vec{w}_j\right)\left(\sum_{j=1}^{d}(\vec{w}_j^T\vec{x})\vec{w}_j\right)^T\right] \tag{3.51}$$

Usando a linearidade do operador de valor esperado e reorganizando os termos resulta em:

$$J_2^{PCA}(T) = E\left[\|\vec{x}\|^2\right] - E\left[\sum_{j=1}^{d}(\vec{w}_j^T\vec{x})(\vec{w}_j^T\vec{x})\right] - E\left[\sum_{j=1}^{d}\vec{w}_j^T(\vec{x}^T\vec{w}_j)\vec{x}\right] + E\left[\left(\sum_{j=1}^{d}\vec{w}_j^T(\vec{x}^T\vec{w}_j)\right)\left(\sum_{j=1}^{d}(\vec{w}_j^T\vec{x})\vec{w}_j\right)\right] \tag{3.52}$$

Simplificando os produtos pontuais, temos:

$$J_2^{PCA}(T) = E\left[\|\vec{x}\|^2\right] - \sum_{j=1}^{d}E\left[(\vec{w}_j^T\vec{x})^2\right] - \sum_{j=1}^{d}E\left[(\vec{w}_j^T\vec{x})^2\right] + \sum_{j=1}^{d}\sum_{k=1}^{d}E\left[(\vec{w}_j^T\vec{x})(\vec{x}^T\vec{w}_k)\vec{w}_j^T\vec{w}_k\right] \tag{3.53}$$

Como $\vec{w}_j$ para $j = 1,2,...,d$ define o conjunto de vetores ortonormais, temos:

$$\begin{aligned} J_2^{PCA}(T) &= E\left[\|\vec{x}\|^2\right] - \sum_{j=1}^{d}E\left[(\vec{w}_j^T\vec{x})^2\right] - \sum_{j=1}^{d}E\left[(\vec{w}_j^T\vec{x})^2\right] + \sum_{j=1}^{d}E\left[(\vec{w}_j^T\vec{x})^2\right] \\ &= E\left[\|\vec{x}\|^2\right] - \sum_{j=1}^{d}E\left[(\vec{w}_j^T\vec{x})^2\right] \\ &= E\left[\|\vec{x}\|^2\right] - \sum_{j=1}^{d}\vec{w}_j^T\Sigma_x\vec{w}_j \end{aligned} \tag{3.54}$$

Observe que o primeiro termo é uma constante (não depende de $\vec{w}_j$), portanto, o problema de otimização é dado por:

$$\vec{w}_j \; - \sum_{j=1}^{d} \vec{w}_j^T \Sigma_x \vec{w}_j \qquad \text{sujeito a} \quad \left\|\vec{w}_j\right\| = 1 \quad \text{para} \quad j = 1,2,...,d \tag{3.55}$$

que é equivalente à maximização da variância.

Como mencionado anteriormente, uma propriedade interessante do PCA é que ele decorrelaciona os dados. Isso pode ser facilmente visto usando a decomposição espectral (*eigendecomposition*) do Σ_x em $Q\Lambda Q^T$, em que Q é a matriz de m autovetores de Σ_x e $\Lambda = diag(\lambda_1, \lambda_2, ..., \lambda_m)$ é a matriz diagonal dos autovalores de Σ_x. Sabemos que antes da redução de dimensionalidade $\vec{y} = Z\vec{x}$, em que $Z = [T^T, S^T]$ e a matriz de covariância do vetor transformado é dada por $\Sigma_y = Z^T \Sigma_x Z$. Mas no PCA, Z é composto pelos autovetores da matriz de covariância, portanto, $Z = Q$, resultando em $\Sigma_y = Z^T Q\Lambda Q^T Z = Q^T Q\Lambda Q^T Q = \Lambda$, onde a ortonormalidade dos autovetores implica em $Q^T Q = I$.

3.7.4 Interpretação Geométrica do PCA

Na transformação PCA, primeiro os dados são centralizados, subtraindo a média de cada amostra em X. Em seguida, o vetor aleatório $\vec{x} \in R^m$ é projetado nos autovalores da matriz de covariância Σ_x, de maneira que a redundância introduzida pela correlação entre os componentes de $\vec{x}$ seja eleminada. Do ponto de vista geométrica, tal condição é alcançada através da rotação dos eixos de coordenadas. Nesse processo, as variâncias das projeções de $\vec{x}$ nos novos eixos são maximizadas e as direções com menores variâncias são descartadas, fazendo com que o erro quadrático médio seja mínimo.

Sob a hipótese gaussiana multivariada, o espaço de coordenadas rotacionado corresponde ao espaço composto pelos eixos principais do hiperelipsoide que definem a distribuição. Figura 3.6, adaptada de (FUKUNAGA, 1990), mostra um exemplo para o caso Gaussiano 2D. Os componentes principais são as projeções dos dados nos dois eixos, ϕ_1 e ϕ_2. As variâncias, indicadas aqui por λ_1 e λ_2, são distintas na maioria dos problemas e um grande número delas é tão pequeno que os componentes correspondentes podem ser descartados.

A seguir, apresentamos um algoritmo para redução de dimensionalidade por PCA.

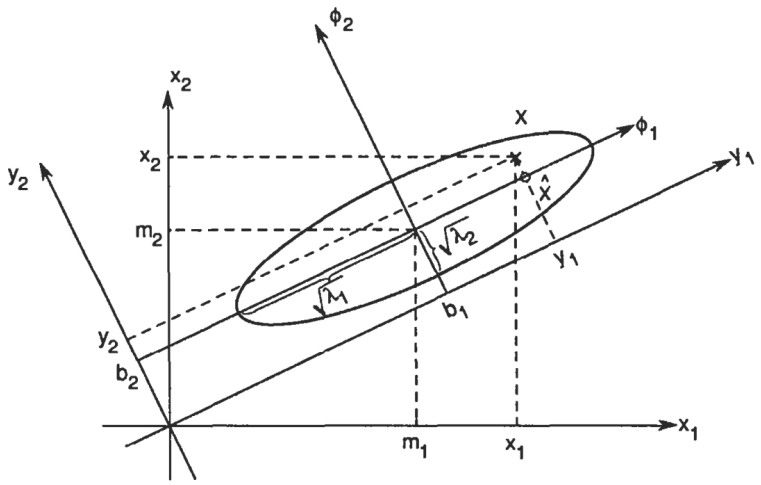


Figura 3.6: Uma ilustração da transformada de Karhunen-Loeve para um modelo Gaussiano 2D.

Algorithm 1 Análise de Componentes Principais

1: **function** PCA(X)
2: Calcule a média amostral e a matriz de covariância da amostra por:
$\mu_x = \frac{1}{n}\sum_{i=1}^{n}\vec{x}_i$
$\Sigma_x = \frac{1}{n-1}\sum_{i=1}^{n}(\vec{x}_i - \mu_x)(\vec{x}_i - \mu_x)^T$
3: Calcule os autovalores e autovetores de Σ_x
4: Define a matriz de transformação $T = [\vec{w}_1, \vec{w}_2, ..., \vec{w}_d]$ com os d autovetores associados aos d maiores autovalores.
5: Projete os dados X no subespaço PCA:
$\vec{y}_i = T\vec{x}_i \quad$ para $\quad i = 1, 2, ..., n$
6: **return** Y
7: **end function**

3.7.5 Para Além do PCA

Existem muitas extensões ao PCA, entre as quais podemos mencionar PPCA ou PCA Probabilístico (TIPPING; BISHOP, 1999; ROWEIS, 1997), em que a Definition 3.de uma medida de verrossimilhança permite a comparação com outras técnicas probabilísticas, facilitando o teste estatístico e permitindo a aplicação de modelos bayesianos e PCA robusto, onde o objetivo principal é introduzir insensibilidade ao *outlier* por meio de diferentes normas vetoriais (GALPIN; HAWKINS, 1987; CHOULAKIAN, 2006). O PCA sofre com o fato de que cada componente principal é uma combinação linear de todas as variáveis originais, portanto, muitas vezes é difícil interpretar os resultados. Para contornar esse problema, PCA tem se tornado

esparso em vários contextos para permititr características com carregamento esparso, ou seja, cada componente principal depende apenas de poucas variáveis (ZOU; HASTIE; TIBSHIRANI, 2006; JOURNEE et al., 2010).

Análise de Componentes Independentes (ICA) é uma classe de métodos que usa estatísticas de ordem superior para ir além da falta de correlação do PCA (HYVARINEN; KARHUNEN; OJA, 2001). A maximização da não-Gaussianidade é motivada pelo diverge do Limite Central, que afirma que a distribuição da soma (média ou combinação linear) de n variáveis aleatórias independentes se aproxima de uma Gaussiana à medida que n cresce. Nesse contexto, não-Gaussianidade significa independência. Esta é a base para o algoritmo FastICA (HYVANINEN, 1999).

Uma abordagem muito popular para estimar componentes independentes é o modelo de máxima verossimilhança, uma vez que é uma técnica fundamental na teoria estatística usada para resolver vários problemas. Freqüentemente, algoritmos de otimização numérica, como gradiente descendente, são aplicados para encontrar a solução. Esta é a ideia por trás do algoritmo Infomax (BELL; SEJNOWSKI, 1995; LEE; GIROLAMI; SEJNOWSKI, 1999). Uma abordagem natural para o ICA é a minimização de informações mútuas, que é uma medida da teória da informação relacionada à dependência entre variáveis aleatórias, que é sempre não-negativa, e zero, se e somente se, as variáveis são estatisticamente independentes, ou seja, pode ser usada como uma função objetiva na estimatição do ICA (AMARI; CICHOCKI; YANG, 1996; PHAM, 2004). Além disso, a informação mútua pode ser interpretada como uma distância, ou seja, como a divergência de Kullback-Leibler entre a distribuição conjunta de $\vec{x}$ e o produto das distribuições marginais de x_i, para $i = 1, 2, .., n$.

3.7.6 Fatoração Matricial Não-Negativa

A fatoração matricial não-negativa (NMF) é uma família de métodos e algoritmos numéricos em análise multivariada e álgebra linear, na qual o objetivo é fatorar uma matriz de dados V em duas matrizes W e H, com a propriedade de que todas as matrizes não tenham elementos negativos. A diferença crucial entre esses métodos e a redução de dimensionalidade linear é que esses métodos geralmente não produzem um mapeamento linear $Y = TX$, mas sim o mapeamento inverso da baixa para alta dimensão (CUNNINGHAM; GHAHRAMANI, 2015).

O NMF é aplicado à redução de dimensionalidade de dados multivariados considerando como entrada um conjunto de vetores de dados multivariados de dimensão m, $V_{m\times n}$ em que n é o número de exemplos no conjunto de dados. Essa matriz é então fatorada aproximadamente em uma $m \times r$ matriz W e uma $r \times n$ matriz H, ou seja, $V \approx WH$. Normalmente, r é escolhido para

ser menor que n ou m, de modo que W e H são menores que a matriz original X. Isso resulta em uma versão compactada da matriz de dados original (LEE; SEUNG, 1999). A importância dessa representação é que $\vec{v} \approx W\vec{h}$, ou seja, cada vetor é aproximado por uma combinação linear das colunas de W. Nesse contexto, H desempenha o papel das características extraídas, ou seja, cada coluna $\vec{h}_j$ de H representa um vetor no espaço transformado. Figura 3.7 mostra uma ilustração de uma decomposição matricial.

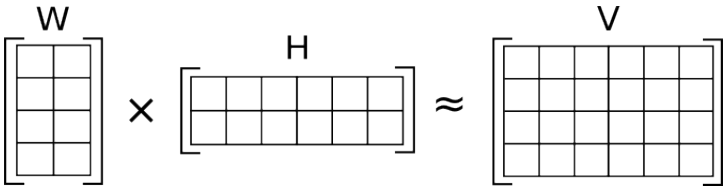


Figura 3.7: Decompondo uma matriz de dados V em duas matrizes menores H e W.

Para medir o quão próximo V é do WH, geralmente é adotada uma função de custo. Para tal, são comumente utilizadas duas medidas de distância: a distância Euclidiana e a divergência Kullback-Leibler, dadas por:

$$D_E(A,B) = \|A-B\|^2 = \sum_{i,j}(A_{i,j}-B_{i,j})^2 \tag{3.56}$$

$$D_{KL}(A,B) = \sum_{i,j}\left[A_{i,j}\, log\left(\frac{A_{i,j}}{B_{i,j}}\right) - A_{i,j} + B_{i,j}\right] \tag{3.57}$$

Observe que ambos $D_E(A,B)$ e $D_{KL}(A,B)$ são limitados inferiormente por zero e são iguais a zero se e somente se $A = B$. Portanto, $D_{KL}(A,B)$ não pode ser chamado de distância porque não é simétrico, ou seja, em geral, $D_{KL}(A,B) \neq D_{KL}(B,A)$. Embora $D_E(A,B)$ and $D_{KL}(A,B)$ sejam convexos na variável W ou na variáve H, eles não são convexos nas duas variáveis. Isso significa que o problema de minimização pode ter vários pontos ótimos locais, tornando a convergência para o mínimo global não garantida. Para obter uma boa relação entre a taxa de convergência e a facilidade de implementação, os autores evitam os algoritmos de gradiente descendente e de gradiente conjugado, propondo regras de atualização multiplicativa (LEE; SEUNG, 1999).

Teorema 3.7.1 A distância Euclidiana $D_E(V,WH) = \|V-WH\|^2$ não é crescente para um

número finito de etapas das regras multiplicativas:

$$H_{i,j}^{k+1} = H_{i,j}^{k} \frac{((W^k)^T V)_{i,j}}{((W^k)^T W^k H^k)_{i,j}} \tag{3.58}$$

$$W_{i,j}^{k+1} = W_{i,j}^{k} \frac{(V(H^{k+1})^T)_{i,j}}{(W^k H^{k+1}(H^{k+1})^T)_{i,j}} \tag{3.59}$$

onde k é o contador de iteração e W^0 and H^0 são inicializados com números aleatórios positivos, tipicamente amostrados de uma distribuição uniforme $[0,1]$.

Observe que o numerador e o denominador do fator multiplicativo na equação (3.58) são $r \times n$ matrizes. A divisão ponto a ponto do numerador pelo denominador resulta em uma matriz $r \times n$ que é multiplicada (ponto a ponto) por H^k. Essas regras multiplicativas são invariantes se e somente se W e H estiverem em um ponto estacionário de $D_E(V, WH)$, ou seja, quando $V = WH$. Observe que os fatores multiplicativos nas equações (3.58) e (3.59) são iguais a um, ou seja, temos convergência:

$$H_{i,j}^{k+1} = H_{i,j}^{k} \frac{((W^k)^T V)_{i,j}}{((W^k)^T W^k H^k)_{i,j}} = H_{i,j}^{k} \frac{((W^k)^T W^k H^k)_{i,j}}{((W^k)^T W^k H^k)_{i,j}} = H_{i,j}^{k} \tag{3.60}$$

$$W_{i,j}^{k+1} = W_{i,j}^{k} \frac{(V(H^{k+1})^T)_{i,j}}{(W^k H^{k+1}(H^{k+1})^T)_{i,j}} = W_{i,j}^{k} \frac{(W^k H^{k+1}(H^{k+1})^T)_{i,j}}{(W^k H^{k+1}(H^{k+1})^T)_{i,j}} = W_{i,j}^{k} \tag{3.61}$$

Teorema 3.7.2 A divergência de Kullback-Leibler $D_{KL}(V, WH)$ não é crescente para um número finito de etapas das regras multiplicativas:

$$W_{i,j}^{k+1} = W_{i,j}^{k} \frac{\sum_r H_{jr}^{k} \frac{V_{ir}}{(W^k H^k)_{ir}}}{\sum_r H_{jr}^{k}} \tag{3.62}$$

$$H_{i,j}^{k+1} = H_{i,j}^{k} \frac{\sum_r W_{ri}^{k+1} \frac{V_{rj}}{(W^{k+1} H^k)_{rj}}}{\sum_r W_{ri}^{k+1}} \tag{3.63}$$

onde k é o contador de iteração e W^0 e H^0 são inicializados com números aleatórios positivos, tipicamente amostrados de uma distribuição uniforme $[0,1]$.

As provas completas do Teorema 1 e 2 são descritas em detalhes no artigo seminal de Lee e Seung (LEE; SEUNG, 1999). Um aspecto importante a ser observado é que, no ponto estacionário, quando $V = WH$, as regras de atualização não alteram as estimativas atuais:

$$W_{i,j}^{k+1} = W_{i,j}^{k} \frac{\sum_r H_{jr}^{k} \frac{(W^k H^k)_{ir}}{(W^k H^k)_{ir}}}{\sum_r H_{jr}^{k}} = W_{i,j}^{k} \frac{\sum_r H_{jr}^{k}}{\sum_r H_{jr}^{k}} = W_{i,j}^{k} \tag{3.64}$$

$$H_{i,j}^{k+1} = H_{i,j}^{k} \frac{\sum_r W_{ri}^{k+1} \frac{(W^{k+1} H^k)_{rj}}{(W^{k+1} H^k)_{rj}}}{\sum_r W_{ri}^{k+1}} = H_{i,j}^{k} \frac{\sum_r W_{ri}^{k+1}}{\sum_r W_{ri}^{k+1}} = H_{i,j}^{k} \tag{3.65}$$

3.8 Manifold Learning

A redução de dimensionalidade não linear envolve encontrar representações de baixa dimensão no espaço de alta dimensão. Esse problema surge naturalmente ao analisar dados como imagens hiperespectrais, faces humanoa, formas de onda de fala e caracteres manuscritos. Algoritmos previamente propostos, como análise de componentes principais, análise de componentes independentes e fatoração matricial não-negativa, geralmente falham em capturar a representação não linear oculta dos dados. Verificou-se que variedades (*manifolds*) são usadas para explicar como nossa percepção visual e aprendizado funcionam, reconhecendo a variabilidade de estímulos perceptivos e outros tipos de dados de alta dimensão (MURASE; NAYAR, 1995; MCCLURKIN; OPTICAN; GAWNE, 1991; SEUNG; LEE, 2000). Para apresentar algoritmos de aprendizado de variedades (*manifold learning*) para redução de dimensionalidade não linear, primeiro precisamos introduzir o problema, o que requer algumas definições preliminares. A questão é: do ponto de vista matemático, o que é uma variedade? Existem três definições que ajudam a responder a essa pergunta (CAYTON, 2005).

Definição 3.8.1 Um homeomorfismo é uma função contínua cuja inversa também é uma função contínua.

Dois espaços com um homeomorfismo entre eles são chamados homeomorfos e, do ponto de vista topológico, são os mesmos.

Definição 3.8.2 Uma variedade M de dimensão d é um conjunto que é localmente homeomórfico com R^d. Para cada $x \in M$, há uma vizinhança aberta em torno de x, N_x, e um homeomorfismo $f : N_x \to R^d$. Essas vizinhanças são chamadas de *patches* de coordenadas e o mapa é referido como sendo *chart* de coordenadas. A imagem dos *charts* de coordenadas é chamada de espaço de parâmetros.

As variedades são objetos conceituais amplamente estudados pelos matemáticos e desem-

penham um papel central em muitas partes da geometria e da física matemática moderna. Resumidamente, uma variedade é um espaço topológico que, localmente e próximo de cada ponto, se assemelha ao espaço Euclidiano. Por exemplo, toda superfície é uma variedade 2D.

Na abordagem de redução de dimensionalidade, o interesse é particularmente no caso em que M é um subconjunto de R^m (o espaço de entrada), onde $m >> d$, com d sendo a dimensionalidade intrínseca, ou seja, a variedade ficará em um espaço de alta dimensão (R^m), mas será homeomórfico com um espaço de baixa dimensão (R^d). Além disso, algoritmos de aprendizado de variedades exigem algumas restrições de suavidade (CAYTON, 2005).

Definição 3.8.3 Uma variedade suave ou diferenciável é uma variedade tal que cada *chart* de coordenadas é diferenciável com um inverso diferenciável (ou seja, cada *chart* de coordenadas é um difeomorfismo).

A imersão de uma variedade M em R^d é de fato um homeomorfismo suave de M para um subconjunto de R^d. Os algoritmos apresentados nesta seção procuram por imersões de um conjunto de dados discreto $X = [\vec{x}_1, \vec{x}_2, ..., \vec{x}_n]$, em que $\vec{x}_i \in R^m$, para um conjunto de pontos em outro subespaço de R^d. O exemplo clássico é o desdobramento do rolo suíço (*the unfold of the swiss roll*) . Figura 3.8 mostra o conjunto de dados rolo suíço clássico e como, apesar dos pontos terem originalmente três coordenadas, eles podem ser representados como um plano 2D pelo mapa de cores (*colormap*). Uma das características mais importantes nos algoritmos de aprendizado de variedades é que os pontos próximos uns dos outros na variedade devem permanecer próximos na representação de baixa dimensão. Evidentemente, métodos lineares como o PCA falhariam ao desdobrar os dados, ou seja, ao tentar encontrar uma representação de baixa dimensão significativa.

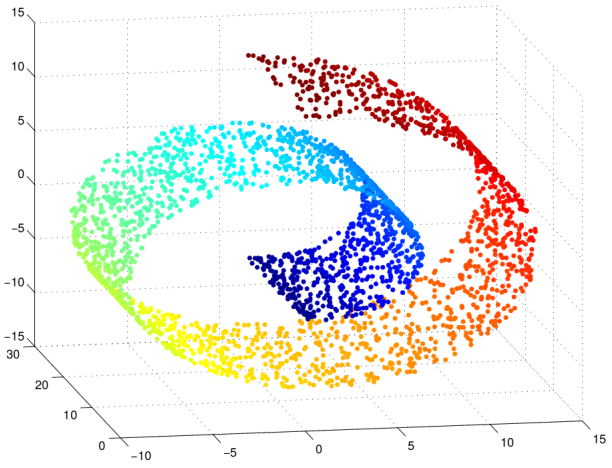


Figura 3.8: No conjunto de dados rolo suíço, cada amostra é originalmente 3D, mas com algoritmos de aprendizado de variedades, é possível expressar cada ponto usando apenas duas coordenadas, desdobrando os dados.

Em outras palavras, o objetivo é mapear a variedade. *Charting* é a tarefa de atribuir um sistema de coordenadas de baixa dimensão aos pontos de dados em um espaço de amostra de alta dimensão. As suposições são de que os dados estão dentro ou próximos de uma variedade de baixa dimensão e imersa no espaço amostral, e existe uma transformação não linear suave de um para um entre a variedade e o espaço vetorial de baixa dimensão (BRAND, 2003). De um modo mais formal, o problema de aprendizado de variedades pode ser descrito da seguinte maneira (CAYTON, 2005).

Definição 3.8.4 (Problema de Aprendizado de Variedades). Dados pontos $\vec{x}_1, \vec{x}_2, ..., \vec{x}_n \in R^m$ que estão dentro ou próximos de uma variedade M de dimensão d que pode ser descrito por um único *chart* de coordenadas $f : M \to R^d$, com $d < m$, encontre $\vec{y}_1, \vec{y}_2, ..., \vec{y}_n \in R^d$ em que $\vec{y}_i = f(\vec{x}_i)$ para $i = 1, 2, .., n$.

Nas próximas seções, apresentaremos uma revisão de três algoritmos para resolver o problema de aprendizado de variedades, com foco em como cada um deles alcança a solução que está sendo procurada.

3.8.1 Isometric Feature Mapping (ISOMAP)

ISOMAP foi um dos algoritmos pioneiros no aprendizado de variedades para redução de dimensionalidade. Os autores propõem uma abordagem que combina as principais características algorítmicas do PCA e do *Multidimensional Scaling* (MSD) (COX; COX, 2001; BORG; GROENEN, 2005) - eficiência computacional, otimização global e garantias de convergência assintótica - com a flexibilidade de aprender uma ampla gama de variedades não lineares (TENENBAUM; SILVA; LANGFORD, 2000). A idéia básica do algoritmo ISOMAP consiste em construir um grafo unindo os k vizinhos mais próximos (KNN) no espaço de entrada, computar os menores caminhos entre cada par de vértices e, conhecendo as distâncias geodésicas aproximadas entre os pontos, encontrar um mapeamento para um subespaço Euclidiano de R^d que preserve essas distâncias.

A hipótese do algoritmo ISOMAP é que os menores caminhos no grafo KNN são boas aproximações para as verdadeiras distâncias geodésicas na variedade. Foi demonstrado que, tanto para a regra do grafo ε-neighborhood (em que o parâmetro de entrada ε indica o raio que define a região de influência de cada vértice) quanto para a regra do grafo KNN (em que o parâmetro de entrada K indica o número de vizinhos de cada vértice), sob certas condições de regularidade, o resultado a seguir é válido (BERNSTEIN et al., 2000).

Teorema 3.8.1 (Teorema da Convergência Assintótica). Dados $\lambda_1, \lambda_2, \mu > 0$ então, para um

número suficientemente grande de amostras $\vec{x}_1, \vec{x}_2, ..., \vec{x}_n \in R^m$ a seguinte desigualidade:

$$(1-\lambda_1)d_M(\vec{x}_i,\vec{x}_j) \leq d_G(\vec{x}_i,\vec{x}_j) \leq (1+\lambda_2)d_M(\vec{x}_i,\vec{x}_j) \tag{3.66}$$

é satisfeita com probabilidade $(1-\mu)$, onde $d_G(\vec{x}_i,\vec{x}_j)$ é a aproximação do menor caminho no grafo e $d_M(\vec{x}_i,\vec{x}_j)$ é a distância geodésica na variedade.

O algoritmo ISOMAP pode ser dividido em 3 grandes passos:

1. Induzir um grafo a partir do conjunto de dados de entrada $\vec{x}_1, \vec{x}_2, ..., \vec{x}_n \in R^m$ usando a regra do KNN ou a regra do ε-neighborhood (LUXBURG, 2007);

2. Montar a matriz de distâncias ponto a ponto D usando n execuções do algoritmo Dijkstra ou uma execução do algoritmo Floyd-Warshall (CORMEN et al., 2009);

3. De posse da matriz D, encontrar um conjunto de pontos no subespaço Euclideano do R^d tal que as distâncias sejam preservadas por meio de uso do algoritmo MDS (*Multidimensional Scaling*).

Fica claro pelo algoritmo que a imersão é criada pelo método MDS, que desempenha um papel central no ISOMAP. Por esse motivo, a seguir, fornecemos uma descrição matemática detalhada do MDS, mostrando como ele recupera os pontos $\vec{y}_1, \vec{y}_2, ..., \vec{y}_n \in R^d$ a partir da matriz de distância D obtida no passo dois.

3.8.1.1 Multidimensional Scaling (MDS)

Basicamente, o objetivo principal do MDS é, dada uma matriz de distâncias par a par de dimensão $n \times n$, recuperar quem são as coordenadas de n pontos $\vec{x}_r \in R^d$ para $r = 1, 2, ..., n$ no subespaço Euclideano, onde d, a dimensionalidade alvo, é o parâmetro do algoritmo (COX; COX, 2001; BORG; GROENEN, 2005).

Começamos observando que a matriz de distâncias par a par é dada por $D = \{d_{rs}^2\}$, para $r, s = 1, 2, ..., n$ onde a distância entre dois pontos arbitrários $\vec{x}_r$ e $\vec{x}_s$ é:

$$d_{rs}^2 = \|\vec{x}_r - \vec{x}_s\|^2 = (\vec{x}_r - \vec{x}_s)^T(\vec{x}_r - \vec{x}_s) \tag{3.67}$$

Vamos denotar a matriz de produtos internos como $B = \{b_{rs}\}$, onde $b_{rs} = \vec{x}_r^T \vec{x}_s$. Para encontrar a imersão, o MDS precisa da matriz B, não D. Sabendo disso, podemos levantar duas questões relevantes:

1. Como encontrar a matriz B a partir da matriz de distância D?

2. Como recuperar as coordenadas dos pontos a partir de B?

A seguir, fornecemos respostas para ambas as perguntas.

3.8.1.2 Encontrar a matriz B

Ao responder a primeira pergunta, precisamos assumir a hipótese de que a média dos dados é nula (pontos estão ao redor do vetor nulo), ou seja:

$$\sum_{r=1}^{n} \vec{x}_r = 0 \tag{3.68}$$

caso contrário, haveria infinitas soluções diferentes, uma vez que a aplicação de qualquer translação arbitrária no conjunto de pontos preservaria as distâncias par a par.

A partir da equação (3.67), aplicando a lei da distributividade temos:

$$d_{rs}^2 = \vec{x}_r^T \vec{x}_r + \vec{x}_s^T \vec{x}_s - 2\vec{x}_r^T \vec{x}_s \tag{3.69}$$

A partir da matriz D, podemos calcular a média de uma coluna arbitrária s por:

$$\begin{aligned}
\frac{1}{n}\sum_{r=1}^{n} d_{rs}^2 &= \frac{1}{n}\sum_{r=1}^{n} \vec{x}_r^T \vec{x}_r + \frac{1}{n}\sum_{r=1}^{n} \vec{x}_s^T \vec{x}_s - 2\frac{1}{n}\sum_{r=1}^{n} \vec{x}_r^T \vec{x}_s \\
&= \frac{1}{n}\sum_{r=1}^{n} \vec{x}_r^T \vec{x}_r + \frac{n}{n}\vec{x}_s^T \vec{x}_s - 2\vec{x}_s \frac{1}{n}\sum_{r=1}^{n} \vec{x}_r^T \\
&= \frac{1}{n}\sum_{r=1}^{n} \vec{x}_r^T \vec{x}_r + \vec{x}_s^T \vec{x}_s
\end{aligned} \tag{3.70}$$

Analogamente, podemos calcular a média de uma linha arbitrária r como:

$$\begin{aligned}
\frac{1}{n}\sum_{s=1}^{n} d_{rs}^2 &= \frac{1}{n}\sum_{s=1}^{n} \vec{x}_r^T \vec{x}_r + \frac{1}{n}\sum_{s=1}^{n} \vec{x}_s^T \vec{x}_s - 2\frac{1}{n}\sum_{s=1}^{n} \vec{x}_r^T \vec{x}_s \\
&= \frac{n}{n}\vec{x}_r^T \vec{x}_r + \frac{1}{n}\sum_{s=1}^{n} \vec{x}_s^T \vec{x}_s - 2\vec{x}_r^T \frac{1}{n}\sum_{s=1}^{n} \vec{x}_s \\
&= \vec{x}_r^T \vec{x}_r + \frac{1}{n}\sum_{s=1}^{n} \vec{x}_s^T \vec{x}_s
\end{aligned} \tag{3.71}$$

Finalmente, podemos calcular a média de todos os elementos de D como:

$$\begin{aligned}\frac{1}{n^2}\sum_{r=1}^{n}\sum_{s=1}^{n} d_{rs}^2 &= \frac{1}{n^2}\sum_{r=1}^{n}\sum_{s=1}^{n} \vec{x}_r^T\vec{x}_r + \frac{1}{n^2}\sum_{r=1}^{n}\sum_{s=1}^{n} \vec{x}_s^T\vec{x}_s - 2\frac{1}{n^2}\sum_{r=1}^{n}\sum_{s=1}^{n} \vec{x}_r^T\vec{x}_s \\ &= \frac{1}{n}\sum_{r=1}^{n} \vec{x}_r^T\vec{x}_r + \frac{1}{n}\sum_{s=1}^{n} \vec{x}_s^T\vec{x}_s = \frac{2}{n}\sum_{r=1}^{n} \vec{x}_r^T\vec{x}_r \end{aligned} \tag{3.72}$$

Note que a partir da equação (3.69), é possível definir b_{rs} como:

$$b_{rs} = \vec{x}_r^T\vec{x}_s = -\frac{1}{2}(d_{rs}^2 - \vec{x}_r^T\vec{x}_r - \vec{x}_s^T\vec{x}_s) \tag{3.73}$$

Mas a partir da equação (3.71) podemos isolar o termo $-\vec{x}_r^T\vec{x}_r$ como:

$$-\vec{x}_r^T\vec{x}_r = -\frac{1}{n}\sum_{s=1}^{n} d_{rs}^2 + \frac{1}{n}\sum_{s=1}^{n} \vec{x}_s^T\vec{x}_s \tag{3.74}$$

E a partir da equação (3.70) podemos isolar o termo $-\vec{x}_s^T\vec{x}_s$ como:

$$-\vec{x}_s^T\vec{x}_s = -\frac{1}{n}\sum_{r=1}^{n} d_{rs}^2 + \frac{1}{n}\sum_{r=1}^{n} \vec{x}_r^T\vec{x}_r \tag{3.75}$$

Agora fazendo equação (3.74) menos equação (3.75) resulta em:

$$-\vec{x}_r^T\vec{x}_r - \vec{x}_s^T\vec{x}_s = -\frac{1}{n}\sum_{r=1}^{n} d_{rs}^2 - \frac{1}{n}\sum_{s=1}^{n} d_{rs}^2 + \frac{2}{n}\sum_{r=1}^{n} \vec{x}_r^T\vec{x}_r \tag{3.76}$$

A partir da equação (3.72) sabemos que:

$$\frac{2}{n}\sum_{r=1}^{n} \vec{x}_r^T\vec{x}_r = \frac{1}{n^2}\sum_{r=1}^{n}\sum_{s=1}^{n} d_{rs}^2 \tag{3.77}$$

Finalmente, podemos expressar um arbitrário b_{rs} como função dos elementos da matriz de distância par a par D como:

$$b_{rs} = -\frac{1}{2}\left(d_{rs}^2 - \frac{1}{n}\sum_{r=1}^{n} d_{rs}^2 - \frac{1}{n}\sum_{s=1}^{n} d_{rs}^2 + \frac{1}{n^2}\sum_{r=1}^{n}\sum_{s=1}^{n} d_{rs}^2\right) \tag{3.78}$$

Fazendo $a_{rs} = -\frac{1}{2}d_{rs}$ podemos escrever:

$$a_{r.} = \frac{1}{n}\sum_{s=1}^{n} a_{rs} \qquad a_{.s} = \frac{1}{n}\sum_{r=1}^{n} a_{rs} \qquad a_{..} = \frac{1}{n}\sum_{r=1}^{n}\sum_{s=1}^{n} a_{rs} \tag{3.79}$$

Assim, podemos expressar b_{rs} como:

$$b_{rs} = a_{rs} - a_{r.} - a_{.s} + a_{..} \tag{3.80}$$

Definindo a matriz $A = \{a_{rs}\}$, para $r,s = 1,2,...,n$ como $A = -\frac{1}{2}D$ e a matriz H como:

$$H = I - \frac{1}{n}\vec{1}\vec{1}^T \tag{3.81}$$

onde $\vec{1}^T = [1,1,...,1]_n$ então temos:

$$\vec{1}\vec{1}^T = U = \begin{bmatrix} 1 & 1 & 1 & ... & 1 \\ 1 & 1 & 1 & ... & 1 \\ . & . & . & ... & . \\ 1 & 1 & 1 & ... & 1 \end{bmatrix}_{n\times n} \tag{3.82}$$

é possível calcular todos os valores da matriz B simultaneamente por $B = HAH$. Para ver isso, observe que:

$$B = HAH = \left(I - \frac{1}{n}U\right)A\left(I - \frac{1}{n}U\right) = A - A\frac{U}{n} - \frac{U}{n}A + \frac{1}{n^2}UAU \tag{3.83}$$

que é a forma matricial da equação (3.80).

3.8.1.3 Recuperar as coordenadas dos pontos

Neste ponto, precisamos encontrar a imersão, ou seja, as coordenadas dos pontos em R^d. Primeiro, note que a matriz B dos produtos internos pode ser expressa por:

$$B_{n\times n} = X^T_{n\times m} X_{m\times n} \tag{3.84}$$

onde n é denota número de amostras e m denota o número de dimensões de entrada e $X_{n\times m} = [\vec{x}_1, \vec{x}_2, ..., \vec{x}_n]$ é a matriz de dados. A matriz B possui 3 propriedades importantes (COX; COX, 2001):

- É simétrica

- O rank de B é m (número de linhas/colunas linearmentre independentes: gera uma base em R^m)
- É positiva semi-definida: $\forall \vec{x} \in R^n, \vec{x}^T B \vec{x} \geq 0$.

Em outras palavras, isso implica em dizer que a matrix B possui m autovalores não negativos e $n-m$ autovalores nulos. Assim, pela decomposição espectral de B (*eigendecomposition*) pode-se escrever:

$$B = V \Lambda V^T \tag{3.85}$$

onde $\Lambda = diag(\lambda_1, \lambda_2, ..., \lambda_n)$ é a matriz diagonal dos autovalores de B e V é a matriz cujas colunas são os autovetores de B:

$$V = \begin{bmatrix} | & | & ... & ... & | \\ | & | & ... & ... & | \\ \vec{v}_1 & \vec{v}_2 & ... & ... & \vec{v}_n \\ | & | & ... & ... & | \\ | & | & ... & ... & | \end{bmatrix}_{n \times n} \tag{3.86}$$

Sem perda de generalidade iremos considerar $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_n$. Devido aos $n-m$ autovalores nulos, B pode ser escrita como:

$$B = \tilde{V} \tilde{\Lambda} \tilde{V}^T \tag{3.87}$$

onde $\tilde{\Lambda} = diag(\lambda_1, \lambda_2, ..., \lambda_m)$ é a matriz diagonal dos autovalores não-negativos de B e $\tilde{V}$ é a matriz $n \times m$ cujas colunas são os m autovetores associados aos m autovalores não-negativos:

$$\tilde{V} = \begin{bmatrix} | & | & ... & ... & | \\ | & | & ... & ... & | \\ \vec{v}_1 & \vec{v}_2 & ... & ... & \vec{v}_m \\ | & | & ... & ... & | \\ | & | & ... & ... & | \end{bmatrix}_{n \times m} \tag{3.88}$$

Agora temos a seguinte identidade em relação à matriz B:

$$B = X^T X = \tilde{V} \tilde{\Lambda} \tilde{V}^T = \tilde{V} \tilde{\Lambda}^{1/2} \tilde{\Lambda}^{1/2} \tilde{V}^T \tag{3.89}$$

o que finalmente significa que:

$$X = \tilde{\Lambda}^{1/2}\tilde{V}^T \tag{3.90}$$

onde $\tilde{\Lambda}^{1/2} = diag(\sqrt{\lambda_1}, \sqrt{\lambda_2}, ..., \sqrt{\lambda_m})$. Na prática, escolhemos a dimensionalidade intrínseca, d, um parâmetro do algoritmo, para ser menor que a dimensionalidade dos dados de entrada, m, para que cada coluna de X represente uma amostra na variedade. Normalmente, optamos por compor a matriz $\tilde{V}$ com $d < m$ autovetores associados aos d maiores autovalores. Portanto, o algoritmo retorna uma matriz de dados de dimensão $d \times n$ que é uma representação mais compacta dos dados de entrada. O algoritmo 2 resume todo o processo em uma sequência de etapas lógicas e objetivas.

Algorithm 2 Isometric Feature Mapping

1: **function** ISOMAP(X)
2: A partir dos dados de entrada $X_{m\times n}$ construir o grafo KNN.
3: Calcular a matriz de distância par a par $D_{n\times n}$.
4: Calcular $A = -\frac{1}{2}D$.
5: Calcular $H = I - \frac{1}{n}U$, onde U é a matriz $n \times n$ de 1's.
6: Calcular $B = HAH$.
7: Encontrar os autovalores e autovetores da matriz B.
8: Tomar os K autovetores associados aos K maiores autovalores de B e definir:

$$\tilde{V} = \begin{bmatrix} | & | & ... & ... & | \\ | & | & ... & ... & | \\ \vec{v}_1 & \vec{v}_2 & ... & ... & \vec{v}_d \\ | & | & ... & ... & | \\ | & | & ... & ... & | \end{bmatrix}_{n\times d} \tag{3.91}$$

$$\tilde{\Lambda} = diag(\lambda_1, \lambda_2, ..., \lambda_d) \tag{3.92}$$

9: Calcular $\tilde{X} = \tilde{\Lambda}^{1/2}\tilde{V}^T$
10: **return** $\tilde{X}$
11: **end function**

O algoritmo ISOMAP não é perfeito e tem seus problemas. A conectividade de cada ponto de dados no gráfico KNN é definida como seus k vizinhos Euclidianos mais próximos no espaço de alta dimensão. Foi demonstrado que o ISOMAP é vulnerável a "erros de curto-circuito"se k for muito grande em relação à estrutura da variedade ou se o ruído nos dados afastar os pon-

tos levemente da variedade (BALASUBRAMANIAN; SCHWARTZ, 2002). Mesmo um erro de curto-circuito simples pode alterar muitas entradas na matriz de distância geodésica D, o que pode levar a uma imersão de baixa dimensão totalmente diferente. Por outro lado, se k for muito pequeno, o grafo da vizinhança pode se tornar muito esparso para aproximar os caminhos geodésicos com precisão. Ao mesmo tempo melhorias têm sido feitas nesse algoritmo para fazê-lo funcionar melhor em conjuntos de dados esparsos e ruídosos (SAXENA; GUPTA; MUKERJEE, 2004). Extensões do algoritmo ISOMAP incluem o kernel ISOMAP, cuja principal ideia é tornar a matriz $B = HAH$ em *Mercer kernel matrix* (em alguns casos, esse cálculo leva a uma matriz não positiva semi-definida) (CHOI; CHOI, 2007), L-ISOMAP ou Landmark ISOMAP, uma variante mais rápida do algoritmo (SILVA; TENENBAUM, 2002) e C-ISOMAP, que envolve amplificar as regiões de alta densidade e atenuar as regiões de baixa densidade dos pontos de dados na variedade (SILVA; TENENBAUM, 2002).

3.8.2 Locally Linear Embedding

O algoritmo ISOMAP é um método global no sentido de que, para encontrar as coordenadas de um determinado vetor de entrada $\vec{x}_i \in R^m$ na variedade, ele usa informações de todas as amostras através da matriz B. Por outro lado, a Imersão Linear Local (LLE), como o nome enfatiza, é um método local, ou seja, as novas coordenadas de qualquer $\vec{x}_i \in R^m$ dependem apenas da vizinhança daquele ponto. A principal hipótese por trás do LLE é que para uma densidade alta de amostras é esperado que um vetor $\vec{x}_i$ e seus vizinhos definam um patch linear, ou seja, pertençam a um mesmo subespacço Euclideano (ROWEIS; SAUL, 2000). Dessa forma, é possível caracterizar a geometria local por coeficientes lineares:

$$\hat{\vec{x}}_i \approx \sum_j w_{ij}\vec{x}_j \qquad \text{para} \qquad \vec{x}_j \in N(\vec{x}_i) \tag{3.93}$$

ou seja, podemos reconstruir um vetor como uma combinação linear dos seus vizinhos.

Basicamente, o algoritmo LLE requer como entrada uma matriz de dados X de dimensão $n \times m$, com linhas $\vec{x}_i$, um número desejado de dimensões $d < m$ e um número inteiro $k > d+1$ para encontrar vizinhanças locais. A saída é uma matriz Y de dimensão $n \times d$, com linhas $\vec{y}_i$. O algoritmo LLE pode ser dividido em três principais etapas (ROWEIS; SAUL, 2000; SAUL; ROWEIS, 2003):

1. A partir de cada $\vec{x}_i \in R^m$ encontre seus k vizinhos mais próximos;

2. Encontre a matriz de pesos W que minimiza o erro de reconstrução para cada ponto

$\vec{x}_i \in R^m$;

$$E(W) = \sum_{i=1}^{n} \left\| \vec{x}_i - \sum_j w_{ij} \vec{x}_j \right\|^2 \tag{3.94}$$

onde $w_{ij} = 0$ a menos que $\vec{x}_j$ seja um dos vizinhos mais próximos de $\vec{x}_i$ e para cada i, $\sum_j w_{ij} = 1$.

3. Encontre as coordenadas Y que minimizam o erro de reconstrução usando os pesos ideais;

$$\Phi(Y) = \sum_{i=1}^{n} \left\| \vec{y}_i - \sum_j w_{ij} \vec{y}_j \right\|^2 \tag{3.95}$$

sujeito a restrições de que $\sum_i Y_{ij} = 0$ para cada j, e que $Y^T Y = I$.

A seguir, descrevemos como obter a solução para cada etapa do LLE.

3.8.2.1 Encontrando Vizinhanças Lineares Locais

No algoritmo LLE usual, um número fixo de vizinhos mais próximos é definido para cada amostra através da distância Euclidiana simples. Outros critérios também podem ser usados para escolher os vizinhos, como escolher todos os pontos dentro de uma bola de raio fixo. Além disso, o número de vizinhos não precisa ser o mesmo para todos os pontos de dados. Regras alternativas também são possíveis, por exemplo, selecionando todos os pontos dentro de um determinado raio até um número máximo ou selecione um certo número de vizinhos, mas nenhum fora do raio máximo (SAUL; ROWEIS, 2003).

Vários critérios devem ser lembrados ao escolher k. Primeiro, é esperado que o algoritmo possa recuperar somente imersões cuja dimensionalidade, d, seja estritamente menor que k. Segundo, o LLE se baseia na suposição de que um ponto e seus vizinhos mais próximos são localmente lineares. Para conjuntos de dados curvos, escolher k muito grande, em geral, violará essa suposição. Finalmente, no caso incomum em que $k > m$, cada ponto pode ser reconstruído perfeitamente a partir de seus vizinhos, e os pesos de reconstrução locais não são mais definidos de maneira exclusiva. Nesse caso, alguma regularização adicional deve ser incluída para interromper a degeneração (SAUL; ROWEIS, 2003).

O algoritmo LLE também precisa verificar se o grafo KNN está conectado. Se o grafo estiver desconectado (ou fracamente conectado), o LLE deve ser aplicado separadamente aos dados em cada um dos componentes do grafo (fortemente) conectado; ou então a regra de seleção de vizinhança deve ser refinada para fornecer um grafo completamente conectado (SAUL; ROWEIS, 2003).

3.8.2.2 Estimativa dos Mínimos Quadrados dos Pesos

O segundo passo do LLE é reconstruir cada ponto a partir dos seus vizinhos mais próximos. Os pesos de reconstrução ideais podem ser calculados de forma fechada. Sem perda de generalidade, podemos expressar o erro de reconstrução local no ponto $\vec{x}_i$ como:

$$E(\vec{w}) = \left\| \sum_j w_j(\vec{x}_i - \vec{x}_j) \right\|^2 = \sum_j \sum_k w_j w_k (\vec{x}_i - \vec{x}_j)(\vec{x}_i - \vec{x}_k)^T \tag{3.96}$$

Definindo a matriz C como:

$$C_{jk} = (\vec{x}_i - \vec{x}_j)^T (\vec{x}_i - \vec{x}_k) \tag{3.97}$$

temos a expressão a seguir para o erro de reconstrução local:

$$E(\vec{w}) = \sum_j \sum_k w_j C_{jk} w_k = \vec{w}^T C \vec{w} \tag{3.98}$$

Sobre a restrição $\sum_j w_j = 1$, ela pode ser entendida de duas maneiras diferentes: geometricamente e probabilística. Do ponto de vista geométrico, fornece invariância na na translação, ou seja, adicionando qualquer vetor constante $\vec{c}$ a $\vec{x}_i$ e todos os seus vizinhos, o erro de reconstrução não é alterado. Vamos supor que $\tilde{\vec{x}}_i = \vec{x}_i + \vec{c}$ e $\tilde{\vec{x}}_j = \vec{x}_j + \vec{c}$. Assim, o novo erro de reconstrução local é dado por:

$$\begin{aligned} \tilde{E}(\vec{w}) &= \left\| \tilde{\vec{x}}_i - \sum_j w_j \tilde{\vec{x}}_j \right\|^2 = \left\| \vec{x}_i + \vec{c} - \sum_j w_j (\vec{x}_j + \vec{c}) \right\|^2 \\ &= \left\| \vec{x}_i + \vec{c} - \sum_j w_j \vec{x}_j - \sum_j w_j \vec{c} \right\|^2 = \left\| \vec{x}_i + \vec{c} - \sum_j w_j \vec{x}_j - \vec{c} \right\|^2 = E(\vec{w}) \end{aligned} \tag{3.99}$$

Em termos de probabilidade, forçar os pesos a somar um torna W uma matriz de transição estocástica (SAUL; ROWEIS, 2003), diretamente relacionada às cadeias de Markov e aos mapas de difusão.

Mostraremos que, na minimização do erro quadrático, a solução é encontrada através de um problema de autovalor. Na verdade, a estimativa da matriz W reduz para n problemas de autovalor: como não há restrições nas linhas de W, podemos encontrar os pesos ideais para cada amostra $\vec{x}_i$ separadamente, o que simplifica drasticamente os cálculos.

Assim, temos n problemas de otimização restrita independentes dados por:

$$\min_{\vec{w}_i} \vec{w}_i^T C_i \vec{w}_i \qquad \text{sujeito a} \quad \vec{1}^T \vec{w}_i = 1 \quad \text{para} \quad i = 1,2,...,n \tag{3.100}$$

Usando multiplicadores de Lagrange, escrevemos a função de Lagrange como:

$$L(\vec{w}_i, \lambda) = \vec{w}_i^T C_i \vec{w}_i - \lambda(\vec{1}^T \vec{w}_i - 1) \tag{3.101}$$

Tomando as derivadas em relação a $\vec{w}_i$:

$$\frac{\partial}{\partial \vec{w}_i} L(\vec{w}_i, \lambda) = 2C_i \vec{w}_i - \lambda \vec{1} = 0 \tag{3.102}$$

o que resulta em

$$C_i \vec{w}_i = \frac{\lambda}{2} \vec{1} \tag{3.103}$$

Se a matriz C_i for invertível, teremos uma solução de forma fechada:

$$\vec{w}_i = \frac{\lambda}{2} C_i^{-1} \vec{1} \tag{3.104}$$

onde λ pode ser ajustado para garantir $\sum_j w_i(j) = 1$. Na verdade, há uma expressão de forma fechada para $w_i(j)$ (ROWEIS; SAUL, 2000):

$$w_i(j) = \frac{\sum_k C_i^{-1}(j,k)}{\sum_k \sum_l C_i^{-1}(k,l)} \tag{3.105}$$

Para acelerar o algoritmo, em vez de calcular a matriz inversa C, é comum resolver o sistema linear:

$$C_i \vec{w}_i = \vec{1} \tag{3.106}$$

e então normalizar a solução para garantir que $sum_j w_i(j) = 1$ dividindo cada coeficiente do vetor $vecw_i$ pela soma de todos os coeficientes:

$$w_i(j) = \frac{w_i(j)}{\sum_j w_i(j)} \qquad \text{for} \qquad j = 1,2,...,m \tag{3.107}$$

Se k, o número de vizinhos, for maior que m, o número de características, então (em geral) o espaço gerado por k vetores distintos é o espaço completo. Isso significa que $\vec{x}_i$ pode ser escrito exatamente como uma combinação linear de seus k vizinhos mais próximos. De fato, se $k > m$, geralmente há infinitas soluções para $\vec{x}_i = \sum_j w_j \vec{x}_j$, porque há mais incógnitas (k) do que as equações (m). Nesse caso, o problema de otimização está incorreto e a regularização é necessária. Uma técnica comum de regularização é a regularização de Tikonov, que em vez de minimizar diretamente:

$$\left\| \vec{x}_i - \sum_j w_j \vec{x}_j \right\|^2 \tag{3.108}$$

adiciona um termo de penalização ao problema dos mínimos quadrados:

$$\left\| \vec{x}_i - \sum_j w_j \vec{x}_j \right\|^2 + \alpha \sum_j w_j^2 \tag{3.109}$$

onde α controla o grau de regularização. Em outras palavras, seleciona os pesos que minimizam a combinação de erro de reconstrução e a soma dos pesos ao quadrado. Como $\alpha \to 0$, temos o problema dos mínimos quadrados. No limite oposto, $\alpha \to \infty$, o termo erro quadrático se torna insignificante e queremos minimizar a norma Euclidiana do vetor de peso $\vec{w}$. Tipicamente, α é definido como um valor pequeno, mas diferente de zero. Nesse caso, os n problemas de otimização restrita e independentes são:

$$_{\vec{w}_i} \ \vec{w}_i^T C_i \vec{w}_i + \alpha \vec{w}_i^T \vec{w}_i \qquad \text{sujeito a} \quad \vec{1}^T \vec{w}_i = 1 \quad \text{para} \quad i = 1, 2, ..., n \tag{3.110}$$

A função de Lagrange é definida por:

$$L(\vec{w}_i, \lambda) = \vec{w}_i^T C_i \vec{w}_i + \alpha \vec{w}_i^T \vec{w}_i - \lambda (\vec{1}^T \vec{w}_i - 1) \tag{3.111}$$

Tomando a derivada em relação a $\vec{w}_i$ e configurando o resultado para zero:

$$2C_i \vec{w}_i + 2\alpha \vec{w}_i = \lambda \vec{1} \tag{3.112}$$

$$(C_i + \alpha I)\vec{w}_i = \frac{\lambda}{2} \vec{1} \tag{3.113}$$

$$\vec{w}_i = \frac{\lambda}{2} (C_i + \alpha I)^{-1} \vec{1} \tag{3.114}$$

onde λ é escolhido para normalizar adequadamente $\vec{w}_i$.

3.8.2.3 Encontrando as coordenadas

Se as vizinhanças locais forem pequenas o suficiente em comparação com a curvatura da variedade, os pesos ideais de reconstrução no espaço de imersão e os pesos de reconstrução na variedade são aproximadamente os mesmos (os dois conjuntos de pesos são exatamente iguais para subespaços lineares e, para variedades gerais, eles podem ser aproximados arbitrariamente um do outro, diminuindo suficientemente a vizinhança) (SHALIZI, 2009).

A idéia principal por trás da terceira etapa do algoritmo LLE é usar os pesos ideais de reconstrução estimados por mínimos quadrados como pesos adequados na variedade e encontrar as coordenadas da variedade local. Assim, fixando a matriz de pesos W, o objetivo é resolver outro problema de minimização quadrática para minimizar:

$$\Phi(Y) = \sum_{i=1}^{n} \left\| \vec{y}_i - \sum_j w_{ij} \vec{y}_j \right\|^2 \tag{3.115}$$

Em outras palavras, temos que responder à pergunta: quais são as coordenadas $\vec{y}_i \in R^d$ (aproximadamente na variedade), que os pesos W reconstroem?

Para evitar a degeneração, precisamos impor duas restrições:

1. A média dos dados no espaço transformado é zero, caso contrário, teríamos um número infinito de soluções;

2. A matriz de covariância dos dados transformados é a matriz de identidade, ou seja, não há correlação entre os componentes de $\vec{y} \in R^d$;

Contudo, improvável a estimativa dos pesos W, encontrar as coordenadas não se simplifica em n problemas independentes, porque cada linha de Y aparece em Φ várias vezes, uma vez como o vetor central y_i e novamente como um dos vizinhos de outros vetores.

Primeiro, reescreveremos a equação (3.115) de uma maneira mais significativa usando matrizes. Observe que:

$$\Phi(Y) = \sum_{i=1}^{n} \left[\left(\vec{y}_i - \sum_j w_{ij} \vec{y}_j \right)^T \left(\vec{y}_i - \sum_j w_{ij} \vec{y}_j \right) \right] \tag{3.116}$$

Aplicando a lei de distribuição, temos:

$$\Phi(Y) = \sum_{i=1}^{n} \left[\vec{y}_i^T \vec{y}_i - \vec{y}_i^T \left(\sum_j w_{ij} \vec{y}_j \right) - \left(\sum_j w_{ij} \vec{y}_j \right)^T \vec{y}_i + \left(\sum_j w_{ij} \vec{y}_j \right)^T \left(\sum_j w_{ij} \vec{y}_j \right) \right] \tag{3.117}$$

Expandindo a soma resulta em:

$$\Phi(Y) = \sum_{i=1}^{n} \vec{y}_i^T \vec{y}_i - \sum_{i=1}^{n} \sum_j \vec{y}_i^T w_{ij} \vec{y}_j - \sum_{i=1}^{n} \sum_j \vec{y}_j^T w_{ji} \vec{y}_i + \sum_{i=1}^{n} \sum_j \sum_k \vec{y}_j^T w_{ji} w_{ik} \vec{y}_k \tag{3.118}$$

Denotando por Y a matriz de dimensão $d \times n$ na qual cada coluna $\vec{y}_i$ para $i = 1, 2, ..., n$ armazena as coordenadas do i-ésimo ponto na variedade e sabendo que $\vec{w}_i(j) = 0$ a menos que $\vec{y}_j$ seja um dos vizinhos de $\vec{y}_i$, podemos escrever $\Phi(Y)$ como:

$$\begin{aligned}
\Phi(Y) &= Tr(Y^T Y) - Tr(Y^T W Y) - Tr(Y^T W^T Y) + Tr(Y^T W^T W Y) \\
&= Tr(Y^T Y) - Tr(Y^T (WY)) - Tr((WY)^T Y) + Tr((WY)^T (WY)) \\
&= Tr(Y^T (Y - WY) - (WY)^T (Y - WY)) \\
&= Tr((Y - WY)^T (Y - WY)) \\
&= Tr(((I - W)Y)^T ((I - W)Y)) \\
&= Tr(Y^T (I - W)^T (I - W) Y)
\end{aligned} \tag{3.119}$$

Defininndo a matriz M de dimensão $n \times n$ como:

$$M = (I - W)^T (I - W) \tag{3.120}$$

temos o problema de otimização a seguir:

$$_Y \; Tr(Y^T M Y) \qquad \text{sujeito a} \quad \frac{1}{n} Y^T Y = I \tag{3.121}$$

Assim, a função de Lagrange é dada por:

$$L(Y, \lambda) = Tr(Y^T M Y) - \lambda \left(\frac{1}{n} Y^T Y - I \right) \tag{3.122}$$

Derivando a função e igualando o resultado a zero dá:

$$2MY - 2\frac{\lambda}{n} Y = 0 \tag{3.123}$$

$$MY = \beta Y \tag{3.124}$$

onde $\beta = \frac{\lambda}{n}$, mostrando que o Y deve ser composto pelos autovetores da matriz M. Como temos um problema de minimização, queremos selecionar para compor Y os d autovetores associados aos menores d autovalores. Observe que M sendo uma matriz de dimensão $n \times n$, possui n autovalores e n autovetores ortogonais. Embora os autovalores sejam reais e não negativos, o menor deles é sempre zero, com o autovetor constante $\vec{1}$. Esse autovetor inferior corresponde à média de Y e deve ser descartado para forçar a restrição de que $\sum_{i=1}^{n} \vec{y}_i = 0$ (RIDDER; DUIN, 2002). Observe que cada linha de W deve somar um e, em seguida:

$$W\vec{1} = \vec{1} \tag{3.125}$$

$$\vec{1} - W\vec{1} = 0 \tag{3.126}$$

$$(I - W)\vec{1} = 0 \tag{3.127}$$

$$(I - W)^T (I - W)\vec{1} = 0 \tag{3.128}$$

$$M\vec{1} = 0 \tag{3.129}$$

Portanto, para obter $\vec{y}_i \in R^d$, onde $d < m$, devemos selecionar os $d + 1$ menores autovetores e descartar o autovetor constante com autovalor zero. Em outras palavras, devemos selecionar

os d autovetores associados aos autovalores inferiores não nulos.

O algoritmo LLE possui três parâmetros importantes: a dimensionalidade intrínseca d, o número de vizinhos mais próximos, k e, em alguns casos, o parâmetro de regularização α. A redução de dimensionalidade com o LLE é bastante sensível a variações nesses parâmetros. Se d estiver definido muito alto, o mapeamento aumentará o ruído; se estiver definido muito baixo, partes distintas do conjunto de dados podem ser mapeadas umas sobre as outras. Se k estiver definido muito pequeno, o mapeamento não refletirá nenhuma propriedade global; se for muito alto, o mapeamento perderá seu caráter não linear e se comportará como o PCA tradicional, pois todo o conjunto de dados é visto como vizinhança local. Por fim, se α estiver definido incorretamente, a análise espectral poderá não convergir (RIDDER; DUIN, 2002). O algoritmo 3 apresenta em grandes passos o método LLE. A entrada é uma matriz de dados X de dimensão $m \times n$ cujas colunas são as amostras e a saída é uma matriz Y de dimensão $n \times d$ cujas linhas representam as coordenadas dos pontos na variedade aprendida.

Algorithm 3 Locally Linear Embedding

function LLE(X, K, d)
  A partir dos dados de entrada $X_{m\times n}$ construir o grafo KNN.
  for $\vec{x}_i \in X^T$ **do**
    Calcular a matriz C_i de dimensão $K \times K$ como:

$$C_i(j,k) = (\vec{x}_i - \vec{x}_j)(\vec{x}_i - \vec{x}_k)^T \tag{3.130}$$

    Resolver o sistema linear $C_i\vec{w}_i = \vec{1}$ para estimar os pesos $\vec{w}_i \in R^K$.
    Normalize os pesos em $\vec{w}_i$ assim $\sum_j \vec{w}_i(j) = 1$.
  end for
  Construir a matriz W de dimensão $n \times n$, cujas as linhas são os estimados $\vec{w}_i$.
  Calcular $M = (I-W)^T(I-W)$.
  Encontrar os autovalores e os autovetores de M.
  Selecionar os d autovetores inferiores não nulos de M e define:

$$Y = \begin{bmatrix} | & | & \dots & \dots & | \\ | & | & \dots & \dots & | \\ \vec{v}_1 & \vec{v}_2 & \dots & \dots & \vec{v}_d \\ | & | & \dots & \dots & | \\ | & | & \dots & \dots & | \end{bmatrix}_{n\times d} \tag{3.131}$$

  return Y
end function

3.8.3 Laplacian Eigenmaps

A idéia básica por trás do método Laplacian Eigenmaps é que, se aproximarmos uma variedade por um grafo básico conectado e não direcionado, é possível encontrar um mapa dos vértices do grafo para um subespaço Euclidiano R^d, de modo que a localidade seja preservada ou, em outras palavras, o mapa seja suave no sentido de que os pontos vizinhos no grafo permanecerão próximos após a geração do mapeamento. Tal mapa é fornecido pelos autovetores do grafo da matriz Laplaciana (BO; YAN-RUI; XIAO-LONG, 2018). O mapa de representação gerado pelo algoritmo pode ser visto como uma aproximação discreta de um mapa contínuo que surge naturalmente da geometria da variedade: o operador Laplace-Beltrami (BELKIN; NIYOGI, 2003). Foi demonstrada a convergência dos autovetores do grafo Laplaciano associados a um conjunto de dados de nuvem de pontos para autofunções do operador Laplace-Beltrami quando os dados são amostrados a partir de uma distribuição de probabilidade uniforme em uma variedade imersa (BELKIN; NIYOGI, 2007). No aprendizado de máquina, o método Laplace Eigenmaps está intimamente relacionado ao agrupamento espectral, uma abordagem de classificação não supervisionada para agrupamento de dados (LUXBURG, 2007).

3.8.3.1 Visão Geral

Basicamente, o algoritmo Laplacian Eigenmaps requer como entrada uma matriz de dados X de dimensão $n \times m$, com cada linha $\vec{x}_i$ definindo um ponto de dados, um número desejado de dimensões $d < m$ e um numero inteiro k para encontrar vizinhanças locais. A saída é uma matriz Y de dimensão $n \times d$, com linhas $\vec{y}_i$. O algoritmo pode ser dividido em três principais etapas (BELKIN; NIYOGI, 2003):

1. Construa o grafo da vizinhança $G = (V,E)$ vinculando os nós v_i e v_j se $\vec{x}_i$ e $\vec{x}_j$ estiverem próximos. As duas variantes são:
   - ε-vizinhança: conecte v_i e v_j por uma aresta se $\left\|\vec{x}_i - \vec{x}_j\right\|^2 \leq \varepsilon$.
   - k-vizinhos mais próximos: conecte v_i e v_j por uma aresta se v_i estiver entre os k-vizinhos mais próximos de v_j ou v_j estiver entre os k-vizinhos mais próximos de v_i.
2. Escolha os pesos para definir a matriz de adjacência W. Existem também duas variações:
   - Núcleo de calor (*Heat kernel*) (com o parâmetro $t \in R$): se os nós v_i e v_j estiverem

conectados, faça

$$W_{ij} = exp\left\{-\frac{\left\|\vec{x}_i - \vec{x}_j\right\|^2}{t}\right\} \tag{3.132}$$

caso contrário, faça $W_{ij} = 0$. A justificativa para essa escolha é dada pela equação do calor.

- Pesos binários: faça $W_{ij} = 1$ se os nós v_i e v_j estiverem conectados por uma aresta e $W_{ij} = 0$ se v_i e v_j não estiverem conectados por uma aresta. Não há necessidade de escolher t.

3. Imersão: encontre as coordenadas Y escolhendo os d autovetores associados aos menores d autovalores não-nulos do grafo Laplaciano L.

A seguir, apresentamos as razões pelas quais o algoritmo Laplacian Eigenmaps é capaz de fornecer imersões ideais em termos de localidade e preservação da vizinhança.

3.8.3.2 Grafo Laplaciano e suas Propriedades

Seja $G = (V,E)$ um gráfico não direcionado com o conjunto de vértices $V = \{v_1, v_2, ..., v_n\}$. Consideramos que o gráfico é ponderado, ou seja, cada aresta entre v_i e v_j tem um peso não negativo $w_{ij} \geq 0$. Tipicamente, os pesos w_{ij} representam uma medida de similaridade ou uma distância entre pares de vetores $\vec{x}_i \in R^m$ e $\vec{x}_j \in R^m$.

Definição 3.8.5 A matriz de adjacência ponderada de um grafo não direcionado $G = (V,E)$ com $|V| = n$ é a matriz simétrica $W = \{w_{ij}\}$ para $i, j = 1, 2, ..., n$. Se $w_{ij} = 0$ os vértices v_i e v_j não estão conectados por uma aresta.

Definição 3.8.6 O grau de um vértice $v_i \in V$ é definido pela soma dos elementos da i-ésima linha de W:

$$d_i = \sum_{j=1}^{n} w_{ij} \tag{3.133}$$

A matriz de graus D é definida como a matriz diagonal com graus $d_1, d_2, ..., d_n$.

As matrizes Laplacianas e suas propriedades foram profundamente investigadas na teoria dos grafos espectrais, um campo de pesquisa muito maduro, cujo objetivo é o estudo de grafos em relação às características polinomiais, autovalores e autovetores de todos os tipos de matrizes associadas a um grafo. (CHUNG, 1997; BROUWER; HAEMERS, 2011; SPIELMAN, 2007; NICA, 2018; MIEGHEM, 2010).

Definição 3.8.7 A matriz do grafo Laplaciano não normalizada é definida por:

$$L = D - W \tag{3.134}$$

onde D é a matriz de graus e W é a matriz de adjacência.

A seguir, apresentamos algumas propriedades matemáticas básicas, mas muito importantes, do grafo Laplaciano (LUXBURG, 2007). Mais detalhes sobre o espectro Laplaciano e propriedades avançadas podem ser encontradas no artigo de Mohar (MOHAR, 1991).

Teorema 3.8.2 A matriz Laplaciano L satisfaz as seguintes propriedades:

1. Para cada vetor de coluna $\vec{f} \in R^n$ temos:

$$\vec{f}^T L \vec{f} = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} (f_i - f_j)^2 \tag{3.135}$$

2. L é simétrico e positivo semi-definido.

3. O menor autovalor de L é zero e o autovetor correspondente é o vetor constante $\vec{1}$

4. L tem n autovalores reais não negativos $0 = \lambda_1 \leq \lambda_2 \leq ... \leq \lambda_m$.

Para provar a primeira afirmação, observe que, pela definição de L e D, temos:

$$\begin{aligned}
\vec{f}^T L \vec{f} &= \vec{f}^T D \vec{f} - \vec{f}^T W \vec{f} \\
&= \sum_{i=1}^{n} d_i f_i^2 - \sum_{i=1}^{n} \sum_{j=1}^{n} f_i w_{ij} f_j \\
&= \frac{1}{2} \left(2 \sum_{i=1}^{n} d_i f_i^2 - 2 \sum_{i=1}^{n} \sum_{j=1}^{n} f_i w_{ij} f_j \right) \\
&= \frac{1}{2} \left(\sum_{i=1}^{n} d_i f_i^2 - 2 \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} f_i f_j + \sum_{i=1}^{n} d_j f_j^2 \right) \\
&= \frac{1}{2} \left(\sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} f_i^2 - 2 \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} f_i f_j + \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} f_j^2 \right) \\
&= \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} (f_i - f_j)^2
\end{aligned} \tag{3.136}$$

A segunda afirmação é dividida em duas partes: a primeira, sobre a simetria, segue diretamente da simetria das matrizes D e W; Na segunda parte, em relação à semidefinitividade positiva, fica claro que $(f_i - f_j)^2 \geq 0, \forall f_i, f_j \in R$, e porque $w_{ij} \geq 0$ para $i, j = 1, 2, ..., n$, $\vec{f}^T L \vec{f} \geq 0$.

Para provar a terceira afirmação, observe que:

$$L\vec{1} = (D-W)\vec{1} = D\vec{1} - W\vec{1} = \sum_{i=1}^{n} d_i - \sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij} = \sum_{i=1}^{n} d_i - \sum_{i=1}^{n} d_i = 0 \tag{3.137}$$

mostrando que o autovetor constante $\vec{1}$ tem zero autovalor.

Finalmente, a quarta afirmação é uma conseqüência direta das afirmações (2) e (3).

3.8.3.3 Imersão Laplaciana na Linha

Nesta seção, veremos que a imersão fornecida pelo Laplacian Eigenmaps é ótima em termos de preservação de informações locais, ou seja, os pontos vizinhos no grafo são próximos e os pontos distantes no grafo são distantes após a imersão. Suponha que tenhamos um grafo ponderado conectado $G = (V,E)$ cujos nós são os pontos de dados em $X = [\vec{x}_1, \vec{x}_2, ..., \vec{x}_n]$. O problema em questão pode ser formulado da seguinte maneira: como mapear os nós de G para uma linha para que os pontos conectados fiquem o mais próximos possível? O objetivo do Laplacian Eigenmaps é responder a essa pergunta motivadora.

Suponha que $\vec{y} = [y_1, y_2, ..., y_n]^T \in R^n$ seja o mapeamento dos vértices $v_1, v_2, ..., v_n$ para a linha real. Uma boa função objetiva deve penalizar fortemente os pontos vizinhos que são mapeados distantes. Uma escolha adequada para uma determinada matriz de adjacência W é a seguinte função:

$$J(\vec{y}) = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}(y_i - y_j)^2 = \vec{y}^T L\vec{y} \tag{3.138}$$

onde L é a matriz da Laplaciana. Observe que $J(\vec{y})$ é uma medida de quão dispersos os pontos estão distribuídos na linha real, portanto, minimizá-la é uma tentativa de garantir que se $\vec{x}_i$ e $\vec{x}_j$ estão próximos no espaço de entrada, então as coordenadas y_i e y_j também estão próximas na linha. Assim, podemos formular o problema de otimização restrita:

$$\min_{\vec{y}} y^T Ly \qquad \text{sujeito a} \quad y^T Dy = 1 \tag{3.139}$$

onde a restrição $y^T D = 1$ remove um fator de escala arbitrário na imersão (BELKIN; NIYOGI, 2003). Em outras palavras, estamos interessados na direção do vetor $\vec{y}$. Se não houvesse restrição, seria possível minimizar ainda mais a função objetiva, simplesmente dividindo os componentes de $\vec{y}$ por uma constante. Mais uma vez, escrevendo a função Lagrangiana, temos:

$$L(\vec{y},\lambda) = y^T Ly - \lambda(y^T Dy - 1) \tag{3.140}$$

Derivando em relação ao $\vec{y}$ e igualando o resultado a zero dá:

$$\frac{\partial}{\partial \vec{y}} L(\vec{y},\lambda) = 2L\vec{y} - 2\lambda D\vec{y} = 0 \tag{3.141}$$

o que resulta em:

$$L\vec{y} = \lambda D\vec{y} \tag{3.142}$$

$$(D^{-1}L)\vec{y} = \lambda\vec{y} \tag{3.143}$$

mostrando que temos um problema generalizado de autovetor. Por se tratar de um problema de minimização, temos que escolher o autovetor de $D^{-1}L$ associado ao menor autovalor. Como o autovetor constante $\vec{1}$ possui zero autovalor, devemos descartá-lo. Faz sentido que, para minimizar a dispersão dos pontos, todos eles sejam mapeados para a mesma coordenada. No entanto, esta solução trivial não tem uso prático. Portanto, $\vec{y}$ deve ser o autovetor associado ao menor autovalor diferente não-nulo, também conhecido como vetor de Fiedler (FIEDLER, 1989; CHUNG, 1997).

De fato, existem algumas variações baseadas no cálculo do grafo Laplaciano. A matriz $D^{-1}L$ é um dos Laplacianos normalizados. Outra maneira de calcular o Laplaciano normalizado é por $D^{-1/2}LD^{-1/2}$. Também é possível usar diretamente o Laplaciano não normalizado L, mas geralmente as versões normalizadas são preferidas devido a interessantes propriedades matemáticas. Foi demonstrado que, no agrupamento espectral, os Laplacianos normalizados têm conexões importantes com problemas de minimização de corte de grafo (*graph-cut*) (LUXBURG, 2007).

3.8.3.4 Imersão Laplaciana em R^d

Considere o problema generalizado de imersão do grafo $G = (V,E)$ em um espaço Euclidiano de dimensão d. Agora, cada nó $v_i \in V$ deve ser mapeado para um ponto em R^d, ou seja, precisamos estimar as coordenadas d para cada nó. Denotamos a imersão final por uma matriz $n \times d$ $Y = [\vec{y}_1, \vec{y}_2, ..., \vec{y}_d]$, onde i -a linha, $\vec{y}^{(i)}$, fornece as coordenadas de v_i na variedade. A função objetiva é generalizada para:

$$J(Y) = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} W_{ij}\left\|\vec{y}^{(i)} - \vec{y}^{(j)}\right\|^2 \tag{3.144}$$

onde $\vec{y}^{(i)} = [\vec{y}_1(i), \vec{y}_2(i), ..., \vec{y}_d(i)]$ é a representação de v_i tendo dimensão d. Observe que, considerando Y como uma matriz de dimensão $n \times d$ na qual cada linha representa um $\vec{y}^{(i)}$, para $i = 1, 2, ..., n$ reescrevemos a função objetiva como:

$$J(Y) = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} W_{ij}(\vec{y}^{(i)} - \vec{y}^{(j)})(\vec{y}^{(i)} - \vec{y}^{(j)})^T \tag{3.145}$$

Expandindo a expressão para $J(Y)$, podemos simplificá-la em:

$$\begin{aligned}
J(Y) &= \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\left[W_{ij}\vec{y}^{(i)}\vec{y}^{(i)^T} - W_{ij}\vec{y}^{(i)}\vec{y}^{(j)^T} - W_{ij}\vec{y}^{(j)}\vec{y}^{(i)^T} + W_{ij}\vec{y}^{(j)}\vec{y}^{(j)^T}\right] \\
&= \frac{1}{2}\left[\sum_{i=1}^{n} d_i\vec{y}^{(i)}\vec{y}^{(i)^T} - 2\sum_{i=1}^{n}\sum_{j=1}^{n} W_{ij}\vec{y}^{(i)}\vec{y}^{(j)^T} + \sum_{j=1}^{n} d_j\vec{y}^{(j)}\vec{y}^{(j)^T}\right] \\
&= \frac{1}{2}\left[2\sum_{i=1}^{n} d_i\vec{y}^{(i)}\vec{y}^{(i)^T} - 2\sum_{i=1}^{n}\sum_{j=1}^{n} W_{ij}\vec{y}^{(i)}\vec{y}^{(j)^T}\right] \\
&= \sum_{i=1}^{n} d_i\vec{y}^{(i)}\vec{y}^{(i)^T} - \sum_{i=1}^{n}\sum_{j=1}^{n} W_{ij}\vec{y}^{(i)}\vec{y}^{(j)^T}
\end{aligned} \tag{3.146}$$

Considerando $Y_{n\times d}$ a matriz das coordenadas de n pontos, $D_{n\times n}$ a matriz diagonal dos graus d_i e $W_{n\times n}$ a matriz de adjacência, podemos reescrever a equação usando uma notação de vetor de matrizes como:

$$J(Y) = Tr(DYY^T) - Tr(WYY^T) \tag{3.147}$$

Como o traço é um operador invariável sob permutações cíclicas, temos:

$$\begin{aligned} J(Y) &= Tr(Y^T DY) - Tr(Y^T WY) \\ &= Tr(Y^T DY - Y^T WY) \\ &= Tr(Y^T (DY - WY)) \\ &= Tr(Y^T (D - W)Y) \\ &= Tr(Y^T LY) \end{aligned} \tag{3.148}$$

Portanto, temos o seguinte problema de otimização restrita:

$$_Y \; Tr(Y^T LY) \qquad \text{sujeito a} \quad Y^T DY = I \tag{3.149}$$

cuja a função Lagrangiana é dada por:

$$L(Y, \lambda) = Tr(Y^T LY) - \lambda (Y^T DY - I) \tag{3.150}$$

Tomando a derivada e igualando o resultado a zero resulta em:

$$\frac{\partial}{\partial Y} L(Y, \lambda) = 2LY - 2\lambda DY = 0 \tag{3.151}$$

resultando em problema de autovetor a seguir:

$$LY = \lambda DY \tag{3.152}$$

Este resultado mostra que devemos selecionar para compor as colunas da matriz Y os d autovetores associados aos d menores autovalores não nulos dos autovalores de Laplaciano normalizado $D^{-1}L$. O algoritmo 4 mostra as principais etapas do método Laplacian Eigenmaps. A entrada é uma matriz de dados X de dimensão $m \times n$ cujas colunas são as amostras e a saída é uma matriz Y de dimensão $n \times d$ cujas linhas representam as coordenadas dos pontos na variedade aprendida.

Algorithm 4 Laplacian Eigenmaps

1: **function** LAPLACEEIGEN(X, K, d)

2: A partir dos dados de entrada $X_{m\times n}$ construir o grafo KNN.

3: Escolher os pesos para definir a matriz de adjacência W.

$$W_{ij} = exp\left\{-\frac{\left\|\vec{x}_i - \vec{x}_j\right\|^2}{t}\right\} \qquad \text{if} \qquad v_j \in N(v_i) \tag{3.153}$$

$$W_{ij} = 0 \qquad \text{if} \qquad v_j \notin N(v_i) \tag{3.154}$$

4: Calcular a matriz diagonal D com graus d_i para $i = 1, 2, ..., n$.

$$d_i = \sum_{j=1}^{n} W_{ij} \tag{3.155}$$

5: Calcular a matriz Laplaciana $L = D - W$

6: Selecionar os d menores autovetores com os autovalores não-nulos de $D^{-1}L$:

$$Y = \begin{bmatrix} | & | & \dots & \dots & | \\ | & | & \dots & \dots & | \\ \vec{v}_1 & \vec{v}_2 & \dots & \dots & \vec{v}_d \\ | & | & \dots & \dots & | \\ | & | & \dots & \dots & | \end{bmatrix}_{n\times d} \tag{3.156}$$

7: **return** Y

8: **end function**

Algumas variantes do algoritmo incluem a decomposição espectral de diferentes versões do grafo Laplaciano. As escolhas mais comuns são outra forma de Laplaciano normalizado, dada por $L_{sym} = D^{-1/2}LD^{-1/2}$, e o puro Laplace não normalizado $L = D - W$. Ao aplicar o Laplacian Eigenmaps a alguns dados do mundo real, várias limitações foram expostas, como amostragem desigual de dados, problema conhecido como fora da amostra, tamanho pequeno da amostra, extração e seleção de características discriminantes, etc. Para contornar esses problemas, um grande número de extensões ao Laplacian Eigenmaps têm sido propostas (BO; YAN-RUI; XIAO-LONG, 2018).

3.8.3.5 O operador Laplace-Beltrami

A motivação para o algoritmo Laplacian Eigenmaps é o papel do operador Laplace-Beltrami em variedades. Esta seção explica as razões pelas quais as autofunções desse operador têm

propriedades desejáveis para encontrar uma imersão na linha real, com base no artigo seminal de Belkin e Niyogi (BELKIN; NIYOGI, 2003). Considere uma variedade M de dimensão d imerso em R^m. Suponha que tenhamos um mapa $f : M \to R$ que atribua um valor real $f(\vec{x})$ a cada ponto $\vec{x}$ na variedade. O gradiente $\nabla f(\vec{x})$ é um campo vetorial que atribui um vetor para cada ponto de M, de modo que, para um pequeno deslocamento $\delta\vec{x}$, temos (BELKIN; NIYOGI, 2002b):

$$|f(\vec{x}+\delta\vec{x}) - f(\vec{x})| \approx |\langle \nabla f(\vec{x}), \delta\vec{x}\rangle| \tag{3.157}$$

A intuição por trás dessa aproximação é que, se o vetor de deslocamento $\delta\vec{x}$ for ortogonal ao gradiente e o produto escalar for zero, estaremos em uma curva de nível da função, portanto $|f(\vec{x}+\delta\vec{x}) - f(\vec{x})| = 0$. Assim, a variação em f é zero. Aplicando a desigualdade de Cauchy-Schwarz, temos:

$$|\langle \nabla f(\vec{x}), \delta\vec{x}\rangle| \leq \|\nabla f(\vec{x})\| \, \|\delta\vec{x}\| \tag{3.158}$$

resultando em:

$$|f(\vec{x}+\delta\vec{x}) - f(\vec{x})| \leq \|\nabla f(\vec{x})\| \, \|\delta\vec{x}\| \tag{3.159}$$

Esse resultado mostra que, se a norma do vetor gradiente for pequena, os pontos nas proximidades de um arbitrário $\vec{x} \in M$ serão mapeados próximos de $f(\vec{x})$. Em outras palavras, para garantir que o mapa f seja suave, ou seja, que o mapeamento seja, em média, o ideal em termos de preservação de localidades, procuramos resolver um problema de minimização:

$$_f \int_M \|\nabla f(\vec{x})\|^2 \qquad \text{com} \qquad \|f\| = 1 \tag{3.160}$$

No entanto, podemos reescrever o problema de minimização em termos do operador Laplaciano. Primeiro, observe que o Laplaciano é um operador diferencial que pode ser calculado pela divergente do gradiente:

$$L(f) = \text{div}(\nabla f) = \sum_{i=1}^{n} \frac{\partial^2 f}{\partial x_i^2} \tag{3.161}$$

onde o operador div transforma um campo vetorial em um campo escalar.

Segundo, foi demonstrado que os operadores divergentes e de gradiente são adjuntos, ou

seja, para uma função $f : M \to R$ e um campo vetorial X (BELKIN; NIYOGI, 2002b):

$$\int_M \langle X, \nabla f \rangle = \int_M \text{div}(X) f \tag{3.162}$$

Assim, a função objetiva no problema de minimização pode ser expressa por:

$$\int_M \|\nabla f(\vec{x})\|^2 = \int_M \langle \nabla f, \nabla f \rangle = \int_M \langle \text{div}(\nabla f), f \rangle = \int_M L(f) f \tag{3.163}$$

onde $L(f)$ é o operador Laplaciano. A solução para esse problema de minimização é conhecida por serem as autofunções do operador Laplaciano. Este resultado estabelece a conexão entre os autovetores da matriz Laplaciana e a construção de mapeamentos suaves e de preservação da localidade para o aprendizado de variedades.

3.8.3.6 A Matriz de Pesos e o Fluxo de Calor

Uma questão importante que permanece em aberto no método Laplacian Eigenmaps é: por que os pesos entre nós de vizinhança no grafo são calculados usando um kernel Gaussiano? Segundo os autores do algoritmo, há uma conexão entre o operador Laplace-Beltrami e o fluxo de calor (BELKIN; NIYOGI, 2003).

Seja $f : M \to R$ a distribuição inicial de calor e $u(\vec{x}, t)$ seja a distribuição de calor no tempo t. Então, fica claro que $u(\vec{x}, 0) = f(\vec{x})$. A equação do calor é uma equação diferencial parcial parabólica dada por:

$$\frac{\partial u}{\partial t} + Lu = 0 \tag{3.164}$$

onde L é o operador Laplaciano. Pode-se demonstrar que a solução dessa equação diferencial é a função:

$$u(\vec{x}, t) = \int_M H_t(\vec{x}, \vec{y}) f(\vec{y}) \tag{3.165}$$

onde $H_t(\vec{x}, \vec{y})$ é o núcleo de calor. Combinando as equações (3.164) e (3.165), podemos escrever:

$$Lf(\vec{x}) = Lu(\vec{x}, 0) = -\frac{\partial u}{\partial t} = -\left(\frac{\partial}{\partial t} \left[\int_M H_t(\vec{x}, \vec{y}) f(\vec{y}) \right] \right)_{t=0} \tag{3.166}$$

Foi demonstrado que H_t é aproximadamente Gaussiano:

$$H_t(\vec{x},\vec{y}) = (4\pi t)^{-d/2} \exp\left\{-\frac{\|\vec{x}-\vec{y}\|^2}{4t}\right\}(\phi(\vec{x},\vec{y}) + O(t)) \tag{3.167}$$

onde $\phi(\vec{x},\vec{y})$ é uma função suave com $\phi(\vec{x},\vec{x}) = 1$. Quando $\vec{x}$ e $\vec{y}$ estao próximos e t é pequeno, a equação (3.167) é simplificada em:

$$H_t(\vec{x},\vec{y}) \approx (4\pi t)^{-d/2} \exp\left\{-\frac{\|\vec{x}-\vec{y}\|^2}{4t}\right\} \tag{3.168}$$

A partir da equação (3.166), aplicando a definição de derivada:

$$Lf(\vec{x}) = -\lim_{\Delta t\to 0}\left[\frac{\int_M H_{t+\Delta t}(\vec{x},\vec{y})f(\vec{y}) - \int_M H_t(\vec{x},\vec{y})f(\vec{y})}{\Delta t}\right] \tag{3.169}$$

Para pequenos valores de t, temos $t \approx \Delta t$:

$$\begin{aligned} Lf(\vec{x}) &= -\lim_{t\to 0}\left[\frac{\int_M H_{t+\Delta t}(\vec{x},\vec{y})f(\vec{x}) - \int_M H_t(\vec{x},\vec{y})f(\vec{y})}{t}\right] \\ &\approx \frac{1}{t}\left[\lim_{t\to 0}\int_M H_t(\vec{x},\vec{y})f(\vec{y}) - \int_M H_{t+\Delta t}(\vec{x},\vec{y})f(\vec{y})\right] \end{aligned} \tag{3.170}$$

A medida que $t \to 0$, o núcleo de calor $H_t(\vec{x},\vec{y})$ tende à função delta de Dirac, e $\vec{x}$ tende a $\vec{y}$, isto é:

$$\lim_{t\to 0}\int_M H_t(\vec{x},\vec{y})f(\vec{y}) = f(\vec{x}) \tag{3.171}$$

o que resulta em:

$$\begin{aligned} Lf(\vec{x}) &\approx \frac{1}{t}\left[f(\vec{x}) - \int_M H_{\Delta t}(\vec{x},\vec{y})f(\vec{y})\right] \\ &\approx \frac{1}{t}\left[f(\vec{x}) - (4\pi\Delta t)^{(-d/2)}\int_M \exp\left\{-\frac{\|\vec{x}-\vec{y}\|^2}{4\Delta t}\right\}f(\vec{y})\right] \end{aligned} \tag{3.172}$$

Se $\vec{x}_1,\vec{x}_2,...,\vec{x}_n$ or uma amostra aleatória de pontos na variedade M, a expressão anterior poderá ser discretizada para:

$$Lf(\vec{x}_i) \approx \frac{1}{t}\left[f(\vec{x}_i) - \frac{1}{k}(4\pi t)^{-(d/2)} \sum_{0\leq\left\|\vec{x}_i-\vec{x}_j\right\|\leq\varepsilon} \exp\left\{-\frac{\left\|\vec{x}_i-\vec{x}_j\right\|^2}{4t}\right\} f(\vec{x}_j)\right] \tag{3.173}$$

onde o coeficiente $1/t$ é global e não afeta os autovetores do Laplaciano discreto. Como a dimensionalidade intrínseca d é frequentemente desconhecida, definimos:

$$\alpha = \frac{1}{k}(4\pi t)^{-(d/2)} \tag{3.174}$$

Sabendo que o resultado do operador Laplaciano aplicado à função constante é zero, segue-se que:

$$L\vec{1} = \left[1 - \alpha \sum_{\vec{x}_j\in\eta_i} \exp\left\{-\frac{\left\|\vec{x}_i-\vec{x}_j\right\|^2}{4t}\right\}\right] = 0 \tag{3.175}$$

onde $\vec{1} = [1,1,...,1]$ e η_i denota a vizinhança de $\vec{x}_i$. A partir disso, temos:

$$\alpha = \left(\sum_{\vec{x}_j\in\eta_i} \exp\left\{-\frac{\left\|\vec{x}_i-\vec{x}_j\right\|^2}{4t}\right\}\right)^{-1} \tag{3.176}$$

Esse resultado motiva a escolha dos pesos entre os nós vizinhos, de acordo com um núcleo Gaussiano no método Laplacian Eigenmaps.

3.9 Aprendizado Profundo

Atualmente, métodos de Aprendizado Profundo, amplamente conhecido como *Deep Learning* em termo inglês, são considerados como sendo estado-da-arte em muitos problemas de processamento de imagens e visão computacional, em particular problemas de classificação de imagens. Esta secão descreve alguns aspectos teóricos que embasam o uso de modelos profundos para a classificação de imagens, em geral.

3.9.1 Fundamentos

Deep Learning é uma classe especial de técnicas que utilizam a abordagem de aprendizado de representação. O aprendizado de representação consiste em um conjunto de métodos que permitem fornecer a uma máquina dados brutos (como os valores de pixel de uma imagem) como

entrada e descobrir automaticamente as representações necessárias para tarefas de detecção ou classificação. Em outras palavras, métodos baseados em *Deep Learning* buscam descobrir um modelo (regras, parâmetros, por exemplo) utilizando um conjunto de dados brutos (exemplos) e um método para guiar o aprendizado do modelo a partir desses exemplos (PONTI; COSTA, 2017). O resultado desse processo de aprendizado consiste em uma função que permita receber como entrada dados brutos e fornecer como saída uma representação adequada para o problema em questão, que pode ser tanto um problema de detecção como um de classificação.

O processo de aprendizado de representação utilizado nos métodos *Deep Learning* é baseado em algoritmos cujo objetivo é aprender vários níveis de representação dos dados a fim de modelar relações complexas entre eles. Representações de níveis superiores são, portanto, definidas em termos de representações de níveis inferiores, compondo módulos simples mas não lineares para a transformacão dos dados. O *Deep Learning* normalmente usa redes neurais artificiais, conhecidas em termo inglês como *Deep Neural Networks* (DNNs), o que seria uma rede neural comum com muitas camadas ocultas podendo conter cada uma um numero n de neurônios.

Em *Deep Learning* temos métodos que aprendem a função $f(.)$ por meio de composições de funções, isto é

$$f(x) = f_L(...f_2(f_1(x_1))...)),$$

em que cada vetor de dados x_l (associado a camada l) serve de entrada para cada função $f_l(.)$, já a saída gerada é o próximo vetor x_{l+1} que será passado para a próxima função.

Cada função f_l usa um conjunto de parâmetros W_l para realizar a transformação dos dados de entrada. Com o uso de W_l, a equação acima torna-se

$$f(x) = f_L(...f_2(f_1(x_1, W_1), W_2)...), W_l).$$

Na equação acima, temos a composição de L funções, ou L camadas.

É importante ressaltar que a ideia base do funcionamento dos métodos *Deep Learning* consiste em aprender sucessivas representações intermediárias dos dados, ou seja, os $x_l, l = 1, 2, ...l$. Além disso, os parâmetros W_l necessários para a execução de cada função f_l são aprendidos diretamente nos dados de entrada e cada representação é definida como combinações de outras mais simples (anteriores). Dessa forma, para se atingir um resultado desejado (de acordo com a

tarefa desejada) e preciso ter uma profundidade (em termos das representações sucessivas) adequada que permita aprender uma sequência de funções para transformar vetores e mapeá-los de um espaço a outro.

Nesse sentido, a noção da hierarquia das representações torna-se de extrema importancia em que cada função opera sobre uma entrada gerando uma representação que é então passada para a próxima função. O autor em (CHOLLET, 2017) estipula que uma das hipóteses do *Deep Learning* é que ao possuir um número suficiente de camadas L, espaços com dimensionalidade alta o suficiente, isto é, o número de parâmetros W em cada função, e dados suficientes para aprender os parâmetros W_l para todo l, então será possivel capturar o escopo das relações nos dados originais, encontrando assim a representação mais adequada para a tarefa desejada. Do ponto de vista geométrico, essa sequência de transformações ajuda a separar as múltiplas variedades (*manifolds*, em termo inglês) que estariam enoveladas nos dados originais.

Um dos modelos *Deep Learning* mais popular e, consequentemente, mais utilizado são as Redes Neurais Convolucionais conhecidas em termo inglês como *Convolutional Neural Networks* (CNNs), devido à incorporação, nas DNNs, das camadas convolucionais. Essas camadas tornam viável o processamento das entradas considerando campos receptivos locais, além de incluir operações conhecidas como *pooling*, responsáveis por reduzir a dimensionalidade espacial das representações. A seção a seguir apresenta uma breve introdução das principais operações envolvidas em CNNs.

3.9.2 Redes Convolucionais (CNNs)

CNNs têm o seu maior uso em aplicações envolvendo informações visuais como por exemplo, processamento de imagens e visão computacional. Uma das vantangens das CNNs em relação a outros modelos de rede *Deep Learning* é sua capacidade de capturar a estrutura espacial dos pixels da imagem de entrada por meio das suas camadas convolucionais. Essas estruturas permitem entender a relação entre os pixels vizinhos em uma determinada imagem.

3.9.2.1 Camada convolucional

Diferentemente de outras camadas, os neurônios contidos nas camadas convolucionais são filtros, isto é, matrizes de pesos aplicadas a uma imagem de entrada. Cada região da imagem, conhecida como campo receptivo local (*local receptive field*), é processada pelo filtro de tamanho $k \times k \times d$, em que k é a dimensão espacial do filtro e d a dimensão de profundidade (essa depende da entrada da camada, por exemplo d é igual a 3 quando a imagem de entrada

possui 3 canais, RGB). A saída da filtragem pode ser representada por $f_{l+1}(i,x,y)$ e consiste no pixel resultante da aplicação do filtro i na imagem vinda da camada anterior l, a partir dos valores da vizinhança centrados na posição (x,y). Outro aspecto importante para se mencionar é o passo ou *stride* cujo objetivo principal é otimizar o tempo de processamento pulando alguns pixels durante o processo de filtragem. Por exmplo, dada uma imagem de entrada de tamanho 64×64, filtramos todos os pixels e geramos uma nova imagem do mesmo tamanho, 64×64. Nesse caso, diga-se que a convolução foi realizada com passo/*stride* 1. Por outro lado, ao usar um *stride* igual a 2, teríamos como saída uma nova imagem com 32×32 de tamanho.

3.9.2.2 Mapas de características (*Feature maps*)

Um mapa de características é a saida/representação gerada por um filtro da camada convolucional. O conjunto de mapas de características gerados pelos n filtros da camada convolucional são empilhados, formando um tensor (utilizamos o termo tensor para denotar matrizes multidimensionais, com profundidade $d = 4$, por exemplo) cuja profundidade é igual a n, número de filtros. Esse tensor servirá de entrada para a próxima camada convolucional.

3.9.2.3 Pooling

Devido ao aumento da profundidade dos tensores, d, ao longo das camadas da rede, é conveniente reduzir a dimensão espacial dos mesmos para dimimuir o custo comptutacional. Essa redução em tamanho é chamada de *pooling* sendo a operação de máximo *maxpooling* comumente empregada. Outro benefício em aplicar a operação *pooling* está na análise multiresolução das imagens. Em outras palavras, reduzindo o tamanho das imagens obtemos um tipo de composição de banco de filtros multiresolução que processa imagens em diferentes espaços-escala. Alguns estudos na literatura (SPRINGENBERG et al., 2015) aconcelham o uso de stride durante as convoluções, no sentido de aumentá-la em vez de utilizar *pooling*.

3.9.2.4 Camadas completamente conectadas (*Fully connected layers (FC)*)

Uma camada FC é a camada em que cada neurônio possui um peso associado a cada elemento do vetor de entrada e, em CNNs, ela é posicionada depois de n camadas convolucionais. A entrada de uma camada FC é a versão vetorizada da saída da última camada convolucional da rede. Por exemplo, se a camada convolucional antes de uma camada FC gerar um tensor $4 \times 4 \times 40$, redimensionamos esses dados de forma que ele possua tamanho $1 \times (4 \times 4 \times 40) = 1 \times 640$. Assim, cada neurônio na camada FC deverá possuir 640 pesos de

forma a produzir uma combinação linear do vetor de entrada.

Atualmente, as arquiteturas mais populares utilizam camadas FC ocultas com função de ativação *ReLU*, e a camada de saída (classificador) com função de ativação *softmax*.

Capítulo 4

METODOLOGIA PROPOSTA

O objetivo deste trabalho de pesquisa foi propor novas abordagens para a concepção e desenvolvimento de novos descritores de textura baseados na teoria da informação e no campo aleatório Markoviano Gaussiano (GMRF); no aprendizado profundo e de variedades. Em primeiro lugar, decidiu-se definir cada subbanda da decomposição *wavelet* da imagem de textura como sendo um sistema complexo modelado por GMRF. A razão pela adoção desta ferramenta matemática, *wavelet*, é que a transformada *wavelet* é ideal para a análise de sinais não estacionários, isto é, sinais cujos componentes de frequência variam no tempo/espaço (possibilitando a análise tempo $\times$ frequência, algo que não é possível com a transformada de Fourier, por exemplo). Devido a linearidade da transformada wavelet, pode-se considerar a hipótese de que no domínio wavelet os dados são aproximadamente Gaussianos, o que facilita muito a tratabilidade matemática.

Já a inclusão de modelos de campos aleatórios Markovianos Gaussianos nessa proposta foi motivada por dois aspectos a seguir: (1) a maioria dos métodos estado da arte em análise de textura utiliza informação contextual na forma de *patches* extraídos da imagem (*patch-based approaches*, em termo inglês) e (2) optou-se também por investigar uma abordagem fortemente baseada em sistemas complexos, uma vez que existe uma analogia direta entre as subbandas dessa transformada e sistemas físicos de partículas, cenários em que o parâmetro temperatura desempenha um papel fundamental na definição da estrutura de dependência espacial (correlações entre elementos) (LEVADA, 2011).

Em segundo lugar, decidiu-se aplicar métodos de aprendizado de variedades para a redução de dimensionalidade não linear em problemas de classificação de texturas atuando especificamente como selecionadores de informações de textura relevantes. Nossa percepção é que técnicas de aprendizado de variedades são capazes de selecionar características discriminativas a partir de um espaço de alta dimensionalidade. Nesse sentido, faz-se necessário a criação de

espaços de padrões texturais de alta dimensionalidade em que serão aplicados esses algoritmos de aprendizado de variedades. Sabe-se que as características extraídas usando os descritores como GLCM, Haralick, HOG e LBP provaram ser discriminativas na classificação de padrões de textura.

Portanto, ao usar separadamente cada um desses quatro descritores para escolher um subconjunto de características que realmente representam a imagem de textura de entrada, algumas informações de textura podem ser perdidas, o que pode prejudicar o desempenho na tarefa de classificação. Neste trabalho, para compensar essa perda de informações texturais e preencher as lacunas deixadas por cada descritor, optou-se por combinar as características extraídas por esses quatro descritores formando assim um vetor de alta dimensionalidade que servirá de entrada para a tarefa de redução de dimensionalidade não linear. Com isso, terá-se como saída um vetor de características de menor dimensionalidade que proporcione uma melhoria no desempenho dos classificadores de interesse.

Por fim, optou-se por combinar, de uma forma elegante, o aprendizado profundo e de variedades. Para isso, foi proposta uma nova CNN progressiva e trainada do zero que consiste em extrair características de textura, seguida da modelagem de mapas, a partir do espaço de alta dimensionalidade gerado na etapa anterior, e que preserva distâncias geodésicas nas variedades (*manifolds*) dos dados. En termo prático, aplicou-se uma versão adaptada do DIMAL com a finalidade de selecionar, progressivamente, características na variedade, maximizando a dispersão de suas distâncias geodésicas entre pares. Como demonstrado no Capitulo 5, as tais características aprendidas são suficientemente discriminativas e podem aumentar bastante o poder de separabilidade de classes do classificador SVM.

4.1 Domínio

Uma das grandes limitações no processo de coleta de informação relevante a partir dos dados brutos baseia-se na adoção da hipótese de que a estrutura dimensional de tais conjuntos pode ser bem representada por um espaço Euclidiano R^n. Pesquisas têm mostrado cada vez mais que essa suposição é bastante fraca, no sentido de que na grande maioria dos casos, as métricas definidas no espaço Euclidiano ambiente (extrínsecas) não são adequadas para mensurar a similaridade entre instâncias dos conjuntos de dados, uma vez que a coleção de tais instâncias define uma variedade (*manifold*, em termo inglês) imersa no espaço ambiente. Em outras palavras, a estrutura dimensional intrínseca costuma ser muito menor que a dimensão do espaço ambiente. Além disso, a curvatura presente em tais espaços é uma característica mar-

cante. Nesse contexto, surge a área conhecida como aprendizado de variedades, ou *manifold learning*, em termo inglês, que tem como foco principal a utilização de técnicas para a imersão de espaços topológicos complexos em subespaços do R^n, não assumindo para isso nenhuma hipótese acerca da distribuição dos dados observados (métodos não paramétricos). Algoritmos de *manifold learning* incluem os métodos ISOMAP, LLE e Lap.Eig., dentre outros. Nessa linha de pensamento, surgem também métodos de aprendizado profundo (CNN's) que, devido à sua característica de não-linearidades presentes nas redes, podem ajudar nesse processo.

Numa outra vertente, foca-se em modelos paramétricos para os dados, ou seja, assume-se que as observações são ocorrências de uma ou mais variáveis aleatórias e portanto seguem modelos estatísticos bem definidos, como por exemplo, campos aleatórios. Nesse cenário, cada modelo com seus parâmetros especificados define um ponto no espaço paramétrico. O objetivo então consiste em extrair medidas e mensurar distâncias entre diferentes modelos nesses espaços paramétricos, com o intuito de quantificar a similaridade entre variáveis aleatórias, que nesse caso representam padrões não determinísticos a serem estudados. Pesquisas mostram que espaços paramétricos de modelos estatísticos definem variedades Riemannianas onde a métrica natural é dada pela matriz de informação de Fisher. Nesse contexto, pode-se estudar a caracterização e evolução de um sistema complexo a partir de deformações na estrutura geométrica de seu espaço paramétrico. Esse é justamente um dos objetos de estudo da área conhecida como geometria da informação, ou *information geometry* em inlgês, que tem como propósito quantificar propriedades intrínsecas de tais espaços paramétricos através do tensor métrico, dado pela matriz de informação de Fisher. A motivação consiste em caracterizar cada subbanda da decomposição *wavelet* de uma imagem de textura como um sistema complexo modelado por um campo aleatório Gaussiano Markoviano. A justificativa para a adoção dessa ferramenta matemática consiste no fato de que a transformada *wavelet* é ideal para a análise de sinais não estacionários, ou seja, sinais em que os componentes de frequência variam no tempo/espaço. Devido a linearidade da transformada *wavelet*, pode-se considerar a hipótese de que no domínio *wavelet* os dados são aproximadamente Gaussianos, o que simplifica sensivelmente a tratabilidade matemática.

Assim sendo, este trabalho de pesquisa tem por objetivo propor novas abordagens para extração de características a partir de imagens de texturas, combinando os conceitos de teoria da informação, aprendizado profundo e de variedades, tendo como prioridade construir descritores de texturas que possam manter e/ou melhorar o poder discriminativo de classificadores de imagens de texturas em geral.

Este capítulo está organizado da seguinte maneira: na seção 4.2.1 é apresentada uma pri-

meira abordagem baseada em teoria da informação em campos aleatórios Markovianos; na seção 4.2.2 é apresentada uma segunda abordagem baseada em aprendizado de variedades na redução de dimensionalidade não linear; na seção 4.2.3 é apresentada uma terceira abordagem baseada na combinação das Redes Neurais Convolucionais (CNN's) com a versão adaptada do *framework* DIMAL na extração de características de imagens de textura; e finalmente, são apresentados e discutidos os resultados experimentais obtidos no Capítulo 5.

4.2 Métodos Propostos

Nesta seção, serão descritas de maneira detalhada as abordagens propostas para atender os objetivos geral e específicos apresentados na seção 1.4.

4.2.1 Novo descritor baseado em teoria da informação em campos aleatórios Markovianos

Nesta abordagem foi proposta um descritor eficiente que combine a teoria da informação, *wavelets* e campos aleatórios Gaussianos Markovianos para a classificação de texturas. Esta proposta põe em destaque os seguintes pontos:

(a) Descritor baseado na teoria da informação para a classificação de textura usando *wavelets* e campos aleatórios Gaussianos Markovianos.

(b) Incorporação da geometria não Euclidiana por meio da matriz de informações de Fisher, que define o tensor métrico do modelo paramétrico subjacente.

(c) Combinação de características de *wavelets* e propriedades estatísticas definindo uma nova abordagem baseada em análise de sistemas complexos.

A seguir, apresentamos a descrição, em detalhes, de como essas idéias foram aplicadas para extrair características eficientes para a tarefa de classificação de textura.

De modo geral, o objetivo principal aqui consiste em cartacterizar a textura explorando as interações não linerares entre os coeficientes de cada subbanda *wavelet* sob o controle do parâmetro β, responsável pelo controle da estrutura de dependência espacial dada por GMRF. Nesse sentido, estipulamos que dada uma subbanda *wavelet* modelada por GMRF, cada pixel x_i da subbanda está associado aos seus vizinhos η_i por meio de $p(x_i|\eta_i,\theta)$ tal como definido em 3.5.2. Assim, cada variável aleatória $p_i, i = 1,2,...,n$, que nesse caso representa algum padrão

textural não determinístico sendo estudado, possui um padrão de configuração local $x_i \bigcup \eta_i$ de dimensão 3×3 devido a escolha de um sistema de vizinhança de segunda ordem (de tamanho $\Delta = 8$, as opções usuais de Δ são $4, 8, 12, 20, 24$, etc.).

4.2.1.1 Considerando como entrada uma imagem em ton de cinza

O esquema proposto na Figura 4.1 resume brevemente a idéia principal adotada nessa abordagem. O primeiro passo consistiu em decompor a imagem de textura de entrada em escala de cinza em subbandas *wavelet*. Depois disso, para cada subbanda, geramos um conjunto de dados com *patches* de dimensão $n \times n$ em que $n \times n$ é igual ao valor do maior número inteiro positivo $\leq$ ordem da vizinhanca $+1$. Assim, cada linha do conjunto de dados gerado é composta por nove elementos (devido a dimensão 3×3 dos *patches* extraídos). É extremamente importante enfatizar que dividimos cada subbanda *wavelet* modelada por GMRF em *patches* de $n \times n$ com o objetivo de realizar uma análise contextual. Após converter cada subbanda *wavelet* em um conjunto de dados de *patches*, calculamos seus componentes da matriz de informações de Fisher e a sua respectiva entropia de Shannon. Finalmente, após concatenar todos os vetores de características de todas as subbandas, criamos uma versão normalizada do vetor de características para servir como descritor da imagem de textura entrada.

Em termos práticos, pode-se computar os tais componentes da matriz de informação de Fisher a partir da matriz de covariância das variáveis aleatórias $p_i, i = 1, 2, ..., n$. Seja Σ_p a matriz de covariância dos vetores $\vec{p}_i, i = 1, 2, \ldots, n$, obtidos a partir da ordenação lexicográfica de padrões de configuração local $x_i \bigcup \eta_i$ de dimensão 3×3 devido a escolha de um sistema de vizinhança de segunda ordem. Assim, como cada vetor $\vec{p}_i$ possui 9 elementos, a matriz de covariância resultante Σ_p tem a dimensão igual a 9×9. Considera agora Σ_p^- como sendo a matriz 8×8 obtida removendo a linha central e a coluna central da matriz Σ_p (esses elementos são as covariâncias entre a variável central x_i e cada um de seus vizinhos $x_j \in \eta_i$). Além disso, seja $\vec{\rho}$ um vetor de dimensão 8×1 formado por todos os elementos da linha central do Σ_p, excluindo o elemento do meio (o que denota a variação de x_i). Figura 4.2 ilustra o processo de decomposição da matriz de covariância Σ_p na matriz Σ_p^- e o vetor $\vec{\rho}$ em um modelo GMRF isotrópico entre pares (conhecido em inglês como *an isotropic pairwise GMRF model*) definido em um sistema de vizinhança de segunda ordem (considerando os 8 vizinhos mais próximos).

Extração de Características : Primeiramente, a imagem de textura de entrada é decomposta em 10 subbandas *wavelet* usando DWT com nível 3. Esse conjunto de subbandas é representado por S na equação (4.1).

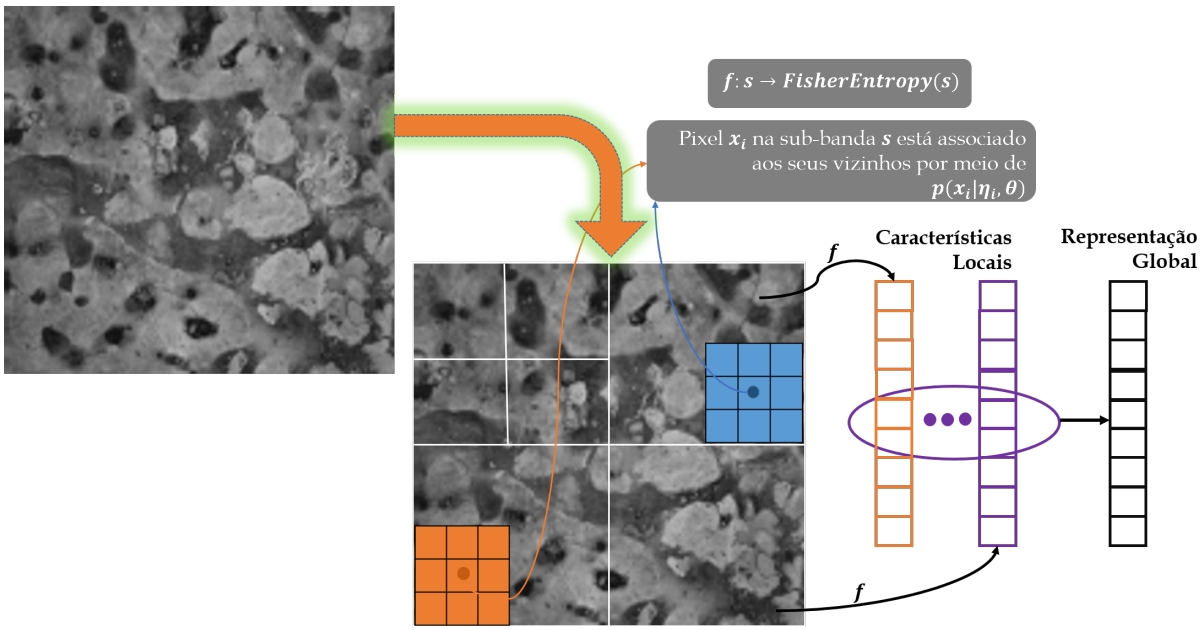


Figura 4.1: O objetivo da representação de textura é transformar a imagem de textura de entrada em um vetor de características que possam descrever as propriedades de tal textura, facilitando assim tarefas subseqüentes, como por exemplo, a classificação de textura. Normalmente, uma imagem de textura é primeiro transformada em um conjunto de características locais, que são então agregadas em uma representação global para toda a imagem ou apenas uma região de interesse.

$$S = \{LL_3, LH_3, HL_3, HH_3, LH_2, HL_2, HH_2, LH_1, HL_1, HH_1\} \tag{4.1}$$

Em seguida, a partir de uma subbanda modelada por GMRF, computamos os respectivos componentes de informação de Fisher e a entropia de Shannon, o que resulta no seguinte vetor de característica

$$V = \left[I^{(1)}_{\mu\mu}(\vec{\theta}), I^{(2)}_{\mu\mu}(\vec{\theta}), I^{(1)}_{\sigma^2\sigma^2}(\vec{\theta}), I^{(2)}_{\sigma^2\sigma^2}(\vec{\theta}), I^{(1)}_{\sigma^2\beta}(\vec{\theta}), I^{(2)}_{\sigma^2\beta}(\vec{\theta}), I^{(1)}_{\beta\beta}(\vec{\theta}), I^{(2)}_{\beta\beta}(\vec{\theta}), H_\beta\right] \tag{4.2}$$

onde H_β denota a entropia de Shannon. Como temos dez subbandas *wavelet*, o vetor de características final possui a seguinte forma

$$F = \left[V'_1 \circ V'_2 \circ V'_3 \circ V'_4 \circ V'_5 \circ V'_6 \circ V'_7 \circ V'_8 \circ V'_9 \circ V'_{10}\right] \tag{4.3}$$

onde $V'_j = {}^{V_j}/_{max(V_j)}$ é a versão normalizada de V_j, que, por sua vez, indica o vetor de características para a subbanda j-ésima e $\circ$ indica a concatenação vetorial.

As expressões para o cálculo do parâmetro β, dos componentes de informação de Fisher e da entropia de Shannon são dadas por

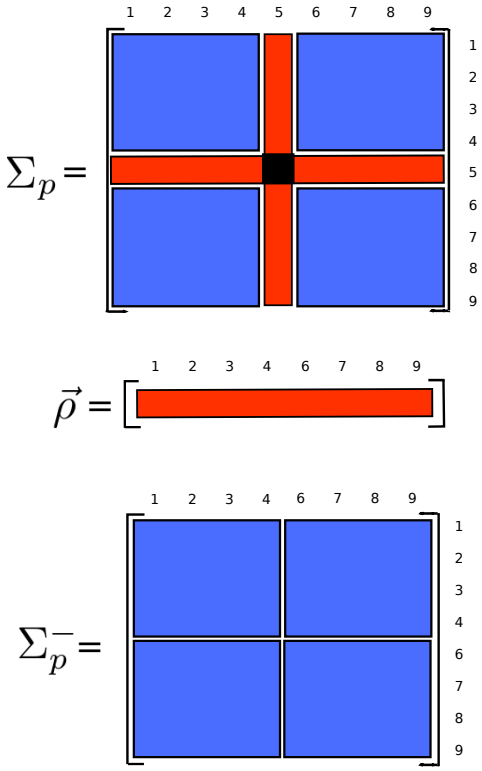


Figura 4.2: Decomposição da matriz de covariância Σ_p em Σ_p^- e $\vec{\rho}$ em um sistema de vizinhança de segunda ordem ($\Delta = 8$).

$$\beta = \frac{\|\vec{\rho}\|_+}{\|\Sigma_p^-\|_+} \tag{4.4}$$

$$I^{(1)}_{\mu\mu}(\vec{\theta}) = \frac{1}{\sigma^2}(1-\beta\Delta)^2\left[1-\frac{1}{\sigma^2}\left(2\beta\|\vec{\rho}\|_+ - \beta^2\left\|\Sigma_p^-\right\|_+\right)\right] \tag{4.5}$$

$$\begin{aligned} I^{(1)}_{\sigma^2\sigma^2}(\vec{\theta}) = \frac{1}{2\sigma^4} &- \frac{1}{\sigma^6}\left[2\beta\|\vec{\rho}\|_+ - \beta^2\left\|\Sigma_p^-\right\|_+\right] \\ &+ \frac{1}{\sigma^8}\left[3\beta^2\|\vec{\rho}\otimes\vec{\rho}\|_+ - 3\beta^3\left\|\vec{\rho}\otimes\Sigma_p^-\right\|_+ + 3\beta^4\left\|\Sigma_p^-\otimes\Sigma_p^-\right\|_+\right] \end{aligned} \tag{4.6}$$

$$\begin{aligned} I^{(1)}_{\sigma^2\beta}(\vec{\theta}) = I^{(1)}_{\beta\sigma^2}(\vec{\theta}) = \frac{1}{\sigma^4}&\left[\|\vec{\rho}\|_+ - \beta\left\|\Sigma_p^-\right\|\right] \\ &- \frac{1}{2\sigma^6}\left[6\beta\|\vec{\rho}\otimes\vec{\rho}\|_+ - 9\beta^2\left\|\vec{\rho}\otimes\Sigma_p^-\right\|_+ + 3\beta^3\left\|\Sigma_p^-\otimes\Sigma_p^-\right\|_+\right] \end{aligned} \tag{4.7}$$

$$I^{(1)}_{\beta\beta}(\vec{\theta}) = \frac{1}{\sigma^2}\left\|\Sigma_p^-\right\|_+ + \frac{1}{\sigma^4}\left[2\left\|\vec{\rho}\otimes\vec{\rho}\right\|_+ - 6\beta\left\|\vec{\rho}\otimes\Sigma_p^-\right\|_+ + 3\beta^2\left\|\Sigma_p^-\otimes\Sigma_p^-\right\|_+\right] \tag{4.8}$$

$$I^{(2)}_{\mu\mu}(\vec{\theta}) = \frac{1}{\sigma^2}\left(1-\beta\Delta\right)^2 \tag{4.9}$$

$$I^{(2)}_{\sigma^2\sigma^2}(\vec{\theta}) = \frac{1}{2\sigma^4} - \frac{1}{\sigma^6}\left[2\beta\left\|\vec{\rho}\right\|_+ - \beta^2\left\|\Sigma_p^-\right\|_+\right] \tag{4.10}$$

$$I^{(2)}_{\sigma^2\beta}(\vec{\theta}) = I^{(1)}_{\beta\sigma^2}(\vec{\theta}) = \frac{1}{\sigma^4}\left[\left\|\vec{\rho}\right\|_+ - \beta\left\|\Sigma_p^-\right\|\right] \tag{4.11}$$

$$I^{(2)}_{\beta\beta}(\vec{\theta}) = \frac{1}{\sigma^2}\left\|\Sigma_p^-\right\|_+ \tag{4.12}$$

$$H_\beta = \frac{1}{2}\left[\log\left(2\pi\sigma^2\right)+1\right] - \left[\frac{\beta}{\sigma^2}\|\overrightarrow{\rho}\|_+ - \frac{\beta^2}{2\sigma^2}\|\Sigma_p^-\|_+\right] \tag{4.13}$$

onde $\|A\|_+$ indica a soma de todas as entradas do vetor / matriz A e $\otimes$ indica o produto de Kronecker (tensor). Para maiores detalhes sobre o processo de derivações matemáticas dessas três medidas, podem consultar os apêndices A, B e C, respectivamente.

4.2.1.2 Considerando como entrada uma imagem colorida

Quando a imagen de entrada é colorida, o primeiro passo consiste em dividir a imagem em três canais R, G e B. Em seguida, extrair as c aracaterísticas de cada canal conforme descrito na seção anterior. Depois desse processo, teremos os seguintes vetores de características

$$V_R = \left[I^{(1)}_{\mu\mu}(\vec{\theta})^R, I^{(2)}_{\mu\mu}(\vec{\theta})^R, I^{(1)}_{\sigma^2\sigma^2}(\vec{\theta})^R, I^{(2)}_{\sigma^2\sigma^2}(\vec{\theta})^R, I^{(1)}_{\sigma^2\beta}(\vec{\theta})^R, I^{(2)}_{\sigma^2\beta}(\vec{\theta})^R, I^{(1)}_{\beta\beta}(\vec{\theta})^R, I^{(2)}_{\beta\beta}(\vec{\theta})^R, H^R_\beta\right] \tag{4.14}$$

$$V_G = \left[I^{(1)}_{\mu\mu}(\vec{\theta})^G, I^{(2)}_{\mu\mu}(\vec{\theta})^G, I^{(1)}_{\sigma^2\sigma^2}(\vec{\theta})^G, I^{(2)}_{\sigma^2\sigma^2}(\vec{\theta})^G, I^{(1)}_{\sigma^2\beta}(\vec{\theta})^G, I^{(2)}_{\sigma^2\beta}(\vec{\theta})^G, I^{(1)}_{\beta\beta}(\vec{\theta})^G, I^{(2)}_{\beta\beta}(\vec{\theta})^G, H^G_\beta\right] \tag{4.15}$$

$$V_B = \left[I^{(1)}_{\mu\mu}(\vec{\theta})^B, I^{(2)}_{\mu\mu}(\vec{\theta})^B, I^{(1)}_{\sigma^2\sigma^2}(\vec{\theta})^B, I^{(2)}_{\sigma^2\sigma^2}(\vec{\theta})^B, I^{(1)}_{\sigma^2\beta}(\vec{\theta})^B, I^{(2)}_{\sigma^2\beta}(\vec{\theta})^B, I^{(1)}_{\beta\beta}(\vec{\theta})^B, I^{(2)}_{\beta\beta}(\vec{\theta})^B, H^B_\beta\right] \quad (4.16)$$

$$F_R = \left[V'_{R1} \circ V'_{R2} \circ V'_{R3} \circ V'_{R4} \circ V'_{R5} \circ V'_{R6} \circ V'_{R7} \circ V'_{R8} \circ V'_{R9} \circ V'_{R10}\right] \quad (4.17)$$

$$F_G = \left[V'_{G1} \circ V'_{G2} \circ V'_{G3} \circ V'_{G4} \circ V'_{G5} \circ V'_{G6} \circ V'_{G7} \circ V'_{G8} \circ V'_{G9} \circ V'_{G10}\right] \quad (4.18)$$

$$F_B = \left[V'_{B1} \circ V'_{B2} \circ V'_{B3} \circ V'_{B4} \circ V'_{B5} \circ V'_{B6} \circ V'_{B7} \circ V'_{B8} \circ V'_{B9} \circ V'_{B10}\right] \quad (4.19)$$

onde $V'_{ij} = {}^{V_{ij}}/_{max(V_{ij})}$ é a versão normalizada de V_{ij}, que, por sua vez, indica o vetor de características da j-ésima subbanda do i-ésimo canal de imagem e $\circ$ indica a concatenação vetorial. Segue-se que i e j assumem os valores $\{R,G,B\}$ and $\{1,2,3,4,5,6,7,8,9,10\}$ respectivamente. Por fim, o vetor de características que representará a imagem colorida de entrada é dado como mostrado abaixo

$$F = [F_R \circ F_G \circ F_B] \quad (4.20)$$

A Figura 4.3 ilustra esse processo. Nela, observa-se que cada subbanda gera um vetor de características de tamanho $270 = n_s \times (n_f + e_s) \times n_c$ em que n_s, n_f, e_s e n_c denotam o número de subbandas para cada canal, o número de componentes da matriz de informação de Fisher para cada canal, a entropia da subbanda relacionada e o número de canais, respectivamente. Aqui estão os valores que levamos em conta neste trabalho: $n_s = 10$, $n_f = 8$, $e_s = 1$ e $n_c = 3$. O algoritmo 5 resume todo o processo dessa proposta na forma de uma sequência de etapas lógicas e objetivas.

4.2.2 Aprendizado de variedades na redução de dimensionalidade não linear em classificação de texturas

Esta seção apresenta uma abordagem baseada na redução de dimensionalidade para a tarefa de classificação de texturas. Essa proposta é primeiramente inspirada pela observação de que, as técnicas de redução da dimensionalidade podem efetivamente aprender informações relevantes a partir dos dados brutos e, segundo, por uma ampla variedade de características que apresentam os vetores de características gerados por cada um dos descritores clássicos como GLCM,

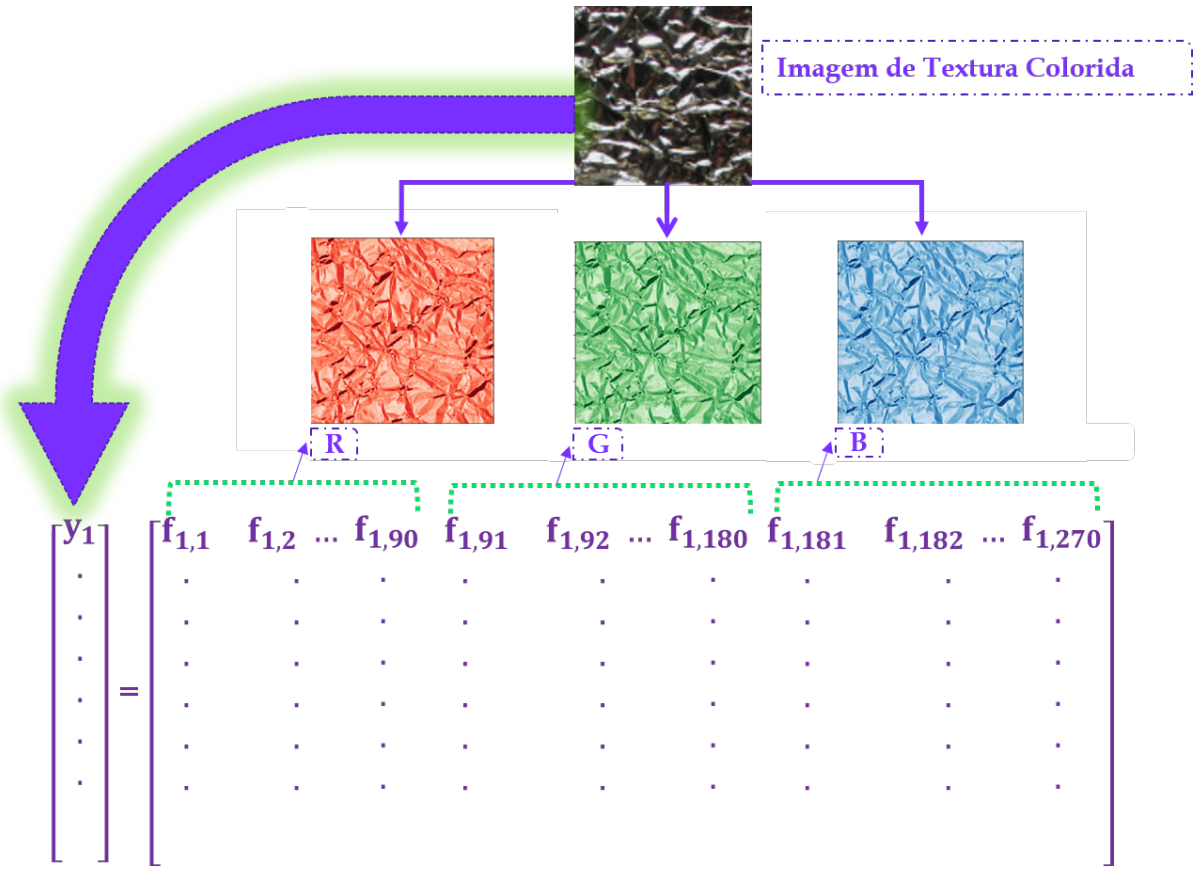


Figura 4.3: Diagrama esquemático mostrando a abordagem proposta para extrair características de textura no espaço RGB.

Haralick, HOG e LBP. Com base nisso, propomos uma solução simples que consiste em concatenar todos esses vetores de características (referidos aqui como primeira abordagem) e também aqueles vetores de características construídos a partir de *patches* extraídos diretamente dos dados brutos (referidos aqui como segunda abordagem) e faz uma seleção de características por meio do PCA e das técnicas de aprendizado de variedades para escolher um conjunto de características mais eficazes para aumentar o poder de separabilidade de classes do classificador de interesse.

A idéia consiste em criar uma abordagem para a classificação de texturas fortemente baseada na aplicação de técnicas de redução de dimensionalidade. De acordo com o diagrama de blocos do método proposto (Figura 4.4), as duas etapas da Figura 4.4 (a) fazem parte da primeira abordagem e consistem em aplicar GLCM, Haralick, HOG e LBP para a extração de características. Em seguida, esses vetores de características são concatenados. O motivo pelo qual concatenamos esses quatro vetores é construir um vetor de características de alta dimensionalidade para servir de entrada para a redução da dimensionalidade usando o PCA, ISOMAP, LLE e Lap. Eig.. A etapa na Figura 4.4 (b) faz parte da segunda abordagem e consiste em

Algorithm 5 Extração de Características baseada em Medidas da Teoria da Informação em GMRF

procedure EXTRACAOCARACTERISTICAS
  Input imagem
  If imagem colorida **Split** em imagens R, G e B
  Convert cada imagem para escala de cinza
  Apply DWT para decompor cada imagem no domínio *wavelet*
  for cada subbanda **do**
    a. Extrair *patches* de dimensão 3×3
    b. Computar a matriz de covariância dos *patches*
    c. Estimar o parâmetro β a partir da equação (4.4)
    d. Computar os componentes da informação de Fisher a partir da equação (4.5) até a equação (4.12)
    e. Computar a entropia a partir da equação (4.13)
  end for
  Normalize todos os valores
  Concatenate os valores acima em um único vetor F
  Repeat passo 2 até o passo 14 para todas as imagens presentes na base de dados.
end procedure

extrair *patches* de dimensão 32×32 a partir da imagem de entrada, e em seguida, concatená-los para formar o vetor de características da imagem. Na próximo etapa do diagrama, é realizada a seleção de características através de algoritmos de PCA, ISOMAP, LLE e Lap. Eig.. A partir desse processo, temos como saída um vetor de característica discriminativo e reduzido que contém apenas os melhores elementos (as informações mais relevantes) dos descritores de entrada. Em seguida, a tarefa de classificação é aplicada sobre o último vetor usando KNN, *Naive Bayes*, Árvores de Decisão e Redes Neurais. Finalmente, é realizada uma análise comparativa usando o teste de Wilcoxon (WILCOXON, 1992) para descobrir se os métodos de redução de dimensionalidade não linear são melhores que os lineares, usando as duas abordagens propostas.

4.2.3 Combinando as Redes Neurais Convolucionais (CNN's) com DIMAL na extração de características de imagens de textura

Nesta seção apresentamos uma nova abordagem de representação de textura, em duas etapas, baseada em CNNs para a classificação de textura, combinando o aprendizado profundo e de variedade atuando em conjunto na extração e seleção de padrões de texturas. Nosso método é inspirado em primeiro lugar pela observação de que, as técnicas de redução de dimensionalidade podem efetivamente aprender informações relevantes a partir dos dados brutos e, em segundo lugar, pela capacidade dos modelos baseados em CNNs em manipular grandes conjuntos de

dados rotulados e aprender características de alta qualidade a partir deles.

Com base nisso, propomos uma solução simples que consiste em extrair o vetor de características de uma imagem de textura usando a rede VGG-19 treinada do zero e, em seguida, faz a seleção de características através do *framework Deep Isometric Manifold Learning* que foi modificado/reimplementado especialmente para atender a necessidade da proposta apresentada neste trabalho.

A motivação para o uso da arquitetura VGG-19 foi a sua capacidade de extrair características de textura de alta qualidade. Portanto, como mostrado em (FUJIEDA; TAKAYAMA; HACHISUKA, 2017), um dos seus pontos fracos na tarefa de classificação de textura é causado à alta dimensionalidade do seu vetor de saída (contendo 4096 elementos). Essa alta dimensionalidade cria o fenômeno de superestimação dos classificadores utilizados na subsequente tarefa de classificação. Nesse contexto, a combinação das redes neurais profundas e das propriedades isométricas de tais variedades imersas no espaço ambiente poderia ajudar a melhorar a capacidade discriminativa desses descritores.

De acordo que a Figura 4.5 (a), a primeira etapa do pipeline proposto consiste em treinar a rede VGG-19 do zero para extrair características de textura da imagem de entrada. Em seguida, usamos o vetor de característica resultante (sendo a segunda camada completamente conectada, FC2, de 4096 neurônios) como entrada do *framework* DIMAL para a seleção de características (Figura 4.5 (b)). Por fim, o vetor de características selecionado serve de entrada para a tarefa de classificação usando o classificador SVM (Figura 4.5 (c)). O algoritmo 6 resume todo o processo dessa abordagem em uma sequência de etapas lógicas e objetivas.

Em poucas palavras, a configuração siamesa utilizada no algoritmo 6 consiste em uma arquitetura de duas redes idênticas que processam duas entradas diferentes. As saídas são então combinadas em uma função de custo que normalmente é uma função da distância entre par de saídas. Para cada p-ésimo par de pontos, a distância geodésica é estimada através de algoritmo de menor caminho e então essa rede e treinada minimizando função de custo a seguir

$$L(\Theta) = \sum_{p} \left(||\chi_{\Theta}(X_1^{(p)}) - \chi_{\Theta}(X_2^{(p)})|| - d^{(p)} \right)^2. \tag{4.21}$$

Vale ressaltar que cada uma das redes dessa configuração modela o mapa $\chi_{\Theta} : R^M \rightarrow R^m$, $m \leq M$. Assim, o esperado aqui é com o número suficiente de pares de examplos, a rede consiga aprender a modelar uma χ_{Θ}, de alta para menor dimensão, que preserve a isometria.

Também, foi utilizada a estratégia de amostragem de pontos mais distante (*farthest point sampling*) (BRONSTEIN; BRONSTEIN; KIMMEL, 2008; HOCHBAUM; SHMOYS, 1985)

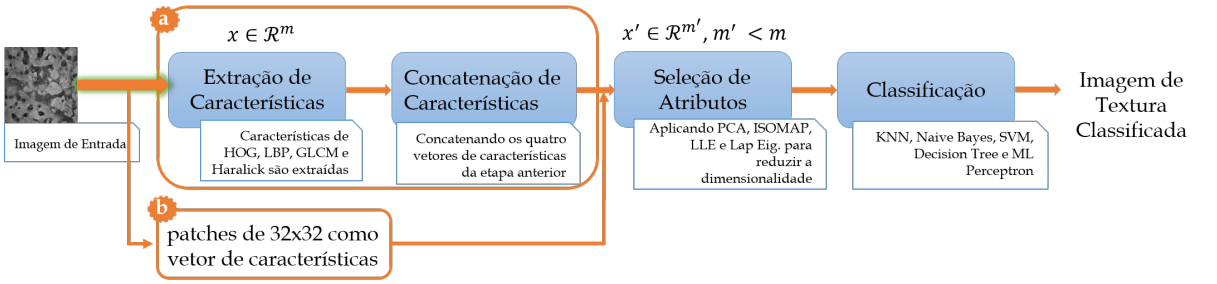


Figura 4.4: Diagrama de blocos do sistema proposto para classificação de imagens de texturas utilizando técnicas de redução de dimensionalidade não lineares (técnicas de aprendizado de variedades).

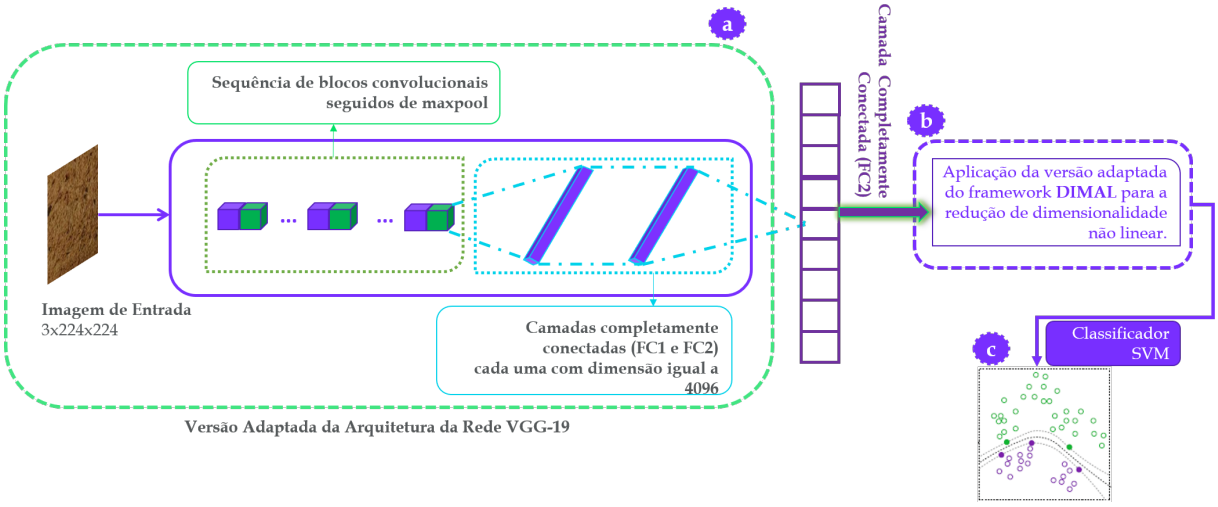


Figura 4.5: Diagrama de blocos do sistema proposto para classificação de imagens de texturas utilizando a combinação de modelos baseados em Redes Neurais Convolucionais (aprendizado profundo) e o *framework* DIMAL (PAI et al., 2018).

(também referida como a estratégia *MinMax* em (SILVA; TENENBAUM, 2004)) no algoritmo 6. Ela consiste em um método para escolher pontos de referência entre os pontos de uma variedade discretamente amostrada de tal forma que, sob certas condições, essas amostras possam cobrir toda variedade (*manifold*) o mais uniformemente possível. Partindo de uma seleção aleatória, pontos de referência são escolhidos um de cada vez de modo que cada nova seleção das amostras não utilizadas tenha a maior distância geodésica ao conjunto de amostras ja selecionadas.

Algorithm 6 Extração de Características baseada em Aprendizado Profundo e de Variedades

Input imagem colorida
Output vetor de características de menor dimensionalidade:
$\{x_i \in R^n, n \leq M\}, i = 1,2,...N,$

1. A partir da imagem de entrada, treine a rede VGG-19 do zero (*from scratch*) e considere a saída da camada completamente conectada (FC2) como o vetor de características de dimensão 4096. Dessa forma, para todas as imagens de entrada teremos o seguinte conjunto de pontos de alta dimensionalidade $\{X_i \in R^M\}, i = 1,2,...N$

2. Calcule o gráfico do vizinho mais próximo a partir do vetor de pontos do passo 1 e obtenha um conjunto de K pontos de referência (*landmark points*) usando o algoritmo de amostragem de ponto mais distante (*farthest point sampling*)

3. Calcule distâncias geodésicas D_s entre pares de pontos de referência usando Dijkstra's ou qualquer outro algoritmo numérico

4. Forme o conjunto contendo pares de pontos de referência com suas respectivas distâncias geodésicas da seguinte maneira $\left\{X_1^{(p)}, X_2^{(p)}, d^{(p)} = d\left(X_1^{(p)}, X_2^{(p)}\right)\right\}$

5. **Training:** Treine a rede χ_Θ com a configuração siamesa para o conjunto de dados obtido no passo 4 minimizando a função de custo da equação (4.21).

6. **Inference:** Obtenha o vetor de características de menor dimensão (imersão no subespaço de R^M) através de uma passagem pela rede tal que $x_i = \chi_\Theta(X_i), \forall i \in \{1,2,...N\}$

Capítulo 5

RESULTADOS E DISCUSSÕES

5.1 Datasets de imagens

Quatro datasets de imagens de textura foram utilizados nos experimentos deste trabalho:

- **Salzburg (KWITT; MEERWALD, 2018)** contém uma coleção de 476 imagens de textura coloridas que foram capturadas em Salzburgo (Áustria). Cada classe (do total de 10 classes), possui imagens de dimensão 128×128, dos quais 80% foram usados para treinar o classificador, enquanto os outros 20% foram usados para o teste.

- **Outex_TC_00010_r (OJALA T. MAENPAA; HUOVINEN, 2002)** contém 24 classes de textura, com 20 imagens por classe totalizando 480 imagens para o treinamento. O conjunto de teste é composto de 3840 imagens, 160 imagens por classe. As imagens foram capturadas com a iluminação "inca" e possuem diferentes rotações entre outros, 0^0, 5^0, 10^0, 15^0, 30^0, 45^0, 60^0, 75^0 e 90^0.

- **Outex_TC_00011_r (OJALA T. MAENPAA; HUOVINEN, 2002)** contém 24 classes de textura, com 20 imagens por classe totalizando 480 imagens para o treinamento, e também, 480 imagens para o conjunto de teste. As imagens foram capturadas com a iluminação "inca" e não possuem nenhuma variância rotacional.

- **KTH-TIPS2b (Caputo; Hayman; Mallikarjuna, 2005)** contém uma coleção de 11 classes de imagens. Cada classe consiste em 432 imagens de textura, divididas em 4 instâncias diferentes. Desse modo, cada classe possui 4 instâncias, com 108 imagens cada. Todas as imagens são capturadas variando escalas, ângulos de visão e condições de iluminação para estudar a capacidade de generalização dos métodos de reconhecimento

de material para as novas instâncias de material. Usamos 3 instâncias das 4 instâncias para treinamento e 1 instância para teste.

5.2 Descrição dos experimentos

Neste trabalho, realizamos os três tipos de experimentos a seguir:

- **Experimentos em que são calculadas as acurácias dos classificadores clássicos usando tanto os descritores tradicionais de textura como o estado da arte para fins de análise comparativa com a abordagem proposta;**
- **Experimentos em que os coeficientes *Kappa* são calculados para se ter uma noção do grau de aceitação da abordagem proposta;**
- **Experimentos em que as médias de precisão, revocação e *F1-score* são calculadas para se ter uma noção da predição em termo da especificidade, da sensibilidade e do *trade-offs* entre as duas medidas, respectivamente.**

No período relacionado às execuções destes experimentos, foram aplicados os classificadores KNN, SVM e Naive Bayes. Os descritores considerados para a análise comparativa foram LBP, GLCM, HOG, Haralick e CNN treinado do zero (*handcrafted CNN*). Todos os experimentos foram realizados em uma validação cruzada $k-fold$ estratificada. Também restringimos o parâmetro k do algoritmo KNN a $1 \leq k \leq \sqrt{n}$, onde n indica o número de amostras. Essa decisão foi baseada na teoria que diz que k deve variar entre 1 e a raiz quadrada do número de amostras do conjunto de dados. Para os descritores LBP e HOG, restringimos R e P a 2 e 16, usamos 128 *bins* e blocos de 1×1 de células de 128×128 pixels, respectivamente.

Essas configurações iniciais dos experimentos foram utilizadas especialmente para a primeira abordagem proposta neste trabalho descrita na seção 4.2.1.1. Para as demais abordagens propostas, usamos ou um subconjunto dessas configurações ou uma extensão delas.

5.3 Medidas Quantitativas de Desempenho

Para avaliar o desempenho de uma classificação supervisionada de maneira objetiva, geralmente é necessário usar critérios quantitativos. Um dos critérios mais utilizados é a taxa de sucesso ou a acurácia global. No entanto, essa medida não permite uma análise estatística robusta nem uma inferência nos resultados obtidos. Com base nisso, neste trabalho, além de usar

a acurácia como uma medida de desempenho, também adotamos o coeficiente *Kappa* de Cohen (COHEN, 1960) como uma medida quantitativa para avaliar o desempenho do método proposto. Uma das motivações que nos levou a escolher o coeficiente *Kappa* de um conjunto de métodos de medidas estatísticas é sua fácil tratabilidade matemática. Na literatura, podemos encontrar expressões fechadas bem definidas para calcular seu valor diretamente da matriz de confusão (CONGALTON, 1991), o que representa uma imensa facilidade para lidar com problemas de classificação supervisionada. A matriz de confusão para um problema de classificação com C classes é definida como

$$Cm = \begin{bmatrix} \in_{11} & \in_{12} & \cdots & \in_{1C} \\ \in_{21} & \ddots & & \vdots \\ \vdots & & \ddots & \\ \in_{c1} & \cdots & & \in_{CC} \end{bmatrix}_{CC}$$

where each element $\in_{ij}$ represents the number of elements of class i classified as elements of class j. Thus, the diagonal of the matrix contains the elements that indicate the number of success. From this, the global accuracy of the model can be computed as:

onde cada elemento $\in_{ij}$ representa o número de elementos da classe i classificados como sendo da classe j. Dessa forma, os elementos da diagonal indicam o número de acertos. A partir dessa matriz, a acurácia global pode ser calculada como

$$Acc = \frac{\sum_{i=1}^{C} c_{ii}}{\sum_{i=1}^{C}\sum_{j=1}^{C} c_{ij}} \tag{5.1}$$

No contexto da classificação supervisionada, o coeficiente *Kappa* determina o grau de concordância entre o resultado esperado e o observado pelo classificador. Quanto melhor o desempenho da classificação, maior o grau de concordância e, consequentemente, maior o valor do *Kappa*. A expressão para calcular o coeficiente *Kappa* a partir da matriz de confusão é dada por

$$\hat{K} = \frac{N\sum_{i=1}^{C} c_{ii} - \sum_{i=1}^{C} x_{i+}x_{+i}}{N^2 - \sum_{i=1}^{C} x_{i+}x_{+i}} \tag{5.2}$$

onde x_{i+} é a soma dos elementos da linha i , x_{+i} é a soma dos elementos da coluna i, C é o número de classes e N é o número total de observações.

Alguns autores definiram possíveis interpretações do desempenho de um método classificação aplicado com base no valor associado ao coeficiente *Kappa*. A Tabela 5.1 ilustra a interpretação encontrada em (CONGALTON, 1991), que tem sido usada em todos os experimentos deste

trabalho.

Tabela 5.1: Desempenho da classificação em função do *Kappa*.

Coeficiente *Kappa*	Desempenho da Classificação
$\hat{k} \leq 0$	Péssimo
$0 < \hat{k} \leq 0.2$	Ruim
$0.2 < \hat{k} \leq 0.4$	Razoável
$0.4 < \hat{k} \leq 0.6$	Bom
$0.6 < \hat{k} \leq 0.8$	Muito Bom
$0.8 < \hat{k} \leq 1.0$	Excelente

Alternativamente, a revocação (ou sensibilidade) (r) foi usada para medir a proporção de instâncias positivas que foram previstas pelo modelo de classificação, enquanto a precisão (pr) foi usada para avaliar o modelo proposto em relação à confiabilidade da sua classificação. Além disso, o $F1-score$ foi utilizado servindo de $trade-off$ entre revocação e precisão. A expressão para o cálculo da versão generalizada do $F1-score$ é definida na equação (5.3). Ela depende do parâmetro β (que não deve ser confundido com o parâmetro β do Capítulo 4) fornecido pelo usuário para ponderar tanto a precisão quanto a revocação.

$$F_\beta = \frac{(\beta^2+1)pr.r}{\beta^2 pr + r} \tag{5.3}$$

$F1-score$ usado neste experimento é a versão da média harmônica da precisão e da revocação quando $\beta = 1$. No entanto, dependendo do tipo de *trade-offs* que se quer analisar entre essas duas medidas, β pode ser ajustado para atingir o objetivo desejado.

As fórmulas generalizadas utilizadas para o cálculo da precisão e da revocação de cada classe i são dadas, respectivamente, por

$$pr_i = \frac{\sum_{j=1}^{C} c_{jj}}{\sum_{j=1}^{C} c_{jj} + \sum_{\substack{j=1 \\ j\neq i}}^{C} c_{ji}} \tag{5.4}$$

$$r_i = \frac{\sum_{j=1}^{C} c_{jj}}{\sum_{j=1}^{C} c_{jj} + \sum_{\substack{j=1 \\ j\neq i}}^{C} c_{ij}} \tag{5.5}$$

5.4 Resultados e Discussões

5.4.1 Novo descritor baseado em teoria da informação em campos aleatórios Markovianos

5.4.1.1 Experimentos no Salzburg

(a) Validação baseada em acurácia: Os resultados apresentados na Tabela D.1 referente ao classificador KNN indicam que a abordagem proposta obteve melhor desempenho em quase todos os valores de k, superando, portanto, demais métodos. No entanto, apenas GLCM apresentou um desempenhp igual ao método proposto para $k = 4$. É importante enfatizar que a abordagem proposta é 2,78% superior tanto ao GLCM quanto ao Haralick (com $k = 2$), que sao os dois segundo melhores métodos. Desse modo, como as acurácias desses dois descritores são próximas de 95%, ter uma diferença de 2,78% (a mais) na acurácia parece muito promissora para essa nova proposta.

Os resultados do classificador SVM estão resumidos na Tabela D.2. Vale ressaltar que o método proposto proporcionou uma taxa de acerto de 98,44% usando o *kernel* polinomial superando assim os quatro métodos comparativos. A diferença de taxa de acerto entre o método proposto e o segundo melhor método (LBP e Haralick) é de 4,00%.

Além disso, de acordo com a Tabela D.3, o método proposto apresentou uma taxa de acerto de 94,44% com o classificador Naive Bayes, superando os quatro métodos comparativos. Esses resultados mostraram também que a proposta é 2,78% superior aos três melhores métodos comparativos (LBP, GLCM e Haralick).

O desempenho do modelo CNN (*from scratch*) é apresentado na Tabela D.4 com uma taxa de acerto de 68.90%. Como pode ser observado, a abordagem proposta é 27,8% superior ao CNN quando comparado com a sua média de desempenho em geral.

Por fim, a abordagem proposta permitiu aumentar em 2,78%, 4,00% e 2,78% o desempenho de KNN, SVM e Naive Bayes, respectivamente. Consequentemente, houve, em média, um aumento de 3,19% na taxa de classificação no dataset Salzburg. Isso aponta claramente para a vantagem e contribuições significativas do método proposto.

(b) Validação baseada no coeficiente *Kappa*: Os coeficientes *Kappa* relacionados aos classificadores KNN, SVM e Naives Bayes são apresentados nas Tabelas D.5, D.6 e D.7. Com base na interpretação mostrada na Tabela 5.1, é evidente que, em geral, o método proposto se destacou, sendo considerado excelente. Não obstante, em apenas um caso o método proposto foi considerado sendo muito bom.

(c) Validação baseada na precisão, revocação e F1-score: Com base na interpretação mostrada na Tabela 5.1 e nos resultados apresentados na Tabelas D.8, D.9 e D.10, é evidente que, em geral, o método proposto se destacou, sendo considerado muito bom.

Tabela 5.2: KNN: Desempenho de classificação em Salzburg.

k	Acurácia (%)				
	LBP	GLCM	HOG	Haralick	Abordagem Proposta
2	88.88	94.44	69.44	94.44	**97.22**
4	86.11	94.44	66.66	83.33	94.44
8	80.55	91.66	69.44	77.77	94.44
12	83.33	86.11	66.66	77.77	91.44

Tabela 5.3: SVM: Desempenho de classificação em Salzburg.

Kernel	Acurácia (%)				
	LBP	GLCM	HoG	Haralick	Abordagem Proposta
Linear	93.22	91.66	86.11	94.44	**97.22**
Rbf	74.21	30.55	91.66	58.33	86.11
LinearSVC	83.33	83.33	75.00	69.44	**97.22**

Tabela 5.4: Naive Bayes: Desempenho de classificação em Salzburg.

Acurácia (%)				
LBP	GLCM	HOG	Haralick	Abordagem Proposta
91.66	91.66	88.88	91.66	**94.44**

Tabela 5.5: CNN: Desempenho de classificação em Salzburg.

Acurácia (%)
CNN
68.90

Tabela 5.6: KNN: Desempenho da classificação do método proposto em função do *Kappa* no Salzburg.

k	$\hat{k}$ (Coeficiente *Kappa*)
2	0.9339
4	0.9006
8	0.9339
12	0.7988

O restante dos resultados experimentais relacionados à primeira abordagem encontram-se no apêncdice D, tendo a sua interpretação semelhante à apresentada nessa seção.

Tabela 5.7: SVM: Desempenho da classificação do método proposto em função do *Kappa* no Salzburg.

Kernel	$\hat{k}$ (Coeficiente *Kappa*)
Linear	0.9670
Rbf	0.8686
LinearSVC	0.8687

Tabela 5.8: Naive Bayes: Desempenho da classificação do método proposto em função do *Kappa* no Salzburg.

$\hat{k}$ (Coeficiente *Kappa*)
0.9010

Tabela 5.9: KNN: Desempenho da classificação do método proposto em função da precisão, revocação e *F1-score* no Salzburg.

	Taxa de sucesso (%)		
Média k	Precisão	Revocação	*F1-score*
Micro 2	91.67	91.67	91.67
Macro 2	94.71	92.59	92.21
Ponderada 2	93.92	91.67	91.28
Micro 4	88.89	88.89	88.89
Macro 4	92.86	88.89	88.98
Ponderada 4	91.60	88.89	88.35
Micro 8	83.33	83.33	83.33
Macro 8	91.01	82.41	81.20
Ponderada 8	89.75	83.33	81.40
Micro 12	83.33	83.33	83.33
Macro 12	91.01	82.41	81.20
Ponderada 12	89.75	83.33	81.40

Tabela 5.10: SVM: Desempenho da classificação do método proposto em função da precisão, revocação e *F1-score* no Salzburg.

	Taxa de sucesso (%)		
Média do Kernel	precisão	Revocação	*F1-score*
Micro Linear	86.11	86.11	86.11
Macro Linear	92.86	86.11	85.54
Ponderada Linear	92.06	86.11	85.17
Micro Rbf	80.56	80.56	80.56
Macro Rbf	91.75	80.56	79.94
Ponderada Rbf	90.95	80.56	79.79
Micro LinearSVC	88.89	88.89	88.89
Macro LinearSVC	93.65	88.89	89.18
Ponderada LinearSVC	92.86	88.89	88.80

Tabela 5.11: Naive Bayes: Desempenho da classificação do método proposto em função da precisão, revocação e *F1-score* no Salzburg.

	Taxa de sucesso (%)		
Média	Precisão	Revocação	*F1-score*
Micro	91.67	91.67	91.67
Macro	93.78	92.59	92.76
Ponderada	92.53	91.67	91.58

5.4.2 Aprendizado de variedades na redução de dimensionalidade não linear em classificação de texturas

Nesta abordagem, realizamos dois tipos de experimentos, que são:

- **Experimentos usando o vetor de características criado concatenando os descritores LBP, GLCM, HOG e Haralick**. Com isso, a seleção de características é realizada através do PCA, ISOMAP, LLE e Lap. Eig.. A saída desse processo serve de entrada para a tarefa de classificação usando KNN, Naive Bayes, Árvores de decisão e MLP. As Tabelas 5.12 e 5.14 mostram os resultados do desempenho dessa proposta no Salzburg e Outex_TC_00011_r, respectivamente;

- **Experimentos usando o vetor de características criado concatenando *patches* de tamanho 32 x 32 extraídos diretamente da imagem de entrada**. Com isso, a seleção de características é realizada através do PCA, ISOMAP, LLE e Lap. Eig.. A saída desse processo serve de entrada para a tarefa de classificação usando KNN, Naive Bayes, Árvores de decisão e MLP. As Tabelas 5.13 e 5.15 mostram os resultados do desempenho dessa proposta no Salzburg e Outex_TC_00011_r, respectivamente

O objetivo principal foi analisar se a redução por PCA produz melhores resultados do que algoritmos de aprendizado de variedades para as duas abordagens de extração. Nesse sentido, usamos o teste Wilcoxon para descobrir qual dos dois métodos possui uma diferença significativa. Os testes foram realizados em pares e $\alpha = 0,05$ foi utilizado como nível de significância.

Tabelas 5.12 e 5.14 apresentam uma visão geral dos resultados. Para cada conjunto de características de tamanho D reduzimos a sua dimensionalidade em $d \in \{1,2,3,4,5,6,7,8,9,10\}$ características (atributos) usando PCA, ISOMAP, LLE e Lap. Assim, a escolha do d pode melhorar ou piorar a capacidade discriminativa dos descritores de textura durante a tarefa de classificação. Como uma possível solução para isso, adotamos, neste trabalho, uma abordagem de estimativa de dimensionalidade intrínseca para encontrar o valor intrínseco de d. Ela consite em calcular a taxa de variância acumulada do i-ésimo componente a partir de uma lista de

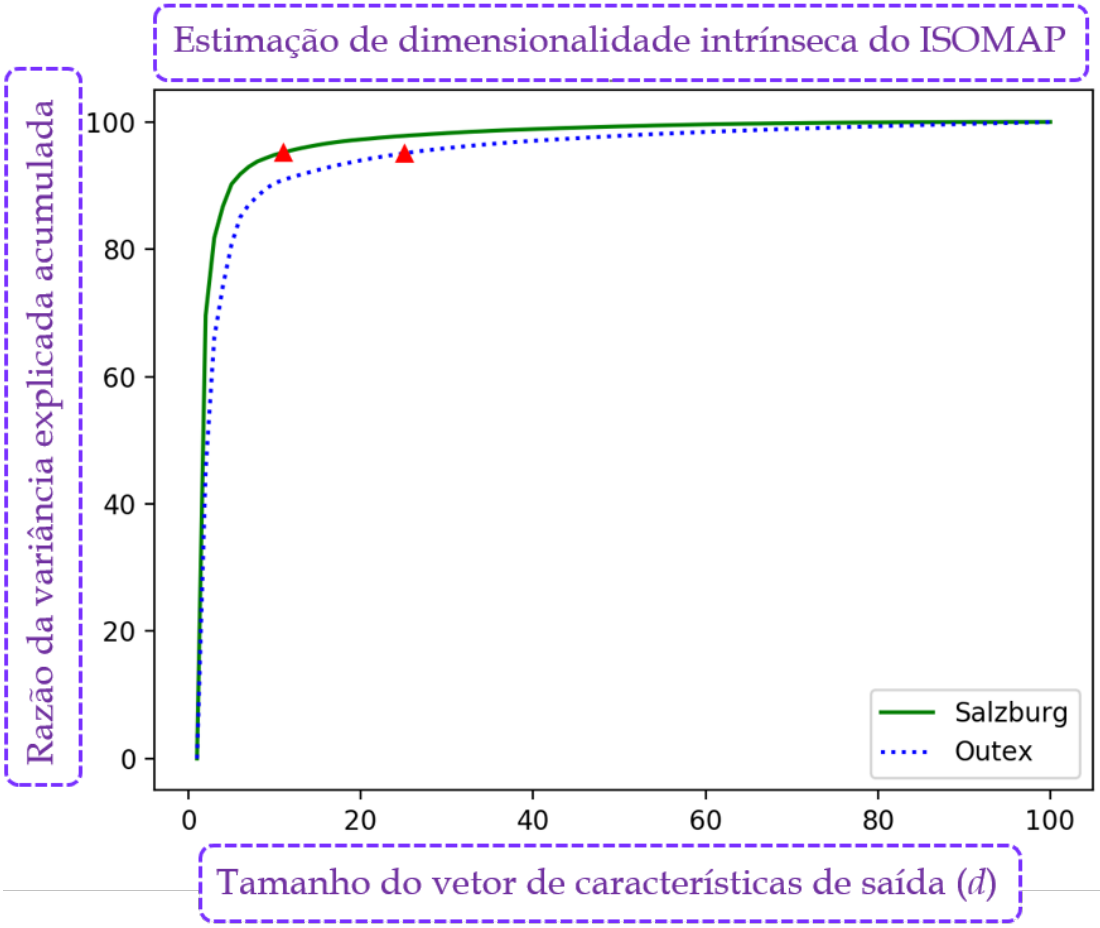
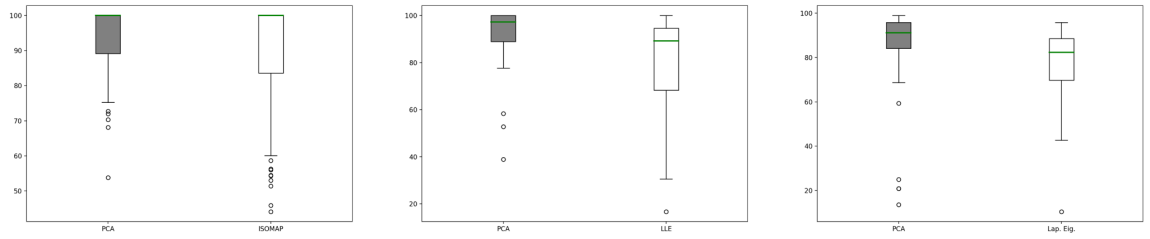


Figura 5.1: Estimação de dimensionalidade intrínsica do ISOMAP para os datasets Salzburg e Outex usando o vetor de características criado pela concatenação dos descritores LBP, GLCM, Haralick e HOG.

n componentes (dimensões). Assim, a dimensionalidade intrínseca do espaço reduzido será o primeiro componente com taxa de variância acumulada maior que 95 % no caso o algoritmo ISOMAP, por exemplo. Uma das observações importantes desse resultado é que todas reduções subsequentes calculadas com um valor maior que d geralmente tem como efeito manter as taxas de acerto e às vezes pode aumentá-las levemente. Figura 5.1 mostra o valor observado de d, variando de 0 a 100, com base na taxa de variância retida nos atributos. Também mostra que as dimensionalidades intrínsecas dos datasets Salzburg e Outex para o algoritmo ISOMAP são 10 e 24, respectivamente.

Também, realizamos análises mais pronfundas que consistem em comparar o desempenho entre pares de métodos de redução de dimensionalidade através do teste de Wilcoxon com um nível de significância $\alpha = 0,05$: PCA com ISOMAP, PCA com LLE e PCA com Lap. Eig. Para observar os resultados com mais detalhes, *boxplots* do Salzburg e Outex consistindo em pares PCA com LLE e PCA com Lap. Eig são mostrados nas Figuras 5.2 (b) e 5.2 (c), respecti-



(a) Salzburg-2 (b) Salzburg-1 (c) Outex-1

Figura 5.2: Acurácias do PCA, ISOMAP, LLE e Lap. Eig. no Salzburg usando como vetor de características *patches* de tamanho 32 x 32 (a), no Salzburg usando como vetor de características a concatenação dos descritores LBP, GLCM, Haralick e HOG (b) e no Outex como vetor de características a concatenação dos descritores LBP, GLCM, Haralick e HOG (c). Boxplots em cinza correspondem às significâncias quando comparado com a acurácia do outro método com p-value $<$ 0.05. Aqui, o nome Salzburg-2 (a) é referente ao segundo tipo de experimentos descritos nessa seção. Analogamente, os nomes Salzburg-1 (b) e Outex-1 (c) são referentes ao primeiro tipo de experimentos descritos nessa seção.

		Número de Características									
		1	2	3	4	5	6	7	8	9	10
PCA	KNN	58.33	88.89	88.89	100.00	100.00	100.00	100.00	100.00	100.00	100.00
	N.B.	80.56	86.11	97.22	100.00	97.22	100.00	97.22	100.00	100.00	100.00
	MLP	38.89	77.78	80.56	94.44	97.22	91.67	97.22	100.00	100.00	100.00
	DT	52.78	86.11	88.89	94.44	94.44	88.89	94.44	94.44	97.22	88.89
ISOMAP	KNN	86.11	100.00	100.00	100.00	100.00	100.00	100.00	100.00	97.22	100.00
	N.B.	75.00	100.00	100.00	97.22	100.00	97.22	100.00	94.44	88.89	100.00
	MLP	44.44	77.78	91.67	91.67	94.44	94.44	100.00	97.22	97.22	100.00
	DT	75.00	97.22	88.89	94.44	97.22	94.44	83.33	83.33	91.67	83.33
LLE	KNN	66.67	69.44	94.44	100.00	97.22	97.22	100.00	100.00	100.00	97.22
	N.B.	41.67	68.75	78.13	83.33	82.29	82.29	94.79	92.71	89.58	90.63
	MLP	30.56	16.67	55.56	55.56	47.22	50.00	69.44	47.22	94.44	83.33
	DT	63.89	75.00	91.67	94.44	91.67	97.22	97.22	88.89	94.44	94.44
Lap. Eig.	KNN	72.22	83.33	100.00	100.00	97.22	100.00	100.00	94.44	100.00	100.00
	N.B.	72.22	83.33	100.00	97.22	100.00	97.22	97.22	94.44	100.00	97.22
	MLP	11.11	55.56	91.67	100.00	91.67	94.44	97.22	91.67	100.00	97.22
	DT	63.89	80.56	100.00	100.00	94.44	94.44	94.44	94.44	91.67	91.67

Tabela 5.12: Resultados quantitativos comparando PCA com as três técnicas de aprendizado de variedades na base Salzburg. Aqui consideramos o vetor de características completo criado concatenando todos os descritores.

		Número de Características										
		5	10	15	20	25	30	35	40	45	50	100
PCA	KNN	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00
	N.B.	53.82	68.14	70.33	72.68	71.99	75.23	78.95	79.15	79.15	80.78	83.24
	MLP	91.11	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00
	DT	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00
ISOMAP	KNN	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00
	N.B.	44.08	45.85	52.97	51.37	54.35	56.01	54.48	56.24	58.69	60.03	68.89
	MLP	88.46	98.76	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00
	DT	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00
LLE	KNN	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00
	N.B.	22.91	36.01	41.08	43.50	48.04	49.51	49.58	51.80	52.75	53.89	63.33
	MLP	43.46	52.55	54.80	57.78	63.89	68.82	69.12	71.86	72.42	77.25	90.20
	DT	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00
Lap. Eig.	KNN	99.97	99.97	100.00	100.00	100.00	99.97	100.00	100.00	100.00	100.00	100.00
	N.B.	47.78	46.54	54.64	52.97	55.33	56.34	55.39	56.67	54.41	56.70	61.01
	MLP	66.86	76.18	80.36	86.44	91.57	92.88	95.26	98.56	98.14	99.08	99.74
	DT	99.90	99.93	99.93	99.93	99.87	99.84	99.97	99.97	99.84	99.87	99.80

Tabela 5.13: Resultados quantitativos comparando PCA com as três técnicas de aprendizado de variedades na base Salzburg. Aqui, o *patch* de dimensão 32 x 32 de uma imagem de textura é considerado como vetor de características.

		Número de Características									
		1	2	3	4	5	6	7	8	9	10
PCA	KNN	20.83	77.08	94.79	90.63	95.83	95.83	98.96	97.92	96.88	97.92
	N.B.	25.00	73.96	91.67	87.50	91.67	92.71	91.67	95.83	96.88	98.96
	MLP	13.54	59.38	81.25	84.38	88.54	88.54	94.79	95.83	96.88	98.96
	DT	20.83	68.75	88.54	83.33	84.38	85.42	86.46	88.54	92.71	92.71
ISOMAP	KNN	31.25	83.33	89.58	92.71	92.71	96.88	97.92	97.92	98.96	95.83
	N.B.	31.25	70.83	81.25	88.54	84.38	94.79	92.71	96.88	94.79	94.79
	MLP	17.71	60.42	82.29	87.50	88.54	91.67	92.71	92.71	94.79	95.83
	DT	22.92	69.79	85.42	83.33	78.13	90.63	88.54	84.38	87.50	86.46
LLE	KNN	40.63	67.71	79.17	88.54	89.58	92.71	98.96	97.92	100.00	96.88
	N.B.	41.67	68.75	78.13	83.33	82.29	82.29	94.79	92.71	89.58	90.63
	MLP	30.56	16.67	55.56	55.56	47.22	50.00	69.44	47.22	94.44	83.33
	DT	63.89	75.00	91.67	94.44	91.67	97.22	97.22	88.89	94.44	94.44
Lap. Eig.	KNN	46.88	84.38	88.54	91.67	88.54	95.83	92.71	88.54	88.54	91.67
	N.B.	50.00	73.96	75.00	78.13	77.08	82.29	82.29	82.29	83.33	86.46
	MLP	10.42	42.71	58.33	58.33	66.67	67.71	73.96	69.79	77.08	86.46
	DT	43.75	69.79	77.08	83.33	83.33	85.42	88.54	88.54	88.54	88.54

Tabela 5.14: Resultados quantitativos comparando PCA com as três técnicas de aprendizado de variedades na base Outex. Aqui consideramos o vetor de características completo criado concatenando todos os descritores.

		Número de Características										
		5	10	15	20	25	30	35	40	45	50	100
PCA	KNN	56.51	62.24	61.98	60.42	58.33	54.69	53.65	52.60	48.96	45.83	37.50
	N.B.	61.98	70.05	73.95	76.30	79.43	78.91	79.69	80.47	79.95	80.47	80.47
	MLP	63.80	73.70	77.08	75.26	76.82	71.88	69.79	67.97	63.80	63.02	49.23
	DT	53.39	56.25	61.20	58.33	57.81	58.85	58.33	55.73	57.81	57.29	53.13
ISOMAP	KNN	70.31	76.82	76.56	76.30	75.78	76.56	76.56	76.56	75.78	75.50	72.14
	N.B.	55.99	68.23	72.66	73.18	73.18	73.70	73.96	72.92	73.96	74.48	73.70
	MLP	51.82	60.42	65.63	63.28	66.41	62.76	60.94	57.81	63.02	56.25	54.95
	DT	58.07	65.36	66.93	66.41	64.58	62.50	63.02	61.20	61.46	63.02	64.32
LLE	KNN	59.11	68.23	66.15	70.57	72.14	70.57	71.88	71.88	73.44	72.40	65.10
	N.B.	58.59	64.32	68.23	71.88	75.00	75.52	77.08	79.17	78.13	79.95	82.55
	MLP	37.24	37.50	42.19	42.71	44.53	45.83	48.18	46.88	46.09	47.66	53.91
	DT	53.13	53.65	50.26	58.33	59.11	55.73	59.11	56.77	53.65	51.82	52.34
Lap. Eig.	KNN	61.46	69.27	69.27	71.61	73.70	73.96	76.56	75.78	74.48	74.48	67.45
	N.B.	47.66	58.59	63.02	61.72	64.58	66.93	67.71	68.23	68.49	68.49	67.45
	MLP	59.38	67.71	74.22	71.09	71.88	75.26	74.22	74.22	74.22	73.96	67.97
	DT	58.85	63.28	62.76	62.50	61.98	65.10	66.41	64.32	63.80	62.50	60.42

Tabela 5.15: Resultados quantitativos comparando PCA com as três técnicas de aprendizado de variedades na base Outex. Aqui, o *patch* de dimensão 32 x 32 de uma imagem de textura é considerado como vetor de características.

vamente. Os *boxplots* sombreados em cinza correspondem a dados com diferença significativa quando comparados à acurácia do outro método, obtendo-se o valor de $p < 0.05$.

Obtivemos resultados significativamente melhores para PCA quando comparado com LLE no Salzburg (Figura 5.2 (b)) e com Lap.Eig para Outex (Figura 5.2 (c)), de acordo com o teste estatístico de Wilcoxon. De modo geral, pode-se observar que os resultados não foram significativamente piores, exceto no caso em que o valor de d é igual a 1 para todos os métodos nos dois datasets.

Uma visão geral do segundo conjunto de experimentos é apresentada na Tabela 5.13 e na Tabela 5.15 em que utilizamos os seguintes valores para d: 5, 10, 15, 20, 25, 30, 35, 40, 45, 50 e 100. A ideia principal aqui consite em averiguar se a extração de *patches* diretamente dos dados brutos de uma imagem de textura pode melhorar ou manter os resultados (taxas de acerto na classificação), e também comparar o uso do PCA e os métodos de aprendizado de variedades.

O único resultado que apresentou diferença significativa entre as comparações entre pares foi entre PCA e ISOMAP no Salzburg, conforme mostrado na Figura 5.2 (a). O PCA apresentou significativamente melhor acurácia do que ISOMAP nesse dataset, de acordo com o teste estatístico de Wilcoxon para o valor de $p < 0.05$. Mais importante, em uma perspectiva mais geral, os resultados não foram significativamente piores.

5.4.3 Combinando as Redes Neurais Convolucionais (CNN's) com DIMAL na extração de características de imagens de textura

Treinando o modelo do zero (*training from scratch*) Comparamos a abordagem proposta com dois modelos existentes considerados como estado de arte dos modelos baseados em CNN para a tarefa de classificação de textura, a saber ScatNet e PCANet. O modelo proposto alcançou um alto desempenho na classificação no dataset Salzburg (100.00 %) e KTH-TIPS2b (99.5 %), enquanto o ScatNet também obteve melhores resultados nos dois datasets, 99.7 % em Salzburg e 99.4 % em KTH-TIPS2b. O PCANet, por sua vez, apresentou um desempenho satisfatório na classificação de textura, 79.2 % em Salzburg e 84.9 % em KTH-TIPS2b. Comparado com ScatNet e PCANet, a extração de características no modelo proposto é muito mais rápida, com bom desempenho, e consequentemente, ideal para aplicações de visão computacional em tempo real, por exemplo.

Capítulo 6

CONCLUSÕES

O problema de quantificar uma medida de similaridade adequada entre diferentes objetos em um conjunto de dados é uma tarefa desafiadora em muitas aplicações em aprendizado de máquina. Encontrar medidas de similaridade alternativas é crucial para a análise de dados, especialmente em situações em que a distância Euclidiana padrão se torna uma escolha não razoável. Medidas baseadas em teoria da informação foram aplicadas com sucesso em estatística para quantificar o grau de similaridade entre variáveis aleatórias. Nesse contexto, os conceitos de entropia e informação de Fisher fornecem uma sólida base matemática para o aprendizado não supervisionado de atributos em imagens de textura.

Com relação a esse problema, a grande contribuição deste projeto de doutorado foi o desenvolvimento de um novo descritor para imagens de texturas baseado nas matrizes de informação de Fisher de modelos de campos aleatórios Gaussianos Markovianos obtidas a partir das subbandas de uma árvore de decomposição *wavelet*. Os resultados obtidos, tanto através das derivações matemáticas das expressões de forma fechada dos componentes da matriz de informação de Fisher quanto pelo bom desempenho na tarefa de classificação, mostraram que a abordagem proposta é válida, pois produz resultados estatisticamente significantes sendo também competitivos com o estado da arte na área.

Numa outra vertente, os métodos de extração de características são ferramentas matemáticas necessárias para análise de dados e aprendizado de métricas não supervisionado, uma vez que a presença de dados de alta dimensão tem se tornado padrão nas aplicações modernas de processamento de imagens e visão computacional. Recentemente, o aprendizado profundo tem sido considerado por muitos profissionais e pesquisadores como o estado da arte na criação de atributos a partir de conjuntos de dados de alta dimensionalidade, especialmente a partir de imagens. Um requisito para o aprendizado profundo ter um bom desempenho é ter um número de amostras elevado para ajustar adequadamente milhões de parâmetros das redes neurais, o

que nem sempre é possível. Os algoritmos de redução de dimensionalidade, por outro lado, são capazes de aprender atributos discriminantes a partir de conjuntos de dados menores, produzindo bons resultados mesmo quando o número de amostras é menor ou igual ao número de atributos originais. Além disso, pode-se mostrar que esses métodos estão profundamente conectados ao aprendizado de métricas não supervisionado, no sentido de que, além de aprender uma representação mais compacta e significativa para o conjunto de dados observado, esses métodos também aprendem uma função de distância que geralmente é geometricamente mais adequada para representar uma medida de similaridade entre um par de objetos na coleção. Em outras palavras, aprendendo a estrutura oculta, em geral, ganhamos uma métrica mais poderosa gratuitamente. Portanto, não parece razoável supor que, eventualmente, o aprendizado profundo substitua todos os algoritmos tradicionais de redução de dimensionalidade e aprendizado de variedades. Além disso, a maioria dos modelos de aprendizado profundo trabalha com o paradigma de aprendizado supervisionado, uma vez que são generalizações de perceptrons multicamadas, o que significa que informações sobre os rótulos das classes são necessárias, o que nem sempre é possível nas tarefas de reconhecimento de padrões.

Nesse contexto, a grande contribuição deste projeto de doutorado foi o desenvolvimento de um novo descritor que possa aplicar métodos de aprendizado de variedades para a redução de dimensionalidade não linear em problemas de classificação de texturas, verificando se tais algoritmos são mais eficazes do que métodos lineares como a Análise de Componentes Principais (PCA). Os resultados obtidos foram promissores e mostraram que a maioria de comparações entre PCA e cada um os algoritmos de aprendizado de variedades não apresentaram diferenças significativas. Isso deixa claro que ao aumentar a dimensionalidade do espaço de atributos extraídos impulsionaria/melhoraria o poder descriminativa dos descritores selecionados através das técnicas de aprendizado de variedades.

Além diss, o presente projeto de doutorado gerou mais uma grande contribuição no que diz respeito ao uso do aprendizado profundo na classificação de imagens de texturas. Foi desenvolvido uma nova abordagem de representação de textura, em duas etapas, baseada em CNNs para a classificação de textura, combinando o aprendizado profundo e de variedade atuando em conjunto na extração e seleção de padrões de texturas. Os resultados obtidos nos permitem concluir que a metodologia proposta é capaz de melhorar significativamente o desempenho da classificação de imagens de texturas nessa área. Dessa forma, a principal vantagem do método proposto em relação aos modelos existentes na literatura é o baixo custo computacional para treinar o modelo.

Por fim, em trabalhos futuros, devem ser explorados outros tipos de *wavelet* (em vez

de *Daubechies* que foi utilizado no contexto deste trabalho), e também, diferentes níveis de decomposição para tentar melhorar o desempenho do descritor proposto. No que diz respeito à proposta relacionada a redução de dimensionalidade não linear, deve-se aumentar a dimensionalide dos espaços de atributos de entrada, e também, estudar melhor os espaços gerados para tais algortimos com objetivo de promover o desempenho das subsequentes tarefas de classificação. Finalmente, outro aspecto a ser considerado é a exploração de outros modelos baseados em CNN, como AlexNet, ResNet ambos treinados do zero, para extrair características texturais, e em seguida, averiguar se houve ou não melhoria no desempenho do método proposto.

6.1 Trabalhos Publicados

6.1.1 Artigos em Periódicos e Conferências

O presente projeto de doutorado gerou alguns artigos científicos em conferências e revistas da área. A seguir encontra-se a lista dos principais trabalhos publicados e também aqueles submetidos para a publicação:

- Nsimba, C.B., Levada, A. An information-theoretic wavelet-based texture descriptor using Gaussian Markov random field models. Multimed Tools Appl 78, 31959–31986 (2019). https://doi.org/10.1007/s11042-019-07916-3 [*Qualis: B1, Status: Já publicado*]

- Nsimba C.B., Levada A.L.M. (2019) Nonlinear Dimensionality Reduction in Texture Classification: Is Manifold Learning Better Than PCA?. In: Rodrigues J. et al. (eds) Computational Science – ICCS 2019. ICCS 2019. Lecture Notes in Computer Science, vol 11540. Springer, Cham [*Qualis: A1, Status: Já publicado*]

- Nsimba, C.B., Levada, A. Exploring Information Theory and Gaussian Markov Random Fields For Color Texture Classification. ICIAR (2020). https://www.aimiconf.org/iciar20/ [*Qualis: B1, Status: Aprovado para publicação*]

- Nsimba, C.B., Levada, A. A Novel Information-theoretic Approach for Texture Image Characterization: Exploring Fisher Information in Gaussian-Markov Random Fields. International Journal of Pattern Recognition and Artificial Intelligence (IJPRAI). https://www.worldscientific.com/worldscinet/ijprai [*Qualis: B1, Status: Em avaliação*]

- Nsimba, C.B., Fiogbé, E., Levada, A. P-WMSD: An integrative framework for preventing Work-related Musculoskeletal Disorders through Image Analysis Techniques. MICCAI (2020). https://www.miccai2020.org/en/ [*Qualis: A1, Status: Em avaliação*]

- Nsimba, C.B., Levada, A. Combining Deep and Manifold Learning For Nonlinear Feature Extraction in Texture Images. EUSIPCO (2020). https://eusipco2020.org/ [*Qualis: A2, Status: Em avaliação*]

- Nsimba, C.B., Levada, A. Rotation-Invariant Feature Extraction in the DTCWT Domain based on Information Theory. CIARP (2020). https://ciarp2020.org/ [*Qualis: B1, Status: Para submeter*]

REFERÊNCIAS

ADDISON, P. *The Illustrated Wavelet Transform Handbook: Introductory Theory and Applications in Science, Engineering, Medicine and Finance*. Taylor & Francis, 2002. ISBN 9781420033397. Disponível em: ⟨https://books.google.com.br/books?id=RUSjIMQACQQC⟩.

AMARI, S.; CICHOCKI, A.; YANG, H. A new learning algorithm for blind source separation. *Advances in Neural Information Processing Systems*, v. 8, p. 757–763, 1996.

ANACONDA. *Solutions for Data Science Practitioners and Enterprise Machine Learning*. 2019. ⟨https://www.anaconda.com/⟩. Accessed: 2019-11-19.

ANDREARCZYK, V.; WHELAN, P. Using filter banks in convolutional neural networks for texture classification. *Pattern Recognition Letters*, v. 84, p. 63–69, 08 2016.

BALASUBRAMANIAN, M.; SCHWARTZ, E. L. The isomap algorithm and topological stability. *Science*, v. 295, p. 7–8, 2002.

BARROW, D. K.; CRONE, S. F. Cross-validation aggregation for combining autoregressive neural network forecasts. *International Journal of Forecasting*, v. 32, n. 4, p. 1120 – 1137, 2016. ISSN 0169-2070. Disponível em: ⟨http://www.sciencedirect.com/science/article/pii/S0169207016300188⟩.

BELKIN, M.; NIYOGI, P. Laplacian eigenmaps and spectral techniques for embedding and clustering. In: DIETTERICH, T. G.; BECKER, S.; GHAHRAMANI, Z. (Ed.). *Advances in Neural Information Processing Systems 14*. MIT Press, 2002. p. 585–591. Disponível em: ⟨http://papers.nips.cc/paper/1961-laplacian-eigenmaps-and-spectral-techniques-for-embedding-and-clustering.pdf⟩.

BELKIN, M.; NIYOGI, P. Laplacian eigenmaps and spectral techniques for embedding and clustering. In: DIETTERICH, T. G.; BECKER, S.; GHAHRAMANI, Z. (Ed.). *Advances in Neural Information Processing Systems 14*. [S.l.]: MIT Press, 2002. p. 585–591.

BELKIN, M.; NIYOGI, P. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, MIT Press, v. 15, n. 6, p. 1373–1396, jun. 2003.

BELKIN, M.; NIYOGI, P. Convergence of laplacian eigenmaps. In: SCHöLKOPF, B.; PLATT, J. C.; HOFFMAN, T. (Ed.). *Advances in Neural Information Processing Systems 19*. [S.l.]: MIT Press, 2007. p. 129–136.

BELL, A. J.; SEJNOWSKI, T. J. An information maximization approach to blind separation and blind deconvolution. *Neural Computation*, v. 7, p. 1129–1159, 1995.

BERNSTEIN, M. et al. Graph approximations to geodesics on embedded manifolds. 2000.

BESAG, J. Spatial interaction and the statistical analysis of lattice systems. *Journal of the Royal Statistical Society. Series B (Methodological)*, [Royal Statistical Society, Wiley], v. 36, n. 2, p. 192–236, 1974. ISSN 00359246. Disponível em: ⟨http://www.jstor.org/stable/2984812⟩.

BHARATI, M. H.; LIU, J.; MACGREGOR, J. F. Image texture analysis: methods and comparisons. *Chemometrics and Intelligent Laboratory Systems*, v. 72, n. 1, p. 57 – 71, 2004. ISSN 0169-7439. Disponível em: ⟨http://www.sciencedirect.com/science/article/pii/S0169743904000528⟩.

BO, L.; YAN-RUI, L.; XIAO-LONG, Z. A survey on laplacian eigenmaps based manifold learning methods. *Neurocomputing*, v. 335, p. 336–351, 2018.

BORG, I.; GROENEN, P. *Modern Multidimensional Scaling: theory and applications*. 2. ed. [S.l.]: Springer-Verlag, 2005.

BRAND, M. Charting a manifold. In: ____. *Advances in Neural Information Processing Systems*. [S.l.]: MIT Press, 2003. v. 15, p. 961–968.

BRONSTEIN, A.; BRONSTEIN, M.; KIMMEL, R. *Numerical Geometry of Non-Rigid Shapes*. 1. ed. [S.l.]: Springer Publishing Company, Incorporated, 2008. ISBN 0387733000.

BROUWER, A. E.; HAEMERS, W. H. (Ed.). *Spectra of Graphs*. [S.l.]: Springer, 2011.

BRUNA, J.; MALLAT, S. Invariant scattering convolution networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 35, n. 8, p. 1872–1886, 2013.

Caputo, B.; Hayman, E.; Mallikarjuna, P. Class-specific material categorisation. In: *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*. [S.l.: s.n.], 2005. v. 2, p. 1597–1604 Vol. 2.

CAYTON, L. *Algorithms for Manifold Learning*. [S.l.], 2005.

CHAN, T. et al. Pcanet: A simple deep learning baseline for image classification? *IEEE Transactions on Image Processing*, v. 24, n. 12, p. 5017–5032, 2015.

CHOI, H.; CHOI, S. Robust kernel isomap. *Pattern Recognition*, v. 40, n. 3, p. 853–862, 2007.

CHOLLET, F. *Deep Learning with Python*. 1st. ed. USA: Manning Publications Co., 2017. ISBN 1617294438.

CHOULAKIAN, V. L1-norm projection pursuit principal component analysis. *Computational Statistics and Data Analysis*, v. 50, n. 6, p. 1441–1451, 2006.

CHUNG, F. R. K. (Ed.). *Spectral Graph Theory*. [S.l.]: American Mathematical Society, 1997.

CIMPOI, M. et al. Deep filter banks for texture recognition, description, and segmentation. *Int. J. Comput. Vision*, Kluwer Academic Publishers, USA, v. 118, n. 1, p. 65–94, maio 2016. ISSN 0920-5691. Disponível em: ⟨https://doi.org/10.1007/s11263-015-0872-3⟩.

COHEN, J. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, v. 20, n. 1, p. 37–46, 1960. Disponível em: ⟨https://doi.org/10.1177/001316446002000104⟩.

CONGALTON, R. G. A review of assessing the accuracy of classifications of remotely sensed data. *Remote Sensing of Environment*, v. 37, n. 1, p. 35 – 46, 1991. ISSN 0034-4257. Disponível em: ⟨http://www.sciencedirect.com/science/article/pii/003442579190048B⟩.

CONNERS, R. W.; HARLOW, C. A. A theoretical comparison of texture algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-2, n. 3, p. 204–222, May 1980.

CORMEN, T. H. et al. *Introduction to Algorithms*. 3. ed. [S.l.]: MIT Press, 2009. 1312 p p.

COX, T. F.; COX, M. A. A. *Multidimensional Scaling*. [S.l.]: Chapman & Hall, 2001. v. 88. 295 p p. (Monographs on Statistics and Applied Probability, v. 88).

CROSS, G. R.; JAIN, A. K. Markov random field texture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-5, p. 25–39, 1983.

CUNNINGHAM, J. P.; GHAHRAMANI, Z. Linear dimensionality reduction: Survey, insights, and generalizations. *Journal of Machine Learning Research*, v. 16, p. 2859–2900, 2015.

DALAL, N.; TRIGGS, B. Histograms of oriented gradients for human detection. In: *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*. [S.l.: s.n.], 2005. v. 1, p. 886–893 vol. 1. ISSN 1063-6919.

DOUGHERTY, G. Contents. In: ____. *Digital Image Processing for Medical Applications*. [S.l.]: Cambridge University Press, 2009. p. v–viii.

DUNTEMAN, G. H. Principal components analysis. In: ____. *Principal components analysis*. [S.l.]: Sage Publications, No. 69, Newbury Road, CA, USA, 1989, 1989. ISBN 9780803931046.

FIEDLER, M. Laplacian of graphs and algebraic connectivity. *Banach Center Publications*, v. 25, n. 1, p. 57–70, 1989.

FUJIEDA, S.; TAKAYAMA, K.; HACHISUKA, T. Wavelet Convolutional Neural Networks for Texture Classification. *arXiv e-prints*, p. arXiv:1707.07394, Jul 2017.

FUKUNAGA, K. *Introduction to Statistical Pattern Recognition*. [S.l.]: Academic Press, 1990.

GALPIN, J. S.; HAWKINS, D. M. Methods of l1 estimation of a covariance matrix. *Computational Statistics and Data Analysis*, v. 5, p. 305–319, 1987.

GEBEJES, A.; HUERTAS, R. Texture characterization based on grey-level co-occurrence matrix. *International Conference on Information and Communication Technologies*, p. 375–378, 01 2013.

GONZALEZ, R. C.; WOODS, R. E. *Processamento Digital de Imagens*. 3. ed. São Paulo: Pearson / Prentice Hall, 2010.

GOOL, L. V.; DEWAELE, P.; OOSTERLINCK, A. Texture analysis anno 1983. *Computer Vision, Graphics, and Image Processing*, v. 29, n. 3, p. 336 – 357, 1985. ISSN 0734-189X. Disponível em: ⟨http://www.sciencedirect.com/science/article/pii/0734189X85901306⟩.

HARALICK, R. M. Statistical and structural approaches to texture. *Proceedings of the IEEE*, v. 67, n. 5, p. 786–804, May 1979. ISSN 1558-2256.

HARALICK, R. M.; SHANMUGAM, K.; DINSTEIN, I. Textural features for image classification. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-3, n. 6, p. 610–621, Nov 1973. ISSN 2168-2909.

HASTIE, T.; TIBSHIRANI, R.; FRIEDMAN, J. *The Elements of Statistical Learning*. 2. ed. [S.l.]: Springer, 2009.

HE, K. et al. Deep residual learning for image recognition. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.], 2016. p. 770–778.

HOCHBAUM, D. S.; SHMOYS, D. B. A best possible heuristic for the k-center problem. *Math. Oper. Res.*, INFORMS, Linthicum, MD, USA, v. 10, n. 2, p. 180–184, maio 1985. ISSN 0364-765X. Disponível em: ⟨https://doi.org/10.1287/moor.10.2.180⟩.

HUANG, G.; LIU, Z.; WEINBERGER, K. Q. Densely connected convolutional networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, p. 2261–2269, 2016.

HYVANINEN, A. Fast and robust fixed-point algorithms for independent component analysis. *IEEE Transactions on Neural Networks*, v. 10, n. 3, p. 626–634, 1999.

HYVARINEN, A.; KARHUNEN, J.; OJA, E. *Independent Component Analysis*. [S.l.]: John Wiley & Sons, 2001.

ISSERLIS, L. ON A FORMULA FOR THE PRODUCT-MOMENT COEFFICIENT OF ANY ORDER OF A NORMAL FREQUENCY DISTRIBUTION IN ANY NUMBER OF VARIABLES. *Biometrika*, v. 12, n. 1-2, p. 134–139, 11 1918. ISSN 0006-3444. Disponível em: ⟨https://doi.org/10.1093/biomet/12.1-2.134⟩.

JAIN, A. *Fundamentals of Digital Image Processing*. Prentice Hall, 1989. (Prentice-Hall information and system sciences series). ISBN 9780133361650. Disponível em: ⟨https://books.google.com.br/books?id=GANSAAAAMAAJ⟩.

JIANG, L.; RICH, W.; BUHL-BROWN, D. Texture analysis of remote sensing imagery with clustering and bayesian inference. *International Journal of Image, Graphics and Signal Processing*, v. 7, p. 1–10, 08 2015.

JOLLIFFE, I. T. *Principal Component Analysis*. 2. ed. [S.l.]: Springer, 2002. 487 pages p.

JOURNEE, M. et al. Generalized power method for sparse principal component analysis. *Journal of Machine Learning Research*, v. 11, p. 517–553, 2010.

JULESZ, B. Experiments in the visual perception of texture. PubMed, v. 232, p. 34–43, 1975. ISSN 0036-8733.

JULESZ, B. Textons, the elements of texture perception, and their interactions. Nature, v. 290, p. 91–97, 1981. ISSN 1476-4687.

KELLER, J. M.; CHEN, S.; CROWNOVER, R. M. Texture description and segmentation through fractal geometry. *Computer Vision, Graphics, and Image Processing*, v. 45, n. 2, p. 150 – 166, 1989. ISSN 0734-189X. Disponível em: ⟨http://www.sciencedirect.com/science/article/pii/0734189X89901308⟩.

KHAN, F. S.; WEIJER, J. V. de; VANRELL, M. Top-down color attention for object recognition. *2009 IEEE 12th International Conference on Computer Vision*, p. 979–986, 2009.

KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. Imagenet classification with deep convolutional neural networks. In: PEREIRA, F. et al. (Ed.). *Advances in Neural Information Processing Systems 25*. Curran Associates, Inc., 2012. p. 1097–1105. Disponível em: ⟨http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf⟩.

KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. Imagenet classification with deep convolutional neural networks. In: PEREIRA, F. et al. (Ed.). *Advances in Neural Information Processing Systems 25*. Curran Associates, Inc., 2012. p. 1097–1105. Disponível em: ⟨http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf⟩.

KWITT, R.; MEERWALD, P. Salzburg texture image database. 2018. Available at ⟨http://www.wavelab.at/sources/STex/⟩, last viewed February 2018.

LEE, D. D.; SEUNG, H. S. Learning the parts of objects by non-negative matrix factorization. *Nature*, v. 401, p. 788–791, 1999.

LEE, T. W.; GIROLAMI, M.; SEJNOWSKI, T. J. Independent component analysis using an extended infomax algorithm for mixed subgaussian and supergaussian sources. *Neural Computation*, v. 11, n. 2, p. 626–634, 1999.

LERSKI, R. A. et al. Mr image texture analysis–an approach to tissue characterization. *Magnetic resonance imaging*, v. 11 6, p. 873–87, 1993.

LEUNG, T.; MALIK, J. Representing and recognizing the visual appearance of materials using three-dimensional textons. *International Journal of Computer Vision*, v. 43, n. 1, p. 29–44, Jun 2001. ISSN 1573-1405. Disponível em: ⟨https://doi.org/10.1023/A:1011126920638⟩.

LEVADA, A. L. M. On the spatial dependence structure of isotropic pairwise gaussian-markov random field models. *CoRR*, abs/1108.4973, 2011. Disponível em: ⟨http://arxiv.org/abs/1108.4973⟩.

LEVADA, A. L. M. Information geometry, simulation and complexity in gaussian random fields. *CoRR*, abs/1703.04464, 2017. Disponível em: ⟨http://arxiv.org/abs/1703.04464⟩.

LEVINE, M. *Vision in man and machine*. McGraw-Hill, 1985. (McGraw-Hill series in electrical engineering: Computer engineering). ISBN 9780070374461. Disponível em: ⟨https://books.google.com.br/books?id=XAVSAAAAMAAJ⟩.

LI, W.; FRITZ, M. Recognizing materials from virtual examples. In: FITZGIBBON, A. et al. (Ed.). *Computer Vision – ECCV 2012*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012. p. 345–358. ISBN 978-3-642-33765-9.

LIU, L.; FIEGUTH, P. Texture classification from random features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 34, n. 3, p. 574–586, March 2012.

LIU, L. et al. Extended local binary patterns for texture classification. *Image and Vision Computing*, v. 30, n. 2, p. 86 – 99, 2012. ISSN 0262-8856. Disponível em: ⟨http://www.sciencedirect.com/science/article/pii/S0262885612000066⟩.

LOWE, D. G. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, v. 60, n. 2, p. 91–110, Nov 2004. ISSN 1573-1405. Disponível em: ⟨https://doi.org/10.1023/B:VISI.0000029664.99615.94⟩.

LUXBURG, U. von. A tutorial on spectral clustering. *Statistics and Computing*, v. 17, p. 395–416, 2007.

MANDELBROT, B. *How Long is the Coast of Britain?* Armando Siciliano Editore, 2007. ISBN 9788874424474. Disponível em: ⟨https://books.google.com.br/books?id=MgtfmAEACAAJ⟩.

MANDELBROT, B.; FREEMAN, W.; COMPANY. *The Fractal Geometry of Nature*. Henry Holt and Company, 1983. (Einaudi paperbacks). ISBN 9780716711865. Disponível em: ⟨https://books.google.com.br/books?id=0R2LkE3N7-oC⟩.

MATERKA, A.; STRZELECKI, M. *Texture analysis methods – a review*. [S.l.], 1998.

MCCLURKIN, J.; OPTICAN, B. L.; GAWNE, T. Concurrent processing and complexity of temporally encoded neuronal messages in visual perception. *Science*, v. 253, p. 675–677, 1991.

MIEGHEM, P. van. *Graph Spectra for Complex Networks*. [S.l.]: Cambridge University Press, 2010.

MIRMEHDI, M.; XIE, X.; SURI, J. *Handbook of Texture Analysis*. PUBLISHED BY IMPERIAL COLLEGE PRESS AND DISTRIBUTED BY WORLD SCIENTIFIC PUBLISHING CO., 2008. Disponível em: ⟨https://www.worldscientific.com/doi/abs/10.1142/p547⟩.

MOHAR, B. The laplacian spectrum of graphs. In: *Graph Theory, Combinatorics, and Applications*. [S.l.]: Wiley, 1991. p. 871–898.

MURASE, H.; NAYAR, S. Visual learning and recognition of 3d objects from appearance. *International Journal Computer Vision*, v. 14, p. 5–24, 1995.

MUSGRAVE, F. K. et al. *Texturing and Modeling: A Procedural Approach*. San Diego, CA, USA: Academic Press Professional, Inc., 1994. ISBN 0-12-228760-6.

MäENPää, T.; PIETIKäINEN, M. Texture analysis with local binary patterns. In: ____. *Handbook of Pattern Recognition and Computer Vision*. [s.n.]. p. 197–216. Disponível em: ⟨https://www.worldscientific.com/doi/abs/10.1142/9789812775320_0011⟩.

NICA, B. *A Brief Introduction to Spectral Graph Theory*. [S.l.]: American Mathematical Society, 2018.

NILSBACK, M.; ZISSERMAN, A. Automated flower classification over a large number of classes. In: *2008 Sixth Indian Conference on Computer Vision, Graphics Image Processing*. [S.l.: s.n.], 2008. p. 722–729.

OHANIAN, P. P.; DUBES, R. C. Performance evaluation for four classes of textural features. *Pattern Recognition*, v. 25, n. 8, p. 819 – 833, 1992. ISSN 0031-3203. Disponível em: ⟨http://www.sciencedirect.com/science/article/pii/003132039290036I⟩.

OJALA, T.; PIETIKAINEN, M.; HARWOOD, D. Performance evaluation of texture measures with classification based on kullback discrimination of distributions. In: *Proceedings of 12th International Conference on Pattern Recognition*. IEEE Comput. Soc. Press. Disponível em: ⟨https://doi.org/10.1109\%2Ficpr.1994.576366⟩.

OJALA, T.; PIETIKAINEN, M.; MAENPAA, T. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 24, n. 7, p. 971–987, July 2002.

OJALA T. MAENPAA, M. P. J. V. J. K. T.; HUOVINEN, S. Outex – new framework for empirical evaluation of texture analysis algorithms. In: . [S.l.: s.n.], 2002. ICPR/1, p. 701–706.

ONG, S. et al. Image analysis of tissue sections. *Computers in Biology and Medicine*, v. 26, n. 3, p. 269 – 279, 1996. ISSN 0010-4825. Information Retrieval and Genomics. Disponível em: ⟨http://www.sciencedirect.com/science/article/pii/0010482596000042⟩.

PAI, G. et al. Dimal: Deep isometric manifold learning using sparse geodesic sampling. In: . [S.l.: s.n.], 2018.

PHAM, D.-T. Fast algorithms for mutual information based independent component analysis. *IEEE Transactions on Signal Processing*, v. 52, n. 10, p. 2690–2700, 2004.

PIETIKÄINEN, M. et al. *Computer Vision Using Local Binary Patterns*. Springer London, 2011. (Computational Imaging and Vision). ISBN 9780857297488. Disponível em: ⟨https://books.google.com.br/books?id=wBrZz9FiERsC⟩.

PIETIKäINEN, M. et al. *Computer Vision Using Local Binary Patterns*. [S.l.: s.n.], 2011. v. 40.

PIETIKäINEN, M.; MäENPää, T.; VIERTOLA, J. Color texture classification with color histograms and local binary patterns. *Workshop on Texture Analysis in Machine Vision*, 01 2002.

PONTI, M. A.; COSTA, G. B. P. d. Como funciona o deep learning. In: ____. *Brazilian Symposium on Databases - SBBD*. [S.l.]: SBC, 2017.

QI, X. et al. Pairwise rotation invariant co-occurrence local binary pattern. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 36, n. 11, p. 2199–2213, Nov 2014. ISSN 0162-8828.

RANDEN, T.; HUSØY, J. H. Filtering for texture classification: A comparative study. *IEEE Trans. Pattern Anal. Mach. Intell.*, IEEE Computer Society, Washington, DC, USA, v. 21, n. 4, p. 291–310, abr. 1999. ISSN 0162-8828. Disponível em: ⟨https://doi.org/10.1109/34.761261⟩.

RAO, A. *A Taxonomy for Texture Description and Identification*. Springer, 1990. (Springer series in perception engineering). ISBN 9783540973027. Disponível em: ⟨https://books.google.com.br/books?id=O-aruwEACAAJ⟩.

REED, T.; DUBUF, J. A review of recent texture segmentation and feature extraction techniques. *CVGIP: Image Understanding*, v. 57, n. 3, p. 359 – 372, 1993. ISSN 1049-9660. Disponível em: ⟨http://www.sciencedirect.com/science/article/pii/S1049966083710247⟩.

RIDDER, D. de; DUIN, R. P. *Locally Linear Embedding for Classification*. [S.l.], 2002.

RIOUL, O.; VETTERLI, M. Wavelets and signal processing. *IEEE Signal Processing Magazine*, v. 8, n. 4, p. 14–38, 1991.

ROWEIS, S.; SAUL, L. Nonlinear dimensionality reduction by locally linear embedding. *Science*, v. 290, p. 2323–2326, 2000.

ROWEIS, S. T. Em algorithms for pca and sensible pca. In: *Advances in Neural Information Processing Systems*. [S.l.: s.n.], 1997.

SANDE, K. V. de; GEVERS, T.; SNOEK, C. Evaluating color descriptors for object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 32, n. 9, p. 1582–1596, Sep. 2010. ISSN 0162-8828.

SANTAFE, G.; INZA, I.; LOZANO, J. A. Dealing with the evaluation of supervised classification algorithms. *Artificial Intelligence Review*, v. 44, n. 4, p. 467–508, Dec 2015. ISSN 1573-7462. Disponível em: ⟨https://doi.org/10.1007/s10462-015-9433-y⟩.

SAUL, L.; ROWEIS, S. Think globally, fit locally: Unsupervised learning of low dimensional manifolds. *Journal of Machine Learning Research*, v. 4, p. 119–155, 2003.

SAXENA, A.; GUPTA, A.; MUKERJEE, A. Non-linear dimensionality reduction by locally linear isomaps. *Lecture Notes in Computer Science*, v. 3316, p. 1038–1043, 2004.

SEUNG, H. S.; LEE, D. D. The manifold ways of perception. *Science*, v. 290, p. 2268–2269, 2000.

SHALIZI, C. *Nonlinear Dimensionality Reduction I: Local Linear Embedding*. 2009. Http://www.stat.cmu.edu/ cshalizi/350/lectures/14/lecture-14.pdf.

SHARAN, L. et al. Recognizing materials using perceptually inspired features. *International Journal of Computer Vision*, v. 103, n. 3, p. 348–371, Jul 2013. ISSN 1573-1405. Disponível em: ⟨https://doi.org/10.1007/s11263-013-0609-0⟩.

SILVA, V.; TENENBAUM, J. Sparse multidimensional scaling using landmark points. *Technology*, 01 2004.

SILVA, V. de; TENENBAUM, J. B. Global versus local methods in nonlinear dimensionality reduction. In: *Proceedings of the 15th International Conference on Neural Information Processing Systems (NIPS'02)*. [S.l.: s.n.], 2002. p. 721–728.

SIMONYAN, K.; ZISSERMAN, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv e-prints*, p. arXiv:1409.1556, Sep 2014.

SPIELMAN, D. A. Spectral graph theory and its applications. In: *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07)*. [S.l.: s.n.], 2007. p. 29–38.

SPRINGENBERG, J. et al. Striving for simplicity: The all convolutional net. In: *ICLR (workshop track)*. [s.n.], 2015. Disponível em: ⟨http://lmb.informatik.uni-freiburg.de/Publications/2015/DB15a⟩.

SUN, Y. et al. Cost-sensitive boosting for classification of imbalanced data. *Pattern Recognition*, v. 40, n. 12, p. 3358 – 3378, 2007. ISSN 0031-3203. Disponível em: ⟨http://www.sciencedirect.com/science/article/pii/S0031320307001835⟩.

SZEGEDY, C. et al. Going deeper with convolutions. In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.], 2015. p. 1–9.

TENENBAUM, J. B.; SILVA, V. de; LANGFORD, J. C. A global geometric framework for nonlinear dimensionality reduction. *Science*, v. 290, p. 2319–2323, 2000.

TIPPING, M. E.; BISHOP, C. M. Probabilistic principal component analysis. *Journal of the Royal Statistical Society, Series B*, v. 61, n. 3, p. 611–622, 1999.

TUCERYAN, M.; JAIN, A. K. Texture analysis. In: ____. *Handbook of Pattern Recognition and Computer Vision*. [s.n.]. p. 235–276. Disponível em: ⟨https://www.worldscientific.com/doi/abs/10.1142/9789814343138_0010⟩.

W. LAM N. N., Q. D. A. E. C. Multi-scale fractal analysis of image texture and patterns. Photogrammetric engineering and remote sensing, 65, p. 51–62, 1999.

WAKIN, M. B. Sparse image and signal processing: Wavelets, curvelets, morphological diversity (starck, j.-l., et al; 2010) [book reviews]. *IEEE Signal Processing Magazine*, v. 28, n. 5, p. 144–146, 2011.

WESZKA, J. S.; DYER, C. R.; ROSENFELD, A. A comparative study of texture measures for terrain classification. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-6, n. 4, p. 269–285, April 1976.

WILCOXON, F. Individual comparisons by ranking methods. In: ____. *Breakthroughs in Statistics: Methodology and Distribution*. New York, NY: Springer New York, 1992. p. 196–202. ISBN 978-1-4612-4380-9. Disponível em: ⟨https://doi.org/10.1007/978-1-4612-4380-9_16⟩.

YOUNG, T. Y.; CALVERT, T. W. *Classification, Estimation and Pattern Recognition*. [S.l.]: Elsevier, 1974.

ZHANG, J.; TAN, T. Brief review of invariant texture analysis methods. *Pattern Recognition*, v. 35, p. 735–747, 03 2002.

Zhao, Y. et al. Classification of high spatial resolution imagery using improved gaussian markov random-field-based texture features. *IEEE Transactions on Geoscience and Remote Sensing*, v. 45, n. 5, p. 1458–1468, May 2007. ISSN 1558-0644.

ZOU, H.; HASTIE, T.; TIBSHIRANI, R. Sparse principal component analysis. *Journal of Computational and Graphical Statistics*, v. 15, n. 2, p. 265–286, 2006.

GLOSSÁRIO

CNN – *Convolutional Neural Network*

DIMAL – *Deep Isometric Manifold Learning Using Sparse Geodesic Sampling*

DWT – *Discret Wavelet Transform*

FC – *Fully Connected*

GLCM – *Gray Level Co-occurence Matrix*

GMRF – *Gaussian Markov Random Field*

HOG – *Histogram of Oriented Gradients*

ISOMAP – *Isometric Feature Mapping*

KNN – *K-Nearest-Neighbor*

LBP – *Local Binary Pattern*

LLE – *Local Linear Embedding*

Lap. Eig. – *Laplacian Eigenmaps*

MLP – *Multi-Layer Perceptron*

MRF – *Markov Random Field*

PCA – *Principal Component Analysis*

SVM – *Support Vector Machine*

Apendice A

DERIVAÇÕES MATEMÁTICAS DO PARÂMETRO β

Seja $p(x_i|\eta_i,\vec{\theta})$ a função densidade de probabilidade condicional local de um GMRF, conforme definido na equação (3.13). Dessa forma, substituindo $p(x;\vec{\theta})$ por $p(x_i|\eta_i,\vec{\theta})$ em (3.21) a nova expressão da matriz de informação de Fisher torna-se

$$\begin{aligned}\left\{I(\vec{\theta})\right\}_{ij} &= E\left[\left(\frac{\partial}{\partial\theta_i}\log p(x_i|\eta_i,\vec{\theta})\right)\left(\frac{\partial}{\partial\theta_j}\log p(x_i|\eta_i,\vec{\theta})\right)\right] \\ &= -E\left[\frac{\partial^2}{\partial\theta_i\partial\theta_j}\log p(x_i|\eta_i,\vec{\theta})\right] \; para\; i,j=1,...,n.\end{aligned} \tag{A.1}$$

A função pseudo-verossimilhança é definida como produto das funções densidade de probabilidade condicional local, conhecidas como LCDF's *(Local Conditional Density Functions)* em termo inglês e é dada por

$$L\left(\vec{\theta};\mathbf{X}\right) = \prod_{i=1}^{n} p(x_i|\eta_i,\vec{\theta}). \tag{A.2}$$

Inserindo a equação (3.13) dentro da equação (A.2) resulta em

$$log\, L\left(\vec{\theta};\mathbf{X}\right) = -\frac{n}{2}log\left(2\pi\sigma^2\right) - \frac{1}{2\sigma^2}\sum_{i=1}^{n}\left[x_i-\mu-\beta\sum_{j\in\eta_i}\left(x_j-\mu\right)\right]^2. \tag{A.3}$$

Ao diferenciar a equação (A.3) em relação a β e ao resolver adequadamente a equação de pseudo-verossimilhança, obtemos o seguinte estimador para o parâmetro inversa da temperatura:

$$\hat{\beta}_{MPL} = \frac{\sum_{i=1}^{n} \left[(x_i - \mu) \sum_{j \in \eta_i} (x_j - \mu) \right]}{\sum_{i=1}^{n} \left[\sum_{j \in \eta_i} (x_j - \mu) \right]^2} = \frac{\sum_{j \in \eta_i} \sigma_{ij}}{\sum_{j \in \eta_i} \sum_{k \in \eta_i} \sigma_{jk}} \tag{A.4}$$

Usando a notação matricial de covariância introduzida no Capítulo 4, temos

$$\hat{\beta}_{MPL} = \frac{\|\vec{\rho}\|_+}{\|\Sigma_p^-\|_+} \tag{A.5}$$

onde $\|A\|_+$ indica a soma de todas as entradas do vetor / matriz A.

Apendice B

DERIVAÇÕES MATEMÁTICAS DOS COMPONENTES DA MATRIZ DA INFORMAÇÃO DE FISHER

B.1 Derivação de $I^{(1)}_{\mu\mu}(\vec{\theta})$

A partir da equação (A.1), pode-se escrever:

$$\begin{aligned} I^{(1)}_{\mu\mu}(\vec{\theta}) &= E\left[\left(\frac{\partial}{\partial\mu}\log p(x_i|\eta_i,\vec{\theta})\right)\left(\frac{\partial}{\partial\mu}\log p(x_i|\eta_i,\vec{\theta})\right)\right] \\ &= E\left[\left(\frac{\partial}{\partial\mu}\log p(x_i|\eta_i,\vec{\theta})\right)^2\right]. \end{aligned} \tag{B.1}$$

Inserindo a equação (3.13) dentro da equação (B.1) resulta em

$$I^{(1)}_{\mu\mu}(\vec{\theta}) = E\left[\left(\frac{\partial}{\partial\mu}\log\left(\frac{1}{\sqrt{2\pi\sigma^2}}\exp\left\{-\frac{1}{2\sigma^2}\left[x_i-\mu-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]^2\right\}\right)\right)^2\right]. \tag{B.2}$$

Aplicando algumas propriedades matemáticas e estatísticas básicas e, em seguida, calculando as respectivas derivadas leva a expressão final do $I^{(1)}_{\mu\mu}(\vec{\theta})$.

$$
\begin{aligned}
I^{(1)}_{\mu\mu}(\vec{\theta}) &= E\left[\underbrace{\frac{\partial}{\partial\mu}\log\frac{1}{\sqrt{2\pi\sigma^2}}}_{0} + \frac{\partial}{\partial\mu}\left\{-\frac{1}{2\sigma^2}\left[x_i-\mu-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]^2\right\}\right]^2 \\
&= E\left\{-\frac{1}{2\sigma^2}2\frac{\partial}{\partial\mu}\left[x_i-\mu-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]\left[x_i-\mu-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]\right\}^2 \\
&= E\left\{-\frac{1}{\sigma^2}\left[-1+\beta\underbrace{\sum_{j\in\eta_i}1}_{\eta_i=\Delta}\right]\left[\underbrace{(x_i-\mu)}_{a}-\underbrace{\beta\sum_{j\in\eta_i}(x_j-\mu)}_{b}\right]\right\}^2 \\
&= E\left\{\frac{1}{\sigma^2}(1-\beta\Delta)^2\frac{1}{\sigma^2}\left[(x_i-\mu)^2-2\beta\sum_{j\in\eta_i}(x_i-\mu)(x_j-\mu)\right.\right. \\
&\qquad\left.\left.+\beta^2\sum_{j\in\eta_i}\sum_{k\in\eta_i}(x_j-\mu)(x_k-\mu)\right]\right\} \\
&= \frac{(1-\beta\Delta)^2}{\sigma^2}\left[\underbrace{\frac{E(x_i-\mu)^2}{\sigma^2}}_{1}-\frac{1}{\sigma^2}\left(2\beta\sum_{j\in\eta_i}\sigma_{ij}-\beta^2\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sigma_{jk}\right)\right] \\
&= \frac{(1-\beta\Delta)^2}{\sigma^2}\left[1-\frac{1}{\sigma^2}\left(2\beta\sum_{j\in\eta_i}\sigma_{ij}-\beta^2\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sigma_{jk}\right)\right]
\end{aligned}
\tag{B.3}
$$

onde Δ denota o suporte do sistema de vizinhança (no nosso caso $\Delta = 8$, pois temos um sistema de segunda ordem), $\sigma_i j$ denota a covariância entre a variável central x_i e um de seus vizinhos $x_j \in \eta_i$ e $\sigma_j k$ denota a covariância entre duas variáveis x_j e x_k na vizinhança η_i.

B.2 Derivação de $I^{(1)}_{\mu\sigma^2}(\vec{\theta})$

A partir da equação (A.1), pode-se escrever $I^{(1)}_{\mu\sigma^2}(\vec{\theta})$ como

$$
I^{(1)}_{\mu\sigma^2}(\vec{\theta}) = E\left[\left(\frac{\partial}{\partial\mu}\log p(x_i|\eta_i,\vec{\theta})\right)\left(\frac{\partial}{\partial\sigma^2}\log p(x_i|\eta_i,\vec{\theta})\right)\right] \tag{B.4}
$$

Inserindo a equação (3.13) dentro da equação (B.4) tem-se

$$I^{(1)}_{\mu\sigma^2}(\vec{\theta}) = E\left[\left(\frac{\partial}{\partial\mu}\log\left(\frac{1}{\sqrt{2\pi\sigma^2}}\exp\left\{-\frac{1}{2\sigma^2}\left[x_i-\mu-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]^2\right\}\right)\right)\right.$$
$$\left.\left(\frac{\partial}{\partial\sigma^2}\log\left(\frac{1}{\sqrt{2\pi\sigma^2}}\exp\left\{-\frac{1}{2\sigma^2}\left[x_i-\mu-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]^2\right\}\right)\right)\right] \tag{B.5}$$

que, depois de algumas expansões matemáticas, resulta em

$$\begin{aligned}
I^{(1)}_{\mu\sigma^2}(\vec{\theta}) &= E\left[\left(\underbrace{\frac{\partial}{\partial\mu}\log\frac{1}{\sqrt{2\pi\sigma^2}}}_{0}+\frac{\partial}{\partial\mu}\left\{-\frac{1}{2\sigma^2}\left[x_i-\mu-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]^2\right\}\right)\right.\\
&\qquad\left.\left(\frac{\partial}{\partial\sigma^2}\log\frac{1}{\sqrt{2\pi\sigma^2}}+\frac{\partial}{\partial\sigma^2}\left\{-\frac{1}{2\sigma^2}\left[x_i-\mu-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]^2\right\}\right)\right]\\
&= E\left[\left\{-\frac{1}{2\sigma^2}2\frac{\partial}{\partial\mu}\left[x_i-\mu-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]\left[x_i-\mu-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]\right\}\right.\\
&\qquad\left.\left\{-\frac{1}{2}\frac{1}{\sigma^2}+\frac{1}{2}\frac{1}{\sigma^4}\left[x_i-\mu-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]^2\right\}\right]\\
&= E\left[\left\{-\frac{1}{\sigma^2}\left[-1+\beta\underbrace{\sum_{j\in\eta_i}1}_{\eta_i=\Delta}\right]\left[(x_i-\mu)-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]\right\}\right.\\
&\qquad\left.\left\{-\frac{1}{2}\frac{1}{\sigma^2}+\frac{1}{2}\frac{1}{\sigma^4}\left[x_i-\mu-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]^2\right\}\right]\\
&= E\left[\underbrace{\left\{\frac{1}{\sigma^2}(1-\beta\Delta)\left[(x_i-\mu)-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]\right\}}_{\text{a ser multiplicado por a e b}}\right.\\
&\qquad\left.\left\{\underbrace{-\frac{1}{2}\frac{1}{\sigma^2}}_{\text{a}}+\underbrace{\frac{1}{2}\frac{1}{\sigma^4}\left[(x_i-\mu)-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]^2}_{\text{b}}\right\}\right]\\
&= E\left\{-\frac{1}{2\sigma^4}(1-\beta\Delta)\left[(x_i-\mu)-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]\right.\\
&\qquad\left.+\frac{1}{2\sigma^6}(1-\beta\Delta)+\left[(x_i-\mu)-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]^3\right\}\\
&= \frac{(1-\beta\Delta)}{2\sigma^6}\underbrace{E\left\{\left[(x_i-\mu)-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]^3\right\}}_{e_1}\\
&\qquad-\frac{(1-\beta\Delta)}{2\sigma^4}\underbrace{E\left\{(x_i-\mu)-\beta\sum_{j\in\eta_i}(x_j-\mu)\right\}}_{e_2}
\end{aligned} \tag{B.6}$$

Devido à linearidade do operador valor esperado e levando em conta que $E[X] = \mu$, e_2 torna-se

$$\begin{aligned} e_2 &= E[x_i - \mu] - \beta \sum_{j \in \eta_i} E[x_j - \mu] \\ &= E[x_i] - \mu - \beta \sum_{j \in \eta_i} E[x_j] - \mu \\ &= 0 - 0 = 0. \end{aligned}$$

Vamos expandir a expressão de e_1 da seguinte maneira

$$\begin{aligned} e_1 &= E\left\{ \left[(x_i - \mu) - \beta \sum_{j \in \eta_i} (x_j - \mu) \right]^3 \right\} \\ &= E\left[(x_i - \mu)^3 \right] - 3\beta \sum_{j \in \eta_i} E\left[(x_i - \mu)(x_i - \mu)(x_j - \mu) \right] \\ &\quad + 3\beta^2 \sum_{j \in \eta_i} \sum_{k \in \eta_i} E\left[(x_i - \mu)(x_j - \mu)(x_k - \mu) \right] \\ &\quad - \beta^3 \sum_{j \in \eta_i} \sum_{k \in \eta_i} \sum_{l \in \eta_i} E\left[(x_j - \mu)(x_k - \mu)(x_l - \mu) \right] \end{aligned}$$

Sabe-se que para variáveis aleatórias Gaussianas os momentos centrais são definidos como

$$E[(x-\mu)^p] = \begin{cases} 0 & \text{se } p \text{ é ímpar} \\ \sigma^p (p-1)!! & \text{se } p \text{ é par.} \end{cases}$$

Aqui $n!!$ indica o duplo fatorial ou semifatorial, ou seja, o produto de todos os números de 1 até n que têm a mesma paridade que n. Levando em consideração essa suposição e de acordo com o teorema de Isserlis (ISSERLIS, 1918), e_1 torna-se zero. Consequentemente, inserindo e_1 e e_2 na equação (B.6) leva a

$$I^{(1)}_{\mu\sigma^2}(\vec{\theta}) = 0.$$

B.3 Derivação de $I^{(1)}_{\mu\beta}(\vec{\theta})$

A expressão de $I^{(1)}_{\mu\beta}(\vec{\theta})$ é definida como

$$I^{(1)}_{\mu\beta}(\vec{\theta}) = E\left[\left(\frac{\partial}{\partial\mu}\log p(x_i|\eta_i,\vec{\theta})\right)\left(\frac{\partial}{\partial\beta}\log p(x_i|\eta_i,\vec{\theta})\right)\right] \tag{B.7}$$

Inserindo a equação (3.13) dentro da equação (B.7) enquanto manipula-se as expressões resultantes leva a

$$\begin{aligned}
I^{(1)}_{\mu\beta}(\vec{\theta}) &= E\Bigg[\Bigg(\underbrace{\frac{\partial}{\partial\mu}\log\frac{1}{\sqrt{2\pi\sigma^2}}}_{0} + \frac{\partial}{\partial\mu}\Bigg\{-\frac{1}{2\sigma^2}\Bigg[x_i-\mu-\beta\sum_{j\in\eta_i}(x_j-\mu)\Bigg]^2\Bigg\}\Bigg)\\
&\qquad\Bigg(\underbrace{\frac{\partial}{\partial\beta}\log\frac{1}{\sqrt{2\pi\sigma^2}}}_{0} + \frac{\partial}{\partial\beta}\Bigg\{-\frac{1}{2\sigma^2}\Bigg[x_i-\mu-\beta\sum_{j\in\eta_i}(x_j-\mu)\Bigg]^2\Bigg\}\Bigg)\Bigg]\\
&= E\Bigg[\Bigg\{-\frac{1}{2\sigma^2}2\frac{\partial}{\partial\mu}\Bigg[x_i-\mu-\beta\sum_{j\in\eta_i}(x_j-\mu)\Bigg]\Bigg[x_i-\mu-\beta\sum_{j\in\eta_i}(x_j-\mu)\Bigg]\Bigg\}\\
&\qquad\Bigg\{-\frac{1}{2\sigma^2}2\frac{\partial}{\partial\beta}\Bigg[x_i-\mu-\beta\sum_{j\in\eta_i}(x_j-\mu)\Bigg]\Bigg[x_i-\mu-\beta\sum_{j\in\eta_i}(x_j-\mu)\Bigg]\Bigg\}\Bigg]\\
&= E\Bigg[\Bigg\{-\frac{1}{\sigma^2}\Bigg[-1+\beta\underbrace{\sum_{j\in\eta_i}1}_{\eta_i=\Delta}\Bigg]\Bigg[(x_i-\mu)-\beta\sum_{j\in\eta_i}(x_j-\mu)\Bigg]\Bigg\}\\
&\qquad\Bigg\{-\frac{1}{\sigma^2}\Bigg[-\sum_{j\in\eta_i}(x_j-\mu)\Bigg]\Bigg[x_i-\mu-\beta\sum_{j\in\eta_i}(x_j-\mu)\Bigg]\Bigg\}\Bigg]\\
&= E\Bigg[\Bigg\{\frac{1}{\sigma^2}(1-\beta\Delta)\Bigg[(x_i-\mu)-\beta\sum_{j\in\eta_i}(x_j-\mu)\Bigg]\Bigg\}\\
&\qquad\Bigg\{\frac{1}{\sigma^2}\Bigg[\sum_{j\in\eta_i}(x_i-\mu)(x_j-\mu)-\beta\sum_{j\in\eta_i}\sum_{k\in\eta_i}(x_j-\mu)(x_k-\mu)\Bigg]\Bigg\}\Bigg]\\
&= E\Bigg\{\frac{(1-\beta\Delta)}{\sigma^4}\Bigg[\sum_{j\in\eta_i}(x_i-\mu)(x_i-\mu)(x_j-\mu)-2\beta\sum_{j\in\eta_i}\sum_{k\in\eta_i}(x_i-\mu)\\
&\qquad(x_j-\mu)(x_k-\mu)+\beta^2\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sum_{l\in\eta_i}(x_j-\mu)(x_k-\mu)(x_l-\mu)\Bigg]\Bigg\}\\
&= \frac{(1-\beta\Delta)}{\sigma^4}\Bigg\{\sum_{j\in\eta_i}E[(x_i-\mu)(x_i-\mu)(x_j-\mu)]-2\beta\sum_{j\in\eta_i}\sum_{k\in\eta_i}E[(x_i-\mu)\\
&\qquad(x_j-\mu)(x_k-\mu)]+\beta^2\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sum_{l\in\eta_i}E[(x_j-\mu)(x_k-\mu)(x_l-\mu)]\Bigg\}.
\end{aligned} \tag{B.8}$$

Como pode-se observar na equação (B.8), todos os momentos de ordem superior são produto de um número ímpar de variáveis aleatórias Gaussianas. Consequentemente, eles são iguais a zero pelo teorema de Isserlis, resultando em $I^{(1)}_{\mu\beta}(\vec{\theta}) = 0$.

B.4 Derivação de $I^{(1)}_{\sigma^2\mu}(\vec{\theta})$

A expressão de $I^{(1)}_{\sigma^2\mu}(\vec{\theta})$ é dada por

$$I^{(1)}_{\sigma^2\mu}(\vec{\theta}) = E\left[\left(\frac{\partial}{\partial\sigma^2}\log p(x_i|\eta_i,\vec{\theta})\right)\left(\frac{\partial}{\partial\mu}\log p(x_i|\eta_i,\vec{\theta})\right)\right] \tag{B.9}$$

Inserindo a equação (3.13) dentro da equação (B.9) enquanto manipula-se as expressões resultantes leva a

$$
\begin{aligned}
I^{(1)}_{\sigma^2\mu}(\vec{\theta}) &= E\left[\left(\frac{\partial}{\partial\sigma^2}\log\frac{1}{\sqrt{2\pi\sigma^2}}+\frac{\partial}{\partial\sigma^2}\left\{-\frac{1}{2\sigma^2}\left[x_i-\mu-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]^2\right\}\right)\right. \\
&\qquad \left.\left(\underbrace{\frac{\partial}{\partial\mu}\log\frac{1}{\sqrt{2\pi\sigma^2}}}_{0}+\frac{\partial}{\partial\mu}\left\{-\frac{1}{2\sigma^2}\left[x_i-\mu-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]^2\right\}\right)\right] \\
&= E\left[\left\{-\frac{1}{2}\frac{1}{\sigma^2}+\frac{1}{2}\frac{1}{\sigma^4}\left[x_i-\mu-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]^2\right\}\right. \\
&\qquad \left.\left\{-\frac{1}{2\sigma^2}2\frac{\partial}{\partial\mu}\left[x_i-\mu-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]\left[x_i-\mu-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]\right\}\right] \\
&= E\left[\left\{-\frac{1}{2}\frac{1}{\sigma^2}+\frac{1}{2}\frac{1}{\sigma^4}\left[x_i-\mu-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]^2\right\}\right. \\
&\qquad \left.\left\{-\frac{1}{\sigma^2}\left[-1+\beta\underbrace{\sum_{j\in\eta_i}1}_{\eta_i=\Delta}\right]\left[(x_i-\mu)-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]\right\}\right] \\
&= E\left[\left\{\underbrace{-\frac{1}{2}\frac{1}{\sigma^2}}_{\text{a}}+\underbrace{\frac{1}{2}\frac{1}{\sigma^4}\left[(x_i-\mu)-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]^2}_{\text{b}}\right\}\right. \\
&\qquad \left.\underbrace{\left\{\frac{1}{\sigma^2}(1-\beta\Delta)\left[(x_i-\mu)-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]\right\}}_{\text{a ser multiplicado por a e b}}\right] \\
&= E\left\{+\frac{1}{2\sigma^6}(1-\beta\Delta)+\left[(x_i-\mu)-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]^3\right. \\
&\qquad \left.-\frac{1}{2\sigma^4}(1-\beta\Delta)\left[(x_i-\mu)-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]\right\} \\
&= \frac{(1-\beta\Delta)}{2\sigma^6}\underbrace{E\left\{\left[(x_i-\mu)-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]^3\right\}}_{e_1} \\
&\qquad -\frac{(1-\beta\Delta)}{2\sigma^4}\underbrace{E\left\{(x_i-\mu)-\beta\sum_{j\in\eta_i}(x_j-\mu)\right\}}_{e_2}
\end{aligned}
\tag{B.10}
$$

Como pode-se observar na equação (B.10), e_1 e e_2 são os mesmos que os mencionados na seção B.2. Consequentemente, são iguais a zero, o que resulta em $I^{(1)}_{\sigma^2\mu}(\vec{\theta}) = 0$.

B.5 Derivação de $I^{(1)}_{\sigma^2\sigma^2}(\vec{\theta})$

A expressão de $I^{(1)}_{\sigma^2\sigma^2}(\vec{\theta})$ é dada por

$$I^{(1)}_{\sigma^2\sigma^2}(\vec{\theta}) = E\left[\left(\frac{\partial}{\partial\sigma^2}\log p(x_i|\eta_i,\vec{\theta})\right)\left(\frac{\partial}{\partial\sigma^2}\log p(x_i|\eta_i,\vec{\theta})\right)\right] \tag{B.11}$$

Inserindo a equação (3.13) dentro da equação (B.11) enquanto manipula-se as expressões resultantes leva ao conjunto de passos a seguir

$$\begin{aligned}
I^{(1)}_{\sigma^2\sigma^2}(\vec{\theta}) &= E\left[\frac{\partial}{\partial\sigma^2}\log\frac{1}{\sqrt{2\pi\sigma^2}} + \frac{\partial}{\partial\sigma^2}\left\{-\frac{1}{2\sigma^2}\left[x_i-\mu-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]^2\right\}\right]^2 \\
&= E\left[\frac{\partial}{\partial\sigma^2}\left(\log\frac{1}{\sqrt{2\sigma^2}}+\log\frac{1}{\sqrt{\pi}}\right)\right. \\
&\qquad \left. + \frac{\partial}{\partial\sigma^2}\left\{-\frac{1}{2\sigma^2}\left[x_i-\mu-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]^2\right\}\right]^2 \\
&= E\left\{-\frac{1}{2\sigma^2}+\frac{1}{2\sigma^4}\left[x_i-\mu-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]^2\right\}^2 \\
&= \frac{1}{4\sigma^4} - \frac{1}{2\sigma^6}\underbrace{E\left\{\left[(x_i-\mu)-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]^2\right\}}_{e_1} \\
&\qquad + \frac{1}{4\sigma^8}\underbrace{E\left\{\left[(x_i-\mu)-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]^4\right\}}_{e_2}
\end{aligned} \tag{B.12}$$

Vamos expandir as expressões de e_1 e e_2, respectivamente da seguinte maneira

$$\begin{aligned}
e_1 &= E\left\{\left[(x_i-\mu)-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]^2\right\}\\
&= E\left\{\left[(x_i-\mu)^2-2\beta\sum_{j\in\eta_i}(x_i-\mu)(x_j-\mu)\right.\right.\\
&\qquad\left.\left.+\beta^2\sum_{j\in\eta_i}\sum_{k\in\eta_i}(x_j-\mu)(x_k-\mu)\right]\right\}\\
&= E[(x_i-\mu)^2]-2\beta\sum_{j\in\eta_i}E[(x_i-\mu)(x_j-\mu)]\\
&\qquad+\beta^2\sum_{j\in\eta_i}\sum_{k\in\eta_i}E[(x_j-\mu)(x_k-\mu)]\\
&= \sigma^2-2\beta\sum_{j\in\eta_i}\sigma_{ij}+\beta^2\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sigma_{jk}
\end{aligned}$$

e

$$\begin{aligned}
e_2 &= E\left\{\left[(x_i-\mu)-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]^4\right\}\\
&= E\left\{\left[(x_i-\mu)^4-4\beta\sum_{j\in\eta_i}(x_i-\mu)^3(x_j-\mu)\right.\right.\\
&+6\beta^2\sum_{j\in\eta_i}\sum_{k\in\eta_i}(x_i-\mu)^2(x_j-\mu)(x_k-\mu)\\
&-4\beta^3\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sum_{l\in\eta_i}(x_i-\mu)(x_j-\mu)(x_k-\mu)(x_l-\mu)\\
&\left.\left.+\beta^4\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sum_{l\in\eta_i}\sum_{m\in\eta_i}(x_j-\mu)(x_k-\mu)(x_l-\mu)(x_m-\mu)\right]\right\}\\
&= E[(x_i-\mu)^4]-4\beta\sum_{j\in\eta_i}E[(x_i-\mu)^3(x_j-\mu)]\\
&+6\beta^2\sum_{j\in\eta_i}\sum_{k\in\eta_i}E[(x_i-\mu)^2(x_j-\mu)(x_k-\mu)]\\
&-4\beta^3\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sum_{l\in\eta_i}E[(x_i-\mu)(x_j-\mu)(x_k-\mu)(x_l-\mu)]\\
&+\beta^4\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sum_{l\in\eta_i}\sum_{m\in\eta_i}E[(x_j-\mu)(x_k-\mu)(x_l-\mu)(x_m-\mu)]
\end{aligned}$$

Usando o teorema de Isserlis para a distribução de variáveis aleatórias Gaussianas, é possível expressar os momentos de ordem superior como funções de momentos de segunda ordem. Por-

tanto, e_2 torna-se

$$
\begin{aligned}
e_2 = {} & 3\sigma^4 - 12\sigma^2\beta \sum_{j\in\eta_i} \sigma_{ij} + 6\beta^2 \sum_{j\in\eta_i}\sum_{k\in\eta_i} \sigma^2\sigma_{jk} + 12\beta^2 \sum_{j\in\eta_i}\sum_{k\in\eta_i} \sigma_{ij}\sigma_{ik} \\
& - 4\beta^3 \sum_{j\in\eta_i}\sum_{k\in\eta_i}\sum_{l\in\eta_i} (\sigma_{ij}\sigma_{kl} + \sigma_{ik}\sigma_{jl} + \sigma_{il}\sigma_{jk}) \\
& + \beta^4 \sum_{j\in\eta_i}\sum_{k\in\eta_i}\sum_{l\in\eta_i}\sum_{m\in\eta_i} (\sigma_{jk}\sigma_{lm} + \sigma_{jl}\sigma_{km} + \sigma_{jm}\sigma_{kl})
\end{aligned}
$$

Depois de inserir as expressões de e_1 and e_2 de volta na equaçao (B.12), temos

$$
\begin{aligned}
I^{(1)}_{\sigma^2\sigma^2}(\vec{\theta}) &= \frac{1}{4\sigma^4} - \frac{1}{2\sigma^6}\left[\sigma^2 - 2\beta\sum_{j\in\eta_i}\sigma_{ij} + \beta^2\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sigma_{jk}\right] \\
&+ \frac{1}{4\sigma^8}\left[3\sigma^4 - 12\sigma^2\beta\sum_{j\in\eta_i}\sigma_{ij} + 6\beta^2\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sigma^2\sigma_{jk}\right. \\
&+ 12\beta^2\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sigma_{ij}\sigma_{ik} - 4\beta^3\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sum_{l\in\eta_i}(\sigma_{ij}\sigma_{kl} + \sigma_{ik}\sigma_{jl} + \sigma_{il}\sigma_{jk}) \\
&\left.+ \beta^4\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sum_{l\in\eta_i}\sum_{m\in\eta_i}(\sigma_{jk}\sigma_{lm} + \sigma_{jl}\sigma_{km} + \sigma_{jm}\sigma_{kl})\right] \\
&= \frac{1}{4\sigma^4} + \frac{3}{4\sigma^8}\sigma^4 - \frac{1}{2\sigma^6}\sigma^2 - \frac{1}{2\sigma^6}\left[-2\beta\sum_{j\in\eta_i}\sigma_{ij}\right] \\
&- \frac{1}{2\sigma^6}\beta^2\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sigma_{jk} + \frac{1}{4\sigma^8}6\beta^2\sigma^2\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sigma_{jk} \\
&+ \frac{1}{4\sigma^8}4\left[-3\sigma^2\beta\sum_{j\in\eta_i}\sigma_{ij} + 3\beta^2\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sigma_{ij}\sigma_{ik}\right. \\
&- \beta^3\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sum_{l\in\eta_i}(\sigma_{ij}\sigma_{kl} + \sigma_{ik}\sigma_{jl} + \sigma_{il}\sigma_{jk}) \\
&\left.+ \frac{1}{4}\beta^4\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sum_{l\in\eta_i}\sum_{m\in\eta_i}(\sigma_{jk}\sigma_{lm} + \sigma_{jl}\sigma_{km} + \sigma_{jm}\sigma_{kl})\right] \\
&= \frac{1}{2\sigma^4} + \frac{1}{\sigma^6}\beta\sum_{j\in\eta_i}\sigma_{ij} - 3\frac{1}{\sigma^8}\sigma^2\beta\sum_{j\in\eta_i}\sigma_{ij} \\
&+ \frac{3}{2\sigma^6}\beta^2\sigma^2\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sigma_{jk} - \frac{1}{2\sigma^6}\beta^2\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sigma_{jk} \\
&+ \frac{1}{\sigma^8}\left[3\beta^2\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sigma_{ij}\sigma_{ik}\right. \\
&- \beta^3\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sum_{l\in\eta_i}(\sigma_{ij}\sigma_{kl} + \sigma_{ik}\sigma_{jl} + \sigma_{il}\sigma_{jk}) \\
&\left.+ \frac{1}{4}\beta^4\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sum_{l\in\eta_i}\sum_{m\in\eta_i}(\sigma_{jk}\sigma_{lm} + \sigma_{jl}\sigma_{km} + \sigma_{jm}\sigma_{kl})\right] \\
&= \frac{1}{2\sigma^4} - \frac{2}{\sigma^6}\beta\sum_{j\in\eta_i}\sigma_{ij} + \frac{1}{\sigma^6}\beta^2\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sigma_{jk} \\
&+ \frac{1}{\sigma^8}\left[3\beta^2\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sigma_{ij}\sigma_{ik}\right. \\
&- \beta^3\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sum_{l\in\eta_i}(\sigma_{ij}\sigma_{kl} + \sigma_{ik}\sigma_{jl} + \sigma_{il}\sigma_{jk}) \\
&\left.+ \frac{1}{4}\beta^4\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sum_{l\in\eta_i}\sum_{m\in\eta_i}(\sigma_{jk}\sigma_{lm} + \sigma_{jl}\sigma_{km} + \sigma_{jm}\sigma_{kl})\right]
\end{aligned}
\tag{B.13}
$$

Depois de expandir um pouco mais a última equação, $I^{(1)}_{\sigma^2\sigma^2}(\vec{\theta})$ torna-se como segue

$$\begin{aligned} I^{(1)}_{\sigma^2\sigma^2}(\vec{\theta}) = \frac{1}{2\sigma^4} - \frac{1}{\sigma^6}\left[2\beta \sum_{j\in\eta_i} \sigma_{ij} - \beta^2 \sum_{j\in\eta_i}\sum_{k\in\eta_i} \sigma_{jk}\right] \\ + \frac{1}{\sigma^8}\left[3\beta^2 \sum_{j\in\eta_i}\sum_{k\in\eta_i} \sigma_{ij}\sigma_{ik} \right. \\ - \beta^3 \sum_{j\in\eta_i}\sum_{k\in\eta_i}\sum_{l\in\eta_i} (\sigma_{ij}\sigma_{kl} + \sigma_{ik}\sigma_{jl} + \sigma_{il}\sigma_{jk}) \\ \left. + \frac{1}{4}\beta^4 \sum_{j\in\eta_i}\sum_{k\in\eta_i}\sum_{l\in\eta_i}\sum_{m\in\eta_i} (\sigma_{jk}\sigma_{lm} + \sigma_{jl}\sigma_{km} + \sigma_{jm}\sigma_{kl})\right] \end{aligned} \tag{B.14}$$

B.6 Derivação de $I^{(1)}_{\sigma^2\beta}(\vec{\theta})$

A expressão de $I^{(1)}_{\sigma^2\beta}(\vec{\theta})$ é dada por

$$I^{(1)}_{\sigma^2\beta}(\vec{\theta}) = E\left[\left(\frac{\partial}{\partial\sigma^2}\log p(x_i|\eta_i,\vec{\theta})\right)\left(\frac{\partial}{\partial\beta}\log p(x_i|\eta_i,\vec{\theta})\right)\right] \tag{B.15}$$

Inserindo a equação (3.13) dentro da equação (B.15)enquanto manipula-se as expressões resultantes leva a

$$
\begin{aligned}
I^{(1)}_{\sigma^2\beta}(\vec{\theta}) &= E\Bigg[\left(\frac{\partial}{\partial\sigma^2}\log\frac{1}{\sqrt{2\pi\sigma^2}}+\frac{\partial}{\partial\sigma^2}\left\{-\frac{1}{2\sigma^2}\left[x_i-\mu-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]^2\right\}\right)\\
&\qquad\underbrace{\left(\frac{\partial}{\partial\beta}\log\frac{1}{\sqrt{2\pi\sigma^2}}\right)}_{0}+\frac{\partial}{\partial\beta}\left\{-\frac{1}{2\sigma^2}\left[x_i-\mu-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]^2\right\}\Bigg)\Bigg]\\
&= E\Bigg[\left\{-\frac{1}{2}\frac{1}{\sigma^2}+\frac{1}{2}\frac{1}{\sigma^4}\left[x_i-\mu-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]^2\right\}\\
&\qquad\left\{-\frac{1}{2\sigma^2}2\frac{\partial}{\partial\beta}\left[x_i-\mu-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]\left[x_i-\mu-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]\right\}\Bigg]\\
&= E\Bigg[\left\{-\frac{1}{2}\frac{1}{\sigma^2}+\frac{1}{2}\frac{1}{\sigma^4}\left[x_i-\mu-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]^2\right\}\\
&\qquad\left\{-\frac{1}{\sigma^2}\left[-\sum_{j\in\eta_i}(x_j-\mu)\right]\left[(x_i-\mu)-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]\right\}\Bigg]\\
&= E\Bigg[\left\{-\frac{1}{2}\frac{1}{\sigma^2}+\frac{1}{2}\frac{1}{\sigma^4}\left[x_i-\mu-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]^2\right\}\\
&\qquad\left\{\frac{1}{\sigma^2}\left[\sum_{j\in\eta_i}(x_j-\mu)\right]\left[(x_i-\mu)-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]\right\}\Bigg]\\
&= -\frac{1}{2\sigma^4}E\left\{\left[\sum_{j\in\eta_i}(x_j-\mu)\right]\left[(x_i-\mu)-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]\right\}\\
&\qquad+\frac{1}{2\sigma^6}E\left\{\left[\sum_{j\in\eta_i}(x_j-\mu)\right]\left[(x_i-\mu)-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]^3\right\}\\
&= -\frac{1}{2\sigma^4}\left\{\sum_{j\in\eta_i}E\left[(x_i-\mu)(x_j-\mu)\right]-\beta\sum_{j\in\eta_i}\sum_{k\in\eta_i}E\left[(x_j-\mu)(x_k-\mu)\right]\right\}\\
&\qquad+\frac{1}{2\sigma^6}E\Bigg\{\left[\sum_{j\in\eta_i}(x_j-\mu)\right]\\
&\qquad\Big[(x_i-\mu)^3-3\beta\sum_{j\in\eta_i}(x_i-\mu)(x_i-\mu)(x_j-\mu)\\
&\qquad+3\beta^2\sum_{j\in\eta_i}\sum_{k\in\eta_i}(x_i-\mu)(x_j-\mu)(x_k-\mu)\\
&\qquad-\beta^3\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sum_{l\in\eta_i}(x_j-\mu)(x_k-\mu)(x_l-\mu)\Big]\Bigg\}
\end{aligned}
\tag{B.16}
$$

Depois de expandir um pouco mais a última equação, $I^{(1)}_{\sigma^2\beta}(\vec{\theta})$ torna-se como segue

$$
\begin{aligned}
I^{(1)}_{\sigma^2\beta}(\vec{\theta}) = -\frac{1}{2\sigma^4}\Bigg\{ & \sum_{j\in\eta_i} E\Big[(x_i-\mu)(x_j-\mu)\Big] - \beta \sum_{j\in\eta_i}\sum_{k\in\eta_i} E\Big[(x_j-\mu)(x_k-\mu)\Big]\Bigg\} \\
& + \frac{1}{2\sigma^6}\Bigg\{ \sum_{j\in\eta_i} E\Big[(x_i-\mu)^3(x_j-\mu)\Big] \\
& - 3\beta \sum_{j\in\eta_i}\sum_{k\in\eta_i} E\Big[(x_i-\mu)^2(x_j-\mu)(x_k-\mu)\Big] \\
& + 3\beta^2 \sum_{j\in\eta_i}\sum_{k\in\eta_i}\sum_{l\in\eta_i} E\Big[(x_i-\mu)(x_j-\mu)(x_k-\mu)(x_l-\mu)\Big] \\
& - \beta^3 \sum_{j\in\eta_i}\sum_{k\in\eta_i}\sum_{l\in\eta_i}\sum_{m\in\eta_i} E\Big[(x_j-\mu)(x_k-\mu)(x_l-\mu)(x_m-\mu)\Big]\Bigg\}
\end{aligned}
\tag{B.17}
$$

Assim, aplicando o teorema de Isserlis na última equação para calcular os momentos de ordem superior como funções de momentos de segunda ordem e, ao agrupar os termos em comun, tem-se

$$
\begin{aligned}
I^{(1)}_{\sigma^2\beta}(\vec{\theta}) = &-\frac{1}{2\sigma^4}\left\{\sum_{j\in\eta_i}\sigma_{ij} - \beta\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sigma_{jk}\right\} \\
&+\frac{1}{2\sigma^6}\left\{3\sigma^2\sum_{j\in\eta_i}\sigma_{ij} - 6\beta\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sigma_{ij}\sigma_{ik}\right. \\
&+3\beta^2\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sum_{l\in\eta_i}(\sigma_{ij}\sigma_{kl}+\sigma_{ik}\sigma_{jl}+\sigma_{il}\sigma_{jk}) \\
&\left.-\beta^3\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sum_{l\in\eta_i}\sum_{m\in\eta_i}(\sigma_{jk}\sigma_{lm}+\sigma_{jl}\sigma_{km}+\sigma_{jm}\sigma_{kl})\right\} \\
= &\left(\frac{3}{2\sigma^4}-\frac{1}{2\sigma^4}-\frac{1}{2\sigma^4}\right)\left\{\sum_{j\in\eta_i}\sigma_{ij} - \beta\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sigma_{jk}\right\} \\
&+\frac{1}{2\sigma^6}\left\{-6\beta\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sigma_{ij}\sigma_{ik}\right. \\
&+3\beta^2\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sum_{l\in\eta_i}(\sigma_{ij}\sigma_{kl}+\sigma_{ik}\sigma_{jl}+\sigma_{il}\sigma_{jk}) \\
&\left.-\beta^3\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sum_{l\in\eta_i}\sum_{m\in\eta_i}(\sigma_{jk}\sigma_{lm}+\sigma_{jl}\sigma_{km}+\sigma_{jm}\sigma_{kl})\right\} \\
= &\frac{1}{\sigma^4}\left[\sum_{j\in\eta_i}\sigma_{ij} - \beta\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sigma_{jk}\right] \\
&-\frac{1}{2\sigma^6}\left[6\beta\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sigma_{ij}\sigma_{ik}\right. \\
&-3\beta^2\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sum_{l\in\eta_i}(\sigma_{ij}\sigma_{kl}+\sigma_{ik}\sigma_{jl}+\sigma_{il}\sigma_{jk}) \\
&\left.+\beta^3\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sum_{l\in\eta_i}\sum_{m\in\eta_i}(\sigma_{jk}\sigma_{lm}+\sigma_{jl}\sigma_{km}+\sigma_{jm}\sigma_{kl})\right]
\end{aligned}
\tag{B.18}
$$

B.7 Derivação de $I^{(1)}_{\beta\mu}(\vec{\theta})$

A expressão de $I^{(1)}_{\beta\mu}(\vec{\theta})$ é definida como segue

$$
I^{(1)}_{\beta\mu}(\vec{\theta}) = E\left[\left(\frac{\partial}{\partial\beta}\log p(x_i|\eta_i,\vec{\theta})\right)\left(\frac{\partial}{\partial\mu}\log p(x_i|\eta_i,\vec{\theta})\right)\right] \tag{B.19}
$$

como $I^{(1)}_{\mu\beta}(\vec{\theta}) = 0$ (como indicado na seção B.3) e ao alterar a ordem do produto, o que não afeta o valor esperado, concluímos que $I^{(1)}_{\beta\mu}(\vec{\theta}) = 0$.

B.8 Derivação de $I^{(1)}_{\beta\sigma^2}(\vec{\theta}$

A expressão de $I^{(1)}_{\beta\sigma^2}(\vec{\theta}$ é dada por

$$I^{(1)}_{\beta\sigma^2}(\vec{\theta}) = E\left[\left(\frac{\partial}{\partial\beta}\log p(x_i|\eta_i,\vec{\theta})\right)\left(\frac{\partial}{\partial\sigma^2}\log p(x_i|\eta_i,\vec{\theta})\right)\right] \tag{B.20}$$

Observa-se que $I^{(1)}_{\sigma^2\beta}(\vec{\theta}) = I^{(1)}_{\beta\sigma^2}(\vec{\theta})$ uma vez que a ordem dos produtos no valor esperado é irrelevante para o resultado final. Consequentemente, a expressão final de $I^{(1)}_{\beta\sigma^2}(\vec{\theta})$ é dada por

$$\begin{aligned} I^{(1)}_{\beta\sigma^2}(\vec{\theta}) = \frac{1}{\sigma^4}&\left[\sum_{j\in\eta_i}\sigma_{ij} - \beta\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sigma_{jk}\right] \\ &-\frac{1}{2\sigma^6}\left[6\beta\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sigma_{ij}\sigma_{ik}\right. \\ &-3\beta^2\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sum_{l\in\eta_i}(\sigma_{ij}\sigma_{kl}+\sigma_{ik}\sigma_{jl}+\sigma_{il}\sigma_{jk}) \\ &\left.+\beta^3\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sum_{l\in\eta_i}\sum_{m\in\eta_i}(\sigma_{jk}\sigma_{lm}+\sigma_{jl}\sigma_{km}+\sigma_{jm}\sigma_{kl})\right] \end{aligned} \tag{B.21}$$

conforme apresentado na seção B.6.

B.9 Derivação de $I^{(1)}_{\beta\beta}(\vec{\theta})$

A expressão de $I^{(1)}_{\beta\beta}(\vec{\theta})$ é dada por

$$I^{(1)}_{\beta\beta}(\vec{\theta}) = E\left[\left(\frac{\partial}{\partial\beta}\log p(x_i|\eta_i,\vec{\theta})\right)\left(\frac{\partial}{\partial\beta}\log p(x_i|\eta_i,\vec{\theta})\right)\right] \tag{B.22}$$

Inserindo a equação (3.13) dentro da equação (B.22) enquanto manipula-se as expressões resultantes leva a

$$
\begin{aligned}
I^{(1)}_{\beta\beta}(\vec{\theta}) &= E\left[\underbrace{\frac{\partial}{\partial\beta}\log\frac{1}{\sqrt{2\pi\sigma^2}}}_{0}+\frac{\partial}{\partial\beta}\left\{-\frac{1}{2\sigma^2}\left[x_i-\mu-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]^2\right\}\right]^2 \\
&= E\left\{-\frac{1}{2\sigma^2}2\frac{\partial}{\partial\beta}\left[x_i-\mu-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]\left[x_i-\mu-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]\right\}^2 \\
&= E\left\{-\frac{1}{\sigma^2}\left[-\sum_{j\in\eta_i}(x_j-\mu)\right]\left[x_i-\mu-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]\right\}^2 \\
&= \frac{1}{\sigma^4}E\left\{\left[\sum_{j\in\eta_i}\sum_{k\in\eta_i}(x_j-\mu)(x_k-\mu)\right]\right. \\
&\qquad \left.\left[(x_i-\mu)^2-2\beta\sum_{j\in\eta_i}(x_i-\mu)(x_j-\mu)+\beta^2\sum_{j\in\eta_i}\sum_{k\in\eta_i}(x_j-\mu)(x_k-\mu)\right]\right\} \\
&= \frac{1}{\sigma^4}\left\{\sum_{j\in\eta_i}\sum_{k\in\eta_i}E\left[(x_i-\mu)^2(x_j-\mu)(x_k-\mu)\right]\right. \\
&\qquad -2\beta\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sum_{l\in\eta_i}E\left[(x_i-\mu)(x_j-\mu)(x_k-\mu)(x_l-\mu)\right] \\
&\qquad \left.+\beta^2\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sum_{l\in\eta_i}\sum_{m\in\eta_i}E\left[(x_j-\mu)(x_k-\mu)(x_l-\mu)(x_m-\mu)\right]\right\}
\end{aligned}
\tag{B.23}
$$

Assim, aplicando o teorema de Isserlis na última equação para calcular os momentos de ordem superior como funções de momentos de segunda ordem e, ao agrupar os termos em comun, resulta no conjunto de passos a seguir

$$
\begin{aligned}
I^{(1)}_{\beta\beta}(\vec{\theta}) &= \frac{1}{\sigma^4}\left\{\sigma^2\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sigma_{jk}+2\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sigma_{ij}\sigma_{ik}\right. \\
&\qquad -2\beta\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sum_{l\in\eta_i}(\sigma_{ij}\sigma_{kl}+\sigma_{ik}\sigma_{jl}+\sigma_{il}\sigma_{jk}) \\
&\qquad \left.+\beta^2\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sum_{l\in\eta_i}\sum_{m\in\eta_i}(\sigma_{jk}\sigma_{lm}+\sigma_{jl}\sigma_{km}+\sigma_{jm}\sigma_{kl})\right\} \\
&= \frac{1}{\sigma^2}\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sigma_{jk}+\frac{1}{\sigma^4}\left[2\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sigma_{ij}\sigma_{ik}\right. \\
&\qquad -2\beta\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sum_{l\in\eta_i}(\sigma_{ij}\sigma_{kl}+\sigma_{ik}\sigma_{jl}+\sigma_{il}\sigma_{jk}) \\
&\qquad \left.+\beta^2\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sum_{l\in\eta_i}\sum_{m\in\eta_i}(\sigma_{jk}\sigma_{lm}+\sigma_{jl}\sigma_{km}+\sigma_{jm}\sigma_{kl})\right]
\end{aligned}
\tag{B.24}
$$

B.10 Derivação de $I^{(2)}_{\mu\mu}(\vec{\theta})$

A expressão de $I^{(2)}_{\mu\mu}(\vec{\theta})$ é definida como

$$I^{(2)}_{\mu\mu}(\vec{\theta}) = -E\left[\frac{\partial^2}{\partial\mu^2}\log p(x_i|\eta_i,\vec{\theta})\right] \tag{B.25}$$

Nas etapas a seguir, aplicamos duas derivadas sucessivas do log da função verossimilhança e o respectivo resultado pode ser deduzido diretamente da terceira linha da equação (B.3). Em seguida, com esse resultado na mão, podemos prosseguir com a computação da última derivada, que é dada por

$$\begin{aligned} I^{(2)}_{\mu\mu}(\vec{\theta}) &= -\frac{(1-\beta\Delta)}{\sigma^2}E\left\{\frac{\partial}{\partial\mu}\left[(x_i-\mu)-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]\right\} \\ &= -\frac{(1-\beta\Delta)}{\sigma^2}E\left[-1+\beta\underbrace{\sum_{j\in\eta_i}1}_{\eta_i=\Delta}\right] \\ &= \frac{1}{\sigma^2}(1-\beta\Delta)^2 \end{aligned} \tag{B.26}$$

B.11 Derivação de $I^{(2)}_{\mu\sigma^2}(\vec{\theta})$

A expressão de $I^{(2)}_{\mu\sigma^2}(\vec{\theta})$ é definida como segue

$$I^{(2)}_{\mu\sigma^2}(\vec{\theta}) = -E\left[\frac{\partial^2}{\partial\mu\partial\sigma^2}\log p(x_i|\eta_i,\vec{\theta})\right] \tag{B.27}$$

Agora vamos aplicar duas derivadas sucessivas do log da função verossimilhança e o respectivo resultado pode ser deduzido diretamente da sexta linha da equação (B.6). Em seguida, com esse resultado na mão, podemos prosseguir com a computação da última derivada, que é dada por

$$
\begin{aligned}
I^{(2)}_{\mu\sigma^2}(\vec{\theta}) &= -E\left[\frac{\partial}{\partial\mu}\left\{-\frac{1}{2}\frac{1}{\sigma^2}+\frac{1}{2}\frac{1}{\sigma^4}\left[(x_i-\mu)-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]^2\right\}\right] \\
&= -E\left\{0+\frac{1}{2}\frac{1}{\sigma^4}2\left[(x_i-\mu)-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]\right. \\
&\qquad \left.\frac{\partial}{\partial\mu}\left[(x_i-\mu)-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]\right\} \\
&= -E\left\{\frac{1}{\sigma^4}\left[(x_i-\mu)-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]\left[-1+\beta\Delta\right]\right\} \\
&= \frac{1}{\sigma^4}(1-\beta\Delta)\ \underbrace{E\left[(x_i-\mu)-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]}_{\text{igual a 0 de acordo com } e_2 \text{ da equação (B.6)}} \\
&= 0
\end{aligned}
\tag{B.28}
$$

B.12 Derivação de $I^{(2)}_{\mu\beta}(\vec{\theta})$

A expressão de $I^{(2)}_{\mu\beta}(\vec{\theta})$ é dada por

$$
I^{(2)}_{\mu\beta}(\vec{\theta}) = -E\left[\frac{\partial^2}{\partial\mu\partial\beta}\log p(x_i|\eta_i,\vec{\theta})\right] \tag{B.29}
$$

Da mesma forma, pode-se deduzir a primeira derivada a partir da oitava linha da equação (B.8). Com esse resultado, podemos prosseguir sucessivamente com a segunda derivada que leva a

$$
\begin{aligned}
I^{(2)}_{\mu\beta}(\vec{\theta}) &= -E\left[\frac{\partial}{\partial\mu}\left\{\frac{1}{\sigma^2}\left[\sum_{j\in\eta_i}(x_i-\mu)(x_j-\mu)-\beta\sum_{j\in\eta_i}\sum_{k\in\eta_i}(x_j-\mu)(x_k-\mu)\right]\right\}\right] \\
&= \frac{1}{\sigma^2}E\left[-\left\{\sum_{j\in\eta_i}\left[(-1)(x_j-\mu)+(x_i-\mu)(-1)\right]\right.\right. \\
&\qquad \left.\left.-\beta\sum_{j\in\eta_i}\sum_{k\in\eta_i}\left[(-1)(x_k-\mu)(x_j-\mu)(-1)\right]\right\}\right] \\
&= \frac{1}{\sigma^2}E\left[(x_i-\mu)\Delta-\Delta\beta\sum_{j\in\eta_i}(x_j-\mu)+\sum_{j\in\eta_i}(x_j-\mu)-\beta\Delta\sum_{j\in\eta_i}(x_j-\mu)\right] \qquad \text{(B.30)} \\
&= \frac{1}{\sigma^2}E\left\{\Delta\left[(x_i-\mu)-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]+(1-\beta\Delta)\left[\sum_{j\in\eta_i}(x_j-\mu)\right]\right\} \\
&= \frac{1}{\sigma^2}\left\{\Delta\underbrace{E\left[(x_i-\mu)-\beta\sum_{j\in\eta_i}(x_j-\mu)\right]}_{0}+(1-\beta\Delta)\underbrace{E\left[\sum_{j\in\eta_i}(x_j-\mu)\right]}_{0}\right\} \\
&= 0+0=0
\end{aligned}
$$

B.13 Derivação de $I^{(2)}_{\sigma^2\mu}(\vec{\theta})$

A expressão de $I^{(2)}_{\sigma^2\mu}(\vec{\theta})$ é dada por

$$
I^{(2)}_{\sigma^2\mu}(\vec{\theta}) = -E\left[\frac{\partial^2}{\partial\sigma^2\partial\mu}\log p(x_i|\eta_i,\vec{\theta})\right] \qquad \text{(B.31)}
$$

Observa-se que $I^{(2)}_{\sigma^2\mu}(\vec{\theta}) = I^{(2)}_{\mu\sigma^2}(\vec{\theta})$ uma vez que a ordem das derivadas parciais no operador do valor esperado é irrelevante para o resultado final. Consequentemente, deduz-se que $I^{(2)}_{\sigma^2\mu}(\vec{\theta}) = 0$.

B.14 Derivação de $I^{(2)}_{\sigma^2\sigma^2}(\vec{\theta})$

A expressão de $I^{(2)}_{\sigma^2\sigma^2}(\vec{\theta})$ é dada por

$$
I^{(2)}_{\sigma^2\sigma^2}(\vec{\theta}) = -E\left[\frac{\partial^2}{\partial(\sigma^2)^2}\log p(x_i|\eta_i,\vec{\theta})\right] \qquad \text{(B.32)}
$$

Analogamente, aplicamos duas as derivadas parciais sucessivamente no log da função verossimilhança

e o respectivo resultado pode ser deduzido diretamente da quarta linha da equação (B.12). Isso resulta no que segue

$$
\begin{aligned}
I^{(2)}_{\sigma^2\sigma^2}(\vec{\theta}) &= -E\left\{\frac{\partial}{\partial\sigma^2}\left[-\frac{1}{2\sigma^2}+\frac{1}{2\sigma^4}\left(x_i-\mu-\beta\sum_{j\in\eta_i}(x_j-\mu)\right)^2\right]\right\} \\
&= -E\left\{\frac{1}{2\sigma^4}-\frac{1}{\sigma^6}\left((x_i-\mu)-\beta\sum_{j\in\eta_i}(x_j-\mu)\right)^2\right\} \\
&= -E\left\{\frac{1}{2\sigma^4}-\frac{1}{\sigma^6}\left[(x_i-\mu)^2-2\beta\sum_{j\in\eta_i}(x_i-\mu)(x_j-\mu)\right.\right. \\
&\qquad \left.\left.+\beta^2\sum_{j\in\eta_i}\sum_{k\in\eta_i}(x_j-\mu)(x_k-\mu)\right]\right\} \\
&= -\frac{1}{2\sigma^4}+\frac{1}{\sigma^6}\left[E[(x_i-\mu)^2]-2\beta\sum_{j\in\eta_i}E[(x_i-\mu)(x_j-\mu)]\right. \\
&\qquad \left.+\beta^2\sum_{j\in\eta_i}\sum_{k\in\eta_i}E[(x_j-\mu)(x_k-\mu)]\right] \\
&= -\frac{1}{2\sigma^4}+\frac{1}{\sigma^6}\left[\sigma^2-2\beta\sum_{j\in\eta_i}\sigma_{ij}+\beta^2\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sigma_{jk}\right] \\
&= -\frac{1}{2\sigma^4}+\frac{1}{\sigma^4}-\frac{1}{\sigma^6}\left[2\beta\sum_{j\in\eta_i}\sigma_{ij}-\beta^2\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sigma_{jk}\right] \\
&= \frac{1}{2\sigma^4}-\frac{1}{\sigma^6}\left[2\beta\sum_{j\in\eta_i}\sigma_{ij}-\beta^2\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sigma_{jk}\right]
\end{aligned}
\tag{B.33}
$$

B.15 Derivação de $I^{(2)}_{\sigma^2\beta}(\vec{\theta})$

A expressão de $I^{(2)}_{\sigma^2\beta}(\vec{\theta})$ é definida como segue

$$
I^{(2)}_{\sigma^2\beta}(\vec{\theta}) = -E\left[\frac{\partial^2}{\partial\sigma^2\partial\beta}\log p(x_i|\eta_i,\vec{\theta})\right] \tag{B.34}
$$

Analogamente, aplicamos duas as derivadas parciais sucessivamente no log da função verossimilhança e o respectivo resultado pode ser deduzido diretamente da oitava linha da equação (B.16). Isso resulta no que segue

$$
\begin{aligned}
I^{(2)}_{\sigma^2\beta}(\vec{\theta}) &= -E\left\{ \frac{\partial}{\partial\sigma^2}\left[\frac{1}{\sigma^2}\left(x_i-\mu-\beta\sum_{j\in\eta_i}(x_j-\mu)\right)\left(\sum_{j\in\eta_i}(x_j-\mu)\right)\right]\right\} \\
&= -E\left\{ -\frac{1}{\sigma^4}\left[\sum_{j\in\eta_i}(x_i-\mu)(x_j-\mu)-\beta\sum_{j\in\eta_i}\sum_{k\in\eta_i}(x_j-\mu)(x_k-\mu)\right]\right\} \\
&= \frac{1}{\sigma^4}\left[\sum_{j\in\eta_i}E[(x_i-\mu)(x_j-\mu)]-\beta\sum_{j\in\eta_i}\sum_{k\in\eta_i}E[(x_j-\mu)(x_k-\mu)]\right] \\
&= \frac{1}{\sigma^4}\left[\sum_{j\in\eta_i}\sigma_{ij}-\beta\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sigma_{jk}\right]
\end{aligned}
\tag{B.35}
$$

B.16 Derivação de $I^{(2)}_{\beta\mu}(\vec{\theta})$

É evidente observar que $I^{(2)}_{\beta\mu}(\vec{\theta})$ é idêntico a sua contraparte simétrica, ou seja, $I^{(2)}_{\beta\mu}(\vec{\theta}) = I^{(2)}_{\mu\beta}(\vec{\theta}) = 0$.

B.17 Derivação de $I^{(2)}_{\beta\sigma^2}(\vec{\theta})$

Analogamente, é evidente observar que $I^{(2)}_{\beta\sigma^2}(\vec{\theta})$ é idêntico a sua contraparte simétrica, ou seja, $I^{(2)}_{\beta\sigma^2}(\vec{\theta}) = I^{(2)}_{\sigma^2\beta}(\vec{\theta}) = \frac{1}{\sigma^4}\left[\sum_{j\in\eta_i}\sigma_{ij}-\beta\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sigma_{jk}\right]$.

B.18 Derivação de $I^{(2)}_{\beta\beta}(\vec{\theta})$

Finalmente, a expressão de $I^{(2)}_{\beta\beta}(\vec{\theta})$ é definida como

$$
I^{(2)}_{\beta\beta}(\vec{\theta}) = -E\left[\frac{\partial^2}{\partial\beta^2}\log p(x_i|\eta_i,\vec{\theta})\right] \tag{B.36}
$$

Analogamente, aplicamos duas as derivadas parciais sucessivamente no log da função verossimilhança e o respectivo resultado pode ser deduzido diretamente da oitava linha da equação (B.16). Isso leva a

$$
\begin{aligned}
I^{(2)}_{\beta\beta}(\vec{\theta}) &= -E\left\{\frac{\partial}{\partial\beta}\left[\frac{1}{\sigma^2}\left(x_i-\mu-\beta\sum_{j\in\eta_i}(x_j-\mu)\right)\left(\sum_{j\in\eta_i}(x_j-\mu)\right)\right]\right\} \\
&= -\frac{1}{\sigma^2}E\left[\left(-\sum_{j\in\eta_i}(x_j-\mu)\right)\left(\sum_{j\in\eta_i}(x_j-\mu)\right)\right] \\
&= \frac{1}{\sigma^2}\sum_{j\in\eta_i}\sum_{k\in\eta_i}E[(x_j-\mu)(x_k-\mu)] \\
&= \frac{1}{\sigma^2}\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sigma_{jk}.
\end{aligned}
\tag{B.37}
$$

Apendice C

DERIVAÇÕES MATEMÁTICAS DA ENTROPIA DE SHANNON

A entropia de Shannon é definida como o valor esperado da auto-informação e é dada por

$$\begin{aligned} H_\beta &= -E\left[\log L(\vec{\theta};X)\right] \\ &= -E\left[\log \prod_{i=1}^{n} p(x_i|\eta_i,\vec{\theta})\right] \end{aligned} \tag{C.1}$$

Inserindo a equação (3.13) dentro da equação (C.1) e aplicando algumas propriedades matemáticas e estatísticas básicas, temos

$$H_\beta = \frac{1}{2}\left[log\left(2\pi\sigma^2\right)+1\right] - \frac{1}{\sigma^2}\left[\beta\sum_{j\in\eta_i}\sigma_{ij} - \frac{\beta^2}{2}\sum_{j\in\eta_i}\sum_{k\in\eta_i}\sigma_{jk}\right]. \tag{C.2}$$

Usando a notação matricial de covariância introduzida no Capítulo 4, temos

$$H_\beta = \frac{1}{2}\left[\log\left(2\pi\sigma^2\right)+1\right] - \left[\frac{\beta}{\sigma^2}\|\overrightarrow{\rho}\|_+ - \frac{\beta^2}{2\sigma^2}\|\Sigma_p^-\|_+\right] \tag{C.3}$$

onde $\|A\|_+$ indica a soma de todas as entradas do vetor / matriz A.

Apendice D

CONTINUAÇÃO DOS RESULTADOS EXPERIMENTAIS

Tabela D.1: KNN: Desempenho de classificação em Outex_00011_r.

k	Acurácia (%)				
	LBP	GLCM	HOG	Haralick	Abordagem Proposta
2	77.50	70.20	73.95	70.00	100.00
4	76.25	67.91	73.54	65.00	100.00
8	73.33	66.87	75.00	60.20	100.00
12	73.12	64.79	72.91	60.20	100.00

Tabela D.2: SVM: Desempenho de classificação em Outex_00011_r.

Kernel	Acurácia (%)				
	LBP	GLCM	HOG	Haralick	Abordagem Proposta
Linear	63.33	79.58	59.79	81.25	96.66
Rbf	63.33	12.70	59.79	11.04	97.50
LinearSVC	39.16	80.83	44.37	50.41	99.16

Tabela D.3: Naive Bayes: Desempenho de classificação em Outex_00011_r.

Acurácia (%)				
LBP	GLCM	HOG	Haralick	Abordagem Proposta
66.66	79.16	71.66	86.04	99.37

Tabela D.4: CNN: Desempenho de classificação em Outex_00011_r.

Acurácia (%)
CNN
69.88

Tabela D.5: KNN: Desempenho da classificação do método proposto em função do *Kappa* em Outex_00011_r.

k	$\hat{k}$ (Coeficiente *Kappa*)
2	0.9453
4	0.9453
8	0.9453
12	0.9453

Tabela D.6: SVM: Desempenho da classificação do método proposto em função do *Kappa* em Outex_00011_r.

Kernel	$\hat{k}$ (Coeficiente *Kappa*)
Linear	0.9562
Rbf	0.9453
LinearSVC	0.9562

Tabela D.7: Naive Bayes: Desempenho da classificação do método proposto em função do *Kappa* em Outex_00011_r.

$\hat{k}$ (Coeficiente *Kappa*)
0.9454

Tabela D.8: KNN: Desempenho da classificação do método proposto em função da precisão, revocação e *F1-score* em Outex_00011_r.

	Taxa de sucesso (%)		
Média k	Precisão	Revocação	*F1-score*
Micro 2	94.79	94.79	94.79
Macro 2	91.15	92.88	90.67
Ponderada 2	96.48	94.79	94.85
Micro 4	95.83	95.83	95.83
Macro 4	89.93	93.40	90.34
Ponderada 4	96.96	95.83	95.83
Micro 8	92.71	92.71	92.71
Macro 8	90.80	91.84	89.26
Ponderada 8	96.40	92.71	92.87
Micro 12	90.63	90.63	90.63
Macro 12	89.06	89.90	87.31
Ponderada 12	95.44	90.63	91.30

Tabela D.9: SVM: Desempenho da classificação do método proposto em função da precisão, revocação e *F1-score* em Outex_00011_r.

Média do Kernel	Taxa de sucesso (%)		
	precisão	Revocação	*F1-score*
Micro Linear	92.71	92.71	92.71
Macro Linear	89.55	91.78	88.81
Ponderada Linear	95.74	92.71	92.95
Micro Rbf	92.71	92.71	92.71
Macro Rbf	89.55	91.77	88.81
Ponderada Rbf	95.74	92.71	92.95
Micro LinearSVC	93.75	93.75	93.75
Macro LinearSVC	90.07	93.15	89.92
Ponderada LinearSVC	96.65	93.75	94.06

Tabela D.10: Naive Bayes: Desempenho da classificação do método proposto em função da precisão, revocação e *F1-score* em Outex_00011_r.

Média	Taxa de sucesso (%)		
	Precisão	Revocação	*F1-score*
Micro	91.67	91.67	91.67
Macro	93.23	94.50	91.60
Ponderada	96.83	91.67	92.75