UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"
Câmpus de Bauru

Nathalia Queiroz Ascenção

RANQUEAMENTO DE INFORMAÇÕES POR FLORESTA DE CAMINHOS ÓTIMOS

BAURU
2020

Nathalia Queiroz Ascenção

Ranqueamento de Informações por Floresta de Caminhos Ótimos

Dissertação apresentada como parte dos requisitos para obtenção do título de Mestre em Ciência da Computação, junto ao Programa de Pós-Graduação em Ciência da Computação, da Faculdade de Ciências da Universidade Estadual Paulista "Júlio de Mesquita Filho", Câmpus de Bauru.

Financiadora: Petrobras - Convênio nº 5850.0102453.16.9 / SAP 4600530925.

Orientador: Prof. Dr. João Paulo Papa

Bauru
2020

A811r

Ascenção, Nathalia Queiroz
Ranqueamento de informações por floresta de caminhos ótimos / Nathalia Queiroz Ascenção. -- Bauru, 2020
74 p. : il., tabs., fotos

Dissertação (mestrado) - Universidade Estadual Paulista (Unesp), Faculdade de Ciências, Bauru
Orientador: Prof. Dr. João Paulo Papa

1. Recuperação de Informações. 2. Ranqueamento de Informações. 3. Floresta de Caminhos Ótimos. I. Título.

Sistema de geração automática de fichas catalográficas da Unesp. Biblioteca da Faculdade de Ciências, Bauru. Dados fornecidos pelo autor(a).

Essa ficha não pode ser modificada.

Nathalia Queiroz Ascenção

Ranqueamento de Informações por Floresta de Caminhos Ótimos

Dissertação apresentada como parte dos requisitos para obtenção do título de Mestre em Ciência da Computação, junto ao Programa de Pós-Graduação em Ciência da Computação, da Faculdade de Ciências da Universidade Estadual Paulista "Júlio de Mesquita Filho", Câmpus de Bauru.

Financiadora: Petrobras - Convênio nº 5850.0102453.16.9 / SAP 4600530925.

Comissão Examinadora

Prof. Dr. João Paulo Papa
UNESP - Câmpus de Bauru
Orientador

Prof. Dr. Kelton Augusto Pontara da Costa
Faculdade de Tecnologia - Câmpus de Bauru

Prof. Dra. Patricia Bellin Ribeiro
Faculdade de Tecnologia - Câmpus de Bauru

.

Bauru, 20 de Fevereiro de 2020.

Agradecimentos

A Deus e Nossa Senhora por todo amor, força e bençãos que Deles recebi durante toda minha vida e em especial durante o desenvolvimento e conclusão deste trabalho.

A minha mãe, pai e irmã por todo amor, compreensão, motivação e apoio que sempre me deram.

Ao meu orientador João Paulo Papa por todo auxílio, preocupação e orientação, esses foram de suma importância para meu desenvolvimento acadêmico e pessoal.

Aos amigos que me acompanharam durante essa etapa pela amizade, alegrias e colaboração para meu crescimento pessoal e profissional.

À Universidade Estadual Paulista "Júlio de Mesquita Filho" (UNESP) por possibilitar a realização desse trabalho e aos colegas de laboratório pelas palavras de incentivo e conhecimentos adquiridos.

À Petrobras referente à bolsa recebida durante pesquisa no Projeto "RADIAR: Método para a Disseminação do Conhecimento"– Convênio nº 5850.0102453.16.9 / SAP 4600530925. E também aos professores e pesquisadores com os quais trabalhei durante o período de pesquisa nesse.

Resumo

A tarefa de aprender a ranquear tem sido amplamente estudada pela comunidade científica de aprendizado de máquina principalmente devido a sua utilização na área de recuperação de informações, mineração de dados e processamento de linguagem natural. O ranqueamento de informações pode ser dividido em criação de ranqueamento e agregação desse. O presente trabalho aborda o ranqueamento de informações sob à ótica da criação desse, na qual tem-se inicialmente uma necessidade, comumente denominada *query*, e deseja-se gerar uma lista ranqueada dos itens oferecidos como resposta para dada *query*, estando os itens relevantes a essa localizados nas primeiras posições da lista. Até o presente momento, classificadores baseados em Floresta de Caminhos Ótimos não foram aplicados à tarefa de aprender a ranquear e este projeto de mestrado tem como principal contribuição a aplicação desses, na versão supervisionada com grafo completo e k-*nn*, ao ranqueamento de informações. Para aplicá-los a esse contexto a informação de custo das amostras do conjunto de treinamento foi utilizada para ranquear as r amostras mais relevantes para dada entrada na fase de teste dos classificadores. Experimentos foram realizados tendo como cenário a recuperação e ranqueamento de imagens utilizando características referentes ao conteúdo visual dessas. Os resultados experimentais obtidos com as abordagens baseadas em OPF foram comparados aos da técnica *Ranking SVM* e ao obtido ao ranquear as imagens utilizando a distância entre os vetores de características que representam a imagem de *query* e cada uma das candidatas a serem relevantes para tal. O classificador OPF com grafo completo forneceu resultados similares aos obtidos com a técnica *Ranking SVM* e ao lado do OPF k-*nn* apresentou os menores tempos de execução durante a criação do ranqueamento. Assim, ambas as abordagens propostas demonstraram ser uma solução promissora para problemas em que o ranqueamento de informações se faz necessário.

Palavras-chave: Recuperação de Informações. Ranqueamento de Informações. Floresta de Caminhos Ótimos.

Abstract

The learning to rank task has been widely studied by the machine learning scientific community mainly due to its use in information retrieval, data mining and natural language processing. Information ranking can be divided into ranking creation and aggregation. This work addresses the information ranking from the perspective of its creation, where there is initially a need, commonly called query, and we aim to generate a ranked list of items offered in response to given query, with the relevant items to query located in the first positions from the list. To date, classifiers based on Optimum-Path Forest have not been applied to the learning to rank task and this master's project has as its main contribution to apply these, in the supervised version with full graph and k-*nn*, to the information ranking. To apply them to this context, the cost information from training set samples was used to rank the r most relevant samples for input sample from the testing phase of the classifiers. Experiments were performed considering the image retrieval and ranking scenario using visual content features. The experimental results obtained with the OPF-based approaches were compared to those of the Ranking SVM technique and to those obtained by ranking the images using the distance between the feature vector representing the query image and those representing the candidates, that is, that may be relevant to it. The complete graph OPF classifier provided similar results to those obtained with the Ranking SVM technique and beside the k-*nn* OPF presented the shortest execution time during the ranking creation. Thus, both proposed approaches have proved to be a promising solution to problems where information ranking is required.

Keywords: Information Retrieval. Information Ranking. Optimum-Path Forest.

Lista de ilustrações

Lista de tabelas

Sumário

1 Introdução

O campo de conhecimento denominado "aprender a ranquear", do inglês *learning to rank* (LTR), surgiu da intersecção de aprendizado de máquina, recuperação de informações e processamento de linguagem natural (LI, 2014). Técnicas para aprender a ranquear são utilizadas em diferentes aplicações, tais como recuperação e recomendação de informações (KARATZOGLOU; BALTRUNAS; SHI, 2013; JI; WANG, 2013; YU et al., 2015; SEVERYN; MOSCHITTI, 2015; PAISITKRIANGKRAI; SHEN; HENGEL, 2015; YANG; TANG; YAO, 2015; CHEN et al., 2015; CHEN et al., 2016), resumo de documentos (SHEN; LI, 2011) e tradução de máquina (WATANABE, 2012), dentre outras. No entanto, apesar da diversa gama de aplicações que utilizam ranqueamento de informações, esse desempenha um papel fundamental na recuperação de informações, do inglês *information retrieval* (IR), principalmente quando a quantidade de dados a ser analisada é consideravelmente grande ou cresce rapidamente, como acontece com os dados disponíveis na internet atualmente (JIN et al., 2015).

Em Ciência da Computação, a recuperação de informações lida com a representação, armazenamento e acesso à informação em formato digital e tem como objetivo recuperar informações relevantes como resposta a uma necessidade comumente denominada *query*. O processo de recuperação é composto por diferentes fases, partindo da representação de dados e culminando na apresentação de informações relevantes ao usuário final (SHARMA; PATEL, 2013). Nesse contexto, sistemas de recuperação de informações mantêm uma coleção de objetos, que podem ser documentos, imagens, textos, entre outros; e, dada uma necessidade do usuário, esses recuperam os itens de acordo com a consulta realizada e por fim apresentam uma lista ordenada de acordo com a relevância entre o objeto de consulta e os itens recuperados. Alguns sistemas utilizam algoritmos de ranqueamento na etapa de ordenação dos itens recuperados a fim de gerar um modelo que apresente no topo da lista dos itens recuperados aqueles mais relevantes (YIN et al., 2016).

Apesar da recuperação de informações poder ser realizada utilizando diferentes dados multimídia, como as imagens exercem papel importante em diferentes campos de pesquisa, tais como medicina, educação, publicidade e propaganda, entre outros; a área de estudo denominada Recuperação de Imagem Baseada em Conteúdo, do inglês *Content-Based Image Retrieval* (CBIR), foi desenvolvida com o objetivo de recuperar imagens relevantes de uma base de dados por meio de características como cor, forma e textura, principalmente, e tem sido aplicada em diferentes trabalhos (ILIADIS et al., 2013; MILOVANOVIC; MINOVIC; STARCEVIC, 2013; BUGATTI et al., 2014; MEMON et al., 2015; DEMIR; BRUZZONE, 2015; FARIA et al., 2010). (FARIA et al., 2010) aplicaram o ranqueamento de informações à recuperação de imagens por conteúdo a fim de analisar a eficácia da recuperação utilizando não somente diferentes descritores de imagens, mas também métodos de ranqueamento. Essa configuração,

composta pela utilização de descritores e técnicas de ranqueamento aplicadas à recuperação, também é explorada no presente trabalho.

O problema de aprender a ranquear, segundo (LI, 2014), pode ser dividido em dois tipos: criação de ranqueamento (ou simplesmente ranqueamento) e agregação de ranqueamento. Na criação, tem-se inicialmente uma consulta, comumente denominada *query*, e deseja-se gerar uma lista ranqueada dos objetos oferecidos como possíveis resultados para a dada consulta com base na comparação entre as características desses e do item consultado. Por outro lado, na agregação de ranqueamento cria-se uma lista final de *rankings* composta por múltiplas listas parciais de ranqueamento, sendo que essas últimas podem ser geradas utilizando diferentes métodos. O presente trabalho aborda o problema de aprender a ranquear sob o olhar de criação do ranqueamento. Nesse contexto, as técnicas empregadas na criação do *ranking* são comumente divididas em abordagens *pointwise*, *pairwise* e *listwise*. O Capítulo 4 aborda em detalhes cada uma delas a fim de que o leitor tenha uma maior compreensão acerca das diferenças entre essas.

Dentre as principais métricas utilizadas para avaliar o desempenho da função, também chamada modelo, aprendida por dado método de ranqueamento estão o NDCG, do inglês *Normalized Discounted Cumulative Gain* e o MAP, do inglês *Mean Average Precision*, (CHEN et al., 2009; WANG et al., 2013). Essas são usadas para avaliar o quanto os resultados fornecidos à dada consulta satisfazem de fato o objetivo do usuário final. De forma resumida, pode-se dizer que o NDGC busca quantificar a qualidade do ranqueamento ao medir a relevância de um item com base na sua posição na lista de resultados para dada consulta. Enquanto o MAP indica a média da precisão dos resultados obtidos, sendo a precisão definida como a porção dos itens relevantes dentro dos recuperados e fornecidos como resultado. Tanto a formulação matemática do NDCG quanto do MAP são apresentadas no capítulo 4 e ambas as métricas são utilizadas nos experimentos realizados nesse trabalho.

Além das tarefas de recuperação e ranqueamento de informações, a classificação dessas também tem sido amplamente estudada pela comunidade científica e aplicada à diferentes cenários (KAHOU et al., 2015; TRIPATHY; AGRAWAL; RATH, 2015; DHUNGEL; CARNEIRO; BRADLEY, 2016; HALL, 2016; HERSHEY et al., 2017). Dentre a gama de técnicas atualmente disponíveis voltadas à classificação de informações, os classificadores baseados em Floresta de Caminhos Ótimos, do inglês *Optimum-Path Forest* (OPF), propostos por (PAPA., 2008), têm ganhado considerável atenção nos últimos anos, principalmente devido aos resultados promissores obtidos por esses em aprendizado não supervisionado, semi-supervisionado e supervisionado. Classificadores baseados em OPF têm sido aplicados em diversos cenários e apresentado ótimos resultados quando comparados às abordagens consideradas estado da arte no contexto de classificação (AMORIM et al., 2016; LOPES et al., 2016; PAPA; FERNANDES; FALCÃO, 2017; AMORIM; FALCÃO; PAPA, 2018). Os classificadores na versão supervisionada com grafo completo e k-*nn* são apresentados em detalhes no capítulo 3.

Como principal contribuição para o presente projeto de mestrado, os classificadores baseados em Floresta de Caminhos Ótimos em sua versão supervisionada com grafo completo e k-*nn* foram adaptados com o objetivo de serem aplicados à tarefa de ranqueamento de informações realizando a criação do ranqueamento de modo semelhante à proposta em abordagens do tipo *pointwise*. A motivação para tal surgiu devido ao fato de, apesar dos classificadores baseados em OPF já terem sido aplicados ao cenário de recuperação de informações (TAVARES; FALCãO; MAGALHãES, 2011; SILVA et al., 2012; LI et al., 2016; DHAWALE; JOGLEKAR; KULKARNI, 2017), esses, até o presente momento, não terem sido utilizados para ranquear informações. Além disso, cabe destacar a necessidade e atual interesse da comunidade da área de aprendizado de máquina em melhorar a eficácia de abordagens utilizadas neste cenário. As adaptações realizadas nos algoritmos dos classificadores para criação do ranqueamento são descritas no capítulo 5.

Os classificadores baseados em OPF adaptados para ranquear informações foram validados no cenário de recuperação de imagens baseada em conteúdo, isto é, tem-se como *background* um contexto em que deseja-se recuperar as r imagens mais similares a uma dada imagem de consulta, sendo r não nulo e $\in$ Z, e tendo como base as características visuais dessas. No processo de extração das características visuais foram utilizados descritores de imagens baseados nos atributos de cor, textura e forma. O parâmetro r foi variado em 10, 15 e 20, sendo que esses valores foram definidos considerando-se a apresentação ideal dos resultados recuperados e apresentados ao usuário final em mecanismos de buscas, um dos principais exemplos de aplicação que utilizam técnicas de recuperação e ranqueamento de informações, uma vez que espera-se que os resultados relevantes a uma dada *query* sejam apresentados nas primeiras páginas desses.

Experimentos foram realizados utilizando os classificadores OPF adaptados ao problema de ranqueamento e a técnica *Ranking SVM*, amplamente utilizada na criação de ranqueamento. Optou-se por comparar os resultados dos classificadores baseados em OPF com os do *Ranking SVM* porque esse último tem apresentado boa acurácia ao criar *rankings* e também pelo fato desse ser derivado do classificador SVM que, por sua vez, apresenta eficiência similar à obtida por classificadores OPF no cenário de classificação de informações (PAPA., 2008).

Os resultados obtidos com essas abordagens também foram comparados ao alcançado no ranqueamento das imagens à partir da distância entre os vetores de características da imagem de consulta e cada uma das imagens possivelmente relevantes a essa. Cabe mencionar que essa abordagem é frequentemente utilizada em *information retrieval* (PATIL; TALBAR, 2012; KHOSLA; RAJPAL; SINGH, 2015; MISTRY; INGOLE; INGOLE, 2018) quando nenhuma técnica de *learning to rank* é aplicada ao problema.

Nos experimentos foram utilizadas imagens das bases de dados Original Brodatz (BRODATZ, 1966), Caltech101 (FEI-FEI; FERGUS; PERONA, 2004) e MPEG-7 (RALPH, 1999) e os descritores globais ACC (HUANG et al., 1997), BIC (STEHLING; NASCIMENTO; FALCãO,

2002), CCV (PASS; ZABIH; MILLER, 1997), LCH (SWAIN; BALLARD, 1991), LBP (OJALA; PIETIKäINEN; MäENPää, 2002), SASI (ÇARKACIOGLU; VURAL, 2003) e SPYTEC (LU; CHANGS, 2007) para extração das características das imagens.

A fim de que o leitor se familiarize com o cenário dos experimentos, conceitos introdutórios acerca da recuperação de informações, recuperação de imagens e ranqueamento dessas são apresentados no capítulo 2. A técnica de ranqueamento *Ranking SVM* e o ranqueamento de informações como área de estudo são abordados no capítulo 4. Os experimentos, metodologia utilizada nesses e resultados obtidos são descritos no capítulo 6. Os objetivos do presente trabalho e estrutura da dissertação são elencados a seguir.

1.1 Objetivos

1.1.1 Objetivo Geral

Aplicar os classificadores baseados em Floresta de Caminhos Ótimos na versão supervisionada com grafo completo e k-*nn* ao ranqueamento de informações.

1.1.2 Objetivos Específicos

a) analisar as principais abordagens utilizadas no cenário de ranqueamento de informações;

b) adaptar os algoritmos dos classificadores baseados em OPF na versão supervisionada à tarefa de ranqueamento;

c) testar e validar os algoritmos propostos em diferentes bases de dados;

d) testar e validar os algoritmos propostos utilizando características extraídas a partir de diferentes descritores de imagens;

e) validar os resultados obtidos com as abordagens propostas por meio de métricas de avaliação baseadas em métodos estatísticos e matemáticos, a fim de prover maior confiabilidade à pesquisa e

f) comparar o desempenho dos algoritmos propostos com os de métodos comumente utilizados em *learning to rank*.

1.2 Estrutura da Dissertação

O restante dessa dissertação está organizada como segue:

- Capítulo 2: Apresenta conceitos e definições sobre a recuperação de informações, com foco na recuperação de imagens baseada em conteúdo, e breve introdução acerca do ranqueamento dessas;

- Capítulo 3: Apresenta conceitos e definições sobre os classificadores baseados em Floresta de Caminhos Ótimos na versão supervisionada com grafo completo e k-*nn*;
- Capítulo 4: Apresenta conceitos e definições sobre ranqueamento de informações, técnicas e métricas utilizadas nos experimentos realizados;
- Capítulo 5: Apresenta as adaptações realizadas nos classificadores baseados em OPF para aplicação desses ao cenário de ranqueamento de informações;
- Capítulo 6: Apresenta os experimentos realizados e seus resultados;
- Capítulo 7: Apresenta a conclusão obtida com a realização do presente trabalho e faz menção a trabalhos futuros.

2 Conceitos Introdutórios

No presente trabalho o ranqueamento de informações será abordado em conjunto com a recuperação de informações, uma vez que o primeiro tópico é peça importante para que o último possa apresentar resultados eficazes no que diz respeito à recuperação de itens relevantes para uma dada busca. A fim de que o tema deste trabalho e os experimentos realizados nesse sejam melhor embasados, a introdução de conceitos necessários à compreensão desses são apresentados neste capítulo. Primeiramente, um breve resumo do campo de estudos denominado recuperação de informações é apresentado, seguido de uma explicação dos conceitos chave referentes à recuperação de imagens baseada em conteúdo e por fim um exemplo que ilustra uma aplicação que une ranqueamento e recuperação de imagens por conteúdo é destacado.

2.1 Recuperação de Informações

Em Ciência da Computação, a Recuperação de Informações consiste no processo de localizar e obter informações de diferentes tipos em conjuntos de dados geralmente extensos. Esses dados podem ser textos, sons, imagens, vídeos, metadados (como palavras-chave), entre outros. As aplicações de IR são diversas e incluem, mas não são limitadas a essas, extração de informação, filtragem de informação, filtragem de *spam*, extração de formas em imagens, resumo automático e recuperação de informações em mecanismos de buscas da *web* (BIJALWAN et al., 2014). Geralmente, considera-se que sistemas baseados em IR possibilitam a satisfação de uma certa necessidade de informação à partir de uma coleção de informações disponíveis (DUH; KIRCHHOFF, 2008).

Apesar da recuperação de informações e a criação/aperfeiçoamento de técnicas que realizem essa tarefa automaticamente e de forma eficaz serem crescentes objetos de estudo dentro da comunidade científica (WAN et al., 2014; GONZáLEZ-DíAZ et al., 2017; MANSOURIAN et al., 2018), o problema de recuperar informações é bastante antigo. Esse remonta à 3000 A.C, aproximadamente, quando os povos Sumérios começaram a armazenar barras de argila contendo inscrições cuneiformes organizadas de acordo com certas categorias. Há milhares de anos, esses povos já haviam percebido que a organização e acesso adequado à informações eram de grande importância para o uso eficiente dessas (SINGHAL; GOOGLE, 2001).

Com o decorrer dos séculos, a necessidade de armazenar e recuperar informações cresceu significativamente, especialmente com a invenção do papel e de diferentes instrumentos que facilitaram a escrita, tais como canetas e máquinas de escrever, por exemplo. Apesar da busca por informações poder ser realizada manualmente, com o aumento do número de fontes de armazenamento disponíveis, essa tarefa tornou-se muitas vezes não trivial e até mesmo ineficiente em questão de tempo de busca. Os sistemas de recuperação de informações

automáticos foram desenvolvidos originalmente nos anos 1940 a fim de gerenciar o conteúdo literário produzido até então. Esses primeiros sistemas foram utilizados principalmente em bibliotecas e universidades e possibilitaram uma busca mais eficiente das informações contidas em livros. Realizar a procura por informações utilizando sistemas de informação facilitou muito o processo de busca (FRAKES; BAEZA-YATES, 1992) (SANDERSON; CROFT, 2012).

Apesar do advento da internet, do aumento da velocidade dos processadores e da capacidade de armazenamento de dados em computadores terem possibilitado a difusão do uso de sistemas para recuperação de dados, o estudo da recuperação de informações dentro do âmbito da Ciência da Computação teve início ainda no século XX. A área de estudos destinada à essa atividade dedica-se à criação e aperfeiçoamento de métodos que possibilitem a busca e recuperação, em meio a coleções de dados semi ou não estruturados, de itens relevantes como resposta a uma dada consulta, comumente denominada *query*, realizada no início do processo de recuperação (SANDERSON; CROFT, 2012).

Textos escritos em linguagem natural, imagens digitais, áudio, vídeo, notícias em tempo real e conhecimento armazenado em documentos são alguns exemplos de dados não estruturados. Devido ao rápido crescimento de dados desse tipo disponíveis na internet e em diferentes sistemas, a necessidade de utilização de técnicas eficazes para automatização da recuperação de informações tornou-se essencial nos últimos anos. Assim, *Information Retrieval* é uma importante área de pesquisa sob a qual são desenvolvidas metodologias que auxiliam sistemas a recuperar dados multimídia de forma eficaz e, em alguns casos, a também lidar com a sobrecarga de informações; uma vez que possibilitam a filtragem de informações buscando retornar, prioritariamente, aquelas mais relevantes (HUANG et al., 2013; SHEN et al., 2014).

De forma resumida, pode-se dizer que sistemas de recuperação de informações têm como finalidade recuperar dados, dispostos em um determinado conjunto, que satisfaçam uma dada necessidade de informação, ou seja, objetivam recuperar dados relevantes para uma *query* previamente informada. Assim, a tarefa de recuperação de informações é realizada partindo da definição de uma necessidade, seguida da busca por respostas que satisfaçam essa necessidade em um conjunto de dados disponíveis e tem fim com a recuperação e apresentação da informação requerida e/ou das mais similares à essa.

Uma informação em uma aplicação baseada em IR é classificada como relevante para uma dada consulta quando, segundo alguma métrica em específico, essa corresponde, apresenta similaridade ou está relacionada à busca inicialmente definida. No entanto, ao recuperar informações referentes a uma consulta, necessariamente não é retornado um único item da coleção de dados; em vez disso, diversos itens podem estar relacionados à busca, apresentando diferentes graus de relevância entre si, enquanto outros podem não ter relação nenhuma com a consulta e ainda assim serem recuperados; logo, nem sempre a informação recuperada satisfaz a necessidade informada. Essa é uma das principais diferenças entre uma busca em um sistema de IR e uma realizada em um banco de dados por meio de uma cláusula SQL, do inglês *Structured*

Query Language, por exemplo; uma vez que, nesse último caso, são recuperados somente os dados estruturados estritamente relacionados à consulta. (GREENGRASS, 2000; MANNING; SCHUTZE., 2009).

Os modelos utilizados para realização da recuperação de informações por sistemas automatizados geralmente são divididos em modelos de conjuntos teóricos, algébricos, probabilísticos e modelos baseados em características. Esses últimos abrangem o caso de modelos aprendidos com base em algoritmos de *learning to rank*. Os sistemas de recuperação baseados nessa última categoria, normalmente avaliam os itens a ser recuperados de acordo com uma pontuação numérica, comumente denominada *score*, que define a relevância de determinado item para o objeto de consulta. Dessa forma, os itens são ranqueados de acordo com os valores de *score* e são apresentados apenas os r-primeiros itens mais relevantes recuperados (FARIA et al., 2010; RANI, 2011).

Para localizar os itens relevantes a uma dada consulta e recuperá-los em uma ordem adequada, isto é, ordenados por relevância, os sistemas de recuperação de informações modernos buscam, dada uma *query* e um conjunto de itens com diferentes graus de relevância para a dada consulta, encontrar o ranqueamento apropriado para os objetos candidatos a serem possíveis respostas para a necessidade previamente explicitada. O modelo que ranqueia os dados desse conjunto é uma das partes essenciais de sistemas baseados em IR. Técnicas e algoritmos capazes de computar tais modelos de ranqueamento são criados e estudados na área de pesquisa definida como *learning to rank*. Essa é detalhadamente apresentada no capítulo 4.

Apesar da diversa gama de tipos de dados que podem ser utilizados como entrada e apresentados como saída em um sistema de recuperação de informações, neste trabalho as imagens digitais foram definidas como objeto de estudo e utilizadas em experimentos referentes ao ranqueamento de informações. Com base nesse cenário, a seção a seguir apresentará maiores detalhes acerca da recuperação automática desse tipo de dado à partir de características baseadas em seu conteúdo.

2.2 Recuperação de Imagem

Melhorias referentes ao armazenamento de imagens digitais e de tecnologias de aquisição de dados desse tipo juntamente com o aumento do número de usuários da internet e da popularização de aplicações que permitem o *upload* e compartilhamento de fotos têm contribuído para o rápido crescimento do volume disponível desses itens em diferentes sistemas de informação. Sistemas especialistas voltados à medicina, biometria e sensoriamento remoto, são alguns dos diversos exemplos de aplicações que utilizam busca e recuperação de imagens como atividades necessárias ou, em alguns casos, primordiais, para o alcance de seus objetivos como aplicação. Frente a esse cenário, o desenvolvimento e aprimoramento de técnicas que manipulem essas coleções de dados eficientemente se faz cada vez mais necessário (TORRES; FALCãO, 2006;

MEHARBAN; PRIYA, 2016; ZHOU; LI; TIAN, 2017).

Como já mencionado, a Recuperação de Informações é uma importante área de pesquisa sob a qual são desenvolvidas metodologias que auxiliam sistemas a recuperar dados multimídia de forma eficaz e, em alguns casos, a lidar com a sobrecarga de informações disponíveis. Como a busca e recuperação manual de imagens em extensas bases de dados apresenta alguns desafios, tais como o tempo destinado à execução da tarefa, além de ser exaustiva e não trivial, nos últimos anos diferentes técnicas para recuperação automática de imagens têm sido propostas com o objetivo de que essa seja realizada de forma menos custosa e mais eficiente (KAUR; JINDAL; KAUR, 2016; TZELEPI; TEFAS, 2017; HE et al., 2018).

Abordagens para recuperação automática de imagens baseiam-se em informação textual contidas em *tags* associadas às imagens ou informação visual. A última abordagem geralmente utiliza *sketchs* e/ou características visuais baseadas no conteúdo da imagem para realizar a recuperação. Em geral, prefere-se as abordagens baseadas em informações visuais devido à natureza subjetiva da informação textual, uma vez que essa nem sempre representa de forma consistente o conteúdo visual de uma imagem e também porque rotular manualmente esses itens pode apresentar-se como uma tarefa inviável, exigindo grande esforço humano dependendo do tamanho do conjunto de dados a ser rotulado (KHOKHER; TALWAR, 2011; ZHOU; LI; TIAN, 2017).

2.2.1 Recuperação de Imagem Baseada em Conteúdo

O campo de estudo denominado Recuperação de Imagem Baseada em Conteúdo tem como objetivo o desenvolvimento de técnicas que recuperem automaticamente imagens de uma base de dados utilizando características visuais baseadas em seu conteúdo, tais como cor, forma e textura (DATTA; LI; WANG, 2005). O fato do processo de recuperação ser realizado de forma automática é uma das vantagens desse tipo de abordagem em comparação com aquelas que se baseiam em palavras-chave e/ou outros dados textuais para recuperar as imagens; uma vez que descrever o conteúdo de uma imagem utilizando características textuais é intrinsecamente mais difícil, além de, possivelmente, apresentar maiores chances de não representá-lo de forma precisa.

Em aplicações baseadas em CBIR (ILIADIS et al., 2013; ALZU'BI; AMIRA; RAMZAN, 2015; WAN et al., 2015; MASOOD; SHAHID; SHARIF, 2018), as propriedades visuais das imagens são utilizadas para definir a similaridade entre imagens armazenadas em uma coleção de dados e àquela fornecida pelo usuário como entrada para o sistema, comumente chamada *query*. Descritores de imagens - caracterizados por um algoritmo de extração que codifica características visuais em vetores de características - e uma medida de similaridade, calculada geralmente utilizando uma função de distância, tal como a euclidiana, são utilizados no processo de recuperação das imagens com o intuito de recuperar automaticamente imagens similares à *query* fornecida inicialmente à aplicação. (TORRES; FALCãO, 2006).

Resumidamente falando, descritores de imagens são compostos por uma função responsável pela extração de características e uma para cálculo de similaridade. Diferentes descritores de imagens podem ser utilizados separadamente ou em conjunto para realizar o processo de recuperação. Esses são comumente divididos em globais e locais, os primeiros descrevem pequenos *clusters* de *pixels* ou trechos dentro da imagem e os últimos descrevem o conteúdo visual da imagem inteira (PAREEK; MANDORIA, 2017).

Dentro dessas duas categorias, os descritores ainda podem ser classificados como descritores de cor, textura e forma. Descritores de cor, como o próprio nome diz, extraem características de cor da imagem; os de textura medem propriedades da imagem tais como suavidade, aspereza, e regularidade; enquanto os de forma extraem propriedades referentes à forma de uma determinada região que está sendo procurada, normalmente por meio da segmentação ou detecção de bordas da imagem (MARDONES; ALLENDE; MORAGA, 2013; JAIN; SINGH, 2016).

Assim, em suma, um sistema baseado em CBIR recebe como entrada uma imagem de consulta e retorna ao usuário as imagens mais similares à essa. Para tal, o sistema deve manter um *dataset* de imagens e, dada uma *query*, comparar seu vetor de características, obtido em um processo de extração de características visuais, com o de cada uma das imagens da coleção de dados, a fim de computar a similaridade existente entre os itens. Lembrando que tanto as características da imagem de consulta quanto as do *dataset* disponível podem ser extraídas utilizando um ou mais descritores de imagens (DHARANI; AROQUIARAJ, 2013; SCHAEFER, 2013; MASOOD; SHAHID; SHARIF, 2018).

Depois que a similaridade entre os vetores de características é computada, as imagens do conjunto de dados devem ser recuperadas e apresentadas ao usuário em ordem decrescente de similaridade, isto é, as mais similares necessariamente devem aparecer no topo da lista dos itens recuperados pelo algoritmo. Note que, diferentes descritores produzem diferentes similaridades e, consequentemente, itens recuperados em ordens diferentes.

A ordenação final das imagens é um aspecto crucial em sistemas para recuperação de imagens baseada em conteúdo. Como o conjunto de imagens mantido por sistemas desse tipo é geralmente grande, os usuários tendem a verificar apenas os resultados localizados no topo da lista de itens recuperados. Assim, a percepção do usuário quanto à qualidade do sistema depende criticamente da relevância dos primeiros resultados apresentados (FARIA et al., 2010).

Tendo em vista a importância dessa relevância, alguns trabalhos (FARIA et al., 2010; LI et al., 2013; WAN et al., 2015) têm abordado técnicas baseadas em CBIR em conjunto com métodos de ranqueamento de informações, ao invés de apenas ordenar decrescentemente as similaridades obtidas à partir da comparação entre imagens, com o intuito de aumentar a probabilidade de que os resultados finais apresentados ao usuário sejam de fato relevantes. Para maior compreensão acerca da tarefa de aprender a ranquear, o ranqueamento de informações é detalhadamente abordado no capítulo 4. No entanto, antes de abordar esse tema, na seção

a seguir um exemplo que ilustra as tarefas de recuperação e ranqueamento de imagens em conjunto é apresentado a fim de introduzir conceitos abordados no referido capítulo.

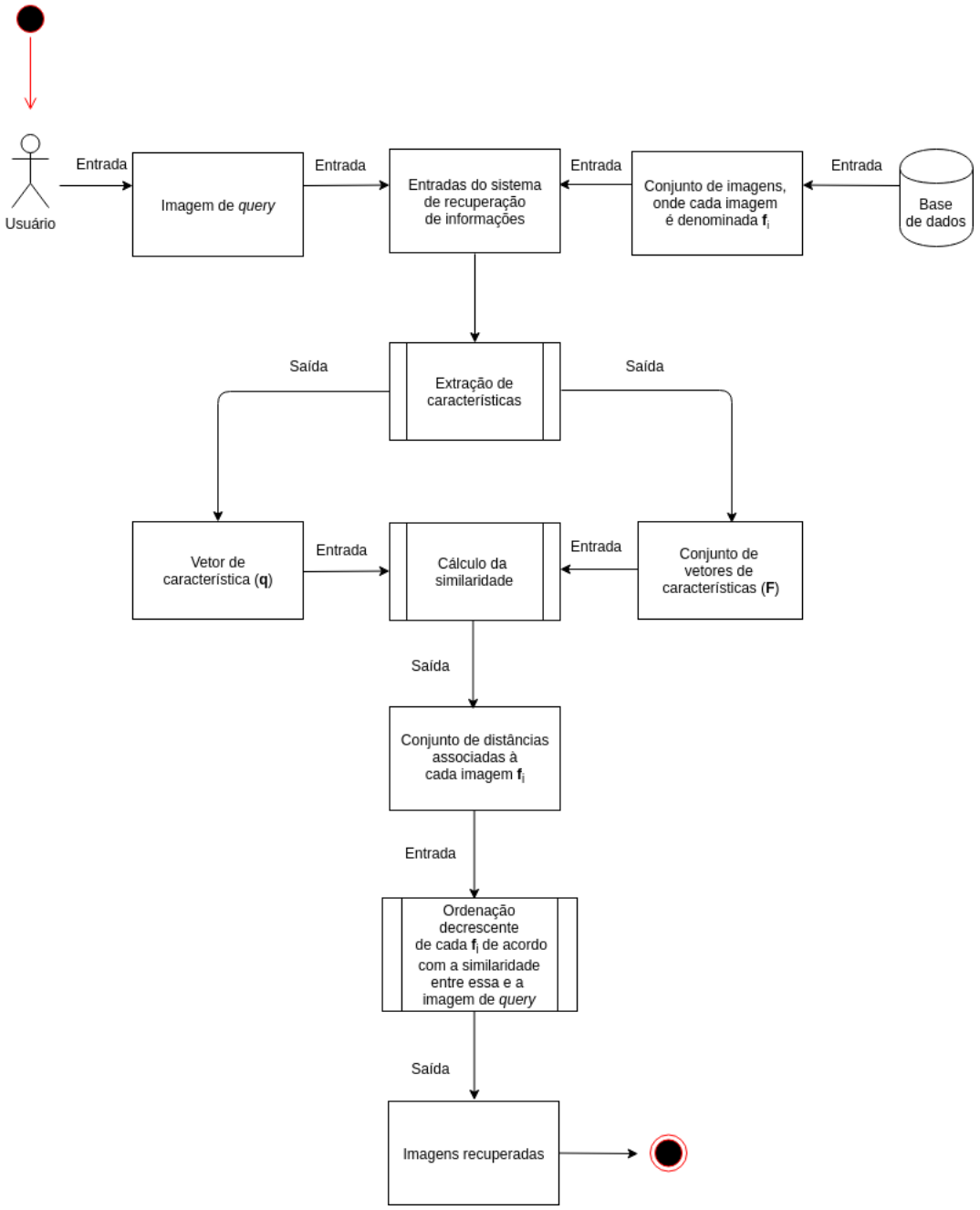
2.3 Recuperação e Ranqueamento de Imagens

Conforme previamente mencionado, um sistema destinado a recuperar informações necessariamente recebe como entrada um objeto que representa uma dada necessidade do usuário. Quando trata-se de uma aplicação que objetiva recuperar imagens baseadas em seu conteúdo, esse objeto deve representar, de modo fiel, o conteúdo que deseja-se recuperar, isto é, a entrada deve ser o mais semelhante possível às saídas que o usuário espera receber do sistema. Por exemplo, suponha que ao utilizar uma aplicação que armazena um catálogo de imagens digitais de diferentes monumentos históricos, o usuário deseje recuperar as imagens da torre Eiffel nesse conjunto de dados; para que o usuário obtenha resultados satisfatórios após o processo de recuperação, isto é, para que as imagens recuperadas sejam todas ou em sua maioria da torre Eiffel espera-se que uma imagem da torre seja fornecida como *input* ao sistema.

Assim, dado um conjunto de imagens, as características visuais de cada uma delas são extraídas e essas são armazenadas em vetores de características. Esses vetores, por sua vez, são armazenados em um conjunto que, à partir daqui, para fins didáticos, chamaremos de F. O vetor de características referente à imagem de consulta também deve ser armazenado; esse é denominado q. Tendo essas características salvas, para cada imagem (f_i) do conjunto F é calculada a similaridade entre ela e o vetor q. Note que cada vetor f_i é uma representação da imagem a qual está associado. Após calculadas as similaridades, lembrando que, geralmente, a similaridade é calculada utilizando uma função de distância, essas devem ser ordenadas de forma decrescente, isto é, os itens f_i que apresentam distâncias menores quando comparados à q devem ser exibidos primeiro ao usuário; partindo da premissa que, quanto menor a distância entre duas imagens, maior a semelhança entre elas.

A fim de ilustrar as etapas que compõem o processo de recuperação de imagens em um sistema padrão baseado em CBIR, um diagrama composto por essas é apresentado na Figura 1.

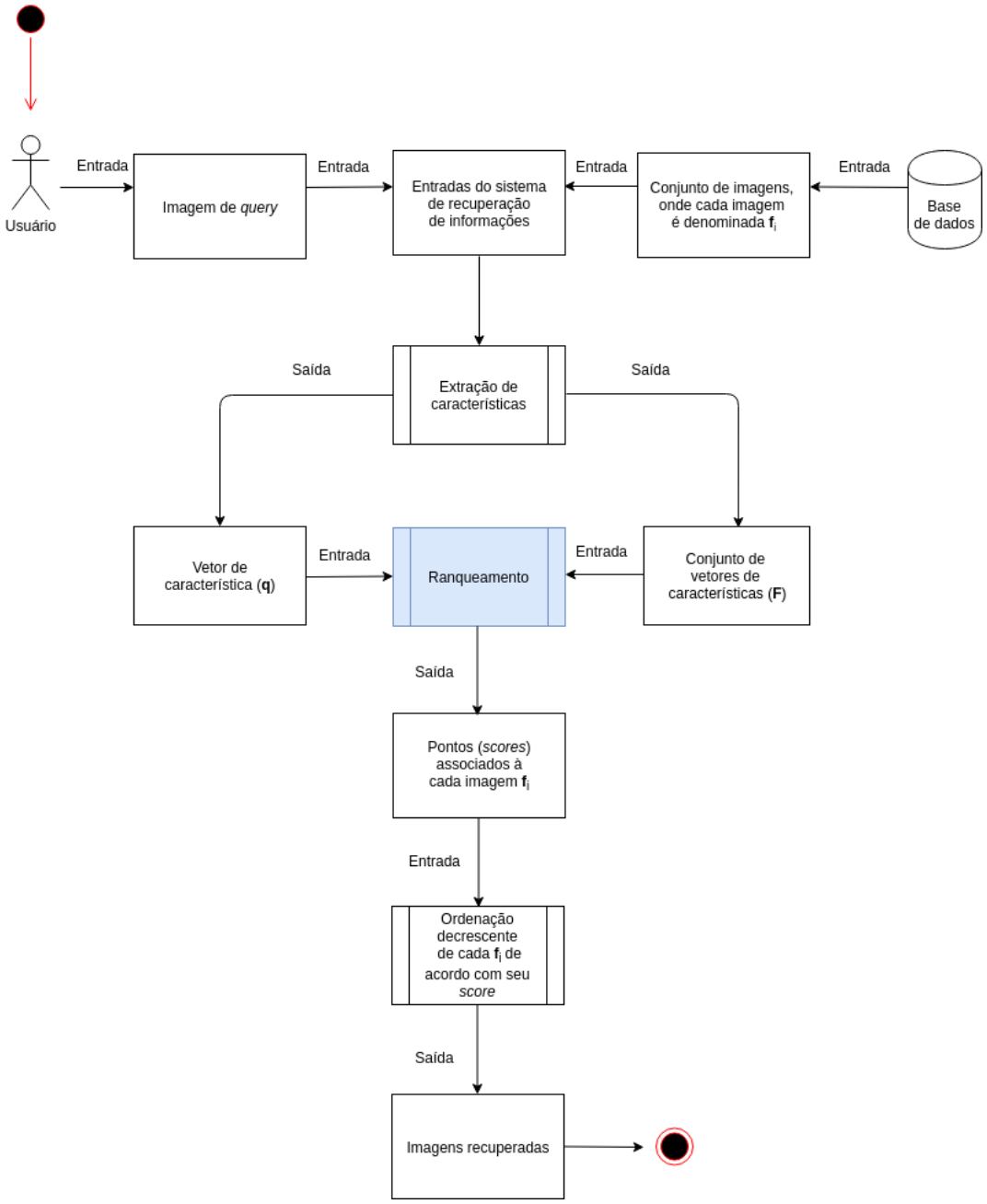
Figura 1 – Diagrama contendo as etapas que compõem o processo de recuperação de imagens por conteúdo.



Fonte: Elaborado pela autora.

Conforme é possível observar no diagrama da Figura 1, não necessariamente um sistema de recuperação de imagens utiliza um algoritmo de ranqueamento para ranquear as imagens por similaridade. No entanto, como mencionado anteriormente, é possível utilizar técnicas para aprender um ranqueamento ideal das imagens em uma base de dados a fim de melhorar a acurácia do sistema e assim apresentar resultados mais relevantes ao usuário. O diagrama da Figura 2 ilustra as etapas necessárias para incorporar o processo de ranqueamento ao sistema.

Figura 2 – Diagrama contendo as etapas que compõem o processo de recuperação de imagens utilizando ranqueamento de informações.



Fonte: Elaborado pela autora.

No diagrama da Figura 2 o processo de ranqueamento das informações, nesse caso, das imagens, é representado pela etapa descrita como 'Ranqueamento'. Maiores detalhes acerca desse processo são descritos no capítulo 4; por hora, a fim de proporcionar melhor compreensão do diagrama, vale antecipar que, em geral, técnicas destinadas a realizar o ranqueamento de itens associam à cada um deles um *score* que representa a similaridade entre esse e a *query* fornecida inicialmente como entrada ao sistema. Esses itens, por fim, são ranqueados por *score* e apresentados ao usuário.

3 Classificadores baseados em Floresta de Caminhos Ótimos

No cenário que abrange a tarefa de classificação em Ciência da Computação, padrões geralmente são representados por vetores de características obtidos a partir de amostras disponíveis em uma base de dados que pode estar totalmente, parcialmente ou não rotulada; sendo o rótulo da amostra equivalente a classe à qual ela pertence.

O algoritmo de um classificador que utiliza Aprendizado de Máquina, do inglês *Machine Learning* (ML), para classificar informações, em geral, necessita de um conjunto de dados para treinamento e outro para testes; sendo que esses devem ser disjuntos e o primeiro é utilizado para o algoritmo aprender os padrões na base de dados, enquanto o último é utilizado para classificar novos itens e validar a acurácia do algoritmo na classificação desses. Logo, o conjunto de amostras de treinamento é utilizado para ensinar o classificador de padrões e o de testes para validação.

Assim, dado um conjunto de amostras de treinamento, o classificador de padrões busca aprender uma função $\mathbf{h} \in \{H\}$ - onde H representa o conjunto de todas as funções hipóteses - que mapeie um determinado vetor de características $\mathbf{x} \in \{X\}$ - em que X representa o conjunto de todos os vetores de características que representam as amostras do problema - em um rótulo **r**, baseado em um conjunto de rótulos L, ou seja, um classificador objetiva predizer corretamente a classe de cada amostra do conjunto de teste com base em seu respectivo vetor de características.

Tome como exemplo um problema de classificação binária, ou seja, em que existem apenas 2 classes: imagens de paisagens (classe A) e imagens de animais (classe B). No contexto de classificação, objetiva-se predizer corretamente a classe correspondente à cada amostra; assim, no exemplo dado, a ideia é classificar corretamente as imagens de um conjunto de teste como sendo de paisagens ou animais.

As técnicas de Aprendizado de Máquina utilizadas para identificação de padrões dividem-se em: supervisionadas – quando todos os rótulos das amostras do conjunto de treinamento são conhecidas - ; não-supervisionadas - quando não se tem ou não se utiliza nenhum conhecimento acerca das classes das amostras de treinamento; e semi-supervisionadas – quando se conhece apenas parte dos rótulos das amostras de treinamento ou se faz uso parcial desses.

Resumidamente falando, o método para classificação supervisionada de padrões baseado em Floresta de Caminhos Ótimos, proposto por (PAPA., 2008), modela o problema de reconhecimento de padrões como sendo um grafo no qual as amostras são representadas pelos nós desse e seus arcos são definidos a partir de uma relação de adjacência. As amostras

que melhor representam as classes do conjunto de treinamento são denominadas protótipos e, após identificação dessas, um processo de competição é iniciado entre elas a fim de oferecer caminhos de custo ótimo para as demais amostras do conjunto de dados e seus respectivos rótulos.

O algoritmo de Floresta de Caminhos Ótimos é o mesmo utilizado na Transformada Imagem Floresta, do inglês *Image Foresting Transform* (IFT) (FALCãO; STOLFI; LOTUFO, 2004), que tem sido utilizada em diversas aplicações de processamento de imagens. O classificador baseado em OPF estende as aplicações da IFT do domínio da imagem para a classificação de padrões no espaço de características (PAPA., 2008).

Em suma, o classificador OPF baseia-se em:

- Um subconjunto de amostras protótipos e
- Uma função de custo de caminho para o grafo de treinamento a fim de agrupar amostras com características similares.

A ideia é dividir o grafo de treinamento em uma floresta de caminhos de custo mínimo, em que cada árvore tem como raiz um protótipo representando uma das *m* classes do conjunto de treinamento e todas as amostras pertencentes a árvore são classificadas com o mesmo rótulo da raiz. A classificação de uma nova amostra se dá a partir da busca pelo protótipo que lhe oferece o caminho ótimo dentre os caminhos oferecidos por todos os protótipos, isto é, a classificação baseia-se na força de conexidade entre o protótipo e a amostra nova; assim, o rótulo de uma nova amostra a ser classificada passa a ser o mesmo do protótipo mais fortemente conexo a ela.

Vale citar que foram realizadas comparações de desempenho entre o classificador baseado em Floresta de Caminhos Ótimos e os classificadores SVM *(Suport Vector Machine)*, ANN-MLP *(Artificial Neural Network – Multi-layer Perceptron)*, K-NN *(K Nearest Neighbor)* e BC *(Bayesian Classifier)* e o OPF demonstrou desempenho similar ao do SVM e superior aos demais (PAPA., 2008).

Nas subseções a seguir são elucidadas duas abordagens supervisionadas de classificadores baseados em OPF: uma que utiliza grafo completo para modelar o problema de classificação e outra que utiliza grafo k-*nn*.

3.1 Abordagem com Grafo Completo

Essa metodologia de classificação supervisionada baseada em Floresta de Caminhos Ótimos modela o problema de modo que as amostras do conjunto sejam os nós de um grafo completo. Os protótipos, amostras mais representativas do conjunto de treinamento, são escolhidos entre as amostras pertencentes à região de intersecção (fronteira) entre as classes.

A partir de um processo de competição entre protótipos, esses oferecem às demais amostras caminhos de menor custo e seus respectivos rótulos. Ao final desse processo, obtém-se um conjunto de treinamento dividido em árvores de caminhos ótimos e a união dessas compõe uma Floresta de Caminhos Ótimos (PAPA., 2008).

3.1.1 Fundamentação Teórica

Seja Z uma base de dados e Z_1 e Z_2 os conjuntos de treinamento e teste, respectivamente, possuindo $|Z_1|$ e $|Z_2|$ amostras, tal que $Z = Z_1 \cup Z_2$. Seja $\lambda(s)$ uma função que associa o rótulo correto i, $i = 1, 2, \ldots, c$ da classe i a qualquer amostra $s \in Z_1 \cup Z_2$. Seja também $S \in Z_1$ um conjunto de protótipos de todas as classes e $\vec{v}$ um algoritmo que extrai n atributos, tais como forma, cor e propriedades de textura, por exemplo, de qualquer amostra $s \in Z_1 \cup Z_2$, e retorna um vetor de atributos $\vec{v}(s) \in \Re^n$. A distância $d(s,t)$ entre duas amostras, s e t, equivale à distância entre seus respectivos vetores de características $\vec{v}(s)$ e $\vec{v}(t)$. O problema de classificação baseado em Floresta de Caminhos Ótimos consiste então em utilizar S, $(\vec{v}, d)$, Z_1 e Z_2 para projetar um classificador ótimo capaz de predizer corretamente o rótulo $\lambda(s)$ de qualquer amostra $s \in Z_2$.

Seja (Z_1, A) um grafo completo cujos nós representam as amostras de Z_1, onde qualquer par de amostras define um arco em A e seja um caminho uma sequência de amostras $\pi = \langle s_1, s_2, \ldots, s_k \rangle$, onde $(s_i, s_{i+1}) \in A$ para $1 \leq i \leq k-1$ e é considerado trivial se $\pi = \langle s_1 \rangle$. A cada caminho π é associado o custo dado por uma função de custos suave f, denotada $f(\pi)$; sendo π um caminho ótimo se $f(\pi) \leq f(\tau)$ para qualquer caminho τ, onde π e τ têm como nó final a mesma amostra s, independentemente de sua origem. Denota-se $\pi \cdot \langle s,t \rangle$ a concatenação do caminho π com término em s e arco (s,t).

Apesar do algoritmo OPF poder ser utilizado com qualquer função de custo suave capaz de agrupar amostras com propriedades similares, esse foi projetado usando a função de custo f_{max}:

$$\begin{aligned} f_{max}(\langle s \rangle) &= \begin{cases} 0 & \text{se } s \in S, \\ +\infty & \text{caso contrário} \end{cases} \\ f_{max}(\pi \cdot \langle s,t \rangle) &= \max\{f_{max}(\pi), d(s,t)\}, \end{aligned} \tag{1}$$

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

O algoritmo OPF associa um caminho ótimo $P^*(s)$ de S à toda amostra $s \in Z_1$, formando assim uma Floresta de Caminhos Ótimos P, a partir de uma função sem ciclos que associa a todo $s \in Z_1$ seu predecessor $P(s)$ em $P^*(s)$, ou uma marca nil quando $s \in S$. Seja $R(s) \in S$ a raiz de $P^*(s)$, que pode ser alcançada a partir de $P(s)$; o algoritmo OPF calcula, para cada $s \in Z_1$, o custo $V(s)$ de $P^*(s)$, o rótulo $L(s) = \lambda(R(s))$ e seu predecessor $P(s)$, conforme explicitado no Algoritmo 1.

ALGORITMO 1: Classificador supervisionado baseado em Floresta de Caminhos Ótimos com grafo completo

Entrada: Um conjunto de treinamento Z_1 λ-rotulado, protótipos $S \subset Z_1$ e o par (v, d) para vetor de características e cálculo das distâncias.
Saída: Floresta de Caminhos Ótimos P, mapa de valores de custo de caminhos V e mapa de rótulos L.

```
1  Fila de prioridades Q e variável cst.
2  para todo s ∈ Z1, faça P(s) ← nil e V(s) ← +∞;
3  para todo s ∈ S, faça V(s) ← 0, P(s) ← nil, L(s) = λ(s) e insira s em Q;
4  enquanto Q é não vazia, faça
5      Remova de Q uma amostra s tal que V(s) é mínimo;
6      para cada t ∈ Z1 tal que s ≠ t e V(t) > V(s) faça
7          Calcule cst ← max{V(s), d(s,t)}.
8          se cst < V(t) então
9              se V(t) ≠ +∞ então
10                 remova t de Q.
11             P(t) ← s, L(t) ← L(s) e V(t) ← cst;
12             Insira t em Q.
```

As linhas 2–3 inicializam os mapas de custo e inserem protótipos em Q. O laço principal computa um caminho ótimo de S para cada amostra $s \in Z_1$ em uma ordem não decrescente de custos (linhas 4–12). A cada iteração um caminho de custo ótimo $V(s)$ é obtido em P (linha 5). Empates são resolvidos em Q utilizando a abordagem FIFO (first-in-first-out), isto é, quando dois caminhos atingem uma determinada amostra s com o mesmo custo mínimo, s é associado ao primeiro caminho que o atingiu. O restante das linhas verifica se o caminho que atinge uma amostra adjacente t através de s é menos custoso que o caminho que termina em t. Se sim, atualiza Q, $P(t)$, $L(t)$ e $V(t)$. No final do algoritmo, V armazena o valor do custo do caminho ótimo de S para cada amostra $s \in Z_1$ de acordo com f_{max}.

3.1.2 Treinamento

S^* é considerado um conjunto ótimo de protótipos quando o Algoritmo 1 propaga os rótulos $L(s) = \lambda(s)$ para todo $s \in Z_1$. Assim, S^* pode ser obtido explorando a relação teórica entre a Árvore de Espalhamento Mínimo, do inglês *Minimum Spanning Tree* (MST), e a árvore de caminhos mínimos para f_{max}. Dessa forma, a fase de treinamento consiste basicamente em encontrar S^* e um classificador OPF enraizado em S^*.

Ao calcular uma MST no grafo completo (Z_1, A), obtém-se um grafo conexo acíclico cujos nós são todas as amostras em Z_1 e as arestas são não direcionadas e ponderadas (Figura 3b). Seus pesos são definidos pela distância d entre os vetores de características de amostras adjacentes. Está árvore de espalhamento é ótima uma vez que a soma dos pesos de seus arcos é mínima quando comparada à outras árvores de espalhamento no grafo completo.

Na MST, cada par de amostras é conectado por um caminho, o qual é ótimo de acordo com f_{max}. Os protótipos ótimos correspondem aos elementos conectados na MST com diferentes rótulos em Z_1, isto é, os elementos mais próximos de classes diferentes. Ao remover os arcos entre classes diferentes, as amostras adjacentes tornam-se protótipos em S^* e o Algoritmo 1 pode calcular uma Floresta de Caminhos Ótimos livre de erros de classificação em Z_1.

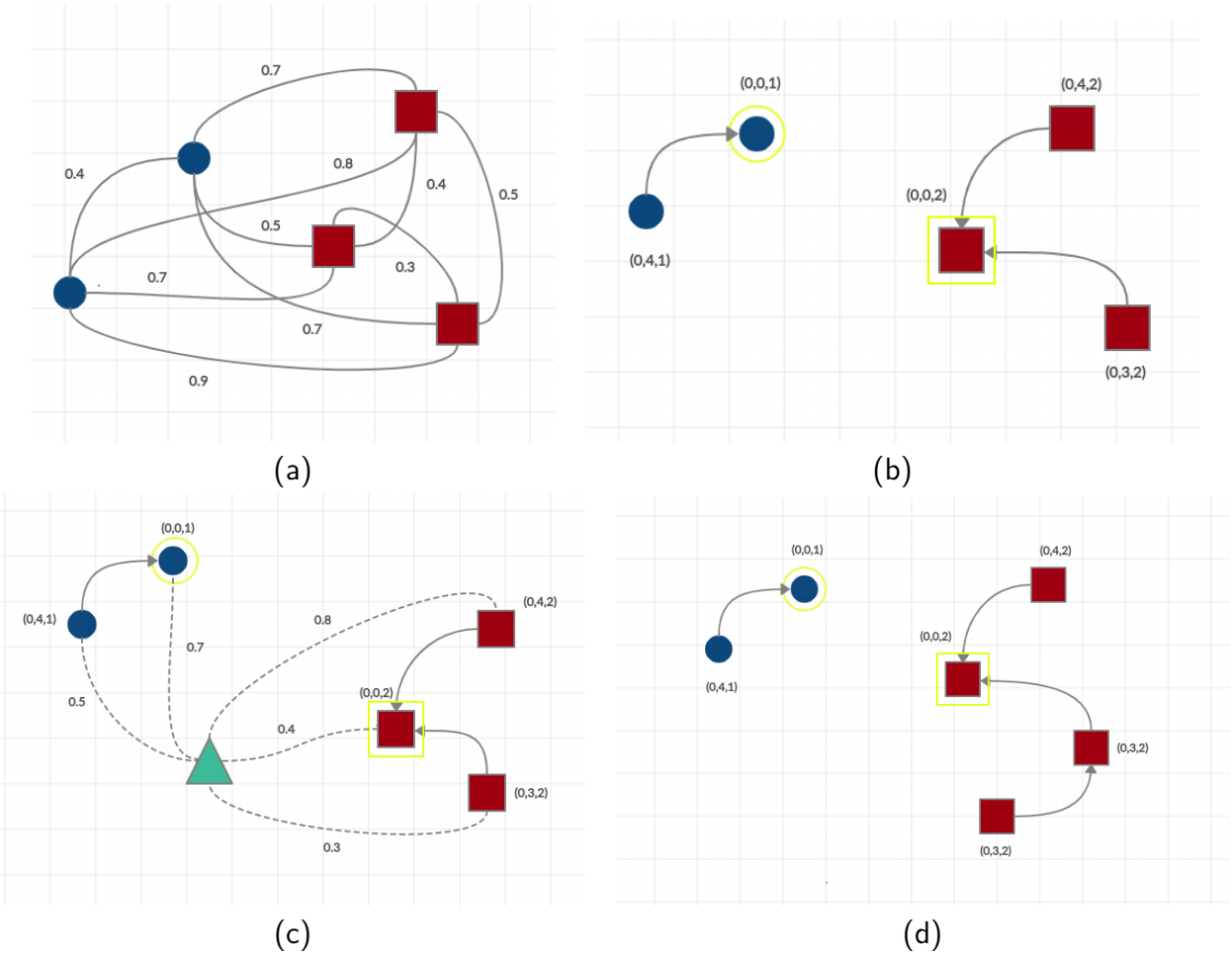
3.1.3 Classificação

Para toda amostra $t \in Z_2$, tem-se os arcos conectando t com amostras $s \in Z_1$, tornando t parte do grafo (ver Figura 3c, na qual a amostra t é representada pelo triângulo no grafo). Ao considerar todos os caminhos possíveis entre S^* e t, almeja-se encontrar o caminho ótimo $P^*(t)$ de S^* até t com a classe $\lambda(R(t))$ de seu protótipo mais fortemente conexo $R(t) \in S^*$. Este caminho pode ser identificado de forma incremental ao avaliar o valor do custo ótimo $V(t)$ como

$$V(t) = \min\{\max\{V(s), d(s,t)\}\},\ \forall s \in Z_1. \tag{2}$$

Seja $s^* \in Z_1$ o nó que satisfaz a Equação 2, isto é, o predecessor $P(t)$ no caminho ótimo $P^*(t)$ e dado que $L(s^*) = \lambda(R(t))$, na etapa de classificação $L(s^*)$ é considerada a classe de t (Figura 3d). Uma amostra é classificada incorretamente quando $L(s^*) \neq \lambda(t)$.

Figura 3 – Etapa de teste do classificador baseado em OPF com grafo completo.



Fonte: Adaptado pela autora à partir de (PAPA., 2008).

Conforme Figura 3, em 3a tem-se um grafo completo ponderado nas arestas para um determinado conjunto de treinamento. Em 3b tem-se uma Floresta de Caminhos Ótimos resultante para f_{max} e dois protótipos (nós contornados). As entradas (x, y) acima dos nós denotam, respectivamente, o custo e o rótulo das amostras. Os arcos direcionados indicam os nós predecessores no caminho ótimo. Em 3c a amostra de teste (triângulo verde) e suas conexões (linhas tracejadas) com os nós de treinamento são representadas. Por fim, em 3d o caminho ótimo do protótipo mais fortemente conexo, seu rótulo (quadrado) e o custo de classificação 0.3 são associados à amostra de teste.

3.2 Abordagem com Grafo k-*nn*

Essa abordagem também tem como base uma Floresta de Caminhos Ótimos e utiliza uma função de densidade de probabilidade (*fdp*) para ponderar os nós de um grafo não completo, no qual os arcos são estabelecidos à partir de uma relação de adjacência em que, para cada nó, considera-se seus k vizinhos mais próximos (grafo k-*nn*). Nesse caso, o peso dos arcos também

corresponde à distância entre as amostras. Por sua vez, os valores de densidade de probabilidade que ponderam os nós são calculados utilizando o peso dos arcos. A função de valor de caminho associa ao nó terminal *s* de cada caminho um valor de densidade inicial e um valor mínimo entre as densidades dispostas ao longo do caminho. Assim, a Floresta de Caminhos Ótimos é obtida ao maximizar os valores das densidades ao longo de todos os caminhos entre as amostras e protótipos. As raízes da floresta, os protótipos de cada classe, formam um subconjunto com os valores máximos de *fdp*. Cada raiz define uma árvore de caminho ótimo composta pelas suas amostras mais fortemente conexas e que corresponde à zona de influência do protótipo. A diferença principal entre esta abordagem com grafo k-*nn* e a que utiliza grafo completo, apresentada na seção 3.1, está na etapa de estimar protótipos. A abordagem com grafo completo estima protótipos na fronteira das classes, enquanto a metodologia apresentada neste tópico estima os protótipos nos pontos do espaço de características onde há alta concentração de amostras.

3.2.1 Fundamentação Teórica

Seja Z uma base de dados λ-rotulada e Z_1 e Z_2 os conjuntos de treinamento e teste, respectivamente, possuindo $|Z_1|$ e $|Z_2|$ amostras, tal que $Z = Z_1 \cup Z_3$. Dado um grafo (Z_1, A_k), o algoritmo OPF com grafo k-nn divide o conjunto Z_1 em uma Floresta de Caminhos Ótimos a partir de uma função de densidade de probabilidade.

Nesta abordagem tem-se um grafo com pesos nos vértices e arcos. A distância $d(s,t)$ entre duas amostras $s, t \in Z_1$ determina o peso do arco (s,t). E os nós são ponderados pelos seus valores de densidade de probabilidade, que são calculados usando o peso dos arcos, conforme Equação 3:

$$\rho(s) = \frac{1}{\sqrt{2\pi\sigma^2}|A(s)|} \sum_{\forall t \in A(s)} \exp\left(\frac{-d^2(s,t)}{2\sigma^2}\right), \tag{3}$$

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

A função de valor de caminho dada pela Equação 4 associa a um nó terminal s de cada caminho o valor mínimo entre os valores de densidade ao longo do mesmo e um valor de densidade inicial.

$$\begin{aligned} f_2(\langle t \rangle) &= \begin{cases} \rho(t) & \text{se } t \in S \\ h(t) & \text{caso contrário} \end{cases} \\ f_2(\pi_s \cdot \langle s,t \rangle) &= \min\{f(\pi_s), \rho(t)\}. \end{aligned} \tag{4}$$

O caminho de valor máximo para cada amostra s, a partir de um conjunto de protótipos (raízes das árvores), divide o conjunto de treinamento Z_1 em uma Floresta de Caminhos Ótimos.

Sendo as raízes da floresta um subconjunto dos máximos da *fdp*, em que cada raiz define uma árvore de caminhos ótimo e é considerada a zona de influência do respectivo máximo; essa é composta por suas amostras mais fortemente conexas, as quais recebem o mesmo rótulo do protótipo que representa sua raiz, como pode ser verificado no Algoritmo 2.

Inicialmente, conforme linha 2 do Algoritmo 2, todos os caminhos são triviais com valores $f_1(\langle t \rangle) = \rho(t) - \delta$. Os máximos globais da *fdp* são os primeiros elementos a serem removidos de Q, os quais são identificados como sendo as raízes da floresta pelo teste $P(s) = nil$ na linha 5, onde atualiza-se o seu valor de caminho correto $f_1(\langle t \rangle) = V(s) = \rho(t)$. Cada nó s removido de Q oferece um caminho $\pi_s \cdot \langle s, t \rangle$ para cada nó t adjacente no laço da linha 7 até a linha 11. Se o valor de caminho $f_1(\pi_s \cdot \langle s, t \rangle) = \min\{V(s), \rho(t)\}$ (linha 8) é melhor que o valor de caminho atual $f_1(\pi_t) = V(t)$ (linha 9), então π_t é atualizado para $\pi_s \cdot \langle s, t \rangle$; isto é, o valor do caminho e o rótulo de t são atualizados (linha 10). Máximos locais da *fdp* são identificados como raízes durante a execução do algoritmo, o qual tem como saída o mapa de rótulos L, que contém os valores de caminho V e a Floresta de Caminhos Ótimos P.

ALGORITMO 2: Classificador supervisionado baseado em Floresta de Caminhos Ótimos com grafo k-nn

Entrada: Grafo k-nn (Z_1, A_k), $\lambda(s)$ para todo $s \in Z_1$ e função de valor de caminho f_1.

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

1 Fila de prioridade Q e a variável tmp.
2 **para todo** $s \in Z_1$, **faça** $P(s) \leftarrow nil$, $V(s) \leftarrow \rho(s) - \delta$, $L(s) \leftarrow \lambda(s)$ e insira s em Q;
3 **enquanto** Q *é não vazia,* **faça**
4 Remova de Q uma amostra s tal que $V(s)$ é máximo;
5 **se** $P(s) = nil$ **então**
6 $V(s) \leftarrow \rho(s)$.
7 **para** *cada* $t \in A_k(s)$ *e* $V(t) < V(s)$ **faça**
8 $tmp \leftarrow \min\{V(s), \rho(t)\}$.
9 **se** $tmp > V(t)$ **então**
10 $L(t) \leftarrow L(s)$, $P(t) \leftarrow s$, $V(t) \leftarrow tmp$;
11 Atualize posição de t em Q.

3.2.2 Treinamento

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

Espera-se que cada classe seja representada por pelo menos um máximo da *fdp* e $L(t) = \lambda(t)$ para todo $t \in Z_1$, ou seja, objetiva-se erro zero de classificação no conjunto de treinamento. Entretanto, essas propriedades não podem ser garantidas com a função de valor

de caminho f_2 e o melhor valor k^*. A fim de assegurar tais propriedades, primeiro encontra-se k^* utilizando f_2 e então executa-se o Algoritmo 2 mais uma vez usando a função de valor de caminho f_3, dada por

$$
\begin{aligned}
f_3(\langle t \rangle) &= \begin{cases} \rho(t) & \text{se } t \in S \\ \rho(t) - \delta & \text{caso contrário} \end{cases} \\
f_3(\pi_s \cdot \langle s, t \rangle) &= \begin{cases} -\infty & \text{se } \lambda(t) \neq \lambda(s) \\ \min\{f_2(\pi_s), \rho(t)\} & \text{caso contrário.} \end{cases}
\end{aligned} \tag{5}
$$

A Equação 5 pondera todos os arcos $(s,t) \in A_k$ tais que $\lambda(t) \neq \lambda(s)$ com $d(s,t) = -\infty$, impedindo que tais arcos pertençam a algum caminho ótimo. O processo de treinamento é realizado pelo Algoritmo 3.

ALGORITMO 3: Etapa de treinamento do classificador OPF supervisionado com grafo k-nn.

Entrada: Conjunto de treinamento Z_1, $\lambda(s)$ para todo $s \in Z_1$, k_{max} e funções de valor de caminho f_2 e f_3.

Saída: Mapa de rótulos L, mapa de valores de caminho V e a Floresta de Caminhos Ótimos P.

```
Variáveis i, k, k*, MaxAcc ← −∞, Acc e vetores FP e FN de tamanho c.
para k = 1 até k_max faça
    Crie um grafo (Z_1, A_k) ponderado nos nós através da Equação 3;
    Calcule (L, V, P) utilizando o Algoritmo 2 com f_2;
    para cada classe i = 1, 2, ..., c, faça
        FP(i) ← 0 e FN(i) ← 0
    para cada amostra t ∈ Z_1 faça
        se L(t) ≠ λ(t) então
            FP(L(t)) ← FP(L(t)) + 1;
            FN(λ(t)) ← FN(λ(t)) + 1.
    Calcule a acurácia.
    se Acc > MaxAcc então
        k* ← k e MaxAcc ← Acc.
    Remova o grafo (Z_1, A_k);
Crie o grafo (Z_1, A_{k*}) ponderado nos nós por meio da Equação 3.;
Calcule (L, V, P) utilizando o Algoritmo 2 com a função f_3.
```

3.2.3 Classificação

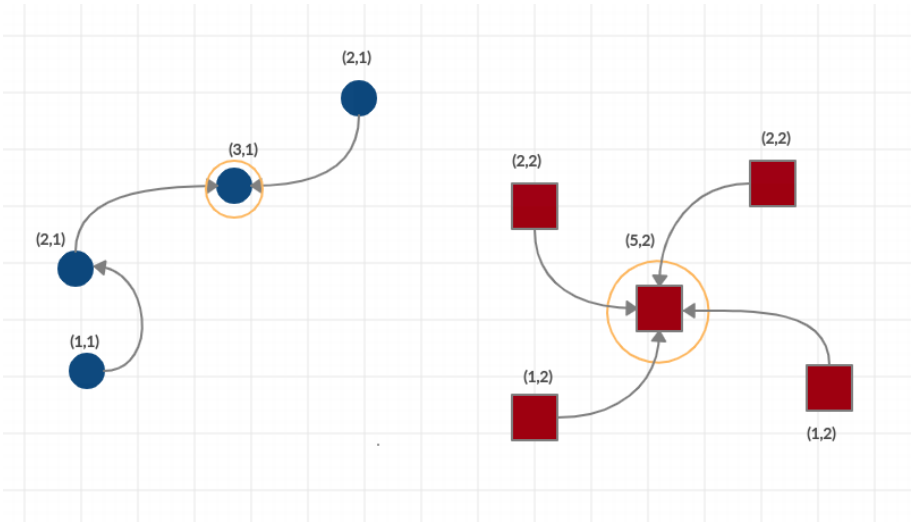
Uma amostra $t \in Z_3$ pode ser associada a uma dada classe $i, i = 1, 2, \ldots, c$ ao identificar qual raiz (protótipo) lhe oferece o caminho ótimo, considerando que a amostra em questão pertence à floresta. Considerando os k-vizinhos mais próximos de t em Z_3, pode-se

utilizar a Equação 3 para calcular $\rho(t)$, avaliar os caminhos ótimos $\pi_s \cdot \langle s, t \rangle$ e selecionar aquele que satisfaz a Equação 6. O rótulo de t será o mesmo rótulo do seu predecessor $P(t)$, ou seja, $L(t) = L(P(s))$. Uma amostra é classificada incorretamente quando $L(t) \neq \lambda(t)$.

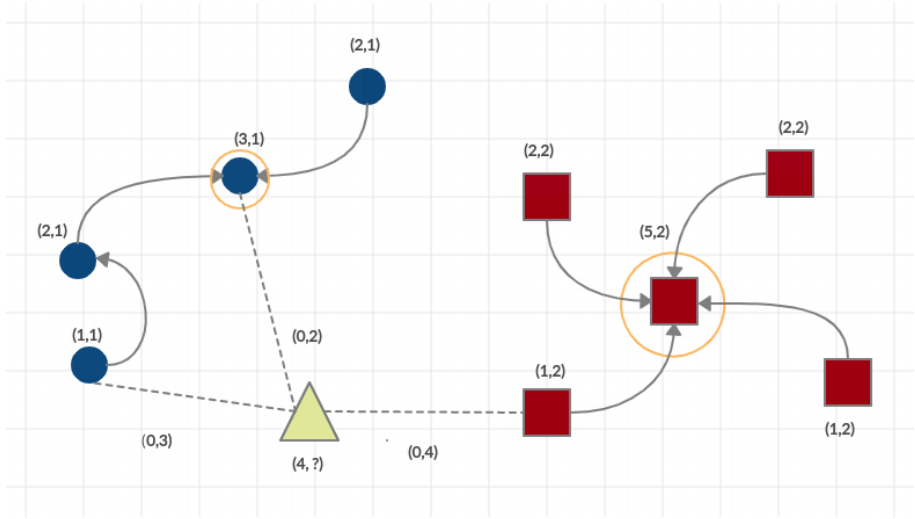
$$V(t) = \max_{\forall (s,t) \in A_{k^*}} \{\min\{V(s), \rho(t)\}\}. \tag{6}$$

A Figura 4 ilustra a etapa de classificação do classificador com grafo k-nn, na qual uma Floresta de Caminhos Ótimos é obtida através do Algoritmo 2 e cujos elementos são ponderados nos nós com seus respectivos valores de densidade. Os indicadores (x, y) acima dos nós são, respectivamente, o seu valor de densidade e o rótulo da classe a qual ele pertence. Os elementos circulados representam os máximos de cada classe. A Figura 4b apresenta o processo de classificação, onde um nó $t \in Z_3$ (triângulo) a ser classificado é adicionado ao grafo e conectado aos seus k-vizinhos mais próximos e seu valor de densidade é calculado, sendo que nesse ponto seu rótulo ainda é desconhecido. A Figura 4c ilustra o processo final da classificação, no qual a amostra t é classificada com o rótulo da amostra $s \in Z_1$ que satisfaz a Equação 6, ou seja, $L(t) = L(s)$. No exemplo apresentado, $k = 3$ e o elemento a ser classificado foi conquistado pelo máximo da classe círculo.

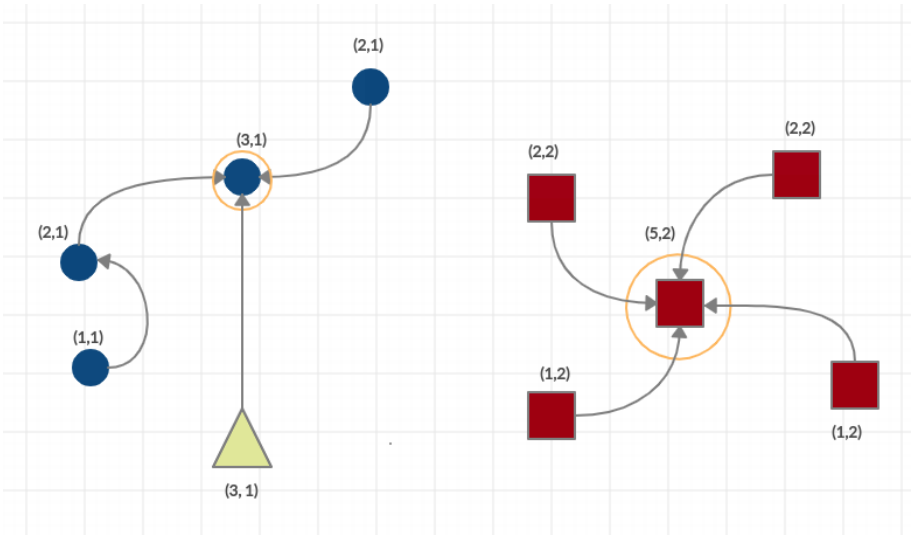
Figura 4 – Etapa de teste do classificador baseado em OPF com grafo k-nn.



(a)



(b)



(c)

Fonte: Adaptado pela autora à partir de (PAPA., 2008).

4 Ranqueamento de Informações

Nas últimas décadas e principalmente nos últimos anos, tem-se presenciado um rápido crescimento da informação digital. Esse crescimento se deve em especial ao advento da internet, às melhorias referentes à capacidade de armazenamento de informação digital, ao crescimento do acesso às tecnologias que permitem a criação e armazenamento dessas informações, tais como *smartphones*, *tablets* e computadores e ao aumento do uso de sistemas de informação, aplicativos e redes sociais por indivíduos.

De acordo com (GANTZ; REINSEL, 2012), em 2005 existiam 130 exabytes de informação digital armazenadas no mundo e estima-se que até 2020 esse número chegará à 40.000 exabytes, o que equivale à 40 trilhões de gigabytes. Com o contínuo crescimento de dados e frente ao imenso número de informação já disponível atualmente, a recuperação de informações realizada de forma eficaz e o correto ranqueamento dos resultados recuperados apresentam-se como atividades essenciais a diferentes sistemas de informação.

Em um primeiro momento, o ranqueamento de informações pode ser considerado um simples processo de ordenação. No entanto, essa consideração se mostra verdadeira somente quando o conjunto de dados a ser ranqueado não é muito grande e sabe-se como realizar essa ordenação, isto é, quando a característica utilizada para definir a ordem dos dados é conhecida. Um exemplo trivial de ranqueamento é o da ordenação das provas de alunos de uma dada turma de acordo com seu desempenho, que, por sua vez, é medido à partir da nota obtida na avaliação. Aqui, a nota da prova é utilizada como característica para definir a ordem no ranqueamento e, como o número de itens a ser ordenado é pequeno, o processo se torna fácil e pode ser realizado de forma manual.

No entanto, dada a quantidade de dados digitais disponíveis atualmente e considerando um cenário no qual aborda-se o ranqueamento em conjunto com a recuperação de informações, esse apresenta-se como não trivial e inviável de ser realizado manualmente. Dessa forma, dado o considerável número de dados disponíveis, que nem sempre apresentam relação ou dependência entre si à primeira vista, torna-se difícil descobrir como definir a ordem dos itens no ranqueamento utilizando somente recursos humanos. Nessa situação, geralmente, sabe-se apenas que, para certo problema, o dado é relevante ou não, mas não se sabe quais dados são mais relevantes entre si e, consequentemente, como realizar o ranqueamento dessas informações de forma assertiva.

Assim, frente ao crescente aumento de informação digital disponível e à utilização de computadores para a realização de diferentes atividades, a recuperação de informações e também o ranqueamento dessas tornaram-se tarefas cujos resultados apresentam-se menos custosos e mais fidedignos quando essas são realizadas automaticamente por meio do uso de

diferentes algoritmos (SINGHAL; GOOGLE, 2001). Nos últimos anos, a utilização de técnicas de aprendizado de máquina voltadas à criação de modelos de ranqueamento apresentou-se como uma boa solução para a melhora na eficácia dos algoritmos destinados a esse fim e cresceu principalmente devido à sua aplicação em mecanismos de busca na web (LI, 2011a).

O termo aprender a ranquear refere-se à técnicas de aprendizado de máquina, em sua maioria supervisionado, cujo objetivo consiste em treinar um modelo na tarefa de ranqueamento. Essas técnicas são utilizadas em diferentes aplicações dentro da Ciência da Computação e essas estão comumente relacionadas à recuperação de informações, mineração de dados e ao processamento de linguagem natural. Tradução de máquina, recuperação de imagens, filtragem colaborativa, resumo de documentos e resposta à questões, do inglês *question answering*, são alguns exemplos de aplicações que utilizam LTR (LI, 2011a).

Ao utilizar técnicas de aprendizado de máquina para aprender a ranquear, tem-se como objetivo aprender uma função e aplicar o modelo gerado à tarefa de ranqueamento, de modo que, ao fim do processo se possa predizer uma lista de objetos pontuados de acordo com sua relevância. Para a construção do modelo, considera-se um conjunto de dados dividido em dois subconjuntos: um composto por dados definidos como *queries* e outro formado pelos demais dados do conjunto. Além desses dados, na fase de treinamento do modelo, utiliza-se uma lista de rótulos de relevância que determina quão relevante um dado, também chamado objeto, é para dada *query*. A função de ranqueamento é definida durante essa fase. Na etapa de testes, utilizando dados não informados anteriormente ao modelo e cujos rótulos de relevância são desconhecidos, espera-se que esse seja capaz de predizer precisamente uma lista ranqueada desses itens (LI, 2011a).

LTR pode ser muito útil quando aplicado como solução para criação de um modelo de ranqueamento em atividades que envolvem recuperação de informações melhorando a eficácia dessas; uma vez que, um modelo treinado na tarefa de ranqueamento tem como objetivo aprender uma função que, dada uma *query* e um conjunto de objetos candidatos, encontre o ranqueamento apropriado dos itens de acordo com sua relevância. Como anteriormente citado, os experimentos realizados nesse trabalho abordam ambos os temas utilizando imagens digitais como objeto de estudo.

LTR e IR têm sido amplamente estudados pela comunidade de aprendizado de máquina nos últimos anos. O interesse por essas áreas de pesquisa se deu principalmente pelo rápido crescimento de dados disponíveis em diferentes sistemas de informação e consequente necessidade de recuperar informações específicas em meio à essa grande massa de dados. Com essa quantidade considerável de dados disponíveis, a oportunidade de utilizá-los de forma inteligente, com o intuito de aumentar a satisfação de usuários, impulsionar vendas ou descobrir informações estratégicas para a manutenção de um cliente, por exemplo, são de grande interesse de organizações que desejam conquistar ou manter uma posição de destaque no mercado; consequentemente, frente à essa e outras aplicações, técnicas capazes de fornecer

essas informações de forma eficaz se fazem cada vez mais necessárias.

Tendo a tarefa de recuperação de informações como exemplo, suponha uma coleção de documentos mantida por um sistema de informação e uma consulta feita por meio de informação textual nesse. Para recuperar os itens mais relevantes para a consulta realizada, o sistema utiliza um algoritmo que contém uma função de ranqueamento que objetiva encontrar a relevância entre cada documento da coleção e a dada consulta, atribuindo a cada documento uma pontuação. Essa pontuação, por sua vez, é utilizada para ranquear os itens da coleção ao final do processo. Essa lista de pontos, também denominados *scores*, é a saída gerada por métodos de ranqueamento.

(LI, 2014) divide o problema de aprender a ranquear em dois tipos: criação de ranqueamento (ou simplesmente ranqueamento), que geralmente utiliza aprendizado supervisionado; e agregação de ranqueamento, que utiliza aprendizado supervisionado ou não supervisionado para gerar o ranqueamento. A criação de ranqueamento consiste em criar uma lista de objetos ranqueados à partir das características extraídas desses; enquanto a agregação de ranqueamento visa criar uma lista de objetos ranqueados utilizando múltiplas listas de ranqueamentos que, por sua vez, podem ser geradas por diferentes técnicas de ranqueamento de informações. As seções 4.1 e 4.2 abrangem um referencial teórico acerca da criação de ranqueamento e agregação, respectivamente.

4.1 Criação de Ranqueamento

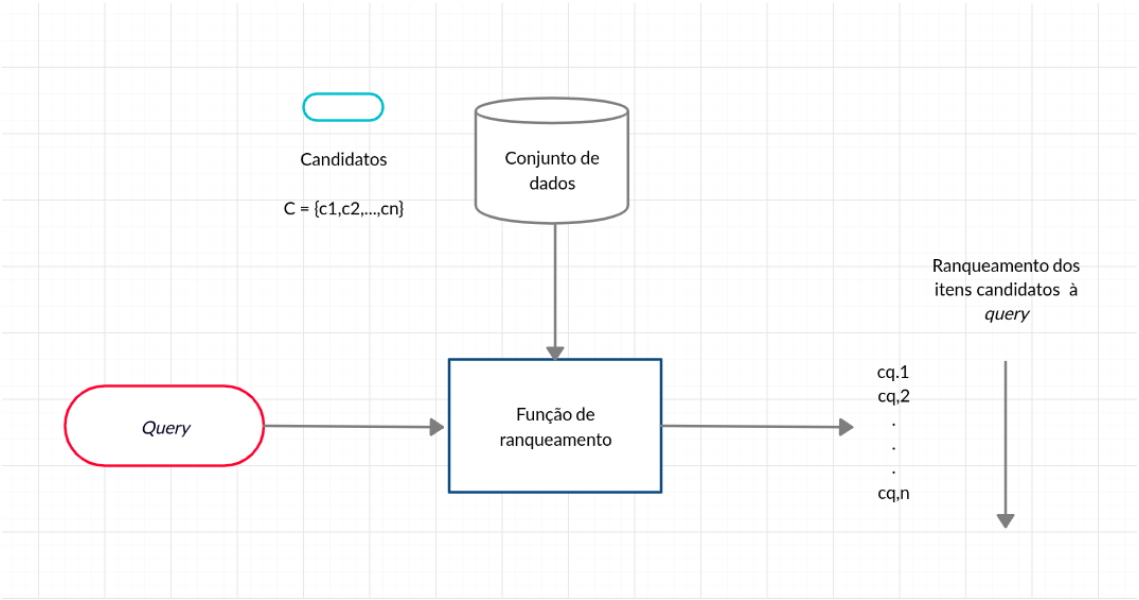
O problema de criar um ranqueamento pode ser compreendido da seguinte maneira: considere dois conjuntos que, para simplificação, à partir daqui são denominados conjunto de *queries* $Q = \{q_1, q_2, ..., q_m\}$ e conjunto de objetos candidatos $C = \{c_1, c_2, ..., c_n\}$, respectivamente. A título de exemplo, Q pode ser um conjunto de requisições, usuários, sentenças, entre outros. Enquanto C pode ser um conjunto de documentos ou imagens, produtos, sentenças alvo, etc. Note que os conjuntos Q e C não necessariamente precisam ter a mesma quantidade de itens e não possuem tamanho limitado.

Dado um elemento $q \in Q$, os elementos em C são ranqueados com base nas informações de q e C. Assim, a criação do ranqueamento é comumente realizada a partir da seguinte função de ranqueamento ou *scoring*: $f(q, c)$: $Q\ x\ C \rightarrow \Re$, onde q é um elemento de Q e c é um elemento de C. Essa função, por sua vez, atribui um *score*, denominado s_c, a cada elemento $c \in C$ e, por fim, os elementos $\in C$ são então ordenados a partir dos *scores*. Isso significa que o ranqueamento π é realizado ao ordenar a função f.

Assim, o modelo local $f(q, c)$ é utilizado para criação do ranqueamento gerando uma lista de *scores* dos objetos para dada *query* q. Esses pontos são atribuídos com base na relevância entre *query* e objeto, que, por sua vez, é definida a partir de características extraídas dos elementos dos conjuntos Q e C (LI, 2014).

A Figura 5 ilustra a criação de ranqueamento para certa *query* q e n itens candidatos.

Figura 5 – Criação de ranqueamento.



Fonte: Elaborado pela autora.

4.2 Agregação de Ranqueamento

Ainda considerando os conjuntos Q e C, apresentados em 4.1, o problema de agregação de ranqueamento pode ser compreendido como segue: para um elemento de consulta $q \in Q$ e o conjunto de itens candidatos C, existem k possíveis listas de ranqueamento π. Assim, tem-se $\{\sum = \pi_i | \pi \in A, i = 1, ..., k\}$, onde A representa o conjunto de todas as listas ranqueadas em C. Logo, a agregação de ranqueamento consiste em considerar uma *query* q e as listas de ranqueamentos $\sum$ dos objetos $\in C$ como entrada para o processo de agregação, de modo que uma nova lista de ranqueamento π é gerada como saída à partir da ordenação dos *scores* gerados pela função $F(q, \sum) : Q\ x\ A^k \rightarrow \Re^n$.

Dessa forma, o processo de agregação de ranqueamento combina múltiplas listas de ranqueamento em um única lista, que, por sua vez, é melhor do que qualquer uma das listas de ranqueamento fornecidas como entrada. Diferentemente do que ocorre na criação de ranqueamento, na qual o *ranking* é gerado com base nas características da *query* q e de cada candidato $c \in C$, na agregação o ranqueamento final é gerado com base em diferentes listas de ranqueamentos dos itens candidatos. Note então que a saída da criação de ranqueamento pode ser utilizada como entrada para a agregação (LI, 2014).

Nesse trabalho, o ranqueamento de informações é abordado sob à ótica da criação de ranqueamento. Dessa forma, as seções e capítulos a seguir abordam a tarefa de aprender a ranquear nesse contexto.

4.3 Tarefa de Aprender a Ranquear

Conforme já mencionado, a tarefa de aprender a ranquear, particularmente aprender por criação de ranqueamento, tem sido bastante estudada nos últimos anos (PAREEK; RAVIKUMAR, 2014; WANG et al., 2016; LI; SONG; LUO, 2017). Para melhor compreender o problema de ranqueamento, pode-se considerar algumas similaridades entre esse e outras tarefas de aprendizado de máquina, tais como classificação, regressão e classificação ordinal.

Na tarefa de classificação, tem-se um vetor de características $x \in \Re^d$ como entrada e um rótulo $y \in Y$ como saída, representando a classe do objeto, onde Y representa o conjunto de rótulos de classes. Tem-se então como objetivo aprender um classificador $f(x)$ capaz de determinar o rótulo (classe) y de um dado vetor de características x.

Em regressão, a entrada também é um vetor de características $x \in \Re^d$, mas a saída é um número real $y \in \Re$. O objetivo nesse caso é aprender uma função $f(x)$ que possa associar um número real y a um determinado vetor de características x.

A classificação ordinal (ou regressão ordinal), por sua vez, é a tarefa de aprendizado de máquina que mais se aproxima da de *learning to rank*, mas ainda assim possui diferenças. A entrada nesse caso continua sendo um vetor de características $x \in \Re^d$ e a saída é um rótulo $y \in Y$, representando um grau, onde Y é um conjunto de rótulos de graus. O objetivo aqui consiste em aprender uma função $f(x)$ que possa determinar um grau y para um dado vetor x. Para isso, primeiro o modelo calcula um *score* usando $f(x)$ e então decide qual rótulo de grau (y) deve ser atribuído a x, considerando um número de *thresholds*. Logo, intervalos são definidos e o grau de uma entrada é determinado de acordo com o intervalo cujo qual seu *score* pertence. Dessa forma, na classificação ordinal a ordenação em categorias (graus) é suficiente, enquanto no ranqueamento a ordenação pontual dos objetos é essencial (LI, 2014).

4.3.1 Etapas de Treinamento e Teste

A criação de ranqueamento é composta por uma etapa de treinamento e outra de teste. Dado um conjunto T de dados, suponha que esse é dividido em dois subconjuntos Q_t e C_t cujos dados são denominados *queries* e candidatos respectivamente. Para compor a entrada do modelo de treinamento, cada uma das *queries* $\in Q_t$ é associada a cada um dos itens candidatos $\in C_t$ e a relevância entre eles é informada, juntamente das características definidas para representar a similaridade entre ambos.

Em problemas de ranqueamento, a informação de relevância entre *query* e candidato pode ser representada de diferentes maneiras. No entanto, o modo mais comumente utilizado para tal consiste na representação dessa por meio de um rótulo, sendo que esse pode assumir um valor de grau binário, 0 (relevante) ou 1 (não relevante) por exemplo, utilizando o conceito de relevância absoluta; ou assumir diferentes graus e, consequentemente, níveis de relevância; sendo que, no último caso, quanto maior o grau de relevância entre um candidato e dada *query*,

mais relevante ele é para essa.

Dado um conjunto V disjunto do conjunto T, suponha que esse também é dividido em dois subconjuntos Q_v e C_v compostos por *queries* e itens candidatos respectivamente. Na etapa de teste, o conjunto utilizado como *input* para o modelo já treinado é composto pelas características que representam a similaridade entre cada item dos conjuntos Q_v e C_v, sendo que as características de similaridade informadas no conjunto de treinamento e no de testes devem ser computadas da mesma maneira. Por fim, objetiva-se ter como saída na fase de teste a relevância entre os itens $\in Q_v$ e $\in C_v$. As subseções 4.3.1.1 e 4.3.1.2 abordam mais profundamente as fases de treinamento e testes aqui introduzidas.

4.3.1.1 Fase de Treinamento

Seja o conjunto de dados T composto pelos subconjuntos Q_t e C_t, considere $Y_t = \{1, 2..., l\}$ o conjunto de possíveis rótulos para dado problema de ranqueamento, onde esses representam o grau de relevância entre os itens $\in Q_t$ e os $\in C_t$. Considere ainda que $\{qt_1, qt_2, ..., qt_m\}$ é o conjunto de *queries* utilizadas no treinamento e qt_i é a $i-ésima$ *query* desse. Seja $C_{ti} = \{ct_{i,1}, ct_{i,2}, ..., ct_{i,n_i}\}$ o conjunto de candidatos associados à *query* qt_i. E $Y_{ti} = \{yt_{i,1}, yt_{i,2}, ..., yt_{i,n_i}\}$ o conjunto que contém as relevâncias entre qt_i e os itens $\in C_{ti}$, onde $_{n_i}$ corresponde aos tamanhos de C_{ti} e Y_{ti}. $ct_{i,j}$ denota o $j-ésimo$ elemento em C_{ti} e $yt_{i,j}$ corresponde ao $j-ésimo$ rótulo de grau $\in Y_{ti}$, que, por sua vez, representa o grau de relevância de $ct_{i,j}$ em relação à *query* qt_i.

Assim, considere $S = \{(qt_i, C_{ti}), Y_{ti}\}_{i=1}^{m}$, lembrando que m equivale ao número de *queries* utilizadas na fase de treino. Seja um vetor de características $xt_{i,j} = \phi\ (qt_i, ct_{i,j})$ obtido a partir de cada par *query*-candidato $(qt_i, ct_{i,j})$, onde i = 1,2,...,m; j = 1,2,...,n_i e ϕ a função utilizada para extração das características que representam a similaridade entre os elementos do par. E seja também $X_{ti} = \{xt_{i,1}, xt_{i,2}, ..., xt_{i,n_i}\}$, o conjunto de treinamento pode ser representado como $S' = \{(X_{ti}, Y_{ti})\}_{i=1}^{m}$.

Dessa forma, um método desenvolvido para aprender a ranquear objetiva treinar um modelo local $f(qt, ct) = f(xt)$ que seja capaz de atribuir um *score* a um dado par de *query* e candidato (qt, ct), isto é, atribuir um *score* ao respectivo vetor de características x_t. Generalizando, um modelo global $F(qt, Ct) = F(X_t)$ pode ser considerado, sendo que o modelo local de ranqueamento fornece como saída um único *score*, enquanto o modelo global retorna um conjunto de *scores*.

Sejam os itens em C_{ti} identificados pelos inteiros 1,2,...n_i, a permutação π_i em C_{ti} é definida como a bijeção de 1,2,...n_i para ela mesma. $\prod_i$ então denota o conjunto de todas as possíveis permutações em C_{ti}, sendo $\pi_i(j)$ a posição no *ranking* do $j-ésimo$ item candidato, isto é, $c_{ti,j}$ na permutação π_i. Para definir o ranqueamento final, seleciona-se uma permutação $\pi_i \in \prod_i$ para uma dada *query* e o conjunto de candidatos C_{ti} associados a ela, utilizando os *scores* computados pelo modelo de ranqueamento $F(q_{ti}, C_{ti})$ (ou $f(q_i, c_{ti})$) (LI, 2014).

4.3.1.2 Fase de Teste

Seja o conjunto de dados $V \cap T = \varnothing$ composto pelos subconjuntos Q_v e C_v que contêm *queries* (q_v) e itens candidatos (c_v) respectivamente. O conjunto de testes fornecido como *input* ao modelo de ranqueamento computado na fase de treinamento é definido como $U = \{(X_{vi})\}$, sendo $X_{vi} = \{xv_{i,1}, xv_{i,2}, ..., xv_{i,n_i}\}$ com n igual ao número de itens $\in C_v$. Vale lembrar que assim como $xt_{i,j}$ apresentado na seção 4.3.1.1, $xv_{i,j}$, com $^{n}_{j=1}$, é obtido de forma análoga àquele. Na fase de testes, esse modelo é utilizado para que um *score* seja atribuído a cada candidato $c_{vi} \in C_v$. Por fim, ao ordenar os itens candidatos decrescentemente por seus *scores* obtém-se como saída a lista de ranqueamento desses para q_{vi} (LI, 2014).

Note que a composição dos dados de treinamento e teste em um problema de ranqueamento são semelhantes, mas essa difere da utilizada no aprendizado supervisionado convencional para tarefas como as de classificação e regressão, por exemplo. Em *learning to rank* a *query* e os candidatos a ela associados compõem um grupo de dados. Esse, por sua vez, é utilizado no treinamento e teste, e não as instâncias que os compõem. Dessa forma, pode-se dizer que um modelo local de ranqueamento é uma função de *query* e candidato, ou seja, uma função que tem como entrada um vetor de características derivado de uma *query* e um item candidato associado à essa.

4.4 Abordagens das Técnicas para Aprender a Ranquear

Conhecendo as semelhanças e diferenças entre as tarefas de classificação, regressão, classificação ordinal e ranqueamento, podemos agora nos atentar aos métodos propostos para criação do ranqueamento. Esses são comumente divididos nas abordagens *pointwise*, *pairwise* e *listwise*.

As duas primeiras abordagens transformam o problema de ranqueamento em um problema de classificação, regressão ou classificação ordinal. Enquanto a última utiliza listas de ranqueamento como instâncias em aprendizado e aprende o modelo de ranqueamento com base nessas. Assim, a principal diferença entre essas abordagens está nas funções de perda empregadas nos modelos que geram o ranqueamento (LI, 2014). Essas abordagens são melhor descritas nas subseções a seguir.

4.4.1 Pointwise

Na abordagem *pointwise* a criação do ranqueamento é transformada em uma tarefa de classificação, regressão ou classificação ordinal e métodos já existentes destinados à resolução dessas tarefas são utilizados para tal. Nessa abordagem, tem-se como entrada para o treinamento um conjunto I composto por vetores de características x e rótulos y. I é formado pela união de todos os grupos (q_i, C_i), assim $I = \{(x_{i,1}, y_{i,1}), ..., (x_{i,ni}, y_{i,ni})\}$, sendo i = 1, ..., m.

Dessa forma, mapeando x para y, os pares (x,y) passam a ser instâncias de treinamento semelhantes às utilizadas no aprendizado supervisionado convencional, em que, quando y é considerado um rótulo de classe tem-se um problema de classificação; se y é considerado um número real, o problema passa ser considerado de regressão; enquanto se y é um rótulo de grau, o problema pode ser resolvido como classificação ordinal. Dessa forma, métodos existentes para classificação, regressão e classificação ordinal podem ser aplicados à resolução do problema de ranqueamento.

Suponha que o modelo aprendido $f(x)$ forneça números reais como saída. Assim, dada uma *query*, esse pode ser utilizado para ranquear itens, ordenando-os de acordo com os *scores* atribuídos a esses. Nessa abordagem, a função de aprendizado é chamada *pointwise* porque é definida em um único vetor de características (LI, 2014).

4.4.2 Pairwise

Na abordagem *pairwise*, o ranqueamento é transformado em uma classificação par a par ou regressão par a par. Para tal, um modelo que defina a ordem de relevância entre pares de itens é utilizado para definir a relevância de cada item, separadamente, e, consequentemente, sua posição no *ranking*. A partir dos dados rotulados referentes à *query* q_i, $\{(x_{i,1}, y_{i,1}), ..., (x_{i,ni}, y_{i,ni})\}$ com i = 1,..., m, a abordagem cria pares de preferências de vetores de características. Por exemplo, se $x_{i,j}$ tem um grau mais alto do que $x_{i,k}$, ou seja, se $y_i, j > y_{i,k}$, então $x_{i,j} \succ x_{i,k}$ compõem um par de preferência, onde $\succ$ denota preferência sobre. Logo, $x_{i,j}$ deve estar posicionado acima de $x_{i,k}$ no ranqueamento.

Os pares de preferências podem então ser considerados como instâncias e rótulos em um novo problema de classificação. Por exemplo, $x_{i,j} \succ x_{i,k}$ é considerada uma instância com rótulo positivo. Assim, métodos de classificação existentes podem ser utilizados para treinar um modelo $f(x)$ e esse pode ser usado para gerar o ranqueamento. Mais precisamente, itens são associados à *scores* por $f(x)$ e ordenados de forma decrescente de acordo com os valores desses. A função de perda utilizada é denominada *pairwise* porque essa é definida em um par de vetores de características (LI, 2014).

Em suma, quando o problema de ranqueamento é mapeado como um problema de classificação, essa metodologia considera pares de objetos como instâncias de aprendizagem e modela o problema de forma que os objetos possam ser classificados em duas categorias: corretamente classificado (ou relevante) e incorretamente classificado (ou irrelevante) (CAO et al., 2007). Durante o aprendizado, pares de itens são considerados e associa-se a cada par um rótulo que representa a relevância relativa entre esses dois objetos. O modelo de classificação é então treinado com os dados rotulados e é utilizado no ranqueamento dos itens.

4.4.3 Listwise

Na abordagem *listwise*, listas de ranqueamento são utilizadas na fase de treinamento e teste do modelo, de modo que os dados rotulados $(x_{i,1}, y_{i,1}), ..., (x_{i,ni}, y_{i,ni})$ associados a uma determinada *query* q_i são considerados uma única instância. A partir dos dados de treinamento, a metodologia aprende um modelo $f(x)$ capaz de atribuir *scores* aos itens representados por vetores de características e a ranqueá-los utilizando esses *scores* de forma que os itens com pontuações maiores ocupem posições mais altas no ranqueamento e vice-versa.

Nessa abordagem, a forma como os dados são utilizados nas etapas de aprendizagem e predição diferem da utilizada no aprendizado supervisionado convencional. Por isso, técnicas convencionais como as utilizadas para classificação ou regressão não podem ser diretamente aplicadas nessa (LI, 2014).

4.5 Técnicas para Ranqueamento de Informações

Nas subseções 4.5.1 e 4.5.2 são apresentados respectivamente um referencial teórico acerca do classificador SVM, do inglês *Support Vector Machine* (SVM) e do método de ranqueamento, do tipo *pairwise*, denominado *Ranking SVM*. Optou-se por abordar essa técnica de ranqueamento no presente trabalho pelo fato desse ser comumente utilizado em tarefas de ranqueamento e apresentar alta acurácia (LI, 2011b); além do fato de classificadores baseados em OPF apresentarem resultados similares aos obtidos pelo classificador SVM em tarefas de classificação. O desempenho de ambos os algoritmos baseados em Floresta de Caminhos Ótimos propostos no capítulo 5 são comparados ao alcançado pelo *Ranking SVM* quando aplicados à tarefa de ranqueamento. Os resultados experimentais obtidos são descritos no capítulo 6.

4.5.1 Máquinas de Vetores de Suporte

A Máquina de Vetores de Suporte, SVM, é uma técnica de aprendizado de máquina amplamente utilizada e que possui diferentes vertentes, por isso, a partir daqui elas serão referenciadas como Máquinas de Vetores de Suporte, no plural. Essas são fundamentadas à partir de teorias estatísticas e, inicialmente, foram propostas para resolver problemas de classificação, depois foram estendidas para também solucionar problemas de regressão e por fim para aprender a ranquear; essas são comumente chamadas, respectivamente, de *Classifying SVM*, *Support Vector Regression (SVR)* e *Ranking SVM* (YU; KIM., 2012).

O modelo inicial de SVM é um classificador binário no qual a saída da função de aprendizado é positiva ou negativa. Mas, com algumas modificações nesse, as SVMs também podem ser utilizadas na resolução de problemas multiclasses (combinando várias SVMs de classificação binária), de regressão e clusterização (aprendizado não supervisionado). Quando aplicadas à tarefas de reconhecimento de padrões, elas têm apresentado resultados superiores

aos alcançados por métodos similares em diversas aplicações. Neste capítulo, somente a proposta original de SVM será abordada, uma vez que ela apresenta a ideia geral do método e é base para as outras variações de SVM (LORENA; CARVALHO, 2003).

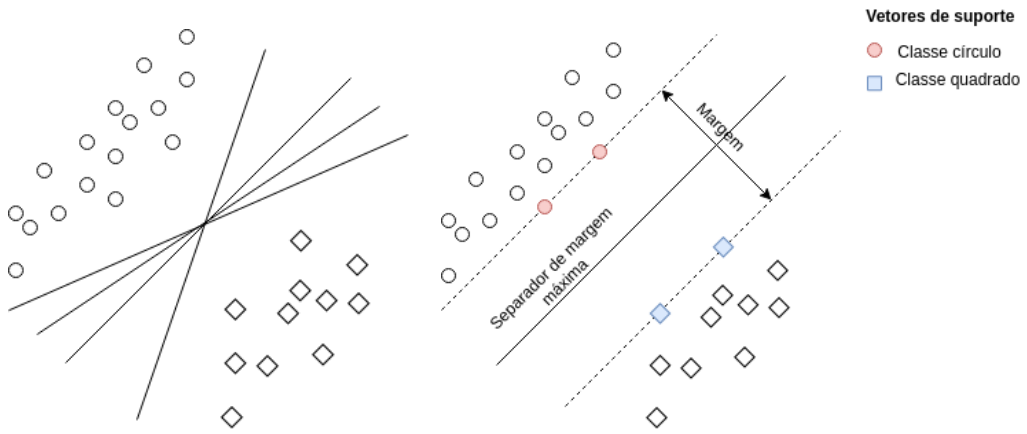
Dentre as características das SVMs, podem-se citar como principais, segundo (LORENA; CARVALHO, 2003):

- boa capacidade de generalização, sendo que essa capacidade é medida pela eficiência do classificador na classificação de dados que não pertençam ao conjunto utilizado na fase de treinamento;
- robustez diante de objetos de grandes dimensões, tais como imagens, sem gerar *overfitting* (super especialização) no classificador e
- convexidade da função objetivo: a utilização das SVMs implica na otimização de uma função quadrática, que apresenta somente um mínimo global.

As SVMs têm a capacidade de incorporar os dados em um espaço de dimensão maior, utilizando um truque de *kernel* que é descrito a seguir. Essa propriedade das SVMs faz com que os dados que, muitas vezes não são separáveis linearmente no espaço de entrada original, tornem-se linearmente separáveis em um espaço de maior dimensão. Pode-se dizer que as SVMs são bem-sucedidas em parte, também, devido à ideia básica de que alguns exemplos no conjunto de treinamento são mais importantes que outros (RUSSEL; NORVIG., 2013).

Na Figura 6a, é apresentado um problema de classificação binária (classe círculo x classe quadrado) com quatro possíveis limiares de decisão, sendo cada um deles um separador linear. Quando amostras de diferentes classes estão localizadas muito próximas da linha de separação entre classes, essas podem facilmente ser classificadas incorretamente; para tratar essa questão, as SVMs buscam minimizar a perda de generalização esperada ao invés da perda empírica sobre os dados de treinamento por meio da construção de um separador de margem máxima, conforme é possível observar em 6b. Note, também em 6b, que o separador de margem máxima localiza-se no ponto médio da margem (área entre as linhas tracejadas) e que os vetores de suporte (pontos destacados nas cores vermelho e azul) são os exemplos de cada classe mais próximos do separador.

Figura 6 – Máquina de Vetores de Suporte e problema de classificação binária.



Fonte: Elaborado pela autora.

Não se sabe onde os exemplos do conjunto de teste ficarão localizados no espaço de características, entretanto, sob o pressuposto probabilístico de que eles são obtidos da mesma distribuição que os utilizados no conjunto de treinamento, alguns argumentos da teoria da aprendizagem computacional sugerem que pode-se minimizar a perda de generalização escolhendo o separador mais distante dos exemplos já conhecidos, ou seja, optando pelo separador de margem máxima (RUSSEL; NORVIG., 2013). A margem é definida pela distância do separador até o ponto de exemplo mais próximo vezes dois. A seguir, são apresentadas as equações que permitem que esse separador seja encontrado. No entanto, algumas considerações devem ser feitas previamente:

- as SVMs utilizam a convenção de que os possíveis rótulos para uma classe são $+1$ e -1;
- a linha de corte é obtida através de um parâmetro b e
- o separador é definido como o conjunto de pontos $x : w.x + b = 0$.

Para encontrar os parâmetros w e b que maximizam a margem, (RUSSEL; NORVIG., 2013) sugerem utilizar uma representação dual na qual a solução ótima é encontrada resolvendo

$$\underset{a}{\operatorname{argmax}} \sum_j a_j - 1/2 \sum_{j,k} a_j a_k y_j y_k (x_j.x_k) \ , \tag{7}$$

sujeito às restrições $a_j \geq 0$ e $\sum_j a_j y_j = 0$.

Uma vez encontrado o vetor x, pode-se voltar a w com a equação $w = \sum_j a_j x_j$. A equação do separador é descrita a seguir:

$$h(x) = sinal(\sum_j a_j y_j (x.x_j) - b). \tag{8}$$

Vale ressaltar que os pesos j associados a cada ponto de dados possuem valor zero, exceto pelos vetores de suporte (os pontos mais próximos do separador). A título de curiosidade, esses vetores são chamados de suporte porque "sustentam" o plano de separação.

O problema de classificação apresentado na Figura 6 é linearmente separável. No entanto, conforme já dito, as SVMs também são capazes de resolver problemas linearmente não-separáveis. Para tal, essas utilizam o chamado truque de *kernel* que consiste em mapear cada um dos vetores de características (que representam os dados em questão) para um novo espaço de características, de dimensão maior que o original, cujo qual possibilite que os dados sejam separados por um separador linear. Para realizar o mapeamento dos vetores de características para o novo espaço utiliza-se uma função denominada *kernel*, sendo que diferentes funções podem ser utilizadas de acordo com as amostras do problema.

4.5.2 Ranking SVM

O método *Ranking SVM* é uma variante do algoritmo Máquina de Vetores de Suporte que é utilizado na resolução de problemas de ranqueamento e foi proposto inicialmente com o intuito de melhorar um mecanismo de busca na web em (JOACHIMS, 2002). A técnica objetiva aprender uma função de ranqueamento (ou preferências) e é comumente utilizada em aplicações relacionadas à recuperação e ranqueamento de informações.

As diferenças entre a tarefa de aprender uma função de ranqueamento e uma função de classificação (como a aprendida no modelo inicial de SVM), segundo (YU; KIM., 2012), são:

a) Classificação

- O conjunto de treinamento é composto por dados de objetos e suas respectivas classes;
- A função de classificação fornece como saída uma classe para um dado objeto.

b) Ranqueamento

- O conjunto de treinamento é uma ordenação de dados. Se "A é mais relevante do que B", é especificado como "$A > B$". Um conjunto de treinamento é então definido como $R = (x_1, y_{1)}, ..., (x_m, y_m)$, onde y_i é a posição de x_i no *ranking*, tal que, $y_i < y_j$ se $x_i > x_j$.
- A função de ranqueamento fornece como saída uma pontuação para cada objeto, a partir do qual uma ordenação global de dados é construída, de modo que a função $F(x_i)$ fornece uma saída tal que $F(x_i) > F(x_j)$ para qualquer $x_i > x_j$.

Generalizando, suponha que $R*$ equivale ao ranqueamento ótimo dos dados. Uma função de ranqueamento F é comumente avaliada por quão próximo sua ordenação R^F se

aproxima de $R*$. Sendo assim, usando uma SVM, uma função global de ranqueamento F pode ser aprendida a partir de uma ordenação R. Suponha, neste momento, que F é uma função linear de ranqueamento tal como:

$$\forall(x_i, x_j) : y_i < y_j \in R : F(x_i) > F(x_j) \iff w.x_i > w.x_j. \tag{9}$$

O vetor de peso w é ajustado por um algoritmo de aprendizado e considera-se que uma ordenação R é linearmente ranqueável se existe uma função F (representada por um vetor de pesos w) que satisfaça a Equação 9 $\forall\ (x_i, x_j) : y_i < y_j \in R$. Objetiva-se então aprender F de modo a se aproximar de R e generalizar tão bem quanto esse, isto é, encontrar o vetor de peso w tal que $wx_i > wx_j$ para a maioria de pares de dados $(x_i, x_j) : y_i < y_j \in R$. A solução para esse problema pode ser obtida ao utilizar técnicas de SVM introduzindo variáveis de folga ξ_{ij} e minimizando o limite superior $\sum \xi_{ij}$ tal como descrito em (YU; KIM., 2012):

$$minimizar : L_1(w, \xi_{ij}) = 1/2w.w + C \sum \xi_{ij}. \tag{10}$$

$$sujeito\ a : \forall(x_i, x_j) : y_i < y_j \in R : w.x_i \geq w.x_j + 1 - \xi_{ij}\ e. \tag{11}$$

$$\forall(i, j) : \xi ij \geq 0. \tag{12}$$

Ao minimizar o limite superior na Equação 10 e dada a restrição da Equação 11 tem-se que o problema de otimização satisfaz a ordenação do conjunto de treinamento R com um erro mínimo. Cabe citar que o parâmetro C refere-se ao parâmetro que controla a relação custo-benefício entre o tamanho da margem e o erro de treinamento.

Ao reorganizar a Equação 11 tal conforme abaixo:

$$w(x_i - x_j) \geq 1 - \xi ij\ , \tag{13}$$

o problema de otimização passa a ser equivalente ao de classificação com SVMs, baseado na diferença entre pares de vetores $(x_i - x_j)$. Dessa forma, o algoritmo SVM pode ser estendido para resolver o problema de ranqueamento.

Observe que os vetores de suporte nesse caso são os pares de dados (x_i^s, x_j^s) tal que a restrição dada pela Equação 13 é satisfeita a partir da igualdade de sinal: $w(x_i^s - x_j^s) = 1 - \xi_{ij}$.

Assim como é feito na classificação com SVM, uma função F também é modelada pelos vetores de suporte para o método *Ranking SVM*. Analogamente ao problema de classificação, o

problema primordial do ranqueamento com SVM pode ser transformado no seguinte problema dual usando os multiplicadores de Lagrange:

$$minimizar : L_2(\alpha) = \sum_{i,j} a_{ij} - \sum_{i,j} \sum_{u,v} a_{ij} a_{uv} K(x_i - x_j, x_u - x_v), \tag{14}$$

$$sujeito\ a : C \geq \alpha \geq 0. \tag{15}$$

Uma vez transformado no problema dual, o *truque do kernel* utilizado na SVM original pode ser aplicado para suportar funções não-lineares de ranqueamento, sendo $K(\cdot)$ uma função de *kernel* e α_{ij} um coeficiente para a diferença entre pares de vetores $(x_i - x_j)$. Note que a função *kernel* é calculada $P^2(\ m^4)$ vezes, onde P é o número de pares de dados e m é o número de pontos no conjunto de treinamento. Logo, resolver um problema de ranqueamento com SVM tem custo mínimo $O(m^4)$. Outros algoritmos de treinamento mais rápidos para ranqueamento com SVM foram propostos na literatura, entretanto esses são limitados a funções lineares (YU; KIM., 2012).

Calculado o valor de α, w pode ser escrito conforme a diferença entre pares de vetores e seus coeficientes, conforme Equação 16:

$$w = \sum_{i,j} \alpha_{ij}(x_i - x_j). \tag{16}$$

Por fim, a função de ranqueamento F para um novo vetor z pode ser calculada da seguinte forma:

$$F(z) = wz = \sum_{i,j} \alpha_{ij}(x_i - x_j)z = \sum_{i,j} \alpha_{ij} K(x_i - x_j, z). \tag{17}$$

4.6 Métricas de Avaliação

A avaliação do desempenho de um modelo de ranqueamento é realizada ao comparar o ranqueamento gerado pelo modelo e o ranqueamento definido como ideal (*ground truth*). Existem diferentes métricas para validar o ranqueamento gerado, mas dentre elas três são comumente utilizadas em diferentes aplicações: o Ganho Cumulativo Descontado Normalizado (NDCG), do inglês *Normalized Discounted Cumulative Gain*, o Ganho Cumulativo Descontado (DCG), do inglês *Discounted Cumulative Gain* e a Média da Precisão Média (MAP), do inglês *Mean Average Precision*.

A formulação matemática e descrição de uso de cada uma dessas medidas (GANJI-SAFFAR, 2011) serão apresentadas nas subseções a seguir. Entretanto, antes de introduzir as métricas, cabe relembrar alguns conceitos e notações:

- Os dados utilizados na fase de treinamento e teste de um modelo de ranqueamento são compostos por *queries*, itens candidatos à ela e rótulos de grau indicando a relevância entre dada *query* e determinado candidato;
- Q representa o conjunto de *queries*, C o conjunto de itens candidatos e Y o conjunto de rótulos;
- Para cada query $q_i \in Q$ tem-se uma lista de m elementos candidatos $C_i = c_{i,1}, c_{i,2}, ..., c_{i,m}$;
- Um método de *learning to rank* tem como objetivo produzir um ranqueamento para os objetos C_i, especificamente uma permutação π_i dos índices dos objetos $\{1, 2, ..., m\}$. Assim, busca-se aprender uma função $f(q, C)$ que produza uma permutação ótima π_i.

Desse modo, assuma que existem n *queries* e que, para cada *query* q_i, tem-se um conjunto de m objetos candidatos $C_i = \{c_{i,1}, ..., c_{i,m}\}$ com rótulos de grau de relevância associados à *query* q_i e representados por $y_{i,1}, ...y_{i,m}$ respectivamente. Considere ainda que o modelo produzirá um ranqueamento π_i de forma que $\pi_{i,j}$ representa o ranqueamento predito para o item $c_{i,j}$. Dado o ranqueamento gerado, as métricas descritas a seguir podem ser utilizadas para validar a performance do modelo.

4.6.1 Discounted Cumulative Gain

Antes de abordar o DCG, é necessário descrever o ganho cumulativo (GC) que, por sua vez, se dá pela soma dos *scores* de relevância dos objetos candidatos em análise. Assim, suponha que se tem:

$\pi_i = \{c_{i,1}, c_{i,2}, c_{i,3}, c_{i,4}\}$ e $y_i = \{5, 2, 5, 0\}$

O objeto no topo do ranqueamento possui um *score* de valor 5, o segundo tem uma relevância de valor 2 e assim por diante. Logo, o ganho cumulativo nesse exemplo é calculado como se segue:

$\sum_i y_i = 5 + 2 + 5 + 0 = 12.$

Observe que essa medida não considera a ordem do ranqueamento, somente os valores dos *scores*. Nesse caso, se a ordem do ranqueamento fosse diferente, ainda assim teríamos o mesmo valor de GC:

- $\pi_i = \{o_{i,4}, o_{i,3}, o_{i,2}, o_{i,1}\}$
- $y_i = \{5, 2, 5, 0\}$
- $\sum_i y_i = 0 + 5 + 2 + 5 = 12$

O ganho cumulativo descontado, por sua vez, leva em consideração a ordem dos itens no ranqueamento. Essa métrica atribui uma recompensa maior aos objetos que possuem alta relevância e aparecem em altos níveis no ranqueamento. Já quando um objeto de alta relevância aparece em uma posição baixa do ranqueamento esse recebe uma recompensa menor. Essa medida é computada como segue:

$$DCG_r = \sum_{j=1}^{r} \frac{2^{y_{i,\pi_{i,j}}} - 1}{log(i+1)}, \tag{18}$$

onde r corresponde ao número de objetos associados à *query* em análise.

No exemplo dado, tem-se o DCG_4 = 31 + 1.9 + 15.5 + 0 = 48.4. Note que na primeira posição do ranqueamento a pontuação de alta relevância 5 vale 31, enquanto na terceira posição vale 15,5. Já a menor pontuação de relevância 2 vale apenas 1,9 na segunda posição do ranqueamento. Como nessa métrica a ordem dos objetos no ranqueamento importa, o valor do DCG aumentaria caso a ordem dos itens $o_{i,2}$ e $o_{i,3}$ fosse trocada.

4.6.2 Normalized Discounted Cumulative Gain

O ganho cumulativo descontado normalizado é a versão normalizada do DCG. Essa normalização é feita encontrando o DCG_r de um ranqueamento ideal. Para o exemplo dado, um possível ranqueamento ideal seria:

- $\pi_i = \{d_{i,1}, d_{i,3}, d_{i,2}, d_{i,4}\}$ resultando no seguinte $DCG_{r_{ideal}}$:

  $DCG_{r_{ideal}} = \sum_{j=1}^{r} \frac{2^{y_{i,\pi_{i,j}}-1}}{log(i+1)} = 31 + 19.5 + 1.5 + 0 = 52$

  O $NDCG$, por sua vez, pode ser calculado da seguinte forma:

$$NDCG_r = \frac{DCG_r}{DCG_{r_{ideal}}}. \tag{19}$$

No exemplo dado, tem-se que o $NDCG = \frac{48.4}{52} = 0.93$. Note que, se o ranqueamento π_i gerado pelo modelo fosse igual ao ranqueamento ideal, o $NDCG_r$ teria valor 1.

4.6.3 Mean Average Precision

O MAP é comumente utilizado na avaliação de métodos de *learning to rank* aplicados a problemas que utilizam *scores* binários de relevância, isto é, correto e incorreto ou relevante e

não relevante, por exemplo. Essa métrica utiliza a precisão em r $(P@r)$ que é definida ao contar o número de objetos relevantes nas r primeiras posições dividido por r, conforme Equação 20:

$$P@r = \frac{1}{r}\sum_{i=1}^{r} I(relevância_i == 1) \tag{20}$$

O MAP consiste na média dos valores de precisão média, do inglês *Average Precision* (AP), de cada *query*. A precisão média, por sua vez, é calculada conforme abaixo:

$$AP = \frac{\sum_{r=1}^{n}(P@r \times I(relevância_r == 1))}{\sum_{r=1}^{n} I(relevância_r == 1)} \tag{21}$$

Note que a AP é computada para uma única *query*, enquanto o MAP consiste no valor médio de AP, isto é, considera todas as *queries*. Dessa forma, o MAP pode ser computado conforme a seguir:

$$MAP = \frac{1}{n}\sum_{q=1}^{n} AP_q \tag{22}$$

, onde n equivale ao número de *queries*.

5 Ranqueamento de informações baseado em Floresta de Caminhos Ótimos

Conforme já mencionado, apesar de classificadores baseados em OPF terem sido utilizados em diferentes áreas e já aplicados à tarefa de recuperação de informações (TAVARES; FALCãO; MAGALHãES, 2011; SILVA et al., 2012; LI et al., 2016; DHAWALE; JOGLEKAR; KULKARNI, 2017), poucos desses trabalhos os aplicaram ao contexto de recuperação de imagens por conteúdo e, quando o fizeram, utilizaram a informação de distância entre as amostras para realizar a recuperação. Ademais, esses classificadores, até o presente momento, não foram aplicados à tarefa de aprender a ranquear.

Dessa forma, o presente trabalho tem como principal motivação aplicar os classificadores baseados em Floresta de Caminhos Ótimos, na versão supervisionada com grafo completo e k-*nn*, à tarefa de ranqueamento de informações, sendo que, nos experimentos realizados nesse as informações consideradas são imagens digitais. Para ranquear essas, características referentes ao conteúdo de cada uma delas, especificamente cor, forma e textura, são extraídas utilizando descritores de imagens globais.

Para aplicar as duas abordagens de classificadores baseados em OPF a esse cenário, a informação de custo das amostras é utilizada para criar o *ranking*. Em suma, utiliza-se a informação de custo atribuída a cada amostra durante a fase de treinamento para definir se essa é ou não uma amostra relevante à amostra de *query* em análise; isto é, o custo oferecido por uma amostra do conjunto de treinamento, candidata à protótipo, é utilizada para definir se essa conquistará a amostra de teste. As r primeiras amostras que oferecerem melhor custo para dada imagem do conjunto de teste são consideradas as r imagens mais relevantes para tal. Dessa forma, constrói-se o ranqueamento das r imagens mais relevantes para dada imagem de *query*. A Seção 5.1 apresenta os passos que descrevem as modificações realizadas na etapa de classificação (teste) do classificador baseado em OPF na versão grafo completo.

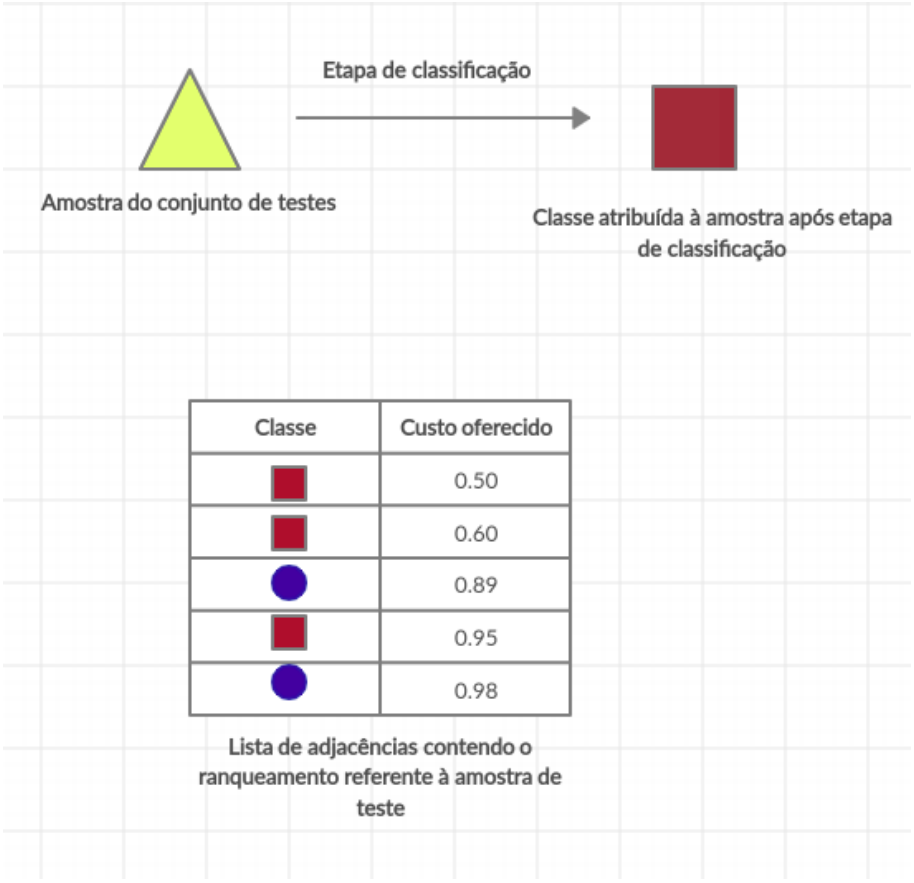
5.1 Abordagem com Grafo Completo

A seguir são descritas em quatro passos as modificações realizadas na fase de teste do classificador baseado em Floresta de Caminhos Ótimos, com grafo completo, a fim de que esse gere uma lista ranqueada com as r amostras do conjunto de treinamento que ofereceram os melhores custos à certa amostra de teste (*query*). Passos:

1. Declaração de variáveis adicionais:
    - Variável r_{costs}, do tipo inteiro, é inicializada com valor de r previamente definido;
    - Variável c do tipo *float* é declarada;
    - Variáveis r_{aux} e l_{aux}, do tipo inteiro, são declaradas;
    - Aloca-se um vetor do tipo inteiro e tamanho r_{costs}, definido como *$labels$;
    - Aloca-se um vetor do tipo *float* e tamanho r_{costs}, definido como *$costs$;
2. Início do laço de repetição 1 - Para cada uma das amostras do conjunto de testes:
    - Atribui-se a cada uma das posições de *$costs$ o valor máximo permitido para um número do tipo *float*;
3. Início do laço de repetição 2 - Busca-se dentre todas as amostras do conjunto de treinamento aquela que conquistará a amostra do conjunto de testes, oferecendo-lhe seu rótulo:
    - Armazena-se em *$costs$ os r melhores custos obtidos para dada amostra de teste;
    - Armazena-se em *$labels$ os rótulos das r amostras de treinamento que ofereceram os melhores custos em *$costs$;
    - Fim do laço 2;
    - Classifica-se a amostra de teste;
4. Inclusão de nós em lista de adjacências associada à amostra de teste:
    - Adiciona-se o custo e rótulo das amostras do conjunto de treinamento de r melhores custos a uma lista de adjacências associada à amostra de teste classificada;
    - Fim do laço 1.

Note que os custos em $costs$ são armazenados em ordem crescente, isto é, as amostras que ofereceram os menores custos à dada amostra de teste localizam-se nas primeiras posições da lista de adjacências associada a essa. Essa lista, por sua vez, representa o ranqueamento em r para a determinada amostra classificada na fase de teste do classificador. A Figura 7 ilustra uma amostra a ser classificada na etapa de teste e a lista de adjacências a ela associada contendo o custo e classe de cada uma das r amostras do conjunto de treinamento que lhe ofereceram os melhores custos, com $r = 5$.

Figura 7 – Amostra a ser classificada na fase de teste e lista de adjacências com classes e custos das r amostras ranqueadas.



Fonte: Elaborado pela autora.

5.2 Abordagem com Grafo k-*nn*

As modificações realizadas na fase de classificação do classificador baseado em Floresta de Caminhos Ótimos com grafo k-*nn* é análoga às incluídas na etapa de teste do classificador com grafo completo, conforme passos descritos na Seção 5.1, com cada posição de $*costs$ sendo inicializada com o mínimo valor permitido para um número do tipo *float*.

Logo, em suma, para aplicar os classificadores baseados em OPF com grafo completo e k-*nn* ao contexto de ranqueamento de informações, para cada amostra de teste a ser classificada armazena-se em uma lista de adjacências os r melhores custos oferecidos a ela juntamente com os rótulos das respectivas amostras que lhe ofereceram esses custos.

Vale mencionar que optou-se por utilizar a informação de custo para ranquear as r amostras mais relevantes porque espera-se que as amostras de treinamento que oferecerem os melhores custos para dada amostra de teste sejam aquelas com maior probabilidade de lhe oferecerem seu rótulo na etapa de classificação e, consequentemente, serem mais similares à essa.

6 Experimentos

Os algoritmos para ranqueamento de informações baseados em Floresta de Caminhos Ótimos, propostos no capítulo 5, foram utilizados em experimentos com as bases de imagens Original Brodatz (BRODATZ, 1966), Caltech101 (FEI-FEI; FERGUS; PERONA, 2004) e MPEG-7 (RALPH, 1999) e diferentes descritores de imagens globais. A base de imagens Brodatz é composta por imagens de texturas em escala de cinza, a Caltech101 por imagens naturais coloridas e a MPEG-7 por imagens binárias de formas. A Figura 8 apresenta uma amostra de cada uma dessas três bases de imagens, respectivamente. Escolheu-se esses *datasets* pelo fato desses serem frequentemente utilizados na literatura em tarefas de classificação de imagens, recuperação de imagens baseada em conteúdo e também para validar a eficácia dos descritores utilizados quanto à extração de características visuais, tais como textura, cor e forma.

Para comparar o desempenho dos classificadores baseados em OPF no cenário de *learning to rank* com o de outras abordagens, também foram realizados experimentos com o método *Ranking SVM*. Como a distância entre os vetores de características de certo par de imagens é frequentemente utilizada em aplicações baseadas em *information retrieval* quando nenhum modelo de ranqueamento é aplicado à recuperação dos itens, experimentos também foram realizados utilizando essa técnica.

Conforme já mencionado, no cenário de ranqueamento de informações, para validar a performance do modelo utilizado define-se como parâmetro um número r não nulo $\in$ Z, de modo que o modelo tem como objetivo recuperar os r itens mais relevantes para dada informação de *query*. Para os experimentos realizados neste trabalho definiu-se os valores de $r \in R = \{10, 15, 20\}$. As métricas $NDGC$ e MAP, apresentadas no capítulo 4, foram eleitas para validar o desempenho de cada uma das abordagens utilizadas. A seção 6.2 referencia os resultados obtidos nos experimentos.

Figura 8 – Amostras das bases de imagens utilizadas nos experimentos.

(a) (b) (c)

Fonte: (a) (BRODATZ, 1966); (b) (FEI-FEI; FERGUS; PERONA, 2004); (c) (RALPH, 1999).

6.1 Metodologia

Nesta seção é apresentada a metodologia utilizada nos experimentos realizados com as diferentes bases de imagens e técnicas de ranqueamento.

6.1.1 Conjuntos de Treinamento e Teste

Conforme explicado no capítulo 4, os conjuntos de treinamento e teste fornecidos aos modelos de ranqueamento são compostos por características referentes às informações de itens de *query* e itens candidatos. Assim, no cenário escolhido para realização dos experimentos, as imagens dos *datasets* são divididas entre imagens de *query* e imagens candidatas. Lembrando que o objetivo das técnicas de ranqueamento consiste em ranquear as imagens candidatas de modo que as r mais relevantes para dada imagem de *query* estejam localizadas nas primeiras posições do *ranking* gerado. As características utilizadas para descrever a similaridade entre *query* e candidatas referem-se ao conteúdo visual dessas, como mencionado anteriormente.

Para realização dos experimentos, o conjunto de dados referente a cada base de imagens utilizada foi dividido em três *folds*. No primeiro *fold*, 25% das imagens do *dataset* são utilizadas na etapa de treinamento e 75% são informadas na fase de teste; no segundo, 50% são consideradas para treinar o modelo e as outras 50% para validá-lo; por fim, no terceiro *fold*, 75% das imagens são para treino e as demais são utilizadas para teste.

Como se sabe, os conjuntos de dados fornecidos como entrada para as fases de treinamento e teste de métodos de ranqueamento são compostos por informações de *query* e candidatos. Assim, para os experimentos com o *Ranking SVM* as imagens dos conjuntos de treinamento e teste foram divididas em imagens de *query* e imagens candidatas. Como os classificadores baseados em OPF foram adaptados com o objetivo de criar um ranqueamento para as imagens do conjunto de teste, as imagens do conjunto de treinamento são consideradas imagens candidatas e as do conjunto de teste as imagens de *query*.

Nos experimentos realizados com a abordagem que utiliza a distância entre os vetores de características do par (*query*, candidata) para recuperar as r imagens mais relevantes, os conjuntos de treinamento e teste são considerados de *queries* e candidatas respectivamente. As configurações dos *folds* utilizados nos experimentos são descritas na subseção 6.1.2.

6.1.2 Configurações dos *Folds*

A base de imagens Brodatz contém 112 imagens, sendo uma de cada classe diferente. Para que cada classe fosse representada por um número maior de amostras nos experimentos, cada imagem do conjunto original foi dividida em 16 partes, formando um *dataset* com 1792 imagens e 112 classes, com 16 amostras por classe.

A base de dados Caltech101 contém 8.677 imagens coloridas, agrupadas em 101 classes

e o número de imagens por classe varia de 40 a 800. Suas classes são mutuamente exclusivas, isto é, uma imagem pode ser associada a apenas uma classe. Todas as imagens desse *dataset* foram consideradas nos experimentos.

A base de imagens MPEG-7 contém 1400 imagens e 70 classes também mutuamente exclusivas. Todas as imagens dessa também foram utilizadas nos experimentos. As Tabelas 1, 2 e 3 demonstram as configurações dos *folds* utilizados nos experimentos para as diferentes bases de imagens.

Tabela 1 – Configuração utilizada para a base de dados Brodatz.

Total de imagens	**Total de classes**	***Folds***	**Treinamento**	**Teste**
1792	112	25% x 75%	448	1344
		50% x 50%	896	896
		75% x 25%	1344	448

Fonte: Elaborado pela autora.

Tabela 2 – Configuração utilizada para a base de dados Caltech101.

Total de imagens	**Total de classes**	***Folds***	**Treinamento**	**Teste**
8677	101	25% x 75%	2170	6507
		50% x 50%	896	896
		75% x 25%	4338	4338

Fonte: Elaborado pela autora.

Tabela 3 – Configuração utilizada para a base de dados MPEG-7.

Total de imagens	**Total de classes**	***Folds***	**Treinamento**	**Teste**
1400	70	25% x 75%	350	1050
		50% x 50%	700	700
		75% x 25%	1050	350

Fonte: Elaborado pela autora.

Para a realização dos experimentos, as imagens de cada *fold* foram escolhidas aleatoriamente para cada uma das bases de imagem e esses foram repetidos 10 vezes para cada cenário. Na seção 6.2 os melhores resultados, para cada cenário, descritor e ranqueamento em r, são destacados em negrito, de acordo com o teste de Wilcoxon (*signed-rank*) com significância de 0.05. Cabe mencionar que o teste de Wilcoxon é um teste de hipóteses não paramétrico utilizado para comparar duas amostras relacionadas ou medidas repetidas em uma dada amostra com o objetivo de verificar se suas distribuições populacionais médias diferem.

6.1.3 Extração de Características

Para extração dos atributos de textura, cor e forma das imagens das três bases de dados supracitadas foram utilizados sete descritores de imagens globais: ACC (HUANG et al., 1997), BIC (STEHLING; NASCIMENTO; FALCãO, 2002), CCV (PASS; ZABIH; MILLER, 1997), LCH (SWAIN; BALLARD, 1991), LBP (OJALA; PIETIKäINEN; MäENPää, 2002), SASI (ÇARKACIOGLU; VURAL, 2003) e SPYTEC (LU; CHANGS, 2007). Sendo que os quatro primeiros descritores extraem características referentes à informação de cor e foram aplicados às imagens do *dataset* Caltech101, os descritores de textura LBP e SASI foram aplicados ao *dataset* Brodatz e, por fim, para o *dataset* MPEG-7 aplicou-se o descritor SPYTEC na extração de características referentes à forma, conforme Tabela 4.

Tabela 4 – Descritores utilizados nos experimentos.

Descritor	**Tipo de característica**	***Dataset***
Auto-color correlation (ACC)	Cor	Caltech 101
Border/Interior Pixel Classification (BIC)		
Color Coherence Vectors (CCV)		
Local Color Histogram (LCH)		
Local Binary Pattern (LBP)	Textura	Brodatz
Statistical Analysis of Structural Information		
Spherical Pyramid-Technique (SPYTEC)	Forma	MPEG-7

Fonte: Elaborado pela autora.

6.1.4 Indicador de Relevância

Para indicar a relevância entre um par de imagens (*query*, candidata) optou-se por utilizar o conceito de relevância absoluta nos experimentos, isto é, os modelos de ranqueamento utilizados consideram uma dada imagem candidata como relevante ou irrelevante para dada *query*. Assim, após o ranqueamento, cada imagem candidata recebe o rótulo 0 ou 1, sendo que se essa possui a mesma classe da *query* com a qual está sendo comparada, então o rótulo 1 lhe é atribuído, caso contrário, atribui-se o valor 0 à informação de relevância. Para validar o desempenho dos modelos, é verificado se as r primeiras imagens ranqueadas são de fato relevantes, isto é, se possuem a mesma classe da imagem de *query*.

6.1.5 Etapas de Treinamento e Teste

As configurações utilizadas nas etapas de treinamento e teste para cada uma das técnicas utilizadas nos experimentos são apresentadas a seguir.

6.1.5.1 Abordagens Baseadas em Floresta de Caminhos Ótimos

Para os classificadores baseados em OPF, cada *dataset* foi dividido em dois conjuntos disjuntos, de treinamento e teste, considerando as configurações dos *folds* 1, 2 e 3. Logo, manteve-se o modelo de entrada de dados para as fases de treinamento e testes que é frequentemente utilizado pelos classificadores em tarefas de classificação.

Para aplicação dos algoritmos no cenário de ranqueamento, como os classificadores objetivam criar um ranqueamento em r das imagens do conjunto de treinamento para dada imagem a ser classificada na etapa de teste, as imagens do primeiro conjunto são consideradas as imagens candidatas e as do conjunto de teste as de *query*.

6.1.5.2 Abordagem de Distância entre Vetores de Características

Na abordagem que utiliza a informação de distância entre vetores de características, para recuperar as r imagens mais relevantes para certa imagem de consulta, os conjuntos de imagens que representam as *queries* e as imagens candidatas também foram divididos de acordo com os três *folds*.

Para ranquear as r imagens mais relevantes à uma dada imagem de *query*, essa última foi comparada a cada uma das candidatas e, após comparação entre a distância dos vetores de características de cada par (*query*, candidata), a informação de distância entre eles foi utilizada como indicativo de similaridade entre as imagens. Posteriormente, para gerar o ranqueamento para dada imagem de consulta, ordenou-se as similaridades de forma crescente, partindo do pressuposto que, quanto menor a distância entre duas imagens, maior a similaridade entre elas. Cabe mencionar que a informação de distância utilizada nos experimentos para cada descritor de imagens consiste na distância proposta originalmente para esse por seus criadores.

6.1.5.3 *Ranking SVM*

Conforme mencionado, optou-se por utilizar o conceito de relevância absoluta nesse trabalho. Nos experimentos realizados com o *Ranking SVM*, os *datasets* utilizados também foram divididos em três *folds* e cada conjunto destinado à fase de treinamento e teste foi novamente dividido em imagens consideradas *query* e imagens candidatas. Dessa forma, os dados fornecidos como entrada ao *Ranking SVM* na fase de treinamento compõem uma matriz computada conforme descrito a seguir:

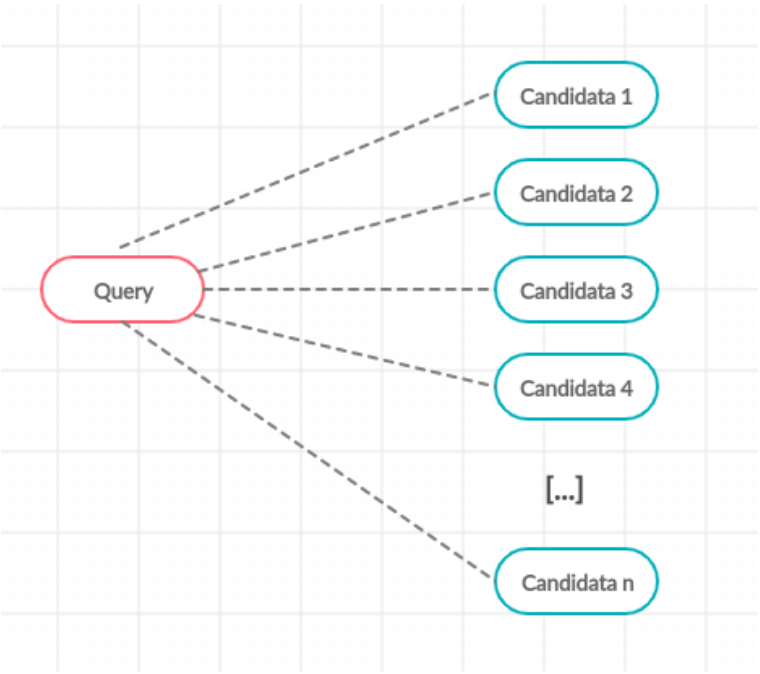
- Se uma imagem candidata possui a mesma classe da imagem *query* com a qual está sendo comparada, então o rótulo 1 é atribuído ao par, caso contrário, atribui-se o valor 0 à informação de relevância. Essa informação é armazenada na primeira coluna da matriz;
- A classe da *query* com a qual uma dada imagem candidata está sendo comparada é armazenada na segunda coluna da matriz;

- Após extrair as características da imagem candidata e comparar com o vetor de características da *query* em questão, a similaridade obtida entre elas é armazenada na terceira coluna da matriz.

Cada linha da matriz representa um par (*query*, candidata). A matriz gerada à partir dos dados do conjunto de treinamento é utilizada para treinar o modelo gerado pelo *Ranking SVM*. Uma outra matriz computada de forma análoga e que contém a comparação entre cada par de *query* e imagem candidata do conjunto de teste em suas linhas é fornecida juntamente com o modelo gerado na fase de treinamento como entrada para a fase de teste do algoritmo. Na fase de teste, o método *Ranking SVM* tem como objetivo gerar um ranqueamento das imagens candidatas do conjunto de teste para dada *query* também pertencente a esse conjunto.

A Figura 9 ilustra uma amostra de *query* e a comparação entre essa e cada uma das amostras candidatas em dado conjunto de dados.

Figura 9 – Amostra de query x amostras candidatas.



Fonte: Elaborado pela autora.

6.2 Resultados Experimentais

Os resultados dos experimentos realizados com os algoritmos baseados em Floresta de Caminhos Ótimos propostos para criação de ranqueamento, a abordagem baseada na distância entre os vetores de características do par *query* - candidata e o método *Ranking SVM* são apresentados em tabelas que representam os diferentes cenários dos experimentos, de acordo com a base de imagens e *fold* utilizados.

Cada uma das tabelas apresenta os valores de $NDGC$ e MAP obtidos para cada r em dado cenário. Lembrando que o *fold* 1 apresenta a seguinte configuração: 25% das imagens compõem o conjunto de treinamento e 75% das imagens compõem o conjunto de testes; *fold* 2: 50% treinamento e 50% teste; *fold* 3: 75% treinamento e 25% teste. Os melhores resultados de $NDGC$ e MAP para cada valor de r são destacados em negrito. Note que OPF GC refere-se à abordagem baseada em OPF com grafo completo e OPF k-*nn* à que utiliza grafo k-*nn*.

6.2.1 Brodatz

Para a base de dados Brodatz, nas três tabelas a seguir, pode-se observar que os melhores resultados alcançados pelas abordagens baseadas em OPF foram obtidos ao utilizar o descritor SASI, independentemente do *fold* utilizado. Quanto à comparação entre essas abordagens e as demais técnicas, a abordagem com grafo completo apresentou melhor desempenho do que a que utiliza grafo k-*nn* no *fold* 25% x 75%. O *Ranking SVM* apresentou os melhores resultados gerais.

Tabela 5 – Resultados obtidos com o *fold* 1 e base de imagens Brodatz.

Brodatz								
Fold	**Técnica**	**Descritor**	r					
			10		**15**		**20**	
			NDGC	MAP	NDGC	MAP	NDGC	MAP
25 x 75	Distância	LBP	**0.370**	**0.225**	0.377	**0.182**	0.382	0.152
		SASI	0.392	0.260	0.393	0.170	0.396	0.140
	OPF GC	LBP	**0.354**	0.162	0.371	0.124	0.375	0.102
		SASI	0.405	0.199	0.410	0.154	0.411	0.127
	OPF k-*nn*	LBP	0.328	0.135	0.345	0.106	0.349	0.087
		SASI	0.378	0.189	0.381	0.146	0.383	0.120
	Ranking SVM	LBP	0.362	0.145	**0.383**	0.115	**0.388**	0.094
		SASI	**0.431**	**0.328**	**0.434**	**0.264**	**0.436**	**0.225**

Fonte: Elaborado pela autora.

Tabela 6 – Resultados obtidos com o *fold* 2 e base de imagens Brodatz.

Brodatz								
Fold	**Técnica**	**Descritor**	r					
			10		**15**		**20**	
			NDGC	MAP	NDGC	MAP	NDGC	MAP
50 x 50	Distância	LBP	0.368	0.202	0.378	0.164	0.386	0.140
		SASI	0.406	0.289	0.411	0.236	0.411	0.201
	OPF GC	LBP	0.391	0.214	0.402	0.174	0.410	0.148
		SASI	0.428	0.222	0.431	0.176	0.432	0.148
	OPF k-*nn*	LBP	0.361	0.197	0.370	0.160	0.378	0.137
		SASI	0.432	0.308	0.435	0.251	0.437	0.212
	Ranking SVM	LBP	**0.409**	**0.224**	**0.420**	**0.182**	**0.429**	**0.155**
		SASI	**0.451**	**0.321**	**0.455**	**0.262**	**0.457**	**0.222**

Fonte: Elaborado pela autora.

Tabela 7 – Resultados obtidos com o *fold* 3 e base de imagens Brodatz.

Brodatz								
Fold	**Técnica**	**Descritor**	r					
			10		**15**		**20**	
			NDGC	MAP	NDGC	MAP	NDGC	MAP
75 x 25	Distância	LBP	0.341	0.145	0.359	0.118	0.364	0.099
		SASI	0.406	0.289	0.411	0.236	0.411	0.201
	OPF GC	LBP	0.410	0.255	0.418	0.210	0.423	0.179
		SASI	0.402	0.209	0.405	0.166	0.407	0.139
	OPF k-*nn*	LBP	0.378	0.235	0.385	0.194	0.389	0.165
		SASI	0.403	0.334	0.403	0.274	0.406	0.236
	Ranking SVM	LBP	**0.429**	**0.267**	**0.437**	**0.220**	**0.442**	**0.187**
		SASI	**0.457**	**0.379**	**0.457**	**0.311**	**0.460**	**0.268**

Fonte: Elaborado pela autora.

6.2.2 Caltech101

Para a base de dados Caltech101, os descritores BIC e LCH apresentaram os melhores resultados nos experimentos. A abordagem baseada em OPF com grafo completo forneceu valores de relevância maiores do que os obtidos com a abordagem de grafo k-*nn* em todos os *folds*. Todas as técnicas, exceto a que utiliza a medida de distância para ranquear as r imagens mais relevantes, apresentaram melhora nos resultados com o aumento do número de amostras do conjunto de treinamento. Os melhores resultados gerais foram obtidos pela técnica *Ranking SVM*.

Tabela 8 – Resultados obtidos com o *fold* 1 e base de imagens Caltech.

Caltech								
Fold	**Técnica**	**Descritor**	r					
			10		**15**		**20**	
			NDGC	MAP	NDGC	MAP	NDGC	MAP
25 x 75	Distância	ACC	0.252	**0.151**	0.270	**0.143**	0.283	**0.136**
		BIC	0.264	**0.162**	0.281	**0.155**	0.294	**0.149**
		CCV	0.242	**0.140**	0.263	**0.132**	0.277	**0.126**
		LCH	0.269	0.162	0.289	0.151	0.302	0.142
	OPF GC	ACC	0.250	0.141	0.274	0.132	0.286	0.125
		BIC	0.259	0.151	**0.284**	0.142	0.267	0.135
		CCV	0.238	0.125	0.260	0.118	0.276	0.112
		LCH	0.264	0.156	0.282	0.144	0.299	0.138
	OPF k-*nn*	ACC	0.232	0.132	0.254	0.124	0.266	0.116
		BIC	0.241	0.142	0.259	0.133	0.273	0.126
		CCV	0.222	0.118	0.242	0.111	0.257	0.105
		LCH	0.246	0.146	0.262	0.136	0.277	0.128
	Ranking SVM	ACC	**0.260**	0.146	**0.258**	0.137	**0.298**	0.129
		BIC	**0.270**	0.157	**0.263**	0.147	**0.306**	0.140
		CCV	**0.248**	0.130	**0.244**	0.122	**0.288**	0.116
		LCH	**0.275**	**0.174**	**0.294**	**0.164**	**0.311**	**0.156**

Fonte: Elaborado pela autora.

Tabela 9 – Resultados obtidos com o *fold* 2 e base de imagens Caltech.

Caltech								
Fold	**Técnica**	**Descritor**	r					
			10		**15**		**20**	
			NDGC	MAP	NDGC	MAP	NDGC	MAP
50 x 50	Distância	ACC	0.283	0.182	0.301	0.174	0.314	**0.167**
		BIC	0.295	**0.186**	0.312	0.186	0.325	**0.180**
		CCV	0.273	0.171	0.294	0.163	0.309	**0.157**
		LCH	0.300	0.193	0.320	0.182	0.333	0.173
	OPF GC	ACC	0.281	0.172	0.305	0.163	0.317	0.157
		BIC	0.290	0.182	0.310	0.173	0.325	**0.175**
		CCV	0.269	0.156	0.291	0.149	0.307	0.143
		LCH	0.295	0.187	0.313	0.175	0.330	0.172
	OPF k-*nn*	ACC	0.263	0.163	0.285	0.155	0.297	0.147
		BIC	0.272	0.173	0.290	0.164	0.304	0.157
		CCV	0.253	0.149	0.273	0.142	0.287	0.137
		LCH	0.277	0.177	0.293	0.167	0.310	0.159
	Ranking SVM	ACC	**0.291**	**0.177**	**0.316**	**0.169**	**0.329**	**0.160**
		BIC	**0.301**	**0.190**	**0.321**	**0.180**	**0.337**	**0.171**
		CCV	**0.264**	**0.146**	**0.287**	**0.138**	**0.304**	**0.132**
		LCH	**0.306**	**0.205**	**0.325**	**0.195**	**0.342**	**0.187**

Fonte: Elaborado pela autora.

Tabela 10 – Resultados obtidos com o *fold* 3 e base de imagens Caltech.

Caltech								
Fold	**Técnica**	**Descritor**	r					
			10		**15**		**20**	
			NDGC	MAP	NDGC	MAP	NDGC	MAP
75 x 25	Distância	ACC	0.340	0.239	0.358	0.231	0.371	0.224
		BIC	0.352	0.241	0.369	0.243	0.382	0.237
		CCV	0.330	0.228	0.351	0.221	0.366	0.214
		LCH	0.357	0.250	0.377	0.239	0.390	0.230
	OPF GC	ACC	0.338	0.229	0.362	0.220	0.374	0.213
		BIC	0.347	0.239	0.367	0.230	0.382	0.223
		CCV	0.326	0.213	0.348	0.206	0.364	0.201
		LCH	0.352	**0.248**	**0.372**	0.232	0.387	0.230
	OPF k-*nn*	ACC	0.321	0.220	0.342	0.212	0.354	0.204
		BIC	0.329	0.230	0.347	0.221	0.361	0.216
		CCV	0.310	0.206	0.330	0.199	0.338	0.193
		LCH	0.336	0.235	0.351	0.224	0.367	0.216
	Ranking SVM	ACC	**0.348**	**0.234**	**0.373**	**0.226**	**0.386**	**0.217**
		BIC	**0.358**	**0.247**	**0.378**	**0.237**	**0.396**	**0.228**
		CCV	**0.336**	**0.218**	**0.359**	**0.210**	**0.376**	**0.205**
		LCH	**0.363**	**0.262**	**0.382**	**0.252**	**0.399**	**0.253**

Fonte: Elaborado pela autora.

6.2.3 MPEG-7

Para a base de imagens MPEG-7, novamente a abordagem baseada em OPF com grafo completo superou os resultados obtidos com a baseada em grafo k-*nn* para todos os *folds*. Para esse *dataset*, a primeira abordagem apresentou resultados competitivos quando comparada à técnica *Ranking SVM*.

Tabela 11 – Resultados obtidos com o *fold* 1 e base de imagens MPEG-7.

MPEG-7								
Fold	**Descritor**	**Técnica**	r					
			10		15		20	
			NDGC	MAP	NDGC	MAP	NDGC	MAP
25 x 75	SPYTEC	Distância	0.061	0.030	0.071	0.027	0.074	0.026
		OPF GC	**0.076**	0.078	0.071	0.030	0.088	0.029
		OPF k-*nn*	0.066	0.031	0.072	0.028	0.082	0.027
		Ranking SVM	0.075	**0.034**	**0.082**	**0.031**	**0.092**	**0.031**

Fonte: Elaborado pela autora.

Tabela 12 – Resultados obtidos com o *fold* 2 e base de imagens MPEG-7.

MPEG-7								
Fold	**Descritor**	**Técnica**	r					
			10		15		20	
			NDGC	MAP	NDGC	MAP	NDGC	MAP
50 x 50	SPYTEC	Distância	0.087	0.051	0.093	0.051	0.108	0.052
		OPF GC	0.094	0.054	0.098	0.053	0.113	0.055
		OPF *k-nn*	0.086	0.050	0.093	0.050	0.107	0.052
		Ranking SVM	**0.095**	**0.055**	**0.101**	**0.056**	**0.118**	**0.057**

Fonte: Elaborado pela autora.

Tabela 13 – Resultados obtidos com o *fold* 3 e base de imagens MPEG-7.

MPEG-7								
Fold	**Descritor**	**Técnica**	r					
			10		15		20	
			NDGC	MAP	NDGC	MAP	NDGC	MAP
75 x 25	SPYTEC	Distância	0.134	0.097	0.140	0.095	0.147	0.095
		OPF GC	0.130	0.103	0.141	0.101	0.145	**0.102**
		OPF *k-nn*	0.126	0.099	0.136	0.098	0.138	0.098
		Ranking SVM	**0.137**	**0.103**	**0.145**	**0.102**	**0.150**	**0.103**

Fonte: Elaborado pela autora.

6.3 Tempos de Execução

Nesta seção, a eficiência das técnicas é analisada considerando-se o tempo de execução gasto por cada uma delas para ranquear as r amostras mais relevantes para dada *query* nos diferentes cenários abrangidos nos experimentos. As Tabelas 14, 15 e 16 apresentam os tempos de execução médios, em segundos, despendidos na fase de teste, para cada uma das técnicas, e bases de imagens Brodatz, Caltech101 e MPEG-7 respectivamente. Esses valores médios foram obtidos em 10 rodadas dos experimentos. Os menores tempos são destacados em negrito.

Tabela 14 – Tempos de execução - Fase de teste - Brodatz.

Técnica	**Descritor**	r								
		10			**15**			**20**		
		25x75	50x50	75x25	25x75	50x50	75x25	25x75	50x50	75x25
Distância	LBP	31.017	33.015	32.148	32.005	35.002	32.378	33.069	36.834	32.952
	SASI	31.492	34.654	33.335	33.262	35.398	34.038	34.511	37.414	34.111
OPF GC	LBP	19.158	**18.235**	**16.226**	20.509	**19.457**	**16.476**	21.15	**19.908**	**17.339**
	SASI	**16.749**	**18.853**	**16.642**	**17.906**	**19.953**	**16.758**	**18.209**	**20.204**	**17.954**
OPF k-*nn*	LBP	**17.298**	19.344	17.246	**18.408**	20.455	17.575	**18.733**	20.745	18.329
	SASI	18.461	20.457	18.375	19.617	21.616	18.553	20.126	22.064	19.187
Ranking SVM	LBP	32.358	34.358	34.508	33.366	35.364	35.017	34.052	35.971	35.213
	SASI	33.125	35.123	36.199	34.494	36.500	36.491	34.878	36.837	37.110

Fonte: Elaborado pela autora.

Tabela 15 – Tempos de execução - Fase de teste - Caltech101.

Técnica	**Descritor**	r								
		10			**15**			**20**		
		25x75	50x50	75x25	25x75	50x50	75x25	25x75	50x50	75x25
Distância	ACC	40.985	40.168	39.251	41.723	40.957	40.095	42.220	41.387	40.497
	BIC	40.662	40.296	39.351	41.069	41.487	40.718	41.623	41.811	40.919
	CCV	41.423	39.169	37.662	41.744	39.483	37.982	42.196	39.691	38.098
	LCH	39.464	38.072	36.876	40.530	38.106	36.608	40.641	38.848	37.381
OPF GC	ACC	**21.031**	**21.711**	**20.662**	**21.861**	**22.127**	**20.982**	**22.558**	**22.524**	**21.442**
	BIC	**22.316**	**22.607**	**21.772**	**23.039**	**23.065**	**21.982**	**23.401**	**23.546**	**22.437**
	CCV	**21.301**	**21.712**	**20.773**	**22.068**	**22.298**	**21.329**	**22.738**	**23.091**	**22.206**
	LCH	**22.557**	**22.828**	**21.769**	**14.091**	**23.101**	**21.979**	**23.381**	**23.461**	**22.329**
OPF k-*nn*	ACC	22.625	22.699	21.552	13.960	23.123	21.973	14.267	23.494	22.351
	BIC	23.459	24.052	22.215	23.953	24.381	22.551	24.390	**24.796**	22.980
	CCV	22.276	23.520	21.662	**14.107**	24.116	22.351	23.719	24.515	22.766
	LCH	**23.138**	25.298	23.329	**23.256**	25.868	23.881	**24.133**	26.145	24.215
Ranking SVM	ACC	41.869	43.083	41.551	42.392	43.192	42.495	43.012	44.672	43.297
	BIC	42.394	41.876	40.329	42.946	42.479	40.971	43.414	43.032	41.440
	CCV	43.506	43.180	41.895	43.933	43.611	42.351	44.277	43.988	42.774
	LCH	42.612	43.939	41.440	43.647	45.060	42.773	44.302	45.374	43.093

Fonte: Elaborado pela autora.

Tabela 16 – Tempos de execução - Fase de teste - MPEG-7.

Técnica	**Descritor**	r								
		10			**15**			**20**		
		25x75	50x50	75x25	25x75	50x50	75x25	25x75	50x50	75x25
SPYTEC	Distance	28.184	28.171	29.665	29.352	29.580	30.868	29.751	29.806	31.123
	OPF GC	**18.352**	**18.119**	**19.350**	**20.237**	**20.270**	**21.767**	**21.559**	**21.723**	**22.614**
	OPF k-*nn*	20.117	20.340	21.322	21.241	21.248	21.959	22.005	22.500	24.174
	Ranking SVM	32.783	33.135	34.661	33.553	33.814	34.637	34.120	34.597	35.464

Fonte: Elaborado pela autora.

7 Conclusão e Trabalhos Futuros

O presente trabalho teve como principal contribuição e objetivo aplicar os classificadores baseados em Floresta de Caminhos Ótimos, na versão supervisionada com grafo completo e k-*nn*, à tarefa de *learning to rank*, uma vez que esses, até o presente momento, não haviam sido aplicados a esse contexto.

Nos experimentos realizados neste trabalho, considerou-se as imagens digitais como objeto de estudo na tarefa de ranqueamento de informações. As características de cor, forma e textura das imagens de três bases de dados comumente utilizadas na literatura foram extraídas por sete descritores de imagens globais. O conceito de relevância absoluta, que considera uma informação candidata como relevante ou não relevante para certa *query*, foi utilizado para ranquear as r imagens mais similares à dada imagem de *query*.

Para aplicar ambos os classificadores à tarefa de aprender a ranquear, foram realizadas modificações em seus respectivos algoritmos na fase de classificação (teste), a fim de que um ranqueamento em r das amostras do conjunto de treinamento que apresentem os melhores custos para dada amostra a ser classificada seja criado.

Nos experimentos realizados, as métricas $NDGC$ e MAP, comumente utilizadas para validar o desempenho de modelos de ranqueamento, foram utilizadas para medir a eficácia dos classificadores. Três *folds* foram gerados, com imagens escolhidas aleatoriamente, para cada um dos *datasets* utilizados e os experimentos foram executados 10 vezes em cada cenário com o intuito de verificar se há constância nos resultados obtidos.

Os resultados alcançados com as abordagens baseadas em OPF foram comparados aos obtidos pelas técnicas *Ranking SVM* e a uma abordagem comumente utilizada no cenário de recuperação de informações, que consiste em recuperar as r imagens mais similares à dada imagem de *query*, isto é, as r mais relevantes para tal, por meio da comparação entre os vetores de características da imagem de consulta e cada uma das imagens candidatas do conjunto de dados.

Com relação aos valores de acurácia obtidos na tarefa de aprender a ranquear, o classificador baseado em OPF com grafo completo apresentou resultados similares aos obtidos com o *Ranking SVM* em diferentes cenários. O método *Ranking SVM*, por sua vez, apresentou os melhores valores gerais de $NDGC$ e MAP para todos os cenários analisados. No entanto, as abordagens propostas demonstraram ser muito mais rápidas do que as técnicas baseadas em distância e *Ranking SVM*, uma vez que apresentaram os menores tempos de execução na fase de teste quando comparadas às demais técnicas. Assim, considerando o *trade-off* entre a relevância no ranqueamento em r e o tempo empregado na etapa de recuperação das r amostras mais relevantes, as abordagens baseadas em OPF demonstraram ter o melhor

custo-benefício, uma vez que essas alcançaram resultados próximos aos do *Ranking SVM*, mas de modo muito mais rápido.

Além disso, nos experimentos realizados o tempo de treinamento despendido por cada técnica não foi considerado, mas os das abordagens baseadas em OPF foram menores do que os das demais. Vale destacar que a abordagem com grafo completo não utiliza nenhum parâmetro previamente informado, tornando-se mais fácil de configurar e não necessitando de uma etapa de ajustes dos parâmetros, como a técnica *Ranking SVM* cujo número de parâmetros varia de acordo com a função de *kernel* utilizada. Outro ponto que cabe ser destacado é que as abordagens propostas neste trabalho foram pouco modificadas para lidar com problemas de ranqueamento; enquanto o *Ranking SVM* possui uma quantidade maior de adaptações quando comparado à versão original do classificador SVM.

Dessa forma, conclui-se que a aplicação dos classificadores baseados em Floresta de Caminhos Ótimos ao contexto de criação de ranqueamento mostrou-se promissora em seus experimentos iniciais. Assim, esses podem ser considerados uma possível solução para aquelas aplicações que necessitam ranquear informações como atividade principal ou estratégica.

Para trabalhos futuros, espera-se obter melhores resultados com maiores modificações nos algoritmos dos classificadores baseados em OPF, de modo que esses lidem melhor com o problema de ranqueamento. Além disso, tem-se a ideia de utilizar os ranqueamentos gerados pelas abordagens com grafo completo e k-*nn* como entrada para métodos destinados à agregação de ranqueamento, a fim de verificar se esses apresentam melhora em sua acurácia quando comparados ao desempenho obtido com entradas fornecidas por outros algoritmos destinados à criação de ranqueamento. Para próximos experimentos, também considera-se realizar o pré-processamento das imagens utilizadas antes de efetuar o *ranking* dessas, além de representar as imagens fornecidas como entrada aos classificadores utilizando diferentes características relevantes combinadas entre si.

Referências

ALZU'BI, A.; AMIRA, A.; RAMZAN, N. Semantic content-based image retrieval: A comprehensive study. *Journal of Visual Communication and Image Representation*, v. 32, p. 20 – 54, 2015.

AMORIM, W. P.; FALCãO, A. X.; PAPA, J. P. Multi-label semi-supervised classification through optimum-path forest. *Information Sciences*, v. 465, p. 86 – 104, 2018.

AMORIM, W. P. et al. Improving semi-supervised learning through optimum connectivity. *Pattern Recognition*, v. 60, p. 72 – 85, 2016.

BIJALWAN, V. et al. Knn based machine learning approach for text and document mining. *International Journal of Database Theory and Application*, 2014.

BRODATZ, P. *Textures: a photographic album for artists and designers, by Phil Brodatz*. [S.l.]: Dover publications, 1966.

BUGATTI, P. H. et al. Prosper: Perceptual similarity queries in medical cbir systems through user profiles. *Computers in Biology and Medicine*, v. 45, p. 8 – 19, 2014.

CAO, Z. et al. Learning to rank: From pairwise approach to listwise approach. In: *Proceedings of the 24th International Conference on Machine Learning*. [S.l.: s.n.], 2007.

CHEN, J. et al. Face image quality assessment based on learning to rank. *IEEE Signal Processing Letters*, v. 22, p. 90–94, 2015.

CHEN, W. et al. Ranking measures and loss functions in learning to rank. In: . [S.l.]: Curran Associates, Inc., 2009. p. 315–323.

CHEN, X. et al. Learning to rank features for recommendation over multiple categories. In: . [S.l.]: ACM, 2016. p. 305–314.

DATTA, R.; LI, J.; WANG, J. Z. Content-based image retrieval—approaches and trends of the new age. In: . [S.l.: s.n.], 2005. v. 40, p. 253–262.

DEMIR, B.; BRUZZONE, L. A novel active learning method in relevance feedback for content-based remote sensing image retrieval. *IEEE Transactions on Geoscience and Remote Sensing*, v. 53, p. 2323–2334, 2015.

DHARANI, T.; AROQUIARAJ, I. L. A survey on content based image retrieval. In: *2013 International Conference on Pattern Recognition, Informatics and Mobile Engineering*. [S.l.: s.n.], 2013. p. 485–490.

DHAWALE, S.; JOGLEKAR, B.; KULKARNI, P. A graph-based active learning approach using forest classifier for image retrieval. In: *Proceedings of the International Conference on Data Engineering and Communication Technology*. [S.l.]: Springer Singapore, 2017. p. 119–129.

DHUNGEL, N.; CARNEIRO, G.; BRADLEY, A. P. The automated learning of deep features for breast mass classification from mammograms. In: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2016*. [S.l.]: Springer International Publishing, 2016. p. 106–114.

DUH, K.; KIRCHHOFF, K. Learning to rank with partially-labeled data. In: *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. [S.l.]: ACM, 2008. p. 251–258.

FALCãO, A.; STOLFI, J.; LOTUFO, R. The image foresting transform: Theory, algorithms, and applications. IEEE Transactions on Pattern Analysis and Machine Intelligence, p. 26(1):19–29, 2004.

FARIA, F. F. et al. Learning to rank for content-based image retrieval. In: . [S.l.]: ACM, 2010. p. 285–294.

FEI-FEI, L.; FERGUS, R.; PERONA, P. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In: *2004 Conference on Computer Vision and Pattern Recognition Workshop*. [S.l.]: IEEE, 2004.

FRAKES, W. B.; BAEZA-YATES, R. (Ed.). *Information Retrieval: Data Structures and Algorithms*. [S.l.]: Prentice-Hall, Inc., 1992.

GANJISAFFAR, Y. *Tree Ensembles for Learning to Rank*. Tese (Doutorado), 2011.

GANTZ, J.; REINSEL, D. *The digital universe in 2020*. 2012.

GONZáLEZ-DíAZ, I. et al. Neighborhood matching for image retrieval. *IEEE Transactions on Multimedia*, p. 544–558, 2017.

GREENGRASS, E. *Information Retrieval: A Survey*. [S.l.: s.n.], 2000. 1-6 p.

HALL, B. Facies classification using machine learning. *The Leading Edge*, v. 35, p. 906–909, 10 2016.

HE, T. et al. Content based image retrieval method based on sift feature. In: *2018 International Conference on Intelligent Transportation, Big Data Smart City (ICITBS)*. [S.l.: s.n.], 2018. p. 649–652.

HERSHEY, S. et al. Cnn architectures for large-scale audio classification. In: *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. [S.l.: s.n.], 2017. p. 131–135.

HUANG, J. et al. Image indexing using color correlograms. In: *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. [S.l.: s.n.], 1997. p. 762–768.

HUANG, P.-S. et al. Learning deep structured semantic models for web search using clickthrough data. In: . [S.l.]: ACM, 2013. p. 2333–2338.

ILIADIS, M. et al. Virtual touring: A content based image retrieval application. In: *2013 IEEE International Conference on Multimedia and Expo Workshops (ICMEW)*. [S.l.: s.n.], 2013. p. 1–4.

JAIN, M.; SINGH, D. A survey on cbir on the basis of different feature descriptor. *British Journal of Mathematics Computer Science*, p. 1–13, 2016.

JI, Z.; WANG, B. Learning to rank for question routing in community question answering. In: . [S.l.]: ACM, 2013. p. 2363–2368.

JIN, X. et al. Significance and challenges of big data research. *Big Data Research*, v. 2, p. 59–64, 2015.

JOACHIMS, T. Optimizing search engines using clickthrough data. *SIGKDD 02*, ACM, p. 1–10, 2002.

KAHOU, S. E. et al. Recurrent neural networks for emotion recognition in video. In: *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*. [S.l.]: ACM, 2015. p. 467–474.

KARATZOGLOU, A.; BALTRUNAS, L.; SHI, Y. Learning to rank for recommender systems. In: . [S.l.]: ACM, 2013. p. 493–494.

KAUR, N.; JINDAL, S.; KAUR, B. Relevance feedback based cbir system using svm and bayes classifier. In: *2016 Second International Conference on Computational Intelligence Communication Technology (CICT)*. [S.l.: s.n.], 2016. p. 214–218.

KHOKHER, A.; TALWAR, R. Content-based image retrieval: State-of-the-art and challenges. 2011.

KHOSLA, G.; RAJPAL, N.; SINGH, J. Evaluation of euclidean and manhanttan metrics in content based image retrieval system. In: . [S.l.: s.n.], 2015. p. 12–18.

LI, H. A short introduction to learning to rank. *IEICE Transactions*, v. 94, p. 1854–1862, 2011.

LI, H. A short introduction to learning to rank. v. 94, p. 1854–1862, 2011.

LI, H. *Learning to Rank for Information Retrieval and Natural Language Processing*. [S.l.]: Morgan Claypool publishers, 2014. 1–121 p.

LI, H. et al. Retrieval of clothing images based on relevance feedback with focus on collar designs. *Vis. Comput.*, Springer-Verlag New York, Inc., v. 32, p. 1351–1363, 2016.

LI, Y.; SONG, Y.; LUO, J. Improving pairwise ranking for multi-label image classification. In: . [S.l.: s.n.], 2017. p. 1837–1845.

LI, Y. et al. A comprehensive study on learning to rank for content-based image retrieval. *Signal Processing*, v. 93, p. 1426 – 1434, 2013.

LOPES, G. S. et al. Recognition of handwritten digits using the signature features and optimum-path forest classifier. *IEEE Latin America Transactions*, v. 14, p. 2455–2460, 2016.

LORENA, A. C.; CARVALHO, A. C. P. L. F. de. *Introdução às Máquinas de Vetores Suporte*. [S.l.]: Universidade de São Paulo, 2003.

LU, T.; CHANGS, C. Color image retrieval technique based on color features and image bitmap. *Inf. Processing and Management*, v. 43, p. 461–472, 2007.

MANNING, P. R. C. D.; SCHUTZE., H. *Introduction to Information Retrieval*. [S.l.]: Cambridge University Press, 2009. 1-6 p.

MANSOURIAN, L. et al. An effective fusion model for image retrieval. *Multimedia Tools and Applications*, p. 16131–16154, 2018.

MARDONES, T.; ALLENDE, H.; MORAGA, C. Combining descriptors obtained through different sampling techniques in image retrieval. In: *2013 32nd International Conference of the Chilean Computer Science Society (SCCC)*. [S.l.: s.n.], 2013. p. 81–84.

MASOOD, A.; SHAHID, M. A.; SHARIF, M. Content-based image retrieval features: A survey. *International Journal of Advanced Networking Applications (IJANA)*, v. 10, p. 3741–3757, 2018.

MEHARBAN, M.; PRIYA, S. A review on image retrieval techniques. *Bonfring International Journal of Advances In Image Processing*, Bonfring, 2016.

MEMON, M. H. et al. Efficient object identification and multiple regions of interest using cbir based on relative locations and matching regions. In: *2015 12th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP)*. [S.l.: s.n.], 2015. p. 247–250.

MILOVANOVIC, M.; MINOVIC, M.; STARCEVIC, D. Walking in colors: Human gait recognition using kinect and cbir. *IEEE MultiMedia*, v. 20, p. 28–36, 2013.

MISTRY, Y.; INGOLE, D.; INGOLE, M. Content based image retrieval using hybrid features and various distance metric. *Journal of Electrical Systems and Information Technology*, v. 5, p. 874 – 888, 2018.

OJALA, T.; PIETIKäINEN, M.; MäENPää, T. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Trans. Pattern Anal. Mach. Intell.*, IEEE Computer Society, v. 24, p. 971–987, jul. 2002.

PAISITKRIANGKRAI, S.; SHEN, C.; HENGEL, A. van den. Learning to rank in person re-identification with metric ensembles. In: . [S.l.: s.n.], 2015.

PAPA., J. P. *Classificação supervisionada de padrões utilizando floresta de caminhos ótimos*. Tese (Doutorado) — UNICAMP, 2008.

PAPA, J. P.; FERNANDES, S. E. N.; FALCãO, A. X. Optimum-path forest based on k-connectivity: Theory and applications. *Pattern Recognition Letters*, v. 87, p. 117 – 126, 2017.

PAREEK, H. H.; RAVIKUMAR, P. K. A representation theory for ranking functions. In: . [S.l.]: Curran Associates, Inc., 2014. p. 361–369.

PAREEK, S.; MANDORIA, H. L. Comparison of image feature descriptor in content based image retrieval system. In: *2017 International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT)*. [S.l.: s.n.], 2017. p. 1509–1513.

PASS, G.; ZABIH, R.; MILLER, J. Comparing images using color coherence vectors. In: *Proceedings of the Fourth ACM International Conference on Multimedia*. [S.l.]: Association for Computing Machinery, 1997. p. 65–73.

PATIL, S.; TALBAR, S. Content based image retrieval using various distance metrics. In: *Data Engineering and Management*. [S.l.]: Springer Berlin Heidelberg, 2012. p. 154–161.

RALPH, R. Mpeg-7 core experiment ce-shape-1 [tar.gz]. 1999.

RANI, C. N. M. Importance of information retrieval. Oriental Journal of Computer Science Technology, p. 459–462, 2011.

RUSSEL, S.; NORVIG., P. *Inteligência Artificial*. [S.l.]: Elsevier Editora Ltda, 2013. 648-652 p.

SANDERSON, M.; CROFT, W. B. The history of information retrieval research. *Proceedings of the IEEE*, v. 100, p. 1444–1451, 2012.

SCHAEFER, G. An introduction to content-based image retrieval. In: *Eighth International Conference on Digital Information Management (ICDIM 2013)*. [S.l.: s.n.], 2013. p. 4–6.

SEVERYN, A.; MOSCHITTI, A. Learning to rank short text pairs with convolutional deep neural networks. In: *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*. [S.l.]: ACM, 2015. p. 373–382.

SHARMA, M.; PATEL, R. A survey on information retrieval models, techniques and applications. In: . [S.l.: s.n.], 2013.

SHEN, C.; LI, T. Learning to rank for query-focused multi-document summarization. In: *2011 IEEE 11th International Conference on Data Mining*. [S.l.: s.n.], 2011. p. 626–634.

SHEN, Y. et al. Learning semantic representations using convolutional neural networks for web search. In: . [S.l.]: ACM, 2014. p. 373–374.

SILVA, A. T. da et al. Incorporating multiple distance spaces in optimum-path forest classification to improve feedback-based learning. *Computer Vision and Image Understanding*, v. 116, p. 510 – 523, 2012.

SINGHAL, A.; GOOGLE, I. Modern information retrieval: A brief overview. *IEEE Data Engineering Bulletin*, v. 24, 2001.

STEHLING, R. O.; NASCIMENTO, M. A.; FALCãO, A. X. A compact and efficient image retrieval approach based on border/interior pixel classification. In: . [S.l.]: ACM, 2002. p. 102–109.

SWAIN, M. J.; BALLARD, D. H. Color indexing. *International Journal of Computer Vision*, v. 7, p. 11–32, 1991.

TAVARES, A. d. S.; FALCãO, A. X.; MAGALHãES, L. P. Active learning paradigms for CBIR systems based on optimum-path forest classification. *Pattern Recognition*, Elsevier Science Inc., v. 44, p. 2971 – 2978, 2011.

TORRES, R.; FALCãO, A. Content-based image retrieval: Theory and applications. v. 13, p. 161–185, 2006.

TRIPATHY, A.; AGRAWAL, A.; RATH, S. K. Classification of sentimental reviews using machine learning techniques. *Procedia Computer Science*, v. 57, p. 821 – 829, 2015. 3rd International Conference on Recent Trends in Computing 2015 (ICRTC-2015).

TZELEPI, M.; TEFAS, A. Deep convolutional learning for content based image retrieval. *Neurocomputing*, Elsevier, v. 275, 2017.

WAN, J. et al. Deep learning for content-based image retrieval: A comprehensive study. In: *Proceedings of the 22Nd ACM International Conference on Multimedia*. [S.l.]: ACM, 2014. p. 157–166.

WAN, J. et al. Online learning to rank for content-based image retrieval. In: . [S.l.]: AAAI Press, 2015. p. 2284–2290.

WANG, S. et al. Ranking-oriented collaborative filtering: A listwise approach. *ACM Trans. Inf. Syst.*, ACM, v. 35, 2016.

WANG, Y. et al. A theoretical analysis of ndcg type ranking measures. In: SHALEV-SHWARTZ, S.; STEINWART, I. (Ed.). *Proceedings of the 26th Annual Conference on Learning Theory*. [S.l.]: PMLR, 2013. v. 30, p. 25–54.

WATANABE, T. Optimized online rank learning for machine translation. In: *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. [S.l.]: Association for Computational Linguistics, 2012. p. 253–262.

YANG, X.; TANG, K.; YAO, X. A learning-to-rank approach to software defect prediction. *IEEE Transactions on Reliability*, v. 64, p. 234–246, 2015.

YIN, D. et al. Ranking relevance in yahoo search. In: *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. [S.l.]: ACM, 2016. p. 323–332.

YU, H.; KIM., S. *SVM Tutorial: Classification, Regression, and Ranking*. [S.l.]: Springer Berlin Heidelberg, 2012. 479-506 p.

YU, J. et al. Learning to rank using user clicks and visual features for image retrieval. *IEEE Transactions on Cybernetics*, v. 45, p. 767–779, 2015.

ZHOU, W.; LI, H.; TIAN, Q. Recent advance in content-based image retrieval: A literature survey. *CoRR*, 2017.

ÇARKACIOGLU, A.; VURAL, F. Y. Sasi: A generic texture descriptor for image retrieval. *Pattern Recognition*, v. 36, p. 2615–2633, 2003.