Universidade Federal de Alagoas
Instituto de Computação

Dissertação de Mestrado

Uma Abordagem Estocástica para Recuperação de Imagens de Roupa Baseada em Conceito Utilizando o SVM e o *Feedback* de Relevância do Usuário

Artur Maia Pereira
amp@ic.ufal.com

Orientadores:

Evandro de Barros Costa
Thales Miranda de Almeida Vieira

Maceió, Janeiro de 2018

Artur Maia Pereira

Uma Abordagem Estocástica para Recuperação de Imagens de Roupa Baseada em Conceito Utilizando o SVM e o *Feedback* de Relevância do Usuário

Dissertação apresentada como requisito parcial para obtenção do grau de Mestre pelo Curso de Mestrado em Informática do Instituto de Computação da Universidade Federal de Alagoas.

Orientadores:

Evandro de Barros Costa

Thales Miranda de Almeida Vieira

Maceió, Janeiro de 2018

Catalogação na fonte
Universidade Federal de Alagoas
Biblioteca Central
Bibliotecário: Marcelino de Carvalho

P436a Pereira, Artur Maia.
Uma abordagem estocástica para recuperação de imagens de roupa baseada em conceito utilizando o SVM e o *feedback* de relevância do usuário / Artur Maia Pereira. – 2019.
43 f. : il.

Orientadores: Evandro de Barros Correia e Thales Miranda de Almeida Vieira.
Dissertação (mestrado em Informática) - Universidade Federal de Alagoas. Instituto de Computação. Maceió, 2018.

Bibliografia: f. 40-43.

1. Processamento de imagens. 2. Processo estocástico. 3. Máquina de Vetores de Suporte. I. Título.

CDU: 004.383.5

UNIVERSIDADE FEDERAL DE ALAGOAS/UFAL
Programa de Pós-Graduação em Informática – PpgI
Instituto de Computação
Campus A. C. Simões BR 104-Norte Km 14 BL 12 Tabuleiro do Martins
Maceió/AL - Brasil CEP: 57.072-970 | Telefone: (082) 3214-1401

Membros da Comissão Julgadora da Dissertação de Artur Maia Pereira, intitulada: *"Uma Abordagem Estocástica para Recuperação de Imagens de Roupa Baseada em Conceito Utilizando o SVM e o Feedback de Relevância do Usuário"*, apresentada ao Programa de Pós-Graduação em Informática da Universidade Federal de Alagoas em 13 de março de 2019, às 14h45, na sala 15, do Instituto de Computação da UFAL.

COMISSÃO JULGADORA

Prof. Dr. Evandro de Barros Costa
UFAL – Instituto de Computação
Orientador

Prof. Dr. Thales Miranda de Almeida Vieira
UFAL – Instituto de Computação
Orientador

Prof. Dr. Marcelo Costa Oliveira
UFAL – Instituto de Computação
Examinador Interno

Prof. Dr. Patrick Henrique da Silva Brito
UFAL – Instituto de Computação
Examinador Interno

Prof. Dra. Patrícia Cabral de Azevedo Restelli Tedesco
UFPE – Centro de Informática
Examinadora Externa

Resumo

Sistemas de recuperação de imagem estão se tornando cada vez mais comum em aplicações online do ramo da moda, permitindo que os consumidores busquem por itens de roupa. Entretanto, explorar um grande conjunto de imagem através de métodos simples de recuperação é geralmente ineficiente e cansativo para usuários que estão buscando por novidades. Quando o *feedback* do usuário é considerado, algoritmos de aprendizagem de máquina podem tirar vantagem das preferências do usuário e aprimorar a experiência de compra. Apesar disso, a maioria das abordagens que consideram *feedback* focam em recuperar apenas imagens relevantes para o usuário, sem se preocupar com a curva de aprendizado da máquina, resultando em uma falta de diversidade. Em particular, quando o usuário acessa o sistema pela primeira vez, não há nenhuma informação sobre ele, levando a recomendações insatisfatórias (problema da partida fria). Neste trabalho, nós propomos uma abordagem de aprendizado de máquina baseada na técnica de *Relevance Feedback* (RF) para recuperar imagens de roupa utilizando a seguinte estratégia: Recuperar imagens relevantes para o usuário; recuperar imagens que não vão de acordo com as preferências do usuário, para evitar a convergência para um mínimo local; e recuperar imagens da região de incerteza para melhorar a curva de aprendizado, todas essas três formas de maneira estocástica. Para contornar a problema da partida fria, nós apresentamos um novo método de seleção que visa melhorar a diversidade das imagens recuperadas combinando um método de projeção multidimensional com uma estrutura de dados espacial adaptável. Nossa abordagem foi validada através de experimentos qualitativos e quantitativos com usuários utilizando uma base de dados anotada de imagens de roupas femininas com o objetivo de avaliar a efetividade e eficiência da recuperação de imagens relevantes. Resultados mostraram que a abordagem proposta pode melhorar rapidamente a recuperação de imagens apropriadas com apenas algumas iterações do usuário, enquanto oferece uma maior diversidade em relação a outras abordagens.

Abstract

Image retrieval systems have become a common approach in online fashion applications, allowing consumers to search for clothing items. However, the task of exploring large data sets of images through naive retrieval methods is generally inefficient and tedious for users. When relevance feedback (RF) from the user is considered, machine learning methods may take advantage of user preferences to enhance the experience. Nevertheless, most RF approaches focus on retrieving only user relevant images, disregarding the learning curve of the machine and resulting in a lack of diversity. In particular, when a new user begins using the system, there is a lack of information about him, resulting in unsatisfactory recommendations (the cold start problem). We propose a machine learning approach based on RF to retrieve clothing images using a threefold strategy: retrieve user relevant images; retrieve images that do not comply with user learned preferences, to avoid convergence to local minimal; and retrieve images from uncertainty regions, to improve the learning curve, all in a stochastic manner. To mitigate the cold start problem, we present a novel sampling method to improve the diversity of retrieved images that employs a combination of a multidimensional projection method with an adaptive spatial data structure. Our approach is validated through quantitative and qualitative user experiments using an annotated clothing images data set to evaluate the effectiveness and efficiency of user-relevant image retrieval. Results revealed that our approach can rapidly improve the retrieval of appropriate images in a few user iterations while providing higher diversity than straightforward approaches.

Keywords: concept-based retrieval, relevance feedback, dimensionality reduction, stochastic algorithm, fashion, clothing retrieval

Agradecimentos

Primeiramente, gostaria de agradecer aos meus pais, por estarem sempre ao meu lado, contribuindo na minha educação e garantindo tudo que eu sempre precisei.

Aos meu orientadores, Evandro e Thales, que embarcaram comigo neste projeto e guiaram todo seu desenvolvimento durante um longo ano sempre com muita paciência e dedicação.

Agradeço aos meus familiares, principalmente tios e tias, que sempre me apoiaram durante meu caminho acadêmico. A Jesus, que sempre me aceitou em sua casa, e a todos os amigos que sempre tiveram do meu lado, em especial os que me acompanham desde o colégio. Por fim, queria agradecer a todos os professores e servidores do instituto de computação que dedicam esforços diários em prol do alunos.

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

Atualmente, com o desenvolvimento crescente das tecnologias e da Internet, diferentes sistemas de informação passaram a ocupar cada vez mais espaço na sociedade, resolvendo problemas de alta complexidade e relevância pertencentes a diferentes contextos, sendo praticamente impossível que uma pessoa viva sem interagir com qualquer tipo de sistema de informação. Seguindo esta tendência, temos as atividades referentes ao comércio eletrônico (e-commerce) que continuam em expansão em quase todas as áreas de negócio de bens e serviços. Um exemplo disso é o comércio online de produtos relacionados à moda[1] que trouxe uma série de benefícios para o consumidor. Uma grande vantagem é a oferta de um catálogo de itens extenso e variado onde, em alguns casos, a quantidade de diferentes produtos disponíveis pode ultrapassar a casa dos milhares e tudo isso ao alcance de alguns cliques. Porém, examinar uma quantidade massiva de produtos para encontrar um item que esteja de acordo com suas preferências pode se tornar uma tarefa exaustiva, consumindo muito tempo do usuário. Apesar das lojas online disponibilizarem mecanismos de busca baseados em palavras-chaves ou categorias, muitas vezes o cliente chega na loja apenas com uma ideia geral do que deseja comprar, como uma blusa para ir a uma festa, sem se preocupar muito com detalhes, deixando de lado características referentes ao design da roupa (manga, gola, tecido, etc). Com isso, se o usuário não for muito específico, o mecanismo de busca do site irá retornar um resultado muito amplo e, caso o sistema de filtros seja muito genérico (preço, tamanho, marca, etc.), a filtragem do resultado inicial acabará não sendo útil. Pode ocorrer também o caso em que o usuário deixa claro as características da roupa que deseja, porém o mecanismo de busca não consegue entender a nomenclatura utilizada, retornando produtos insignificantes, requerendo um certo investimento em processamento de linguagem natural, como no estudo proposto por Marcelino *et al.* (2018).

O cenário de problema anteriormente mencionado pode fazer com que o consumidor gaste muito tempo procurando uma peça de roupa que combine com seu estilo e, caso não obtenha sucesso em tempo útil, ele pode acabar desistindo da compra, partindo para outra loja ou bus-

[1] https://www.statista.com/statistics/379046/worldwide-retail-e-commerce-sales/. Acessado em: 01/2019

cando outra forma de compra mais ágil. Para mitigar este problema, foram propostos Sistemas de Recomendação (Ricci *et al.* (2010)) com o objetivo de melhorar a experiência do usuário através da recuperação de imagem que considera seu *feedback* a respeito de um conjunto de roupas apresentadas, por exemplo, o trabalho apresentado por Li *et al.* (2016). Entretanto, tais sistemas geralmente sofrem do problema da partida fria, isto é, quando um novo usuário começa a interagir com o sistema, suas preferências são desconhecidas e um critério padrão como a seleção aleatória pode ser empregada para montar a primeira galeria de imagens (Silva *et al.* (2010)). Em particular, quando uma grande base de dados está sendo explorada, as chances de recuperar imagens relevantes na primeira iteração é pequena. Além disso, imagens são geralmente recuperadas usando apenas as preferências do usuário como critério, ignorando a curva de aprendizado da máquina.

Analisando alguns sites de comércio eletrônico[2], podemos perceber que a maioria deles utilizam a técnica de filtragem colaborativa para fazer recomendações baseando-se na similaridade do histórico de itens visualizados ou comprados pelos usuários em geral, por exemplo, caso eu compre o produto X, o sistema retornará como recomendação outros itens que receberam uma boa classificação ou que foram adquiridos por usuários que também demonstraram preferência pelo produto X. Tal abordagem revela-se pouco eficiente, pois desconsidera os gostos específicos e pessoais de cada usuário. Além disso, em tese, a cada navegação o usuário irá buscar peças de roupas para ocasiões diferentes, ou até mesmo itens para terceiros. Sendo assim, considerar o histórico de navegação do usuário pode não ser muito útil, visto que ele nem sempre está interessado em roupas com as mesmas características.

Para contornar estes problemas visto no domínio de moda, surge a questão central deste projeto, que é recuperar imagens de peças de roupa, através de um processo iterativo, tendo como base com as preferências específicas de cada usuário em relação às características das roupas. Partindo do princípio de que não há informação a priori do usuário, as primeiras imagens serão selecionadas visando exibir ao usuário todas as características reconhecidas pelo sistema (tipo de manga, tipo de gola, variações de estampa, diferentes tecidos, etc.), de modo que, no início da navegação ele já tenha acesso a uma grande variedade de escolha.

1.1 Objetivo

O trabalho proposto apresenta uma diferente abordagem de recomendação de peças de roupa seguindo a linha de recuperação de imagem baseado em conceito que considera o *feedback* do usuário para aprender suas preferências. Afim de montar a galeria de imagens que será exibida inicialmente, utilizaremos uma abordagem que tenta garantir uma galeria diversificada, evitando selecionar várias imagens com a mesma característica, para que o usuário possa ter uma ampla opção de escolha e não fique preso em uma região local do catálogo de imagens. Tal abordagem

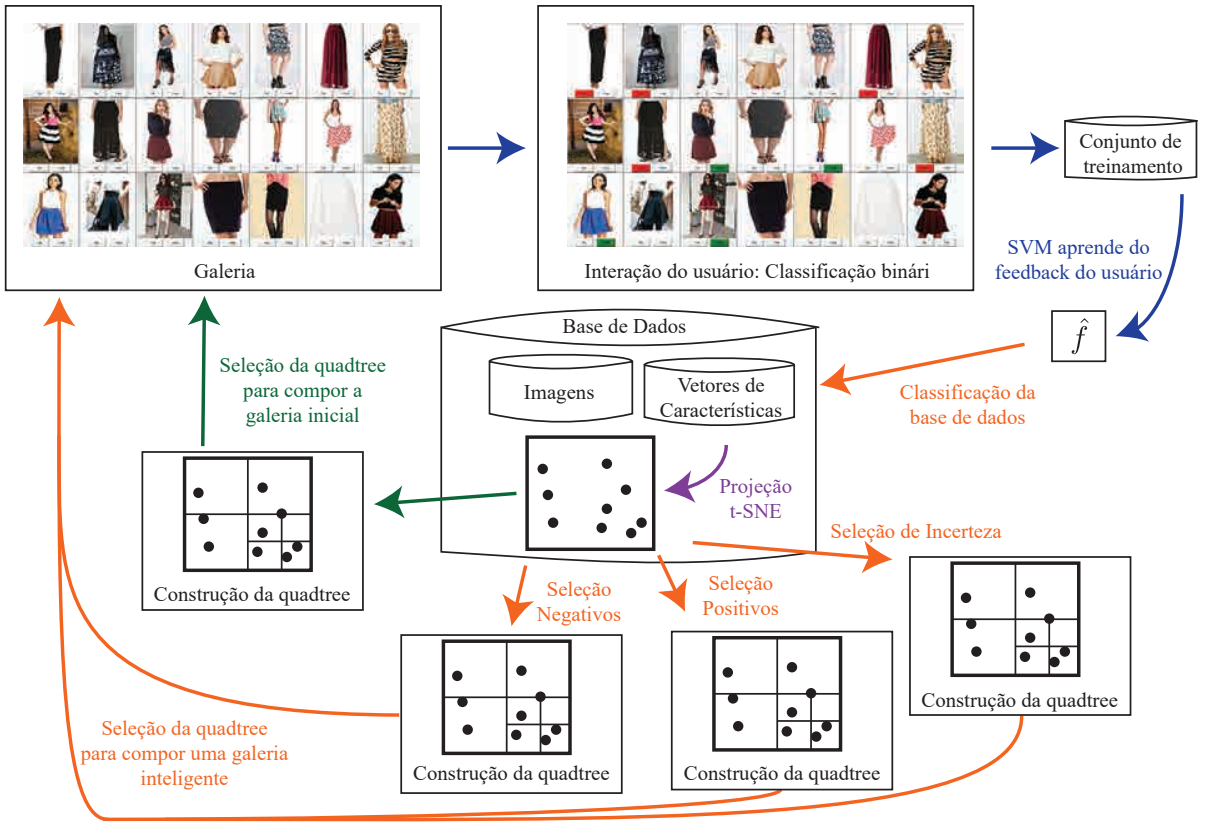
[2] https://www.forever21.com/, https://www.cea.com.br/, https://www.cea.com.br/. Acessado em: 03/2019

combina o método de projeção multidimensional t-SNE Van der Maaten & Hinton (2008) com a estrutura de dados Quadtree Samet (1990) para contornar o problema da partida fria. Montada a galeria inicial, o usuário irá classificar as imagens como relevantes ou irrelevantes, resultando no conjunto de treinamento inicial.

Após a fase inicial de classificação, o sistema verificará as principais características das roupas presentes nas imagens e, em seguida, utilizando um classificador SVM (support vector machine), o sistema irá montar uma nova galeria de imagens de acordo com as escolhas do usuário. O SVM foi escolhido porque, a partir da sua classificação, é possível ter uma melhor percepção do quão relevante ou irrelevante é a imagem, considerando a distância algébrica da imagem para o hiperplano do SVM. Tal percepção de distância não seria possível em classificadores baseados em árvore de decisão. Vale ressaltar que a abordagem proposta segue a ideia de coleta iterativa de *feedback*, ou seja, para cada conjunto de imagens retornado, o usuário pode continuar interagindo com o sistema, classificando mais imagens para aperfeiçoar o mecanismo de busca, até que ele fique satisfeito com as recomendações. Como o objetivo destas galerias subsequentes é coletar o *feedback* do usuário para aprimorar o aprendizado do sistema, tais galerias serão montadas com imagens pertencentes a três diferentes grupos para evitar que o indivíduo fique preso em uma sub-região do conjunto total de imagens. Os três grupos são: Imagens que o sistema classificou como relevantes, com o objetivo de recomendar itens de acordo com as preferências; Imagens classificadas como irrelevantes, com o objetivo de evitar uma convergência do processo iterativo para uma pequena região (mínimo local); e imagens da região de incerteza do classificador, para aperfeiçoá-lo. Então, ao fim da etapa de classificação, finalizando o processo iterativo de *feedback*, caso o usuário deseje, o sistema formará a última galeria contendo as recomendações para o usuário, considerando apenas os itens classificados pelo SVM como relevantes.

A partir da Figura 1.1 é possível ter uma compreensão mais precisa do ciclo de funcionamento do sistema. O usuário marca as imagens presentes na galeria inicial, em seguida, o sistema aprenda suas preferências e recupere outras similares da base de dados levando em consideração os vetores (conjuntos) de características já extraídos. Após a classificação do SVM, um nova galeria será formada com imagens relevantes, irrelevantes e da região de incerteza, dando início ao processo iterativo de coleta de *feedback*.

Figura 1.1: Visão geral do modelo proposto.



Fonte: Autoria Própria

1.2 Estrutura do Trabalho

Os próximos capítulos deste documento de dissertação estão estruturados da seguinte maneira:

- No capítulo 3, serão apresentados os estudos relacionados ao proposto nesta dissertação, destacando o que já foi feito e os resultados obtidos, para fins de comparação e complementação.

- O capítulo 2 traz uma fundamentação teórica a respeito dos conceitos e algoritmos que embasam esta pesquisa. Sendo assim, serão apresentados conceitos sobre o modelo *Content-Based Image Retrieval* (CBIR) e o processo iterativo de *Relevance Feedback* (RF), o algoritmo de aprendizado de máquina SVM (*support vector machine*), a estrutura de dados Quadtree e a técnica de redução de dimensionalidade t-SNE (*T-distributed Stochastic Neighbor Embedding*).

- No capítulo 4 será descrito, passo a passo, a metodologia do protótipo desenvolvido onde o usuário irá interagir e receber as recomendações.

- O capítulo 5 detalha os experimentos realizados e os resultados obtidos através de simulações e testes com usuários, trazendo uma discussão sobre eles.
- Por fim, o capítulo 6 aborda as considerações finais sobre a pesquisa desenvolvida e as possibilidades de trabalhos futuros.

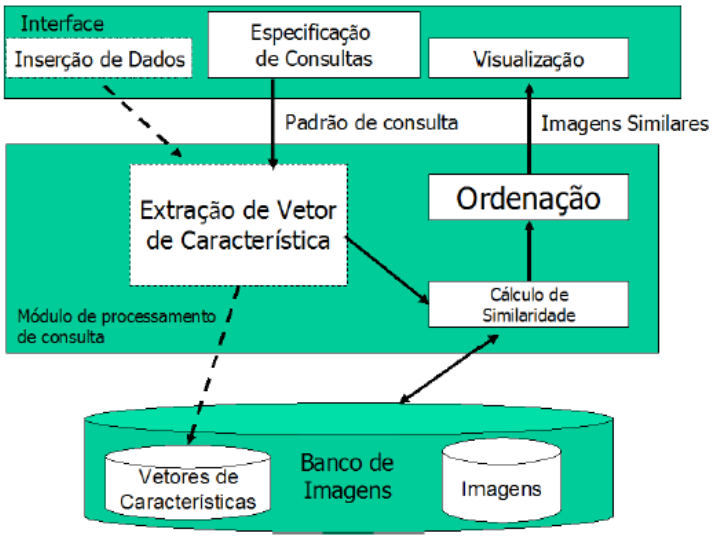
Capítulo 2

Fundamentação Teórica

2.1 *Content-Based Image Retrieval* (CBIR)

A técnica de visão computacional *content-based image retrieval* (recuperação de imagem baseada em conteúdo) pode ser descrita como um processo de recuperação de imagens desejadas a partir de uma grande base de dados, levando em consideração descrições textuais ou certas características (cores, texturas, formas, etc.) que podem ser extraídas automaticamente das próprias imagens (Eakins *et al.* (1999)).

Figura 2.1: Arquitetura do modelo de recuperação de imagens baseada em conteúdo.



Fonte: Torres & Falcão (2006).

A Figura 2.1 apresenta a arquitetura da abordagem CBIR que, segundo Torres & Falcão (2006), pode ser dividida em dois módulos principais. Um deles é o módulo de inserção de dados, representados na imagem pelas linhas tracejadas, responsável por extrair o vetor de características das imagens e armazená-lo em uma base de dados que será utilizada pelo módulo de processamento de consulta. Inicialmente, o usuário vai especificar uma consulta por meio

de um padrão (foto, desenho, texto, etc.) que será processada e terá seu vetor características extraído. Em seguida, será aplicada alguma métrica, como distância euclidiana, para avaliar a similaridade (proximidade) entre a consulta do usuário e as imagens da base. Por fim, o módulo retorna para o usuário as imagens mais similares à sua consulta inicial.

Segundo Smeulders *et al.* (2000), os sistemas que seguem a linha de CBIR podem ser divididos em três classes que variam de acordo com o objetivo do usuário. São elas:

- **Busca por associação:** Refere-se a sistemas que geralmente possuem um processo iterativo de refinamento de busca através de imagens ou esboços, pois o usuário não possui nenhum objetivo inicial específico, ele está apenas buscando novidades interessantes.
- **Busca direcionada por exemplo:** Apresenta uma solução que recebe como entrada uma imagem específica e busca cópias ou imagens muito similares em um determinado catálogo. Esse tipo de sistema pode ser aplicado, por exemplo, para busca de estampas ou artes.
- **Busca por categoria:** Visa buscar imagens arbitrárias de uma mesma categoria. Se encaixa no caso em que o usuário tem um exemplo e está buscando outros elementos da mesma classe.

Ao longo das últimas três décadas, várias pesquisas foram desenvolvidas, tanto na academia, quanto na indústria, visando modelos CBIR que consideram apenas imagens como dados de entrada. O estudo feito por Veltkamp & Tanase (2000) apresenta uma breve revisão sobre vários sistemas de CBIR aplicados a problemas reais. A partir delas, alguns problemas foram levantados em relação a como interpretar as imagens e o que considerar como entrada para as consultas ao sistema. Com isso, dois desafios considerados cruciais foram identificados, conhecidos como, *intention gap* e *semantic gap* (Zhou *et al.* (2017)). O *intention gap* refere-se a dificuldade que o usuário tem em expressar precisamente qual seu interesse e o que espera que seja retornado pelo sistema, dadas as limitações da consulta, visto que uma imagem pode apresentar várias características. Já o *semantic gap*, que recebe uma maior atenção das pesquisas, diz respeito as diferentes interpretações de uma mesma imagem: de um lado temos as informações que o sistema consegue extrair e do outro lado as características provenientes da visualização do usuário, tendo um forte impacto no resultado final.

2.1.1 *Relevance feedback* (RF)

Grande parte dos sistemas CBIR apresentam uma característica conhecida como *Relevance feedback* (*feedback* de relevância), no qual o usuário e o sistema interagem, de forma controlada, para ajustar sua consulta inicial à base de imagens, pois considera-se que a maioria dos usuários possuem dificuldades para formular consultas que atendem de imediato o seu propósito. Um cenário típico do processo de RF pode ser descrito nos 3 seguintes passos (Zhou & S. Huang (2003)):

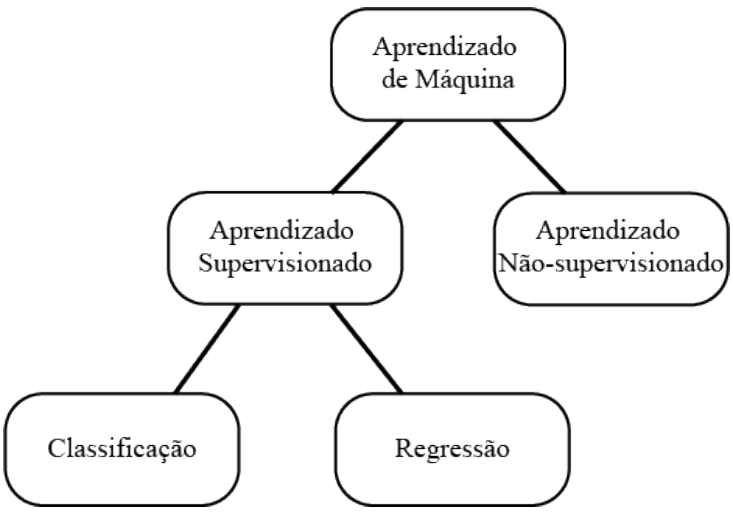
- **Passo 1:** Feita a consulta inicial do usuário, o conjunto de imagens similares retornado servirá como base para o restante do processo.
- **Passo 2:** Dadas as imagens retornadas, o usuário irá classificá-las como relevantes ou irrelevantes levando em consideração o que espera receber do sistema.
- **Passo 3:** As imagens classificadas pelo usuário (*feedback*) serão usadas para alimentar o mecanismo de busca, que passará a ter mais informações a respeito das preferências do usuário, e então, o sistema irá formar um novo conjunto de imagens resultante, voltando ao segundo passo do processo.

O processo iterativo de RF continua até que o usuário esteja satisfeito com o resultado retornado ou até ele atingir um número máximo de repetições, porém, para este caso de coleta de *feedback* explícito, o ideal é chegar a um resultado satisfatório com o menor número de iterações possíveis, evitando incomodar muito o usuário. Esta etapa, geralmente, faz uso de um algoritmo de aprendizagem de máquina.

2.2 Aprendizagem de Máquina

Aprendizagem de máquina é uma área da inteligência artificial que busca desenvolver técnicas computacionais que aprendem a executar uma tarefa e, através de um processo contínuo, melhoram automaticamente seu desempenho por meio de experiências passadas (Mitchell (1997)). Para tal, aplicam um princípio de inferência denominado indução, no qual se obtém conclusões a partir de um conjunto particular de exemplos, ou seja, dado um determinado problema, um algoritmo de aprendizado de máquina vai receber como entrada dados que representam problemas semelhantes já solucionados e vai induzir uma hipótese a respeito da solução do problema proposto (Faceli *et al.* (2011)).

Figura 2.2: Tipos de aprendizado de máquina.



Como podemos ver na Figura 2.2, os algoritmos de aprendizado de máquina podem ser divididos em duas categorias. O conjunto dos algoritmos supervisionados, que simula a presença de um "supervisor externo", tem como objetivo executar uma tarefa de previsão, isto é, o foco é encontrar um modelo que possa ser utilizado para prever um rótulo ou valor que caracterize um novo exemplo, com base nos valores dos atributos de entrada. A aprendizagem supervisionada, possui duas subclasses que diferem pelo tipo de dados que se usa. No caso de regressão, tenta-se prever o resultado através de uma saída contínua. Por outro lado, nos problemas de classificação procura-se um resultado através de uma variável discreta, mapeando-a em categorias (Faceli *et al.* (2011)).

A segunda categoria engloba algoritmos cujo principal objetivo é explorar ou descrever um conjunto de dados, seguindo o paradigma de aprendizado não supervisionado. Como exemplo, podemos citar as tarefas de agrupamento de dados que buscam formar grupos de objetos similares dentro de uma base de dados, ou as tarefas de associação que relacionam atributos da base para tentar identificar padrões (Faceli *et al.* (2011)).

2.2.1 Máquina de Vetores de Suporte (SVM)

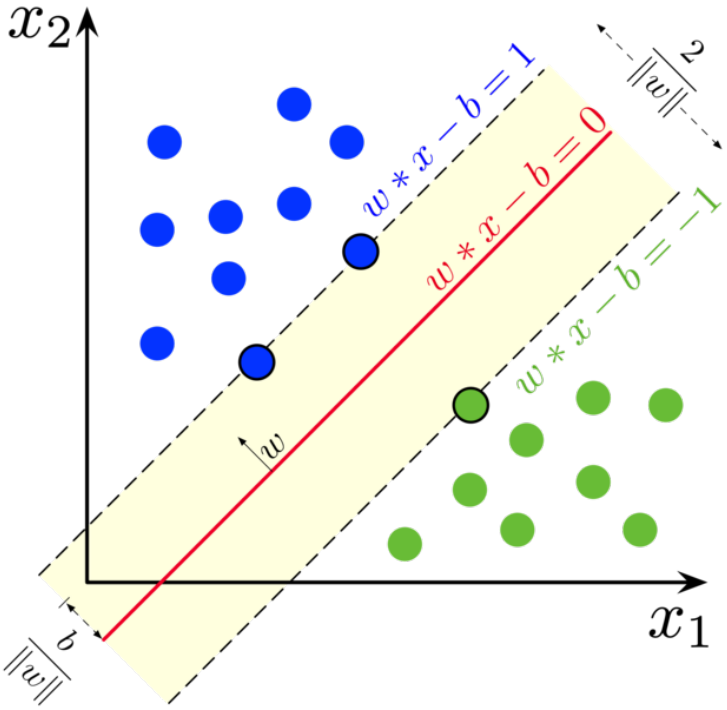
Introduzido nos estudos elaborados por Cortes & Vapnik (1995), o SVM pode ser descrito como um algoritmo que segue o modelo de aprendizado supervisionado e é utilizado para análise de dados e reconhecimento de padrões, tanto para o caso de classificação, quanto para regressão (Shawe-Taylor & Cristianini (2004)). Dado um conjunto de exemplos classificados de acordo com duas classes, o SVM define um modelo que irá atribuir uma classe a novos exemplos, isto é, o SVM tenta encontrar um hiperplano, em um espaço n-dimensional, maximizando a margem de separação entre os elementos das classes. Com isso, a previsão da classe de novos casos depende da região do espaço delimitada pelo hiperplano, à qual o exemplo pertence.

A abordagem descrita anteriormente é conhecida como SVM linear. Em uma definição mais formal, seja X um conjunto de treinamento de objetos x_i e seus respectivos rótulos $y_i \in Y$, tal que $Y = \{+1, -1\}$ são as possíveis classes. X é linearmente separável se for possível separar os objetos das classes $+1$ e -1 por um hiperplano descrito pela Equação 2.1.

$$h(x) = w \cdot x + b, \tag{2.1}$$

onde $w \cdot x$ é o produto escalar entre os vetores w e x, w é o vetor normal ao hiperplano e $\frac{b}{\|w\|}$ corresponde ao deslocamento do hiperplano a partir da origem ao longo do vetor normal, com $b \in \mathbb{R}$

Figura 2.3: Exemplo de classificação com SVM linear.



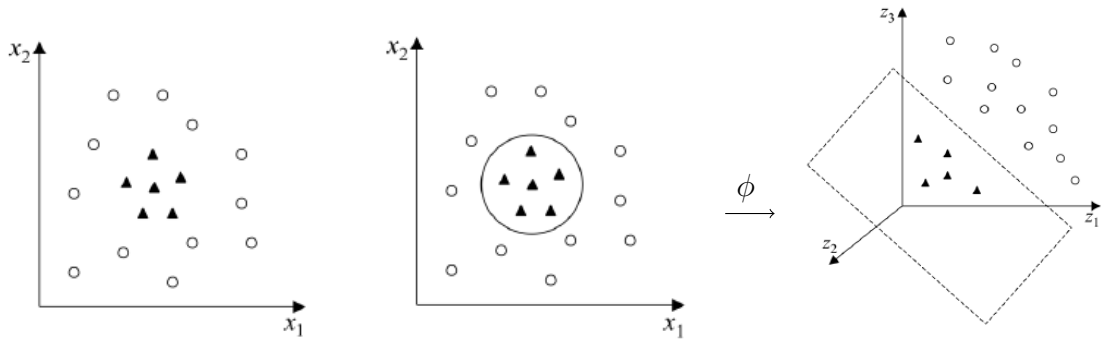
Fonte: https://commons.wikimedia.org/wiki/File:SVM_margin.png. Acessado em: 12/2018.

A Figura 2.3 apresenta o caso mais simples do SVM linear, sem a presença de ruídos ou valores atípicos. Nela podemos ver que a Equação 2.1 foi usada para dividir o espaço de entrada X em duas regiões, $w \cdot x + b > 0$ e $w \cdot x + b < 0$, resultando na função $g(x)$ que será usada para obter a classificação de novos exemplos, conforme descrita na Equação 2.2

$$g(x) = \begin{cases} +1 & \text{se } w \cdot x + b > 0 \\ -1 & \text{se } w \cdot x + b < 0 \end{cases} \tag{2.2}$$

O algoritmo original do SVM mostrou ser muito eficaz na classificação de conjuntos de dados que possuem uma distribuição aproximadamente linear. Porém, nem todos os casos reais são linearmente separáveis, sendo assim, Boser *et al.* (1992) sugeriu uma maneira de realizar uma classificação não linear com o SVM. Para isso é feito um mapeamento não linear do conjunto de dados do espaço original, para um novo espaço de maior dimensão, chamado de espaço de característica, tal que a nova distribuição dos dados possibilite que as duas classes sejam separadas por um hiperplano, como podemos ver na Figura 2.4

Figura 2.4: Transformação dos dados com o SVM não linear para um espaço de característica.



(a) Conjunto de dados (b) Separação das duas classes (c) Espaço de característica separado por um hiperplano.

Fonte: Faceli *et al.* (2011).

Esta transformação entre espaços, na prática, é implícita, ou seja, a função ϕ não é calculada explicitamente. Em vez disso, substituímos o produto escalar $w \cdot x$ por uma função *kernel* K que recebe dois pontos x_i e x_j na origem de entrada e calcula o produto escalar desses objetos no espaço de características (Faceli *et al.* (2011)). Cada função apresenta alguns parâmetros que afetam diretamente o desempenho do classificador, pois eles irão definir a fronteira de decisão, ficando a cargo do usuário definir o valor de cada parâmetro. Como exemplo, podemos citar o caso da função de *kernel* polinomial, onde o usuário precisa definir qual o grau do polinômio que será usado. Duas populares funções de *kernel* são:

- **Polinomial:** $K(x_i, x_j) = (x_i \cdot x_j)^d$, onde $d > 0$ representa o grau do polinômio.
- **Radial:** $K(x_i, x_j) = \exp(-\gamma \|x_i - x_j\|^2)$, para $\gamma > 0$, onde $\|x_i - x_j\|^2$ representa a distância Euclidiana entre dois vetores.

2.3 Quadtree

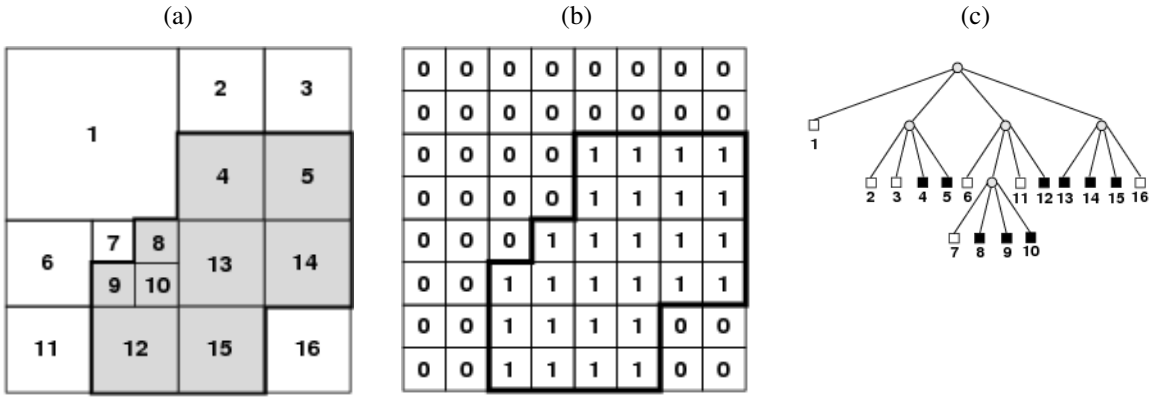
Proposta inicialmente por Finkel & Bentley (1974), a Quadtree é uma estrutura de dados em árvore baseada na decomposição recursiva do espaço de solução bidimensional, muito utilizada na área de computação gráfica, apresentando um comportamento similar ao método de dividir e conquistar. Criada uma região retangular inicial (nó raiz), ela será dividida em quatro quadrantes (sub-regiões), obedecendo o padrão de que cada nó interno da árvore terá sempre quatro nós filho. A partir desta divisão inicial, cada nó interno continuará sendo dividido em quatro quadrantes enquanto o nó possuir uma quantidade de dados superior a um limiar de parada definido.

A Quadtree pode ser classificada de acordo com os dados que estão sendo representados, como áreas, pontos, retas ou curvas, e é comumente utilizada para processamento e representação de imagens, detecção de colisões em duas dimensões, simulação de dinâmica de fluidos, entre outros. Os principais tipos de quadtree serão descritos a seguir.

2.3.1 Quadtree de Região

Representa o exemplo mais simples e consiste em processar os dados (*pixels*) de uma imagem em preto e branco através de regiões, por exemplo, um quadtree de profundidade n pode ser utilizado para armazenar uma imagem de $2^n * 2^n$ pixels. Neste caso, o nó raiz representará a imagem inteira, se os pixels de uma sub-região não for apenas 0s ou 1s, isto é, todos os pixels preto ou branco, ela é dividida novamente, até que se chegue a um nó folha com blocos de pixels somente com 0s ou 1s. Sendo assim, a quadtree de região pode ser considerada uma estrutura de dados de resolução de imagem (Samet (1988)). Considere a região representada em blocos na Figura 2.5a que pode ser descrita através de um array binário de $2^3 * 2^3$ (Figura 2.5b), onde 1 corresponde aos elementos dentro da região e 0 aos elementos de fora. Resultando na árvore de profundidade 4 apresentada na Figura 2.5c

Figura 2.5: Exemplo de imagem binária representada com a quadtree de região. (a) Decomposição em blocos. (b) Array binário. (c) Árvore da quadtree.

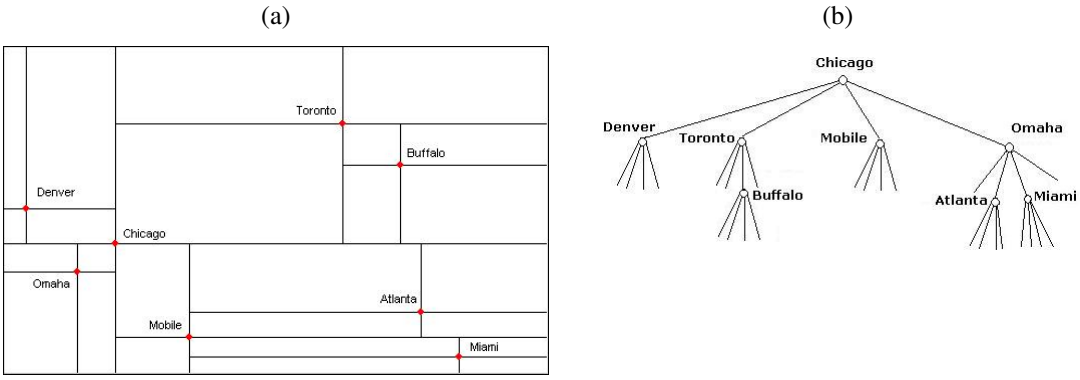


Fonte: Samet (1988).

2.3.2 Quadtree de Pontos

A quadtree de pontos pode ser definida como uma adaptação de uma árvore binária usada para armazenar dados bidimensionais, onde o centro da divisão é sempre um ponto. Cada nó é representado por basicamente 2 informações: um ponteiro para cada um de seus quatro nós filhos (quadrantes), e as coordenadas x e y, podendo armazenar outros atributos, caso seja necessário (Finkel & Bentley (1974)). A Figura 2.6b apresenta uma árvore gerada pela quadtree a partir dos pontos descrito na Figura 2.6a.

Figura 2.6: Exemplo de pontos armazenados em uma quadtree de pontos. (a) Pontos em um plano cartesiano. (b) Pontos armazenados em uma árvore.



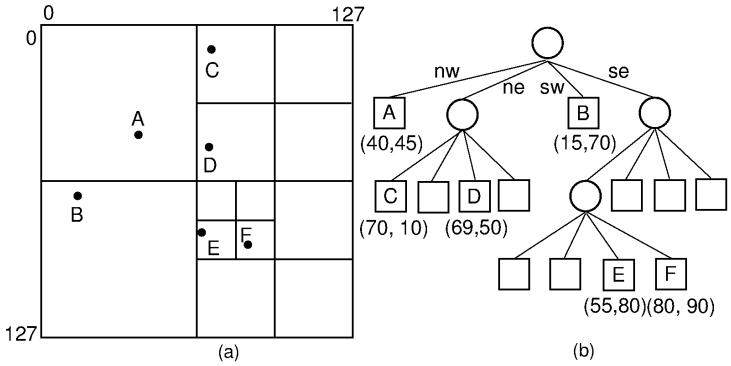
Fonte: Finkel & Bentley (1974).

Para inserir um novo ponto, é feita uma comparação no nó raiz para saber em qual quadrante (nó filho) o ponto deve ser inserido. Caso o quadrante já esteja ocupado, a busca continua até encontrar uma folha. O quadrante representado por esta folha será subdividido e o ponto é inserido no local apropriado. A capacidade de cada quadrante é de apenas um ponto.

2.3.3 Quadtree Ponto-Região (PR)

Este tipo de quadtree é similar ao de região, porém a grande diferença está no tipo de informação armazenada pelo quadrante. No quadtree de região, o valor armazenado se refere a toda a área do quadrante. Já na quadtree PR, guarda-se uma lista com os pontos que pertencem ao quadrante em questão. Dois métodos podem ser usados como critério de parada da divisão recursiva dos quadrantes: definir um número máximo de pontos pertencentes a região; ou estabelecer um limite no nível de profundidade da árvore (D'Angelo (2016)). A Figura 2.7 apresenta a árvore gerada com a aplicação da quadtree PR em uma determinada distribuição de pontos, na qual cada quadrante só pode conter nó máximo um ponto. Neste trabalho utilizaremos este tipo de quadtree, pois a seleção das imagens que serão exibidas para o usuário terá como base a distância entre as regiões, visando selecionar imagens da maior quantidade de quadrantes diferentes.

Figura 2.7: Exemplo da quadtree ponto-região.(a) Pontos do plano dividido em regiões. (b) Árvore gerada a partir da quadtree.



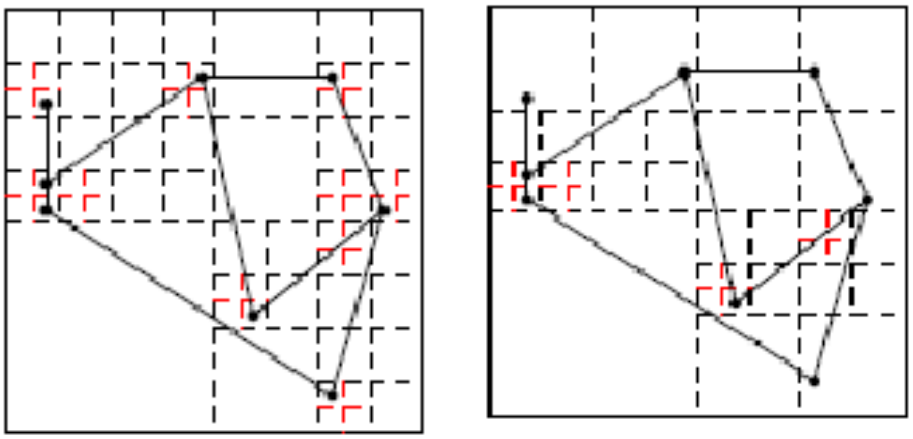
Fonte: https://opendsa-server.cs.vt.edu/ODSA/Books/Everything/html/PRquadtree.html. Acessado em: 12/2018.

2.3.4 Quadtree De Aresta e de Mapa Poligonal (MP)

Tanto a quadtree de arestas quanto a MP são utilizadas para armazenar linhas em vez de pontos. Sendo assim, o espaço é subdividido até que reste um segmento de linha por quadrante. A grande diferença entre estes dois tipos é que nos locais próximos dos vértices, a quadtree de aresta continuará dividindo o quadrante até atingir o nível máximo de profundidade da árvore pré estabelecido (Shneier (1981)), enquanto que na quadtree MP o quadrante em consideração não é subdividido se um vértice for encontrado(Samet & Webber (1985)). Na Figura 2.8, podemos ver claramente a diferença entre a divisão dos quadrantes de um mesmo polígono usando as duas abordagens.

Figura 2.8: Diferença da divisão de quadrantes entre (a) a quadtree de aresta e (b) a quadtree de mapa poligonal.

(a) Aresta (b) Mapa Poligonal



Fonte: D'Angelo (2016).

Como foi citado no início da seção 2.3, a quadtree opera apenas em espaço bidimensionais. Porém, é possível desenvolver variações que trabalham com mais dimensões, por exemplo, a octree (Samet (1990)) que opera em espaços tridimensionais. No entanto, quanto maior o número de dimensões, mais demorado será o processo de decomposição recursiva do espaço, sendo necessário realizar 2^n divisões, onde n corresponde ao número de dimensões. Sendo assim, quando se trabalha com base de dados complexas, com muitas dimensões, e não se tem um poder de processamento muito alto, como no nosso caso, é mais vantajoso aplicar alguma técnica de redução de dimensionalidade como o PCA (Jolliffe (1986)) ou o t-SNE (Van der Maaten & Hinton (2008)), do que adaptar a quadtree para um espaço de n dimensões.

2.4 *T-distributed Stochastic Neighbor Embedding* (t-SNE)

Bases de dados complexas e com um grande número de atributos costumam ser representadas em espaços de alta dimensionalidade. Para o computador, não há problema em processar muitas dimensões, porém nós humanos somos limitados a visualizar no máximo 2 dimensões, surgindo a necessidade de se desenvolver algoritmos para auxiliar a visualização dos dados, preservando as propriedades dos atributos originais (Oliveira & Levkowitz (2003)). O t-SNE, desenvolvido por Van der Maaten & Hinton (2008), é uma técnica de redução de dimensionalidade de dados não linear, bem difundida na área de aprendizagem de máquina, baseada em distribuições de probabilidade e nas relações entre nós vizinhos, que recebe como entrada dados com alta dimensionalidade e retorna sua projeção bidimensional buscando preservar a distância entre os pontos mais próximos (similares).

O primeiro passo do t-SNE é converter a distância euclidiana entre os pontos vizinhos em uma probabilidade condicional de similaridade, de acordo com sua posição no espaço de solução. Para o espaço original, supondo que seja selecionado um ponto x_i, então a probabilidade de um ponto x_j ser seu vizinho é definida pela distribuição de probabilidade $p_{i|j}$, representada na Equação 2.3

$$p_{i|j} = \frac{\exp\left(\frac{-\|x_i - x_j\|^2}{2\sigma_i^2}\right)}{\sum_{k \neq i} \exp\left(\frac{-\|x_i - x_k\|^2}{2\sigma_i^2}\right)}, \tag{2.3}$$

tal que σ_i representa a variância de uma Gaussiana centrada em x_i. Por outro lado, para mapear x_i em um espaço bidimensional, representado por y_i, Van der Maaten & Hinton (2008) propôs a utilização da distribuição t de Student(STUDENT (1908)), apresentada na Equação 2.4

$$q_{i|j} = \frac{(1 + (\|y_i - y_j\|^2))^{-1}}{\sum_{k \neq i}(1 + \|y_i - y_k\|^2)^{-1}}. \tag{2.4}$$

.

A partir disto, tenta-se minimizar a diferença entre as distribuições de probabilidade através da divergência de Kullback-Leibler com o método de gradiente simétrico SNE (Van der Maaten & Hinton (2008)) onde, quanto mais próxima de 0 for a diferença entre as distribuições, mais semelhantes serão os comportamentos, enquanto que, o resultado da divergência próximo de 1 indica que as distribuições se comportam de maneira diferente.

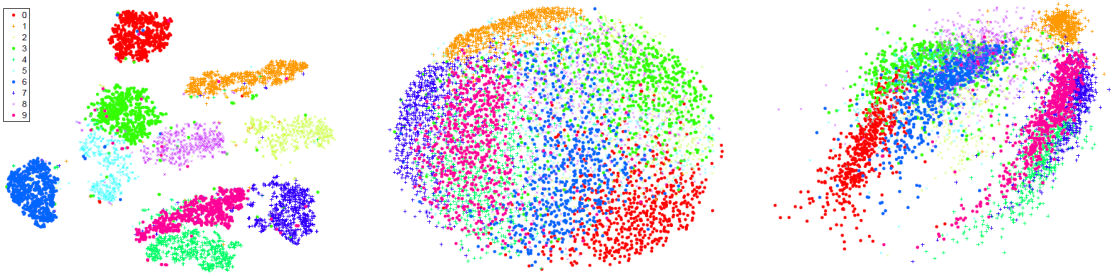
Para fins de avaliação, Van der Maaten & Hinton (2008) realizaram experimentos para comparar a redução de dimensionalidade do t-SNE com outras 7 técnicas, focando na visualização dos dados, ou seja, casos com 2 ou 3 dimensões, constatando que o t-SNE apresentou uma representação bidimensional com a melhor separação entre os *clusters* de pontos em bases com classificação multiclasse, como podemos ver na Figura 2.9 que utiliza a base de dados de dígitos MNIST[1].

Figura 2.9: Visualização bidimensional de dados após aplicação de técnicas de redução de dimensionalidade.

(a) Visualização pelo t-SNE.

(b) Visualização pelo Sammon *mapping*.

(c) Visualização pelo Isomap.



Fonte: Van der Maaten & Hinton (2008)

2.5 Considerações Finais

As técnicas e conceitos apresentados neste capítulo servirão como base para o desenvolvimento do modelo proposto, que segue a ideia de um sistema de recuperação de imagem baseada em conteúdo, através da aplicação de um algoritmo de aprendizagem de máquina.

Tendo em vista o objetivo de recuperar imagens diversificadas, explorando toda a base de dados, escolhemos como solução utilizar a estrutura de árvore da quadtree com o auxílio da técnica de projeção multidimensional t-SNE para adaptar os dados de acordo com as necessidades da quadtree.

[1]Base de dados pública disponível em: http://yann.lecun.com/exdb/mnist/

Capítulo 3

Trabalhos Relacionados

A seguir serão apresentados um conjunto de trabalhos que fundamentam ou possuem uma certa relação com o modelo apresentado. Analisar tais trabalhos é importante para se destacar a relevância do nosso projeto no cenário dos sistemas de recomendação que utilizam imagens. Embora o nosso método seja considerado como uma abordagem baseada em conceito, onde as imagens são representadas por anotações, vamos destacar os trabalhos que envolvem a abordagem de recuperação de imagem baseada em conteúdo (CBIR), uma vez que seu processo é mais semelhante ao modelo proposto.

3.1 CBIR e Modelos Estocásticos

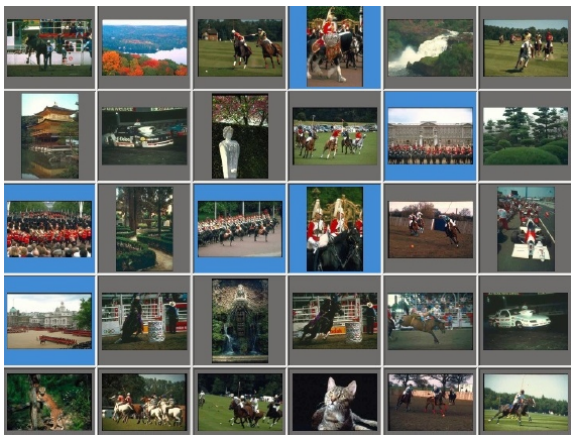
Em relação aos estudos já desenvolvidos na área de recuperação de imagem baseada em conteúdo (CBIR), temos como base o trabalho desenvolvido por Silva *et al.* (2010) que utiliza a abordagem CBIR junto com a coleta iterativa de *feedback* do usuário, com o objetivo de retornar apenas imagens que são relevantes de acordo com suas escolhas. No entanto, diferente do nosso caso, são utilizadas imagens de propósito geral, sem focar em um tema específico, levando em consideração apenas atributos relacionados a cores e formas. Em nosso trabalho, utilizamos o conjunto de imagens elaborado por Liu *et al.* (2016), descrito na seção 5.1, que apresenta atributos de alto nível previamente anotados, representando características de itens de roupa como estampa, tecido, tipo de manga, etc. Para realizar a recuperação, dado um determinado conjunto de imagens selecionadas como relevante pelo usuário, são utilizados três algoritmos para fazer a classificação do restante das imagens que poderão compor a galeria seguinte, são eles: O *Optimum-Path Forest* (OPF), desenvolvido pelos autores; o *Support Vector Machine* (SVM) e o *Query Point Movement* (QPM) que, segundo os autores, estão bem consolidados como técnicas de classificação de imagens (Tong & Chang (2001)). No fim, foi constatado que o algoritmo OPF apresentou os resultados mais efetivos, com um menor número de iterações.

A Figura 3.1a apresenta a interface do protótipo desenvolvido por Silva *et al.* (2010). Nela podemos ver o conjunto de imagens apresentado inicialmente para o usuário onde estão selecionadas as imagens que ele considerou como relevantes, enquanto que a Figura 3.1b mostra o conjunto final de imagens, formado com o resultado da classificação do algoritmo OPF e retornado após 3 iterações do usuário, número este considerado ideal pelos autores do trabalho no caso de situações práticas.

O modelo apresentado por Yildizer *et al.* (2012) trabalha também com imagens de propósito geral, divididas em diferentes categorias, considerando características extraídas através da técnica de transformada de *wavelets* de Daubechies, realizando a recuperação de imagens similares em duas etapas. Dada uma imagem de entrada, a primeira etapa utilizará o algoritmo SVM multi-classe para identificar qual categoria ela tem maior probabilidade de pertencer. Definida a categoria da imagem, o próximo passo vai calcular a distância euclidiana entre ela e todas as outras da mesma classe, e então as imagens com menor distância serão retornadas como similares.

Figura 3.1: Galerias de imagens do protótipo desenvolvido por Silva *et al.* (2010). (a) Apresenta a galeria inicial exibida para o usuário. (b) Apresenta a galeria final com os resultados do OPF depois de 3 iterações do usuário.

(a) Galeria inicial



(b) Galeria após 3 iterações

Fonte: Silva *et al.* (2010).

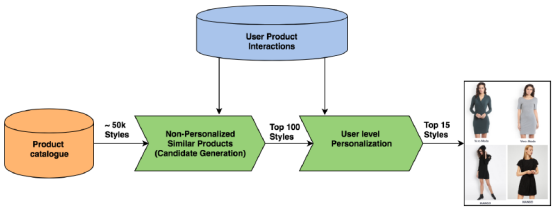
Para garantir a recuperação de imagens diversificadas existe a possibilidade de combinar a técnica de *feedback* relevante com modelos estocásticas, como a abordagem clássica com o uso do algoritmo *particle swarm optimization* (PSO) (Kennedy *et al.* (2001)), para fazer com que o usuário explore bem o espaço de solução, além de evitar que o mecanismo de seleção de imagens similares termine estagnado em uma região local sub-ótima da base, atribuindo a cada imagem uma probabilidade de ser escolhida ao invés de selecionar deterministicamente as mais similares. Como exemplos, podemos citar o estudo proposto por Broilo & Natale (2010) que utiliza o PSO para realizar a recuperação de imagens relevantes de propósito geral, obtendo resultados melhores que abordagens determinísticas que derivam de um método beaseado na

equação de Rocchio (Rijsbergen (1979). Outro trabalho relevante foi proposto por Saffawi *et al.* (2014) que combina o PSO com método de *clustering* K-means, apresentando melhores resultados tanto quando comparado com abordagens determinísticas, como a elaborada por Afifi & Ashour (2012), quanto em relação a outra abordagem estocástica, como o método desenvolvido por Jhanwar *et al.* (2004). Porém, em nenhum destes métodos há a preocupação em aumentar a diversidade dos itens selecionados. Em nosso método, utilizamos uma estrutura de dados espacial Quadtree com este objetivo.

3.2 Sistemas de Recomendação de Roupa

Entrando na área de sistemas de recomendação, já foram desenvolvidos alguns protótipos no ramo da moda, porém eles apresentam certas diferenças em relação ao modelo proposto nesta dissertação. O modelo desenvolvido por Agarwal *et al.* (2018) utiliza dados da plataforma *Myntra*[1] e propõe um modelo de recomendação personalizada, exibido na Figura 3.2, que combina o grau de similaridade de duas abordagens de acordo com os produtos visualizados pelo usuário. O primeiro modelo visa encontrar itens similares através da aplicação de uma técnica de filtragem colaborativa item-item, ao passo que a segunda abordagem utiliza uma medida de similaridade usuário-item para tentar identificar atributos das roupas que o usuário possa estar interessado, preservando o contexto da busca do produto e do gosto do usuário. Este cenário com o uso de filtragem colaborativa parte do princípio de que o perfil do usuário já está disponível, ou seja, que o sistema possui acesso a um histórico de escolha de todos os usuários que acessaram o sistema. Porém, este *feedback* implícito (histórico de navegação) do usuário pode levar a um conclusão insatisfatória, porque o fato do usuário apenas visualizar um item não significa diretamente que ele gostou. Além disso, é preciso também considerar que o histórico tem um curto tempo de vida, pois pode ocorrer do usuário não ter mais interesse em recomendações baseadas em itens que ele visualizou a meses atrás. Já o modelo proposto neste trabalho considera que o usuário está acessando o sistema pela primeira vez, iniciando o processo de recomendação sem nenhuma informação a priori dele, propondo uma solução para o problema da partida fria (Lika *et al.* (2014)).

Figura 3.2: Modelo de recomendação com personalização e filtragem colaborativa proposto por Agarwal *et al.* (2018)

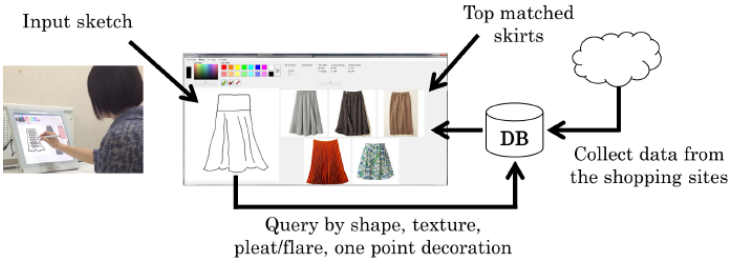


Fonte: Agarwal *et al.* (2018).

[1] https://www.myntra.com/

Os trabalhos propostos por Liu *et al.* (2012) e Kiapour *et al.* (2015) oferecem um modelo de recomendação de peças de roupa que aborda um problema conhecido como *street-to-shop*, onde o sistema recebe como entrada do usuário uma foto de uma roupa tirada no dia a dia em qualquer pose e ambiente, retornando para ele, com o auxílio de técnicas de aprendizagem profunda, peças iguais ou similares encontradas em um catálogo de roupas. Já o modelo desenvolvido por Kondo *et al.* (2014), apresentado na Figura 3.3, requer que o usuário consiga expressar o modelo de saia que deseja através de um rascunho que será utilizado pelo sistema para recuperar imagens de modelos semelhantes ao desenho inicial. Nestes casos, o usuário já possui uma ideia bem definida a respeito do que deseja encontrar, o sistema vai apenas ajuda-lo na busca pelo item similar, sem apresentar nada de novo que possa despertar o interesse do usuário. Sendo assim, o foco principal destes sistemas é extrair características da roupa que está na foto ou rascunho para comparar com as imagens do catálogo disponível. Por outro lado, nosso modelo interage com o usuário, de forma repetitiva, para tentar descobrir suas preferências e encontrar peças que o agradem.

Figura 3.3: Arquitetura do modelo CBIR proposto por Kondo *et al.* (2014)



Fonte: Kondo *et al.* (2014).

Mais relacionado com o nosso trabalho, Li *et al.* (2016) desenvolveram um protótipo de recomendação de roupas superiores que leva em consideração características referentes ao colarinho das blusas que, segundo o estudo, é geralmente a parte da roupa que chama mais atenção das pessoas devido à posição de perspectiva horizontal próximo ao olho do observador. Para isso, são extraídos três elementos importantes do formato do colarinho, são eles: a sua forma de abertura, seu volume (existência de dobras) e o estilo frontal do colarinho (presença de fitas, laços ou outros detalhes). Inicialmente, o protótipo apresenta para o usuário dez imagens correspondentes aos dez diferentes tipos de colarinhos presentes na base de imagens, para que ele escolha as que mais se adequam ao seu gosto. A partir desta primeira escolha, o sistema utiliza o algoritmo de classificação Optimum-Path Forest (OPF) para encontrar imagens com características semelhantes às marcadas pelo usuário como relevantes. Este processo de classificação por parte do usuário é repetido até que ele fique satisfeito com o conjunto de imagens apresentado pelo sistema. A Figura 3.4 apresenta alguns resultados práticos obtidos com este protótipo. Porém, diferente do nosso trabalho, este caso desconsidera caraterísticas de outras regiões da roupa, limitando a interpretação da imagem apenas ao colarinho. Isto pode levar o

usuário a uma certa confusão, por exemplo, o usuário pode gostar do formato do colarinho, e por outro lado não gostar do tecido da roupa, levando a um conflito de interesse. Para o nosso caso estamos considerando uma base de imagens, detalhada na seção 5.1, que está anotada com atributos de alto nível de todas regiões da roupa.

Figura 3.4: Galerias de imagens do protótipo desenvolvido por Li *et al.* (2016) com o resultado das duas primeiras iterações do usuário

Fonte: Li *et al.* (2016).

3.3 Considerações Finais

No geral, todos os modelos de recomendação de roupa apresentados focam em recuperar apenas em aprender o que o usuário considerou relevante, evitando as classificações irrelevantes. Esta abordagem tende a guiar o usuário para uma região sub-ótima do espaço de solução, fazendo com que ele receba sempre mais do mesmo, abrindo mão de arriscar apresentar diferentes modeles que não foram apresentados inicialmente. Por outro lado, o nosso modelo, descrito no próximo capítulo, visa explorar ao máximo o espaço de solução, recuperando imagens de acordo com o *feedback* do usuário, mas buscando exibir sempre roupas com características diversificadas, arriscando as novidades, ao invés de recuperar sempre as imagens mais similares ao conjunto de relevantes.

Capítulo 4

Modelo Proposto

O modelo desenvolvido neste trabalho pode ser divido em 3 fases sequenciais. A primeira fase engloba a montagem de uma galeria de imagens que será apresentada inicialmente para o usuário. Em seguida, vem a fase de interação com usuário, entrando em um processo iterativo de coleta do seu *feedback*. Por fim, temos a seleção do conjunto de imagens que serão recomendadas para o usuário de acordo com suas escolhas anteriores. O funcionamento dessas 3 fases será detalhado a seguir, esclarecendo o uso das técnicas apresentadas no Capítulo 2.

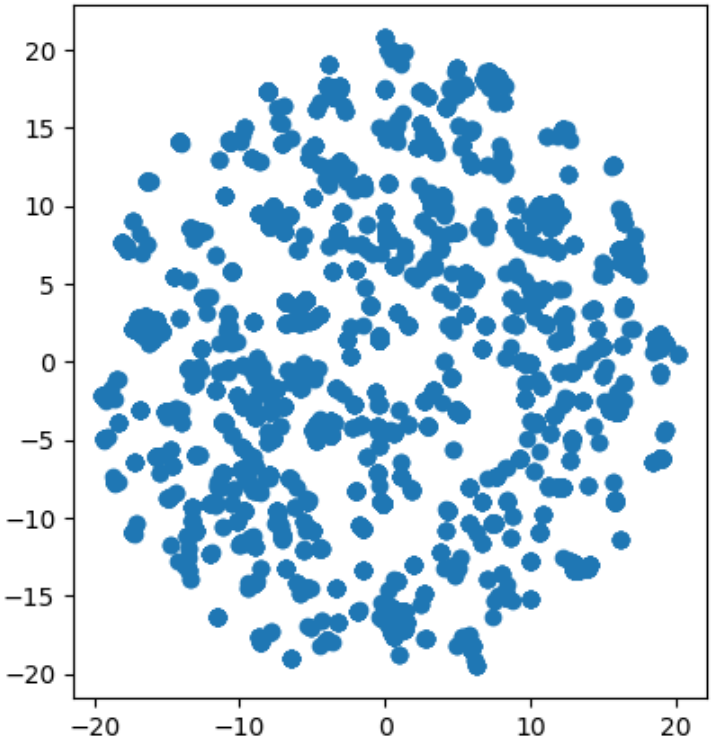
4.1 Montagem da Galeria Inicial

Inicialmente, fixamos empiricamente a quantidade ótima de 21 imagens por galeria, considerando as dimensões das imagens em relação a tela de de um *notebook* médio utilizado. O primeiro desafio encontrado neste trabalho foi em relação à como selecionar de um pequeno conjunto de imagens para compor a galeria inicial que será apresentada ao usuário, a partir de uma base de dados com mais de milhares de exemplos. Como a base de imagens de roupas utilizada neste projeto não possui nenhuma informação a priori sobre o perfil do usuário, nem um histórico de itens visualizados ou comprados por ele, foi preciso definir uma maneira de selecionar as imagens apenas com base nas anotações das imagens que representam os atributos das roupas. Em uma definição mais formal do espaço de solução, seja $\mathcal{D} = \{x_1, x_2, \dots\}$ um conjunto de itens de roupa, representados pela instância $x_i = (v_i, I_i)$, onde $v_i \in \mathbb{R}^n$ é um vetor de característica que representa cada atributo do item, I_i corresponde a representação visual (imagem), e $\mathcal{V} = \{v_1, v_2, \dots\}$ representa o conjunto de vetores característica de $\mathcal{D}$. No total, as anotações utilizadas possuíam 17 atributos binários (seguindo a abordagem *one-hot encoding*), selecionados através de um refinamento explicado na seção 5.1.

Para garantir a diversidade da galeria inicial, a seleção das imagens foi feita por um algoritmo baseado na estrutura quadtree ponto-região, apresentado na seção 2.3.3. Considerando que as anotações de cada imagem correspondem a um ponto em um espaço de 17 dimensões, antes de aplicar a quadtree foi necessário projetar os pontos em um espaço bidimensional, uti-

lizando a técnica t-SNE, visto que a quadtree é uma estrutura restrita a pontos bidimensionais, isto é, o primeiro passo foi computar a projeção multidimensional de $\tilde{v}_i \in \mathbb{R}^2$ para todos os vetores característica em $\mathcal{D}$. É interessante destacar que generalizações da quadtree para dimensões mais altas, como a octree, poderiam ser usadas. Porém, na prática, a quantidade de regiões do espaço após algumas poucas subdivisões inviabilizaria a aplicação. A Figura 4.1 apresenta o resultado da projeção das anotações utilizadas, de acordo com a técnica de projeção multidimensional t-SNE.

Figura 4.1: Projeção das anotações utilizadas (Liu *et al.* (2016)) em um plano bidimensional através do t-SNE.



Fonte: Autoria Própria.

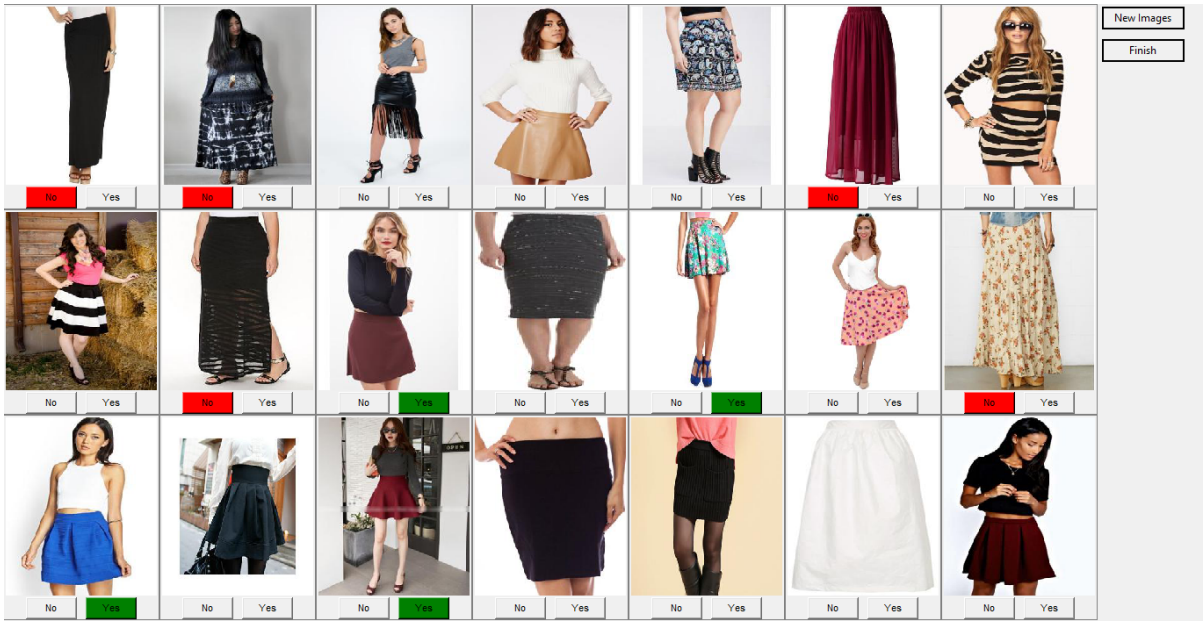
Analisando a estrutura da árvore gerada pela quadtree, cada nó vai ser responsável por armazenar a quantidade de imagens da região representada pelo nó. De modo geral, seja n o total de pontos (imagens) que serão selecionadas num determinado quadrante, o algoritmo de busca deve selecionar aproximadamente $\frac{n}{4}$ pontos de cada subregião (quadrante) do quadrante atual, para garantir diversidade das imagens que serão exibidas para o usuário. O Algoritmo 1 esclarece como a quadtree foi construída para armazenar os pontos resultantes da projeção do t-SNE. Cada região será dividida até que a árvore atinja o limiar da quantidade de pontos (menor ou igual a 6 pontos), ou o limite de profundidade, estabelecendo uma árvore com um bom número de nós (no máximo 4^{10}) sem prejuízo crítico ao desempenho do protótipo (linha 3). No fim, as 4 sub-regiões geradas serão cadastradas como filhos da região originária.

Construída a estrutura do quadtree, a seleção das imagens obedece às instruções do Algoritmo 2. Iniciando no nó raiz, sendo n a quantidade de imagens requeridas de uma região, a busca recursiva nos quadrantes para quando é alcançada umas das 3 condições (linhas 5, 7 e 9), descritas a seguir:

- **1º Condição:** Se a região não tiver nenhum nó filho, indicando que ela é um nó folha, será selecionada desta região uma amostra correspondente ao mínimo entre o valor de n e a quantidade de pontos da região.
- **2º Condição:** Se a quantidade de pontos da região for menor que n, todos os pontos da região serão selecionados.
- **3º Condição:** Tendo em vista que a quadtree seleciona $\frac{n}{4}$ de cada quadrante, então, se $n < 4$, não faz sentido continuar descendo na busca em árvore. Sendo assim, o algoritmo vai selecionar n amostras da região em questão.

A Figura 4.2 apresenta a interface do protótipo com a galeria inicial formada através da seleção de imagens com a quadtree. Para fazer a coleta do *feedback*, estamos utilizando uma escala binária (sim e não) para saber o que o usuário achou da peça de roupa apresentada. Vale ressaltar que não é necessário que o usuário classifique todas as imagens da galeria. Porém, para a galeria inicial, é preciso que o usuário classifique no mínimo 4 imagens como relevante e outras 4 imagens como irrelevante, gerando um conjunto de treinamento mínimo para que o sistema possa aprender.

Figura 4.2: Galeria inicial de imagens apresentada ao usuário.



Fonte: Autoria própria.

Algorithm 1 Pseudocódigo: Construir Quadtree

```
1:  nivel ← 0
2:  function CONSTQUADTREE(reg, nivel)                    ▷ reg contém pontos
3:      if reg.tamanho() ≤ limPontos or nivel ≥ limProf then
4:          return
5:      else
6:          listReg ← dividirReg(reg)                     ▷ dividir em 4 quadrantes
7:
8:          R1 ← listReg[0]
9:          constQuadtree(R1, nivel + 1)
10:         R2 ← listReg[1]
11:         constQuadtree(R2, nivel + 1)
12:         R3 ← listReg[2]
13:         constQuadtree(R3, nivel + 1)
14:         R4 ← listReg[3]
15:         constQuadtree(R4, nivel + 1)
16:
17:         reg.setFilhos(R1, R2, R3, R4)
18:         return
19:     end if
20: end function
```

Algorithm 2 Pseudocódigo: Seleção com o Quadtree

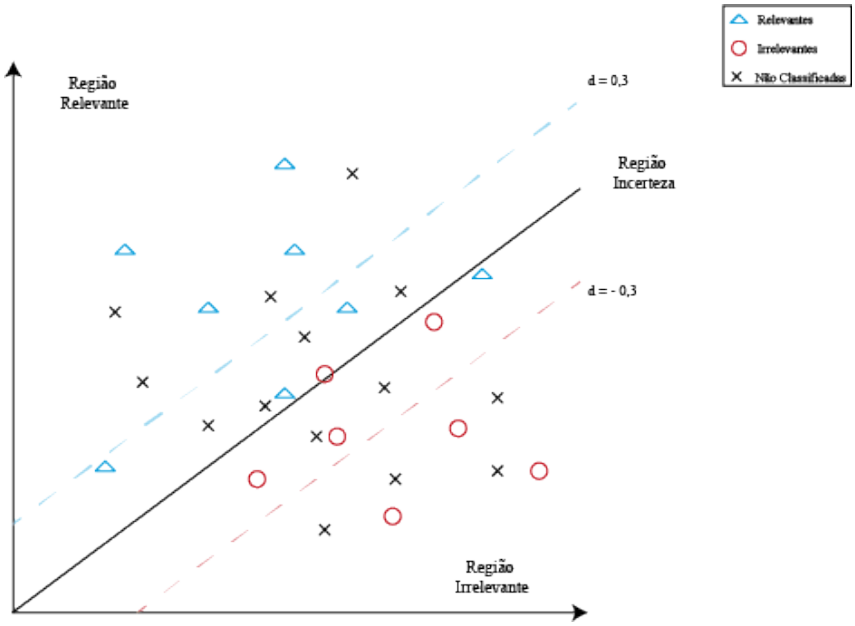
1: **function** SELECQUADTREE(reg, n) ▷ serão selecionados n pontos
2: $x = []$ ▷ lista com os pontos selecionados
3:
4: **if** $reg.getFilhos() == 0$ **then**
5: $x \leftarrow x + amostra(reg.getPts, \min(n, reg.tamanho())$
6: **else if** $reg.tamanho() \leq n$ **then**
7: $x \leftarrow x + reg.getPts()$
8: **else if** $n < 4$ **then**
9: $x \leftarrow x + amostra(reg.getPts(), n)$
10: **else**
11: $listReg \leftarrow reg.getFilhos()$
12: $listReg \leftarrow$ ordenarCresc($listReg$) ▷ ordenar pela qtd. de pontos
13: $faltam \leftarrow n$
14:
15: $busca \leftarrow \frac{faltam}{4}$
16: $R1 \leftarrow listReg[0]$
17: $x \leftarrow x+$ selecQuadtree($R1, \min(busca, R1.tamanho())$)
18: $faltam = faltam - \min(busca, R1.tamanho())$
19:
20: $busca \leftarrow \frac{faltam}{3}$
21: $R2 \leftarrow listReg[1]$
22: $x \leftarrow x+$ selecQuadtree($R2, \min(busca, R2.tamanho())$)
23: $faltam = faltam - \min(busca, R2.tamanho())$
24:
25: $busca \leftarrow \frac{faltam}{2}$
26: $R3 \leftarrow listReg[2]$
27: $x \leftarrow x+$ selecQuadtree($R3, \min(busca, R3.tamanho())$)
28: $faltam = faltam - \min(busca, R3.tamanho())$
29:
30: $R4 \leftarrow listReg[3]$
31: $x \leftarrow x+$ selecQuadtree($R4, \min(busca, R4.tamanho())$)
32: **end if**
33: **return** x
34: **end function**

4.2 Processo Iterativo de *Feedback* - Galerias intermediárias

O conjunto de vetores característica das imagens classificadas pelo usuário na galeria inicial serão as primeiras informações que teremos a respeito das suas preferências, formando o conjunto de treinamento do classificador SVM para classificar o restante das imagens da base de dados, considerando. Combinando o resultado do SVM com uma abordagem estocástica e a quadtree, uma nova galeria de imagens será formada, dando início ao processo iterativo de coleta de *feedback* do usuário, cujo objetivo é aumentar o conjunto de treino, aprimorando cada vez mais a classificação do SVM. Este processo se repete até que o usuário esteja satisfeito ou que se atinja um número máximo de 5 repetições, tendo como base os experimentos realizados em Silva *et al.* (2010).

Baseado no conjunto de treino formado pelas imagens classificadas pelo usuário, o SVM vai tentar encontrar um hiperplano que melhor separe os objetos das duas classes em questão. Sendo assim, a classificação de novas imagens vai depender da sua distância algébrica para o hiperplano, ou seja, considerando o hiperplano como a origem do espaço, todas as imagens não classificadas que estiverem no eixo positivo do espaço serão consideradas relevantes, enquanto que as imagens que estiverem no eixo negativo serão consideradas como irrelevantes para o usuário.

Figura 4.3: Exemplo da aplicação do SVM nas imagens classificadas pelo usuário dividindo o espaço em 3 regiões.



Fonte: Autoria própria.

As galerias intermediárias serão formadas por imagens provenientes de 3 diferentes regiões divididas de acordo com o resultado do SVM, como podemos ver na Figura 4.3. A primeira região será composta por imagens consideradas relevantes (eixo positivo) que apresentam uma distância normalizada maior que 0,3 em relação ao hiperplano. A segunda região engloba as imagens classificadas como irrelevantes (eixo negativo) que apresentam uma distância para o hiperplano menor que -0,3. A última região contém as imagens que estão muito próximas do hiperplano, podendo ser chamada de região de incerteza, pois, apesar das imagens receberem uma classificação, elas estão no limite da divisão entre as duas classes. Em tese, apresentar uma galeria com imagens destes 3 conjuntos fará com que o conjunto de treino aumente tanto a quantidade de imagens relevantes, quanto a quantidade de irrelevantes, aprimorando o aprendizado sobre o que o usuário gosta e o que ele não gosta, criando uma boa separação entre os elementos de cada classe.

Para montar a primeira galeria intermediária, será selecionada uma quantidade igual de imagens de cada uma das 3 regiões (relevante, irrelevante, incerteza). Porém, usando o *feedback* do usuário, as galerias subsequentes terão um comportamento mais dinâmico, variando de acordo com a quantidade de erros e acertos da previsão do SVM, seguindo o princípio de uma matriz de confusão. Exemplos da região relevante que o usuário gostou (verdadeiro positivo) e da região irrelevante que o usuário não gostou (verdadeiro negativo) serão contabilizados como acertos. Casos da região relevante que sejam classificadas como irrelevante (falso positivo) e da região irrelevante classificados como relevantes (falso negativo) serão contabilizados como erros. Os exemplos da região de incerteza não influenciarão nos acertos e erros, sendo contabilizados independente da classificação. Dados esses 3 contadores, a próxima galeria intermediária poderá apresentar uma das seguintes mudanças:

- **1º Caso:** Se forem contabilizados mais acertos do que erros e incertezas, a próxima galeria apresentará um exemplo relevante a mais.
- **2º Caso:** Se os erros forem maioria em relação aos outros dois contadores, a galeria seguinte terá um exemplo a mais da região irrelevante.
- **3º Caso:** Se o número de imagens classificadas pelo usuário da região de incerteza for maior que erros e acertos, a próxima galeria contará com mais uma imagem da região de incerteza.

Para cada acréscimo de uma imagem a região com o contador de o maior valor, em consequência, será removido um exemplo da região que apresentar a menor contagem, mantendo sempre a galeria exibida ao usuário com um total de 21 imagens. Como podemos ver na Figura 4.2 cada galeria formada possui 3 linhas de imagens. Sendo assim, para a primeira galeria formada com o auxílio do *feedback* do usuário, teremos que: A primeira linha de imagens vai apresentar os exemplos considerados relevantes; a segunda linha representará as imagens da

região de incerteza; a terceira linha será composta pelas imagens da região classificada como irrelevante.

Para selecionar as imagens que irão compor cada uma dessas galerias intermediárias, primeiro é atribuída a cada imagem uma probabilidade de ser sorteada, de modo que, para as regiões relevante e irrelevante, quanto mais longe do hiperplano, maior a probabilidade dela ser sorteada, e para a região de incerteza, quanto mais próximo de hiperplano do SVM, maior a probabilidade de sorteio. A Equação 4.1 esclarece como foi calculada a probabilidade (p_j) de cada imagem, onde $distSVM_j$ representa a distancia algébrica do ponto j para o hiperplano.

$$p_j = \begin{cases} \frac{distSVM_j}{\sum_i^n distSVM_i} & \text{se } distSVM_j > 0,3 \\ \frac{0,3-|distSVM_j|}{\sum_i^n 0,3-|distSVM_i|} & \text{se } -0,3 \leq distSVM_j \leq 0,3 \\ \frac{|distSVM_j|}{\sum_i^n |distSVM_i|} & \text{se } distSVM_j < -0,3 \end{cases} \tag{4.1}$$

Definida a probabilidade de cada imagem, serão sorteadas algumas imagens elegíveis para compor a galeria, formando o conjunto X_{prob}. Seja n a quantidade de imagens necessárias de uma região para preencher a galeria, o conjunto X_{prob} terá um total de $10 * n$ imagens, ou seja, caso a galeria precise de 7 exemplos da região relevante, a priori, serão selecionadas 70 imagens da região. Para isso foi usado o método de amostragem de transformação inversa. Seja x_j uma imagem com probabilidade p_j de ser sorteada, e U um número gerado aleatoriamente, uniformemente distribuído no intervalo [0,1], a seleção da variável aleatória X será dada pela Equação 4.2

$$X = \begin{cases} x_0 & \text{se } U < p_0 \\ x_1 & \text{se } p_0 \leq U < p_0 + p_1 \\ \vdots & \\ x_j & \text{se } \sum_{i=0}^{j-1} p_i \leq U < \sum_{i=0}^{j} p_i \\ \vdots & \end{cases} \tag{4.2}$$

Formado o conjunto de imagens X_{prob} de cada região, seus elementos serão projetados em um espaço bidimensional, usando o t-SNE, e então, os n exemplares que irão de fato compor a galeria serão selecionados através do algoritmo do quadtree ponto-região, coletando $\frac{n}{4}$ imagens de cada quadrante do espaço de solução, sempre com o objetivo de garantir a diversidade das peças apresentadas ao usuário.

4.3 Recomendação Final

A última fase do modelo vai apresentar uma galeria de imagens com as recomendações para o usuário. Após o usuário atingir o limite máximo de 5 iterações, tendo como base os experimentos realizados por Silva *et al.* (2010), ou se sentir satisfeito com a quantidade de imagens

classificadas, o processo de coleta do *feedback* será finalizado. Em seguida, o sistema vai aplicar o mesmo princípio usado na montagem das galerias intermediárias, ou seja, considerando os exemplos da região relevante que estão a uma distância euclidiana do classificador SVM superior a 0.5, serão selecionadas 210 imagens através do método de amostragem de transformação inversa e, em seguida, será aplicada a quadtree para selecionar as 21 imagens da galeria de recomendação.

Tendo como base a galeria inicial apresentada na Figura 4.2 realizamos uma simulação onde são selecionadas como relevantes as saias curtas e irrelevantes as saias longas. Vale ressaltar que foram escolhidos apenas dois atributos (curto e longo) por se tratar de uma simulação. Em casos de execução com usuários reais, ele pode classificar as imagens da maneira que desejar, cabendo ao sistema analisar as anotações referentes as características das roupas e buscar peças que corresponde ao seu *feedback* de maneira geral, independente da combinação de atributos. A Figura 4.4 apresenta galeria final com as recomendações após 5 rodadas de coleta de *feedback*, tendo cerca de 100 exemplos classificados. Nela podemos ver que para cada imagem há uma escala de 5 níveis para o usuário classifica-lá. Este último *feedback* servirá apenas para avaliação do sistema durante os testes, permitindo averiguar se as recomendações realmente agradaram o usuário e se houve um aumento na quantidade de imagens relevantes comparando a primeira e a última galeria.

Figura 4.4: Galeria de imagens com as roupas recomendadas para o usuário apresentada após a finalização do processo iterativo, seguindo seu *feedback*.

Fonte: Autoria própria.

Capítulo 5

Experimentos e Discussão

Para validar o modelo proposto, foi realizada uma série de testes, tanto simulados, quanto com usuários reais, visando deixar claro que o método de seleção de imagens desenvolvido apresenta galerias com imagens diversificadas, garantindo ao usuário uma ampla variedade de escolha e um conjunto de recomendação satisfatório. Vale ressaltar que, para todos os testes (simulados e reais), o hiperplano do SVM foi elaborado através da função de *kernel* radial com $\gamma = 0,05$. O valor de γ foi definido através de uma série de testes de precisão com a técnica de validação cruzada utilizando um conjunto com cerca de 100 instâncias classificadas.

5.1 Base de Imagens

Para realizar os experimentos, nós utilizamos a base de imagens DeepFashion (Liu *et al.* (2016)) de domínio público e que contém cerca de 300.000 imagens de roupas provenientes de duas diferentes fontes: Uma parte foi proveniente do website de duas lojas, a *Forever 21*[1], que para cada peça de roupa tem tem de 4 a 5 imagens em diferentes poses e pontos de vista, e da loja *Mogujie*[2] que contém imagens retiradas pela loja e pelos consumidores; a outra parte das imagens foi coletada através de consultas ao *Google Images*[3].

A Figura 5.1 apresenta algumas das categorias e atributos extraídos das imagens, deixando claro que as imagens do DeepFashion estão anotadas em dois níveis. O primeiro nível se refere ao tipo da roupa, que pode ser enquadrada dentro de 50 categorias diferentes, como, blusas, shorts, saias, vestidos, etc. No segundo nível, as informações passam a ser mais específicas, referindo-se às características de *design* das roupas, que foram descritas através de 1.000 atributos distribuídos em cinco classes, são elas: textura, tecido, formato, cortes e estilo. As anotações referentes as categorias, por apresentarem uma quantidade moderada de variações (50 tipos) e serem mutualmente exclusivas foram atribuídas manualmente por profissionais. Por outro lado, os atributos foram anotados automaticamente através de metadados que acompa-

[1] https://www.forever21.com
[2] http://www.mogujie.com/
[3] https://www.google.com/imghp

Figura 5.1: Exemplo de categorias e atributos das diferentes imagens da base DeepFashion.

Fonte: Liu *et al.* (2016)

nham as imagens, seguindo a abordagem binária do *one-hot encoding*, por exemplo, caso o tecido de uma saia seja jeans, a coluna do atributo jeans será marcada com 1, caso contrário, ela será marcada com 0. Estas anotações foram feitas automaticamente pois o trabalho manual seria difícil de gerenciar devido a grande quantidade de atributos e a possibilidade de uma peça de roupa apresentar várias características. Sendo assim, como as imagens foram recolhidas de duas diferentes fontes (Google e sites e-commerce), não há um padrão na nomenclatura dos atributos, resultando na ocorrência de vários atributos iguais, porém definidos com nomes diferentes, fazendo-se necessário um refinamento manual da nossa parte para ajustar as anotações, removendo o máximo do ruído possível.

Por se tratar de uma versão inicial, decidimos limitar o escopo do nosso projeto para apenas uma categoria, sendo assim, os testes foram realizados apenas com as 13.300 imagens da categoria saia. Também foi necessário realizar refinamentos nos atributos. Primeiro, foram removidos atributos que não pertencem a saias, como tipos de manga e colarinho. Em seguida foi feita a união de dois ou mais atributos sinônimos, ou seja, que se referiam à mesma característica, por exemplo, os atributos *"dots"* e *"dotted"* ambos estão relacionados a roupas com estampa de pontos, portanto foram transformados em um único atributo. Por fim, foram selecionados apenas os atributos que estão presente em pelo menos 3% das imagens para garantir uma base de dados densa e com uma grande variedade de imagens para cada característica da roupa. Os atributos utilizados foram: *a-line, denim, dots, faux, faux leather, floral, knit , lacy, leather, maxi, midi, mini, pencil, pleated, printed, skater, stripes*.

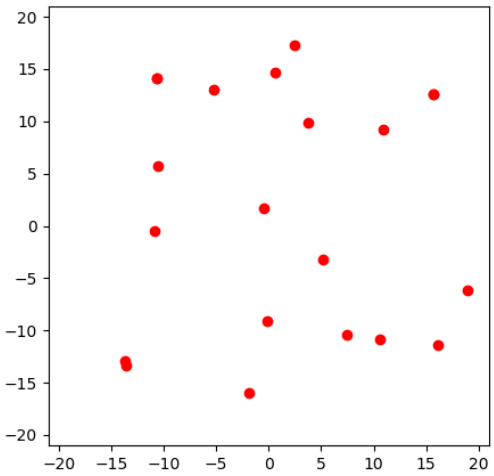
Outras bases de imagens de roupa estão disponíveis, como as elaboradas por Xiao *et al.* (2017) e Kiapour *et al.* (2015), porém elas não apresentam informações especificando as características de design das roupas, apenas categorizam elas de forma simples. Por causa dessa variedade de detalhes optamos por usar o DeepFashion.

5.2 Comparativo da Distância entre Imagens: Quadtree x Abordagem Aleatória

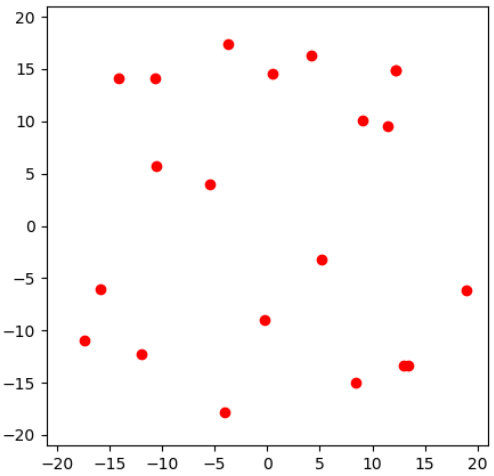
Tratando especificamente do modelo de seleção de imagens com o uso do algoritmo Quadtree, realizamos algumas simulações comparando-o com a abordagem de seleção aleatória. Os trabalhos apresentados por Silva *et al.* (2010) e Broilo & Natale (2010) também utilizam uma abordagem aleatória para compor a galeria inicial, por isso que decidimos compará-la com a nossa abordagem com a quadtree. Sendo assim, Em relação à montagem da galeria inicial, a Figura 5.2 apresenta a projeção das imagens selecionadas com o Quadtree em um plano cartesiano, enquanto que a Figura 5.3 projeta as imagens selecionadas aleatoriamente, onde ambos os casos seguem a técnica de visualização t-SNE. Como podemos perceber nos gráficos 5.2a e 5.2b o conjunto de imagens montado com o quadtree está bem distribuído, garantindo um certo espaço entre cada amostra, onde, em média, cada quadrante do plano é responsável por fornecer um quarto das imagens que compõem a galeria. Já no caso da seleção aleatória, existe a possibilidade do conjunto escolhido não explorar bem o espaço amostral, como podemos ver em 5.3a, onde a grande maioria das imagens estão concentradas no quarto quadrante e em 5.3b no qual a maioria das imagens se encontram no segundo e terceiro quadrante. Sendo assim, pode ocorrer o caso onde, inicialmente, o usuário seja apresentado apenas a uma sub-região da base de dados.

Figura 5.2: Dois exemplos, (a) e (b), da projeção num plano cartesiano das imagens da galeria inicial selecionadas com o quadtree.

(a) Simulação 1 com Quadtree na Galeria inicial



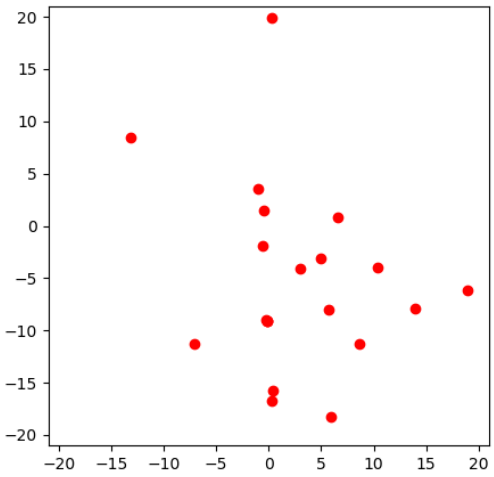
(b) Simulação 2 com Quadtree na Galeria Inicial



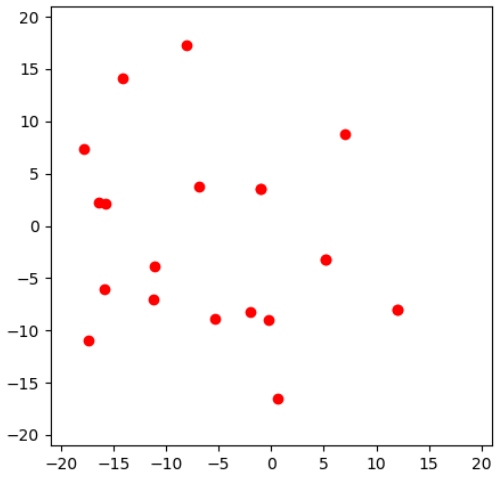
Fonte: Autoria própria

Figura 5.3: Dois exemplos, (a) e (b), da projeção num plano cartesiano das imagens da galeria inicial selecionadas aleatoriamente.

(a) Simulação 1 com Seleção Aleatória na Galeria inicial



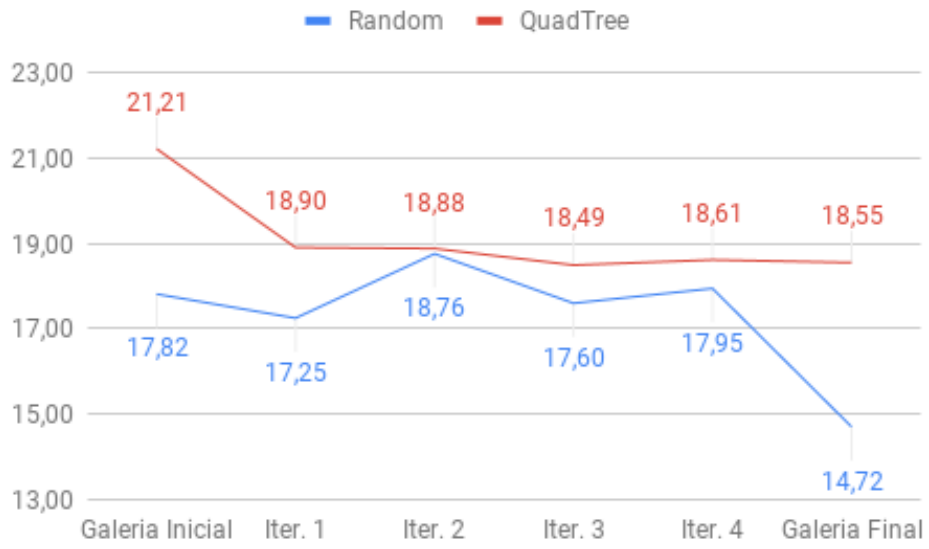
(b) Simulação 2 com Seleção Aleatória na Galeria Inicial



Fonte: Autoria própria

Para avaliar o uso da quadtree durante todo o processo iterativo, foram executadas 10 simulações, com 5 iterações de coleta de *feedback* por simulação, obedecendo a um determinado padrão de seleção de imagens como relevantes e irrelevantes, onde as imagens classificadas como relevantes possuíam uma característica em comum, enquanto que as classificadas como irrelevantes tinham em comum um atributo oposto ao das imagens relevantes, criando conjuntos bem distintos. Como exemplo de simulação, temos o seguinte caso: as mini saias foram classificadas como relevantes e as saias longas como irrelevantes, ou as saias jeans eram relevantes e as saias de couro eram irrelevantes. Para cada galeria formada através da simulação (quadtree/aleatória), foi calculada a média da distância euclidiana bidimensional de uma imagem (ponto) para todas as outras imagens da galeria, usando as coordenadas das projeções bidimensional dos pontos calculadas pelo t-SNE. Como podemos ver na Figura 5.4, que apresenta um comparativo da média de todas as simulações executadas, do ínício ao fim das iterações, as galerias formadas com o auxílio do algoritmo quadtree tiveram uma média de distância entre as imagens maior que a seleção de forma aleatória. Com isso, temos que o modelo com a quadtree oferece ao usuário, na prática, galerias mais diversificadas, para que ele possa ter acesso a diferentes estilos de roupa, evitando que o sistema retorne sempre mais do mesmo.

Figura 5.4: Comparação entre os dois modelos de seleção de imagens, apresentando a média da distância de uma imagem para todas as outras imagens de uma galeria para cada fase de iteração.



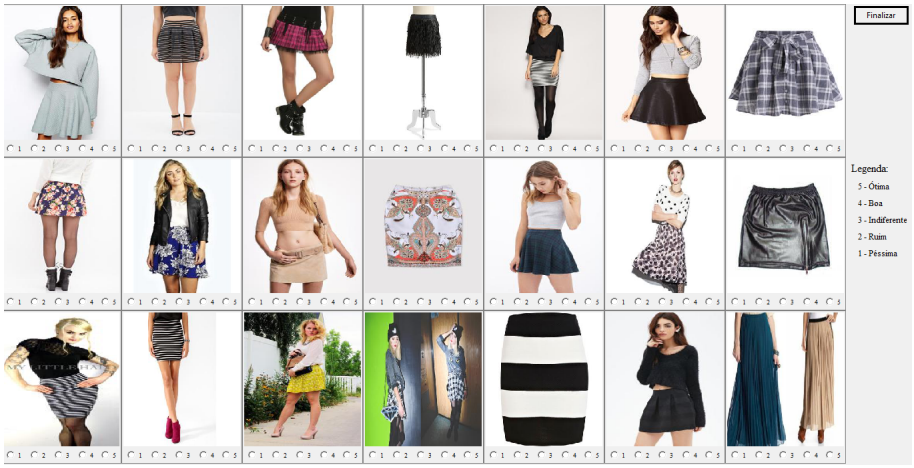
Fonte: Autoria própria.

Como o foco do sistema é garantir a diversidade das peças exibidas ao usuário, a métrica de precisão pode não ser útil para avaliar o sistema. Como o sistema não recupera apenas as imagens mais similares as classificadas como relevante pelo usuário, tentando agrada-lo com novos itens, a métrica de precisão pode ter seu desempenho afetado. Além disso, há o fato do gosto por uma roupa ser subjetivo, podendo ocorrer o caso em que roupas muito similares sejam classificadas de maneira diferente pelo usuário. Sendo assim, a métrica de precisão foi descartada devido a natureza subjetiva dos itens experimentados.

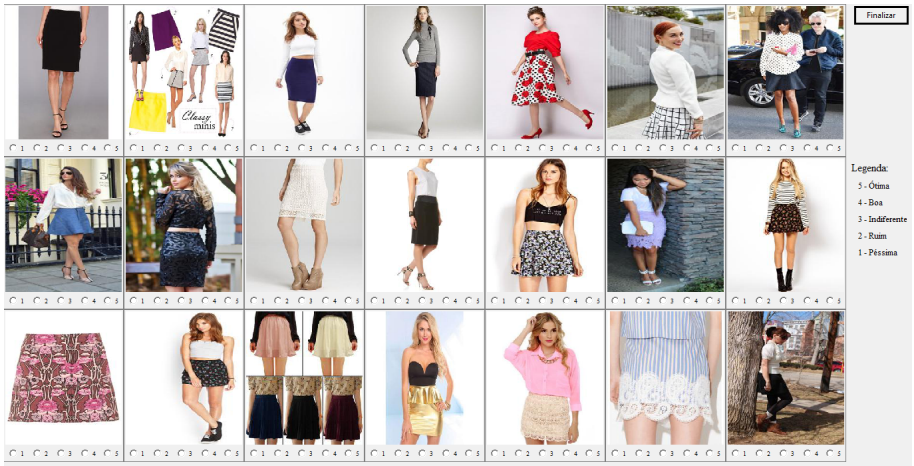
Apresentamos também um resultado visual na Figura 5.5, expondo a galeria final de duas simulações, variando de acordo com o mecanismo de seleção utilizado (aleatório e quadtree), onde são consideradas relevantes todas as saias curtas, independente das outras características associadas à elas, para evitar que consultas muito específicas. Na galeria final gerada aleatoriamente (5.5a) podemos ver uma boa quantidade de saias listradas, sendo praticamente 3 delas iguais (terceira e primeira linha). Enquanto que na galeria montada com o auxílio da quadtree (5.5b) dificilmente identificamos um padrão que se repete por mais de 3 exemplos, com exceção do atributo saia curta, escolhido como relevante.

Figura 5.5: Resultado da galeria final formada com as duas abordagens através de uma simulação. (a) Considera a abordagem aleatória. (b) Considera a abordagem com o quadtree.

(a) Abordagem aleatória



(b) Abordagem com quadtree



Fonte: Autoria própria.

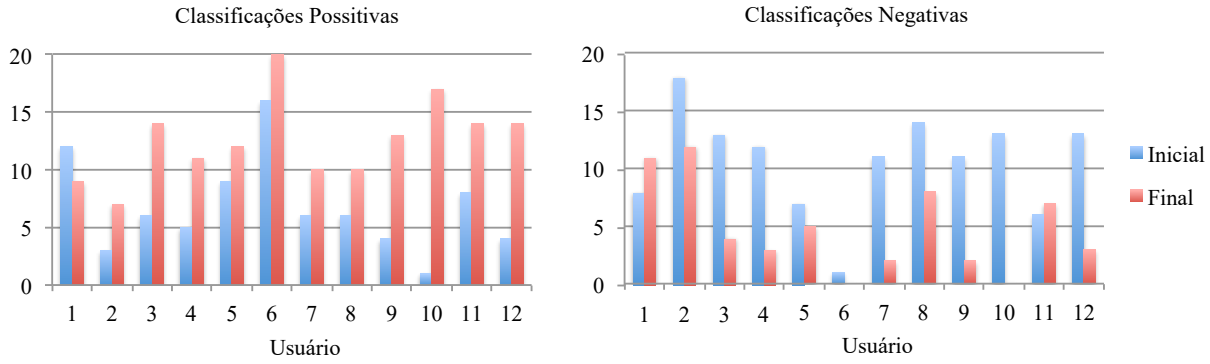
5.3 Experimentos com Usuários

A última fase de testes envolveu experimentos com usuários reais, cujo objetivo foi averiguar se o modelo está realmente aprendendo quais atributos são relevantes para o usuário, recomendando peças de roupa diversificadas e agradáveis. O protótipo foi experimentado por 12 indivíduos do sexo feminino, maiores de 18 anos que estão cursando ou completaram o ensino superior, no qual 10 delas já realizaram alguma compra de roupa online.

Para cada usuário, foi registrado a quantidade de imagens classificadas como positivas e negativas ao longo de 7 galerias consecutivas, sendo que a última galeria foi composta apenas por exemplos da região relevante, para melhor avaliar se o sistema foi capaz de identificar as preferências do usuário após 6 iterações. Vale ressaltar que o usuário não é obrigado a classificar todas as imagens da galeria, caso a peça de roupa apresentada seja indiferente ao seu gosto.

A Figura 5.6 apresenta os resultados obtidos. Em média, os usuários classificaram positivamente 6,66 imagens da galeria inicial e 12,58 imagens da galeria final, resultando em um aumento médio de 5,91 imagens classificadas como relevantes, de um total de 21 imagens da galeria (28%). Em contraste, os usuários classificaram negativamente, em média, 10,58 imagens da galeria inicial e 4,75 imagens da galeria final, resultando em uma redução média de 5,83 imagens classificadas como irrelevante (27%). Estes resultados confirmam a hipótese que nossas galerias inteligentes aprendem as preferências do usuário e oferecem itens relevantes, de uma maneira diversificada, como já foi discutido na 5.2.

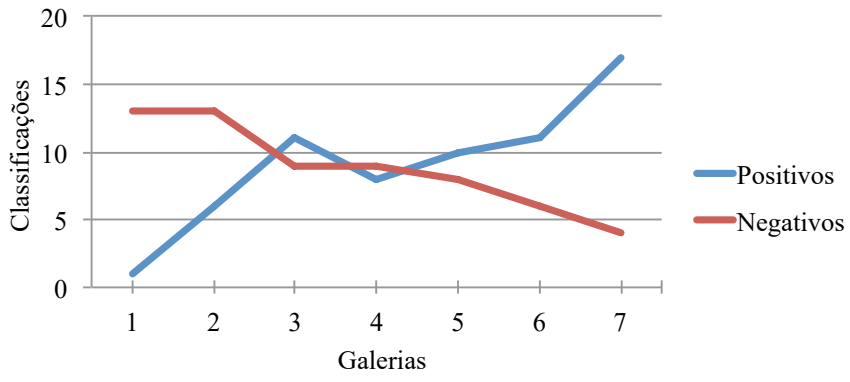
Figura 5.6: Experimentos com usuários: Quantidade imagens classificadas positivamente (esquerda) e negativamente (direita) para cada usuário, da galeria inicial e da última galeira (após 6 iterações), tendo 21 imagens em cada uma das galerias.



Fonte: Autoria própria.

Os gráficos da Figura 5.7 esclarecem a quantidade de classificações positivas e negativas obtidos em cada fase da execução do processo iterativo para um dos indivíduos submetido aos testes. Podemos observar que a evolução ao longo das galerias apresentam um comportamento uniforme, tal que o número de classificações positivas por galeria tende a aumentar, enquanto que o número de classificações negativas diminuem ao longo do tempo.

Figura 5.7: Evolução do processo iterativo de aprendizado para um único experimento. O eixo horizontal representa o número da iteração e o eixo vertical a quantidade de imagens classificadas.



Fonte: Autoria própria.

5.4 Tempo de Execução

Afim de avaliar tempo de execução do protótipo, dividimos ele em 3 partes, como podemos ver na Tabela 5.1. Para realizar essa medição foi utilizado um processador Intel i5-8250U 1,60 Ghz de um computador portátil comum. A etapa de pré-processamento envolve a aplicação do t-SNE em todas as imagens, levando quase 2 minutos para ser concluída. As coordenadas bidimensionais resultantes são salvas e utilizadas para todos os testes. Sendo assim, esta etapa não toma tempo do usuário, pois o sistema já vai iniciar com as coordenadas predefinidas

O Tempo para montar as galerias compreende algumas operações. Para a galeria inicial temos duas operações, primeiro a construção da quadtree com todos os elementos da base de dados e, em seguida, a seleção das 21 imagens que irão compor a galeria. Já para a galeria intermediária marcamos o tempo para treinar o classificador SVM e classificar todas as imagens da base, dividir as imagens em 3 regiões e calcular a probabilidade de seleção de cada imagem e, por fim, sortear $10 * n$ imagens de cada região e aplicar a quadtree para selecionar as n imagens que irão compor a galeria.

A montagem da galeria intermediária levou menos tempo porque a quadtree foi aplicado com apenas $10 * n$ pontos, uma vez que na galeria inicial a quadtree foi usado com as 13.300 imagens da base de dados.

	Tempo de Execução (s)
Pré-Processamento	111,25
Geração Galeria Inicial	0,1455
Geração Galeria Intermediária	0,04321

Tabela 5.1: Tempo de execução de cada fase do protótipo.

Capítulo 6

Conclusão e Trabalhos Futuros

Neste trabalho foi proposto uma diferente abordagem de recuperação de imagem considerando a coleta de *feedback* do usuário. Como contribuição principal, apresentamos uma estratégia de recuperação tripla: selecionar imagens relevantes, selecionar imagens irrelevantes e selecionar imagens da região de incerteza. Visamos também aprimorar a diversidade combinando projeção multidimensional com a estrutura de dados quadtree para realizar a seleção. Experimentos revelaram que nossa abordagem pode melhorar rapidamente a recuperação de itens relevantes em apenas poucas iterações do usuário, até para grandes bases de dados, enquanto uma maior diversidade do que a seleção aleatória. Nossa solução também tem potencial para ser adotada em outras aplicações que requerem a exploração base de dados com imagens anotadas.

Algumas ideias estão sendo consideradas para evoluir este trabalho. De imediato, há a possibilidade de explorar mais a fase de classificação dos dados, aplicando outros algoritmos de aprendizagem de máquina baseados em distância ou buscar variações do SVM, com o objetivo de realizar uma comparação de desempenho com o classificador SVM utilizado. Há também a necessidade de melhorar a qualidade das imagens utilizadas, visto que muitos usuários alegaram este problema. Sendo assim, pretende-se desenvolver um módulo responsável por extrair as características das imagens, tornando o modelo de recuperação de imagem independente da base de dados utilizada, gerando anotações automaticamente. Além disso, consideramos utilizar mais informações do usuário, como tipo de corpo ou montar o perfil do usuário de acordo com seu estilo, que são exemplos bem difundidos no ramo da moda, visando aprimorar a recomendação.

Referências Bibliográficas

AFIFI, A. J., & ASHOUR, W. M. 2012 (Dec). Content-Based Image Retrieval Using Invariant Color and Texture Features. *Pages 1–6 of: 2012 International Conference on Digital Image Computing Techniques and Applications (DICTA)*.

AGARWAL, PANKAJ, VEMPATI, SREEKANTH, & BORAR, SUMIT. 2018. *Personalizing Similar Product Recommendations in Fashion E-commerce*.

BOSER, BERNHARD E., GUYON, ISABELLE M., & VAPNIK, VLADIMIR N. 1992. A Training Algorithm for Optimal Margin Classifiers. *Pages 144–152 of: Proceedings of the Fifth Annual Workshop on Computational Learning Theory*. COLT '92. New York, NY, USA: ACM.

BROILO, M., & NATALE, F. G. B. DE. 2010. A Stochastic Approach to Image Retrieval Using Relevance Feedback and Particle Swarm Optimization. *IEEE Transactions on Multimedia*, **12**(4), 267–277.

CORTES, CORINNA, & VAPNIK, VLADIMIR. 1995. Support-vector networks. *Machine Learning*, **20**(3), 273–297.

D'ANGELO, ANTHONY. 2016. A Brief Introduction to Quadtrees and Their Applications. *Canadian Conference on Computational Geometry*.

EAKINS, JOHN, GRAHAM, MARGARET, & FRANKLIN, TOM. 1999. Content-based Image Retrieval. *Library and Information Briefings*, **85**, 1–15.

FACELI, KATTI, LORENA, ANA CAROLINA, GAMA, JOÃO, & CARVALHO, ANDRÉ CARLOS PONCE DE LEON FERREIRA DE. 2011. *Inteligência artificial: uma abordagem de aprendizado de máquina*. LTC.

FINKEL, R. A., & BENTLEY, J. L. 1974. Quad trees a data structure for retrieval on composite keys. *Acta Informatica*, **4**(1), 1–9.

JHANWAR, N., CHAUDHURI, S., SEETHARAMAN, G., & ZAVIDOVIQUE, B. 2004. Content based image retrieval using motif cooccurrence matrix. *Image and Vision Computing*, **22**(14), 1211 – 1220. The Indian Conference on Vision, Graphics and Image Processing.

JOLLIFFE, I.T. 1986. *Principal Component Analysis*. Springer Verlag.

KENNEDY, JAMES, EBERHART, RUSSELL C., & SHI, YUHUI. 2001. *Swarm intelligence*. M. Kaufmann.

KIAPOUR, M. HADI, HAN, XUFENG, LAZEBNIK, SVETLANA, BERG, ALEXANDER C., & BERG, TAMARA L. 2015. Where to Buy It: Matching Street Clothing Photos in Online Shops. *2015 IEEE International Conference on Computer Vision (ICCV)*, 3343–3351.

KONDO, SHIN-ICHIRO, TOYOURA, MASAHIRO, & MAO, XIAOYANG. 2014. Sketch Based Skirt Image Retrieval. *Pages 11–16 of: Proceedings of the 4th Joint Symposium on Computational Aesthetics, Non-Photorealistic Animation and Rendering, and Sketch-Based Interfaces and Modeling*. SBIM '14. New York, NY, USA: ACM.

LI, HONGLIN, TOYOURA, MASAHIRO, SHIMIZU, KAZUMI, YANG, WEI, & MAO, XIAOYANG. 2016. Retrieval of Clothing Images Based on Relevance Feedback with Focus on Collar Designs. *Vis. Comput.*, **32**(10), 1351–1363.

LIKA, BLERINA, KOLOMVATSOS, KOSTAS, & HADJIEFTHYMIADES, STATHES. 2014. Facing the Cold Start Problem in Recommender Systems. *Expert Syst. Appl.*, **41**(4), 2065–2073.

LIU, S., SONG, Z., LIU, G., XU, C., LU, H., & YAN, S. 2012 (June). Street-to-shop: Cross-scenario clothing retrieval via parts alignment and auxiliary set. *Pages 3330–3337 of: 2012 IEEE Conference on Computer Vision and Pattern Recognition*.

LIU, Z., LUO, P., QIU, S., WANG, X., & TANG, X. 2016 (June). DeepFashion: Powering Robust Clothes Recognition and Retrieval with Rich Annotations. *Pages 1096–1104 of: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

MARCELINO, JOSÉ, FARIA, JOÃO, BAÍA, LUÍS, & SOUSA, RICARDO GAMELAS. 2018. A Hierarchical Deep Learning Natural Language Parser for Fashion. *CoRR*, **abs/1806.09511**.

MITCHELL, THOMAS M. 1997. *Machine Learning*. 1 edn. New York, NY, USA: McGraw-Hill, Inc.

OLIVEIRA, M. C. F., & LEVKOWITZ, H. 2003. From visual data exploration to visual data mining: a survey. *IEEE Transactions on Visualization and Computer Graphics*, **9**(3), 378–394.

RICCI, FRANCESCO, ROKACH, LIOR, SHAPIRA, BRACHA, & KANTOR, PAUL B. 2010. *Recommender Systems Handbook*. 1st edn. Berlin, Heidelberg: Springer-Verlag.

RIJSBERGEN, C. J. VAN. 1979. *Information Retrieval*. 2nd edn. Newton, MA, USA: Butterworth-Heinemann.

SAFFAWI, ZEYAD, MOHAMAD, DZULKIFLI, SABA, TANZILA, ALKAWAZ, MOHAMMED, REHMAN, AMJAD, AL-RODHAAN, MZNAH, & AL-DHELAAN, ABDULLAH. 2014. Content-based image retrieval using PSO and k-means clustering algorithm. *Arabian Journal of Geosciences*, **8**(aug), 6211–6224.

SAMET, HANAN. 1988. An Overview of Quadtrees, Octrees, and Related Hierarchical Data Structures. *Pages 51–68 of:* EARNSHAW, RAE A. (ed), *Theoretical Foundations of Computer Graphics and CAD*. Berlin, Heidelberg: Springer Berlin Heidelberg.

SAMET, HANAN. 1990. *The Design and Analysis of Spatial Data Structures*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc.

SAMET, HANAN, & WEBBER, ROBERT E. 1985. Storing a Collection of Polygons Using Quadtrees. *ACM Trans. Graph.*, **4**(3), 182–222.

SHAWE-TAYLOR, JOHN, & CRISTIANINI, NELLO. 2004. *Kernel Methods for Pattern Analysis*. New York, NY, USA: Cambridge University Press.

SHNEIER, MICHAEL. 1981. Two hierarchical linear feature representations: Edge pyramids and edge quadtrees. *Computer Graphics and Image Processing*, **17**(3), 211 – 224.

SILVA, ANDRÉ TAVARES DA, FALCÃO, ALEXANDRE X., & MAGALHÃES, LÉO PINI. 2010. A new CBIR approach based on relevance feedback and optimum-path forest classification. *Journal of WSCG*, **18**(1-3), 73–80.

SMEULDERS, ARNOLD W. M., WORRING, MARCEL, SANTINI, SIMONE, GUPTA, AMARNATH, & JAIN, RAMESH. 2000. Content-Based Image Retrieval at the End of the Early Years. *IEEE Trans. Pattern Anal. Mach. Intell.*, **22**(12), 1349–1380.

STUDENT. 1908. THE PROBABLE ERROR OF A MEAN. *Biometrika*, **6**(1), 1–25.

TONG, SIMON, & CHANG, EDWARD. 2001. Support Vector Machine Active Learning for Image Retrieval. *Pages 107–118 of: Proceedings of the Ninth ACM International Conference on Multimedia*. MULTIMEDIA '01. New York, NY, USA: ACM.

TORRES, RICARDO DA SILVA, & FALCÃO, ALEXANDRE XAVIER. 2006. Content-Based Image Retrieval: Theory and Applications. *Revista de Informática Teórica e Aplicada*, **13**, 161–185.

VAN DER MAATEN, LAURENS, & HINTON, GEOFFREY E. 2008. Visualizing Data using t-SNE. *Journal of Machine Learning Research*.

VELTKAMP, REMCO, & TANASE, MIRELA. 2000. Content-Based Image Retrieval Systems: A Survey. *Technical report, Utrecht University*, 11.

XIAO, HAN, RASUL, KASHIF, & VOLLGRAF, ROLAND. 2017. Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. *CoRR*, **abs/1708.07747**.

YILDIZER, ELA, BALCI, ALI METIN, HASSAN, MOHAMMAD, & ALHAJJ, REDA. 2012. Efficient Content-based Image Retrieval Using Multiple Support Vector Machines Ensemble. *Expert Syst. Appl.*, **39**(3), 2385–2396.

ZHOU, WENGANG, LI, HOUQIANG, & TIAN, QI. 2017. Recent Advance in Content-based Image Retrieval: A Literature Survey. *CoRR*, **abs/1706.06064**.

ZHOU, XIANG, & S. HUANG, THOMAS. 2003. Relevance feedback in image retrieval: A comprehensive review. *Multimedia Syst.*, **8**(04), 536–544.