



Universidade Estadual de Campinas
Instituto de Computação



Bárbara Caroline Benato

Anotação de Dados guiada por Projeção de Características

CAMPINAS
2019

Bárbara Caroline Benato

Anotação de Dados guiada por Projeção de Características

Dissertação apresentada ao Instituto de Computação da Universidade Estadual de Campinas como parte dos requisitos para a obtenção do título de Mestra em Ciência da Computação.

Orientador: Prof. Dr. Alexandre Xavier Falcão

Este exemplar corresponde à versão final da Dissertação defendida por Bárbara Caroline Benato e orientada pelo Prof. Dr. Alexandre Xavier Falcão.

CAMPINAS
2019

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca do Instituto de Matemática, Estatística e Computação Científica
Silvania Renata de Jesus Ribeiro - CRB 8/6592

B431a Benato, Bárbara Caroline, 1994-
Anotação de dados guiada por projeção de características / Bárbara Caroline Benato. – Campinas, SP : [s.n.], 2019.

Orientador: Alexandre Xavier Falcão.
Dissertação (mestrado) – Universidade Estadual de Campinas, Instituto de Computação.

1. Aprendizado de máquina. 2. Visualização de informação. 3. Aprendizagem semi-supervisionado. I. Falcão, Alexandre Xavier, 1966-. II. Universidade Estadual de Campinas. Instituto de Computação. III. Título.

Informações para Biblioteca Digital

Título em outro idioma: Data annotation guided by feature projection

Palavras-chave em inglês:

Machine learning

Information visualization

Semi-supervised learning

Área de concentração: Ciência da Computação

Titulação: Mestra em Ciência da Computação

Banca examinadora:

Alexandre Xavier Falcão [Orientador]

Pedro Jussieu de Rezende

Moacir Antonelli Ponti

Data de defesa: 10-09-2019

Programa de Pós-Graduação: Ciência da Computação

Identificação e informações acadêmicas do(a) aluno(a)

- ORCID do autor: <https://orcid.org/0000-0003-0806-3607>

- Currículo Lattes do autor: <http://lattes.cnpq.br/4508195004910105>



Universidade Estadual de Campinas
Instituto de Computação



Bárbara Caroline Benato

Anotação de Dados guiada por Projeção de Características

Banca Examinadora:

- Prof. Dr. Alexandre Xavier Falcão
Instituto de Computação - UNICAMP (Orientador)
- Prof. Dr. Pedro Jussieu de Rezende
Instituto de Computação - UNICAMP
- Prof. Dr. Moacir Antonelli Ponti
Instituto de Ciências Matemáticas e de Computação - USP

A ata da defesa, assinada pelos membros da Comissão Examinadora, consta no SIGA/Sistema de Fluxo de Dissertação/Tese e na Secretaria do Programa da Unidade.

Campinas, 10 de setembro de 2019

Agradecimentos

Primeiramente, agradeço às agências de financiamento que me deram a oportunidade de realizar essa pesquisa:

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001.

O presente trabalho foi realizado com apoio da Fundação de Amparo à Pesquisa do Estado de São Paulo (Fapesp) pela concessão de bolsa regular de mestrado (Processo #2016/25776-0) e de bolsa de estágio pesquisa no exterior (Processo #2017/25327-3).

Agradeço a Deus pelos seus propósitos e pelos caminhos em que me conduziu, desde a minha inscrição no programa de pós-graduação, minha mudança para Campinas, até o fim desse mestrado.

Agradeço imensamente ao Prof. Alexandre Falcão por ter me aceitado como sua aluna. Por todo o incentivo, ajuda, conhecimento e orientação. Agradeço, também, por sua confiança, preocupação e atenção, além de sua paciência pelas vezes em que me ausentei do laboratório. Também agradeço ao Prof. Alexandru Telea, meu supervisor na Holanda, pela breve mas significativa orientação e pela oportunidade no exterior de amadurecimento pessoal e profissional.

Agradeço à Unicamp e todos os funcionários que estiveram diretamente e indiretamente envolvidos na minha trajetória.

Não menos importante, agradeço aos meus pais, Júlio e Helen, pelo incentivo, suporte, carinho e acolhimento, quando os obstáculos apareceram pelo caminho. Agradeço aos meus familiares Eustasio (*in memoriam*) e Bela, Mauro (*in memoriam*) e Lurdinha, Valéria, Patrici, João Paulo, Helena, Gustavo, seus “agregados” e Bento.

Agradeço ao meu namorado, Pedro, que acompanhou de perto a minha transição, por todo o suporte emocional, longas conversas, conselhos e apoio. Obrigada por sempre me entender e não abandonar o nosso barco.

Por fim, agradeço aos meus amigos e amigas, bem como aos colegas do *Laboratory of Image Data Science* (LIDS) pelas ajudas, conversas, revisões, colaborações e confraternizações.

Resumo

Anotação de dados baseada na inspeção visual do usuário (supervisão) de cada exemplo de treinamento pode ser um processo trabalhoso, especialmente quando o número de exemplos a serem rotulados é alto — um conhecido problema no aprendizado de parâmetros de redes neurais com muitas camadas escondidas em Aprendizado de Máquina. A anotação de dados pelo usuário pode ser ainda mais custosa, particularmente, em áreas que requerem um profissional com conhecimento especializado, como Medicina e Biologia. Tradicionalmente, estudos têm apresentado soluções que empregam o aprendizado semi-supervisionado para lidar com tal questão, a fim de propagar rótulos a partir de poucas amostras supervisionadas para amostras não-supervisionadas, explorando a distribuição dessas amostras no espaço de características. Contudo, tais trabalhos não consideram a habilidade cognitiva do usuário para entender projeções do espaço de características com o propósito de aumentar a quantidade de amostras rotuladas para o aprendizado de máquina. Neste trabalho, explorou-se a habilidade cognitiva do usuário assistido por uma ferramenta de analítica visual na tarefa de propagar rótulos para uma grande quantidade de amostras não-supervisionadas. O usuário é guiado pelo conhecimento de rótulo de poucas amostras, bem como pela informação visual da distribuição das amostras na projeção do espaço de características. Além disso, investigou-se uma abordagem de anotação de dados semi-automática, ou seja, que combina as estratégias de propagação de rótulos manual e automática, utilizando uma projeção apropriada do espaço de características e estimação de rótulo semi-supervisionada com medida de certeza para reduzir o esforço do usuário. O método é validado em dois contextos: em uma base conhecida de imagens, MNIST, e em bases de imagens de parasitos intestinais de humanos com e sem impurezas fecais (uma classe adversa que torna o problema mais desafiador). O estudo avaliou duas abordagens automáticas para aprendizado semi-supervisionado nos espaços latente e projetado. Além disso, avaliou dois classificadores supervisionados treinados com os conjuntos rotulados. Por fim, os experimentos visaram escolher a solução que melhor reduz o esforço do usuário para a anotação de dados e, ao mesmo tempo, aumenta os resultados de acurácia na classificação do conjunto de teste. Os resultados sugerem que ferramentas de analítica visual, quando utilizadas para combinar as habilidades complementares de humanos e máquinas, possibilitam um aprendizado de máquina mais eficaz.

Abstract

Data annotation using visual inspection (supervision) of each training example can be a laborious process, especially when the number of examples to be supervised is high — a well-known requirement for deep learning. The data annotation by the user can be even more costly, particularly in areas that requires an expert with specialized knowledge, such as Medicine and Biology. Traditionally, studies have presented solutions that employ semi-supervised learning to deal with such issue to propagate labels from a few supervised samples to unsupervised samples by exploring the distribution of those samples in the feature space. However, such works do not consider the user’s cognitive ability to understand feature space projections for the purpose of increasing the number of labeled samples for machine learning. In this work, we present data annotation methods in which the user is assisted by a visual analytics tool in the task of propagating labels to a large number of unsupervised samples. The user is guided by the knowledge of few labeled samples as well as the visual information of the sample distribution in feature space projection. Also, we investigate a semi-automatic data annotation approach. That is, we combine manual and automatic label propagation using an appropriate feature space projection and semi-supervised label estimation based on a certainty measure to reduce user effort in data annotation. We validate the method in two contexts: on a known image database, MNIST, and on images of human intestinal parasites with and without fecal impurities (an adverse class that makes the problem even more challenging). We evaluate two automatic approaches to semi-supervised learning in latent and projected spaces. In addition, we evaluate two supervised classifiers, trained with the labeled sets. Finally, the experiments aim to choose the solution that best reduces the user effort for data annotation and also increases the classification accuracy on test sets. The results suggest that visual analytics tools can provide more effective machine learning whenever they combine the complementary skills of humans and machines.

Lista de Figuras

3.1	Exemplo de uma rede neural artificial. Cada neurônio é representado por um círculo na figura e as ligações entre eles são os pesos. A rede apresenta uma camada de entrada, duas camadas escondidas (neste caso) e uma camada de saída, que determinam o número de classes do problema.	20
3.2	Representação de um neurônio artificial.	20
3.3	Exemplo de uma Rede Neural Autocodificadora.	22
3.4	Operação de convolução. Neste exemplo, o filtro K_i de tamanho 2×2 , menor do que a imagem I , é aplicado na primeira posição $(0, 0)$ na imagem I . Com $stride=1$, ou seja, o deslocamento do filtro sobre a imagem, o filtro é aplicado na posição seguinte à direita na imagem.	26
3.5	A operação de <i>pooling</i> varre uma imagem I a partir da posição $(0, 0)$ com uma janela, neste caso, de 2×2 e $stride=2$, e o maior valor da imagem em cada janela é utilizado para criar a imagem P resultante.	27
3.6	Conjunto de treinamento e teste. Projeção das características. (Imagem retirada do trabalho de Rauber <i>et al.</i> [56])	31
3.7	Conjunto de treinamento. Projeção da observação e das características, para características selecionadas. (Imagem retirada do trabalho de Rauber <i>et al.</i> [56])	32
3.8	Separação de categorias distintas (representadas pelos círculos e quadrados) pelo hiperplano no espaço de características.	34
4.1	Método semi-automático proposto para anotação de dados.	47
4.2	Representação das cores segundo o esquema de cor HSV. (Figura reproduzida da documentação do <i>Qt.</i>)	49
4.3	Projeção de características com amostras não-supervisionadas de vermelho (baixa confiança) até o verde (alta confiança).	50
4.4	O usuário (a) manualmente seleciona uma amostra de um grupo para (b) propagar os rótulos para amostras não-supervisionadas. (Figura reproduzida de [7]).	50
4.5	Projeção com grupos bem separados em (a) e <i>tooltip</i> mostrado para cada ponto no 2D em (b). (Figura reproduzida de [7].)	51
4.6	Propagação semi-automática de rótulos é realizada, primeiramente, a partir das amostras supervisionadas (pontos coloridos pela classe, borda vermelha) automaticamente para as amostras não-supervisionadas e com maior confiança (cores mais claras, sem borda). As amostras restantes de menor confiança (preto) são candidatas para propagação manual.	52
5.1	Exemplos da base de dados de Parasitos para cada espécie em <i>Helminth Eggs</i> (esquerda) e impurezas fecais similares (direita).	56

5.2	Arquiteturas da AuNN para as bases de dados MNIST (superior) e parasitos (inferior). As camadas em amarelo são as camadas de convolução, as camadas em vermelho no começo de cada rede são as operações de <i>Max-Pooling</i> e as camadas em azul são operações de <i>Up-Sampling</i>	57
5.3	Projeções coloridas pela confiança do rótulo (vermelho=baixa confiança, verde=alta confiança). Linhas são base de dados (mais fáceis em cima, mais difíceis embaixo). Colunas mostram o conjunto todo de amostras supervisionadas e não-supervisionadas $S \cup U$, o conjunto L_c das amostras com maior confiança rotuladas por ALP e o conjunto $U \setminus L_c$ das amostras com menor confiança que serão apresentadas ao usuário para a propagação manual de rótulos.	59
5.4	Comparação de diferentes métodos de propagação de rótulos (colunas) para diferentes bases de dados (linhas). Da esquerda para a direita: ILP, SALP, rótulos propagados automaticamente pelo OPF-Semi e resultado final de propagação de rótulos do SALP juntamente com o OPF-Semi. Cores indicam rótulos de amostras supervisionadas (ILP, SALP) ou amostras não-annotadas e rótulos propagados (OPF-Semi, OPF-Semi+user). Amostras em preto são amostras a serem consideradas para a propagação de rótulos manual (três colunas à esquerda), e amostras desconsideradas na propagação de rótulos manual (coluna à direita). Os tamanhos dos conjuntos de amostras são apresentados à direita.	61
5.5	Valores de κ para as bases de dados estudadas, considerando os classificadores OPF e SVM (colunas) e métodos de propagação de rótulos automáticos LapSVM e OPF-Semi (linhas). Curvas mostram valores de κ para ILP e SALP para fins de comparação. As bases de dados estão ordenadas da mais fácil para a mais difícil para classificar (esquerda para direita) com base nos resultados de SALP.	64
5.6	(a) porcentagem de amostras rotuladas em U pelos métodos de propagação ILP e SALP, e (b) o número de amostras nas seis bases de dados estudadas.	65
5.7	Ganho normalizado, isto é, κ dividido pela porcentagem de amostras manualmente rotuladas pelo ILP e SALP para os classificadores SVM e OPF para as seis bases de dados estudadas (ordenadas pelo tamanho da esquerda para a direita).	66
6.1	Método proposto para trabalhos futuros.	68

Lista de Tabelas

5.1	Número de amostras nos conjuntos S , U e T para cada base de dados: MNIST, parasitos, e parasitos com impureza (I).	55
5.2	Valores de κ médio e desvio padrão para os classificadores SVM e OPF no conjunto T para MNIST, parasitos sem impureza e parasitos com impureza (I). Os melhores resultados para cada base de dados são mostrados em negrito.	58
5.3	Valores de κ médio e desvio padrão para os classificadores SVM e OPF no conjunto T para as bases de dados estudadas. Os melhores resultados por base de dados são mostrados em negrito.	63

Sumário

1	Introdução	13
1.1	Desafios e objetivos	13
1.2	Contribuições	14
1.3	Organização do texto	15
2	Trabalhos relacionados	16
3	Referencial Teórico	19
3.1	Redes neurais artificiais	19
3.1.1	Modelo de um neurônio artificial	19
3.1.2	Redes neurais autocodificadoras	21
3.1.3	Operação de convolução	24
3.1.4	Operação de <i>pooling</i>	26
3.2	Técnicas de projecção	27
3.2.1	t-SNE	28
3.3	Analítica visual de dados	30
3.4	Classificadores de padrões	31
3.4.1	Aprendizado supervisionado	33
3.4.2	Aprendizado semi-supervisionado	41
4	Abordagem Semi-automática para Anotação de Dados	46
4.1	Aprendizado não-supervisionado de características	47
4.2	Projeção do espaço de características	47
4.3	Estimação de rótulos semi-supervisionada	48
4.4	Propagação de rótulos manual	49
5	Experimentos e Resultados	53
5.1	Configuração experimental	53
5.2	Experimentos base	54
5.3	Bases de dados	55
5.4	Detalhes de implementação	56
5.4.1	Extração de características	56
5.4.2	Projeção	56
5.5	Resultados experimentais	56
5.5.1	Influência da redução do espaço de características do nD para 2D	57
5.5.2	A escolha do limiar de confiança	58
5.5.3	A melhor abordagem de propagação de rótulos	59
5.5.4	Valor agregado pelo SALP	62
5.5.5	Como os resultados dependem da qualidade da projecção	62

5.6	Discussão	63
5.6.1	O espaço de características: nD <i>versus</i> 2D	63
5.6.2	Redução de esforço do usuário	64
5.6.3	Limitações	66
6	Conclusão e Trabalhos Futuros	67
	Referências Bibliográficas	70

Capítulo 1

Introdução

Com o aumento da disponibilidade de dados em diversos domínios de aplicação, a atenção da comunidade de Aprendizado de Máquina, do inglês *Machine Learning* (ML), tem mudado o foco de problemas que consideram poucos dados para aqueles com grandes quantidades de dados. Nesse contexto, novas técnicas têm sido desenvolvidas e outras existentes são melhoradas para lidar com dados massivos do mundo real. O aprendizado de parâmetros de uma rede neural com muitas camadas, do inglês *Deep Learning* (DL), é um exemplo.

Redes Neurais Convolucionais, do inglês *Convolutional Neural Networks* (CNNs) [33, 30], são técnicas de DL de bastante relevância para tarefas de classificação de imagens. Originalmente desenvolvidas para classificar códigos de endereçamento postal escritos a mão, as CNNs foram aperfeiçoadas de forma a serem capazes de ultrapassar os seres humanos no contexto de várias aplicações. Contudo, esta abordagem necessita de uma grande quantidade de amostras rotuladas para seu treinamento [84], a fim de se obter alta acurácia na classificação de amostras ainda não vistas.

Em aplicações do mundo real, o cenário revela, frequentemente, grandes quantidades de amostras não-supervisionadas disponíveis, enquanto amostras supervisionadas são pouco viáveis. Amostras supervisionadas são de dispendiosa obtenção, uma vez que requerem considerável esforço humano e podem ser de alto custo pela necessidade de haver um especialista da área para anotar os dados. Como exemplo, as aplicações em ciências médicas, biológicas e biomédicas. Por este motivo, a obtenção de dados supervisionados e corretamente anotados é um grande desafio em ML, especialmente quando a anotação é subjetiva, isto é, quando rótulos diferentes podem ser atribuídos para uma mesma amostra. Contudo, o escopo do presente trabalho não inclui a anotação subjetiva.

1.1 Desafios e objetivos

Os resultados de classificação e reconhecimento de padrões de bons conjuntos de treinamento sujeitam-se à quantidade obtida de dados corretamente anotados e à eficácia de técnicas de ML. Dado o esforço humano necessário para a anotação desses dados, pesquisadores têm explorado novas formas de anotação pela inspeção visual das amostras (supervisão). Com o objetivo de reduzir o esforço necessário para a anotação de dados,

soluções que envolvem aprendizado semi-supervisionado e aprendizado ativo, do inglês *Active Learning* (AL) [3], têm sido propostas [26, 63, 60, 89]. Métodos de aprendizado semi-supervisionado exploram a propagação de rótulos a partir de dados supervisionados para não-supervisionados com o objetivo de aprender um modelo com acurácia de classificação equivalente ou maior a do modelo treinado apenas com as amostras supervisionadas. A propagação de rótulos não é considerada de fato uma anotação, uma vez que não há como garantir que os rótulos atribuídos estejam corretos. Ao passo que métodos de aprendizado ativo, a fim de reduzir o erro de propagação, inserem o usuário no processo através de iterações de supervisão. O intuito é reduzir a quantidade de supervisões necessárias pela escolha apropriada de amostras que serão anotadas pelo usuário a cada iteração.

Contudo, quando a quantidade de amostras supervisionadas é realmente pequena e de amostras não-supervisionadas é muito grande, ou seja, quando não é capaz de representar bem a distribuição das classes do problema em questão, o uso de tais técnicas pode ser um empecilho ou, ainda, requerer muitas iterações do aprendizado ativo para obtenção de um resultado promissor. Além disso, tais técnicas não consideram a habilidade cognitiva do usuário de entender e analisar a distribuição dos dados, a fim de reduzir o esforço do usuário para a anotação de grandes quantidades de amostras.

Logo, o presente trabalho tem como principal objetivo explorar a habilidade cognitiva do usuário em abstrair informações para anotar dados. Tal habilidade não é necessariamente baseada na supervisão de imagem por imagem, mas é embasada na análise da distribuição dos dados pelo usuário e a interação dele com a projeção do espaço de características desses dados. Consequentemente, espera-se que o esforço do usuário na tarefa de anotação de dados seja reduzido. Para tanto, técnicas de analítica visual de dados e visualização da informação, do inglês *Visual Analytics* (VA) e *information visualization* respectivamente, são exploradas. Ademais, uma outra fonte de informação — o nível de confiança — dada por um algoritmo de aprendizado semi-supervisionado auxilia no processo de anotação de dados, uma vez que fornece ao usuário as amostras que o algoritmo tem menor confiança em relação aos rótulos.

1.2 Contribuições

Como contribuição, o presente trabalho apresenta o uso de técnicas de analítica visual para aprimorar o aprendizado de máquina na tarefa de anotação de dados. Até onde se tem conhecimento, o uso de projeções do espaço de características e da habilidade cognitiva do ser humano em analisar a distribuição desse espaço ainda não tinham sido exploradas para fins de anotação de dados. Além disso, é proposto um método semi-automático, o qual combina a propagação de rótulos automática com a propagação de rótulos manual feita pelo usuário. Ainda, o presente trabalho estudou o impacto da anotação de dados no espaço projetado em comparação àquela usualmente realizada no espaço latente.

Os resultados preliminares referentes à propagação de rótulos manual feita pelo usuário foram publicados em [7]. Os resultados obtidos utilizando o método semi-automático proposto foi apresentado em [6] (submetido, em revisão).

1.3 Organização do texto

O Capítulo 2 apresenta os trabalhos relacionados aos principais desafios levantados nesta dissertação, no que diz respeito à anotação de dados. O Capítulo 3 apresenta os fundamentos para entender os métodos usados neste trabalho: as Seções 3.1, 3.2, 3.3 e 3.4 cobrem, em detalhes, redes neurais artificiais, técnicas de projeção, análise visual de dados e classificadores de padrões, respectivamente. O Capítulo 4 descreve o método semi-automático proposto para a anotação de dados. Os detalhes sobre configuração experimental, base de dados, protocolo de comparação, resultados experimentais e a discussão dos resultados são apresentados no Capítulo 5. Por fim, as conclusões e principais contribuições acerca do presente trabalho, bem como os trabalhos a serem considerados no futuro são apresentados no Capítulo 6.

Capítulo 2

Trabalhos relacionados

Modelos de ML têm sido extensivamente investigados e empregados para diversos problemas como para classificação de padrões com as conhecidas *support vector machines* [19], o probabilístico *naive bayes* para categorização automática de texto [62], regressão linear para segmentação [59], classificação de imagens com CNNs [30], segmentação semântica com *fully-CNNs* [41] e imagens de alta resolução com *deep-CNNs* [27]. De forma geral, CNNs têm mostrado bons resultados em várias aplicações, como em classificação de imagens/textos [33] e reconhecimento de fala [20], uma vez que tais técnicas são mais genéricas e não necessitam de um conhecimento prévio da aplicação, quando comparadas a técnicas de extração de características tradicionais [33]. Contudo, CNNs requerem uma grande quantidade de amostras anotadas para o seu treinamento [84] e a anotação desses dados baseada na inspeção visual (supervisão) de cada objeto pode ser um processo trabalhoso.

Utilizar um conjunto com poucas amostras supervisionadas para treinamento pode prejudicar a habilidade de um modelo de classificar corretamente novas amostras. Esse problema pode ser causado pelo super-treinamento do modelo, conhecido como *over-fitting* [13, 69]), isto é, quando o modelo aprendeu muito a partir dos dados de treinamento, ou pode ser causado pelo sub-treinamento do modelo, conhecido como *under-fitting* [13], quando o modelo aprendeu muito pouco sobre os dados de treinamento. Ao passo que utilizar um conjunto com muitas amostras não-supervisionadas torna a inspeção e a anotação de amostras inviável para o usuário. Soluções para os problemas de generalização do modelo, *over-fitting* e *under-fitting*, incluem aumento de dados [43] e métodos de regularização [48]. No segundo caso, classificadores semi-supervisionados e métodos de aprendizado ativo têm sido utilizados para propagação de rótulos a partir de pequenos conjuntos supervisionados para grandes conjuntos não-supervisionados.

Tendo em vista os métodos de aprendizado semi-supervisionado, muitos estudos têm explorado a distribuição das amostras em algum espaço de características, a fim de propagar rótulos das amostras supervisionadas para as amostras não-supervisionadas utilizando métodos baseados em grafos [39, 87, 2, 71], aprendizado de métricas [18, 79, 16] e *manifolds* [64, 5, 38, 25]. Outros, ainda, têm investigado redes neurais generativas com muitas camadas escondidas para o aprendizado semi-supervisionado utilizando redes neurais de várias camadas [28], autocodificadoras [53, 66, 35, 54, 17] e generativas adversariais [68, 31, 49, 29].

Usualmente, abordagens de AL exploram o conceito de informatividade de amostra,

do inglês *sample informativeness*. Nessas abordagens, amostras são selecionadas para a supervisão do usuário baseadas em sua informatividade, isto é, o potencial de melhorar o projeto de um classificador a partir do conhecimento do rótulo [63], medido pela confiança de um rótulo atribuído pelo classificador para uma amostra utilizando abordagens baseadas em grafos [45, 67, 11], *support vector machines* [40, 50] e densidade de pontos [36, 37]. Neste contexto, alguns trabalhos utilizam, inclusive, redes com muitas camadas convolucionais [83, 80] e generativas adversariais [26] para tal fim.

Uma vez que abordagens exclusivamente baseadas em AL podem não ser viáveis devido à necessidade do usuário anotar muitas amostras, alguns trabalhos buscam combinar o aprendizado semi-supervisionado e AL para anotação de dados. Neste caso, o aprendizado semi-supervisionado é utilizado em cada iteração do AL para propagar rótulos e selecionar amostras para a supervisão do usuário [63]. A exemplo disso, estudos forneceram para o usuário rotular amostras selecionadas pelo classificador *support vector machines* [89, 22], por grafos [61], por CNNs [60], por *restricted Boltzmann machines* [88] e com discordância de rótulos oferecidos por dois classificadores diferentes [46]. Contudo, tais trabalhos ainda requerem a inspeção e supervisão do usuário de imagem por imagem, além de necessitar de um conhecimento especializado para a supervisão de imagens de determinadas aplicações. Adicionalmente, nenhum desses trabalhos têm combinado a habilidade cognitiva do ser humano em abstrair informação e identificar padrões sem a supervisão do usuário de fato com a habilidade de máquinas em processar dados, considerando ambos com o objetivo de aumentar o número de amostras corretamente anotadas.

Estudos recentes têm investigado o uso de projeções dos espaços de características e análise visual de dados para entender e projetar modelos de ML. Tais estudos consideram as técnicas mencionadas para o entendimento de métodos de projeção [57] e redes neurais convolucionais [58], o aumento de dados interativo [51] e anotação de dados interativa [24, 8, 7] guiada por projeções de espaço de características. Hoferlin *et al.* [24] exploram o usuário não somente para anotar os dados no aprendizado ativo, mas também para entender o modelo do classificador utilizando ferramentas de visualização da informação. Uma dessas ferramentas permite que o usuário apenas visualize (sem interação) uma projeção do espaço de características para auxiliá-lo na compreensão da qualidade do classificador e identificação de áreas rotuladas erroneamente. Bernard *et al.* [8] compararam a anotação interativa de dados utilizando projeções do espaço de características com uma técnica de aprendizado ativo, em que o usuário supervisiona e anota amostras selecionadas por um classificador, que é, então, retreinado para anotar e selecionar mais amostras no espaço de características original. Neste mesmo trabalho, o usuário interage a partir da escolha de diferentes técnicas de projeção e agrupamento. Os autores descobriram que a anotação interativa de dados na projeção do espaço de características é mais eficaz do que o aprendizado ativo.

Em estudos iniciais e pertinentes ao presente trabalho [7], foi mostrado que, quando o usuário propaga rótulos para muitas amostras não-supervisionadas guiado pelo conhecimento de rótulo de poucas delas e da informação visual da distribuição das amostras no espaço de características projetado, o conjunto rotulado resultante de treinamento é mais apropriado do que comparado ao gerado por um classificador semi-supervisionado no espaço de características original. Como consequência, classificadores treinados a partir

desses conjuntos rotulados interativamente podem melhor prever os rótulos das amostras do conjunto de teste (amostras ainda não conhecidas pelo classificador), do que quando treinados a partir dos conjuntos rotulados automaticamente. Contudo, Hoferlin *et al.* [24], Bernard *et al.* [8] e o estudo previamente realizado [7] não *combinaram* abordagens automáticas e interativas para propagação de rótulo, isto é, não consideraram o esforço do usuário para a inspeção e anotação visual de dados.

Desta forma, o presente estudo objetiva preencher tal lacuna. Neste trabalho optou-se por focar no aprendizado semi-supervisionado exclusivamente, a fim de demonstrar a utilidade da interação do usuário com a projeção do espaço de características combinado com a informatividade do classificador para a anotação de dados.

Capítulo 3

Referencial Teórico

3.1 Redes neurais artificiais

As conhecidas Redes Neurais Artificiais, do inglês *Artificial Neural Networks* (ANN), foram desenvolvidas inspiradas no sistema de reconhecimento de padrões do cérebro humano. O cérebro é um órgão altamente complexo capaz de se organizar em estruturas conhecidas como neurônios, que têm como objetivo realizar alguns processamentos, como reconhecimento de padrões, percepção e controle motor. O sistema visual é responsável por fornecer uma representação do ambiente ao redor para que o ser humano tenha informação necessária para interagir com o meio. Desde o nascimento, o cérebro é bem estruturado e tem a capacidade de construir suas próprias regras através de experiências.

De forma similar, uma ANN é uma máquina projetada para modelar a forma como o cérebro realiza tarefas particulares ou funções de interesse. Uma ANN pode ser desenvolvida utilizando componentes eletrônicos ou simulada por um *software* de computador. Para alcançar um bom desempenho, ANNs requerem uma numerosa interconexão entre as células de cômputo, conhecidas como neurônios ou unidades de processamento. A Figura 3.1 apresenta um exemplo de uma rede neural artificial.

3.1.1 Modelo de um neurônio artificial

Em sua versão clássica, o neurônio artificial é a unidade de processamento de informação fundamental para a operação de uma ANN. Um neurônio artificial apresenta principalmente três elementos básicos: as sinapses, uma função de soma e uma função de ativação.

Cada sinapse ou conexão é caracterizada por um peso. Especificamente, um sinal de entrada x_j da sinapse j conectado ao neurônio k é multiplicado pelo peso sináptico w_{kj} . Assim como as sinapses do cérebro, o peso sináptico de um neurônio artificial pode variar em um intervalo de valores negativos e positivos. A função de soma $\sum$ é responsável por somar os sinais de entrada multiplicados pelos pesos das respectivas sinapses do neurônio. Tal operação descrita constitui uma combinação linear. A função de ativação ϕ é um limitador da amplitude da saída de um neurônio para um valor finito. No modelo neuronal, pode-se também considerar um *bias* b_k , que tem o efeito de aumentar ou diminuir a entrada da rede da função de ativação. A Figura 3.2 apresenta um exemplo de um modelo de neurônio artificial.

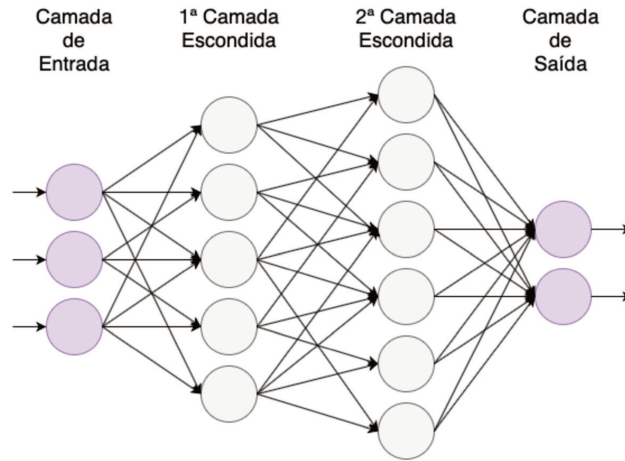


Figura 3.1: Exemplo de uma rede neural artificial. Cada neurônio é representado por um círculo na figura e as ligações entre eles são os pesos. A rede apresenta uma camada de entrada, duas camadas escondidas (neste caso) e uma camada de saída, que determinam o número de classes do problema.

Em termos matemáticos, um neurônio k é descrito por:

$$u_k = \sum_{j=1}^m x_j w_{kj} \quad (3.1)$$

e

$$y_k = \varphi(u_k + b_k), \quad (3.2)$$

em que $x_1, x_2, \dots, x_m$ são os sinais de entrada; $w_{k1}, w_{k2}, \dots, w_{km}$ são os pesos sinápticos do neurônio k ; u_k é a saída do combinador linear em relação aos sinais de entrada; b_k é o **bias**; $\varphi(\cdot)$ é a função de ativação; e y_k é o sinal de saída do neurônio. O uso do *bias* b_k tem efeito de permitir uma transformação afim com relação à saída u_k do combinador linear dada por:

$$v_k = u_k + b_k. \quad (3.3)$$

O *bias* b_k é um parâmetro externo do neurônio artificial k .

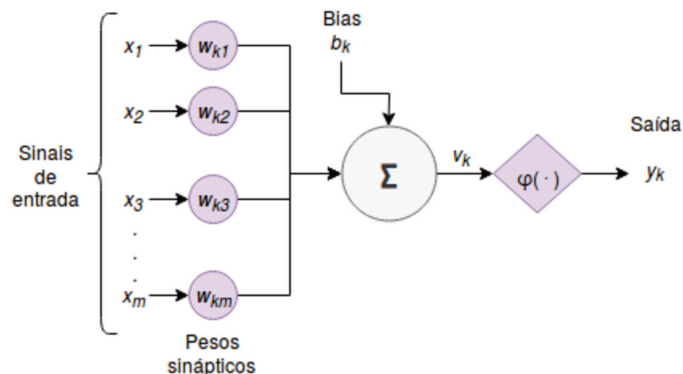


Figura 3.2: Representação de um neurônio artificial.

A função de ativação $\varphi(v)$ define a saída de um neurônio em termos do potencial de ativação v . Atualmente, existem várias funções de ativação, sendo que as mais conhecidas são a função de limiar, função sigmoial e função *Rectified Linear Unit* (ReLU). Para a primeira, a função de limiar, têm-se a saída do neurônio k expresso por:

$$\varphi(v) = \begin{cases} 1, & \text{if } v \geq 0 \\ 0, & \text{if } v < 0. \end{cases} \quad (3.4)$$

em que v é o potencial de ativação do neurônio. O outro tipo de função de ativação conhecida, é a função sigmoial, em que seu gráfico mostra uma forma de s e é uma das formas mais comuns de ativação utilizada para redes neurais artificiais. É definida como uma função estritamente crescente que tem comportamento balanceado entre funções lineares e não-lineares. Um exemplo de função sigmoial é a função logística, definida como:

$$\varphi(v) = \frac{1}{1 + \exp(-av)} \quad (3.5)$$

em que a é um parâmetro de inclinação para a função. Como visto, a função de limiar tem saída entre 0 e 1, mas em alguns casos pode ser desejável que a saída esteja entre -1 e 1. De forma resumida a ativação dada pela função sigmoial pode ser entendida por:

$$\varphi(v) = \begin{cases} 1, & \text{if } v > 0 \\ 0, & \text{if } v = 0 \\ -1, & \text{if } v_k < 0. \end{cases} \quad (3.6)$$

Por fim, a função de ativação ReLU, uma função não-linear, é definida por:

$$f(x) = \max(0, v), \quad (3.7)$$

que desconsidera os valores de ativações v menores do que 0.

3.1.2 Redes neurais autocodificadoras

Um grande problema encontrado nas aplicações relacionadas à classificação e visualização de padrões refere-se a alta dimensionalidade dos dados utilizados, uma vez que são dados do mundo real e, por isso, apresentam muitas características de categorias distintas. Normalmente, o dado de alta dimensionalidade não se dispersa por todo o espaço e é composto por dimensões menores [81]. Para lidar com este tipo de dado, a dimensionalidade precisa ser reduzida. No processo de redução de dimensionalidade, é necessário lidar com alguma perda de informação ao descartar algumas dimensões. Desta forma, o principal objetivo é tentar manter as informações mais relevantes do dado original na versão reduzida de sua representação.

Tradicionalmente, muitos trabalhos utilizam técnicas como a Análise de Componentes Principais (do inglês, *Principal Component Analysis*) ou a Análise de Fator (do inglês, *Factor Analysis*) para redução de dimensionalidade. Embora sejam técnicas conhecidas e muito exploradas, elas são normalmente indesejadas, quando a dimensão de entrada é

muito alta e a representação é esparsa (isto é, muitos zeros), o que demanda abordagens não-lineares [77].

Dentre as técnicas de redução de dimensionalidade não-lineares, alguns estudos ocuparam-se em explorar as Redes Neurais, em virtude de tais redes serem capazes de aprender uma representação não-linear de menor dimensão do espaço de características [4, 32]. Especificamente, uma Rede Neural Autocodificadora, do inglês *Autoencoder Neural Networks* (AuNN), é uma rede neural capaz de codificar os dados de entrada de alta dimensionalidade para uma representação de menor dimensão pelo aprendizado não-supervisionado. Isto se dá por meio da diminuição do número de neurônios da camada escondida em relação aos neurônios das camadas de entrada.

Considerando o tipo mais simples de AuNN, uma rede neural totalmente conectada com apenas uma camada escondida, e um conjunto de dados de treinamento não-supervisionado $x_1, x_2, \dots, x_i$, tal que $i \leq m$, em que m é o número de amostras e $x_i \in \mathbb{R}^N$, a AuNN apresenta uma etapa de *encoder*, em que o dado é codificado em uma representação reduzida e uma etapa de *decoder*, em que o dado é decodificado para uma representação reconstruída. A Figura 3.3, a seguir, ilustra as etapas de *encoder* e *decoder* de uma Rede Neural Autocodificadora com apenas uma camada escondida.

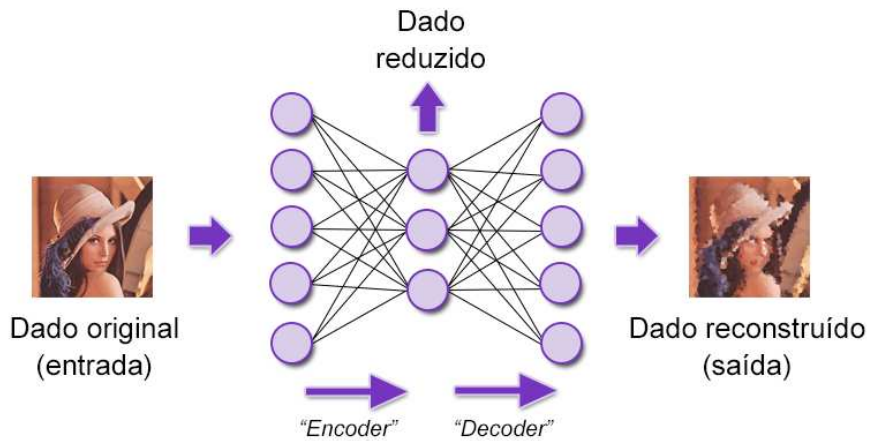


Figura 3.3: Exemplo de uma Rede Neural Autocodificadora.

A etapa de *encoder* representa uma rede codificadora multicamadas, do inglês *multilayer encoder network*, capaz de transformar a alta dimensionalidade dos dados em um código de menor dimensão, cuja função é dada por:

$$r = f(x_i, w), \quad (3.8)$$

em que x_i é a entrada da rede; f , a função codificadora; r , uma representação codificada da entrada x_i mapeada por f ; e w , os pesos. Ainda, a rede Autocodificadora na etapa de *decoder* apresenta uma rede similar, denominada rede decodificadora multicamadas, do inglês *multilayer decoder network*, capaz de reconstruir os dados codificados, seguindo a função:

$$y_i = g(r, v), \quad (3.9)$$

em que r é representação codificada; g é a função decodificadora; e y_i é a reconstrução obtida a partir de r , mapeada por g e com os pesos v . De uma forma geral, uma rede Autocodificadora pode, então, ser entendida como:

$$y_i = g(f(x_i, w), v). \quad (3.10)$$

Desta forma, tal modelo de rede sugere que, a partir de duas redes configuradas inicialmente com pesos aleatórios, w e v , ambas possam ser simultaneamente treinadas por retropropagação para amenizar a diferença (erro) entre o dado inicial e sua versão reconstruída, que, neste caso, espera-se que seja a própria imagem de entrada. Assim, x_i e y_i pertencem ao mesmo domínio e possuem diferença mínima, de forma que $y_i \approx x_i$ e $y_i = x_i + e_i$ com e_i sendo o erro para dada reconstrução. Na retropropagação, a função de custo que se tenta reduzir é dada pelo erro médio quadrático, do inglês *mean squared error* (MSE), e é definida por:

$$MSE = \frac{1}{2} \sum_{i=1}^n \|y_i - x_i\|_2^2, \quad (3.11)$$

que resulta no erro que é retropropagado pela rede, a fim de atualizar seus pesos.

Em suma, o problema de treinamento de redes Autocodificadoras é, na verdade, uma tarefa de otimização, onde a ideia consiste em encontrar o conjunto de parâmetros (pesos) da rede que minimiza o erro de reconstrução entre o dado de entrada e o de saída. Assim sendo, mínimos locais podem ser um problema quando essas redes são inicializadas com pesos numericamente grandes. Já para pesos pequenos, seu treinamento pode ser inviável para muitas camadas escondidas, dificultando a sua otimização, isto é, o gradiente da função de erro cai para zero antes de corrigir os pesos das camadas iniciais).

Diferentes variações e tipos de Redes Neurais Autocodificadoras surgiram na literatura diante do amplo uso de redes neurais em diversas aplicações como, por exemplo, para redução de ruído, representações de código esparsa, aprendizado de grande quantidade de características em imagens, construção e reconstrução de novas imagens, bem como para a criação de dados genéricos.

A primeira ideia de rede Autocodificadora, também conhecida como Rede Auto-associadora, do inglês *Auto-associator*, surgiu em 1987. Sua arquitetura possuía uma configuração totalmente conectada incluindo uma camada de entrada, uma camada escondida (com número de neurônios menor do que o da camada de entrada) e a camada de saída de mesmo tamanho que a camada de entrada [4, 32]. Tal rede foi utilizada em um algoritmo de aprendizado específico desenvolvido para a etapa de pré-treinamento de redes neurais de modo não-supervisionado. Logo, outros estudos a cerca da técnica de reduzir a camada escondida consolidaram o método não-supervisionado. O conceito da arquitetura de redes Autocodificadoras [9] foi explorado com o intuito de mostrar que as unidades escondidas da rede são responsáveis por aprender funções não-lineares.

A Autocodificadora Esparsa é uma variação das redes Autocodificadoras, em que se é adicionada uma constante de regularização de esparsidade como penalidade no treinamento junto ao erro de reconstrução [52]. Assim, essa constante objetiva forçar a esparsidade de código, ou seja, que apenas alguns neurônios de cada camada sejam ativados.

O problema desta abordagem consiste em ativar sempre os mesmos poucos neurônios, enquanto os outros são sempre desativados. A alternativa encontrada para tal questão foi sugerida por Ranzatto *et al.* [52] e refere-se a adicionar um módulo de esparsidade, ou seja, uma não-linearidade denominada *Sparsifying Logistic*, entre as redes codificadoras e decodificadoras. Tal função logística apresenta um comportamento adaptativo capaz de rastrear a média da entrada. O intuito da esparsidade nesse contexto é fornecer um vetor reduzido codificado que seja um vetor esparso e facilmente separável.

Com os ganhos obtidos a partir de arquiteturas de muitas camadas, a técnica de utilizar uma grande quantidade de camadas escondidas também é aplicada às Redes Autodificadoras [78]. A ideia é que os dados de saída de uma camada sejam fornecidos como entrada para a camada seguinte e, assim, aprender uma representação hierárquica e relevante da informação a cada camada. Por este motivo, Redes Neurais Autocodificadoras com arquiteturas de muitas camadas escondidas forneceram bons resultados quando utilizadas para pré-treinar redes não-supervisionadas de muitas camadas [21], como Máquinas Restritas de Boltzmann e Máquinas Restritas de Boltzmann Empilhadas [1].

Como mencionado anteriormente, uma AuNN tradicional apresenta uma arquitetura de rede em que seus neurônios são totalmente conectados de uma camada escondida para outra. Embora muitos trabalhos tenham explorado tal configuração, estudos recentes explorando as técnicas de aprendizado de muitas camadas escondidas mostraram ser conveniente a substituição da camada de uma rede totalmente conectada por uma camada de convolução, muito utilizada em Redes Neurais Convolucionais. Assim, têm-se uma Rede Neural Autocodificadora Convolucional [42] com a presença de camadas de convolução. Segundo o trabalho de Masci *et al.* [42], tal proposta é promissora no contexto da extração hierárquica de características para entradas de alta dimensão.

A seguir são descritas em detalhes as principais operações necessárias para o uso de Redes Neurais Autocodificadoras Convolucionais, como a camada de convolução e a operação de *pooling*.

3.1.3 Operação de convolução

As camadas de convolução são usualmente utilizadas para o processamento de imagens e sinais, além de constituírem as conhecidas Redes Neurais Convolucionais. Tais redes têm obtido sucesso em aplicações práticas no processamento de dados e nela empregam a operação matemática denominada convolução. A convolução é um tipo especializado de operação linear.

Considerando que uma imagem pode ser definida como $\hat{I} = (D_I, \mathbf{I})$, em que D_I é o domínio da imagem com $D_I \subset \mathbb{Z}^n$ e definido pelas coordenadas $x = 0, 1, \dots, n_x - 1$ e $y = 0, 1, \dots, n_y - 1$. O vetor de atributos (escalares) é entendido como:

$$\mathbf{I}(p) = \{I_1(p), I_2(p), \dots, I_m(p)\}, \quad (3.12)$$

em que cada elemento $p = (x_p, y_p) \in D_I$ é um *pixel* (elemento da imagem) de m bandas. Um *pixel*, de forma genérica, pode ser denominado como *spel* (elemento do espaço $\mathbb{Z}^n$), onde n é a dimensão espacial, neste caso, $n = 2$.

A relação de adjacência $A(p)$, centrada em p de tamanho $S_A \times S_A$ com $A \subset D_I \times D_I$

de uma imagem $\hat{I}$, é definida como uma relação binária entre *spels* p e q , com $q \in A(p)$, se $\max(|x_q - x_p|, |y_q - y_p|) \leq (S_A - 1)/2$. Assim, um *kernel* $\hat{K} = (A, \mathbf{K})$ é uma imagem de pesos $K(q - p) \in \{w_1, w_2, \dots, w_d\}$ definidos sobre os deslocamentos em $A(p)$. Assim, desloca-se uma imagem $\hat{K}$ sobre a imagem $\hat{I}$, em que A em $\hat{K}$ tem origem $(0, 0)$ referente ao *spel* p .

Desta forma, a convolução $\hat{I} * \hat{K}$ entre $\hat{I} = (D_I, I)$ e $\hat{K} = (A, K)$, requer a reflexão $\hat{K}'$ de $\hat{K}$:

$$\hat{K}' = (A', K) \quad (3.13)$$

com

$$A' = \{(-dx_1, -dy_1), (-dx_2, -dy_2), \dots, (-dx_d, -dy_d)\} \quad (3.14)$$

em que $d = |A(p)|$, com $K(p - q) \in \{w_1, w_2, \dots, w_d\}$.

Assim, essa convolução $\hat{J} = (D_J)$ em cada *pixel* p é dada por:

$$J(p) = \sum_{\forall q \in A'(p)} I(q)K(p - q) = \sum_{i=1}^d I(q_i)w_i. \quad (3.15)$$

para todo $p \in D_I$, em que q_i com $i = \{1, 2, \dots, d\}$ são adjacentes de p em $A'(p)$. A rigor, a imagem $\hat{K}$ desloca-se desde o infinito negativo até o infinito de cada eixo, mas na prática força-se que $D_J = D_I$.

Considerando, agora, a convolução em duas dimensões entre a imagem $\hat{I} = (D_I, I)$ e $\hat{K} = (D_K, K)$ que define a imagem filtrada $\hat{J} = (D_J, J)$ como:

$$J(x_p, y_p) = \sum_{y=-\infty}^{\infty} \sum_{x=-\infty}^{\infty} I(x, y)K(-x + x_p, -y + y_p), \quad (3.16)$$

em que para todo $p \in D_J$ e (x_p, y_p) é a origem da imagem móvel nos deslocamentos dos *pixels* em que q_i com $i = 1, 2, 3, \dots, d$ são os adjacentes de p em $A'(p)$. A imagem filtrada $J(x_p)$ só existe na interseção entre $\hat{I} = (D_I, I)$ e $\hat{K} = (D_K, K)$, ou seja, se D_I apresentar $n_x \times n_y$ *pixels* e D_K apresentar $m_x \times m_y$ *pixels*, então D_J têm $(n_x + m_x - 1) \times (n_y + m_y - 1)$ *pixels*. A convolução calcula a soma dos produtos entre o valor de $\hat{I}$ de cada *spel* adjacente e seu peso em $\hat{K}$. A Figura 3.4 ilustra a operação de convolução.

Um banco de filtros pode ser definido como:

$$\hat{K} = (A, \mathbf{K}_i), \quad (3.17)$$

com $i = \{1, 2, 3, \dots, n\}$ em que n é o número de filtros do banco e que $\mathbf{K}_i$ é um filtro do banco e $\mathbf{K}_i = (K_{1,i}, K_{2,i}, K_{3,i}, \dots, K_{m,i})$. Desta forma a convolução com um banco de filtros é dada por:

$$J_i(p) = \sum_{\forall q \in A'(p)} \mathbf{I}(q) \cdot K_i(p - q) \quad (3.18)$$

em que i é cada banda da imagem $\hat{J} = (D_J, \mathbf{J})$ com $\mathbf{J} = (J_1, J_2, \dots, J_N)$ e $\cdot$ representa

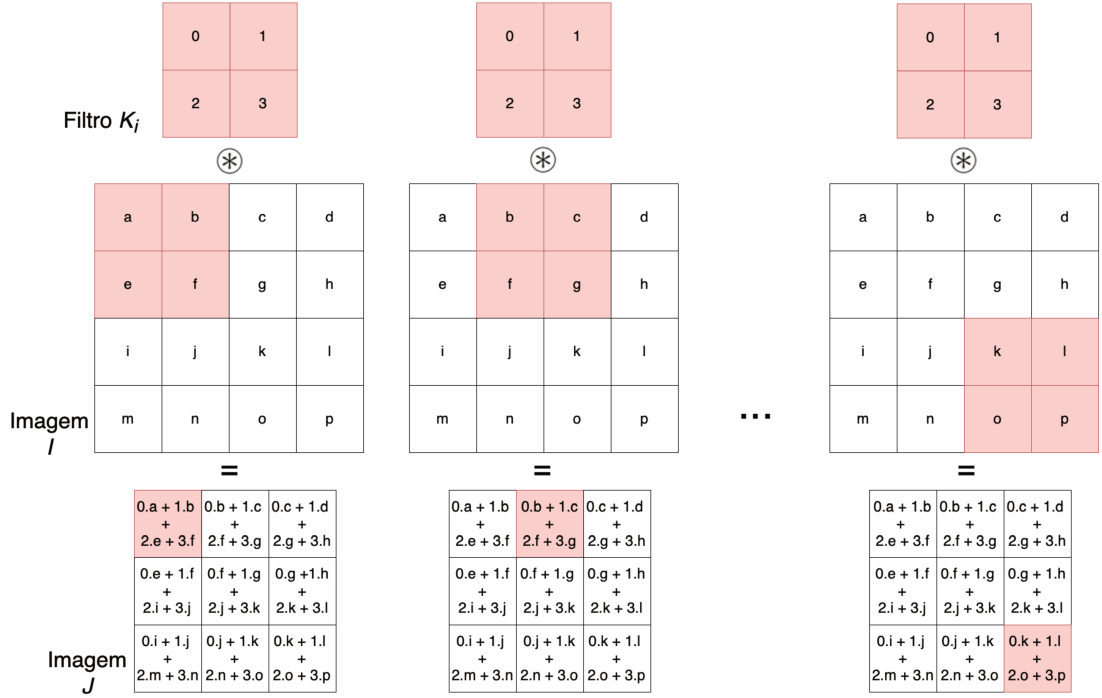


Figura 3.4: Operação de convolução. Neste exemplo, o filtro K_i de tamanho 2×2 , menor do que a imagem I , é aplicado na primeira posição $(0,0)$ na imagem I . Com $stride=1$, ou seja, o deslocamento do filtro sobre a imagem, o filtro é aplicado na posição seguinte à direita na imagem.

o produto interno. Posteriormente os valores de $J_i(p)$ ainda são submetidos a uma função de ativação.

3.1.4 Operação de *pooling*

A operação de *pooling* pode ser definida considerada uma adjacência C circular de raio r e centrada no *pixel* p e aplicada a cada d *pixels*, também é conhecido como *stride*. Para valores de $d > 1$, a resolução espacial da imagem é reduzida.

Assim, o *pooling*, que resulta em uma nova imagem $\hat{P} = (D_P, \mathbf{P})$ com $\mathbf{P} = (P_1, P_2, P_3, \dots, P_N)$, pode ser definido por:

$$P_i(p) = \sqrt[\alpha]{\sum_{q \in C(p)} I_i(q)^\alpha} \quad (3.19)$$

em que $i = \{1, 2, \dots, N\}$ e α controla a sensibilidade da operação. Para valores altos de α , maior importância é dada ao maior coeficiente de imagem de entrada. Para valores altos de α , têm-se o *max pooling*. A operação de *max pooling* na imagem I pode ser resumida como ilustra a Figura 3.5.

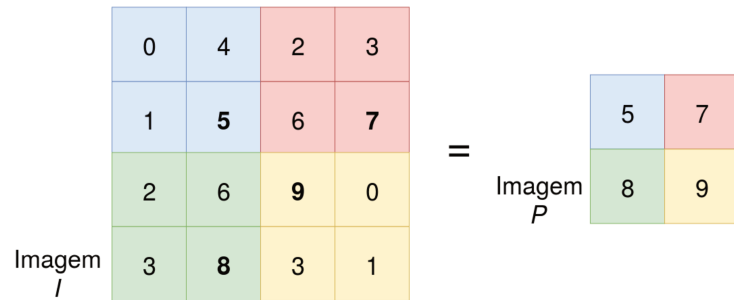


Figura 3.5: A operação de *pooling* varre uma imagem I a partir da posição $(0,0)$ com uma janela, neste caso, de 2×2 e $\text{stride} = 2$, e o maior valor da imagem em cada janela é utilizado para criar a imagem P resultante.

3.2 Técnicas de projecção

A visualização de dados de alta dimensionalidade, isto é, dados com uma quantidade de informação (ou características) considerável, é um problema que cerca diferentes aplicações. Para o entendimento desses dados, duas abordagens podem ser exploradas: técnicas de visualização e métodos de redução de dimensionalidade. A primeira, consiste em ferramentas para mostrar duas ou mais dimensões de um dado para que o ser-humano possa interpretá-lo. Dentre tais ferramentas, por exemplo, estão aquelas baseadas em *pixels*, representações das dimensões de um dado como vértices de um grafo ou na exibição um conjunto de imagens significativas. Ainda assim, tais ferramentas estão limitadas a problemas do mundo real, em que é necessário interpretar milhares de dados de alta dimensão.

Desta forma, a segunda abordagem, que consiste em estudar métodos de redução de dimensionalidade, tem como principal objetivo converter o conjunto de dados de alta dimensionalidade $X = \{x_1, x_2, \dots, x_n\}$ em um dado de duas ou três dimensões $Y = \{y_1, y_2, \dots, y_n\}$ que possa ser mostrado como um gráfico de dispersão (*scatterplot*, uma ferramenta de visualização). Tal redução deve ser feita de forma preservar ao máximo as estruturas relevantes do dado de alta dimensionalidade ao ser mapeado para uma menor dimensão. Dentre as técnicas de redução de dimensionalidade conhecidas estão os métodos lineares *Principal Components Analysis* [23] e *Multidimensional Scaling* [74], que buscam manter dados distintos (de informações distintas) distantes no espaço mapeado em menor dimensão. Em contrapartida, existem aquelas técnicas conhecidas que são exemplos não-lineares de redução de dimensionalidade, como *Curvilinear Components Analysis* [12], *t-distributed Stochastic Neighbor Embedding* [76] ou *Isomap* [73], que objetivam preservar, principalmente, as estruturas locais do dado em sua representação de baixa dimensão.

Em dados que apresentam milhares de características, o uso de técnicas de redução de dimensionalidade não-lineares para o espaço de duas dimensões ainda é a abordagem mais apropriada para entender a distribuição de dados [21, 76]. Dentre elas, uma técnica conhecida é o algoritmo de incorporação de vizinhança estocástica da distribuição “t”, do inglês *t-distributed Stochastic Neighbor Embedding* (t-SNE) [75]. Optou-se por focar o presente trabalho em tal técnica de redução de dimensionalidade, devido a estudos recentes [58, 57, 86] que indicam uma forte relação entre a separação de classes

no espaço de características de alta dimensão e suas amostras correspondentes no espaço de características gerado pelo t-SNE. Adicionalmente, foi mostrado que o algoritmo t-SNE pode preservar a distribuição de dados no espaço projetado melhor do que outros algoritmos [76].

3.2.1 t-SNE

A técnica t-SNE é uma técnica capaz de calcular visualizações de um dado de alta dimensionalidade, fornecendo para cada amostra do dado um ponto correspondente mapeado para, normalmente, duas ou três dimensões. Tal técnica é uma variação do algoritmo de incorporação de vizinhança estocástica, do inglês *Stochastic Neighbor Embedding*, que apresenta fácil otimização e produz melhores visualizações pela redução da tendência de aglomerar pontos no centro da projeção. Em suma, os dados de alta dimensionalidade são convertidos em uma matriz de similaridades de pares, de forma a capturar melhor as estruturas locais dos dados de alta dimensão, ao mesmo tempo que revelam estruturas globais pela presença de agrupamentos em várias escalas. Assim, os dados de alta dimensão são mapeados para um espaço de menor dimensão para que possibilite a identificação de padrões pelos agrupamentos de dados baseado na similaridade entre pontos de características distintas. O t-SNE minimiza a divergência entre duas distribuições: a distribuição que mede as similaridades de pares de objetos de entrada e a distribuição que mede as similaridades de pares dos pontos correspondentes na dimensão reduzida na incorporação. As formulações matemáticas referentes ao algoritmo foram baseadas em [76].

Dado um conjunto de dados (com alta dimensionalidade) de entrada de objetos $D = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \dots, \mathbf{x}_n\}$ e uma função $d(\mathbf{x}_i, \mathbf{x}_j)$ que computa as distâncias entre pares de objetos, neste caso, a distância Euclidiana $d(\mathbf{x}_i, \mathbf{x}_j) = \|\mathbf{x}_i - \mathbf{x}_j\|$, o objetivo é aprender uma incorporação de s dimensões, em que cada objeto é representado por um ponto $E = (\mathbf{y}_1, \mathbf{y}_2, \mathbf{y}_3, \dots, \mathbf{y}_n)$, com $\mathbf{y}_i \in \mathbb{R}^s$ em que valores típicos para s são dois ou três. Desta forma, o t-SNE define duplas probabilidades p_{ij} que medem a similaridade de pares entre os objetos $\mathbf{x}_i$ e $\mathbf{x}_j$ pela simetria de duas probabilidades condicionais:

$$p_{j|i} = \frac{\exp(-d(\mathbf{x}_i - \mathbf{x}_j)^2 / 2\sigma_i^2)}{\sum_{k \neq i} \exp(-d(\mathbf{x}_i - \mathbf{x}_k)^2 / 2\sigma_i^2)}, \quad p_{i|i} = 0, \quad (3.20)$$

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}. \quad (3.21)$$

Nas equações acima, a largura de banda dos *kernels* Gaussianos σ_i é definida de forma que a perplexidade da distribuição condicional P_i seja igual a perplexidade pré-definida u . Como resultado um valor ótimo de σ_i varia por objeto: em regiões em que o espaço do dado apresenta maior densidade de pontos, σ_i tende a ser menor do que em regiões em que o espaço do dado com menor densidade. O valor ótimo de σ_i para cada objeto de entrada pode ser encontrado utilizando uma busca binária ou utilizando um método robusto para encontrar raízes.

Na incorporação E de s dimensões, as similaridades entre dois pontos $\mathbf{y}_i$ e $\mathbf{y}_j$, isto é, dos pontos de dimensão reduzida de $\mathbf{x}_i$ e $\mathbf{x}_j$, são medidas utilizando um *kernel* de cauda pesada normalizado. Especificamente, a incorporação de similaridade q_{ij} entre dois pontos

$\mathbf{y}_i$ e $\mathbf{y}_j$ é computada como um *kernel* de *t-Student* normalizado como um único grau de liberdade:

$$q_{ij} = \frac{(1 + \|\mathbf{y}_i - \mathbf{y}_j\|^2)^{-1}}{\sum_{k \neq l} (1 + \|\mathbf{y}_k - \mathbf{y}_l\|^2)^{-1}}, \quad q_{ii} = 0. \quad (3.22)$$

As caudas pesadas do *kernel* de *t-Student* normalizado permitem distintos objetos de entrada $\mathbf{x}_i$ e $\mathbf{x}_j$ serem modelados por respectivos objetos de dimensão reduzida $\mathbf{y}_i$ e $\mathbf{y}_j$ que estão consideravelmente longe um dos outros. Isso se faz desejável, uma vez que é capaz de criar mais espaço para modelar de forma acurada as distâncias de pares pequenas, isto é, a estrutura local do dado, na incorporação de dimensão reduzida.

Os locais dos pontos incorporados $\mathbf{y}_i$ são determinados pela minimização da divergência de *Kullback-Leibler* entre duplas distribuições P e Q :

$$C(E) = KL(P \parallel Q) = \sum_{i \neq j} p_{ij} \log \frac{p_{ij}}{q_{ij}} \quad (3.23)$$

Devido a assimetria da divergência de *Kullback-Leibler*, a função objetivo foca em modelar valores altos de p_{ij} (objetos similares) para valores altos de q_{ij} (pontos próximos no espaço incorporado). A função objetivo é não-convexa na incorporação E e é tipicamente minimizada pela descida ao longo do gradiente:

$$\frac{\delta C}{\delta \mathbf{y}_i} = 4 \sum_{j \neq i} (p_{ij} - q_{ij}) q_{ij} Z(\mathbf{y}_i - \mathbf{y}_j), \quad (3.24)$$

em que o termo de normalização Z é definido como $Z = \sum_{k \neq l} (1 + \|\mathbf{y}_k - \mathbf{y}_l\|^2)^{-1}$.

Uma versão simplificada da implementação [75] do algoritmo t-SNE pode ser dada de acordo com o Algoritmo 1. Primeiramente, o algoritmo calcula a probabilidade da similaridade dos pontos no espaço de alta dimensão e calcula a probabilidade de similaridade dos pontos no espaço correspondente de dimensão reduzida. A similaridade dos pontos é calculada como uma probabilidade condicional em que um ponto A seria escolheria o ponto B como seu vizinho se vizinhos fossem escolhidos na proporção de suas densidades de probabilidade sob uma Gaussiana (distribuição normal) centrada em A. Então, o algoritmo tenta minimizar a diferença entre tais probabilidades condicionais (ou similaridades) no espaço de alta e reduzida dimensões para uma representação perfeita dos dados no espaço de menor dimensão. A fim de medir a minimização da soma das diferenças das probabilidades condicionais, o t-SNE minimiza a soma da divergência de *Kullback-Leibler* de todos os pontos utilizando uma decida de gradiente. A divergência de *Kullback-Leibler* é uma medida de como uma distribuição de probabilidade diverge de uma segunda distribuição de probabilidade esperada.

Algoritmo 1: Versão simplificada do algoritmo t-SNE.

Entrada: Conjunto de dados $X = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \dots, \mathbf{x}_n\}$, perplexidade u , número de iterações T , taxa de aprendizado η e *momentum* $\alpha(t)$.

Saída: Representação de dimensão reduzida $Y^{(T)} = \{\mathbf{y}_1, \mathbf{y}_2, \mathbf{y}_3, \dots, \mathbf{y}_n\}$.

- 1 Computar as similaridades de pares $p_{j|i}$ com perplexidade u (Equações 3.20 e 3.21);
 - 2 Definir a amostra inicial da solução $Y^{(0)} = \{\mathbf{y}_1, \mathbf{y}_2, \mathbf{y}_3, \dots, \mathbf{y}_n\}$ de $\mathbb{N}(0, 10^{-4}\mathbb{I})$;
 - 3 **para** $t = 1$ *até* T **faça**
 - 4 Computar as similaridades de pares q_{ij} (Equação 3.22);
 - 5 Computar o gradiente $\frac{\delta C}{\delta \mathbf{y}_i}$ (Equação 3.24);
 - 6 Definir $Y^{(t)} = Y^{(t-1)} + \eta \frac{\delta C}{\delta Y} + \alpha(t)(Y^{t-1} - Y^{t-2})$.
 - 7 **fim**
-

3.3 Analítica visual de dados

A análise das informações extraídas de imagens tem sido explorada em vários problemas nos contextos de segmentação e classificação de imagens. Uma técnica conhecida e capaz de auxiliar na análise de dados trata-se da analítica visual, ou seja, a visualização interativa da informação. Em [55], Rauber enfatiza a importância da analítica visual aplicada à classificação de imagens por Redes Neurais Artificiais.

Suplementarmente, Rauber *et al.* [56] demonstraram o potencial de duas abordagens para a visualização de ANNs capazes de fornecer resultados visuais intuitivos utilizando técnicas de redução de dimensionalidade. A primeira abordagem corresponde à exploração dos relacionamentos entre as representações alternativas de observações aprendidas por ANNs, enquanto a segunda visa explorar o relacionamento entre os neurônios artificiais.

A primeira abordagem explora a projeção das amostras de treinamento com a informação de rótulo representada por cores diferentes, isto é, cada cor representa uma classe distinta no domínio supervisionado. Desta forma, a intenção é visualizar as amostras no espaço de características e analisar a separabilidade das classes num espaço de menor dimensão, isto é, o quanto amostras de classes diferentes se distanciam e não se misturam. Segundo Rauber *et al.* [56], uma boa separabilidade das amostras no espaço de características indica uma melhor eficácia de classificação. A Figura 3.6 foi reproduzida de seu estudo para elucidar a técnica. Nela, características de treinamento e teste são projetadas em duas dimensões. Em (a) e (b), são projetadas de todas as 500 características presentes em um dado. Em (c) e (d), são projetados um subconjunto de 20 características. Diferentes cores na projeção referem-se a diferentes classes. A partir da figura, pode-se perceber que para a projeção com todas as características existe uma mistura entre as cores (classes) tanto no conjunto de treinamento, quanto no teste. Ao projetar apenas 20 das características em questão, é notável uma maior separabilidade entre pontos de classes distintas e maior agrupamentos de pontos de mesma classe para os conjuntos de treinamento e teste.

Com relação a segunda abordagem, estudos recentes [72, 86] mostram resultados promissores no que diz respeito ao uso de técnicas que exploram a informação visual em redes neurais. Tais técnicas buscam investigar a influência de determinados neurônios pela sua alta ou baixa ativação quando do processamento dos dados de entrada, bem como seu

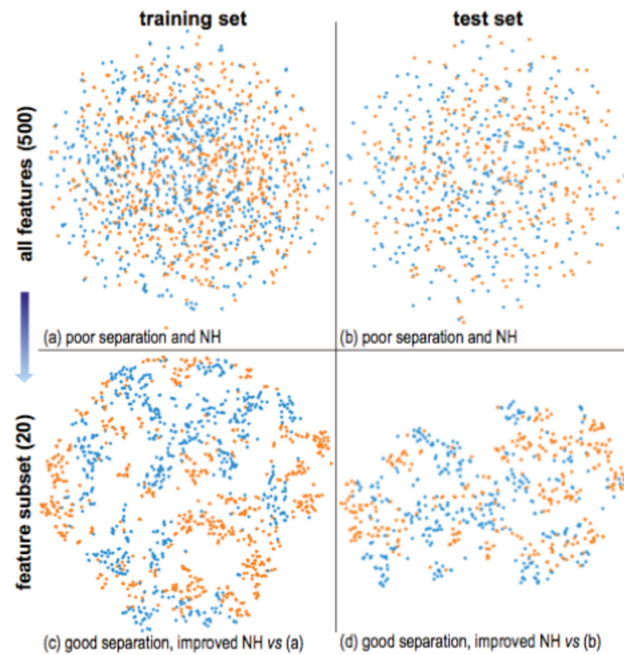


Figura 3.6: Conjunto de treinamento e teste. Projeção das características. (Imagem retirada do trabalho de Rauber *et al.* [56])

desempenho no processo de classificação, por exemplo, para intervenção de tais neurônios. Rauber *et al.* [56] mostraram que é possível visualizar interativamente as ativações dos neurônios de cada camada durante o processo de aprendizado.

A Figura 3.7 foi reproduzida de seu estudo para elucidar a técnica. Nela, subconjuntos de características de treinamento são projetado em duas dimensões. Em (a) e (b), são projetadas as observações de 20 características previamente selecionadas. Em (c) e (d), são projetadas as observações de 19 características, ou seja, uma característica foi removida. A visualização apresentada permite observar o espaço de características utilizando uma técnica que preserva a similaridade entre as características, ou seja, as características de maior proximidade são as mais parecidas entre si. Na figura, as características mais escuras, são aquelas de maior relevância para a classificação do problema (ativadas), e as mais claras, aquelas que não são relevantes para o problema. Desta forma, é possível observar que existe uma característica de cor mais escura, ou seja, que foi considerada relevante para o problema, mas que não está agrupada com as demais. O trabalho mostrou que a remoção dessa característica que é ativada e não tem semelhança com as demais aumenta a acurácia da classificação no conjunto de teste.

3.4 Classificadores de padrões

Ações, como o reconhecimento de um rosto, compreensão de palavras ditas, leitura de caracteres escritos a mão ou identificação de uma chave no bolso pelo toque, fazem parte das ações complexas do dia-a-dia de seres humanos que fundamentam o reconhecimento de padrões [14]. Ao longo dos anos, os seres humanos desenvolveram sistemas cognitivos sofisticados para o reconhecimento de padrões, definido como o ato de, a partir de

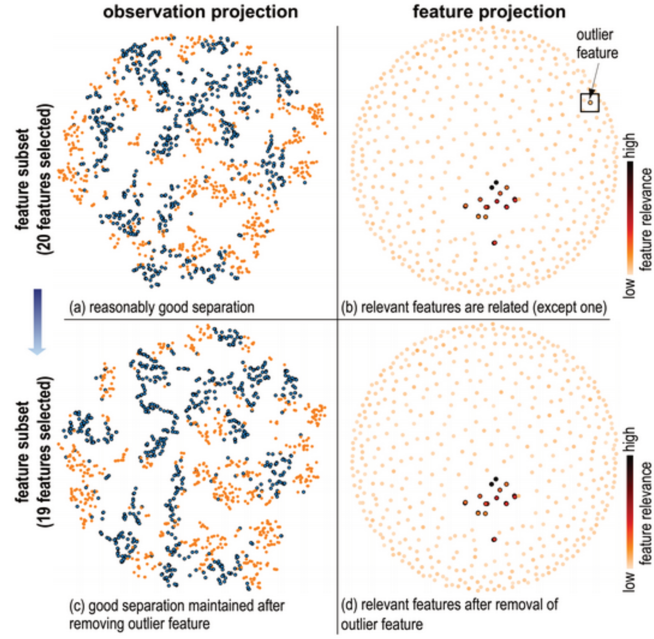


Figura 3.7: Conjunto de treinamento. Projeção da observação e das características, para características selecionadas. (Imagem retirada do trabalho de Rauber et al. [56])

dados brutos, executar uma ação baseada na categoria de um dado padrão. A inteligência artificial buscou, e continua buscando, desenvolver máquinas e sistemas para o reconhecimento desses padrões de forma automática, como, por exemplo, para aplicações de reconhecimento visual.

Em suma, um sistema de classificação de padrões para a aplicações de reconhecimento visual, pode ser explicado da seguinte forma: uma câmera captura imagens de um objeto de interesse, os sinais obtidos pela câmera são pré-processados e, então, enviados a um extrator de características, que tem como objetivo sintetizar o dado de acordo com algumas características ou propriedades, produzindo um vetor de características. Um vetor de características $\mathbf{x}$ para um dado objeto pode ser representado como:

$$\mathbf{x} = \begin{bmatrix} x_1, \\ x_2, \\ x_3, \\ \dots \\ x_i \end{bmatrix} \quad (3.25)$$

em que cada valor de x em $x_1, x_2, x_3, \dots, x_i$, com $x_i \in \mathbb{R}^n$ e $i \leq n$ representa uma característica diferente, e n , a dimensão do vetor de características, isto é, do espaço de características. Um vetor de características pode ser entendido como um ponto no espaço. Por fim, essas características do dado sintetizado são passadas a um classificador que avalia a evidência apresentada e dá uma decisão final de acordo com o domínio do objeto de interesse.

O classificador de padrões aprende a partir de diferentes exemplos de treinamento de

múltiplos objetos dentro do domínio de aplicação. Desta forma, o classificador é treinado e pode, a cada novo exemplo (agora, de um conjunto de teste), decidir se o exemplo é de determinada categoria ou não. Em suma, em problemas de classificação, é esperado que o classificador especifique em qual das k categorias um dado de entrada pertence. Os métodos que incorporam informações de amostras de treinamento no projeto de um classificador utilizam-se do aprendizado. O algoritmo de aprendizado produz uma função $f : \mathbb{R}^n \rightarrow \{1, 2, 3, \dots, k\}$. Quando $y = f(\mathbf{x})$, o modelo atribui uma entrada descrita por um vetor $\mathbf{x}$ a uma categoria identificada por um número codificado y . O aprendizado envolve uma forma do algoritmo reduzir o erro a partir de dados de treinamento. O aprendizado do classificador pode se dar de diferentes formas, segundo a disponibilidade de informação de classe: supervisionado, semi-supervisionado e não-supervisionado. Abaixo os aprendizados supervisionados e semi-supervisionados são descritos em detalhe.

3.4.1 Aprendizado supervisionado

Uma das formas comumente exploradas é o aprendizado supervisionado, em que é fornecido um rótulo ou custo para determinado padrão de categoria em um conjunto de amostras de treinamento. O objetivo é que o classificador seja capaz de reduzir a soma dos custos para tais padrões. Dentre os classificadores difundidos na literatura, foram escolhidos dois.

Máquinas de vetores de suporte

Dentre os principais abordagens considerados para a tarefa de reconhecimento de padrões e aprendizado de máquina, Máquinas de Vetores de Suporte, do inglês *Support Vector Machines* (SVMs) [19] é uma das mais utilizadas, além de obter resultados significativos na área. Tal algoritmo tem como principal objetivo aprender os parâmetros de uma função a partir do dado de treinamento, que consiste em pares (x, y) de objetos de entrada e saída desejada. Depois de treinado o algoritmo, a função gerada pode ser utilizada para prever a saída de um novo objeto. O SVM é um classificador linear em que a função de classificação é um hiperplano no espaço de características. Dentre suas propriedades, a alta capacidade de generalização pela maximização da margem e aprendizado eficiente de funções não-lineares pelo truque de *kernel* (*kernel trick*). A fundamentação teórica, bem como as formulações matemáticas foram baseadas em [85].

Máquinas de vetores de suporte linear

Primeiramente, é necessário considerar o problema mais simples, no caso em que existe apenas duas categorias para a classificação: um problema binário. Para tais aplicações, um classificador SVM binário é necessário para discriminar pontos no espaço de características de duas categorias distintas. Cada ponto (amostra) é representado por um vetor de n dimensões e pertence a uma das duas classes. Um classificador linear separa tais pontos com classes distintas com um hiperplano, de forma que cada cor fique de um lado do hiperplano em questão. A principal diferença do SVM dentre os classificadores lineares se faz presente a medida que o algoritmo seleciona o hiperplano que apresenta a maior margem entre o hiperplano e os pontos de categorias distintas, a fim de obter a maior

separação entre categorias. Essa margem representa a soma da menor distância do hiperplano até o dado mais próximo de cada categoria. A Figura 3.8 exemplifica a separação de pontos de categorias distintas pelo SVM.

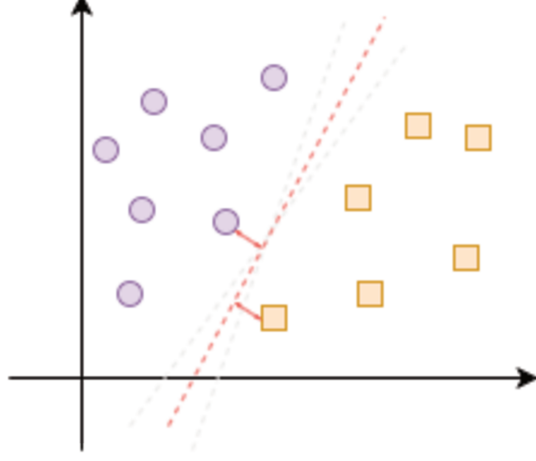


Figura 3.8: Separação de categorias distintas (representadas pelos círculos e quadrados) pelo hiperplano no espaço de características.

No treinamento do SVM linear, os pontos P do conjunto de treinamento podem ser expressados matematicamente da seguinte forma:

$$P = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), (\mathbf{x}_3, y_3), \dots, (\mathbf{x}_m, y_m)\}, \quad (3.26)$$

em que $\mathbf{x}_i$ é um vetor real de n dimensões, y_i é um valor 1 ou -1 referente a classe que o ponto $\mathbf{x}_i$ pertence e m o número de amostras no conjunto de treinamento. A função de classificação $f(\mathbf{x})$ do SVM é dada por:

$$f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} - b, \quad (3.27)$$

em que w é o vetor de pesos e b é o *bias* (um influenciador), que é computado pelo SVM durante o processo de treinamento.

Para classificar corretamente o conjunto de treinamento, $f(\cdot)$ deve retornar números positivos para pontos positivos e, caso contrário, números negativos, isso é, para cada ponto $\mathbf{x}_i$ em P ,

$$\begin{aligned} \mathbf{w} \cdot \mathbf{x}_i - b &> 0, & \text{se } y_i = 1 \text{ e} \\ \mathbf{w} \cdot \mathbf{x}_i - b &< 0, & \text{se } y_i = -1, \end{aligned} \quad (3.28)$$

e que pode ser entendida como:

$$y_i(\mathbf{w} \cdot \mathbf{x}_i - b) > 0, \quad \forall (\mathbf{x}_i, y_i) \in P. \quad (3.29)$$

Se existir uma função linear f que classifique corretamente cada ponto em P ou satisfaça a Equação 3.29, então P é um problema linearmente separável e existe f que

satisfaça:

$$y_i(\mathbf{w} \cdot \mathbf{x}_i - b) \geq 1, \quad \forall(\mathbf{x}_i, y_i) \in P, \quad (3.30)$$

de forma que as distância dos pontos mais próximos ao hiperplano e de classes distintas sejam mínimas. A distância do hiperplano a um vetor $\mathbf{x}_i$ é dada por:

$$\frac{|f(\mathbf{x}_i)|}{\|\mathbf{w}\|}, \quad (3.31)$$

então a margem ϵ , de acordo com a Equação 3.30 que $f(\mathbf{x}_i) = 1$, pode ser entendida como:

$$\epsilon = \frac{1}{\|\mathbf{w}\|}. \quad (3.32)$$

Assim, os vetores mais próximo que satisfaçam a Equação 3.30 são denominados *vetores de suporte*. Uma vez que maximizar a margem ϵ significa minimizar $\|\mathbf{w}\|$, o treinamento do SVM se torna um problema de otimização com restrição, em que o objetivo é minimizar:

$$C(\mathbf{w}) = \frac{1}{2} \|\mathbf{w}\|^2, \quad (3.33)$$

desde que $C(\mathbf{w})$ esteja sujeita à Equação 3.30.

Resolução do problema de otimização com restrição

O problema de otimização com restrição, dado pelas Equações 3.30 e 3.33, é denominado problema primário e tem como suas principais características (a) a função objetivo (Equação 3.30) ser uma função convexa de $\mathbf{w}$ e (b) as restrições serem lineares em $\mathbf{w}$. De acordo com isso, tal problema de otimização com restrição pode ser resolvido pelo método de multiplicadores de Lagrange. Primeiramente, a função de Lagrange é descrita:

$$J(\mathbf{w}, b, \lambda) = \frac{1}{2} \mathbf{w} \cdot \mathbf{w} - \sum_{i=1}^m \lambda_i \{y_i(\mathbf{w} \cdot \mathbf{x}_i - b) - 1\}, \quad (3.34)$$

em que as variáveis auxiliares e não-negativas λ são chamadas de multiplicadores de Lagrange. A solução para tal problema otimização com restrição é determinada pelo ponto de sela da função de Lagrange $J(\mathbf{w}, b, \lambda)$, que deve ser minimizada em relação a $\mathbf{w}$ e b e maximizada em relação a λ . Desta forma, ao derivar $J(\mathbf{w}, b, \lambda)$ em relação a $\mathbf{w}$ e b e igualar seu resultado a 0, duas condições para ser ótima são obtidas. A primeira condição é dada por:

$$\frac{\delta J(\mathbf{w}, b, \lambda)}{\delta \mathbf{w}} = 0, \quad (3.35)$$

que, ao substituir seus termos, obtém-se:

$$\mathbf{w} = \sum_{i=1}^m \lambda_i y_i \mathbf{x}_i; \quad (3.36)$$

e a segunda condição é dada por:

$$\frac{\delta J(\mathbf{w}, b, \lambda)}{\delta b} = 0, \quad (3.37)$$

que, ao substituir seus termos, obtém-se:

$$\sum_{i=1}^m \lambda_i y_i = 0. \quad (3.38)$$

O vetor solução w é definido em termos de uma expansão que envolve m exemplos de treinamento.

Assim, como o *problema primal* lida com uma função de custo convexa e com restrições lineares, é possível desenvolver um outro problema: o *problema dual*. No problema dual, o valor ótimo é o mesmo do problema primal, contudo os multiplicadores de Lagrange fornecem a solução ótima. Para derivar o problema dual do problema primal anteriormente descrito, a Equação 3.34 é expandida como se segue:

$$J(\mathbf{w}, b, \lambda) = \frac{1}{2} \mathbf{w} \cdot \mathbf{w} - \sum_{i=1}^m \lambda_i y_i \mathbf{w} \cdot \mathbf{x}_i - b \sum_{i=1}^m \lambda_i y_i + \sum_{i=1}^m \lambda_i. \quad (3.39)$$

Na Equação 3.39, a primeira somatória é zero pela Equação 3.38. A partir da Equação 3.37, têm-se:

$$\mathbf{w} \cdot \mathbf{w} = \sum_{i=1}^m \lambda_i y_i \mathbf{w} \cdot \mathbf{x} = \sum_{i=1}^m \sum_{j=1}^m \lambda_i \lambda_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j. \quad (3.40)$$

Ao definir a função objetivo $J(\mathbf{w}, b, \lambda) = C(\lambda)$, a Equação 3.39 pode ser rescrita como:

$$C(\lambda) = \sum_{i=1}^m \lambda_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \lambda_i \lambda_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j, \quad (3.41)$$

em que λ_i são valores não negativos. Assim, o problema dual é definido ao maximizar a Equação 3.41, sujeito à:

$$\sum_{i=1}^m \lambda_i y_i = 0, \quad (3.42)$$

e

$$\lambda_i \geq 0. \quad (3.43)$$

O problema dual depende diretamente dos dados de treinamento. A função $C(\lambda)$ a ser otimizada depende somente dos padrões de entrada na forma de um conjunto de produtos internos $\{\mathbf{x}_i \cdot \mathbf{x}_j\}_{(i,j)=1}^m$.

A partir dos multiplicadores de Lagrange ótimos λ_i^* encontrados, o vetor de pesos w^* pode ser computado a partir da Equação 3.36 e escrito como:

$$\mathbf{w}^* = \sum_i \lambda_i^* y_i \mathbf{x}_i, \quad (3.44)$$

que deve satisfazer a condição de acordo com a propriedade Kuhn-Tucker da teoria de otimização:

$$\lambda_i * \{y_i(\mathbf{w} \cdot \mathbf{x}_i - b) - 1\} \quad \text{para } i = 1, 2, 3, \dots, m, \quad (3.45)$$

e λ_i^* e sua restrição correspondente $\{y_i(\mathbf{w} \cdot \mathbf{x}_i - b) - 1\}$ devem ser diferente de zero.

Uma vez computado o valor de λ_i^* não-negativo e seus correspondentes vetores de suporte, o bias b pode ser computado utilizando um vetor de suporte positivo x_i a partir de:

$$b^* = 1 - \mathbf{w}^* \cdot \mathbf{x}_i. \quad (3.46)$$

Por fim, a classificação da Equação 3.27 é dado, agora, por:

$$f(\mathbf{x}) = \sum_i \lambda_i y_i \mathbf{x}_i \cdot \mathbf{x} - b. \quad (3.47)$$

Truque de kernel para classificação não-linear

Se o dado de treinamento não é linearmente separável, não existe um hiperplano direto que possa separar diferentes classes. Desta forma, é necessário aprender uma função não-linear que separe as classes. O SVM linear pode ser estendido para SVM não-linear para a classificação de dados não separáveis linearmente. Assim, o SVM pode fazer o mapeamento do espaço de entrada para de um espaço de características que considere problemas de classificação não lineares. Pelo truque de *kernel*, é possível permitir a ausência de uma formulação exata para a função de mapeamento, a qual poderia introduzir ao problema do mal da dimensionalidade. Isso faz com que a classificação linear em um novo espaço seja equivalente a classificação não-linear no espaço original, pelo mapeamento dos vetores de entrada para um espaço de dimensão maior em que a separação máxima é encontrada.

O processo de encontrar funções de classificação utilizando SVM não-lineares consiste em dois passos. Primeiramente, os vetores de entrada são transformados em vetores de características de alta dimensionalidade, em que o dado de treinamento possa ser linearmente separável. Então, o SVM é utilizado para encontrar um hiperplano com margem máxima no novo espaço de características. O hiperplano é uma função linear no espaço de características transformado, mas é uma função não-linear no espaço de características original.

No treinamento do SVM não-linear, seja $\mathbf{x}$ um vetor na dimensão n do espaço de entrada e $\phi(\cdot)$ uma função não-linear de mapeamento do espaço de entrada para um espaço de características de alta dimensionalidade. O hiperplano que representa a fronteira de decisão no espaço de características é dado por:

$$\mathbf{w} \cdot \phi(\mathbf{x}) - b = 0, \quad (3.48)$$

em que w é o vetor de pesos capaz de mapear o dado de treinamento para um espaço de características de maior dimensão para o espaço de saída, e b , o bias. Utilizando a

função $\phi(\cdot)$, os pesos são dados por:

$$\mathbf{w} = \sum \lambda_i y_i \phi(\mathbf{x}_i). \quad (3.49)$$

A partir disso, a função de decisão da Equação 3.47 pode ser reescrita como:

$$f(\mathbf{x}) = \sum_i^m \lambda_i y_i \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}) - b. \quad (3.50)$$

As funções de mapeamento de características no problema de otimização e também na função de classificação sempre aparecem como produtos escalar, por exemplo, $\phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j)$, que refere-se ao produto interno entre pares de vetores no espaço de características transformado. O cômputo do produto interno é espaço de característica transformado é complexo e sofre do problema do mal da dimensionalidade. Para evitar esse problema, o truque de *kernel* é utilizado. O truque de *kernel* substitui o produto interno no espaço de características pela função de *kernel* K no espaço original, como a seguir:

$$K(\mathbf{u}, \mathbf{v}) = \phi(\mathbf{u}) \cdot \phi(\mathbf{v}). \quad (3.51)$$

O teorema de Mercer's prova que uma função de *kernel* K é válida se, e somente se, algumas condições sejam satisfeitas [85]. Tal teorema garante que uma função de *kernel* possa ser sempre expressada como o produto interno entre pares de vetores de entrada em algum espaço de alta dimensionalidade. Assim, o produto interno pode ser calculado utilizando uma função de *kernel* somente com os vetores de entrada no espaço original, sem transformar os vetores de entrada em vetores de características de alta dimensão. Desta forma, o problema dual com restrições é agora definido utilizando a função de *kernel*, maximizando:

$$C_2(\lambda) = \sum_i \lambda_i - \sum_i \sum_j \lambda_i \lambda_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j), \quad (3.52)$$

desde que seja sujeito a:

$$\sum_i \lambda_i y_i = 0, \quad (3.53)$$

e :

$$C \geq \lambda \geq 0. \quad (3.54)$$

Por fim, a função de classificação é dada por:

$$f(\mathbf{x}) = \sum_i \lambda_i y_i K(\mathbf{x}_i, \mathbf{x}) - b. \quad (3.55)$$

Uma vez que $K(\cdot)$ é computada no espaço de entrada, nenhuma transformação de características será feita ou nenhum $\phi(\cdot)$ será computado. Assim, o vetor de peso $\mathbf{w} = \sum \lambda_i y_i \phi(\mathbf{x})$ também não será computado para SVMs não-lineares. Abaixo, algumas funções de *kernel* populares são descritas.

- Polinomial: $K(a, b) = (a \cdot b + 1)^d$;
- Função de base radial, do inglês *radial basis function* (RBF): $K(a, b) = \exp(-\epsilon \|a - b\|^2)$; e
- Sigmóide: $K(a, b) = \tanh(\kappa a \cdot b + c)$.

A função de *kernel* é um tipo de função de similaridade entre dois vetores, em que a função de saída é maximizada quando dois vetores se tornam equivalentes. Por causa disso, o SVM pode aprender uma função de qualquer formato de dados além de vetores (como árvores e grafos), a medida que a função de similaridade é computada entre qualquer par de objetos de dados.

Floresta de caminhos ótimos

O algoritmo OPF é um classificador de padrões que foi desenvolvido como uma alternativa a classificadores que demandam elevado custo computacional, quando utilizado para grandes quantidades de dados. Tal classificador é baseado em um grafo completo derivado das amostras de treinamento. Cada amostra de treinamento é interpretada como nó de um grafo, em que seus pesos são as distâncias entre nós são dadas por um dado espaço de características. As relações de adjacência entre essas amostras são arestas. Cada caminho em um grafo é atribuído de acordo com a sua função de conectividade e sua minimização resulta em uma floresta de caminhos ótimos enraizada nas amostras representativas de cada classe. A formulação matemática e os algoritmos de treinamento e classificação são descritos abaixo.

Considerando que (Z_1, A) é um grafo completo, seus nós são amostras em Z_1 e qualquer par de amostras definem um arco em $A = Z_1 \times Z_1$, cujo A está implícito. Um caminho é uma sequência de amostras distintas $\pi_t = \langle s_1, s_2, \dots, s_{k-1}, t \rangle$ terminando em t , em que $(s_i, s_{i+1}) \in A$ para $1 \leq i \leq k-1$. Para cada caminho π_t um custo $f(\pi_t)$ é calculado por uma função de custo de caminho. Um caminho π_t é considerado ótimo se $f(\pi_t) \leq f(\Pi_t)$ para qualquer outro caminho Π_t terminando em mesmo t . Também é considerado $\pi_s \cdot \langle s, t \rangle$ como a concatenação de um caminho π_s e seu arco (s, t) .

O treinamento do classificador consiste em encontrar uma floresta de caminho ótimo em (Z_1, A) enraizada em um conjunto $S \subset Z_1$ de protótipos. O conjunto S é representado pelas amostras que compartilham arcos entre classes distintas em uma árvore de extensão mínima, do inglês *minimum-spanning tree* (MST), de (Z_1, A) . Para a função de custo de caminho f_{max} esses protótipos tendem a minimizar os erros de classificação em Z_1 quando seus rótulos são propagados para os nós de suas árvores. A função de custo de caminho f_{max} é dada por:

$$\begin{aligned} f_{max}(\langle s \rangle) &= \begin{cases} 0, & \text{se } s \in S, \\ +\infty, & \text{caso contrário.} \end{cases} \\ f_{max}(\pi_s \cdot \langle s, t \rangle) &= \max\{f_{max}(\pi_s), d(s, t)\}, \end{aligned} \quad (3.56)$$

em que $d(s, t)$ refere-se à distância entre as amostras s e t . Assim, $f_{max}(\pi_s)$ calcula a distância máxima entre as amostras adjacentes em um caminho π_s . O Algoritmo 2 apresenta a implementação para o treinamento do classificador OPF.

Algoritmo 2: Algoritmo de treinamento para o classificador OPF.

Entrada: Um conjunto λ – *labeled* de treinamento rotulado de Z_1 e o par (v, d) para o compute do vetor de características e distâncias.

Saída: Floresta de caminhos ótimos P_1 , mapa de custos C_1 , mapa de rótulos L_1 e conjunto ordenado Z'_1 .

Auxiliary: Fila de prioridade Q , conjunto S de protótipos e variável de custo cst .

```

1 Definir  $Z'_1 \leftarrow \emptyset$  e computar MST pelo conjunto de protótipos  $S \subset Z_1$ ;
2 Para cada  $s \in Z'_1 \setminus S$ , definir  $C_1(s) \leftarrow +\infty$ ;
3 para cada  $s \in S$  faça
4   |  $C_1(s) \leftarrow 0, P_1(s) \leftarrow nil, L_1(s) \leftarrow \lambda(s)$ , e insira  $s$  in  $Q$ ;
5 fim
6 enquanto  $Q$  não está vazia faça
7   | Remover de  $Q$  uma amostra  $s$  tal que  $C_1(s)$  é mínimo;
8   | Inserir  $s$  em  $Z'_1$ ;
9   | para Para cada  $t \in Z_1$  tal que  $C_1(t) > C_1(s)$ , faça
10    | Computar  $cst \leftarrow \max\{C_1(s), d(s, t)\}$ ;
11    | se  $cst < C_1(t)$  então
12      | se  $C_1(t) \neq +\infty$  então
13        | Remover  $t$  de  $Q$ .;
14      | fim
15      |  $P_1(t) \leftarrow s, L_1(t) \leftarrow L_1(s), C_1(t) \leftarrow cst$ .;
16      | Inserir  $t$  em  $Q$ .;
17    | fim
18  | fim
19 fim
20 retorna um classificador  $[P_1, C_1, L_1, Z'_1]$ .
```

A classificação de cada nova amostra $t \in Z_2$ é dada pela distância $d(s, t)$ entre t e cada nó de treinamento $s \in Z_1$ seguindo a avaliação dada pela seguinte formulação:

$$C_2(t) = \min\{\max\{C_1(s), d(s, t)\}\}, \forall s' \in Z'_1, \quad (3.57)$$

em que $s^* \in Z'_1$ é o nó s' que satisfaz essa equação e sua avaliação pode ser suspensa quando $\max\{C_1(s), d(s, t)\} < C_1(s')$ para um nó s' , cuja posição em Z'_1 sucede a posição de s . O Algoritmo 3 apresenta a implementação para a classificação do classificador OPF.

Algoritmo 3: Algoritmo de classificação para o classificador OPF.

Entrada: Classificador $[P_1, C_1, L_1, Z'_1]$, conjunto para avaliação Z_2 e o par (v, d) para o computo do vetor de características e distâncias.

Saída: Rótulos L_2 e mapa de predecessores P_2 definidos por Z_2 .

Auxiliary: Variáveis de custo tmp e $mincost$.

```

1 Definir  $Z'_1 \leftarrow 0$  e computar MST pelo conjunto de protótipos  $S \subset Z_1$ ;
2 Para cada  $s \in Z'_1 \setminus S$ , definir  $C_1(s) \leftarrow +\infty$ ;
3 para cada  $t \in Z_2$  faça
4    $i \leftarrow 1, mincost \leftarrow \max\{C_1(k_i, dist(k_i, t))\}$ ;
5    $L_2(t) \leftarrow L_1(k_i)$  and  $P_2(t) \leftarrow k_i$ ;
6   enquanto  $i < |Z'_1|$  and  $mincost > C_1(k_i + 1)$  faça
7     Computar  $tmp \leftarrow \max\{C_1(k_i + 1), d(k_i + 1, t)\}$ ;
8     se  $tmp < mincost$  então
9        $L_2(t) \leftarrow L(k_i + 1)$  and  $P_2(t) \leftarrow k_i + 1$ ;
10    fim
11     $i \leftarrow i + 1$ ;
12  fim
13 fim
14 retorna  $[L_2, P_2]$ .
```

3.4.2 Aprendizado semi-supervisionado

No aprendizado semi-supervisionado, é fornecido um rótulo para determinada categoria de determinado padrão para algumas amostras do conjunto de treinamento, ou seja, existem amostras supervisionadas e não-supervisionada no conjunto de treinamento. No aprendizado, são exploradas outras informações a partir das amostras não-supervisionadas que não, seus rótulos, como distâncias ou agrupamento nos dados. Desta forma, os classificadores semi-supervisionados podem ser utilizados como propagadores de rótulos, uma vez que um rótulo é atribuído às amostras não-supervisionadas. Assim, dois classificadores foram escolhidos dentre aqueles disponíveis na literatura, seguindo os classificadores supervisionados definidos anteriormente.

Nesta seção, os algoritmos Máquina de Vetores de Suporte Laplaciano, do inglês *Laplacian Support Vector Machine* (LapSVM), e Floresta de Caminhos Ótimos Semi-Supervisionada, do inglês *Semi-Supervised Optimum Path Forest* (OPF-Semi).

Máquina de vetores de suporte laplaciano

Abordagens relacionadas ao aprendizado semi-supervisionado buscam explorar, não diretamente a sua classificação, mas o próprio dado e informações a partir de sua geometria. Técnicas de regularização de *manifold*, por exemplo, baseiam-se na geometria da distribuição dos dados, e têm como objetivo obter uma função de decisão “suave” e assim evitar o super treinamento no conjunto de dados. *Manifolds* são regiões conectadas e, matematicamente, são um conjunto de pontos associados a uma vizinhança ao redor de cada ponto.

A partir de qualquer ponto dado, o *manifold* parece ser um espaço Euclidiano localmente. O conceito da vizinhança em volta de cada ponto implica na existência de transformações que podem ser aplicada para mover um *manifold* de uma posição para uma vizinhança.

Embora exista uma formulação matemática formal para *manifold*, em aprendizado de máquina o termo costuma ser utilizado para referenciar um conjunto conexo de pontos que podem ser aproximados de maneira satisfatória considerando somente um número pequeno de grau de liberdade ou dimensões, embutido no espaço de alta dimensionalidade. Técnicas de regularização têm sido estendidas para o contexto semi-supervisionado. O LapSVM é uma extensão do conhecido SVM que segue os princípios da regularização de *manifold*. A seguir, a formulação matemática é apresentada, sendo que em sua maior parte foi baseada nos artigos [64, 5].

Assume-se que a classe de dois pontos próximos na geometria intrínseca da distribuição dos dados é a mesma ou similar, de forma que a probabilidade da distribuição mude pouco entre esses dois pontos. A partir dessa ideia geométrica, pôde-se estender convenções utilizadas para aprendizado de funções. Algoritmos como SVM e funções de base radial podem ser interpretados como algoritmos de regularização com diferentes funções de custo empíricas e medidas de complexidade em um Espaço de Hilbert de Reprodução de *Kernel*, do inglês *Reproducing Kernel Hilbert Space* (RKHS). A partir dos princípios de regularização de *manifold*, chega-se ao problema de otimização dado por:

$$\mathbf{f}^* = \underset{\mathbf{f} \in H_K}{\operatorname{argmin}} \frac{1}{l} \sum_{i=1}^l V(x_i, y_i, \mathbf{f}) + \gamma_A \|\mathbf{f}\|_K^2 + \frac{\gamma_I}{(u+l)^2} \hat{\mathbf{f}}^T L \hat{\mathbf{f}} \quad (3.58)$$

em que l é um conjunto de exemplos supervisionados $\{x_i, y_i\}_{i=1}^l$; u é um conjunto de exemplos não-supervisionados $\{x_j\}_{j=l+1}^{l+u}$; a norma $\|\cdot\|_K$ é um RKHS H_K associado correspondente, em que a penalidade adiciona uma suavidade para a solução; $(x_i, y_i), i = 1, 2, \dots, l$ é um conjunto de exemplos rotulados; V é uma função de perda (como uma função de perda de margem suave para o SVM); γ_A e γ_I são regularizadores, em que γ_A controla a complexidade da função no espaço ambiente, enquanto γ_I controla a complexidade da função na geometria intrínseca da distribuição; e o termo $\frac{\gamma_I}{(u+l)^2} \hat{\mathbf{f}}^T L \hat{\mathbf{f}}$ é uma aproximação baseada em dados supervisionados e não-supervisionados de um caso particular em que a distribuição é um sub-*manifold* compacto de (x_i) . Deste termo, $\hat{\mathbf{f}} = [\mathbf{f}(x_1), \dots, \mathbf{f}(x_{l+u})]^T$; $\frac{1}{(u+l)^2}$ é o coeficiente de normalização para a estimação do operador de Laplace; e L é o grafo Laplaciano. O grafo Laplaciano dado por $L = D - W$, sendo W_{ij} os pesos de arestas no grafo de adjacências do dado, D a matriz diagonal dada por $D_{ii} = \sum_{j=1}^{l+u} W_{ij}$.

Depois de apresentada a Equação 3.58, o LapSVM resolve o problema de otimização da dada equação explorando a função de perda de margem suave definida como $V(x_i, y_i, \mathbf{f}) = \max(0, 1 - y_i \mathbf{f}(x_i))$, $y_i \in \{-1, +1\}$. Ao introduzir as variáveis de folga da restrição utilizando o multiplicador de Lagrange para derivar SVMs, a seguinte formulação é encontrada:

$$\beta^* = \max_{\beta \in \mathbb{R}^l} \sum_{i=1}^l \beta_i - \frac{1}{2} \beta^T Q \beta, \quad (3.59)$$

sujeito as restrições:

$$\sum_{i=1}^l y_i \beta_i = 0, \quad (3.60)$$

e

$$0 \leq \beta_i \leq \frac{1}{l}, \quad (3.61)$$

em que $i = 1, 2, \dots, l$; e Q equivale a:

$$Q = YJK(2\gamma_A I + 2\frac{\gamma_I}{(u+l)^2}LK)^{-1}J^T Y. \quad (3.62)$$

Na Equação 3.62, Y é a matriz diagonal $Y_{ii} = y_i$, K é a matriz Gram sobre os dados supervisionados e não-supervisionados; L é o grafo Laplaciano de adjacências do dado; J é uma matriz $l \times (l+u)$ dado por $-J_{ij} = 1$ se $i = j$ e x_i é um exemplo rotulado, e $J_{ij} = 0$ caso contrário. Para obter a vector coeficiente de expansão ótima $\alpha^* \in \mathbb{R}^{l+u}$, é necessário resolver o sistema linear a seguir:

$$\alpha^* = (2\gamma_A I + 2\frac{\gamma_I}{(u+l)^2}LK)^{-1}J^T Y \beta^*. \quad (3.63)$$

Pode-se notar que quando $\gamma_I = 0$ nas Equações 3.62 e 3.63, os valores de coeficientes de expansão é zero para o dado não rotulado. Os coeficientes de expansão para os dados rotulados e para a matriz Q são com os padrões do SVM neste caso.

Floresta de caminhos ótimos semi-supervisionada

O OPF-Semi, assim como a forma supervisionada do OPF, é um algoritmo baseado em grafo, que interpreta amostras supervisionadas e não-supervisionadas como nós de um grafo e computa uma floresta de caminhos ótimos enraizada apenas nas amostras supervisionadas. Nesta floresta, cada amostra é conquistada (rotulada) por uma raiz que oferece um caminho com custo mínimo a ela.

Seja Z o conjunto de dados em que as amostras s são representadas por um vetor de características $\mathbf{v}(s) \in \mathbb{R}^N$. O conjunto $Z = Z^l \cup Z^u$ consiste dos subconjuntos de amostras supervisionadas Z^l e não-supervisionadas Z^u . Seja, também, $\lambda(s) \in \{1, 2, 3, \dots, c\}$ para c categorias o verdadeiro rótulo de cada amostra s e $d(s, t) \geq 0$ a função de distância simétrica entre amostras. Para o treinamento, o método deve conectar amostras de Z em um grafo e propagar rótulos para Z^u de forma que o classificador seja uma floresta de caminhos ótimos enraizada em Z^l .

Considera-se arcos como uma relação de adjacência $A = Z \times Z$, que define um grafo completo e ponderado (Z, A) . Um caminho simples é uma sequência de amostras distintas $\pi_t = \langle s_1, s_2, s_3, \dots, s_{k-1}, s_t \rangle$ com término $t = s_k$, em que $(s_i, s_{i+1}) \in A$ para $1 \leq i \leq k-1$. O custo $F(\pi_t)$ de π_t é dado por uma função de valor de caminho f . Um caminho π_t é considerado mínimo se $f(\pi_t) \leq f(\tau_t)$ para qualquer outro caminho τ_t com o mesmo término t .

O treinamento consiste em encontrar um conjunto S^* de amostras chaves (os melhores protótipos) para proteger suas categorias. A abordagem semi-supervisionada sugere computar uma árvore geradora de custo mínimo nas amostras supervisionadas (Z_l, A) e selecionar amostras que compartilham um arco na árvore geradora mínima entre classes distintas. Isso garante que todas as categorias tenham protótipos dentro de uma fronteira entre categorias com amostras supervisionadas, onde estão as amostras mais difíceis para classificação. Assumindo que as amostras de mesma classe estão conectadas aos protótipos por um caminho de pares de amostras próximas e forçando as raízes em S^* , a função de custo de caminho é dada por:

$$\begin{aligned} f_{max}(\langle s \rangle) &= \begin{cases} 0, & \text{se } s \in S^*, \\ +\infty, & \text{caso contrário,} \end{cases} \\ f_{max}(\pi_s \cdot \langle s, t \rangle) &= \max\{f_{max}(\pi_t), d(s, t)\}. \end{aligned} \quad (3.64)$$

Assim, $f_{max}(\pi_t)$ computa o máximo peso de arco ao longo de caminhos simples. A floresta de caminho ótimo é obtida pela minimização de um mapa de custo de caminhos $C_1(t)$ para todo $t \in Z$.

$$C(t) = \min_{\forall \pi_t \in \Pi(Z, A, t)} \{f_{max}(\pi_t)\}, \quad (3.65)$$

em que $\Pi(Z, A, t)$ é o conjunto de todos os caminhos no grafo (Z, A) com término t . O Algoritmo 4 descreve o processo de treinamento semi-supervisionado.

Os valores de certeza segundo o OPF-Semi podem ser computados como descrito em [45, 67, 11]. Sejam A e B duas raízes tal que A conquista a amostra s e B é a raiz de classe diferente de A , que oferece um segundo melhor custo a s . O valor de certeza atribuído a s é o quociente entre o custo oferecido por A e a soma dos custos oferecidos por A e B . É esperado que o custo oferecido por B seja muito maior do que o custo oferecido para A , de forma que o rótulo atribuído a s esteja correto. Caso contrário, o valor de certeza de s pode ser maior considerando o intervalo $[0, 1]$.

Algoritmo 4: Algoritmo de treinamento para o classificador OPF semi-supervisionado.

Entrada: Um conjunto λ – *labeled* de treinamento supervisionado de Z_l , conjunto não supervisionado Z_u e o par (v, d) para o computo do vetor de características e distâncias.

Saída: Floresta de caminhos ótimos P , mapa de custos C , mapa de rótulos L e conjunto ordenado Z' .

Auxiliary: Fila de prioridade Q , conjunto S^* de protótipos e variável de custo cst .

```

1 Definir  $Z' \leftarrow \emptyset$  e computar MST pelo conjunto de protótipos  $S^* \subset Z$ ;
2  $Z = (Z_l) \cup (Z_u)$ ;
3 Para cada  $s \in Z' \setminus S^*$ , definir  $C(s) \leftarrow +\infty$ ;
4 para cada  $s \in S^*$  faça
5   |  $C(s) \leftarrow 0, P(s) \leftarrow nil, L(s) \leftarrow \lambda(s)$ , e insira  $s$  in  $Q$ ;
6 fim
7 enquanto  $Q$  não está vazia faça
8   | Remover de  $Q$  uma amostra  $s$  tal que  $C(s)$  é mínimo;
9   | Inserir  $s$  em  $Z'$ ;
10  para Para cada  $t \in Z$  tal que  $C(t) > C(s)$ , faça
11    | Computar  $cst \leftarrow \max\{C(s), d(s, t)\}$ ;
12    | se  $cst < C(t)$  então
13      | se  $C(t) \neq +\infty$  então
14        | Remover  $t$  de  $Q$ .;
15      | fim
16      |  $P(t) \leftarrow s, L(t) \leftarrow L(s), C(t) \leftarrow cst$ .;
17      | Inserir  $t$  em  $Q$ .;
18    | fim
19  fim
20 fim
21 retorna um classificador  $[P, C, L, Z']$ .
```

Capítulo 4

Abordagem Semi-automática para Anotação de Dados

Considerando o referencial teórico, foi proposta uma abordagem semi-automática que possa reduzir o esforço do usuário com melhores acurácias de classificação nos conjuntos de testes não conhecidos pelo classificador. Dado um conjunto de treinamento com quantidade reduzida de amostras supervisionadas e quantidade considerável de amostras não-supervisionadas o método semi-automático de anotação de dados proposto tem 4 etapas:

- aprendizado não-supervisionado: primeiramente, características são extraídas do dado de entrada. Com o objetivo de melhorar o espaço de características original da imagem, um procedimento não-supervisionado de aprendizado de características foi adotado;
- projeção do espaço de características: uma projeção 2D do espaço de características foi gerada, de forma a capturar bem a distribuição das amostras no espaço de características latente para análise visual posterior;
- estimação de rótulos semi-supervisionada: os rótulos são propagados automaticamente para as amostras não-supervisionadas com maior confiança, de forma a aumentar a quantidade de rótulos do conjunto de treinamento com mínimo esforço e alta eficiência;
- análise visual: o usuário fornece rótulos adicionais às amostras acima, propagando rótulos interativamente para as amostras com menos confiança utilizando projeção 2D de características.

A Figura 4.1 ilustra o método proposto. Nele, as características são extraídas do conjunto de treinamento por um algoritmo de aprendizado não-supervisionado. Então, essas características são utilizadas para projetar em um gráfico de dispersão (do inglês, *scatterplot*) no 2D. O conjunto de treinamento é enriquecido pelos rótulos propagados a partir das amostras supervisionadas para aquelas não-supervisionadas pelo método automático (nos espaços latente e projetado) e pelos métodos manuais controlados pelo

usuário. Finalmente, compara-se a eficiência dos classificadores treinados em tais conjuntos de treinamento para decidir qual o melhor método de propagação. Na Figura, a cor vermelha indica adições realizadas em comparação com estudos anteriores [7]. Abaixo, as principais etapas são descritas em detalhes.

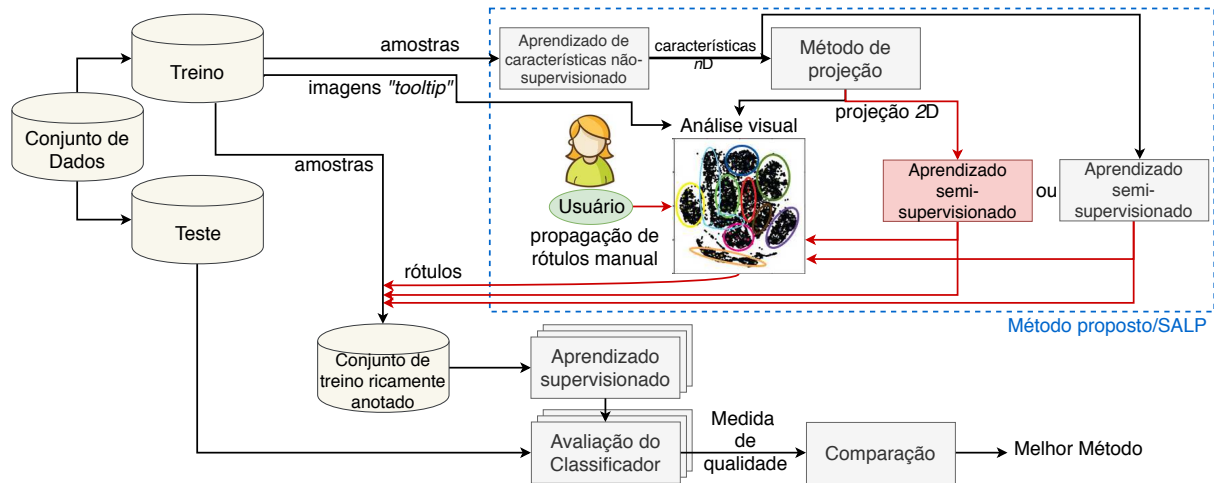


Figura 4.1: Método semi-automático proposto para anotação de dados.

4.1 Aprendizado não-supervisionado de características

Para o aprendizado não-supervisionado de características, testaram-se diferentes tipos de AuNNs: arquiteturas totalmente conectadas com uma única camada [4, 32, 9], redes esparsas com constante de regularização adicionada ao erro de reconstrução [52], além de arquiteturas com poucas camadas escondidas [78, 21]. No entanto, AuNN com camadas de convolução [42] mostrou-se mais interessante devido a sua capacidade em extrair características de forma e cor. O *encoder* mapeia as amostras de entrada para pontos em um espaço de características reduzido (latente); enquanto o *decoder* reconstrói tais amostras. Essas duas partes são treinadas juntas por retropropagação, a fim de minimizar o erro médio quadrático entre a amostra original e a reconstruída. Para os experimentos, optou-se por utilizar base de dados de imagens. Contudo, esse aprendizado de características latente pode ser utilizado para qualquer outro tipo de dado que possa ser mapeado de forma adequada para a camada de entrada do *encoder*.

4.2 Projeção do espaço de características

Estudos anteriores indicam que projeções 2D, criadas pelo algoritmo t-SNE [21, 76], são capazes de atingir esse objetivo de forma adequada [58, 8, 7]. Assim, optou-se em utilizar o algoritmo t-SNE como utilizado em tais estudos. A dimensão do espaço de características fornecido pela AuNN pode ser ainda considerada muito alta (com, normalmente, centenas ou milhares de características) e impossível para a inspeção visual da distribuição das amostras. Como mencionado anteriormente, é necessário reduzir o espaço latente para

um espaço de duas dimensões de forma que facilite a inspeção visual pelo usuário, além de preservar as estruturas relevantes dos dados. Dentre as técnicas mais adequadas para tal tarefa, o algoritmo t-SNE mostra-se apropriado, uma vez que preserva as distâncias locais entre as amostras [75] ao computar a similaridade entre os pontos mais próximos. Desta forma, o t-SNE é calculado a partir das características fornecidas pela AuNN com o objetivo de obter um espaço de características, agora, com duas dimensões e facilitando as ações do usuário para a propagação de rótulos.

4.3 Estimação de rótulos semi-supervisionada

Para a estimação de rótulos semi-supervisionada, foram consideradas duas técnicas que exploram a distribuição de amostras em um dado espaço de características para a propagação de rótulos das amostras supervisionadas para as não-supervisionadas: Máquinas de Vetores de Suporte Laplacianas [64, 5] (LapSVM), e classificação semi-supervisionada pela Floresta de Caminhos Ótimos [2] (OPF-Semi). Ambos os métodos são avaliados nos espaços latente e projetado. Dada a performance do OPF-Semi na propagação de rótulos ser muito superior do que do LapSVM (Capítulo 5), escolheu-se o OPF-Semi para fornecer os valores de confiança, utilizados mais tarde para propagação de rótulos manual. Adicionalmente, o OPF-Semi no espaço projetado superou seus resultados no espaço latente (Capítulo 5). Por este motivo, utilizou-se a versão em 2D do OPF-Semi para a anotação de dados semi-automática.

O OPF-Semi mapeia amostras supervisionadas e não-supervisionadas como nós de um grafo e computa uma floresta de caminhos ótimos enraizada nas amostras supervisionadas. Nesta floresta, cada nó s é conquistado (rotulado) pela raiz R que oferece um caminho como custo mínimo $k(A, s)$ para s . O custo é utilizado para computar os valores de confiança $c(s)$ do rótulo como em [45, 67, 11]. Em suma, sejam A e B duas raízes para a amostras s de forma que A é aquele que conquistou s , isto é, $k(R, s)$ é mínimo, e B , tendo rótulo diferente de A , oferece o segundo melhor custo $k(B, s)$ para s . O valor de confiança é dado por:

$$c(s) = \frac{1 - k(A, s)}{(k(A, s) + k(B, s))}, \quad c(s) \in [0, 1], \quad (4.1)$$

para o rótulo de s dado por A . Isto é, se o segundo melhor custo $k(B, s)$ é muito maior do que o custo mínimo $k(A, s)$, o rótulo A tem maior confiança. A confiança é utilizada como se segue: todos os rótulos atribuídos pelo OPF-Semi com confiança acima de um limiar τ são utilizados tal como no processo de treinamento. O limiar τ é escolhido pelo usuário baseado na análise visual da projeção de características com as amostras não-supervisionadas coloridas pelos seus valores de confiança: que vai de vermelho (baixo c) a verde (alto c), como demonstrado na Figura 4.3. As amostras são coloridas de acordo com o modelo de cor HSV, do inglês *Hue, Saturation, and Value*, que representam *Matiz* (H), *Saturação* (S) e *Valor* (V) respectivamente (Figura 4.2).

Além de ter sido utilizado para a elaboração da interface, o *framework* para desenvolvi-

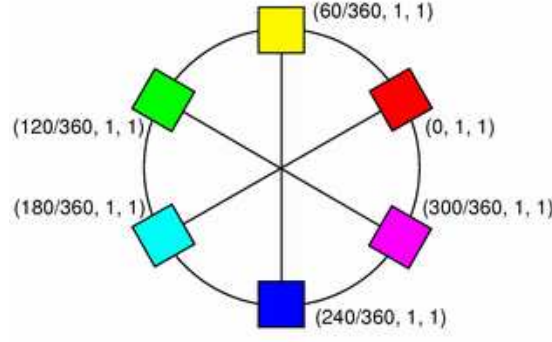


Figura 4.2: Representação das cores segundo o esquema de cor HSV. (Figura reproduzida da documentação do Qt .)

mento de interfaces gráficas Qt ¹ foi também empregado para o mapeamento da confiança dada pelo classificador. Os valores de S , V e H podem variar no intervalo $[0, 1]$, segundo as funções de cor do Qt . Para a implementação do mapeamento de cor da confiança em escala HSV, definiu-se $S = 1$, $V = 1$ e H é calculado da seguinte forma:

$$H = c(s) \times \left(\frac{b - e}{360^\circ} \right), \quad (4.2)$$

em que $c(s)$ é o valor de confiança no intervalo $[0, 1]$ dado pelo OPF-Semi para cada amostra (nó) s ; e b e e são as angulações (cores) que se quer mapear na escala de cor, neste caso, $b = 120^\circ$, que é a angulação referente a cor verde e $e = 0^\circ$, que é a angulação referente a cor vermelha. As cores de H podem variar em um intervalo de $(0^\circ/360^\circ)$ a $(120^\circ/360^\circ)$, ou seja, de vermelho (baixa confiança) a verde (alta confiança), e sendo $(60^\circ/360^\circ)$, amarelo.

Mudando o valor de τ iterativamente com uma ferramenta de deslizar, o usuário pode (a) escolher as amostras com maior confiança que terão seus rótulos atribuídos pelo OPF-Semi e (b) focar nas amostras restantes com menor confiança que terão seus rótulos atribuídos pela propagação de rótulos manual. Assim, o usuário pode escolher o limiar τ exato que equilibra o quanto o usuário deseja confiar no OPF-Semi e quantas amostras ele está disposto a lidar manualmente.

4.4 Propagação de rótulos manual

Na primeira etapa deste estudo, o objetivo foi entender se a propagação de rótulos manual utilizando projeções t-SNE do espaço de características era viável para a anotação de dados. Assim, a importância do usuário na propagação de rótulos manual foi demonstrada pela técnica de propagação de rótulos interativa, em inglês *interactive label propagation* (ILP), publicada em [7]. A denominação foi mantida, a fim de facilitar a comparação. Para ILP, foi desenvolvida uma ferramenta que mostra uma projeção t-SNE de pontos supervisionados e não-supervisionados, em que a classe das amostras supervisionadas são codificadas por um mapeamento de cores categórico e as amostras não-supervisionadas

¹<https://www.qt.io/>

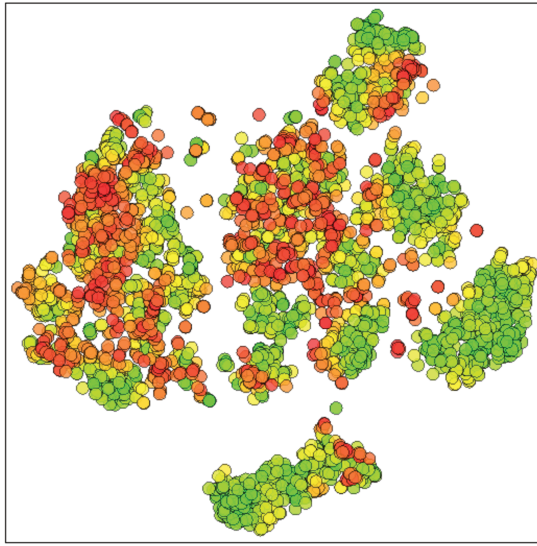


Figura 4.3: Projeção de características com amostras não-supervisionadas de vermelho (baixa confiança) até o verde (alta confiança).

são apresentadas em preto. A identificação visual de grupos que contenham poucas amostras de mesma classe e o restante em preto guia o usuário a propagar o rótulo de dada classe para as amostras não-supervisionadas daquele grupo. A extensão dessa propagação de rótulos é decidida manualmente pelo delineamento baseado num desenho livre utilizando o *mouse* (Figura 4.4) e então é atribuída da cor do ponto supervisionado para as amostras não-supervisionadas selecionadas. Tal ação pode ser repetida múltiplas vezes para qualquer região na projeção. Uma pequena imagem (*tooltip*) é também mostrada a medida que o usuário move o *mouse* na projeção sobre os pontos correspondentes em 2D (Figura 4.5). O *tooltip* não é considerado uma supervisão, uma vez que indica rapidamente se a maior parte das imagens daquele grupo são da mesma classe e auxiliam o usuário a limitar a extensão da propagação de rótulos.

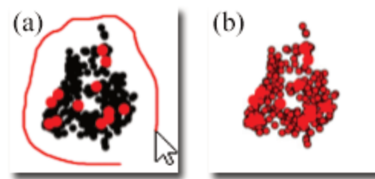


Figura 4.4: O usuário (a) manualmente seleciona uma amostra de um grupo para (b) propagar os rótulos para amostras não-supervisionadas. (Figura reproduzida de [7]).

Contudo, a propagação ILP é fundamentalmente afetada pela qualidade das características latentes extraídas pela AuNN, bem como pela qualidade da projeção t-SNE. Se ambas as operações preservarem rigorosamente a similaridade das amostras originais, então o usuário pode ser capaz de propagar rótulos bem, selecionando pontos próximos às amostras supervisionadas na projeção. Se o espaço latente ou a projeção criarem erros, o que de fato acontece [47], a propagação ILP possivelmente criará rótulos errados. Dando continuidade aos estudos realizados, o objetivo é auxiliar o usuário neste processo como segue.

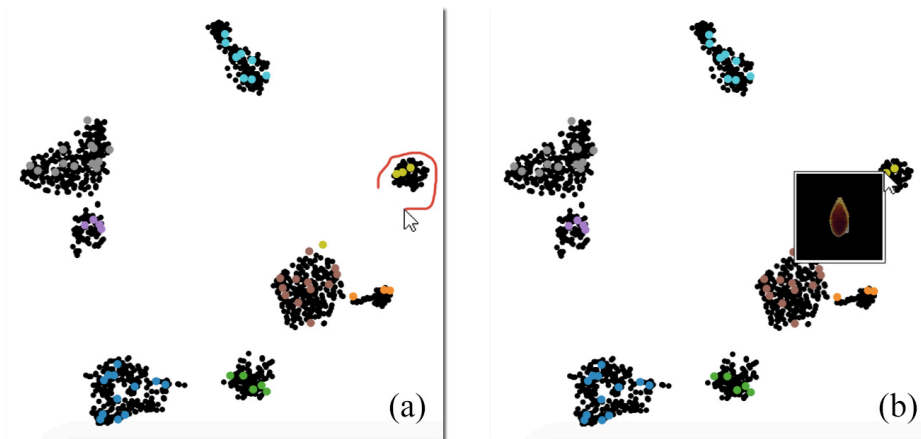


Figura 4.5: Projeção com grupos bem separados em (a) e *tooltip* mostrado para cada ponto no 2D em (b). (Figura reproduzida de [7].)

As amostras supervisionadas são coloridas na projeção pelos seus rótulos e todas as amostras não-supervisionadas com baixa confiança s com $c(s) < \tau$ são apresentadas em preto, como mostra a Figura 4.6. Os pontos em preto são projetados antes dos pontos coloridos, a fim de minimizar as oclusões indesejadas, isto é, um ponto preto sobre um ponto colorido. Ao mover o *mouse* sobre um ponto projetado, uma pequena imagem como um *tooltip* é mostrada. O usuário utiliza três diferentes fontes de informação, a fim de decidir quais amostras não-supervisionadas obterão determinado rótulo supervisionado: a proximidade de pontos não-supervisionados (pretos) na projeção 2D em relação aos supervisionados (coloridos), o valor de baixa confiança dos pontos não-supervisionados e a similaridade das imagens do *tooltip* das amostras não-supervisionadas para as supervisionadas. A propagação de rótulos é então realizada facilmente pela seleção dos pontos desejados na projeção e, depois, pelo clique para atribuir a eles um rótulo de um ponto supervisionado.

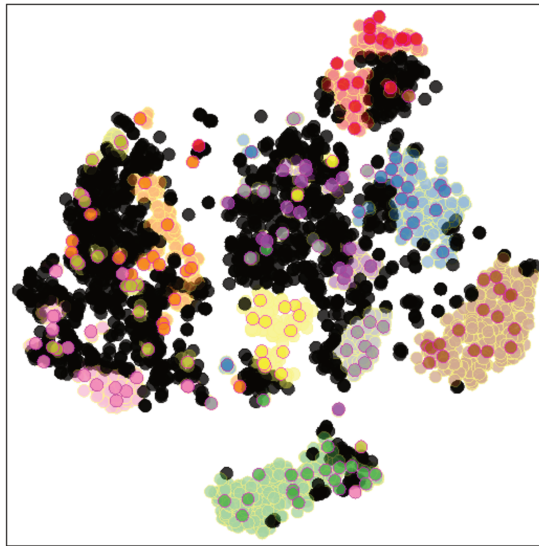


Figura 4.6: Propagação semi-automática de rótulos é realizada, primeiramente, a partir das amostras supervisionadas (pontos coloridos pela classe, borda vermelha) automaticamente para as amostras não-supervisionadas e com maior confiança (cores mais claras, sem borda). As amostras restantes de menor confiança (preto) são candidatas para propagação manual.

Capítulo 5

Experimentos e Resultados

Neste capítulo, a configuração experimental, os experimentos utilizados como comparação, as bases de dados para validação, os detalhes de implementação e os resultados experimentais são apresentados.

5.1 Configuração experimental

Cada base de dados D disponível foi dividida em subconjuntos para validação: um conjunto S muito pequeno de treinamento de poucas amostras supervisionadas por classe ($3\%|D|$); um conjunto U consideravelmente maior de treinamento de amostras não-supervisionadas para propagação de rótulos ($67\%|D|$); e um conjunto T com amostras de teste não conhecidas. A fim de se aprender as características em um espaço latente, os conjuntos S e U foram utilizados para treinar a AuNN e, em seguida, o conjunto T foi utilizado apenas para fins de extração de características a partir da rede pré-treinada. Por tanto, os conjuntos S , U e T mencionados adiante neste trabalho, bem como aqueles utilizados para classificação, referem-se às características aprendidas pela AuNN.

Baseado na escolha do usuário do limiar de confiança τ , o conjunto U foi dividido em amostras L_c com alta confiança, que terão seus rótulos obtidos pelo OPF-Semi, e amostras L_i com baixa confiança, que serão rotuladas interativamente pelo usuário. Note que $L_c \cap L_i = \emptyset$ e $L_c \cup L_i \neq U$, uma vez que o usuário pode escolher não rotular todo o conjunto L_i para minimizar o esforço da propagação manual. Cada D foi dividido três vezes de forma aleatória nos conjuntos S , U e T da forma descrita e a avaliação foi repetida, isto é, a propagação de rótulos de S para U seguida de treinamento supervisionado em $S \cup U$ e teste em T , para fins estatísticos.

Após os rótulos serem propagados de S para U , um classificador supervisionado foi treinado em $S \cup U$ utilizando o espaço de características latente. Para tal tarefa, utilizamos o OPF e SVM (Capítulo 3). O OPF não possui hiper-parâmetros a serem configurados e, assim, é mais fácil de ser utilizado. Para o SVM, os valores ótimos foram encontrados para seus hiper-parâmetros σ (raio de influência), C (regularização) e tipo de *kernel* utilizando o procedimento de busca em grade nos intervalos $[0.1, 0.000001]$, $[1, 10000]$ e funções de *kernel* gaussiana de base radial e linear, respectivamente, utilizando 3 partições e amostragem aleatória e estratificada com 70% e 30% das amostras de $S \cup U$ utilizados

para treinamento e validação, respectivamente.

Os classificadores treinados foram testados em T e sua eficiência foi medida pelo coeficiente κ de *Cohen* [15, 65]. Tal medida é mais adequada, uma vez que as bases de dados são desbalanceadas e a simples acurácia poderia revelar bons resultados e ainda assim errar todas as amostras de uma única classe. Assim, a performance de classificação de cada método no conjunto de teste é medida por:

$$\kappa = \frac{p_o - p_e}{1 - p_e}, \quad (5.1)$$

em que p_o é a acurácia comumente utilizada e

$$p_e = \frac{1}{N^2} \sum_k n_{k\alpha} n_{k\beta}, \quad (5.2)$$

em que k é o número de categorias, N é o numero de amostras e $n_{k\alpha}$ e $n_{k\beta}$ são as categorias preditas k dado os classificadores α e β respectivamente. No caso específico deste trabalho, α são as categorias reais do dado e β , as categorias preditas por um dos classificadores empregados. O coeficiente κ está no intervalo $[-1, 1]$, em que $\kappa \leq 0$ indica nenhuma concordância e $\kappa = 1$ indica total concordância entre dois anotadores.

Adicionalmente, foi computada a acurácia da propagação de rótulos em U para cada abordagem, que refere-se ao número de amostras rotuladas corretamente dividido pelo número total de amostras não-supervisionadas ($|U|$).

Por fim, a melhor abordagem para propagação de rótulos é aquela que fornece o melhor classificador supervisionado. Uma vez que considerou-se o κ como medida de eficiência, o melhor classificador supervisionado é aquele que fornece o melhor resultado de κ .

5.2 Experimentos base

Como descrito no Capítulo 4, a abordagem proposta para propagação de rótulos semi-automática, do inglês *semi-automatic label propagation* (SALP), utiliza o OPF-Semi no espaço projetado do t-SNE no 2D para propagar rótulos para amostras com maior confiança e o usuário para propagar rótulos para amostras com baixa confiança, respectivamente. O SALP é comparado com os experimentos base a seguir:

1. *Sem propagação de rótulos (NLP)*: SVM e OPF são treinados somente em S , ignorando o conjunto U .
2. *Propagação de rótulos automática (ALP)*: o conjunto U é inteiramente rotulado por um dos quatro métodos ALP abaixo e SVM e OPF são treinados em $S \cup U$.
 - (a) LapSVM utilizando o espaço de características latente nD .
 - (b) LapSVM utilizando o espaço projetado do t-SNE 2D.
 - (c) OPF-Semi utilizando o espaço de características latente nD .
 - (d) OPF-Semi utilizando o espaço projetado t-SNE 2D.

3. *Propagação de rótulos interativa (ILP)*: o conjunto U é totalmente ou parcialmente rotulado pelo usuário e SVM e OPF são treinados em $S \cup U$, como em [7]. Um usuário com conhecimento em aprendizado de máquina e classificação de padrões realiza a propagação manual.

Em todos os casos acima, o SVM e OPF treinados previamente são testados em T .

5.3 Bases de dados

A primeira base de dados considerada contém 5000 imagens de dígitos manuscritos de 0 a 9, um subconjunto da base original selecionado de forma aleatória, da conhecida e pública base de dados MNIST [34]. As imagens são de dimensão 28×28 pixels.

As três bases de dados seguintes contêm imagens provenientes de um procedimento automático que separa imagens microscópicas de parasitos intestinais humanos em três grupos: (i) *Helminth larvae* e suas impurezas fecais (3514 imagens); (ii) *Helminth eggs* e suas impurezas fecais (5112 imagens); e (iii) *Protozoan cysts* e suas impurezas fecais (9568 imagens). As imagens são de dimensão 200×200 pixels. A impureza fecal é uma classe adversa que apresenta exemplos muito similares às classes de parasitos. Na Figura 5.1, é possível notar a semelhança entre as diferentes categorias de parasitos (esquerda) e as impurezas muito similares às categorias de parasitos (direita). Ainda, considerou-se estas três bases de dados com e sem imagens de impurezas fecais, constituindo as cinco bases de dados para validação da proposta, além da MNIST. O número de categorias em cada base de dados é descrita a seguir: (i) *H.Larvae* apresenta uma categoria e impureza; (ii) *H.Eggs*, nove categorias (*H.nana*, *H.diminuta*, *Ancilostomideo*, *E.vermicularis*, *A.lumbricoides*, *T.trichiura*, *S.mansoni*, *Taenia*, e impureza); e (iii) *P.cysts*, sete categorias (*E.coli*, *E.histolytica*, *E.nana*, *Giardia*, *I.butschlii*, *B.hominis*, e impureza) respectivamente. Essas são as espécies mais comuns de parasitos intestinais humanos no Brasil, que são responsáveis por problemas de saúde pública na maioria dos países tropicais [70]. As três bases de dados são desbalanceadas e apresentam um número consideravelmente maior de imagens de impurezas. As imagens de parasitos foram anotadas por especialistas do grupo de pesquisa do Laboratório de Ciência de Dados de Imagem, em inglês *Laboratory of Image Data Science*, no Instituto de Computação, na Unicamp.

A Tabela 5.1 apresenta o número de imagens em cada conjunto S , U e T após a divisão aleatória descrita na Seção 5.1.

Tabela 5.1: Número de amostras nos conjuntos S , U e T para cada base de dados: MNIST, parasitos, e parasitos com impureza (I).

Dataset	$ S $	$ U $	$ S \cup U $	$ T $
MNIST	175	3325	3500	1500
H.Eggs	61	1176	1237	531
P.cysts	134	2562	2696	1156
H.Larvae (I)	122	2337	2459	1055
H.Eggs (I)	178	3400	3578	1534
P.cysts (I)	334	6363	6697	2871

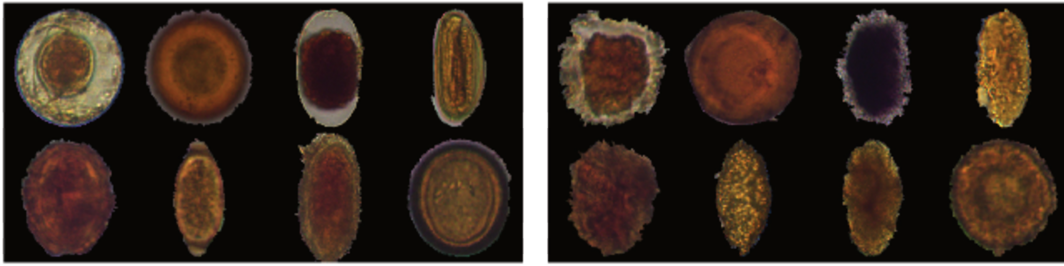


Figura 5.1: Exemplos da base de dados de Parasitos para cada espécie em *Helminth Eggs* (esquerda) e impurezas fecais similares (direita).

5.4 Detalhes de implementação

5.4.1 Extração de características

A Figura 5.2 apresenta as arquiteturas de AuNN empregadas para as bases de dados MNIST e de parasitos. As redes neurais foram implementadas em Keras [10] com 6 camadas de convolução com filtros de tamanho 3×3 : 3 para o *encoder* e 3 para o *decoder*. Após cada camada convolucional, empregou-se a função de ativação ReLU, além de operações de *Max-Pooling* no *encoder* e de *Up-Sampling* no *decoder*. As imagens de entrada foram normalizadas no intervalo $[0, 1]$, enquanto na última camada do *decoder* utilizou-se a função de ativação *sigmoid* para fornecer uma imagem reconstruída normalizada. Os números de filtros foram escolhidos de acordo com a base de dados: para a MNIST, as camadas de convolução apresentam 16, 8, 8, 8, 8 e 16 filtros; e para as 5 bases de parasitos, 32, 16, 8, 8, 16 e 32 filtros respectivamente. Como função de custo, considerou-se o erro médio quadrático, que foi empregado para o tipo de AuNN escolhida em [42]. Utilizou-se 50 para as bases de dados mais fáceis (MNIST e *Helminth Eggs* sem impureza) e 100 para as outras bases. Para a MNIST, o espaço de características latente contém 128 dimensões e, para as bases de parasitos, 5000 dimensões.

5.4.2 Projeção

Diferentes escolhas de parâmetros do t-SNE fornecem diferentes projeções no 2D [82]. Em [76], os autores sugerem que o algoritmo é robusto para valores de perplexidade entre 5 e 50. Desta forma, para o intervalo de 1000 a 7000 amostras em $S \cup U$, escolheu-se configurar a perplexidade do t-SNE em 40 e o número máximo de iterações em 1000, uma vez que tal configuração fornece projeções adequadas para a propagação de rótulos [7].

5.5 Resultados experimentais

A discussão do desempenho da abordagem proposta, mensurada pela performance dos classificadores treinados em $S \cup U$ no espaço de características latente e testados em T , é feita respondendo as seguintes questões:

- Qual espaço (latente nD , projetado 2D) é melhor para ALP? (Seção 5.5.1)

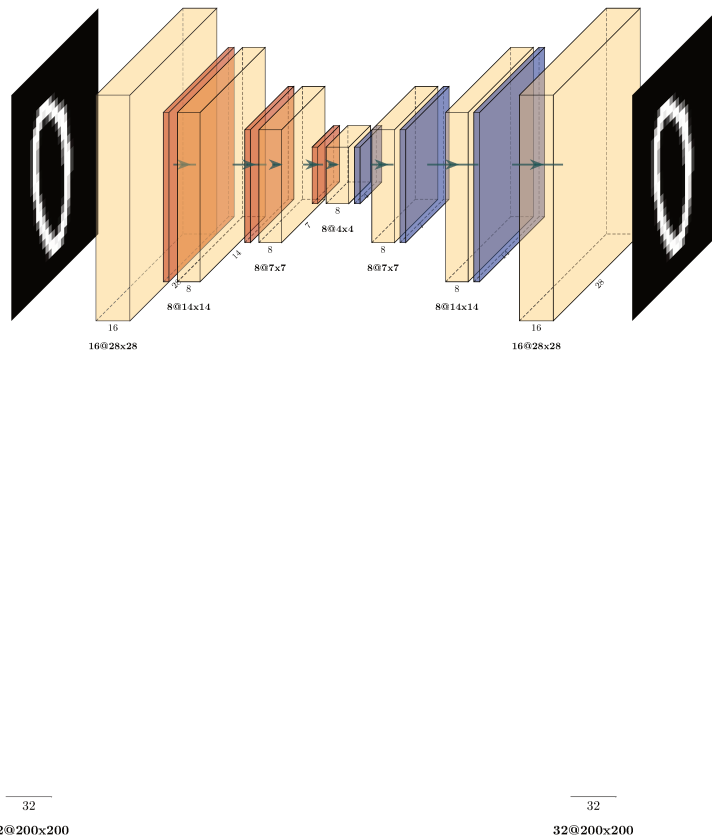


Figura 5.2: Arquiteturas da AuNN para as bases de dados MNIST (superior) e parasitos (inferior). As camadas em amarelo são as camadas de convolução, as camadas em vermelho no começo de cada rede são as operações de *Max-Pooling* e as camadas em azul são operações de *Up-Sampling*.

- Como escolher o limiar de confiança τ ? (Seção 5.5.2)
- Qual abordagem (manual, semi-automática, automática) melhor propaga rótulos de S para U ? (Seção 5.5.3)
- Qual é o valor agregado pelo SALP? (Seção 5.5.4)
- Como os resultados dependem da qualidade da projeção? (Seção 5.5.5)

5.5.1 Influência da redução do espaço de características do nD para 2D

A Tabela 5.2 apresenta valores de média e desvio padrão de κ de *Cohen* para os classificadores supervisionados no conjunto T para cada base de dados, bem como os tamanhos dos conjuntos S , U e $S \cup U$, e os valores de acurácia médios na propagação de rótulos para LapSVM e OPF-Semi empregados no espaço de características nD e também no projetado 2D, além da opção sem propagação de rótulos. Primeiramente, é possível notar que o LapSVM em algumas vezes tem desempenho melhor e outras vezes pior em nD comparado com 2D, dependendo da base de dados. Porém, o OPF-Semi consistentemente

mostra um impacto positivo na redução do espaço de características independentemente da base de dados. Tal fato pode ser notado mesmo quando o desempenho na propagação de rótulos não é o melhor.

Tabela 5.2: Valores de κ médio e desvio padrão para os classificadores SVM e OPF no conjunto T para MNIST, parasitos sem impureza e parasitos com impureza (I). Os melhores resultados para cada base de dados são mostrados em negrito.

Base de Dados	Técnica de Propagação	$ S $	$ U $	Acurácia de Propagação Média	$ S \cup U $	κ médio (SVM)	κ médio (OPF)
MNIST	Sem prop. rótulos	175	-	-	175	0.813415 $\pm$ 0.001	0.709450 $\pm$ 0.021
	LapSVM (nD)	175	3325	0.095639	3500	0.000000 $\pm$ 0.000	0.051110 $\pm$ 0.006
	OPF-Semi (nD)	175	3325	0.763308	3500	0.736685 $\pm$ 0.053	0.716913 $\pm$ 0.048
	LapSVM (2D)	175	3325	0.574236	3500	0.521970 $\pm$ 0.065	0.580445 $\pm$ 0.047
	OPF-Semi (2D)	175	3325	0.796592	3500	0.780197 $\pm$ 0.008	0.751794 $\pm$ 0.010
H.Eggs	Sem prop. rótulos	61	-	-	61	0.961366 $\pm$ 0.023	0.941358 $\pm$ 0.026
	LapSVM (nD)	61	1176	0.886338	1236	0.873472 $\pm$ 0.035	0.877344 $\pm$ 0.037
	OPF-Semi (nD)	61	1176	0.947563	1236	0.938317 $\pm$ 0.049	0.938323 $\pm$ 0.051
	LapSVM (2D)	61	1176	0.768141	1236	0.722691 $\pm$ 0.041	0.727673 $\pm$ 0.043
	OPF-Semi (2D)	61	1176	0.982993	1236	0.983621 $\pm$ 0.011	0.982146 $\pm$ 0.008
P.cysts	Sem prop. rótulos	134	-	-	134	0.823106 $\pm$ 0.016	0.762682 $\pm$ 0.008
	LapSVM (nD)	134	2562	0.521598	2696	0.346761 $\pm$ 0.001	0.371770 $\pm$ 0.005
	OPF-Semi (nD)	134	2562	0.802238	2696	0.740287 $\pm$ 0.036	0.724182 $\pm$ 0.053
	LapSVM (2D)	134	2562	0.787666	2696	0.597541 $\pm$ 0.212	0.592956 $\pm$ 0.187
	OPF-Semi (2D)	134	2562	0.838017	2696	0.801953 $\pm$ 0.021	0.786383 $\pm$ 0.022
H.Larvae (I)	Sem prop. rótulos	122	-	-	122	0.375378 $\pm$ 0.333	0.531080 $\pm$ 0.035
	LapSVM (nD)	122	2337	0.882613	2459	0.121253 $\pm$ 0.086	0.173416 $\pm$ 0.088
	OPF-Semi (nD)	122	2337	0.919075	2459	0.601003 $\pm$ 0.073	0.602001 $\pm$ 0.066
	LapSVM (2D)	122	2337	0.127086	2459	0.000000 $\pm$ 0.000	0.008665 $\pm$ 0.005
	OPF-Semi (2D)	122	2337	0.924405	2459	0.569164 $\pm$ 0.070	0.556642 $\pm$ 0.059
H.Eggs (I)	Sem prop. rótulos	178	-	-	178	0.705972 $\pm$ 0.037	0.568304 $\pm$ 0.034
	LapSVM (nD)	178	3400	0.654118	3578	0.000000 $\pm$ 0.000	0.076043 $\pm$ 0.016
	OPF-Semi (nD)	178	3400	0.504510	3578	0.373763 $\pm$ 0.029	0.391894 $\pm$ 0.021
	LapSVM (2D)	178	3400	0.146274	3578	0.086608 $\pm$ 0.031	0.109734 $\pm$ 0.035
	OPF-Semi (2D)	178	3400	0.729608	3578	0.611144 $\pm$ 0.071	0.544552 $\pm$ 0.047
P.cysts (I)	Sem prop. rótulos	334	-	-	334	0.628584 $\pm$ 0.024	0.476051 $\pm$ 0.010
	LapSVM (nD)	334	6363	0.622662	6697	0.232800 $\pm$ 0.104	0.202826 $\pm$ 0.030
	OPF-Semi (nD)	334	6363	0.468804	6697	0.343356 $\pm$ 0.034	0.337168 $\pm$ 0.032
	LapSVM (2D)	334	6363	0.128818	6697	0.045538 $\pm$ 0.038	0.079854 $\pm$ 0.024
	OPF-Semi (2D)	334	6363	0.605427	6697	0.429731 $\pm$ 0.013	0.396645 $\pm$ 0.009

5.5.2 A escolha do limiar de confiança

Como mencionado na Seção 4.3, o usuário precisa escolher um limiar τ para especificar quais rótulos propagados automaticamente são mantidos e quais são substituídos pela propagação manual. A Figura 5.3 apresenta as projeções de todas as amostras e aquelas mais confiáveis selecionadas pelo usuário para as seis bases de dados estudadas. É possível observar que o limiar τ varia relativamente pouco (entre 0,5 ou 0,6) para as bases de dados. Esse resultado indica que um bom valor a ser configurado inicialmente é $\tau = 0,6$, que pode ser melhorado pelo usuário dependendo da atual distribuição da confiança na projeção. De modo geral, pode-se entender que quanto mais desafiadora é a base de dados, menor é o limiar τ .

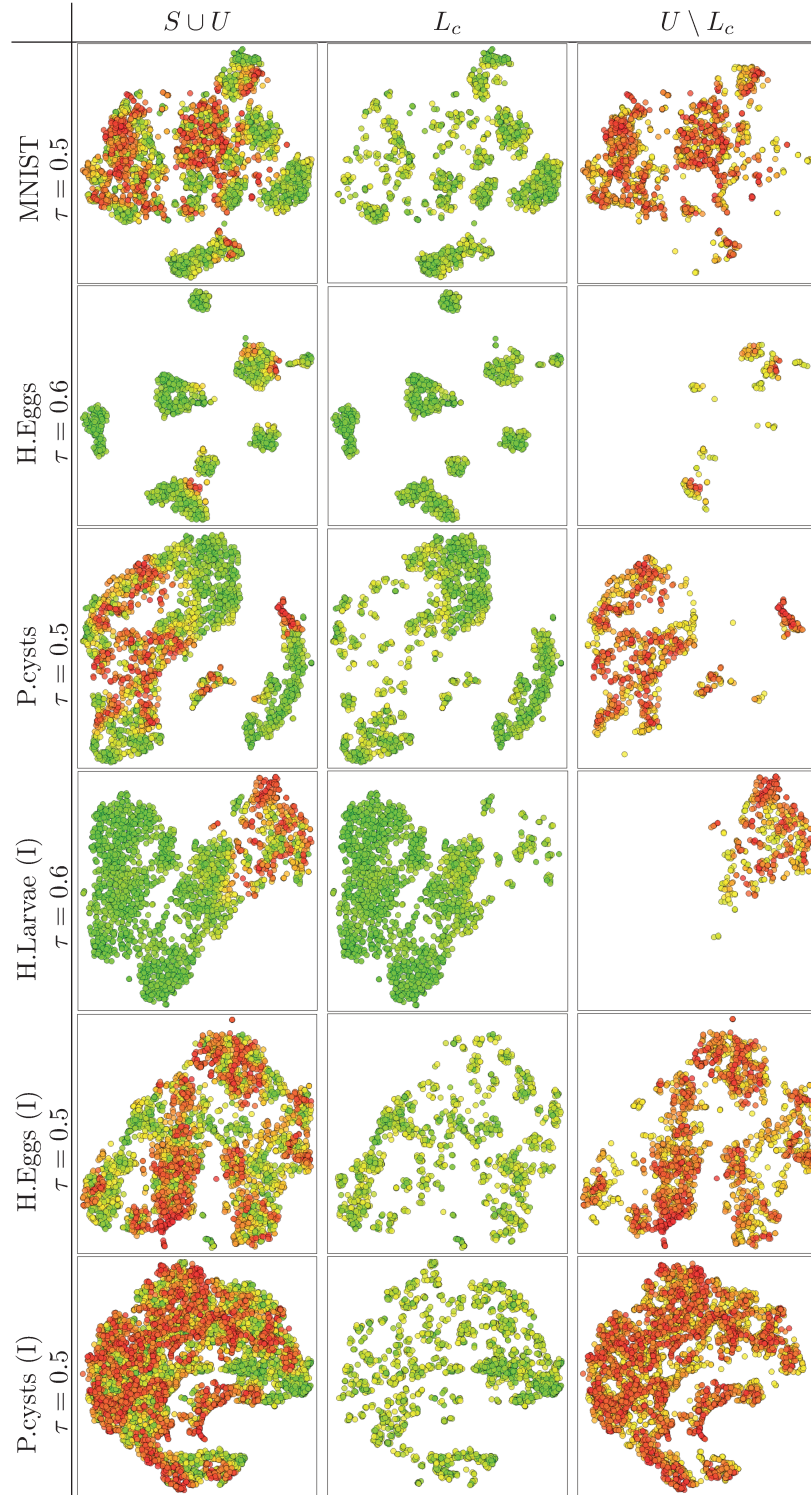


Figura 5.3: Projeções coloridas pela confiança do rótulo (vermelho=baixa confiança, verde=alta confiança). Linhas são base de dados (mais fáceis em cima, mais difíceis embaixo). Colunas mostram o conjunto todo de amostras supervisionadas e não-supervisionadas $S \cup U$, o conjunto L_c das amostras com maior confiança rotuladas por ALP e o conjunto $U \setminus L_c$ das amostras com menor confiança que serão apresentadas ao usuário para a propagação manual de rótulos.

5.5.3 A melhor abordagem de propagação de rótulos

A Tabela 5.2 mostra que o OPF-Semi em 2D é a melhor abordagem para propagação automática de rótulos (ALP). Consequentemente, a próxima questão é quão bem esse método

seria comparado à propagação interativa de rótulos (ILP) [7], que pode fazer a propagação manual para qualquer amostra não-supervisionada, ou o novo método proposto para propagação semi-automática de rótulos (SALP), que faz a propagação manual apenas para as amostras não-supervisionadas de menor confiança. A Figura 5.4 ilustra as projeções para ILP e SALP para as bases de dados estudadas. Uma vantagem fundamental do SALP sobre o ILP é que o SALP apresenta somente as amostras menos confiáveis (de acordo com o OPF-Semi no 2D) para o usuário, de forma a reduzir o esforço necessário para entender a imagem (e também reduzir a confusão e a sobreposição na projeção), fazendo a tarefa de propagar rótulos interativamente mais simples. A seguir, são discutidas várias observações que relacionam o ILP ao SALP na Figura 5.4, bem como as observações obtidas durante o atual processo de propagação interativa.

Para a MNIST, o usuário propaga rótulos para 1864 amostras não-supervisionadas em média (sobre as três partições consideradas) quando utiliza o ILP. Usando o SALP, esse número cai para 1182 amostras. Esse padrão revela um menor esforço para o SALP é consistente para todas as outras bases de dados, discutidas a seguir.

Para a base *H.Eggs*, a projeção do ILP apresenta amostras com melhor separação de grupos de classes distintas (cores). Isso indica que separar tais classes no espaço de características é relativamente fácil, que pode ser confirmado pelo fato de que existem muito poucas amostras com baixa confiança depois de empregar o OPF-Semi no 2D (pontos pretos na projeção do SALP). Assim, enquanto a propagação de rótulos utilizando o ILP pode ser feita facilmente dada a boa separação de agrupamentos, a propagação de rótulos utilizando o SALP é ainda mais fácil, uma vez que, além da boa separação dos agrupamentos, existe um menor número de amostras para rotular. Neste caso, o usuário propagou rótulos para 1171 amostras utilizando ILP e somente 154 amostras utilizando SALP.

Para *P.cysts*, as projeções apresentam uma separação visual menos clara de pontos de mesma classe (mesma cor) nos grupos. Isso torna a propagação interativa de rótulos mais desafiadora para ILP e SALP. O usuário propagou rótulos para 1999 amostras utilizando ILP e 919 amostras utilizando SALP. Para o SALP, o OPF-Semi no 2D propagou rótulos em regiões mais centrais dos grupos visíveis, em que, conseqüentemente, a confiança é alta. O restante das regiões de confusão (pontos pretos) é resolvido pelo usuário.

Para *H.Larvae*, é possível notar que as amostras supervisionadas de impurezas (verde) estão espalhadas por toda a projeção, enquanto as amostras supervisionadas de *H.Larvae* (red) estão mais concentradas na parte superior e à direita da projeção. Dada essa separação visual consideravelmente boa, propagar o rótulo das impurezas utilizando ILP é relativamente fácil para a maior parte da projeção, considerando que o esforço do usuário está em selecionar um único grande grupo. Contudo, tal projeção ainda requer esforço manual. Utilizando o SALP, as áreas mais simples são resolvidas automaticamente, enquanto somente as regiões mais complicadas são deixadas para o usuário resolver, como visto na parte superior direita onde o verde se mistura com o vermelho.

Para a base *H.Eggs* com impurezas, as amostras supervisionadas de impureza (cinza) estão localizadas entre grupos de pontos coloridos (classes de *H.Eggs* de fato) na projeção. Comparado com as bases de dados anteriores, existem mais pontos em preto para o SALP, devido a dificuldade do OPF-Semi no 2D em propagar rótulos automaticamente.

	ILP	SALP	Rotulado (OPF-Semi)	Rotulado (OPF-Semi+user)	$ U $	$ L_c $	$ U \setminus L_c $	$ L_i $
MNIST $\tau = 0.5$					3325	1690	1635	1182
H.Eggs $\tau = 0.6$					1176	1022	154	154
P.cysts $\tau = 0.5$					2562	1643	919	666
H.Larvae (I) $\tau = 0.6$					2337	1813	524	524
H.Eggs (I) $\tau = 0.5$					3400	983	2417	2076
P.cysts (I) $\tau = 0.5$					6363	2215	4148	1733

Figura 5.4: Comparação de diferentes métodos de propagação de rótulos (colunas) para diferentes bases de dados (linhas). Da esquerda para a direita: ILP, SALP, rótulos propagados automaticamente pelo OPF-Semi e resultado final de propagação de rótulos do SALP juntamente com o OPF-Semi. Cores indicam rótulos de amostras supervisionadas (ILP, SALP) ou amostras não-anotadas e rótulos propagados (OPF-Semi, OPF-Semi+user). Amostras em preto são amostras a serem consideradas para a propagação de rótulos manual (três colunas à esquerda), e amostras desconsideradas na propagação de rótulos manual (coluna à direita). Os tamanhos dos conjuntos de amostras são apresentados à direita.

Isso coincide com o fato de bases de dados com impurezas serem consideravelmente mais

difíceis. Para essa base de dados, o usuário propaga rótulos para mais pontos utilizando SALP (2076) do que ILP (1787). Tal diminuição na propagação de rótulos entre os métodos suporta a evidência de que a simplificação da projeção do SALP pela remoção dos pontos com maior confiança, ainda que pequena neste caso, foi suficiente para auxiliar o usuário a ver mais estruturas na projeção, que o usuário pôde propagar os rótulos. Também, como para *P.cysts*, pode-se observar que o OPF-Semi no 2D propaga rótulos nas regiões centrais dos grupos visíveis, deixando o resto para o usuário.

Finalmente, para *P.cysts* com impurezas, as amostras supervisionadas de impureza (marrom) estão espalhadas ao longo de toda a projeção. As amostras supervisionadas de *P.cysts* (outras cores diferentes de marrom) estão significativamente misturadas e a projeção mostra pouca estrutura: de forma geral, um grupo grande e outro menor com forma crescente. Essa é a base de dados mais desafiadora para propagação de rótulos manual. Como para *H.Eggs*, o OPF-Semi no 2D encontrou poucas amostras confiáveis, fazendo com que o esforço da propagação de rótulos manual seja similar para ILP e SALP. Isso pode ser comprovado pelo número de pontos que o usuário de fato propagou rótulos (1787 com ILP e 1733 com SALP). Ainda que essas figuras pareçam quase idênticas, a maior vantagem do SALP é que o OPF-Semi no 2D é capaz de filtrar os pontos do caso mais fácil (com maior confiança), deixando o esforço do usuário para os casos mais difíceis.

5.5.4 Valor agregado pelo SALP

Como visto anteriormente, o SALP é capaz de reduzir o esforço do usuário na propagação de rótulos. Uma questão final a ser respondida é: quanto é o valor agregado pelo SALP em termos de classificação, comparado ao método anterior e similar (ILP) ou, em contrapartida, ao melhor método totalmente automático (OPF-Semi no 2D)? A Tabela 5.3 responde essa questão apresentando os valores de média e desvio padrão de κ no conjunto T para cada base de dados. A tabela também mostra o tamanho dos conjuntos S , U , e $S \cup U$, além da acurácia média na propagação de rótulos para OPF-Semi no 2D, ILP e SALP. É importante salientar que a acurácia de propagação para SALP considera não somente as amostras com menor confiança rotuladas pelo usuário, mas também aquelas de maior confiança automaticamente rotuladas pelo OPF-Semi no 2D. É possível notar que o SALP obteve consistentemente os melhores resultados de classificação nos conjuntos T não vistos anteriormente, para todas as bases de dados exploradas neste estudo. Isso prova que o SALP agrega, de fato, valor em relação aos outros métodos existentes, ou seja, utilizar o SALP resulta em melhores classificadores no final. Separadamente, é possível observar que para todas as bases de dados, exceto as mais simples (MNIST e *H.Eggs*), o SALP também obteve a melhor acurácia na propagação de rótulos.

5.5.5 Como os resultados dependem da qualidade da projeção

Ademais, foram realizados os mesmos experimentos discutidos nas seções anteriores utilizando o UMAP [44] ao invés do t-SNE como técnica de projeção. De forma geral, notou-se resultados piores em termos de acurácia na propagação de rótulos e resultado de classificação (κ) do que utilizando o t-SNE. Isso indica que preservar a qualidade da

Tabela 5.3: Valores de κ médio e desvio padrão para os classificadores SVM e OPF no conjunto T para as bases de dados estudadas. Os melhores resultados por base de dados são mostrados em negrito.

Base de Dados	Técnica de Propagação	$ S $	$ L_i \cup L_c $ médio	Acurácia de Propagação Média	$ S \cup L_i \cup L_c $ médio	κ médio (SVM)	κ médio (OPF)
MNIST	OPF-Semi (2D)	175	3325	0.796592	3500	0.780197 ± 0.008	0.751794 ± 0.010
	ILP	175	1864	0.974718	2039	0.844264 ± 0.027	0.776241 ± 0.036
	SALP	175	2872	0.947192	3047	0.885855 $\pm$ 0.030	0.839161 $\pm$ 0.018
H.Eggs	OPF-Semi (2D)	61	1175	0.982993	1236	0.983621 ± 0.011	0.982146 ± 0.008
	ILP	61	1171	0.996014	1232	0.986624 $\pm$ 0.009	0.987364 $\pm$ 0.003
	SALP	61	1175	0.992347	1236	0.989582 $\pm$ 0.005	0.983639 $\pm$ 0.007
P.cysts	OPF-Semi (2D)	134	2562	0.838017	2696	0.801953 ± 0.021	0.786383 ± 0.022
	ILP	134	1999	0.947177	2133	0.851948 ± 0.006	0.841023 ± 0.002
	SALP	134	2309	0.951119	2443	0.877566 $\pm$ 0.011	0.850232 $\pm$ 0.015
H.Larvae (I)	OPF-Semi (2D)	122	2337	0.924405	2459	0.569164 ± 0.070	0.556642 ± 0.059
	ILP	122	2080	0.981273	2202	0.727843 ± 0.013	0.723049 ± 0.016
	SALP	122	2337	0.986730	2459	0.805388 $\pm$ 0.014	0.748340 $\pm$ 0.036
H.Eggs (I)	OPF-Semi (2D)	178	3400	0.729608	3578	0.611144 ± 0.071	0.544552 ± 0.047
	ILP	178	1547	0.914358	1725	0.683544 ± 0.033	0.593104 ± 0.034
	SALP	178	3059	0.959611	3237	0.866121 $\pm$ 0.043	0.725803 $\pm$ 0.025
P.cysts (I)	OPF-Semi (2D)	334	6363	0.605427	6697	0.429731 ± 0.013	0.396645 ± 0.009
	ILP	334	1787	0.826867	2121	0.589643 ± 0.036	0.472148 ± 0.008
	SALP	334	3948	0.864390	4282	0.648831 $\pm$ 0.043	0.543963 $\pm$ 0.016

vizinhança de uma projeção (que é maior para o t-SNE do que para UMAP) é um fator importante para o método proposto. Também, foi possível observar que os padrões que relacionam o tamanho de cada base de dados com a sua dificuldade se mantiveram nos resultados obtidos utilizando o UMAP. Tal padrão indica que os resultados do t-SNE não são decorrentes do uso de projeções que favorecem o t-SNE em relação ao UMAP e de uma configuração favorável de parâmetros do método de projeção: tanto UMAP quanto t-NSE são, de fato, métodos não-determinísticos.

5.6 Discussão

A seguir, são discutidos vários aspectos do método proposto.

5.6.1 O espaço de características: nD versus 2D

Uma questão interessante diz respeito a qual o desempenho da propagação de rótulos automática (ALP) ao usar o espaço de características latente nD comparado com o espaço projetado 2D. A Figura 5.5 ilustra os valores médios de κ de classificação para LapSVM e OPF-Semi utilizando esses dois espaços para os classificadores OPF e SVM. As bases de dados foram ordenadas ao longo do eixo x em ordem decrescente dos valores de κ do OPF-Semi no 2D. É possível notar que o LapSVM apresenta melhores resultados no 2D do que no nD para metade das bases de dados, enquanto o OPF-Semi, para todas as bases de dados. Isso indica, essencialmente, que o espaço projetado no 2D criado pelo t-SNE é capaz de reter toda a informação necessária que possibilite a propagação de rótulos desejada e, depois, a construção de um bom classificador. Tal resultado é importante a medida que justifica apresentar a projeção do espaço no 2D para o usuário como única informação para o usuário realizar a propagação de rótulos manual. Também, é possível observar um padrão dos valores de κ ao longo do eixo x tanto para o 2D, quanto para o

nD , que é compatível com a dificuldade percebida das bases de dados: valores altos de κ correspondem a bases mais fáceis (esquerda), enquanto valores baixos de κ correspondem a bases mais complexas com impurezas (direita). Por fim, os valores de κ para ILP e SALP também foram colocados no gráfico (curvas nas figuras). Em todos os casos, tais curvas estão acima dos métodos automáticos, mostrando o esforço manual adicionado. A curva do SALP está acima da curva do ILP, mostrando que a melhor abordagem é alcançada pela *combinação* da propagação de rótulos automática e manual (ambos executados no espaço 2D).

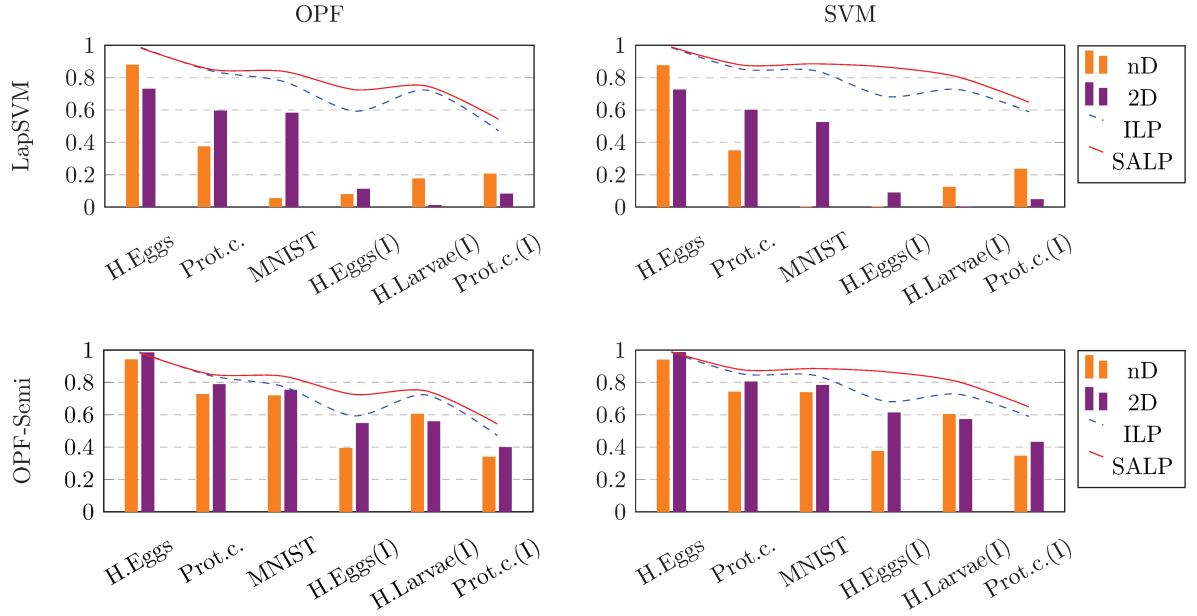


Figura 5.5: Valores de κ para as bases de dados estudadas, considerando os classificadores OPF e SVM (colunas) e métodos de propagação de rótulos automáticos LapSVM e OPF-Semi (linhas). Curvas mostram valores de κ para ILP e SALP para fins de comparação. As bases de dados estão ordenadas da mais fácil para a mais difícil para classificar (esquerda para direita) com base nos resultados de SALP.

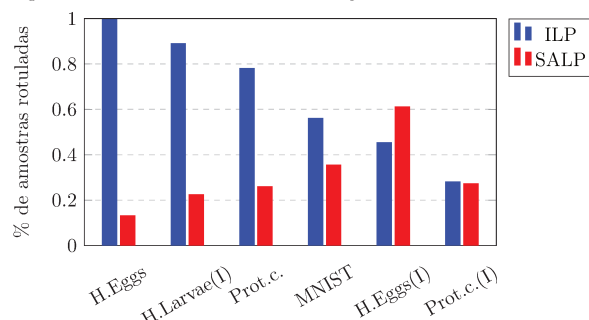
5.6.2 Redução de esforço do usuário

Embora tenha alcançado os melhores resultados de classificação comparado com os métodos de propagação de rótulos totalmente automáticos e totalmente manual (ILP), o SALP também reduziu o esforço manual quando comparado ao ILP. A Figura 5.6 ilustra isso retratando a porcentagem de amostras rotuladas pelo usuário dividida pelo número total de amostras a serem rotuladas (U) por base de dados, considerando ILP e SALP. Para o SALP, essa medida exclui as amostras rotuladas automaticamente pelo OPF-Semi no 2D. As bases de dados são ordenadas ao longo do eixo x de forma crescente pelo valor de $|U|$, isto é, da menor base de dados para a maior. Várias percepções podem ser obtidas a partir da Figura 5.6. Primeiramente, assumindo que o esforço do usuário é proporcional ao número de amostras rotuladas e que o esforço por amostra é o mesmo para o ILP e SALP — uma vez que os dois métodos compartilham da mesma visualização e

interação—, é possível observar que o esforço do usuário com o ILP é sempre maior do que com o SALP, com exceção para *H.Eggs* com impurezas. Depois, a porcentagem de amostras propagadas com ILP cai de acordo com o tamanho da base de dados. Isso pode ser explicado pela dificuldade em propagar rótulos em projeções com muitos pontos, nas quais as aglomerações e sobreposições se tornam um problema. A tendência oposta é percebida para o SALP: a porcentagem de amostras propagadas aumenta com o tamanho da base de dados. Essa tendência é quebrada para a maior base de dados (*Prot.c(I)*, 6363 amostras), duas vezes maior do que a segunda maior base (*H.Eggs(I)*, 3400 amostras). Na maior base de dados a projeção é consideravelmente densa e aglomerada, o que faz com que a propagação manual seja igualmente difícil para ILP e SALP.

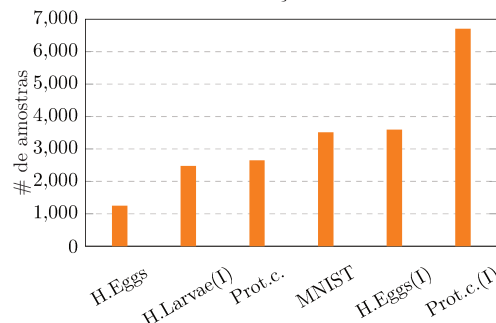
Em paralelo, é possível observar que o número de amostras em $U \setminus L_c$, aquelas abaixo o limiar τ e de rótulos com baixa confiança para o OPF-Semi, também aumenta com o tamanho da base de dados. Assim, a quantidade de amostras em $U \setminus L_c$ apresentada ao usuário para propagar rótulos com o SALP aumenta com o tamanho da base de dados. Um caso, por exemplo, é a base de dados *H.Eggs* com impurezas. Tal base apresenta a maior porcentagem de amostras anotadas pelo SALP, ultrapassando também o ILP. Isso pode ser explicado pelo tamanho da base de dados (segunda maior) e ao fato de a projeção parecer razoavelmente mais fácil para propagar rótulos para o grande grupo de impurezas (Figuras 5.6 e 5.7).

Porcentagem de amostras rotuladas em função de bases de dados distintas



(a)

Número de amostras em função de bases de dados distintas



(b)

Figura 5.6: (a) porcentagem de amostras rotuladas em U pelos métodos de propagação ILP e SALP, e (b) o número de amostras nas seis bases de dados estudadas.

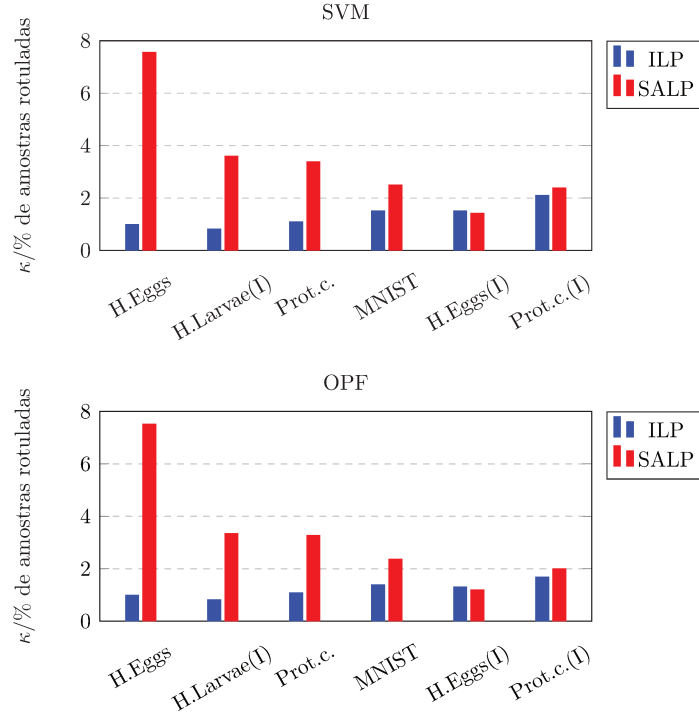


Figura 5.7: Ganho normalizado, isto é, κ dividido pela porcentagem de amostras manualmente rotuladas pelo ILP e SALP para os classificadores SVM e OPF para as seis bases de dados estudadas (ordenadas pelo tamanho da esquerda para a direita).

5.6.3 Limitações

Várias limitações existem da validação do método proposto. Primeiramente, a validação do método é limitada a seis bases de dados, duas técnicas de classificação e somente um usuário realizando a propagação de rótulos manual. Quantificar o valor agregado pelo SALP para mais combinações (de bases de dados, classificadores, usuários) traria maiores convicções sobre a eficácia do método proposto. Ainda, enquanto a capacidade dos espaços projetados pelo t-SNE no 2D em capturar informação necessária para uma razoável propagação de rótulos foi demonstrada para os métodos automáticos e manuais, o efeito das distorções resultantes da projeção t-SNE ainda não foi mensurado de forma quantitativa. Métricas de acurácia de projeções, como estresse, confiabilidade, continuidade e acertos na vizinhança [47], podem ser usadas de forma a encontrar tal correlação. Por outro lado, utilizar ferramentas visuais [47] que destacam tais erros em áreas específicas da projeção podem auxiliar o usuário a obter uma propagação de rótulos manual mais rápida ou acurada.

Capítulo 6

Conclusão e Trabalhos Futuros

No presente trabalho, foi proposta uma abordagem que combina o usuário e métodos automáticos para fornecer amostras rotuladas para casos de base de dados insuficientemente anotadas para o propósito de treinar modelos de classificação. Para isso, características foram extraídas utilizando Redes Neurais Autocodificadoras e, depois, reduzidas a um espaço de duas dimensões utilizando o algoritmo t-SNE. Depois, os rótulos foram propagados automaticamente das poucas amostras supervisionadas para as não-supervisionadas neste espaço 2D, enquanto a confiança na propagação é monitorada. Para amostras rotuladas de baixa confiança, é permitido que o usuário propague rótulos manualmente baseado em percepções visuais codificadas nas projeções 2D, que foram coloridas pelos rótulos das amostras supervisionadas. Vários resultados quantitativos foram encontrados. Primeiramente, mostrou-se que o espaço projetado no 2D conduz a maiores acurácias para propagação de rótulos automática, do que o espaço latente de alta dimensão extraído pela rede neural. Até onde se sabe, esse resultado é novo e sugere novos caminhos para redução de dimensionalidade tendo em vista a propagação de rótulos. Ainda, mostrou-se que o método proposto semi-supervisionado, que combina a propagação de rótulos automática do OPF-Semi como a propagação de rótulos manual realizada pelo usuário — ambas realizadas no espaço 2D — alcança melhores resultados de classificação do que a propagação de rótulos totalmente automática ou totalmente manual. Tal resultado abre caminhos para diferentes abordagens com o objetivo de combinar métodos automáticos e baseados no usuário para a engenharia de melhores sistemas de aprendizado de máquina.

Como trabalhos futuros, o intuito é continuar os estudos acerca do método de propagação de rótulos proposto, além de considerá-lo no cenário de aprendizado ativo. No aprendizado ativo, um algoritmo iterativamente aumenta a quantidade de dados supervisionados, a fim de possibilitar um desempenho similar ou maior do que aquele alcançado utilizando toda a base de dados, pela diminuição de tempo e/ou custo de rotulação da base de dados inteira. O método proposto, SALP, ainda não explorou a interação do usuário com abordagens de aprendizado ativo. Espera-se que o ciclo conduzido pelo aprendizado ativo possa melhorar os resultados de classificação, à medida que a acurácia de propagação aumenta. Ademais, estudos extensivos de validação serão realizados para mensurar o valor agregado do método proposto para mais tipos de bases de dados e métodos de classificação, além de utilizar técnicas adicionais de análise visual para auxiliar usuários a propagarem rótulos de forma melhor e mais rapidamente.

A Figura 6.1 ilustra os principais pontos a serem explorados, além combinar o aprendizado de características, aprendizado ativo e analítica visual de dados de forma paralela. Além do método não-supervisionado utilizado para o aprendizado de características, pode-se também explorar os métodos semi-supervisionados para este fim, bem como empregar técnicas de projeção generativas.

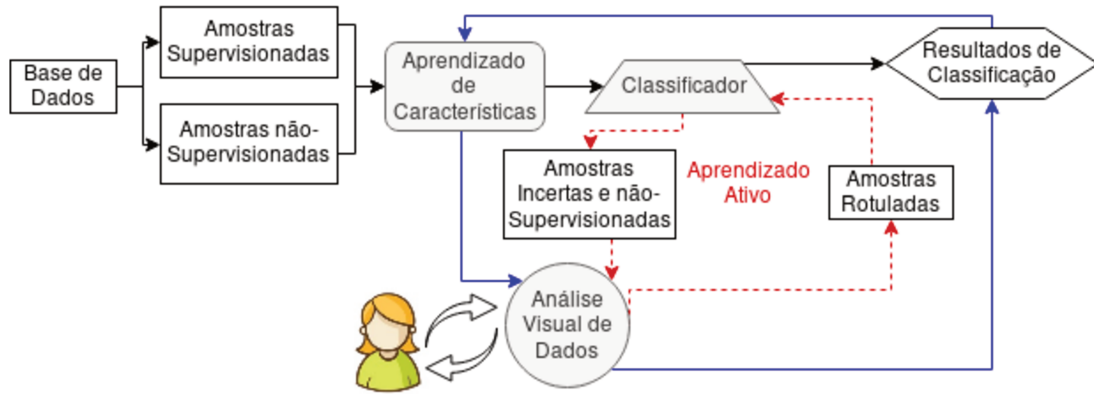


Figura 6.1: Método proposto para trabalhos futuros.

A partir de amostras não-supervisionadas, um espaço de características melhorado por analítica visual de dados pode ser dado por um algoritmo de aprendizado de características. O usuário pode começar supervisionando o representante dos agrupamentos, caso todas as amostras sejam não-supervisionadas. Tal espaço de características é dado como entrada a um classificador semi-supervisionado (ou não-supervisionado) que oferece as amostras mais incertas a serem projetadas para um usuário. O usuário propaga os rótulos para as amostras não-supervisionadas, que são utilizadas para realimentar o classificador. O ciclo de aprendizado ativo (vermelho) ocorre até que (i) o usuário esteja satisfeito, ou (ii) o método esteja estagnado. Se (ii), o conjunto rotulado é retornado para o ciclo de aprendizado de características (azul) para aprender um novo espaço de características e então, o aprendizado ativo sucede-se novamente. O processo termina quando (i) ocorre.

Trabalhos publicados

Conferências

- B. C. Benato, A. C. Telea, and A. X. Falcão. Semi-Supervised Learning with Interactive Label Propagation guided by Feature Space Projections. *The 31th SIBGRAPI - Conference on Graphics, Patterns and Images*, Foz do Iguaçu, PR, IEEE, Oct 2018.
- A. Z. Peixinho, B. C. Benato, L. G. Nonato, and A. X. Falcão. Delaunay Triangulation Data Augmentation guided by Visual Analytics for Deep Learning. *The 31th SIBGRAPI - Conference on Graphics, Patterns and Images*, Foz do Iguaçu, PR, IEEE, Oct 2018.
- S. B. Martins, B. C. Benato, B. F. Silva, C. L. Yasuda, and A. X. Falcão. Modeling normal brain asymmetry in MR images applied to anomaly detection without segmentation and data annotation. *SPIE on Medical Imaging: Computer-Aided Diagnosis*, San Diego, CA, Feb 2019.

Periódico

- B. C. Benato, J. F. Gomes, A. C. Telea, and A. X. Falcão. Semi-automatic data annotation guided by feature space projection. *Pattern Recognition*, 2019. (Submetido, em revisão)

Referências Bibliográficas

- [1] D. H. Ackley, G. E. Hinton, and T. J. Sejnowski. A learning algorithm for boltzmann machines. *Cognitive Science*, 9:147–169, 1985.
- [2] W. P. Amorim, A. X. Falcão, J. P. Papa, and M. H. Carvalho. Improving semi-supervised learning through optimum connectivity. *Pattern Recognition*, 60(C):72–85, December 2016.
- [3] D. Angluin. Queries and concept learning. *Machine Learning*, 2(4):319–342, Apr 1988.
- [4] D. H. Ballard. Modular learning in neural networks. In *Proceedings of the Sixth National Conference on Artificial Intelligence*, pages 279–284. San Francisco, CA: Morgan Kaufmann, 1987.
- [5] M. Belkin, P. Niyogi, and V. Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *J. Mach. Learn. Res.*, 7:2399–2434, December 2006.
- [6] B. C. Benato, J. F. Gomes, A. C. Telea, and A. X. Falcão. Semi-automatic data annotation guided by feature space projection. *Pattern Recognition*, 2019. (Submitted).
- [7] B. C. Benato, A. C. Telea, and A. X. Falcão. Semi-supervised learning with interactive label propagation guided by feature space projections. In *2018 31st SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*, pages 392–399, Oct 2018.
- [8] J. Bernard, M. Hutter, M. Zeppelzauer, D. Fellner, and M. Sedlmair. Comparing visual-interactive labeling with active learning: An experimental study. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):298–308, Jan 2018.
- [9] H. Bourlard and Y. Kamp. Auto-association by multilayer perceptrons and singular value decomposition. Manuscript M217, Philips Research Laboratory, 1987.
- [10] F. Chollet et al. Keras. <https://keras.io>, 2015.
- [11] A. T. da Silva, J. A. dos Santos, A. X. Falcão, R. S. Torres, and L. P. Magalhães. Incorporating multiple distance spaces in optimum-path forest classification to improve feedback-based learning. *Computer Vision and Image Understanding*, 116(4):510 – 523, 2012.

- [12] P. Demartines and J. Herault. Curvilinear component analysis: a self-organizing neural network for nonlinear mapping of data sets. *Neural Networks, IEEE Transactions on*, 8:148 – 154, 02 1997.
- [13] T. Dietterich. Overfitting and undercomputing in machine learning. *ACM Computing Surveys*, 27(3):326–327, September 1995.
- [14] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley, 2 edition, 2001.
- [15] J. L. Fleiss and J. Cohen. The equivalence of weighted kappa and the intraclass correlation coefficient as measures of reliability. *Educational and Psychological Measurement*, 33(3):613–619, 1973.
- [16] G. Forestier and C. Wemmert. Semi-supervised learning using multiple clusterings with limited labeled data. *Information Sciences*, 361–362:48–65, 2016.
- [17] A. Gogna and A. Majumdar. Semi supervised autoencoder. In Akira Hirose, Seiichi Ozawa, Kenji Doya, Kazushi Ikeda, Minhoo Lee, and Derong Liu, editors, *Neural Information Processing*, pages 82–89. Springer International Publishing, 2016.
- [18] M. Guillaumin, T. Mensink, J. Verbeek, and C. Schmid. Tagprop: Discriminative metric learning in nearest neighbor models for image auto-annotation. In *2009 IEEE 12th International Conference on Computer Vision*, pages 309–316, Sept 2009.
- [19] M. A. Hearst, S. T. Dumais, E. Osuna, J. Platt, and B. Scholkopf. Support vector machines. *IEEE Intelligent Systems and their Applications*, 13(4):18–28, July 1998.
- [20] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97, Nov 2012.
- [21] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, July 2006.
- [22] S. C. H. Hoi and M. R. Lyu. A semi-supervised active learning framework for image retrieval. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05)*, volume 2, pages 302–309 vol. 2, June 2005.
- [23] H. Hotelling. Analysis of a complex of statistical variables with principal components. *Journal of Educational Psychology*, 24:417–441, 1933.
- [24] B. Höferlin, R. Netzel, M. Höferlin, D. Weiskopf, and G. Heidemann. Inter-active learning of ad-hoc classifiers for video visual analytics. In *2012 IEEE Conference on Visual Analytics Science and Technology (VAST)*, pages 23–32, Oct 2012.
- [25] P. Ji, N. Zhao, S. Hao, and J. Jiang. Automatic image annotation by semi-supervised manifold kernel density estimation. *Information Sciences*, 281:648 – 660, 2014. Multimedia Modeling.

- [26] C. Käding, E. Rodner, A. Freytag, O. Mothes, B. Barz, and J. Denzler. Active learning for regression tasks with expected model output changes. In *British Machine Vision Conference 2018, BMVC 2018, Northumbria University, Newcastle, UK, September 3-6, 2018*, page 103, 2018.
- [27] J. Kim, J. Kwon Lee, and K. Mu Lee. Accurate image super-resolution using very deep convolutional networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.
- [28] D. P. Kingma, S. Mohamed, D. J. Rezende, and M. Welling. Semi-supervised learning with deep generative models. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 3581–3589. Curran Associates, Inc., 2014.
- [29] T. Kobayashi. Gan-based semi-supervised learning on fewer labeled samples. In *British Machine Vision Conference 2018, BMVC 2018, Northumbria University, Newcastle, UK, September 3-6, 2018*, page 99, 2018.
- [30] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.
- [31] A. Kumar, P. Sattigeri, and T. Fletcher. Semi-supervised learning with gans: Manifold invariance with improved inference. In *Advances in Neural Information Processing Systems 30*, pages 5534–5544. Curran Associates, Inc., 2017.
- [32] Y. LeCun. *Modeles connexionnistes de l'apprentissage (connectionist learning models)*. PhD thesis, Université P. et M. Curie (Paris 6), June 1987.
- [33] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, Nov 1998.
- [34] Y. LeCun and C. Cortes. MNIST handwritten digit database. 2010.
- [35] D. H. Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *International Conference on Machine Learning (ICML) 2013 Workshop : Challenges in Representation Learning (WREPL)*, 07 2013.
- [36] D. D. Lewis and W. A. Gale. A sequential algorithm for training text classifiers. In *SIGIR '94*, pages 3–12. Springer London, 1994.
- [37] David D. Lewis and Jason Catlett. Heterogeneous uncertainty sampling for supervised learning. In *Machine Learning Proceedings 1994*, pages 148 – 156. Morgan Kaufmann, San Francisco (CA), 1994.
- [38] W. Li and M. Sun. Semi-supervised learning for image annotation based on conditional random fields. In *Image and Video Retrieval*, pages 463–472. Springer Berlin Heidelberg, 2006.

- [39] J. Liu, M. Li, Q. Liu, H. Lu, and S. Ma. Image annotation via graph learning. *Pattern Recognition*, 42(2):218 – 228, 2009. Learning Semantics from Multimedia Content.
- [40] R. Liu, Y. Wang, T. Baba, D. Masumoto, and S. Nagata. Svm-based active feedback in image retrieval using clustering and unlabeled data. *Pattern Recognition*, 41(8):2645 – 2655, 2008.
- [41] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.
- [42] J. Masci, U. Meier, D. Cireşan, and J. Schmidhuber. *Stacked Convolutional Auto-Encoders for Hierarchical Feature Extraction*, pages 52–59. Springer Berlin Heidelberg, 2011.
- [43] R. Mash, B. Borghetti, and J. Pecarina. Improved aircraft recognition for aerial refueling through data augmentation in convolutional neural networks. In *Advances in Visual Computing*, pages 113–122, Cham, 2016. Springer International Publishing.
- [44] L. McInnes, J. Healy, and J. Melville. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. *ArXiv e-prints*, February 2018.
- [45] P. A. V. Miranda and A. X. Falcão. Links between image segmentation based on optimum-path forest and minimum cut in graph. *Journal of Mathematical Imaging and Vision*, 35(2):128–142, Oct 2009.
- [46] I. Muslea, S. Minton, and C. A. Knoblock. Active + semi-supervised learning = robust multi-view learning. In *Proceedings of the Nineteenth International Conference on Machine Learning, ICML '02*, pages 435–442. Morgan Kaufmann Publishers Inc., 2002.
- [47] L. Nonato and M. Aupetit. Multidimensional projection for visual analytics: Linking techniques with distortions, tasks, and layout enrichment. *IEEE Transactions on Visualization e Computer Graphics*, 2018.
- [48] S. J. Nowlan and G. E. Hinton. Simplifying neural networks by soft weight-sharing. *Neural Computation*, 4(4):473–493, 1992.
- [49] N. Papernot, M. Abadi, Ú. Erlingsson, I. Goodfellow, and K. Talwar. Semi-supervised knowledge transfer for deep learning from private training data. In *Proceedings of the International Conference on Learning Representations*, 2017.
- [50] S. Patra and L. Bruzzone. A batch-mode active learning technique based on multiple uncertainty for svm classifier. *IEEE Geoscience and Remote Sensing Letters*, 9(3):497–501, May 2012.
- [51] A. Z. Peixinho, B. C. Benato, L. G. Nonato, and A. X. Falcão. Delaunay triangulation data augmentation guided by visual analytics for deep learning. In *2018 31st SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*, pages 384–391, Oct 2018.

- [52] M. A. Ranzato, C. S. Poultney, S. Chopra, and Y. LeCun. Efficient learning of sparse representations with an energy-based model. In Bernhard Schölkopf, John C. Platt, and Thomas Hofmann, editors, *Neural Information Processing Systems (NIPS)*, pages 1137–1144. MIT Press, 2006.
- [53] M. A. Ranzato and M. Szummer. Semi-supervised learning of compact document representations with deep networks. In *Proceedings of the 25th International Conference on Machine Learning*, International Conference on Machine Learning (ICML) '08, pages 792–799. ACM, 2008.
- [54] A. Rasmus, M. Berglund, M. Honkala, H. Valpola, and T. Raiko. Semi-supervised learning with ladder networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 3546–3554. Curran Associates, Inc., 2015.
- [55] P. E. Rauber. Análise visual aplicada à análise de imagens. In Campinas SP Universidade Estadual de Campinas, Instituto de Computação e Universidade de Groningen, editor, *Tese (doutorado)*, page 159. 2017.
- [56] P. E. Rauber, S. G. Fadel, A. X. Falcão, and A. C. Telea. Visualizing the hidden activity of artificial neural networks. *IEEE Transactions on Visualization and Computer Graphics (Proceedings of the Visual Analytics Science and Technology 2016)*, 23(01), January 2017.
- [57] P.E. Rauber, A.X. Falcão, and A.C. Telea. Visualizing time-dependent data using dynamic t-sne. In *Proceedings of the Eurographics / IEEE VGTC Conference on Visualization: Short Papers*, EuroVis '16, pages 73–77, 2016.
- [58] P.E. Rauber, A.X. Falcão, and A.C. Telea. Projections as visual aids for classification system design. *Information Visualization*, 2017.
- [59] X. Ren and J. Malik. Learning a classification model for segmentation. In *Proceedings Ninth IEEE International Conference on Computer Vision*, pages 10–17 vol.1, Oct 2003.
- [60] P. K. Rhee, E. Erdene, S. D. Kyun, M. U. Ahmed, and S. Jin. Active and semi-supervised learning for object detection with imperfect data. *Cognitive Systems Research*, 45:109 – 123, 2017.
- [61] P. T. M. Saito, W. P. Amorim, A. X. Falcão, P. J. d. Rezende, C. T. N. Suzuki, J. F. Gomes, and M. H. d. Carvalho. Active semi-supervised learning using optimum-path forest. In *2014 22nd International Conference on Pattern Recognition*, pages 3798–3803, Aug 2014.
- [62] F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, March 2002.
- [63] B. Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.

- [64] V. Sindhwani, P. Niyogi, and M. Belkin. Beyond the point cloud: From transductive to semi-supervised learning. In *Proceedings of the 22Nd International Conference on Machine Learning*, ICML '05, pages 824–831. ACM, 2005.
- [65] N. C. Smeeton. Early history of the kappa statistic. *Biometrics*, 41:795–795, 1985.
- [66] R. Socher, J. Pennington, E. H. Huang, A. Y. Ng, and C. D. Manning. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 151–161. Association for Computational Linguistics, 2011.
- [67] T. V. Spina, P. A. V. De Miranda, and A. X. Falcão. Intelligent understanding of user interaction in image segmentation. *International Journal of Pattern Recognition and Artificial Intelligence*, 26(02):1265001, 2012.
- [68] J. T. Springenberg. Unsupervised and semi-supervised learning with categorical generative adversarial networks, 2015.
- [69] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, January 2014.
- [70] C. T. N. Suzuki, J. F. Gomes, A. X. Falcão, S. H. Shimizu, and J. P. Papa. Automated diagnosis of human intestinal parasites using optical microscopy images. In *2013 IEEE 10th International Symposium on Biomedical Imaging*, pages 460–463, April 2013.
- [71] J. Tang, R. Hong, S. Yan, T. S. Chua, G. J. Qi, and R. Jain. Image annotation by knn-sparse graph-based label propagation over noisily tagged web images. *ACM Transactions on Intelligent Systems and Technology*, 2(2):14:1–14:15, February 2011.
- [72] A. C. Telea. *Data Visualization: Principles and Practice, Second Edition*. A. K. Peters, Ltd., 2nd edition, 2014.
- [73] J. B. Tenenbaum, V. Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.
- [74] W. S. Torgerson. Multidimensional scaling: I. theory and method. *Psychometrika*, 17(4):401–419, Dec 1952.
- [75] L. van Der Maaten. Accelerating t-SNE using tree-based algorithms. *Journal of Machine Learning Research*, 15(1):3221–3245, 2014.
- [76] L. van Der Maaten and G. Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008.
- [77] L. van Der Maaten, E. Postma, and J. van den Herik. Dimensionality reduction: a comparative review. *Journal of Machine Learning Research*, 10:66–71, 2009.

- [78] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P. A. Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, 11:3371–3408, December 2010.
- [79] F. Wang and C. Zhang. Label propagation through linear neighborhoods. *IEEE Transactions on Knowledge and Data Engineering*, 20(1):55–67, Jan 2008.
- [80] K. Wang, D. Zhang, Y. Li, R. Zhang, and L. Lin. Cost-effective active learning for deep image classification. *IEEE Transactions on Circuits and Systems for Video Technology*, 27(12):2591–2600, Dec 2017.
- [81] Y. Wang, H. Yao, and S. Zhao. Auto-encoder based dimensionality reduction. *Neurocomputing*, 184(C):232–242, April 2016.
- [82] M. Wattenberg, F. Viegasand, and I. Johnson. How to use t-SNE effectively. *Distill*, 2016.
- [83] H. Ye, W. Shao, H. Wang, J. Ma, L. Wang, Y. Zheng, and X. Xue. Face recognition via active annotation and learning. In *Proceedings of the 24th ACM International Conference on Multimedia*, MM ’16, pages 1058–1062. ACM, 2016.
- [84] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural networks? In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, Neural Information Processing Systems (NIPS), pages 3320–3328. MIT Press, 2014.
- [85] H. Yu and S. Kim. Svm tutorial: Classification, regression, and ranking. *Handbook of Natural Computing*, 01 2012.
- [86] M. D. Zeiler and R. Fergus. *Visualizing and Understanding Convolutional Networks*, pages 818–833. Springer International Publishing, 2014.
- [87] M. Zhao, T. W.S. Chow, Z. Zhang, and B. Li. Automatic image annotation via compact graph based semi-supervised learning. *Knowledge-Based Systems*, 76:148 – 165, 2015.
- [88] S. Zhou, Q. Chen, and X. Wang. Active deep networks for semi-supervised sentiment classification. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, COLING ’10, pages 1515–1523. Association for Computational Linguistics, 2010.
- [89] X. Zhu, J. Lafferty, and Z. Ghahramani. Combining active learning and semi-supervised learning using gaussian fields and harmonic functions. In *International Conference on Machine Learning (ICML) Workshop on The Continuum from Labeled to Unlabeled Data in Machine Learning and Data Mining*, pages 58–65, 2003.