Uma abordagem visual para apoio ao aprendizado multi-instâncias

Sonia Castelo Quispe

Sonia Castelo Quispe

Uma abordagem visual para apoio ao aprendizado multi-instâncias

Dissertação apresentada ao Instituto de Ciências Matemáticas e de Computação - ICMC-USP, como parte dos requisitos para obtenção do título de Mestra em Ciências - Ciências de Computação e Matemática Computacional. *VERSÃO REVISADA*

Área de Concentração: Ciências de Computação e Matemática Computacional

Orientadora: Profa. Dra. Rosane Minghim

USP – São Carlos
Outubro de 2015

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados fornecidos pelo(a) autor(a)

Castelo, Sonia
C348a Uma abordagem visual para apoio ao aprendizado multi-instâncias / Sonia Castelo; orientadora Rosane Minghim. -- São Carlos, 2015.
94 p.

Dissertação (Mestrado - Programa de Pós-Graduação em Ciências de Computação e Matemática Computacional) -- Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, 2015.

1. Classificação visual de dados. 2. Aprendizado múltipla instância. 3. Árvore de similaridade. 4. Mineração de dados. I. Minghim, Rosane, orient. II. Título.

Sonia Castelo Quispe

A visual approach for support to multi-instances learning

Master dissertation submitted to the Instituto de Ciências Matemáticas e de Computação - ICMC-USP, in partial fulfillment of the requirements for the degree of the Master Program in Computer Science and Computational Mathematics. *FINAL VERSION*

Concentration Area: Computer Science and Computational Mathematics

Advisor: Profa. Dra. Rosane Minghim

USP – São Carlos
October 2015

Aos meus pais, Simón e Gabina.

Agradecimentos

Em primeiro lugar quero agradecer a Deus, pela oportunidade de chegar até aqui, que nunca me deixo sozinha e me deu a força para realizar este trabalho.

Agradeço aos meus amados pais, Gabina e Simón, pelas orações, amor, apoio, palavras de encorajamento e por sempre incentivar-me a trabalhar por lograr meus sonhos. Aos meus irmãos: Ever, Wilber e Lidia, pelas alegrias. Obrigada Ever pelas divertidas conversações. Obrigada Lidia pela ajuda e por sempre cuidar de mim, sempre tiveste palavras sinceras e de alento pra mim. Obrigada Wilber pelo incentivo aos estudos, foi o melhor que me ensinaste. Obrigada Mamagrande porque sei que desde o céu me acompanha e me cuida, sinto saudade de você. Agradeço também a Roque Enrique, meu namorado, pelo amor e compreensão, que soube me apoiar nos momentos de tristeza e fazer os momentos de alegria mais longos. Te amo.

Agradeço à minha orientadora, Profa. Dra. Rosane Minghim pela excelente orientação fornecida, pela paciência, conselhos e apoio na realização deste projeto. Espero seguir seu exemplo de dedicação, trabalho e comprometimento. Saiba que eu nunca vou esquecer tudo o que você fez por mim. Muito obrigada!

Ao Prof. Dr. Moacir Ponti pela colaboração, nunca se negando a me ajudar, pelos conselhos e amizade. Sua ajuda foi fundamental para concluir meu trabalho. Obrigada professor Moacir!

Ao pesquisador Renato Rodrigues, pelos conselhos, paciência, sempre disposto a ajudar me desde o começo deste projeto até o final, mesmo longe. Muito obrigada Renato. Ao José Gustavo Paiva por disponibilizar o sistema VCS. Ao Leonardo pelo apoio no processamento das imagens.

Agradeço aos meus amigos do laboratório VICG pela sua amizade, colaboração direta e indireta para a conclusão deste projeto: Vinicius, Samuel, Danilo, Felipe, Karina, Fabinho, Chicão, Gabriela, Gabriel, Lucas, Evinton, Jose, Giancarlo. Aos demais muito obrigada também.

Agradeço aos meus queridos amigos Fábio, Jorge, Carlos e Gladys pela amizade, companheirismo, risadas e todos os bons momentos juntos. Eu vou me lembrar de vocês sempre!

Agradeço aos amigos que fiz aqui no ICMC. Especialmente a: Aurea, Oscar, Paola, Marco, Alessandro, Mayra, Jorge, Edwin, e aos demais amigos, agradeço a torcida que, de alguma forma, me ajudou a chegar até aqui.

Ao Instituto de Ciências Matemáticas e de Computação (ICMC-USP) pela oportunidade de realizar o curso de mestrado. Aos professores pelos ensinamentos e aos funcionarios pelos excelentes serviçoes prestados.

Ao conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) pelo apoio financeiro. Processo 134238/2013-3.

À Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP) pela concessão da bolsa de mestrado e pelo apoio financeiro para realização desta pesquisa. Processo 2013/25055-2.

Resumo

Aprendizado múltipla instância (MIL) é um paradigma de aprendizado de máquina que tem o objetivo de classificar um conjunto (*bags*) de objetos (*instâncias*), atribuindo rótulos só para os *bags*. Em MIL apenas os rótulos dos *bags* estão disponíveis para treinamento, enquanto os rótulos das instâncias são desconhecidos. Este problema é frequentemente abordado através da seleção de uma instância para representar cada *bag*, transformando um problema MIL em um problema de aprendizado supervisionado padrão. No entanto, não se conhecem aborgagens que apoiem o usuário na realização desse processo. Neste trabalho, propomos uma visualização baseada em árvore multi-escala chamada MILTree que ajuda os usuários na realização de tarefas relacionadas com MIL, e também dois novos métodos de seleção de instâncias, chamados MILTree-SI e MILTree-Med, para melhorar os modelos MIL. MILTree é um layout de árvore de dois níveis, sendo que o primeiro projeta os *bags*, e o segundo nível projeta as instâncias pertencentes a cada *bag*, permitindo que o usuário explore e analise os dados multi-instância de uma forma intuitiva. Já os métodos de seleção de instâncias objetivam definir uma instância protótipo para cada *bag*, etapa crucial para a obtenção de uma alta precisão na classificação de dados multi-instância. Ambos os métodos utilizam o layout MILTree para atualizar visualmente as instâncias protótipo, e são capazes de lidar com conjuntos de dados binários e multi-classe. Para realizar a classificação dos *bags*, usamos um classificador SVM (Support Vector Machine). Além disso, com o apoio do layout MILTree também pode-se atualizar os modelos de classificação, alterando o conjunto de treinamento, a fim de obter uma melhor classificação. Os resultados experimentais validam a eficácia da nossa abordagem, mostrando que a mineração visual através da MILTree pode ajudar os usuários em cenários de classificação multi-instância.

Palavras-chave: Classificação visual de dados; Aprendizado múltipla instância; Árvore de similaridade; Mineração de dados.

Abstract

Multiple-instance learning (MIL) is a paradigm of machine learning that aims at classifying a set (*bags*) of objects (*instances*), assigning labels only to the *bags*. In MIL, only the labels of *bags* are available for training while the labels of instances in *bags* are unknown. This problem is often addressed by selecting an instance to represent each *bag*, transforming a MIL problem into a standard supervised learning. However, there is no user support to assess this process. In this work, we propose a multi-scale tree-based visualization called MILTree that supports users in tasks related to MIL, and also two new instance selection methods called MILTree-SI and MILTree-Med to improve MIL models. MILTree is a two-level tree layout, where the first level projects *bags*, and the second level projects the instances belonging to each *bag*, allowing the user to understand the data multi-instance in an intuitive way. The developed selection methods define instance prototypes of each *bag*, which is important to achieve high accuracy in multi-instance classification. Both methods use the MILTree layout to visually update instance prototypes and can handle binary and multiple-class datasets. In order to classify the *bags* we use a SVM classifier. Moreover, with support of MILTree layout one can also update the classification model by changing the training set in order to obtain a better classifier. Experimental results validate the effectiveness of our approach, showing that visual mining by MILTree can help the users in MIL classification scenarios.

Keywords: Visual data classification; Multiple instance learning; Similarity tree; Data mining.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Siglas

ML	*Machine Learning*
MIL	*Multiple Instance Learning*
CBIR	*Content-Based Image Retrieval*
PCA	*Principal Component Analysis*
MDS	*Multidimensional Scaling*
FDP	*Force-Directed Placement*
LSP	*Least Square Projection*
PLP	*Piecewise Laplacian-based Projection*
LAMP	*Local Affine Multidimensional Projection*
NJ	*Neighbor-Joining*
MCMIL	*Multiple Class Multiple Instance Learning*
APR	*Axis Parallel Rectangle*
DD	*Diverse Density*
SVM	*Support Vector Machine*
CCE	*Constructive Clustering-based Ensemble*
DD-SVM	*Diverse Density Support Vector Machine*
MILES	*Multiple-Instance Learning via Embedded Instance Selection*
MILIS	*Multiple Instance Learning with Instance Selection*
EM-DD	*Expectation Maximization - Diversity Density*
MI-SVM	*Support Vector Machines for Multiple-Instance Learning*
RW-SVM	*Random Walk with SVM*
SMILES	*Similarity-Based Multiple-Instance Learning*
RGB	*Color Spaces RGB*
MILSIS	*Salient Instance Selection for Multiple Instances Learning*
IDMAP	*Interactive Document Map*
MILD-B	*Multiple-Instance Learning via Disambiguation*
MILDE	*Multiple Instance Learning by Discriminative Embedding*
PLS	*Partial Least Squares*
NH	*Neighborhood Hit*
CS	*Coeficiente de Silhueta*
VCS	*Visual Classification System*

Capítulo

1

Introdução

1.1 Contextualização e Motivação

Muitos problemas de aprendizado de máquina podem ser resolvidos usando técnicas de aprendizado supervisionado, onde um objeto é representado por um único vetor de características o qual chamamos instância (Faceli et al., 2011). No entanto, existem problemas que não se encaixam nessa definição. Em alguns casos, é conveniente representar um objeto por uma composição de várias instâncias, sendo cada uma delas representada por um vetor de características. Esta é uma forma mais flexível de representar um objeto e os algoritmos de aprendizado que trabalham com esta definição são chamados coletivamente de *Multiple instance learning* (MIL) (Fu et al., 2011). MIL, ou Aprendizado Multi-Instâncias, é considerado uma variação do aprendizado supervisionado. Em MIL, diferente do aprendizado supervisionado tradicional, as instâncias estão organizadas em pacotes (*bags*) que são rotuladas para o treinamento, diferente das instâncias individuais, para as quais o rótulo é comumente desconhecido. Podemos chamar de '*bags*' a representação conjunta de múltiplas instâncias.

MIL foi introduzido por Dietterich et al. (1997), a fim de resolver o problema de predição de reações químicas de drogas, mas também já foi utilizada em uma ampla variedade de outras aplicações no mundo real, tais como a classificação de imagens (Amores, 2015), categorização de texto (Ray e Craven, 2005a), reconhecimento de voz (Reynolds et al.,

2000), mineração web (Zafra et al., 2008), entre outros. Um cenário prático do MIL é em *Content-Based Image Retrieval*(CBIR), em que a imagem representa um *bag* e as regiões segmentadas da imagem representam as instâncias pertencentes a esse bag. Um exemplo de aplicação do MIL para a classificação de imagens de câncer de pulmão é apresentada por Zhu et al. (2008).

Para lidar com estratégias MIL, foram propostos diferentes métodos supervisionados (Andrews et al., 2003), (Fu e Robles-Kelly, 2009), (Shen et al., 2009), (Xiao et al., 2014). Entre eles, a estratégia mais utilizada é a de converter o problema MIL em um problema de aprendizado supervisionado tradicional através da seleção de uma instância protótipo, a qual será a instância mais representativa entre todas as instâncias pertencentes a um *bag*, assumindo que a instância selecionada é suficiente para representar corretamente o *bag* (Chen et al., 2006), (Fu et al., 2011), (Yuan et al., 2012). Assim, essa instância protótipo pode depois ser utilizada para treinar um classificador padrão. Por exemplo, na tarefa de classificação de imagens onde uma imagem é considerada um *bag* e suas regiões como instâncias, apenas uma região (instância) seria escolhida para representar a imagem. Nesse contexto, a seleção da instância protótipo de cada *bag* é uma tarefa crucial em cenários MIL, refletindo diretamente na qualidade dos resultados da classificação. Porém, os resultados finais de uma classificação dependem também de outros fatores como medidas de similaridade e um adequado conjunto de treinamento (Foody e Mathur, 2006). Neste trabalho propomos, entre outros resultados, uma estratégia que permita a interferência do usuário no processo de MIL, através da visualização dos *bags* e instâncias utilizando técnicas de visualização de informação, os quais se sabe que podem garantir uma experiencia de exploração e análise satisfatória por parte do usuário com o intuito de melhorar o processo completo de classificação de dados (Paiva et al., 2015).

Nesse contexto, o objetivo das técnicas de visualização de informação é a criação de mapeamentos visuais e estratégias interativas para compreensão e aquisição de conhecimentos a partir dos dados em um contexto complexo, reduzindo o trabalho cognitivo para a realização dessas tarefas. Algumas ferramentas de visualização multidimensional[1] foram desenvolvidas para tentar superar os desafios de dimensionalidade e heterogeneidade dos dados.

Atualmente, existem aplicativos e sistemas que suportam a exploração visual de coleções de dados, como imagens (Eler et al., 2008a; Yang et al., 2006; Choo et al., 2010; Chavarro et al., 2013; Eler et al., 2009), textos (Eler et al., 2008b; Dou et al., 2013; Zhao

[1]O termo "dimensões" se refere às variáveis ou atributos de dados. Assim, dados com mais de três variáveis são chamados dados multidimensionais. Portanto, uma visualização multidimensional denota a visualização que é capaz de exibir dados multidimensionais (Hoffman e Grinstein, 2002; Soukup e Davidson, 2002).

et al., 2013), dados musicais (Chan et al., 2010; Dalhuijsen e Velthoven, 2010; Torrens e Arcos, 2004), e redes sociais (Shen et al., 2006; Martins et al., 2012; Xu et al., 2013; Ghani et al., 2013), dentre outros, os quais, em sua grande maioria, estão baseados em técnicas de visualização multidimensional. Nestes aplicativos e sistemas os usuários têm acesso a uma interface visual em que podem interagir e reconhecer similaridades, fazer seleções, recuperar informações e desenvolver outras atividades.

As técnicas de visualização também foram já utilizadas com sucesso para ajudar aos usuários nas tarefas de classificação padrão, em que objetos são representados por simples instâncias, como mostrado em (Keim e Kriegel, 1996), (Xu et al., 2007), (Zhang et al., 2012b) e (Paiva et al., 2015). No entanto, não se tem conhecimento da existência de trabalhos que apoiem o aprendizado multi-instâncias a partir de uma perspectiva visual, que permita a inserção do usuário na exploração e análise dos dados multi-instâncias, aproveitando o conhecimento do usuário para criar melhores modelos de classificação, assim como para detectar padrões, tendências ou características particulares dos *bag* que não se pode reconhecer se a análise dos dados for feita instância por instância.

Um dos desafios para o uso de técnicas de visualização em MIL, é a grande quantidade de dados que frequentemente têm que ser projetado na tela (Fu e Robles-Kelly, 2009). Como os objetos em MIL são representados por mais de uma instância, ao tentar visualizar todas as instâncias do conjunto de dados multi-instância num único espaço de projeção, aparecerão problemas como a sobreposição de pontos. Por exemplo, o conjunto de dados Musk1 contém 6598 instâncias, as quais são agrupadas em 102 *bags*, sendo o número de instância contidas em cada *bag* entre 1 a 1044. Além disso, a visualização dos dados completos num único layout não refletiria de forma natural a estrutura de um dado multi-instância, o que é primordial para a seleção da instância protótipo para cada *bag*.

A visualização multi-escala (Chen, 2006) é uma abordagem visual que se mostra promissória tanto para reduzir o número de dados a ser projetado na tela, assim como para representar dados que, como no caso dos conjuntos de instâncias em MIL, são adequadamente representados como grupos.

Neste contexto, o presente projeto visa desenvolver uma nova abordagem visual para a exploração, análise e classificação de dados multi-instância, através de uma nova forma de visualização baseada em árvore chamada MILTree. MILTree é uma visualização multi-escala baseada na árvore de *Neighbor-Joining* (NJ) gerada no trabalho de Cuadros et al. (2007) e melhorada no trabalho de Paiva et al. (2011), o qual otimiza a busca de nós, além de diminuir o custo computacional. MILTree é uma árvore de dois níveis. No primeiro nível, apenas *bags* são projetadas, reduzindo a quantidade de dados a visualizar. O segundo nível projeta as instâncias pertencentes a um determinado *bag*, permitindo que o usuário

faça uma exploração e análise de cada *bag* na procura da instância protótipo. Cada *bag* no primeiro nível está ligada à suas instâncias num segundo nível, mostrando de forma intuitiva a estrutura de um conjunto de dados multi-instância (*bags* e instâncias).

Além da visualização baseada em árvores NJ para apoiar MIL, este projeto propõe dois novos métodos de seleção da instância protótipo: MILTree-SI e MILTree-Med, que enriquecem o processo de classificação de dados multi-instâncias e permitem o mapeamento dos *bags* no MILTree. MILTree-SI é baseado no método MILSIS (Yuan et al., 2012) que assume que os *bags* negativos só contém instâncias negativas, enquanto o nosso método considera que os *bags* negativos podem conter também instâncias positivas, o qual se reflete mais nos dados multi-instância do mundo real. Assim, MILTree-SI utiliza tanto as informações das instâncias protótipo em *bags* positivos como negativos no momento da construção de modelos de classificação. O método MILTree-Med utiliza o algoritmo de agrupamento k-Medoids (Shen et al., 2009) no espaço das instâncias dentro de um *bag* para agrupar essas instâncias em dois subclusters, com uma tentativa de encontrar um subcluster positivo e um subcluster negativo. Para *bags* positivos o medoid do subcluster positivo é considerado a instância protótipo e para *bags* negativos o medoid do subcluster negativo é considerado como a instância protótipo. Os dois métodos fazem uso do layout MILTree para explorar os *bags* e reafirmar ou atualizar a instância protótipo de cada *bag*.

A maioria dos estudos MIL incluem apenas classificação binária, em que um *bag* pode pertencer a uma classe positiva ou negativa: um *bag* é rotulado como positivo se contiver, pelo menos, uma instância positiva, de outro modo é rotulado como um *bag* negativo. Este trabalho aborda problemas binários e multiclasse. Neste trabalho também se adapta a metodologia proposta por Paiva et al. (2015), de apoio visual à classificação de dados, para trabalhar com dados multi-instancia, permitindo aos usuários participar de todas as etapas do processo de classificação de dados multi-instâncias com o intuito de criar modelos de classificação mais robustos.

1.2 Objetivo

O objetivo principal deste trabalho é o desenvolvimento de uma abordagem visual para contribuir no processo de classificação visual de dados multi-instancias, usando uma técnica de visualização multi-escala e métodos de seleção de instâncias protótipo, a fim de permitir que um usuário explore e analise os conjuntos de dados multi-instancia buscando criar modelos de classificação robustos apartir das instâncias protótipo adequadas.

Os objetivos específicos serão apresentados a seguir:

- Utilizando a técnica de visualização multi-escala baseado em árvores de similaridade *Neighbor Joining*, permitir ao usuário explorar e analisar os conjuntos de dados multi-instâncias buscando escolher os conjuntos de treinamento relevantes para a criação de modelos de classificação multi-instâncias.

- Establecer métodos de seleção de instâncias protótipo dentro de cada *bag*, buscando transformar o problema MIL em um problema de aprendizado supervisionado tradicional. Além de, permitir ao usuário explorar, validar ou atualizar visualmente essas instâncias protótipo.

- Adaptar a metodologia de classificação visual de dados aprensentado em Paiva et al. (2015) para conseguir trabalhar com dados multi-instância, permitindo aos usuários participar do processo completo de classificação multi-instância, possibilitando ao usuário a seleção do conjunto te treinamento, criação dos modelos, avaliação dos resultados da classificação e a atualização desses modelos que melhorem os resultados finais na classificação de dados multi-instâncias.

- Entender a ferramenta VCS (iniciada por Paiva (2013)) e incluir as propostas aqui definidas e tornando-as de uso livre.

1.3 Organização da Monografia

O restante desta monografia está estruturada da seguinte maneira:

- No Capítulo 2 são abordados trabalhos relacionados ao contexto desta pesquisa. São introduzidas algumas técnicas de Visualização de Informação, base para o início dos estudos. Em seguida é abordado o aprendizado multi-instância, com seus principais algoritmos e aplicações. Por fim são apresentadas pesquisas relacionadas à classificação visual de dados.

- No Capítulo 3 é apresentada a metodologia desta pesquisa.

- No Capítulo 4 são apresentados os casos de estudo com o objetivo de mostrar a aplicação da nova visualização de dados multi-instância MILTree no contexto de aprendizado multi-instância.

- No Capítulo 5 são apresentados todos os experimentos realizados para medir a qualidade da proposta.

- No Capítulo 6 são apresentadas as conclusões.

Capítulo

2

Revisão Bibliográfica

2.1 Considerações Iniciais

Este capítulo apresenta trabalhos relacionados às várias sub-áreas deste projeto. Inicialmente, a Seção 2.2 apresenta as técnicas de visualização de informação, com principal foco nas técnicas de visualização baseada em árvore de similaridade, as quais tem o intuito de produzir mapeamentos que reflitam a estrutura dos dados e permitam uma exploração satisfatória, além de apoiar o processo de classificação visual de dados.

A Seção 2.3 apresenta um estudo sobre MIL, seus principais algoritmos e sua aplicação no processo de classificação de dados multi-instância. Finalmente, a Seção 2.4 apresenta conceitos relacionados à classificação visual de dados. A seguir são detalhadas as principais abordagens existentes que são de interesse deste projeto de mestrado.

2.2 Visualização de Informação

Existem diversos trabalhos na literatura que descrevem métodos para realizar classificação e exploração visual de grandes coleções de dados.

Apesar disso, a eficácia para uma boa análise, classificação e demais atividades sobre as coleções de dados depende muito da forma de representação desses dados, tornando a tarefa de representação de dados um desafio importante. A representação dos dados deve

conter informações relevantes, que possibilitem diferenciar os dados, permitindo que eles sejam facilmente agrupáveis.

Outra tarefa ainda mais importante para a classificação e exploração de dados multidimensionais é encontrar representações visuais simples, que mostrem o verdadeiro comportamento e as relações entre os dados. Para isto, se empregam técnicas de visualização de informação, considerando a limitação da dimensionalidade na apresentação em diferentes dispositivos. Essas técnicas podem ser agrupadas em (Cuadros et al., 2007): as projeções multidimensionais e as estruturas em árvore, e são descritas a seguir.

2.2.1 Técnicas de Projeção Multidimensional

As técnicas de projeção multidimensional tem por objetivo reduzir o número de dimensões dos dados, mapeando os indivíduos em um espaço de baixa dimensionalidade, como 2D ou 3D (Tejada et al., 2003).

O resultado da projeção é uma representação visual do conjunto de dados multidimensionais. Nesta representação, cada instância de dados (vetor de características) é mapeada para um elemento visual, como um círculo, ponto, ou esfera. A posição relativa destas representações pode refletir algum tipo de relação entre instâncias de dados, sendo a relação mais frequente a similaridade ou vizinhança dada pela proximidade no espaço de visualização (Tejada et al., 2003; Paulovich e Minghim, 2008).

As técnicas de projeção multidimensional podem ser divididas em dois grupos de acordo com as funções empregadas: técnicas de projeção lineares e não lineares (Cuadros et al., 2007; Paulovich et al., 2007).

Principal Component Analysis (PCA) (Jolliffe, 2002) é uma técnica linear que transforma um conjunto de variáveis correlacionadas, cujos valores possuem alta variância, em um conjunto menor de variáveis não correlacionadas chamadas de "componentes principais", com o objetivo de reter tanto quanto possível a variação no conjunto de dados. Quando a métrica de distância no espaço multidimensional é Euclidiana, o resultado alcançado pelo PCA é o mesmo da técnica *Classical Scaling* (Torgerson, 1965). Assim, PCA também pode ser utilizada como uma técnica de projeção multidimensional que permite extrair padrões que realcem as similaridades e diferenças entre as instâncias de um conjunto de dados. O maior problema com PCA, no entanto, é a geração de *layouts* de baixa qualidade no que se refere à segregação de dados. Em geral, as técnicas lineares não conseguem capturar padrões relevantes nos dados que apresentam estruturas não lineares, como por exemplo quando os dados são agrupados de formas arbitrárias.

Por outro lado, as técnicas não lineares conseguem lidar com estruturas não lineares, considerando um processo de otimização tentando minimizar a perda de informações ocor-

rida na projeção (Paulovich, 2008). A técnica de projeção não linear *Multidimensional Scaling* (MDS) (Cox e Cox, 2001), compreende uma família de técnicas baseados em métodos matemáticos para realizar projeções. Dentre as técnicas que seguem o modelo MDS destaca-se a *Force-Directed Placement* (FDP) (Fruchterman e Reingold, 1991) que geralmente tem alto grau de precisão, mas ainda tem um alto custo computacional ($O(n^3)$), com n sendo a quantidade de instâncias.

Para reduzir a complexidade dos algoritmos, no trabalho de Paulovich (2008) é proposta a *Least Square Projection* (LSP) (Paulovich et al., 2008). A LSP combina os benefícios de técnicas lineares e não lineares, com o objetivo de criar uma superfície onde os dados estejam agrupados por relações de proximidade. LSP apresenta uma complexidade computacional de $O(n\sqrt{n})$, com n igual ao número de instâncias, e é rápida e precisa sobre dados complexos e tamanho moderado. No entanto, a LSP requer uma grande quantidade de memória e possui um alto custo computacional para coleções de dados muito grandes.

A fim de acelerar o processo da LSP, uma nova técnica baseada nos seus mesmos princípios foi desenvolvida, denominada *Piecewise Laplacian-based Projection* (PLP) (Paulovich et al., 2011). Nessa técnica, o conjunto de dados é dividido em grupos com aproximadamente o mesmo tamanho e cada um desses grupos é projetado individualmente usando a LSP. A técnica permite a seleção de amostras chamadas de "pontos de controle", o que mantem a coerência espacial entre os grupos. Uma grande vantagem da PLP é que permite a manipulação das amostras da coleção de uma maneira visual. Quando alguma modificação é feita pelo usuário na posição dessas amostras no *layout*, seus grafos de vizinhança e pontos de controle associados são dinamicamente atualizados, produzindo um melhor *layout*. A Figura 2.1 apresenta a Projeção PLP de uma coleção de imagens, antes e depois de uma sequência de manipulações realizadas pelo usuário. Na figura, a cor da borda nas imagens representa a classe de cada imagem, e a janela na parte superior direita representa os pontos de controle.

Outra técnica que também se baseia na utilização de "pontos de controle" é a técnica *Local Affine Multidimensional Projection* (LAMP) (Joia et al., 2011). LAMP está baseada na teoria de mapeamento ortogonal afim que projeta os dados do espaço multidimensional para o espaço visual, sendo que o mapeamento pode ser modificado de acordo com interações por parte do usuário sobre os pontos de controle. A técnica é muito útil nos cenários onde é preciso levar em consideração o conhecimento do usuário sobre o conjunto de dados. O diferencial da LAMP é permitir o uso de um conjunto pequeno de amostras. A Figura 2.2 apresenta a Projeção LAMP usando só 3 pontos de controle por classe de uma coleção de 7 classes.

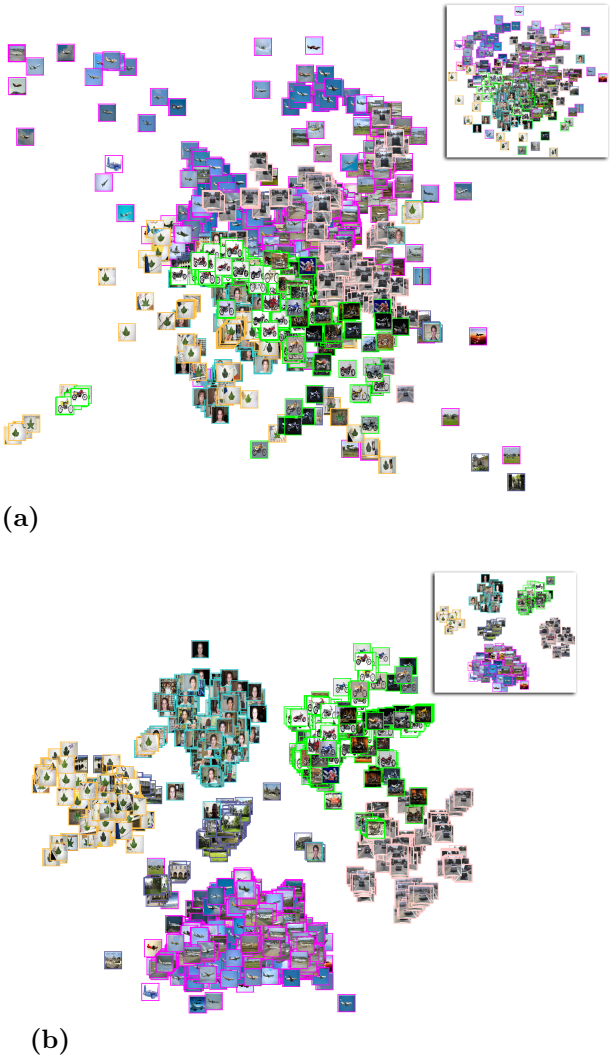
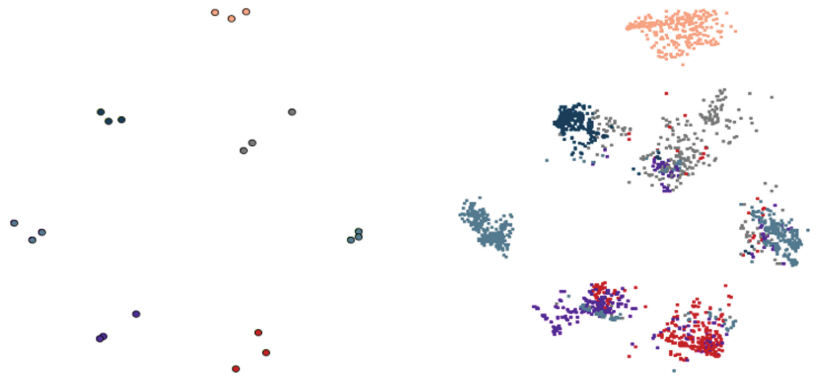


Figura 2.1: Projeção PLP de uma coleção de imagens, antes e depois de uma sequência de manipulações realizadas pelo usuário. (**a**) Projeção Inicial da Coleção de imagens com pontos de controle posicionados usando o esquema baseado em força; (**b**) Projeção depois do reposicionamento de pontos de controle, o que resulta em uma boa separação de grupos das imagens. Retirado de (Paulovich et al., 2011).

A principal desvantagem nas técnicas de projeção multidimensional continua sendo a baixa qualidade dos *layouts* resultantes em algumas aplicações no que se refere à precisão. Além disso, podem apresentar alto grau de sobreposição de instâncias e confusão visual, devido à baixa dimensão (2D) para expressar características importantes em conjuntos de dados heterogêneos (Cuadros et al., 2007).



(a) (b)

Figura 2.2: Projeção LAMP com 3 pontos de controle por classe. A coleção tem 7 classes (**a**) e os 3 pontos de controle por classe foram escolhidos aleatoriamente na coleção; (**b**) Mapeamento produzido pela LAMP utilizando o conjunto de pontos de controle de (a). Retirado de (Joia et al., 2011).

2.2.2 Técnicas de Visualização Baseada em Árvore de Similaridade

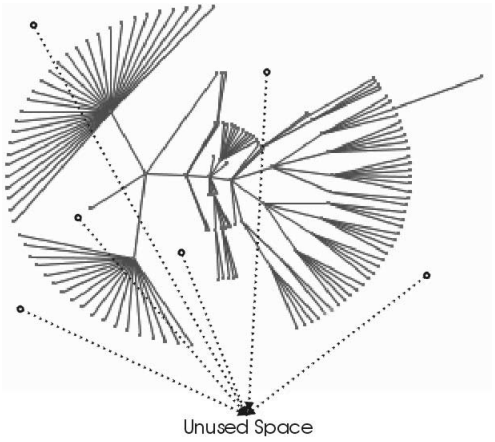
Outra alternativa para representar dados multidimensionais é a utilização de árvores. Uma árvore apresenta de forma natural uma estrutura hierárquica, sobre algum relacionamento significativo dos dados, que organiza as entidades de dados por meio de conexões pai-filhos. Esta hierarquia pode expressar, por exemplo, similaridade, isto é quanto mais similares sejam as instâncias, mais próximos estariam posicionados nos ramos da árvore.

Quando se usa uma árvore como estrutura visual, é muito importante o método de posicionamento dos nós. Este método pode ser adaptado para produzir um *layout* que cobre todo espaço visível.

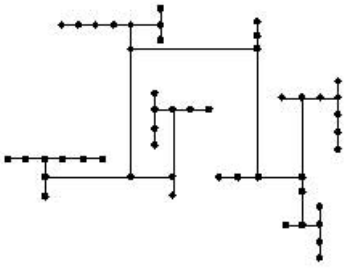
No trabalho de Nguyen e Huang (2002) são apresentadas diferentes técnicas para *layout* de árvores como *Árvore-H*, *Visão Radial*, *Visão em Balão* e *Mapa em Árvore*. O Mapa em Árvore *(Tree-Maps)*, se destaca porque trabalha muito bem na representação das variáveis que podem ser decompostas hierarquicamente, aproveitando todo o espaço visual disponível. Porém, apresentação de uma *Tree-Map* começa a ter deficiência quando o volume dos dados cresce, tornando a visualização densa e confusa. A Figura 2.3 mostra representações gráficas alternativas das técnicas para *layout* de árvores.

Árvore Neighbor-Joining

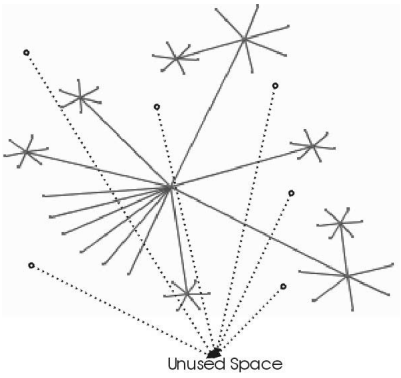
Em termos de análise de dados, o método para geração da hierarquia é tão importante quanto o método de *layouts* utilizado para apresentar a árvore.



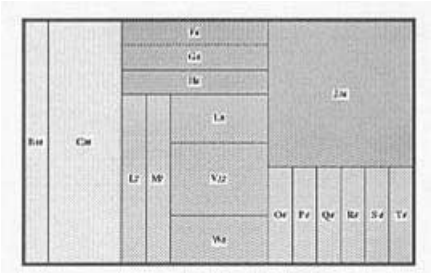
(a)



(b)



(c)



(d)

Figura 2.3: Técnicas para apresentar árvores de similaridade. (**a**) Visão Radial, conseguimos ver que a técnica desperdiça grande parte do espaço de exibição ; (**b**) Árvore-H, uma técnica de desenho clássico para representar árvores binárias; (**c**) Visão em Balão, similar à Visão Radial; (**d**) TreeMap, cada nó é mapeado para uma área retangular, depois essa área é subdividida nos eixos x ou y, para mostrar o tamanho relativo dos filhos do nó. O processo é recursivamente aplicado aos nós filhos com as subdivisões no eixo x ou y. Retirado de (Nguyen e Huang, 2002).

A árvore de similaridade *Neighbor-Joining* (NJ) é um método para geração da hierarquia e é gerada a partir de um método de reconstrução de filogenias (Saitou e Nei, 1987), que possui capacidade de segregação de grupos. As árvores de similaridade podem

ser construídas a partir das relações de distâncias, que são geralmente representadas por matrizes de distâncias.

O método NJ produz uma árvore sem raiz e tem como objetivo minimizar o comprimento e o número de ramos da árvore gerada, encontrando pares de instâncias mais próximas entre si e mais distantes das restantes: implicitamente encontrando um par de instâncias vizinhas.

A técnica NJ começa com uma árvore estrela. Essa árvore é formada por todos os n objetos da matriz de distâncias, representados por nós folha, dispostos em uma configuração circular e conectados por ramos a um único nó central. Em seguida, iterativamente encontra um par de vizinhos mais próximos entre todos os possíveis pares de nós pelo critério de evolução mínima[1]. Depois o par de nós mais próximos é agrupado em um novo nó interno, e as distâncias desse nó para o resto dos nós da árvore são computadas e usadas em iterações posteriores. O algoritmo termina quando $n - 2$ nós internos forem inseridos na árvore, ou seja, quando a árvore estrela é completamente resolvida em uma árvore binária. A Figura 2.4 ilustra o processo de construção da hierarquia final da árvore NJ.

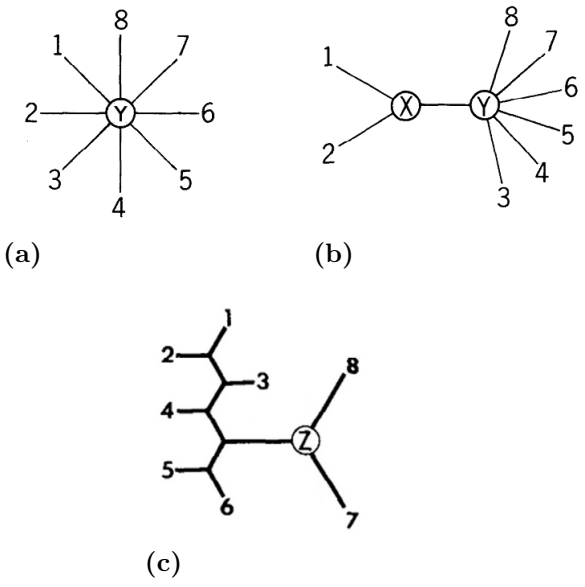


Figura 2.4: Processo de construção da hierarquia final da árvore utilizando o algoritmo *Neighbor Joining*. (**a**) Árvore estrela . (**b**) Par de nós mais próximos (1 e 2) agrupados em um novo nó interno. (**c**) Hierarquia final da árvore. Retirado de (Saitou e Nei, 1987).

A técnica NJ é empregada por Cuadros et al. (2007) para construir árvores de similaridade para visualizar coleções de documentos textuais. Como descreve Cuadros et

[1]O critério de evolução mínima tenta minimizar a soma dos tamanhos de todos os nós da árvore.

al. (2007) o algoritmo NJ, apresentado no Algoritmo 1, pode ser resumido nos seguintes elementos:

- Cálculo da medida de divergência. Como passo inicial se calcula para cada nó a medida de divergência r utilizando a Equação 2.1, onde D denota a matriz de distâncias, e, i e j denotam um par de nós qualquer.

$$r_{(i)} = \sum_{j \neq i} d_{i,j} \tag{2.1}$$

- Critério utilizado para selecionar pares de nós. Para isso, inicialmente é calculada a matriz de distâncias ajustadas M criada a partir de D e das divergências r, por meio da Equação 2.2, onde n é o número de objetos da matriz D. Em seguida, entre todos os possíveis pares de nós a serem unidos são escolhidos os nós i e j em D que tem o menor valor $m_{i,j}$ em M.

$$m_{i,j} = d_{i,j} - \frac{r_i + r_j}{n - 2} \tag{2.2}$$

- Cálculo do tamanho dos ramos. Utilizando as Equações 2.3 e 2.4, são calculadas as distâncias do novo nó para os demais, onde s representa os tamanhos dos novos ramos que ligam o objeto u aos objetos i e j. Nessa Equação u é o novo nó interno.

$$s_{i,u} = \frac{d_{i,j}}{2} + \frac{r_i + r_j}{2(n - 2)} \tag{2.3}$$

$$s_{j,u} = d_{i,j} - s_{i,u} \tag{2.4}$$

- Estratégia de Substituição. É feita a substituição dos objetos i e j na matriz de distâncias D pelo novo objeto u e o cálculo da distância entre o novo objeto u e os demais dado pela Equação 2.5, onde $k \neq i$, $k \neq j$ e $j = 1...n$. Conseguindo, desse modo, excluir i e j do processo.

$$d_{k,u} = \frac{d_{i,k} + d_{j,k} + d_{i,j}}{2} \tag{2.5}$$

- Finalmente, o número de objetos denotado por n é reduzido em 1 e o processo é repetido a partir do cálculo das divergências, encontrando e unindo em cada iteração os dois vizinhos mais próximos da matriz D até que $n-2$ nós internos foram inseridos na árvore.

Algoritmo 1: Algoritmo NJ

Entrada: Matriz de distâncias (D), número de objetos (n)
Saída: Árvore
para n *até* 3 **faça**
  R = calcularDivergências (D, n), com a Equação 2.1;
  Selecionar um par de nós (i, j) com o mínimo valor para r_{ij};
  Criar um novo nó u conectado aos nós i e j com a aresta de tamanho S_{iu} e S_{ju}, com as Equações 2.3 e 2.4;
  Calcular as distâncias entre o novo objeto u e os objetos restantes na matriz D, com a Equação 2.5;
  Ajustar a matriz atual;
  Conectar o nó u como pai dos nós i e j;
fim
retorna *Árvore com os nós e arestas*

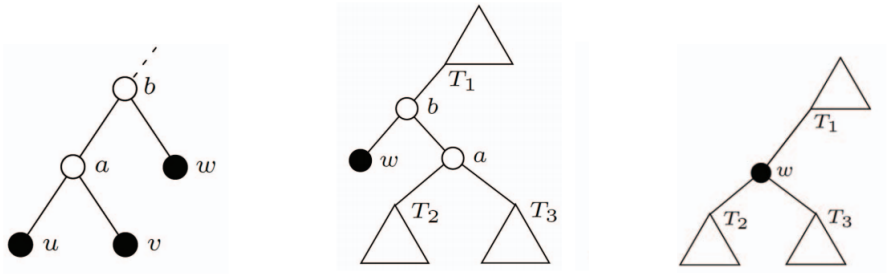
A árvore NJ gerada no trabalho de Cuadros et al. (2007) tem pouca profundidade, permitindo o uso racional do espaço de visualização, enquanto a sua útil interpretação hierárquica permite apreciar as relações de similaridade local e global. A principal vantagem da técnica NJ é que ela coloca as instâncias similares em uma vizinhança apropriada com um elevado grau de precisão, o que facilita a compreensão entre instâncias semelhantes utilizando hierarquias. A principal desvantagem, entretanto, é o seu elevado custo computacional que é $O(n^3)$.

O algoritmo NJ foi modificado e melhorado por vários trabalhos considerando-se vários critérios. O trabalho de Paiva et al. (2011) otimiza a busca de pares de nós, aproximando a complexidade do algoritmo para $O(n^2)$. Para este trabalho utiliza-se esta modificação do algoritmo *NJ*.

Além de diminuir o custo computacional, o trabalho de Paiva et al. (2011) aborda o problema de nós intermediários desenvolvendo uma estratégia de promoção de nós. Essa estratégia consiste na substituição de um nó interno com uma folha sempre que o padrão mostrado na Figura 2.5 ocorre. O padrão relacionado é, se três folhas (u, v e w) estão ligados a dois nós internos (a e b) por quatro arestas (fig. 3.2a), em seguida, a folha mais distante (w) pode tornar-se um nó interno, assim, essa folha escolhida (w) conecta as outras duas folhas e substitui ambos os nós internos (a e b). Para este trabalho usa-se a promoção de nós só nos problemas que precisem projetar grandes quantidades de *bags*.

A Figura 2.6 mostra duas visões de uma árvore NJ de um subconjunto da coleção COREL[2], contendo 300 imagens e 10 classes, da qual o usuário selecionou 44 imagens para o conjunto de treinamento.

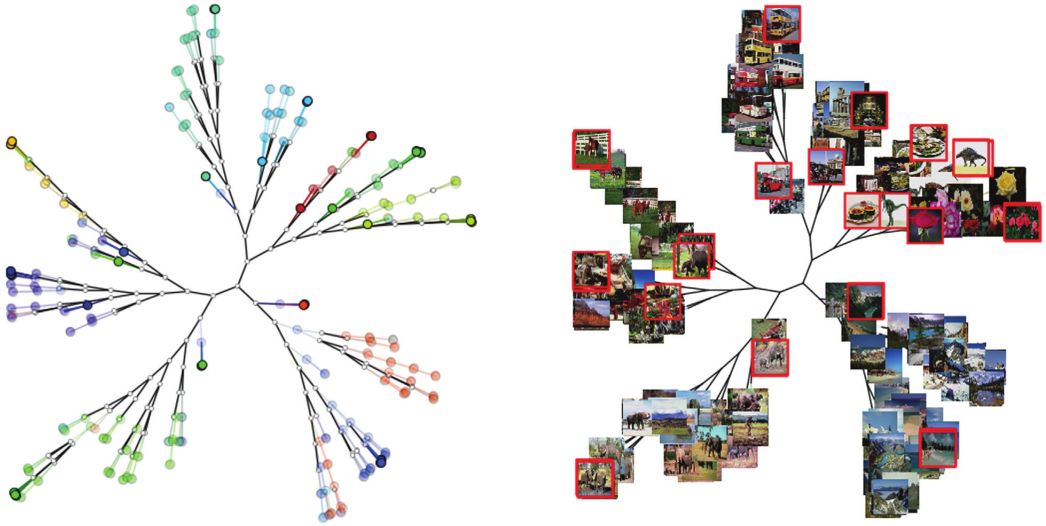
[2]http://www.corel.com/



(a) **(b)** **(c)**

Figura 2.5: Promoção de nós. (**a**) Exemplo. (**b**) Padrão. (**c**) Substituto. Retirado de (Paiva et al., 2011). Círculos preenchidos representam objetos reais, círculos brancos representam os nós internos e os triângulos representam sub-árvores folhas.

A seguir, apresentamos os trabalhos relacionado com MIL e suas aplicações no processo de classificação de dados.



(a) **(b)**

Figura 2.6: Exemplo de árvore NJ para a coleção COREL-300, com 44 instâncias selecionadas, representadas por círculos (**a**) e por imagens (**b**). Retirado de (Paiva, 2013).

2.3 Aprendizado Multi-Instâncias

O aprendizado de máquina estuda como os computadores aprendem através de um ambiente que possui informações rotuladas e não rotuladas (Faceli et al., 2011). O problema mais estudado em aprendizado de máquina é o aprendizado supervisionado, no qual são abordados dois tipos de espaços: o espaço de entrada $\mathcal{X}$ (também chamado de espaço de

instâncias) e o espaço de saída $\mathcal{Y}$ (espaço rótulo). O mapeamento $f : \mathcal{X} \rightarrow \mathcal{Y}$ é chamado de classificador.

No aprendizado supervisionado os exemplos de treinamento consistem de instâncias X_i e dos rótulos corretos Y_i sobre essas instâncias (Zhou e Zhang, 2007a). O aprendizado supervisionado tem como objetivo encontrar uma relação funcional entre as instâncias e os rótulos, a partir de um conjunto de dados rotulados (classes), para construir um modelo que será usado para prever o rótulo de outros dados ainda não rotulados (Faceli et al., 2011; Luxburg e Scholkopf, 2011).

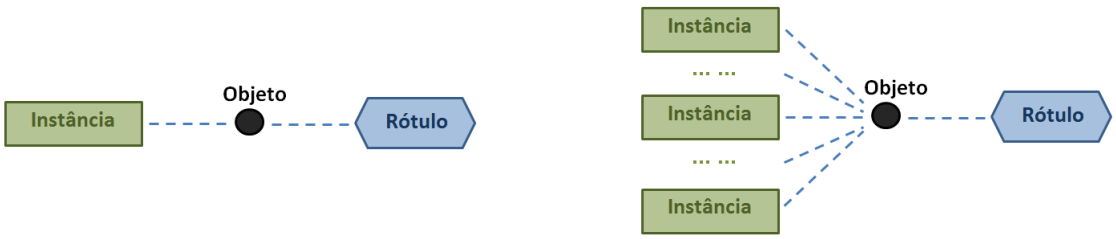
Existem problemas do mundo real que não se encaixam bem nesse modelo, um desses casos é quando um objeto é representado por múltiplas instâncias. Por exemplo, uma imagem geralmente contém várias regiões de interesse e cada uma pode ser uma instância e o conjunto de essas instâncias representa a imagem (Huang et al., 2012; Wang et al., 2013; Xu e Shih, 2013). Outro exemplo é a categorização do texto, em que um documento normalmente contém várias seções, sendo cada uma representada como uma instância (Ray e Craven, 2005a; Andrews et al., 2003). Mineração Web é outro exemplo na qual cada uma das ligações, entre sites web, pode ser considerada como uma instância (Zafra et al., 2008).

Por sua vez, *Multiple Instance Learning* (MIL) (Xiao et al., 2013), uma variação de aprendizado supervisionado, é a área de conhecimento que estuda o problema em que um objeto do mundo real, descrito por um conjunto de instâncias (*bag*), está associado a um rótulo de classe. A tarefa em MIL, é aprender um conceito a partir de dados que consistem de *bags* de instâncias. Cada *bag* é rotulado como positivo ou negativo (em problemas de duas classes), e cada um descrito como um conjunto de vetores. Um *bag* é positivo se pelo menos um dos vetores em seu conjunto se encontra dentro do conceito pretendido e negativo se nenhum dos vetores reside no conceito.

Sejam B_i^+ o *bag* positivo, Y_i^+ o rótulo positivo, B_i^- um *bag* negativo e Y_i^- um rótulo negativo. Em entornos binários $Y_i = 1$ e $Y_i = 0$. Como cada *bag* contém um conjunto de instâncias, a *j-ésima* instância em B_i^+ e B_i^- é representada como B_{ij}^+ e B_{ij}^-, respectivamente. Por questões de simplicidade, um *bag* é denotado como B quando ele representa um *bag* positivo ou negativo.

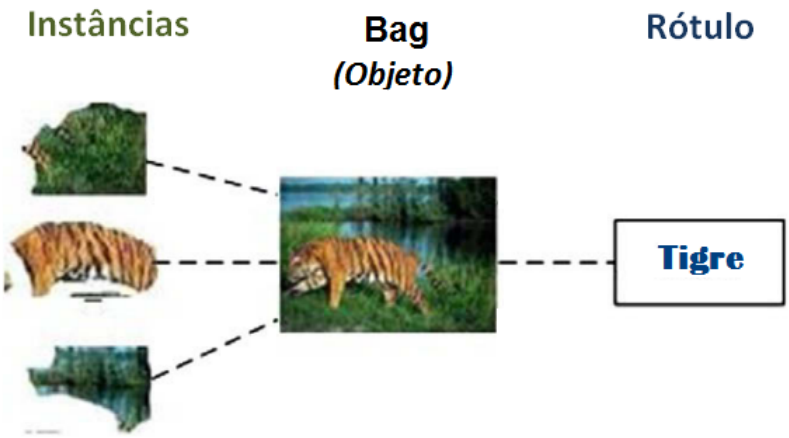
Formalmente, a tarefa do MIL é aprender uma função (Zhou e Zhang, 2007a): $f_{MIL} : 2^{\mathcal{X}} \rightarrow \{-1, +1\}$ a partir de um determinado conjunto de *bags* B_i, com $i = 1, \cdots, n$. Cada *bag* contém um conjunto de instâncias; a *j-ésima* instância dentro de um *bag* é representada por B_{ij}, e $j = 1, \cdots, n_i$, no qual n_i é o número de instâncias dentro de um *bag* B_i. Considerando um cenário de classificação binária, *bag* positivos e negativos são representados, respectivamente, por B_i^+ e B_i^-.

Na Figura 2.7 é ilustrada a diferença entre os frameworks de aprendizado supervisionado tradicionais e de aprendizado multi-instância, e na Figura 2.8 é apresentada um exemplo real de classificação multi-instância na anotação de imagens, onde a imagem representa o *bag* e as regiões segmentadas da imagem representam as instâncias desse *bag*.



(a) **(b)**

Figura 2.7: Representações visuais das diferenças entre os frameworks de aprendizado. Em (**a**) Frameworks de Aprendizado Supervisionado Tradicionais ; em (**b**) Frameworks MIL. Adaptado de (Zhou e Zhang, 2007a).



(a)

Figura 2.8: Um exemplo de classificação multi-instância na anotação de imagens, na qual uma imagem (*bag*) contem múltiplas regiões segmentadas (instâncias) e só o *bag* esta associado a uma classe. Adaptado de (Maron e Ratan, 1998).

2.3.1 Algoritmos de Aprendizado Multi-Instâncias

No MIL, cada exemplo de treinamento é um conjunto de instâncias chamadas de "*bag*", que pode ser representado por uma única instância (vetor de características). O MIL poderia ser resolvido selecionando uma única instância do *bag* para representá-lo, por exemplo,

usando a instância de maior valor (instância máxima), no entanto essa não é uma tarefa trivial. Por isso, o problema surge por não saber qual dos vetores de características representa melhor o conceito que queremos aprender a partir dos dados.

A fim de resolver esta tarefa, a primeira solução foi proposta pelo trabalho apresentado por Dietterich et al. (1997). O trabalho visa a predição de reações químicas de drogas, cujo objetivo é predizer se uma molécula de droga candidata vai se ligar à proteína alvo. As moléculas são flexíveis e podem assumir diversas formas. Assim, uma amostra positiva não representa a forma específica que a molécula possui para se ligar, e sim apenas que uma das formas que a molécula possa assumir seja a correta. Por outro lado uma amostra negativa significa que nenhuma das formas da molécula permite a ligação. Esse cenário é considerado a primeira aplicação de MIL, em que o aprendizado depende de múltiplas instâncias. Nesse trabalho se propõe o aprendizado de um *Axis Parallel Rectangle* (APR) (Dietterich et al., 1997), que vai representar o conceito aprendido, contendo pelo menos uma instância de cada *bag* positivo, e nenhuma instância dos *bags* negativos. Em geral, os resultados alcançados pelos algoritmos de aprendizado de APRs funcionam melhor que os tradicionais nesse tipo de problemas.

O algoritmo *Diverse Density* (DD) é outro dos primeiros algoritmos de MIL proposto por Maron e Lozano-Pérez (1998). DD é um dos algoritmos mais populares nesse tipo de aprendizagem e que tem algumas variantes importantes (ver seção 2.3.1.3). DD resolve o problema MIL examinando a distribuição das instâncias. Ele procura por um ponto que esteja perto de, pelo menos, uma instância de cada *bag* positivo, e que esteja longe das instâncias dos *bags* negativos. Seu objetivo é encontrar um ponto no espaço que tem um alto valor de DD, porque quanto maior a DD, maior probabilidade de ser o conceito alvo.

Logo, dos algoritmos apresentados acima, uma vasta série de métodos têm sido propostos na literatura, os quais podem ser organizadas em três principais categorias (Xiao et al., 2010, 2013).

2.3.1.1 Primeira Categoria MIL: Uso de todas as instâncias

A primeira categoria de trabalhos inicialmente define as instâncias de um *bag* com o mesmo rótulo para todos, e logo utiliza um método de aprendizagem supervisionado padrão ou um framework interativo para treinar o classificador. Por exemplo, Ray e Craven (2005a) etiqueta todas as instâncias em *bags* positivos como positivo e o *Support Vector Machine* (SVM) (Cherkassky, 1997) padrão é usado para treinar o classificador diretamente.

Ao invés de treinar o classificador de uma só vez, *mi-SVM* (Andrews et al., 2003) e *MILBoost* (Viola et al., 2006) treinam o classificador iterativamente e utilizam um método heurístico para refinar o limite de decisão do SVM.

A abordagem *mi-SVM* de Andrews et al. (2003) estende o SVM para MIL. No *mi-SVM*, um SVM é treinado para os dados de treinamento disponíveis, usando todas as instâncias negativas e instâncias positivas selecionadas. O objetivo de *mi-SVM* é maximizar conjuntamente a margem de erro dos rótulos das instâncias desconhecidas e uma função discriminante kernelizada. Uma vez que a função discriminante é obtida, atualizam-se os rótulos de um ou de várias instâncias nos *bags* positivos. Através desta modificação, o algoritmo ajusta os rótulos das instâncias positivas até ter certeza que todos os *bags* positivos seguem a definição MIL. No entanto, se há ruído de rotulagem em *bags* positivas, a precisão do *mi-SVM* pode ser prejudicada.

O algoritmo *MILBoost* (Viola et al., 2006), baseado no framework *AnyBoost* (Mason et al., 2000), treina um classificador iterativamente para refinar o limite de decisão do classificador. Em *MILBoost*, todas as instâncias inicialmente têm o mesmo rótulo do *bag* a qual pertencem, isso é feito com o intuito de treinar um classificador inicial, em seguida, um peso (o qual indica o rótulo) é assignado a cada instância. Depois em cada iteração, o peso das instâncias é atualizado utilizando o logaritmo da versossimilhança (*log-likelihood*) e classificadores subsequentes são treinados. A equação 2.6 apresenta o logaritmo da versossimilhança.

$$L = \sum_{i} (y_i log(p_i) + (1 - y_i) log(1 - (p_i)) \tag{2.6}$$

Onde p_i é a probabilidade de que o *i-ésimo bag* seja positivo. Essa probabilidade p_i é calculada usando a aproximação Noisy-OR, definada na Equação 2.7.

$$p_i = 1 - \prod_{B_{ij} \in B_i} (1 - p(B_{ij})) \tag{2.7}$$

A justificativa para essa aproximação é que a probabilidade que um *bag* seja positivo é alta, se pelo menos uma das instâncias possui alta probabilidade de ser positiva. A probabilidade de uma instância é definida na Equação (2.8)

$$p(B_{ij}) = \sigma \left(\sum_{t} \alpha_t h_t(B_{ij}) \right) \tag{2.8}$$

Onde σ é uma função sigmoidal, $h_t \in H$ é o "classificador base", i e j representam *bags* e instâncias respectivamente e α_t são os pesos positivos. O framework *AnyBoost* (Mason et al., 2000) é usado para treinar o "classificador base" h_t.

2.3.1.2 Segunda Categoria MIL: Mapeamento para espaço de bags

A segunda categoria de trabalhos, utilizam um pressuposto multi-instâncias generalizado para mapear um *bag* de instâncias em um vetor de treinamento "*bag-level*", e cada instância serve como uma dimensão no novo espaço de características.

Um dos algoritmos que cai nesta categoria é o *Constructive Clustering-based Ensemble* (CCE) proposto por Zhou e Zhang (2007b). Este algoritmo aplica estratégias de clustering baseado no fato que as instâncias não têm rótulo. Em *CCE*, primeiro, as instâncias de todos os bags são coletadas e agrupadas em d grupos usando o algoritmo de clustering *k-means*. Em seguida, cada *bag* é representado como um novo vetor de características no "*bag-level*". Esse novo vetor é um vetor de características binário, onde o valor da *ith*-característica é definido como um (1), se o *bag* em questão possui instâncias que caem no *ith*-grupo e zero (0) em caso contrário. Assim, ao *bag* ser representado por um único vetor de característica binário, classificadores padrões (nesse trabalho o classificador SVM) já podem ser utilizados para distinguir diferentes classes de *bags*. Também se conhecem outros algoritmos que fazem uso de técnicas de clustering para abordar problemas multi-instância, como o algoritmo *BARMIT* proposto por Zhang e Zhou (2009), o qual diferente de *CCE* trabalha no espaço dos *bags*, adaptando o algoritmo *k-medoids* para classificar os *bags*.

O algoritmo *Diverse Density Support Vector Machine* (DD-SVM) (Chen et al., 2004) e seu sucessor *Multiple-instance Learning via Embedded Instance Selection* (MILES) (Chen et al., 2006) são outros exemplos típicos nesta categoria.

O algoritmo DD-SVM (Chen et al., 2004), é um predecessor do algoritmo *MILES* (Chen et al., 2006), que é conceitualmente muito similar. *DD-SVM* assume que a classificação dos *bags* está apenas relacionada com algumas propriedades dos *bags*, e assume que os rótulos estão relacionados com um conjunto de instâncias protótipo, os quais são selecionados a partir de uma função local máxima de DD. As instâncias protótipo representam uma classe de instâncias que é mais provável que apareça em *bags* positivos do que em *bags* negativos.

Consequentemente, ele resolve o problema MIL, transformando o espaço de característica original para um novo espaço de características de *bags* usando as instâncias protótipo para mapear cada uns dos *bags* a um ponto em um novo espaço de características de *bags*. Nesse novo espaço, o problema original MIL torna-se um problema de aprendizado supervisionado normal, no qual o treinamento de um SVM é realizado.

O algoritmo MILES (Chen et al., 2006) incorpora *bags* em um espaço de características definido por instâncias, com base em pontuações de similaridade, e depois é aplicado o

algoritmo *1-norma SVM* para o conjunto de dados transformados, para selecionar as instâncias importantes para a predição.

MILES usa as instâncias nos *bags* de treinamento, como candidatos para pontos de destino (*target points*). Um mapeamento do espaço de característica é definido, em que cada atributo representa a proximidade de uma instância para um ponto de destino candidato.

Cada *bag* de treinamento é mapeado para este espaço (com rótulos de classe anexos), em que as coordenadas de um determinado *bag* no espaço de características representam a similaridade dos *bags* com várias instâncias no conjunto de treinamento. Quando o número de instâncias no conjunto de treinamento é grande, o mapeamento de características (instâncias) produz um espaço possivelmente de alta-dimensão e muitas funções podem ser redundantes ou irrelevantes. As características irrelevantes não contribuem para a classificação de *bags*, e as características redundantes podem ser semelhantes umas às outras.

Por isso fica essencial e indispensável selecionar um subconjunto de características, os quais serão mapeados depois, que sejam mais relevantes para o problema de classificação em estudo. Embora quaisquer abordagens de seleção de características possam ser aplicadas para esse fim, eles utilizaram o algoritmo *1-norma SVM*, o qual é uma abordagem conjunta que constrói o classificador e seleciona características importantes ao mesmo tempo.

Embora, *DD-SVM* e *MILES* mantêm uma similar precisão na classificação de dados, ambos podem transformar o MIL em um problema de alta dimensionalidade. Isto ocorre porque a dimensão do vetor de treinamento "*bag-level*" é igual ao número total de instâncias no conjunto de treinamento. Se o número de instâncias é grande, o vetor de treinamento "*bag-level*" pode passar a ser um problema de alta dimensionalidade.

Para reduzir a dimensão dos vetores "*bag-level*", se propom outros métodos multi-instância (mostrados na seguinte seção 2.3.1.3), os quais têm como objetivo selecionar uma instância de cada *bag* positivo chamada de instância protótipo ao invés de criar uma nova instância no "*bag-level*" para representar um *bag*.

2.3.1.3 Terceira Categoria MIL: Seleção de instâncias protótipo

A terceira categoria de trabalhos se foca na seleção de um subconjunto de instâncias a partir de *bags* positivos para aprendizado do classificador.

Dentro desta categoria seria encontrado o algoritmo DD (Maron e Lozano-Pérez, 1998) descrito como os primeiros algoritmos que tratam o problema MIL. Um refinamento deste

algoritmo é o algoritmo *Expectation Maximization - Diversity Density* (EM-DD) (Zhang e Goldman, 2001). No *EM-DD*, apenas uma instância por *bag* é considerado determinante para o rótulo do *bag*. Esta instância é estimada utilizando *Expectation Maximization* (EM).

EM-DD começa com uma estimativa inicial de um ponto alvo p obtido na forma padrão à tratar os pontos desses *bags* positivos, e depois executa repetidamente duas etapas que combina EM com DD para procurar a hipótese de máxima verossimilhança. Na primeira etapa (E-Expectation), a hipótese atual p é usada para escolher uma instância de cada *bag* que é o mais provável para ser o responsável pelo rótulamento dado um *bag*. Na segunda etapa (M-Maximization) é usada a gradiente ascendente do algoritmo DD padrão para encontrar um novo p_0 que maximiza DD(p), ou seja, a segunda etapa reduz o MIL ao problema de aprendizado supervisionado padrão. Uma vez que a segunda etapa for concluída, a hipótese proposta p é redefinida para p_0 e retorna de novo para a primeira etapa até que o algoritmo convergir.

Support Vector Machines for Multiple-Instance Learning (MI-SVM) (Andrews et al., 2003) adota um quadro interativo para treinar o classificador. A cada iteração, apenas uma instância de cada *bag* positivo é selecionada. *MI-SVM* é adequado para tarefas onde apenas os rótulos dos *bags* são procurados. Para isto, *MI-SVM* visa maximizar a margem do *bag*, que é definida como a margem da instância protótipo mais confiável no caso de *bags* positivas, ou as margens da instância menos negativa em caso de *bags* negativos. As instâncias selecionadas são usadas para treinar o classificador.

Multiple Instance Learning with Instance Selection (MILIS) proposto por Fu et al. (2011), seleciona uma instância de cada *bag* positivo e negativo para produzir instâncias protótipo. Para isso, primeiro, as instâncias nos *bags* negativos são modelados como uma função de distribuição de probabilidade com base na estimativa da densidade kernel. Inicialmente, a instância mais positiva ea instância mais negativa são selecionados em cada *bag* positivo e *bag* negativo respectivamente, baseado na função de distribuição calculado anteriormente, as quais são chamadas de instâncias protótipo. Essas instâncias formam o espaço de características para criar o "*bag-level*" em que um SVM linear é treinado. Além disso, *MILIS* emprega um esquema de otimização para a seleção das instâncias protótipo.

Outro método que tenta selecionar uma instância protótipo dentro de cada *bag* é o *Similarity - Based Multiple-Instance Learning* (SMILES) proposto por Xiao et al. (2013). Em *SMILES* as instâncias dentro de bags positivos são chamados de *instâncias ambíguas* desde que um *bag* positivo pode conter instâncias positivas assim como instâncias negativas (os rótulos das instâncias não são conhecidas). Inicialmente, *SMILES* seleciona uma instância ambígua desde cada *bags* positivo como um candidato positivo inicial e, em

seguida, cada instância é associada a dois pesos de similaridade medindo a sua semelhança com as classes positivas e negativas (um peso de similaridade para as classes positivas e um para as classes negativas). Finalmente, as instâncias ambíguas junto com seus pesos são incorporadas dentro de uma formulação estendida do SVM, onde a seleção de positivos candidatos e a similaridade de pesos pode ser atualizada para melhorar a classificação baseado num *framework de aprendizado heurístico*.

SMILES é um método que obteve resultados promissórios, no entanto, como os mesmos autores indicam Xiao et al. (2013), o método de geração de pesos de similaridade poderia resultar em informações imprecisas para a classificação e fazer o classificador tendencioso. Outro problema, é o alto custo computacional da formulação estendida do SVM o qual é $O(n_1 * m + n_2 * m + n_1 * (m-1))^2$ em comparação com o custo computacional do SVM padrão o qual é $O(n_1 * m + n_2 * m)^2$, onde n_1 representa um *bag* positivo, n_2 representa um *bag* negativo e m representa a quantidade total de instâncias. Pelo qual os métodos multi-instância que utilizam o SVM padrão para treinar o conjunto de treinamento têm ainda vantagens sobre o *SMILES*.

Salient Instance Selection for Multiple Instances Learning (MILSIS)

MILSIS proposto por Yuan et al. (2012) é outro método que pode ser classificado dentro da terceira categoria MIL, que têm resultados comparáveis com o *SMILES* e utiliza um SVM padrão para treinar o conjunto de treinamento multi-instância. *MILSIS* identifica instâncias protótipo chamadas de instâncias saliente, as quais são consideradas como as *verdadeiras* instâncias positivas em *bags* positivos. *MILSIS* assume que um *bag* positivo contem instâncias positivas e negativas, enquanto um *bag* negativo contém apenas instâncias negativas. Por isso o objetivo do MILSIS se foca na seleção das *verdadeiras* instâncias positivas em *bags* positivos. No entanto, a seleção da instância saliente não é uma tarefa fácil, uma vez que o *bag* positivo contém instâncias positivas e negativas, mas pelo menos uma instância positiva.

O método *MILSIS* está formado por dois processos denominados "Seleção Áspera" e "Seleção Fina" para a obtenção das instâncias salientes.

No processo de Seleção Áspera se tem como objetivo selecionar apenas uma *óptima* instância positiva entre todos os *bags* positivos. Para isso, primeiro, todas as instâncias em *bags* negativos são agrupadas em um conjunto B^-. Em seguida, é computado o valor da saliência para cada instância $Sal(B_{ij}^+)$ em um determinado *bag* B_i^+, com a seguinte Equação:

$$Sal(B_{ij}) = \sum_{B_{ik} \in B_i \setminus \{B_{ij}\}} d(B_{ij}, B_{ik}), \tag{2.9}$$

Onde $d(.,.)$ representa a Distância Euclidiana entre duas instâncias. Um valor alto de saliência indica que a instância é diferente das outras instâncias no mesmo *bag*.

Algoritmo 2: MILSIS: Salience Instances Selection

Entrada: B, $SalNum$
Saída: Instâncias protótipo T (instâncias salientes)
$B^- = \{B_{rt} | B_{rt} \in B_r^-, r = 1, 2, ..., n^-\}$
//Seleção Áspera.
para $i = 0$ *até* n^+ **faça**
  Calcular $Sal(B_{i1}^+)$ para cada instância em B_i^+ usando a Equação 2.9;
  Reordenar todas as instâncias contidas em B_i^+ de forma descendente (usando os valores da saliência).
  Calcular $D(B_{i1}^+, B^-)$ e $D(B_{im}^+, B^-)$.
  if $D(B_{i1}^+, B^-) > D(B_{im}^+, B^-)$ *e* $D(B_{i1}^+, B^-) > maxDist$ **then**
    $maxDist = D(B_{i1}^+, B^-)$
    $optPosInst = B_{i1}^+$
  else
    if $D(B_{im}^+, B^-) > D(B_{i1}^+, B^-)$ *e* $D(B_{im}^+, B^-) > maxDist$ **then**
      $maxDist = D(B_{im}^+, B^-)$
      $optPosInst = B_{im}^+$
  end
fim
//Seleção Fina.
$optNegInst = \arg\max_{t \in B^-}(d(t, optPosInst))$
para $i = 1$ *até* n^+ **faça**
  if $d(B_{i1}^+, optNegInst) > d(B_{im}^+, optNegInst)$ **then**
    Agregar $B_{i1}^+, ..., B_{iSalNum}^+$ a T
  else
    Agregar $B_{i(m-SalNum+1)}^+, ..., B_{im}^+$ a T
  end
fim
retorna T

Depois de calcular as salientes dentro de B_i^+, as instâncias são reordenadas segundo seus valores de saliência, desde o máximo ($j = 1$) até o mínimo ($j = m$) valor de saliência. Em seguida, este é utilizado para calcular a probabilidade que as instâncias B_{i1}^+ (máximo)

e B_{im}^{+} (mínimo) são positivas dada B^{-} usando a Equação 2.10. Isso é realizado com o intuito de selecionar a *ótima* instância positiva, o qual vai representar o *bag* B_i.

$$Pr(l(B_{ij}) = +1|B^{-}) = 1 - \exp(-D(B_{ij}, B^{-})/\sigma^{2}), \tag{2.10}$$

Onde σ é um factor de escala maior do que 0. $D(B_{ij}, B^{-})$ é a distância mínima entre B_{ij} e todas as instâncias em B^{-}.

$$D(B_{ij}, B^{-}) = \min_{B_{rt} \in B^{-}} d(B_{ij}, B_{rt}), \tag{2.11}$$

Desde que $Pr(l(B_{ij}) = +1|B^{-})$ é proporcional a $D(B_{ij}, B^{-})$ (Yuan et al., 2012), dada a Equação 2.11 é possível ver que se uma instância esta longe de um conjunto de instâncias negativas, eles teriam uma baixa similaridade e, portanto, a instância é susceptível de ser rotulada como positiva; de outra forma, a instância é susceptível de ser rotulada como negativa. Finalmente, todas as probabilidades de cada *bag* são comparados a fim de encontrar a *ótima* instância positiva.

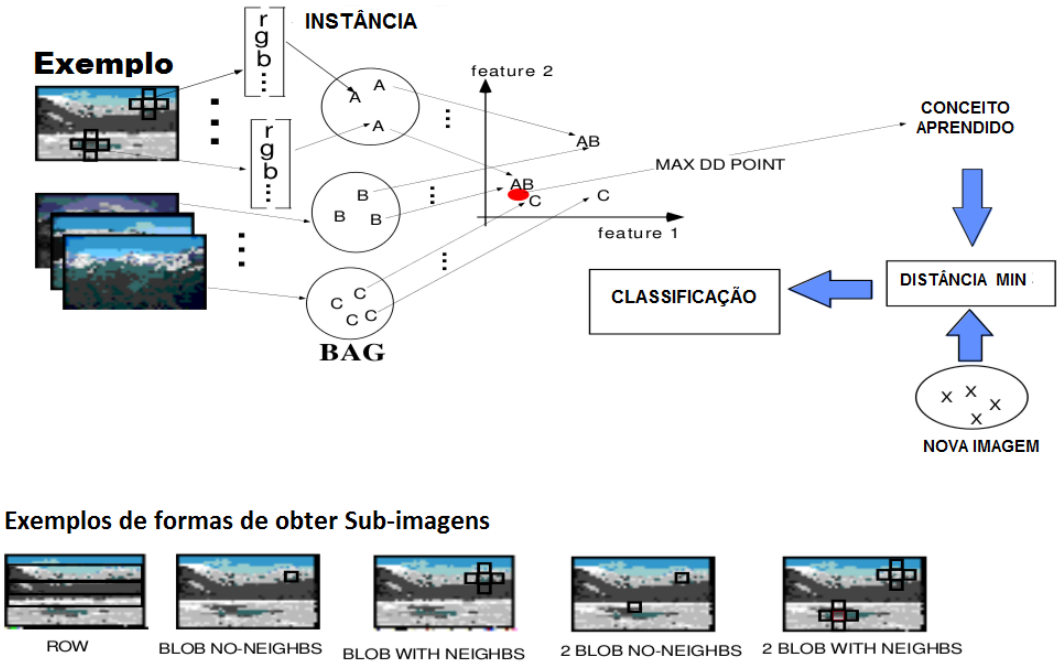
No processo de *Seleção Fina*, se tem como objetivo selecionar uma *verdadeira* instância positiva (instância saliente) para cada *bag*, utilizando a *ótima* instância positiva encontrado acima. Para isso, primeiro, uma *ótima* instância negativa é selecionada a partir de B^{-} usando a *ótima* instância positiva, e as instâncias B_{i1}^{+} e B_{im}^{+} de cada *bag* calculado acima, tudo isso, com o objetivo de selecionar uma *verdadeira* instância positiva dentro de cada *bag*. Essas *verdadeiras* instâncias positivas farão parte das instâncias salientes (Intâncias Protótipo). O Algoritmo 2 resume todo o processo de seleção de instâncias salientes. Note que $B = B^{+} \cup B^{-}$, e *SalNum* é o número de instâncias salientes.

Depois de obter a instância saliente (instância protótipo) para cada *bag* positivo, em seguida, é treinado um SVM padrão para a classificação de *bags*. Note que *MILSIS* apenas utiliza a maior instância saliente para representar um bag e treinar o classificador SVM. Neste trabalho apresentamos uma variação do *MILSIS* chamado MILTree-SI (ver seção 3.3), onde também trabalhamos com as instâncias em *bags* negativos e com mais de uma instância protótipo por *bag*.

2.3.2 Aplicações de Aprendizado Multi-Instância

Técnicas MIL foram utilizadas com sucesso em diversas aplicações para a classificação de dados, como a classificação de cenas (Zhang et al., 2012a; Chen et al., 2004; Maron e

Ratan, 1998). Na Figura 2.9 é ilustrado o Esquema do sistema MIL usando o algoritmo DD para classificação de cenas naturais (Maron e Ratan, 1998). Nesse trabalho uma imagem é considerada como um *bag* e as instâncias do *bag* são varias sub-imagens. Estas sub-imagens podem ser obtidas a partir de sete diferentes formas (Figura 2.9 apresenta alguns exemplos). Cada sub-imagem (instância) é representado por um vetor de características dos valores médios RGB do bloco central (sub-imagem). A ideia principal é utilizar o algoritmo DD para encontrar áreas no espaço de características que estejam o mais perto possível de pelo menos uma instância positiva desde os *bags* positivos e o mais longe possível dos *bags* negativos. Quando essa área (máximo ponto DD) é encontrada, uma nova imagem é classificada como positiva se pelo menos uma das suas sub-imagens esta localizado dentro da área do ponto máximo DD.



(a)

Figura 2.9: Esquema do Sistema MIL para classificação de Cenas Naturais, e alguns exemplos das formas de obter sub-imagens (*Row,Blob no-neighbs*, etc). Adaptado de (Maron e Ratan, 1998).

A representação de Múltiplas Instâncias também foi usada para CBIR (Maron e Lozano-Pérez, 1998; Zhang et al., 2002) e classificação de imagens (Venkatesan et al., 2012; Wang et al., 2013; Xu e Shih, 2013). Por exemplo, o problema de classificação de imagens foi abordado no contexto de MIL por Xu e Li (2007b), onde as imagens são considerados bags e as regiões segmentadas das imagens são considerados de instâncias. Nesse trabalho foi utilizado o EM-DD, um método MIL, para aprendizado das instâncias

protótipo (IPs) desde uma coleção de instâncias positivas e negativas para cada categoria. Cada instância protótipo (alguma região da imagem) representa um tipo de conceito alvo, que distingue uma classe de outra. Nesse trabalho uma SVM multi-classe é treinada em um novo espaço de características definido pelos protótipos de instância de todas as categorias. para produzir os limites de classificação multi-classe. Como caso de estudo foi utilizado o conjunto de dados SIMPLIcity-II que tem cinco categorias: *object*, *people*, *portrait*, *scene*, e *structure*. Na Figura 2.10 são ilustrados alguns conceitos-alvo de um conjunto de imagens.

(a)

Figura 2.10: Os conceitos-alvo de quatro categorias no conjunto de dados SIMPLIcity-II são representados por regiões conceituais. Para "People", as diferentes formas dos humanos são retornados como conceitos-alvo. Para "Portrait", os conceitos-alvo são principalmente a pele e o cabelo com cores variadas. Para "Scene", os principais conceitos-alvo são montanha, céu, água do mar e plantas. Para a "Structure", os conceitos-alvo são alguns tipos de estruturas de construção, como paredes, janelas e telhados. Retirado de (Xu e Li, 2007a).

O problema de categorização do texto também foi abordado no contexto de MIL, em que um documento é dividido em várias seções, sendo cada uma representada como uma instância (Xiao et al., 2013; Ray e Craven, 2005a; Andrews et al., 2003). A definição Múltipla Instância também foi utilizada no contexto de atividades para predição de drogas (Dietterich et al., 1997) em que os conjuntos de instâncias foram organizadas naturalmente em *bags*, que são rotuladas para o treinamento, em vez de instâncias individuais.

2.4 Classificação Visual de Dados

As bases de dados contêm uma grande quantidade de informações ocultas que podem ser utilizadas para tomar decisões inteligentes. A classificação é uma forma de análise de

dados que tenta reduzir um grande número de dados para um número menor de grupos, visando facilitar a descrição e ilustração dos dados.

A classificação de dados é o processo que divide um conjunto de dados em grupos mutuamente exclusivos, de modo que cada membro de um grupo esteja o mais próximo possível, um do outro, e diferentes grupos estejam afastados (Langley, 1996; Mitchell, 1997). A classificação automática de dados é um processo que envolve dois passos: no primeiro passo se constrói um modelo que descreve o conjunto preliminar de classes. Esse modelo é construído através da análise de um conjunto de dados de treinamento. Cada dado pertence a uma classe específica conhecida, por isso, a classificação é parte do que é conhecido como aprendizado supervisionado. Em contraste, no aprendizado não supervisionado a classe à que pertence cada dado e o número de classes é desconhecida.

No segundo passo, o modelo é utilizado para gerar a classificação de um conjunto de dados desconhecidos. Em seguida, estimamos a precisão do modelo. A precisão é a porcentagem de exemplos no conjunto de treinamento que foram corretamente classificados.

Diversos autores (Rüger, 2006; Chen et al., 2008; Belattar e Mostefai, 2013; Pipanmaekaporn, 2013) abordam a ideia de utilizar o conhecimento do usuário no processo de recuperação de imagens e no processo de classificação de dados, de modo a melhorar os resultados finais. *Relevance feedback* (Sivakamasundari e Seenivasagam, 2012) tem-se mostrado uma ferramenta poderosa para concretizar essa ideia. *Relevance feedback* cobre uma variedade de técnicas destinadas a melhorar a consulta do usuário e facilitar a recuperação de informações relevantes para o usuário.

Neste sentido, o *Relevance feedback* se mostra útil em um sistema de classificação visual de dados, já que permiti ajustar o classificador às necessidades do usuário mediante um processo iterativo. A classificação visual dos dados foi abordado no trabalho de (Migut e Worring, 2010). Nesse trabalho apresenta-se um *framework* que integra a exploração visual interativa com o aprendizado de máquina para apoiar o processo de tomada de decisões. Esse *framework* permite a visualização da fronteira de decisão do classificador e a sua correspondente curva de performance, permitindo que o usuário explore visualmente os custos das diferentes configurações realizadas sobre o modelo, e, portanto, usar o modelo de classificação mais adequada e tomar decisões mais informadas e confiáveis. As mudanças nos modelos podem ser realizadas visualmente através da: mudança na posição dos pontos na curva de performance e o re-etiquetado das instâncias com o intuito de treinar novamente o classificador.

Heimerl et al. (2012) também apresenta um *framework* que permite a construção de classificadores de uma forma iterativa e visualmente. Nesse trabalho se mostram três abordagens para a criação de um classificador, a primeira abordagem só utiliza aprendi-

Figura 2.11: Visualização da fronteira de decisão (fila dois) e a sua correspondente curva de performance (fila um) de três classificadores para o conjunto de dados de imagens *Forensic Psychiatric*. Adaptado de (Migut e Worring, 2010)

zado ativo e as outras duas abordagens utilizam visualizações para a criação dos modelos de classificação. Estas visualizações permitem aos usuários explorar visualmente o estado do classificador no contexto dos documentos etiquetados, bem como avaliar a qualidade do classificador. Nesse trabalho, a visualização (layout) dos documentos está baseada na relação de similaridade entre os documentos. Além disso, suas visualizações ampliam a área da fronteira de decisão, a fim de ajudar os usuários em rotular aqueles documentos que têm o maior potencial para melhorar o classificador. O *framework* apresentado nesse trabalho também oferece varias visualizações ligadas à principal visualização do conjunto que está sendo classificado, com o intuito de fornecer informações adicionais sobre a coleção de documento, bem como o modelo de classificação atual.

O processo de classificação visual dos dados foi abordado no trabalho de Paiva et al. (2011) no contexto de aprendizado supervisionado e semi−supervisionado, tendo em consideração o *feedback* do usuário. Nesse trabalho, apresenta se uma aplicação em que as visualizações de similaridade são visualizadas utilizando uma árvore NJ melhorada, tanto no uso do espaço e velocidade de processamento, para ajudar o usuário em várias etapas do processo de classificação de dados (imagens e textos), com ênfase na classificação de imagens.

O aplicativo desenvolvido oferece um conjunto de ferramentas para a classificação e análise visual, graças às propriedades oferecidas pelas projeções. Este conjunto de ferramentas é utilizado para a classificação interativa. A partir de um conjunto de

Figura 2.12: Ferramenta visual para o treinamento do classificadores. Adaptado de (Heimerl et al., 2012)

dados rotulado ou não−rotulado pode se acumular conjuntos de treinamento dentro de conjuntos classificados, combinando classificações automáticas e classificações manuais apoiadas visualmente.

O aplicativo tem três características principais que apoiam o processo de classificação de dados aproveitando o *feedback* do usuário. A primeira característica é apoiar à usuário a compreender as razões para o fracasso na classificação automática. Para isso, foi desenvolvida uma funcionalidade chamada de *Class Matching* que verifica os resultados da classificação, localizando os pontos onde não houve uma classificação adequada. *Class Matching* é utilizada em situações nas quais existe um esquema de rotulamento ideal para as instâncias, para outros casos é possível apresentar o número de casos corretamente classificados e outras medidas convencionais de classificações. A figura 2.13 ilustra o processo de classificação de um conjunto de dados, utilizando um subconjunto de treinamento de 350 imagens e mostrando no final o *Class Matching* da classificação.

Outra característica da ferramenta é que os usuários podem ver ou alterar a classificação à vontade, mesmo que tenham obtido uma adequada classificação automática. A terceira característica da ferramenta é tratar a rotulagem atual do conjunto de dados quando ela não é adequada para os objetivos do usuário. Nesse caso, o aplicativo permite fazer duas atividades: primeiro, ela permite rotular novamente os conjuntos de treinamento já rotulados com o objetivo de corrigir a classificação, e, segundo, permite rotular

conjuntos de dados não rotulados. O aplicativo utiliza o *feedback* visual para melhorar outras classificações, bem como corrigir uma classificação atual.

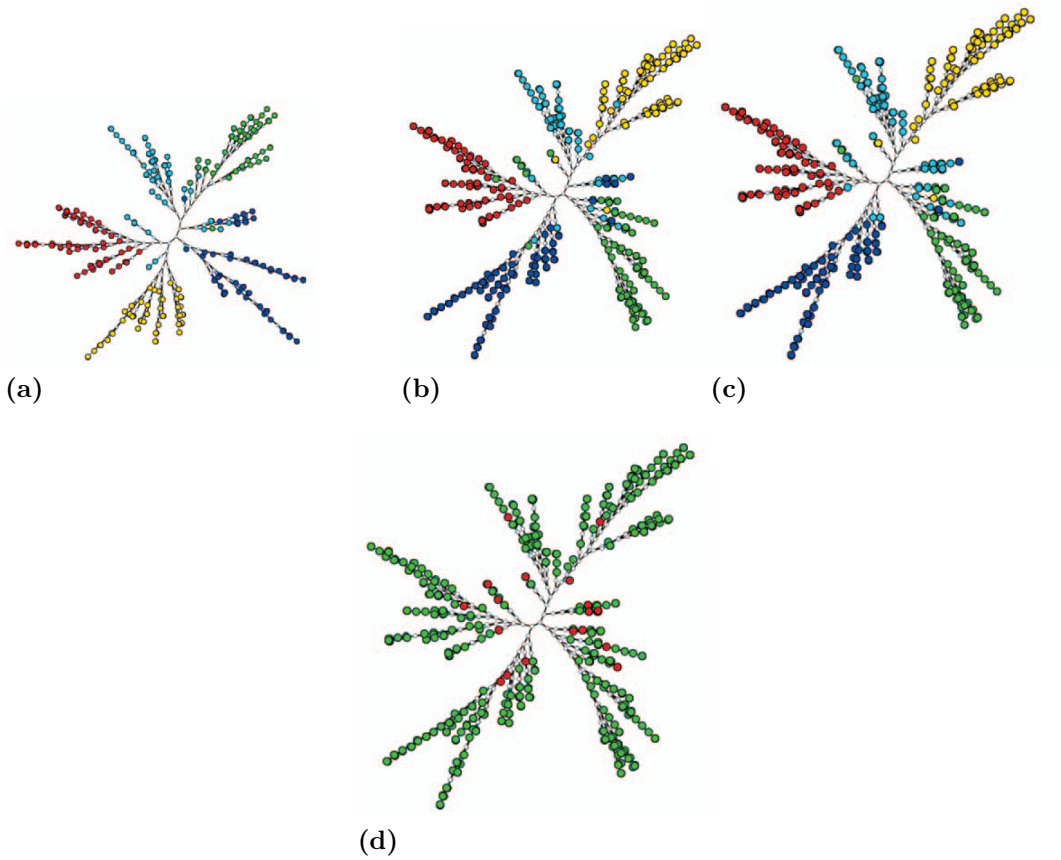


Figura 2.13: Ilustração do processo de classificação de forma iterativa utilizando um subconjunto dos dados COREL como conjunto de treinamento. (**a**) Conjunto de treinamento de 350 imagens. (**b**) Conjunto de dados com 500 imagens. A cor define a classe alvo. (**c**) Dados Classificados com 500 imagens. (**d**) *Class Matching* do conjunto de dados. Pontos em vermelho foram erroneamente classificados. Retirado de (Paiva et al., 2011).

No trabalho de Paiva et al. (2012) se apresenta uma abordagem para a classificação visual de dados, utilizando uma associação entre o método de redução de dimensionalidade baseada no *Partial Least Squares* (PLS)(Wold, 1985) e técnicas de visualização. Nesse trabalho se apresenta o PLS como uma ferramenta flexível e precisa para a análise visual de conjuntos de dados, incluindo o *feedback* do usuário no processo da redução de dimensão. O modelo PLS melhorado, além de poder ser aplicado em coleções previamente rotuladas, pode ser aplicado em coleções sem nenhum rótulo prévio. Para coleções não rotuladas a coleção é submetida a um prévio procedimento de agrupamento, considerando um agrupador como um classificador. As técnicas de visualização são utilizadas para gerar um *layout* que vai guiar o usuário na escolha de instâncias em cada classe (coleções rotuladas)

ou grupos (coleções não rotuladas), produzindo bons modelos PLS que destaquem as diferenças entre as classes de dados.

O PLS para a classificação de coleções de dados utiliza o valor da regressão das instâncias de teste no(s) modelo(s) criado(s) a partir de um conjunto de amostras, sendo capaz de trabalhar com baixo número de amostras, e permitindo a reutilização do modelo criado para uma coleção em outras coleções.

Recentemente, uma metodologia de classificação visual de dados foi proposto por (Paiva et al., 2015). Esta metodologia apoia aos usuários em tarefas relacionadas com a classificação de dados, como, por exemplo, a seleção do conjunto de treinamento; a criação do modelo de classificação, aplicação e verificação; e ajuste do classificador, apoiada em visualizações baseados na similaridade. As técnicas de visualização usadas para apoiar a classificação são as projeções multidimensionais e as árvores *Neighbor Joining*. O trabalho também apresenta uma abordagem para a criar modelos de classificação incremental através do algoritmo *Locally Weighted Projection Regression* (Vijayakumar et al., 2005). Esta metodologia está implementada na ferramenta *Visual Classification System* (VCS), o qual é descrito a seguir.

Visual Classification System (VCS)

VCS é uma ferramenta desenvolvida no grupo de Visualização, Imagens e Computação Gráfica (VICG) do ICMC -USP São Carlos. Esta ferramenta é resultado de uma tese de doutorado "Técnicas computacionais de apoio a classificação visual de imagens e outros dados" (doutorado José Gustavo Paiva) (Paiva, 2013).

VCS é uma ferramenta computacional que provê um conjunto de funcionalidades para realizar a classificação visual de dados como imagens e textos, seguindo a metodologia apresentada em (Paiva et al., 2015). A Figura 2.14 mostra uma imagem da interface do *VCS*.

VCS utiliza técnicas de visualização baseadas em posicionamento de pontos, como as árvore de similaridade *NJ* (Paiva et al., 2011), *LSP* (Paulovich et al., 2008), projeção *Interactive Document Map* (IDMAP) (Minghim et al., 2006) e a projeção ISOMAP (Tenenbaum et al., 2000), para construir layouts que ajudem aos usuários:

- Na criação e ajuste dos conjuntos de treinamento.
- Construção e aplicação de modelos de classificação utilizando várias técnicas conhecidas como o *SVM*.
- Realização de uma análise profunda dos resultados da classificação.

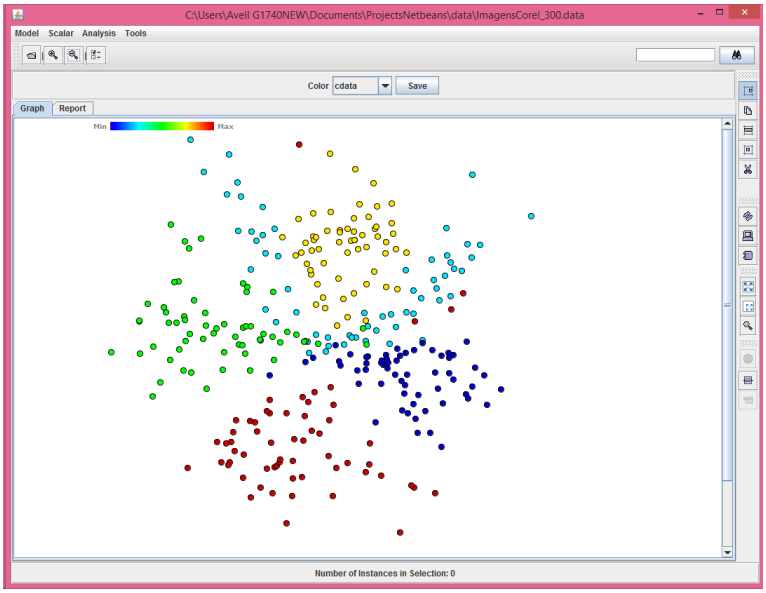


Figura 2.14: Tela principal do *Visual Classification System* (VCS).

- Utilizando os layouts criados, os usuários podem também ajustar os modelos, a fim de adaptá-las a cenários de classificação específicos.

Além das tarefas relacionadas com o processo de classificação, o *VCS* também oferece várias funcionalidades para explorar os layouts, tais como aproximações (*zoom*), a visualização de um determinado dado (por exemplo imagem ou texto) ou de todos os dados ou um grupo selecionado, a fim de detectar particularidades e tendências no conjunto de dados que ajude a compreender os resultados dos classificadores.

Uma funcionalidade adicional para análise visuais dos dados implementado no *VCS* é a redução de dimensionalidade, o qual é realizado utilizando a técnica *Partial Least Squares* (Wold, 1985).

VCS tem se mostrado como uma ferramenta útil para apoio ao usuário em tarefas de classificação de dados, no entanto, *VCS* não consegue realizar a visualização nem a classificação de dados multi-instância. É por isso que uns dos objetivos deste trabalho é incluir no *VCS* o layout criado para a visualização de dados multi-instância e os métodos necessário para realizar a classificação multi-instância.

2.5 Considerações Finais

Este capítulo apresentou um conjunto de conceitos e técnicas relacionadas ao processo de visualização da informação, aprendizado multi-instâncias e classificação visual de dados.

Como foi descrito, uma área dentro do aprendizado de máquina que trabalha com múltiplas instâncias ao invés de instâncias individuais, é a técnica MIL. Ao contrário do aprendizado supervisionado tradicional, aprendizado multi-instância lida com problemas em que as informações do rótulo são apenas parcialmente expressadas como o rótulo de *bags* de instâncias. O objetivo do MIL é obter uma hipótese a partir dos *bags* de treinamento para predizer os rótulos para os novos *bags*, utilizando as informações contidas nas instâncias de cada *bag*. Muitas abordagens foram desenvolvidas ao longo do tempo para tentar resolver o problema de classificação de dados multi-instância em cenários MIL. No entanto não se tem conhecimento de trabalhos que abordem o problema a partir de uma perspectiva visual (utilizando técnicas de visualização da informação), o qual permitiria a inserção do usuário no processo de classificação.

Sabe-se que as ferramentas de visualização de informação permitem fazer a exploração e análise visual de coleções de dados ajudando a comunicar ao usuário o seu conteúdo informacional. O usuário consegue também visualizar as relações entre os dados e interagir com o *layout*. É assim que vários problemas de classificação padrão (onde os objetos estão representados por simples instâncias ao invés de *bags*) já têm usado técnicas de visualização para ajudar os usuários a participar e inserir seus conhecimentos no processo de classificação com o intuito de melhorar os resultados da classificação, como é apresentado na seção de classificação visual de dados.

No entanto, as técnicas utilizadas na classificação visual dos dados dos trabalhos anteriores não conseguem trabalhar bem com os dados multi-instância, devido a que eles não escalam visualmente aparecendo problemas como a oclusão dos dados e também não mostram de forma intuitiva e natural a estrutura dos dados multi-instância (um nível de *bags* e um nível de instâncias). Desta forma, que neste trabalho pretende-se desenvolver uma visualização multi-escala e métodos de seleção de instâncias protótipo para melhorar as projeções dos *bags* e para obter melhores resultados na classificação dos dados multi-instância. O próximo capítulo apresenta a metodologia proposta para este trabalho.

Capítulo

3

Metodologia

3.1 Considerações Iniciais

A visualização de informação consiste em utilizar interfaces visuais interativas, cujo principal objetivo é representar claramente o conteúdo informacional de um conjunto de dados para o usuário final. Os processos de classificação guiados pelo usuário, em particular, podem se beneficiar consideravelmente do emprego da visualização (Paiva, 2013). Técnicas de Visualização já foram aplicadas em processos de aprendizado padrão, onde os conjuntos de treinamento são simples instâncias, conforme descrito no Capítulo 2. Porém, não se tem conhecimento da aplicação de técnicas de visualização ao aprendizado multi-instâncias que, além de ser um modelo importante em diversas aplicações, pela sua própria complexidade de utilização sugere a necessidade de apoio visual.

Como foi mencionado na Seção 2.3, a ideia fundamental do aprendizado multi-instâncias é permitir a classificação de *bags* (coleções de instâncias), ao invés de instâncias individuais. Um conjunto de treinamento típico para um problema MIL inclui *bags* e rótulos binários associados a esses *bags*. Assim, um *bag* de rótulo *negativo* contém apenas instâncias negativas, enquanto um *bag* com rótulo *positivo* inclui ao menos uma instância positiva, podendo também incluir instâncias negativas (Maron e Lozano-Pérez, 1998). Nossa abordagem assume que tanto *bags* positivos quanto negativos podem conter instâncias positivas e negativas, o que é uma suposição mais realista em dados complexos reais.

Em visão computacional e aprendizado de máquina, diversos problemas podem ser considerados como do tipo MIL, como, por exemplo,CBIR onde cada imagem contém diversas regiões, porém apenas aquelas contendo informação de uma categoria específica são as imagens de interesse. Uma região de interesse pode ser um objeto ou uma região da qual é extraída um vetor de características. Nesse ponto de vista, cada imagem é uma coleção de instâncias, e cada região uma instância do problema (Fu et al., 2011).

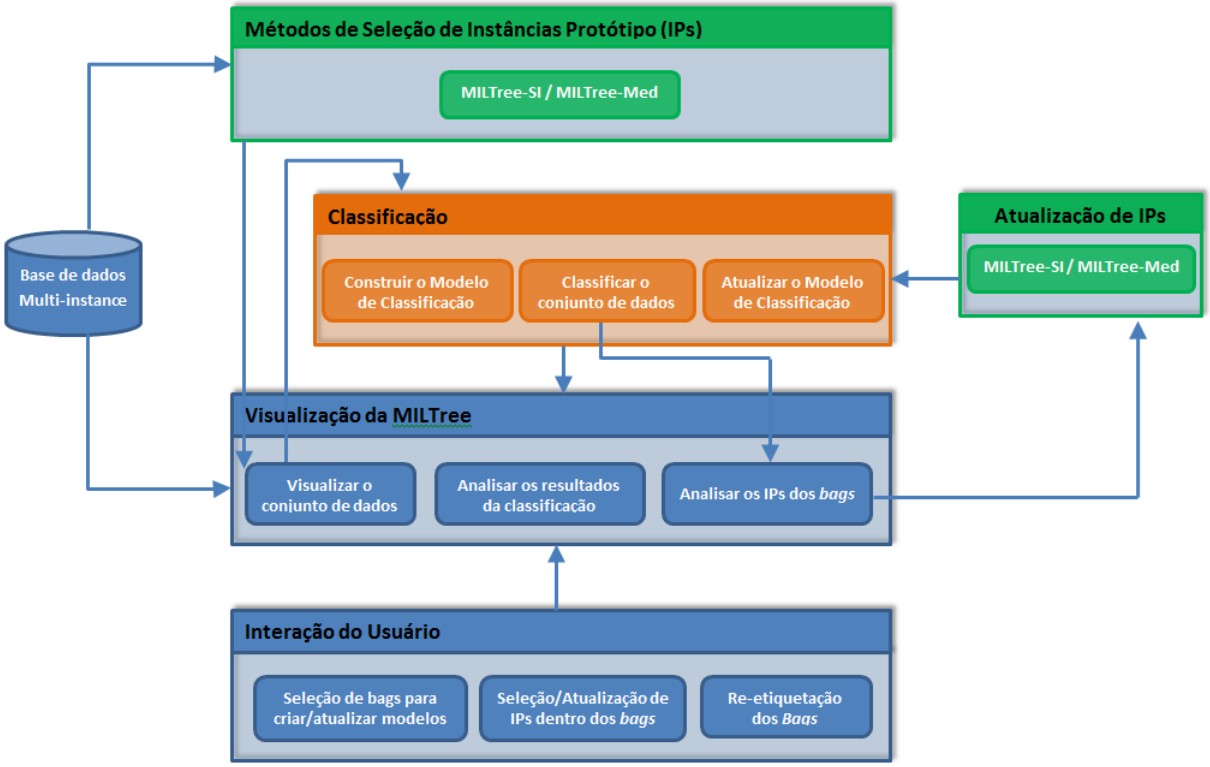
Nossa abordagem para resolver o problema de aprendizado multi-instância compreende duas características principais: uma abordagem de visualização baseada em árvore para codificar os dados multi-instâncias (incluindo as representações de *bags* e instâncias), e uma heurística para converter MIL em um problema de aprendizado de máquina padrão com base nessa visualização. Isto permite que os dados multi-instância sejam visualizados em dois espaços de projeção chamados de espaço de projeção de *bags* e espaço de projeção de instâncias usando a MILTree. Nossa abordagem identifica instâncias protótipo que são relevantes para cada *bag* e, em seguida, treina um classificador baseada nas instâncias protótipo. Essas instâncias protótipo podem ser identificados por dois métodos de seleção de instâncias: MILTree-SI ou MILTree-Med usando a visualização MILTree, proposto neste trabalho.

A Figura 3.1 apresenta o diagrama que resume o processo de classificação multi-instância usando métodos de seleção de instâncias protótipo e o layout MILTree. O componente "Seleção de Instâncias Protótipo" contem os métodos de seleção de instâncias MILTreeSI e MILTreeMed. Qualquer desses métodos pode ser utilizado para seleção das instâncias protótipo de um *bag*. O componente "Visualização da MILTree" provê o suporte visual para a projeção dos *bags* e instâncias (cada um num espaço de projeção diferente) para a compreensão dos dados multi-instância, seleção das instâncias protótipo e ajuda ao processo de classificação. O componente "Classificação" concentra o fluxo da classificação automática. O componente "Atualização de IPs" se foca na validação e atualização de instâncias protótipo através de análises feitas sobre as instâncias e *bags* usando a visualização. O componente "Interação do Usuário" provê diferentes maneiras para que o usuário possa influenciar no processo de classificação interagindo com visualizações.

Nas seções seguintes, primeiro se apresenta o layout MILTree e, em seguida, os métodos de seleção de instâncias protótipo MILTree-SI e MILTree-Med.

Notação

Usamos a mesma notação descrita na Seção 2.3. Além disso, consideramos que cada *bag* contem duas instâncias protótipo chamadas $B_{protoProj}$ e $B_{protoClass}$.



(a)

Figura 3.1: O processo de classificação multi-instância usando o layout MILTree com os métodos de seleção de instâncias protótipo MILTree-SI e MILTree-Med.

$B_{protoProj}$ representa a instância protótipo que será utilizada para mapear os *bags* no espaço de projeção de *bags* da MILTree e $B_{protoClass}$ representa a instância protótipo que será utilizada para criar o modelo de classificação multi-instância. No começo $B_{protoProj}$ e $B_{protoClass}$ têm os mesmos valores, no entanto, $B_{protoClass}$ pode ser atualizado visualmente através do layout MILTree (Seção 3.3.1 apresenta as formas de atualização).

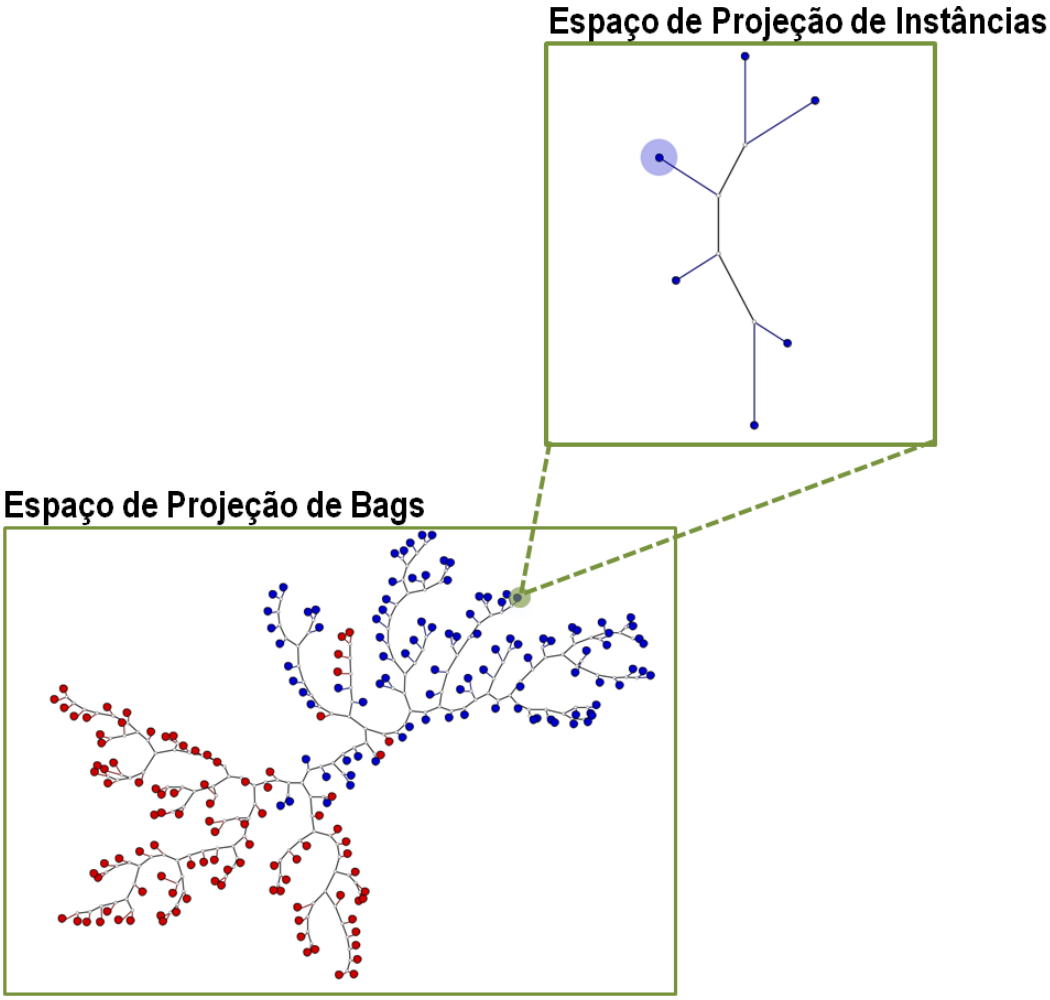
3.2 Árvore Multi-Instância (MILTree)

Uma árvore de similaridade NJ (veja Seção 2.2.2) é utilizada como base para a construção da MILTree. Com o intuito de lembrar como é construída uma árvore NJ, a seguir é descrito resumidamente o algoritmo da árvore NJ. O algoritmo NJ começa com uma árvore estrela formada por todos os m objetos da matriz de distância, representada por nós folha dispostos em uma configuração circular e conectados por ramos a um nó central. Depois, iterativamente, o par de vizinhos mais próximos entre todos os possíveis pares de nós são encontrados utilizando o critério de evolução mínima, o qual tenta minimizar a soma dos tamanhos de todos os nós da árvore. Em seguida, esse par de nós é agrupado

num novo nó interno, e as distâncias deste nó para o resto dos nós na árvore são calculadas para serem utilizadas nas iterações subsequentes. O algoritmo termina quando $n-2$ nós virtuais foram inseridos na árvore, que é quando a árvore estrela está completamente resolvida em uma árvore binária. Mais detalhes sobre a árvore NJ foram apresentados na Seção 2.2.2, assim como o algoritmo completo (Algoritmo 1).

Com o objetivo de mapear os dados multi-instância numa estrutura visual de árvore sem ter problemas de sobreposição de instâncias, dado que os conjuntos de dados multi-instância frequentemente são grandes, a MILTree foi desenvolvida como uma árvore NJ de dois níveis. Nosso objetivo é projetar os bags e as instâncias em dois diferentes níveis (dois espaços de projeção). De modo a obter isto, nós agrupamos os dados (instâncias) em *bags* previamente conhecidos. Assim, cada *bag* é composto por uma quantidade de instâncias, sendo que uma delas será utilizada para representar o *bag*. Essa instância é chamada de instância protótipo $B_{protoProj}$, e é projetada no espaço de projeção de *bags* da MILTree usando uma árvore NJ.

Na Figura 3.2 um subconjunto do Corel-1000 (descrito na Seção 5.4) é utilizado para ilustrar os espaços de projeção no *layout* MILTree. Nesse conjunto de dados cada imagem é um *bag* representado por alguns vetores de características (instâncias) que são extraídas das regiões da imagem, com uma média de 4,46 instâncias por *bag*. Na projeção do primeiro nível (espaço de projeção de *bags*) os pontos vermelhos representam *bags* positivos — 100 imagens da categoria flor — e os pontos azuis representam *bags* negativos — 100 imagens selecionadas uniformemente a partir das categorias restantes do conjunto de dados Corel-1000. Ao selecionar um *bag*, é possível visualizar as instâncias no interior do *bag*, projetada no segundo nível da árvore multi-instância MILTree. A Figura 3.3 apresenta o *bag* (imagem) selecionado na figura 3.2 e suas correspondentes instâncias (regiões segmentadas da imagem).



(a)

Figura 3.2: Espaços de projeção de bags e espaço de projeção de instâncias para um subconjunto Corel-1000 (com a instância protótipo destacado no espaço de instâncias) na MILTree usando a distância Euclidiana, com um total de 200 *bags* e 824 instâncias.

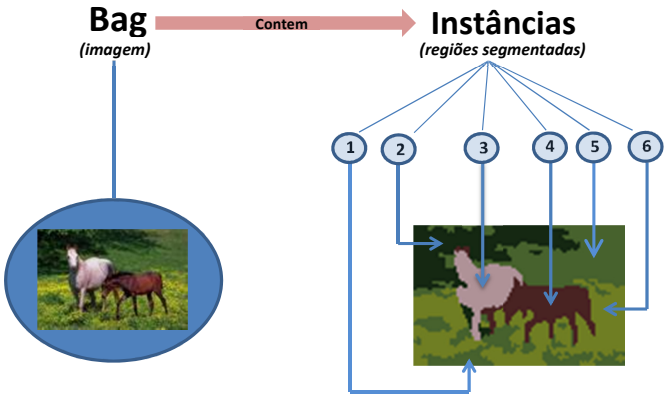


Figura 3.3: O bag selecionado na figura 3.2 e suas correspondentes instâncias.

O Algoritmo 3 inclui o procedimento completo para construir a MILTree. Ele começa por agrupar a matriz de instâncias D em uma quantidade de *bags* B_i, onde i denota o índice dos *bags*. Note que uma instância representa o vetor de características. A fim de obter esses *bags*, iteramos sobre a matriz de instâncias D, onde cada linha D_m representa a distância de uma determinada instância para todas as outras instâncias. Em cada iteração, um novo *bag* B_i é criado e todas as instâncias B_{ij} que pertencem a este *bag* são adicionadas a B_i. O índice j denota o índice da instâncias pertencente a algum *bag* B_i. Note que se conhece a priori qual instância pertence a um determinado *bag*.

Em seguida, o par de instâncias protótipo $B_{i,protoProj}$ e $B_{i,protoClass}$ são selecionadas para cada *bag* B_i usando qualquer dos métodos de seleção de instâncias protótipo MILTree-SI ou MILTree-Med propostos neste trabalho. Note que $B_{i,protoProj}$ é utilizado para construir a MILTree, e $B_{i,protoClass}$ é usado no processo de classificação. Assim, todas as instâncias protótipo $B_{i,protoClass}$ são adicionadas a um vetor T. Depois T é utilizado para criar uma nova matriz de distâncias P entre qualquer instância protótipo $B_{i,protoClass}$ e as restantes instâncias protótipo contidas no vetor, as quais representam explicitamente os *bags*.

Finalmente, essa matriz P será utilizada para criar uma árvore NJ de *bags*, a qual será chamada $B.tree$, no espaço de projeção de *bags* da MILTree.

Projeção de Instâncias: Uma vez que MILTree é uma árvore de dois níveis, quando um usuário acessa um *bag*, no intuito de explorá-lo ou fazer outro tipo de análise, como, por exemplo, a atualização de instâncias protótipo, as instâncias B_{ij} pertencentes ao *bag* B_i são projetadas numa nova árvore NJ chamada $B_i.tree$, criando um novo espaço de projeção de instâncias. Para criar essa nova árvore NJ são necessárias as instâncias B_{ij}, as quais serão projetadas no novo espaço, e a instância protótipo $B_{i,protoProj}$, a qual é utilizada para enlaçar (comunicar) o espaço de *bags* e o novo espaço de instâncias criado.

Promoção da árvore NJ: Para a promoção dos nós da árvore NJ é utilizado o algoritmo de promoção de nós desenvolvido por Paiva et al. (2011), esse algoritmo baseia-se na substituição de um nó virtual por uma folha sempre que uma configuração de nós aconteça. A promoção de nós é detalhada na Seção 2.2.2. Neste trabalho a decisão de promover os nós da árvore no espaço das instâncias e no espaço dos bags depende do usuário. É recomendável promover os nós só quando projetar grandes quantidades de

bags, porque quando o conjunto de *bags* é pequeno a visualização da árvore NJ é suficiente e reflete o comportamento dos dados sem poluir excessivamente a visualização.

Algoritmo 3: ALGORITMO MILTREE

Entrada: Matrix de Similaridade D, Vetor de tamanhos dos *bags* T.
Saída: Árvore Multi-Instância MILTree.
início
  //Criação de *bags*.
  para $m = 0$, $i = 0$ *to* $m < D.size$ **faça**
    Criar o bag B_i.
    $B_i.size = T_i$.
    para $j = 0$ *to* $B_i.size \in D$ **faça**
      $B_{ij} = D_{m0}$. // Onde $j = 0, 1, 2, ..., B_i.size$;
      Agregar B_{ij} a B_i. Onde $B_{ij} \in D_n$, $i = idBag$, $j = idInstances$.
      Conectar B_i como pai da instância B_{ij};
      $m = m + 1$; //Pegar a seguinte instância.
    fim
    $i = i + 1$; //Criar o seguinte *bag*.
  fim
  //Calculando instâncias protótipo.
  para $i = 0$ *to* $B.size$ **faça**
    Selecionar $B_{i,protoProj}$ usando MILTree-Med ou MILTree-SI.
    Selecionar $B_{i,protoClass}$ usando MILTree-Med ou MILTree-SI.
    //O Métodos de seleção de instancias protótipo MILTree-Med e MILTree-SI são apresentados na Seção3.3.
    Agregar $B_{iprotoProj}$ a P. //P é um vetor de instâncias protótipo $B_{iprotoProj}$.
  fim
  //Projetar *bags* no espaço de projeção de *bags* do MILTree.
  $B.tree = NJTree(B, P)$.
  Realizar a promoção de nós da árvore $B.tree$;
fim
retorna *Árvore MILTree.*

3.3 Métodos de Seleção das instâncias Protótipo

Neste trabalho, dois novos métodos de seleção de instâncias protótipo são apresentados, chamados MILTree-SI e MILTree-Med. Estes métodos são baseados naqueles propostos por Yuan et al. (2012) e Zhou e Zhang (2007b) (descritos na Seção 2.3).

Ambos os métodos SI e Med têm como objetivo computar duas instâncias protótipo por *bag*: B_{ix} e B_{iy}. A primeira instância protótipo, B_{ix}, é utilizada tanto para a visualização como para a construção do modelo de classificação. Posteriormente, aqueles *bags* que são erroneamente representados pela instância protótipo x, podem ter seu protótipo alterado para y, atualizando o modelo de classificação de múltiplas instâncias para um modelo melhor.

MILTree-SI: Este método está baseado na estratégia de seleção de instâncias salientes (Yuan et al., 2012), o qual computa uma instância positiva *ótima* apenas para *bags* positivos, e assume que os bags negativos somente têm instâncias negativas. MILTree-SI, no entanto, assume que os *bags* negativos podem ter instâncias positivas e negativas, o que acreditamos que é um cenário mais comum considerando dados reais complexos, por exemplo, texto e imagens. Assim, uma instância negativa *ótima* para cada *bag* negativo também é calculada em nossa abordagem.

A fim de encontrar a instância negativa *ótima* para cada *bag* negativo, executamos o seguinte processo baseados na estratégia de instâncias salientes (Yuan et al., 2012) (ver Seção 2.3.1.3): Primeiro, o valor de saliência para cada instância dentro dos *bags* negativos é calculado utilizando a equação 2.9, onde $Sal(B_{ij}^{-})$ representa o valor de saliência para uma instância B_{ij}^{-} dentro do *bag* negativo B_{i}^{-} . Em seguida, duas instâncias: uma instância como um valor de saliência máxima $Sal(B_{im}^{-})$ e outra instância com um valor de saliência mínima $Sal(B_{i1}^{-})$, são escolhidas para cada *bag* negativo. O índice m representa o número de instâncias dentro de um *bag*, B_{im}^{-} representa a instância com maior valor de saliência e B_{i1}^{-} representa a instância com menor valor de saliência. Depois, selecionamos a instância mais negativa para cada *bag*, a qual será chamada de instância negativa *verdadeira*. Essa instância pode ser a instância B_{im}^{-} ou a instância B_{i1}^{-}. Para obtê-la, utilizamos a equação 2.11. As equações 2.9 e 2.11, respectivamente, estão incluídos aqui novamente (mas no contexto de *bags* negativos) para fins de clareza:

$$Sal(B_{ij}) = \sum_{B_{ik} \in B_i \setminus \{B_{ij}\}} d(B_{ij}, B_{ik}).$$

$$D(B_{ij}, B^{-}) = \min_{B_{rt} \in B^{-}} d(B_{ij}, B_{rt}),$$

Depois, uma única instância entre todas as instâncias negativas *verdadeiras* identificadas previamente é escolhida como a instância negativa mais representativa entre todos os *bags* negativos, o qual será chamado de instância negativa *global*. Note que essa

instância está o mais longe possível das instâncias dos *bags* positivos B^+. Em seguida, uma instância positiva *global* é identificada utilizando a instância negativa *global* identificada previamente. A instância positiva *global* é uma instância que está o mais longe possível da instância negativa *global*, ou seja, a mais positiva instância entre todas as instâncias dos *bags* positivos.

Finalmente, com a instância positiva *global* já calculada, MILTree-SI seleciona duas instâncias protótipo com os mais altos valores de saliência $Sal(B_{ij}^-)$ para cada *bag* negativo, os quais estão o mais longe possível da instância positiva *global*. Para ilustrar o processo de seleção de uma instância protótipo dentro dos *bags* negativos, na Figura 3.4 se apresenta um conjunto de *bags* positivos B^+ e *bags* negativos B^-. Como se observa na Figura 3.4 a instância protótipo dentro de um *bag* negativo será a instância negativa que esteja o mais longe possível da instância positiva *global*.

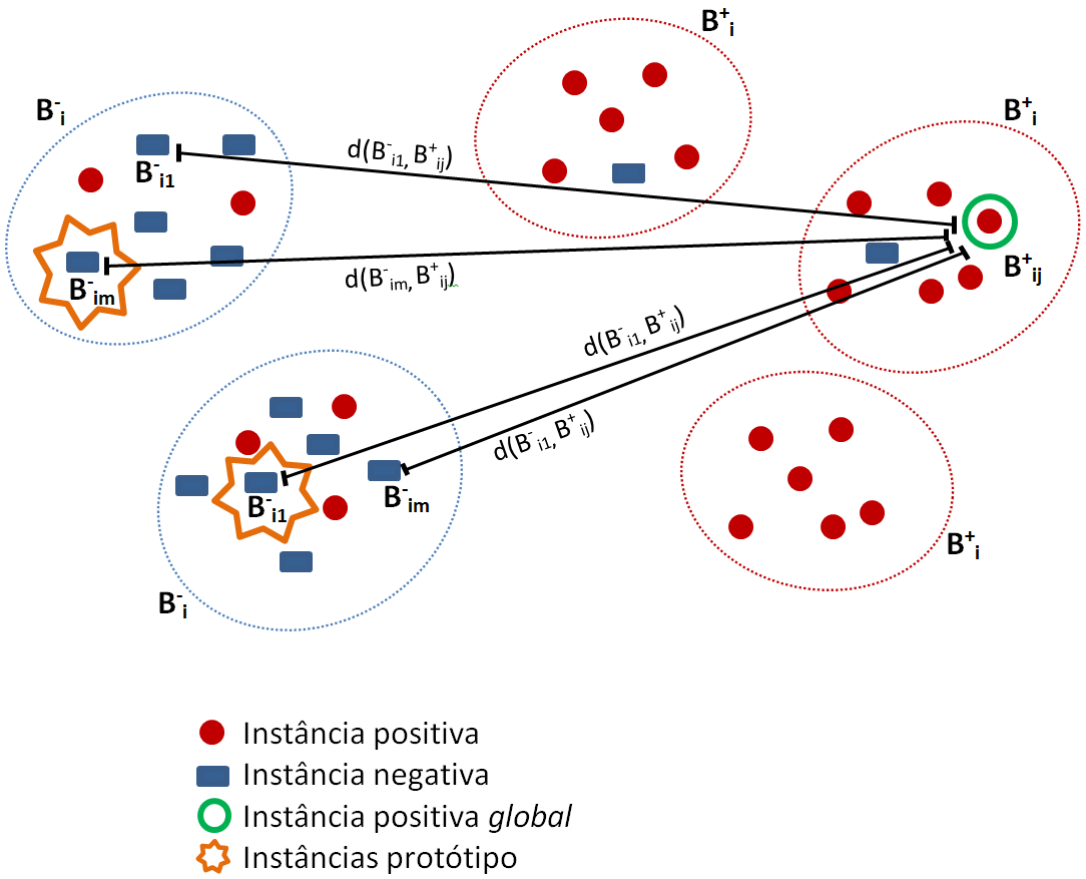


Figura 3.4: Seleção de uma instância protótipo dentro dos *bags* negativos utilizando o método de seleção de instâncias protótipo MILTree-SI.

Assim, MILTree-SI calcula as instâncias protótipo B_{ix} e B_{iy} para *bags* positivos e para *bags* negativos. Note que as instâncias protótipo B_{ix} e B_{iy} são similares porque B_{ix} é a instância com maior valor de saliência e B_{iy} é a segunda instância com maior valor de saliência.

MILTree-Med: Algoritmos de agrupamento já foram utilizados no contexto MIL no espaço de *bags* e no espaço de instâncias, com o intuito de criar um novo vetor de características que represente um *bag* e em seguida criar um classificador usando esses vetores. Alguns deles foram apresentados na Seção 2.3.1.2.

Diferente dos métodos apresentados por esses trabalhos, MILTree-Med tenta identificar uma instância real dentro do *bag* que melhor represente o *bag*, ao invés de criar uma nova. MILTree-Med trabalha no espaço das instâncias de cada *bag* ao invés de trabalhar no espaço dos *bags* ou no espaço de instâncias de todos os *bags*. Acredita-se que a informação de uma instância real dentro do *bag* contem informação suficiente para representar o conceito do *bag* à qual pertence.

MILTree-Med assume que os bags podem conter instâncias positivas e negativas, e que o rótulo delas não são conhecidos (no contexto MIL os rótulos das instâncias não são conhecidas, só os rótulos dos *bags*). Uma vez que MILTree-Med quer obter duas instâncias protótipo por cada *bag*, ele começa agrupando as instâncias pertencentes a um *bag* em dois sub-grupos utilizando o método de agrupamento *K-Medoids*. A lógica por trás desse procedimento é que um *bag* pode conter instâncias positivas e negativas e, portanto, tentamos separá-los em um grupo formado por instâncias positivas e um grupo de instâncias negativas. Na Figura 3.5 se apresenta um *bag*, o qual pode ser positivo ou negativo, dividido em dois grupos (*clusters*).

Como MILTree-Med tenta selecionar instâncias reais de cada *bag* como as instâncias protótipo, os medoids dos sub-grupos serão as instâncias protótipo B_{ix} e B_{iy} do *bag* B_i. Porém, o *bag* precisa escolher uma delas para inicializar $B_{i,protoClass}$ e $B_{i,protoProj}$. Nesse caso, a instância protótipo B_{ix} ou B_{iy} que esteja mais perto do medoid do *bag* (nesse contexto, o *bag* é considerado um grupo) é escolhido para inicializar $B_{i,protoClass}$ e $B_{i,protoProj}$. Como se apresenta na Figura 3.5 o medoid que está mais perto do centroide c do *bag* é o medoid m_1, o qual será considerado como a instância protótipo B_{ix} desse *bag*. O medoid m_1 (instância) também inicializa o $B_{i,protoClass}$ e $B_{i,protoProj}$. Entre tanto, o medoid m_2 é considerado a segunda instância protótipo B_{iy} do *bag*. Note que as instâncias protótipo B_{ix} e B_{iy} são muito diferentes porque B_{ix} contem informações da instância representativa do grupo positivo e B_{iy} da instância representativa do grupo negativo.

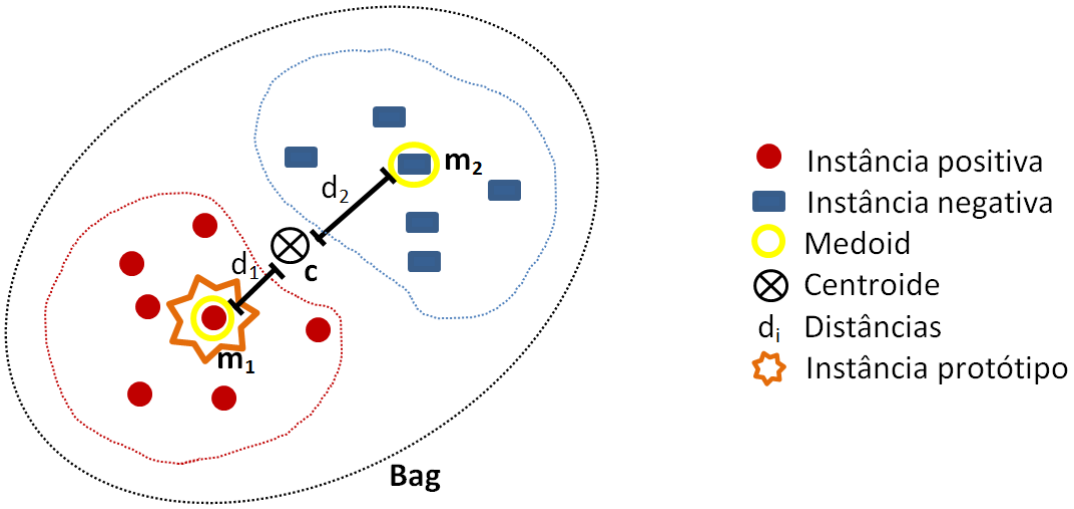


Figura 3.5: Seleção da instância protótipo B_{ix} dentro de um *bag* utilizando o método de seleção de instâncias protótipo MILTree-MED.

3.3.1 Atualização das Instâncias Protótipo usando MILTree

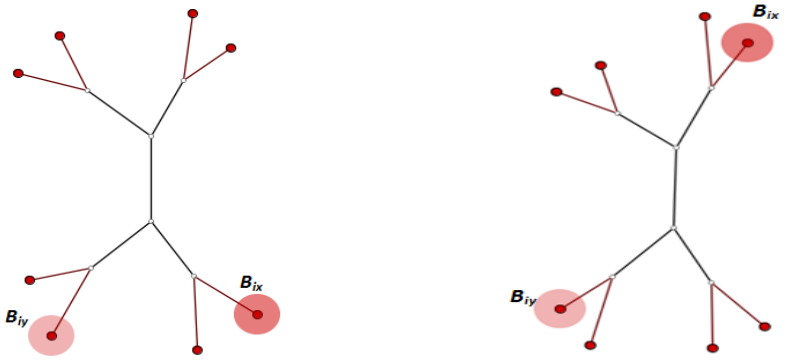
A fim de criar o layout MILTree, bem como o classificador, primeiro utilizamos a primeira instância protótipo $B_{i,x}$ para inicializar $B_{i,protoClass}$ e $B_{i,protoProj}$. Isso ocorre porque o protótipo x é considerado pelos métodos MILTree-SI e MILTree-Med as instâncias mais representativas de cada *bag*.

Como foi mencionado na Seção 2.4, a inserção do usuário na análise e exploração dos dados pode ajudar na criação de um modelo de classificação melhor. No contexto deste trabalho, a MILTree foi desenvolvida para que o usuário possa identificar uma seleção errada das instâncias protótipo dos *bags*. Assim, os usuários podem então definir B_{iy} ou outra instância como a mais representativa de um bag B_i segundo seu conhecimento do problema. Por esta razão, incluímos o usuário no processo de atualização de instância protótipo usando MILTree.

Duas representações visuais podem ser utilizadas pelo usuário:

1. Destaque das Instâncias Protótipo: MILTree destaca as instâncias $B_{protoClass}$ e também B_{iy} no espaço de instâncias, as quais são duas alternativas de instâncias protótipo. Assim, observando no espaço de instâncias a instância protótipo selecionada pelo algoritmo de seleção de instância (MILTree-SI ou MILTree-Med), o usuário pode validar $B_{protoClass}$ ou atualizá-lo de acordo com seu conhecimento selecionando B_{iy}, ou até mesmo outra instância projetada no espaço de instâncias. A Figura 3.6 mostra as instâncias protótipo B_{ix} e B_{iy} projetados no espaço de projeção de instâncias da

MILTree. Na Figura 3.6a o método de seleção de MILTree-SI é utilizado, e na Figura 3.6b é usado o método de seleção de medoids MILTree-Med.



(a) Método de seleção de instância saliente (MILTree-SI).

(b) Método de seleção de instância medoids (MILTree-Med).

Figura 3.6: Métodos propostos para a seleção das instâncias protótipo B_{ix} e B_{iy}. Ambos (a) e (b) projetam as mesmas instâncias a partir de um *bag* positivo B_i^+ do conjunto de dados MUSK1 (ver a Seção 5.3) no espaço de projeção de instâncias da MILTree.

2. Árvores InstancePrototype ClassMatch: A segunda forma com a qual o usuário pode atualizar $B_{protoClass}$ é através da utilização da árvore *InstancePrototypes ClassMatch*. Essa árvore usa a cor para contrastar os *bags* que foram erroneamente classificados, considerando só o conjunto de treinamento de *bags* para os quais são conhecidos os rótulos. Este método foi utilizado com sucesso em (Paiva et al., 2015) e (Paiva et al., 2011). Nesta abordagem, $B_{protoClass}$ é usado para construir um classificador SVM. Em seguida, a MILTree gera um layout da árvore com os *bags* classificados e coloridos de acordo com o resultado da classificação. Uma vez que as etiquetas são conhecidas, é possível visualizar os erros de treinamento, e o usuário pode então alterar os protótipos de instância para aqueles *bags* erroneamente classificados e que são visualmente distinguíveis na árvore *InstancePrototypes ClassMatch*. A Figura 3.7a apresenta um subconjunto da base de dados Corel-1000 projetada no espaço de *bags* de MILTree, onde os *bags* vermelhos representam *bags* positivos (imagens de cavalos) e os *bags* azuis representam *bags* negativos (imagens a partir de outras categorias do Corel-1000). A fim de encontrar a árvore *InstancePrototypes ClassMatch* para este conjunto de dados, primeiro selecionamos um conjunto de treinamento para criar um classificador SVM. A Figura 3.7b mostra o conjunto de treinamento usado para criar o classificador, onde *bags* vermelhos representam o conjunto de treinamento e *bags* azuis representam conjunto de dados de teste. Finalmente, baseados no resultado

da classificação usando o classificador criado anteriormente é projetada a árvore *InstancePrototypes ClassMatch* para o conjunto de dados de treino. A Figura 3.7c mostra a árvore *InstancePrototypes ClassMatch*, onde os *bags* vermelhos representam os *bags* com uma instância protótipo inadequada. Os resultados apresentados nos Capítulos 4 e 5 mostram a eficácia deste método de atualização no processo de classificação de múltiplas instâncias.

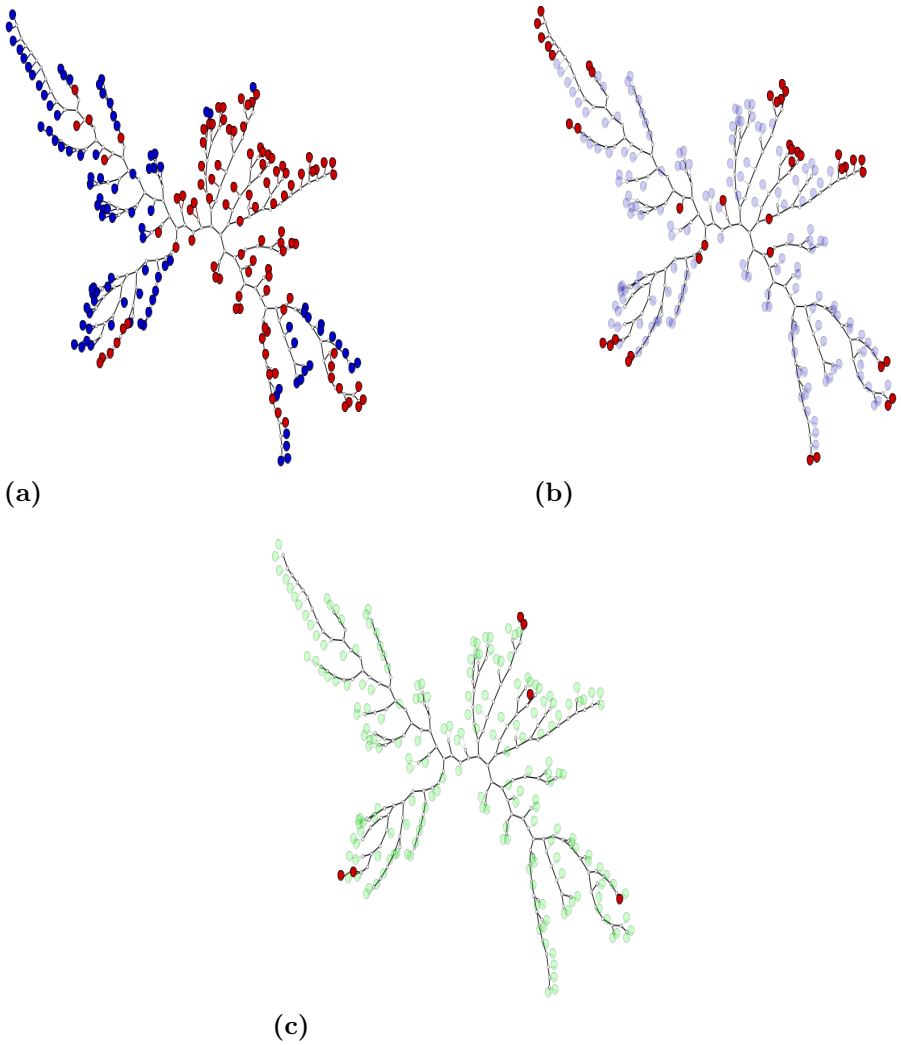


Figura 3.7: Espaço de projeção de *bags* da MILTree para um sub-conjunto do conjunto de dados Corel-1000 (100 imagens da categoria Horse e 100 imagens selecionadas uniformemente das demais categorias do conjunto de dados), com a projeção do *ground truth* (a), o conjunto de amostras selecionadas para treinamento (b) e a árvore *InstancePrototypes ClassMatch*(c). Todas as árvores MILTree geradas em (a),(b) e (c) usam a distância Euclidiana.

3.4 Considerações Finais

Neste Capítulo foi apresentada uma nova abordagem para resolver problemas de aprendizado multi-instâncias, o qual combina a visualização de dados multi-instâncias usando MILTree com novos métodos de seleção de instâncias protótipo a fim de obter melhores resultados no processo de classificação de dados multi-instâncias. Avaliamos esta abordagem através de casos de estudo (no próximo Capítulo) e também com resultados experimentais (Capítulo 5).

Capítulo

4

Aplicação da Árvore MILTree para Aprendizado Multi-Instâncias

4.1 Considerações Iniciais

A MILTree e os métodos de seleção das instâncias protótipo MILTree-Med e MILTree-SI permitem ao usuário analisar os conjuntos de dados multi-instância utilizando a visualização das similaridades entre os *bags*, cada uma delas representada por uma instância protótipo, a qual foi encontrada utilizando MILTree-Med ou MILTree-SI. Com o uso da árvore *InstancePrototypes ClassMatch*, é possível visualizar os *bags*, dentro do conjunto de treinamento, que têm instâncias protótipo que não são as adequadas.

Nesse contexto, as intâncias protótipo que representam *bags* erroneamente classificadas podem ser alteradas pelo usuário. Esses *bags* podem corrigir ou atualizar suas instâncias protótipo, a partir do espaço de *bags*, usando as instâncias protótipo alternativas, as quais foram definidas utilizando os métodos de seleção MILTree-Med ou MILTree-SI. O usuário também pode explorar visualmente os *bags* no intuito de visualizar as instâncias desse *bag* no espaço de instâncias da MILTree para corrigir ou atualizar manualmente a instância protótipo de cada *bag*. Note que, no espaço de instâncias da MILTree são mostradas as instâncias protótipo alternativas que são calculados automaticamente e, segundo a análise do usuário, ele pode escolher a instância alternativa ou selecionar manualmente outra instância qualquer.

Além disso, MILTree permite visualizar o processo completo de classificação, desde a criação e ajuste do conjunto de treinamento até a visualização dos resultados da classificação. Além disso, permite incluir novos *bags* no modelo de classificação multi-instância que contribuem para melhorar o modelo e obter melhores resultados.

Assim, a árvore MILTree representa uma ferramenta útil para explorar e analisar visualmente os dados multi-instâncias, com o intuito de melhorar a classificação de dados multi-instância. Na próxima Seção, três casos de estudo são apresentados para ilustrar a utilidade prática da MILTree, e os métodos de seleção MILTree-Med e MILTree-SI na tarefa de classificação de dados multi-instância.

4.2 Casos de Estudo

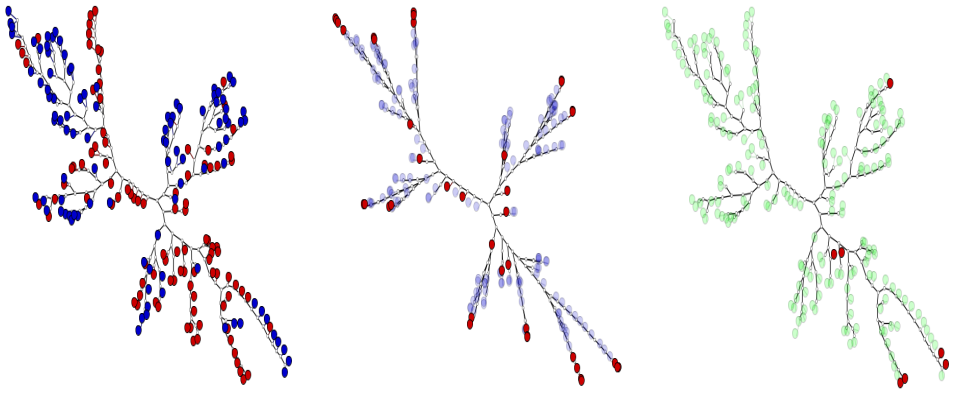
Os casos de estudo foram realizados num computador Dell Z620 equipado com um núcleo CPU Intel (E5-2690, 3.40 GHz) e 16 GB de memória. No primeiro caso de estudo foi utilizado um conjunto de dados para estudar o cenário de classificação binária de dados multi-instância. O conjunto de dados é chamado de Corel-1000 People, que inclui 100 imagens da categoria People do conjunto de dados Corel-1000, e mais 100 imagens que são selecionados aleatoriamente a partir de todas as outras categorias do conjunto de dados Corel-1000.

Para o segundo caso foi utilizado o conjunto de dados multiclasse Corel-300, que contem cinco classes. Finalmente, no terceiro caso de estudo usamos o conjunto de dados Musk1 que é um conjunto de dados de referência para métodos MIL. Mais informações sobre cada conjunto de dados pode ser encontrada na Seção 5.

4.2.1 Caso de Estudo 1: Espaço de Projeção de Instâncias para um Problema de Classificação Binária

No primeiro caso de estudo testamos nossa proposta usando a categoria People do conjunto de imagens Corel-1000. Este conjunto de dados binário tem 200 *bags* (imagens) e 938 instâncias. A Figura 4.1a apresenta a projeção do conjunto de dados People no espaço de projeção de *bags* da MILTree, onde *bags* vermelhos representam *bags* positivos (imagens de pessoas) e *bags* azuis representam *bags* negativos (imagens de outras categorias). Neste caso de estudo, pretende-se mostrar como a seleção correta das instâncias protótipo pode influenciar na precisão do modelo de classificação multi-instância. O layout MILTree é utilizado para a projeção e MILTree-Med como método de seleção das instâncias protótipo.

A correção ou atualização das instâncias protótipo é realizada no espaço de projeção de instâncias.



(a) **(b)** **(c)**

Figura 4.1: Espaço de projeção de *bags* da MILTree para a Categoria People do conjunto de dados Corel-1000, com a projeção do seu *ground truth*(a), o conjunto de amostras selecionadas para treinamento (b) e a árvore *InstancePrototypes ClassMatch* (c). Todas as árvores MILTree geradas em (a),(b) e (c) usam a distância Euclidiana.

Começamos o processo de classificação selecionando 20% dos *bags* como o conjunto de treinamento, enquanto o restante 80% de *bags* será o conjunto de teste.

Devido à natureza da árvore NJ, MILTree posiciona os *bags* que melhor caracterizam a classe à qual pertencem longe do núcleo da árvore (*bags* externos). Por outro lado, a MILTree posiciona os *bags* que representam aqueles cujas características não se encaixam bem em qualquer classe, ou que se encaixam em mais do que uma classe, próximos do núcleo da árvore (*bags* internos). Nós selecionamos alguns *bags* externos e internos para criar nosso conjunto de dados de treinamento. Assim, com base nesse conjunto de treinamento será criado um classificador que não é demasiadamente restritivo e também não é muito geral. Esta estratégia de seleção do conjunto de treinamento foi já mencionada em Paiva et al. (2015), onde é ressaltado que a combinação de *instâncias* externas e internas (nosso casso *bags* externos e internos) como conjunto de treinamento vai gerar melhores resultados na classificação.

A Figura 4.1b apresenta o conjunto de treinamento selecionado para conjunto de dados binário People, onde *bags* vermelhos representam o conjunto de treinamento e *bags* azuis representam o conjunto de dados de teste. Note-se que em MIL, o conjunto de treinamento é um conjunto de *bags*, e por isso, quando o usuário seleciona um *bag*, o que ele está realmente selecionando é a instância protótipo chamada $B_{protoClass}$ (considerada a instância mais representativa desse *bag*). Depois que o usuário selecionou o conjunto de treinamento, um primeiro modelo de classificação multi-instância é criado por meio de

um classificador SVM padrão. O modelo é baseado no conjunto de treino previamente selecionado, a fim de detectar possíveis *bags* com uma instância protótipo inadequada.

Quando o modelo é aplicado sobre os dados multi-instância a árvore *InstancePrototypes ClassMatch* exibe, em cores contrastantes, os *bags* que foram erroneamente classificados apenas no conjunto de treinamento, a fim de identificar os *bags* com um protótipo inadequado. A Figura 4.1c apresenta a árvore *InstancePrototypes ClassMatch* para o conjunto - People, onde os *bags* vermelhos representam os *bags* com uma instância protótipo inadequada no conjunto de treinamento.

Com o objetivo de corrigir ou atualizar a instância protótipo desses *bags* e melhorar o modelo de classificação multi-instância, exploramos o espaço de projeção de instâncias de cada *bag* vermelho (fig. 4.1c). Isto é possível, uma vez que o layout MILTree é uma árvore NJ de dois níveis (espaço de projeção de instâncias e espaço de projeção de *bags*).

Os usuários têm duas opções para corrigir ou atualizar as instâncias protótipo dos *bags* identificados na etapa anterior. A primeira opção permite ao usuário atualizar a instância protótipo $B_{protoClass}$ de todos os *bags* identificados na árvore *InstancePrototypes ClassMatch* por as instâncias protótipo B_{iy}.

A segunda opção, escolhida para a realização deste caso de estudo, é fazer uma exploração visual em cada *bag* classificado erroneamente, os quais foram evidenciadas pela árvore *InstancePrototypes ClassMatch*.

Na Figura 4.2, os rótulos A, B, C, D, E, F e G representam os *bags* vermelhos da árvore *InstancePrototypes ClassMatch*. Quando cada projeção do espaço de instâncias é explorada, é possível ver uma nova árvore NJ formada por instâncias que pertencem ao *bag* explorado. Essas instâncias são mostradas como pontos. Os pontos verdes e vermelhos representam instâncias corretamente classificadas e erroneamente classificadas, respectivamente, no conjunto de treinamento. De acordo com nossos experimentos, quando as instâncias protótipo foram corretamente classificadas se obtem um modelo de classificação mais robusto. Com base nessa premissa, foram atualizadas algumas instâncias protótipo no espaço de projeção das instâncias. Para isso, conforme exibido na Figura 4.2, seguimos quatro etapas. Na **primeira etapa**, os protótipos instância atuais (B_{ix}) são destacados (com a iluminação da instância) para diferenciá-los do resto e identificar instâncias protótipo corretamente classificadas. As instâncias dos *bags* “D”, “E”, “F” e “G” não tem nenhuma instância classificada corretamente, por isso suas instâncias protótipo não são atualizadas. No entanto, dentro dos *bags* “A”, “B” e “C” pode-se encontrar algumas instâncias elegíveis a se tornar a nova instância protótipo do *bag*. Na **segunda etapa**, a instância protótipo B_{iy} de cada *bag*, selecionadas por MILTree-Med, são mostradas. Isto irá ajudar os usuários a não escolherem aleatoriamente a nova instância

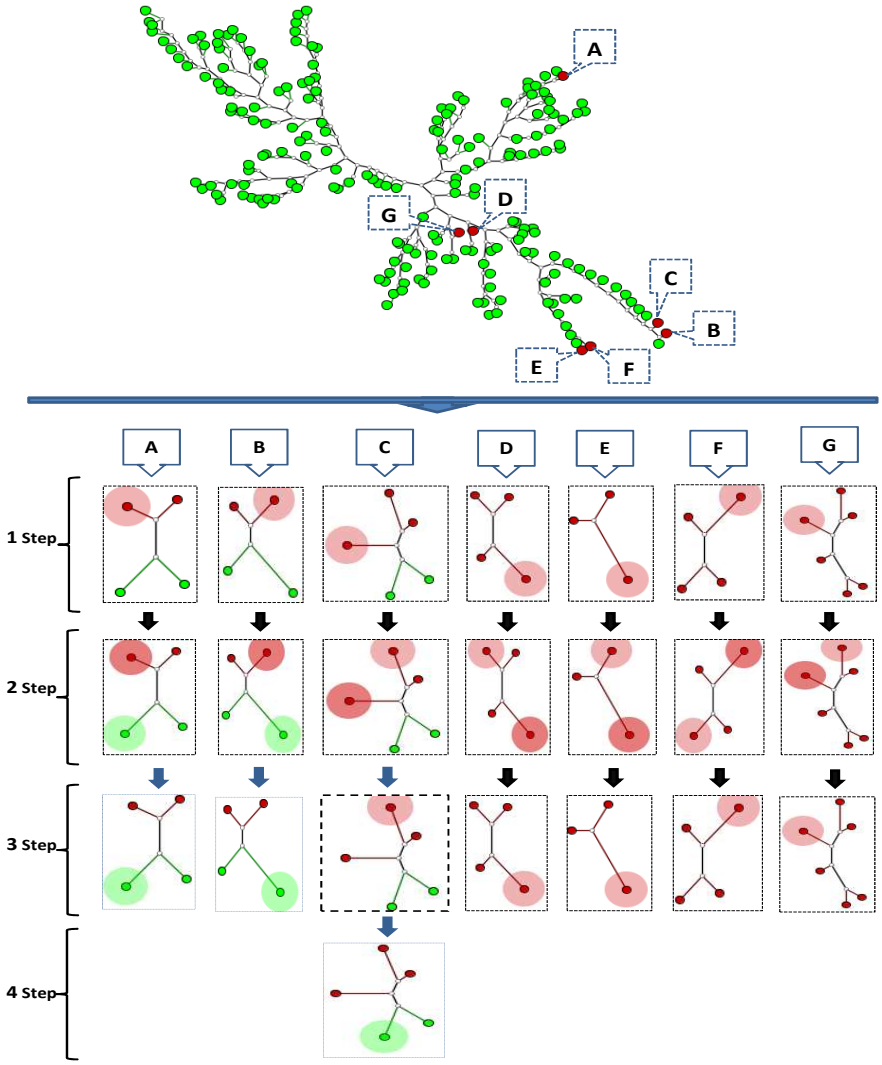


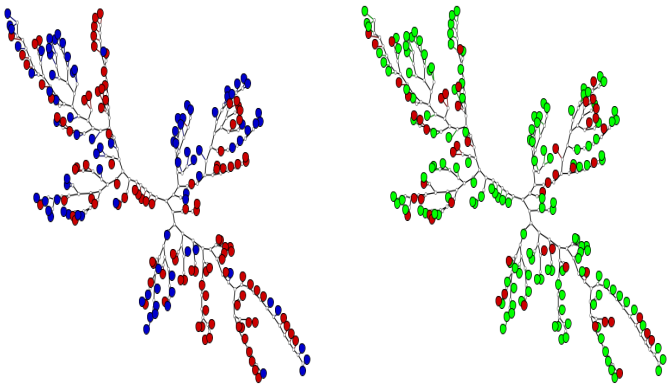
Figura 4.2: Árvore *InstancePrototypes ClassMatch* e os espaço de projeção de instâncias para cada *bag*, os quais têm uma instância protótipo inadequada. Os rótulos A, B, C, D, E, F e G representam os *bags* vermelhos. Todas as árvores usam a distância Euclidiana.

protótipo. Na **terceira etapa**, as novas instâncias protótipo dos *bags*"A", "B" e "C" são mostradas. Dentro dos *bags* "A" e "B", suas instâncias protótipo B_{iy} foram corretamente classificadas, e portanto estas são as melhores instâncias elegíveis. No entanto, no *bag* "B", a instância protótipo B_{iy} foi erroneamente classificada, e por isso, esta não seria a melhor opção de instância elegível como instância protótipo.

Devido a esse tipo de ocorrência, é realizada uma **quarta etapa**, onde se selecionou manualmente a instância protótipo dentro do *bag* "C". Os usuários poderiam escolher entre as duas instâncias corretamente classificadas (instâncias de cor verde), mas de acordo com as experiências realizadas, os melhores resultados são obtidos se a nova instância protótipo

está mais próxima da instância protótipo original. Assim, para este caso de estudo, foram corrigidas e atualizadas as instâncias protótipo de três *bags* ("A", "B" e "C").

Finalmente, depois de corrigir ou atualizar as instâncias protótipo dos *bags* detectados através da árvore *InstancePrototypes ClassMatch*, o modelo de classificação é reconstruído utilizando as novas instâncias protótipo dos *bags* do conjunto de treinamento. A Figura 4.3a apresenta, no espaço de projeção de *bags* da MILTree, o conjunto de dados binário People já classificadas e a Figura 4.3b apresenta sua correspondente árvore *ClassMatch*, onde os pontos verdes e vermelhos representam *bags* classificados corretamente e erroneamente, respectivamente. A precisão obtida para o conjunto de dados binário People sem realizar a correção ou atualização de instâncias protótipo foi 72,2%, e a precisão alcançada após corrigir ou atualizar as instâncias protótipo de apenas três *bags* foi de 75%. Isso demonstra o forte impacto de uma correta seleção de protótipos de instância sobre a criação do modelo de classificação multi-instância. Este caso de estudo também demonstra que a exploração visual no espaço de projeção de instâncias do layout MILTree desempenhou um papel crucial para produzir uma seleção satisfatória de instâncias protótipo dos *bags*.



(a) Resultados da Classificação.

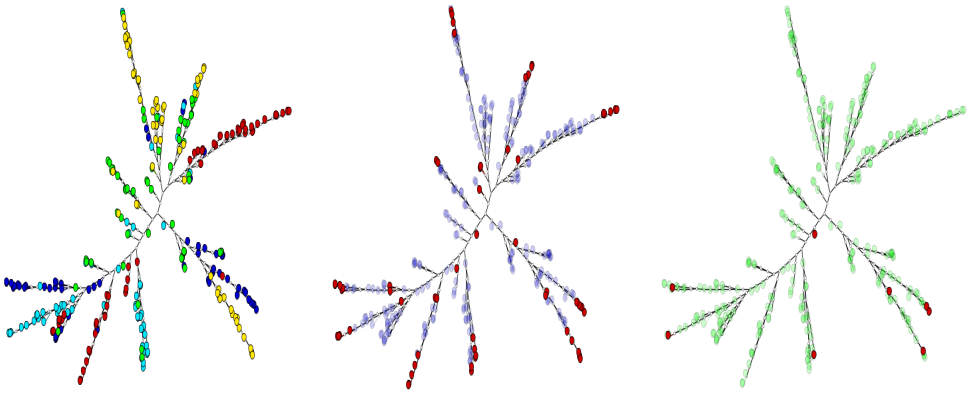
(b) Árvore *ClassMatch*.

Figura 4.3: Resultado da classificação no espaço de projeção dos *bags* da MILTree para a Categoria People do conjunto de dados Corel-1000 usando um modelo de classificação com novas instâncias protótipo (a) e sua correspondente árvore *classMatch* (b). Ambas as árvores usando a distância Euclidiana.

4.2.2 Caso de Estudo 2: Espaço de Projeção de Bags e um Problema de Classificação Multiclasse

No segundo caso de estudo, testamos nossa proposta usando o conjunto de dados multiclasse Corel-300. Este conjunto de dados multiclasse é formado por 300 *bags* e 1.293

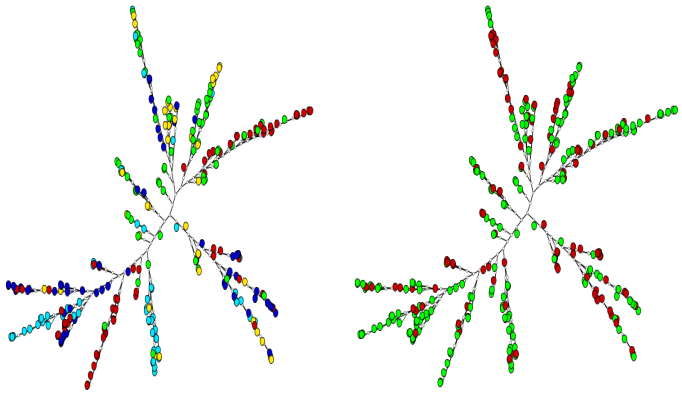
instâncias. A Figura 4.4a apresenta a projeção do conjunto de dados Corel-300 no espaço de projeção de *bags* da MILTree, onde os *bags* são representados como pontos e a cor representa a classe. Este conjunto de dados tem cinco classes. Neste caso de estudo, pretende-se mostrar o impacto da adição de instâncias protótipo B_{iy} de alguns *bags*, que pertencem ao conjunto de treinamento original, sobre o desempenho do modelo de classificação multi-instância. Nós usamos MILTree para a projeção e MILTree-SI como método de seleção das instâncias protótipo. A atualização do conjunto de treinamento é realizado no espaço de projeção de *bags*.



(a) Ground truth.

(b) Conjunto de treinamento selecionado.

(c) Árvore *InstancePrototypes ClassMatch*.



(d) Resultados da classificação.

(e) Árvore *ClassMatch*.

Figura 4.4: Espaço de projeção de *bags* em MILTree para Corel-300, usando a distância Euclidiana. Visualização do processo de classificação desde a seleção do conjunto de treinamento (b) utilizando a visualização de *ground truth* do conjunto de dados (a), a identificação de *bags* com instâncias protótipo inadequadas (c), até a visualização do resultado da classificação final (d) e seu correspondente árvore *ClassMatch* (e).

Como foi feito para o primeiro caso de estudo, primeiramente selecionamos um conjunto de treinamento usando MILTree a fim de criar um primeiro modelo de classificação

que vai ser útil para identificar os *bags* com instâncias protótipo inadequadas. A Figura 4.4b mostra o conjunto de treinamento selecionado para conjunto de dados Corel-300, onde *bags* vermelhos representam o conjunto de treinamento e *bags* azuis representam conjunto de dados de teste.

Depois, o primeiro modelo criado anteriormente é utilizado para classificar Corel-300, e em seguida é exibida a árvore *InstancePrototypes ClassMatch*, a fim de identificar os *bags* com uma instância protótipo inadequada. A Figura 4.4c apresenta a árvore *InstancePrototypes ClassMatch* de Corel-300, onde *bags* vermelhos representam *bags* com uma instância protótipo inadequada. Em seguida, todos os *bags* vermelhos são selecionados e suas instâncias protótipo $B_{protoClass}$ representados por B_{ix} são atualizados por B_{iy}, e em seguida, estas novas instâncias protótipo B_{iy} são adicionadas ao modelo de classificação previamente desenvolvido. Note-se que neste caso de estudo, novas instâncias protótipo são adicionadas ao modelo desenvolvido anteriormente ao invés de serem substituídas. Isto é feito porque as instâncias protótipo B_{ix} e B_{iy} selecionadas pelo método MILTree-SI são semelhantes. Diferente das instâncias protótipo B_{ix} e B_{iy} selecionadas pelo método MILTree-Med as quais são completamente diferentes, devido ao processo de agrupamento de instâncias positivas e negativas realizado nesse método.

Finalmente, após a atualização do modelo de classificação utilizando a árvore *InstancePrototypes ClassMatch* no espaço de projeção de *bags* da MILTree, esse modelo é usado para classificar o conjunto de dados multiclasse Corel-300. A Figura 4.4d apresenta, no espaço de projeção de *bags* da MILTree, o conjunto de dados Corel-300 já classificado e a Figura 4.4e apresenta sua correspondente árvore *ClassMatch*, onde os pontos verdes e vermelhos representam *bags* classificados corretamente e erroneamente, respectivamente. A precisão obtida para conjunto de dados de teste do Corel-300 sem o modelo de classificação atualizado foi 82,6% e a precisão obtida após a atualização do modelo de classificação usando as instâncias protótipo de oito *bags* foi 83,8%. Também vale a pena mencionar que a precisão obtida no ambiente de classificação padrão e não multi-instância do Corel-300 foi de 78%. Isto demonstra que o fortalecimento de um modelo de classificação multi-instância, usando novas instâncias protótipo, a partir dos *bags* que possivelmente não tinham a mais adequada instância protótipo, ajuda a criar um modelo de classificação multi-instância mais robusto e que melhora o desempenho final da classificação.

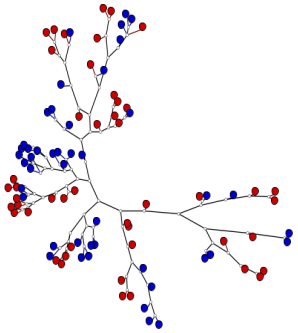
4.2.3 Caso de Estudo 3: Adicionando Novos Bags Usando o Layout MILTree

No terceiro caso de estudo, testamos nossa proposta utilizando o conjunto de dados Musk1. Este conjunto de dados binário é formado por 92 *bags* e 476 instâncias. A Figura 4.5a mostra a projeção do conjunto de dados Musk1 no espaço de projeção de *bags* da MILTree, onde os *bags* são representados como pontos e a cor representa a classe. *Bags* vermelhos representam *bags* positivos e *bags* azuis representam *bags* negativos.

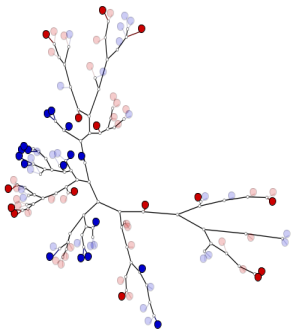
Neste estudo de caso pretende-se mostrar o uso de visualização para selecionar alguns novos *bags* com o intuito de atualizar o modelo de classificação multi-instância. Para isso, utilizamos a árvore *ClassMatch* do resultado da classificação inicial para identificar *bags* erroneamente classificados no conjunto de teste. Neste caso de estudo usamos MILTree para a projeção e MILTree-SI como método de seleção das instâncias protótipo. A seleção de novos *bags* é realizada no espaço de projeção de *bags* da MILTree.

Assim como no primeiro e segundo casos de estudo, selecionamos um conjunto de treinamento usando MILTree, no espaço de *bags*, para criar um modelo de classificação multi-instância inicial. A Figura 4.5b mostra o conjunto de treinamento selecionado para o conjunto Musk1. Utilizamos esse modelo inicial para classificar o conjunto de dados Musk1 e, em seguida, mostrar a árvore *InstancePrototypes ClassMatch*. Note que essa árvore só mostra *bags* que têm uma instância protótipo inadequada pertencente ao conjunto de treinamento. A Figura4.5c apresenta a árvore *InstancePrototypes ClassMatch* de Musk1, onde *bags* vermelhos representam *bags* com uma instância protótipo inadequada. Seguindo o mesmo procedimento feito no segundo caso de estudo, todos os *bags* vermelhos são selecionados e suas instâncias protótipo $B_{protoClass}$ são atualizados por B_{iy}. Depois, essas novas instâncias protótipo são adicionadas ao modelo inicial.

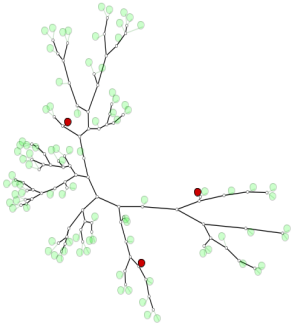
Diferente do segundo caso de estudo onde só atualizamos o modelo inicial com novas instâncias protótipo desde *bags* classificados incorretamente, neste terceiro caso de estudo o modelo inicial será atualizado com novos *bags*. Esses novos *bags* serão identificados fazendo uso da árvore *ClassMatch*. Note que a árvore *ClassMatch* mostra *bags* erroneamente classificadas de todo conjunto de dados usando o modelo inicial, e não só do conjunto de treinamento (como feito por a árvore *InstancePrototypes ClassMatch*). A Figura 4.5d apresenta a árvore *ClassMatch* do resultado da classificação inicial para o conjunto de dados Musk1, onde *bags* vermelhos representam os *bags* que foram erroneamente classificados. A Figura 4.5e destaca (usando uma elipse) os ramos onde a maior quantidade de *bags* foram erroneamente classificados.



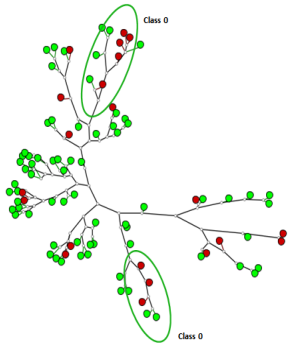
(a) Ground truth.



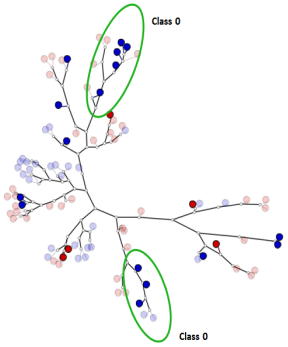
(b) Conjunto de treinamento selecionado.



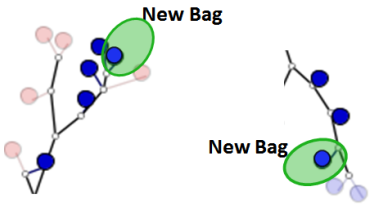
(c) Árvore *InstancePrototypes ClassMatch*.



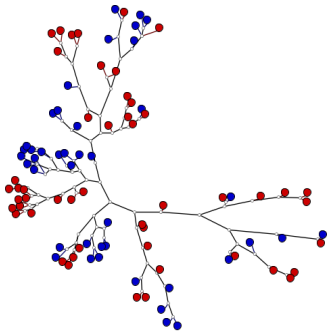
(d) Árvore *ClassMatch* do resultado de classificação inicial.



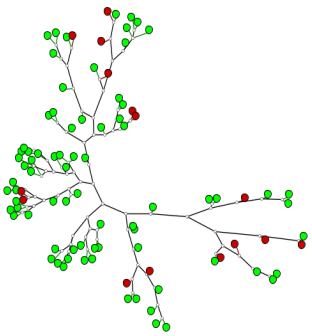
(e) *Bags* erroneamente classificados destacados no *Ground truth* dos dados.



(f) Seleção de novos *bags* localizados nos ramos onde maior quantidade de *bags* foram erroneamente classificados.



(g) Resultados da classificação usando o modelo atualizado com novas instâncias protótipo e novos *bags*.



(h) Árvore *ClassMatch* do resultado de classificação final.

Figura 4.5: Visualização do processo de Classificação multi-instância para o conjunto de dados Musk1. Todas as árvores usam a distância Euclidiana. Nas árvores (a), (b), (e), (f) e (g), os *bags* vermelhos e azuis representam *bags* positivos e negativos, respectivamente. Na árvore (c) os *bags* vermelhos representam *bags* com instâncias protótipo inadequadas. Nas árvores (d) e (h) os *bags* verdes representam *bags* corretamente classificados e os *bags* vermelhos representam *bags* incorretamente classificados.

A análise dos ramos pode ajudar a entender o comportamento do classificador. No caso do Musk1, a partir da matriz de confusão, podemos ver que grande número de *bags* negativos (*bags* azuis) foram classificados erroneamente como *bags* positivos (*bags* vermelhos), como se mostra na Figura 4.6[1]. O layout, por sua vez, mostra que estes ramos pertencem a mesma classe e que estão localizados em regiões diferente da árvore. Em outras palavras, eles não são vizinhos, como é mostrado na Figura 4.5e. Isto pode indicar que esta classe cobre uma vasta variedade de características, e por isso esses *bags* foram divididos em subclasses de maior homogeneidade.

		Classification	
		0.0	1.0
Ground Truth	0.0	14	14
	1.0	4	28

Figura 4.6: Matriz de confusão do resultado da classificação do conjunto de teste Musk1 usando o modelo de classificação inicial.

Usando a projeção dos *bags* na MILTree (espaço de *bags*), os usuários podem criar diferentes estratégias para re-atualizar o modelo de classificação multi-instância. A estratégia utilizada para este experimento consistiu na verificação dos ramos com as maiores taxas de erro de classificação, e procurar *bags* nesses ramos que melhor descrevam as classes a qual eles pertencem. Para isso, foi utilizado a árvore *ClassMatch* como mostrado na Figura 4.5d. Note-se que a árvore *ClassMatch* só pode ser construida apartir do *ground truth* dos dados. Na ausência deste, o usuário decide quais *bags* foram classificadas erroneamente visualizando seu conteúdo.

Após a análise do layout MILTree e identificação dos ramos com maior número de *bags* erroneamente classificados (como mostrado na Figura 4.5e), o usuário seleciona alguns *bags* para re-atualização do modelo inicial. A Figura 4.5f destaca os dois *bags* escolhidos. Observe que o modelo inicial já tinha sido atualizado com as instâncias protótipo detectadas na árvore *InstancePrototypes ClassMatch*. Em outras palavras, o modelo inicial realmente está sendo re-atualizado.

Finalmente, a partir da atualização do modelo inicial, é criado o modelo final utilizado para classificar o conjunto de dados Musk1. A Figura 4.5g apresenta, no espaço de projeção de *bags* da MILTree, o conjunto de dados Musk1 já classificado e a Figura 4.5h apresenta sua correspondente árvore *ClassMatch*, onde os pontos verdes e vermelhos representam instâncias corretamente e erroneamente classificados, respectivamente. A

[1]Apresenta-se a matriz de confusão com palavras em inglês, devido à escolha da linguagem para o sistema *VCS*.

precisão obtida para o conjunto de dados de teste Musk1 usando o modelo inicial foi de 73,9 %; a precisão obtida usando o modelo atualizado com as novas instâncias protótipo foi 75,2% . Por fim, a precisão obtida usando o modelo final, a qual foi atualizada com novos *bags* identificados usando a árvore *ClassMatch* foi 83,2%. Isso demonstra que a seleção de novas instâncias protótipo e de novos *bags* guiadas pela visualização dos dados ajuda na criação do modelo de classificação multi-instância mais robustos e com melhor desempenho.

4.3 Considerações Finais

Neste Capítulo nossa abordagem foi demonstrada através da execução de três casos de estudo. O primeiro caso de estudo demostrou que a seleção correta de instâncias protótipo nos *bags*, através da exploração visual, e a atualização de instâncias protótipo, influenciam de forma positiva os resultados finais da classificação. No segundo caso de estudo, se demostrou que a utilização das instâncias protótipo alternativas dos *bags* que não tinham uma instância protótipo adequada (mostradas na árvore *InstancePrototypes ClassMatch*), para atualizar o modelo de classificação, melhora os resultados alcançados. No último caso de estudo se demostrou que, através da árvore *ClassMatch* ou do exame de exemplares após a classificação, pode-se fazer uma análise visual dos resultados da classificação, com o intuito de identificar quais *bags* podem contribuir na melhora dos modelos de classificação.

O Capítulo 5 apresenta todos os experimentos realizados sobre diferentes conjuntos de dados com o intuito de avaliar o desempenho dos métodos MILTree-Med e MILTree-SI, baseados no layout MILTree para seleção de amostras e atualização de istâncias protótipo.

Capítulo

5

Resultados Experimentais

5.1 Considerações Iniciais

Esta Seção apresenta os experimentos realizados para medir a qualidade da proposta. Nós avaliamos o desempenho dos métodos MILTree-SI e MILTree-Med, ambos utilizando o layout proposto MILTree para a atualização dos modelos de classificação multi-instância. Para avaliar o desempenho na classificação de conjuntos de dados binários (utilizados frequentemente em cenários MIL), foram usados cinco conjuntos de dados considerados *benchmark* em aprendizado multi-instância e o conjunto de imagens Corel-1000. Para medir a qualidade dos métodos foram calculadas a média da acurácia, a média da precisão e a média da sensitividade. O desempenho dos métodos MILTree-SI e MILTree-Med, antes de atualizar as instâncias protótipo dos *bags* ou agregar novos *bags* ao modelo de classificação multi-instância com a ajuda do layout MILTree, são comparados com o desempenho alcançado depois de fazer as atualizações nos modelos. O desempenho dos métodos propostos também são comparados com o desempenho dos métodos MIL do estado da arte disponíveis para cada conjunto de dados. Além disso, avaliamos nossa proposta nos conjuntos de dados multiclasse Corel-1000, Corel-2000 e Biocreative, os quais são considerados problemas de classificação de grande escala.

Os resultados destes experimentos sugerem que a utilização de um layout visual em entornos MIL, assim como incluir novos *bags* no modelo, melhora substancialmente o

desempenho na classificação de dados multi-instância. Todos os experimentos são realizados usando o sistema VCS (apresentado na Seção 2.4), a qual foi incluída a visualização MILTree e os métodos de seleção de instâncias protótipo MILTree-Med e MILTree-SI. Além disso, a ferramenta LIBSVM[1] (Chang e Lin, 2011) usada pelo sistema VCS foi empregada para executar os treinamentos com SVM.

5.2 Medidas de Avaliação

As medidas de avaliação da classificação multi-instância utilizadas nos experimentos foram acurácia, precisão e sensitividade. Uma vez que o objetivo do aprendizado multi-instâncias é classificar um conjunto (*bags*) de objetos (instâncias), atribuindo rótulos só para os *bags*, as medidas de acurácia, precisão e sensitividade são calculadas só sobre o número de *bags* e não sobre o número de instâncias. Lembre-se que cada *bag* está representado por uma instância protótipo.

Nesse contexto, **Acurácia** mede a proporção de *bags* corretamente classificados, dentre todos os *bags* do conjunto de dados multi-instância. A **Precisão** mede a proporção de *bags* corretamente categorizadas em uma determinada classe, dentre todos os *bags* categorizados nessa mesma classe. Por último, a **Sensitividade** mede a proporção de *bags* corretamente categorizados em uma determinada classe, dentre todos os *bags* que realmente pertencem a essa classe.

Vale ressaltar que a medida de acurácia é fortemente utilizada na avaliação dos métodos de classificação multi-instâncias. No entanto, se sabe que a acurácia é dependente do balanceamento das classes do problema, para superar esse problema nos experimentos realizados, a acurácia sempre se associa e se mostra junto com as outras medidas, tentando dessa forma garantir uma melhor análise dos resultados da classificação. Nas Equações 5.1, 5.2 e 5.3 são apresentadas as fórmulas para cálculo da Acurácia, Precisão e Sensitividade. Para avaliar a classificação geral em todas as classes existentes no conjunto de dados, foi utilizada a média dos valores dessas medidas.

$$Acuracia_i = \frac{TP_i}{T} \tag{5.1}$$

$$Precisao_i = \frac{TP_i}{(TP_i + FP_i)} \tag{5.2}$$

$$Sensitividade_i = \frac{TP_i}{(TP_i + FN_i)} \tag{5.3}$$

[1]LIBSVM (http://www.csie.ntu.edu.tw/ cjlin/libsvm/)

Onde TP_i representa o número de *bags* da classe i classificadas como i. FP_i representa o número de *bags* de outra classe classificados como i. FN_i representa o número de *bags* da classe i classificadas em outra classe. FP_i representa o número de *bags* de outra classe classificados como i, e finalmente, T representa o número total de bags do conjunto de dados multi-instância.

5.3 Conjuntos de Dados Benchmarks

Descrição dos benchmarks:

Nós avaliamos MILTree-Med e MILTree-SI em cinco conjuntos de dados de referência MIL padrão (benchmarks): Musk1 e Musk2 descritos em (Dietterich et al., 1997), e os conjuntos de dados de imagens chamados Elefante, Fox e Tiger introduzidos em (Andrews et al., 2003), os quais têm sido amplamente utilizados em muitos estudos de aprendizado multi-instâncias.

Musk1 e Musk2 são conjuntos de dados de teste *benchmark* do mundo real, ambos publicamente disponíveis no repositório do aprendizado de máquina UCI[2] (Lichman, 2013). O conjunto de dados *Musk* é utilizado como o ponto de referência para testar praticamente todos os algoritmos MIL. *Musk* foi gerado na pesquisa sobre o problema de predição de reações químicas de drogas (ver 2.3.1). O problema consiste em determinar se uma molécula de droga irá se ligar fortemente a uma proteína alvo. Cada molécula de droga pode adotar uma variedade de formas ou conformações diferentes. Este problema poderia ser representado de uma forma muito natural em ambientes MIL: Cada molécula de droga seria um '*bag*' e as conformações que pode adotar seriam as 'instâncias' em esse '*bag*'.

Uma molécula é chamada de *musk* quando apenas uma das formas que a molécula possa assumir é a correta, ou seja, a molécula de droga iria se ligar à proteína alvo. Uma molécula é chamada de não *musk* quando nenhuma das formas da molécula permite a ligação. O objetivo da classificação no problema de predição de drogas é aprender a prever se novas moléculas serão *musks* (*bag* positivo) ou não *musks* (*bag* negativo). No entanto, as 166 características que descrevem essas moléculas dependem da forma exata, ou conformação, da molécula. Para gerar este conjunto de dados, as conformações de baixa energia das moléculas foram geradas e, em seguida, filtradas para remover as conformações altamente semelhantes. Esse processo resultou em 476 conformações.

[2]University of California at Irvine. https://archive.ics.uci.edu/ml/datasets

Assim, *Musk1* contém 47 *bags* positivos e 45 *bags* negativos, e o número de instâncias contidos em cada *bag* varia de 2 a 40. Musk2 contém 39 *bags* positivos e 63 *bags* negativos, e o número de instâncias contidas em cada *bag* varia entre 1 a 1044. Cada instância é representada por 166 atributos contínuos. A Tabela 5.1 mostra as informações detalhadas sobre os dados Musk.

	Bags		Instâncias		
Dataset	Total	Pos./Neg.	Total	Min/Max	Dim
Musk1	92	47/45	476	2/40	166
Musk2	102	39/63	6598	1/1044	166

Tabela 5.1: Conjuntos de dados Musk e a quantida média de instâncias por *bag* (Inst/Bag) para cada conjunto de dados.

Por outro lado, os conjuntos de dados de imagens conformada por Elephant, Fox e Tiger têm como objetivo diferenciar as imagens que contem elefantes, raposas e tigres daqueles que não possuem, respectivamente. Neste caso, os *bags* são imagens, e as instâncias são as regiões segmentadas na imagem. Os detalhes sobre esses conjuntos de dados são apresentados na Tabela 5.2.

	Bags		Instâncias		
Dataset	Total	Pos./Neg.	Total	Avg. inst./bag	Dim
Elephant	200	100/100	1391	6.96	230
Fox	200	100/100	1220	6.10	230
Tiger	200	100/100	1320	6.60	230

Tabela 5.2: Conjuntos de dados de imagens e a quantidade média de instâncias por *bag* (Inst/Bag) para cada conjunto de dados.

Experimentos:

Dois testes foram realizados. O primeiro teste usa o método de selecção de instâncias protótipo MILTree-SI, enquanto que o segundo teste utiliza o método dede instâncias protótipo MILTree-Med. Ambos utilizam o layout MILTree para atualizar as instâncias protótipo e para ajudar ao usuário a identificar novos *bags* com o intuito de atualizar o modelo inicial de classificação multi-instância.

Nos experimentos de ambos os testes, o conjunto de dados foi dividido em dois subconjuntos: aproximadamente 30% dos dados para o conjunto de treinamento e 70% dos dados para o teste. Note que nossos métodos apenas precisam de uma pequena quantidade

de *bags* como dados de treinamento, diferente de outros métodos MIL que precisam de mais do 80% dos dados para treinamento. Em relação à atualização de instâncias protótipo, nos conjuntos de dados musk foram atualizados aproximadamente dois *bags* por conjunto de treinamento. Isso é devido ao fato de que as instâncias protótipo identificadas para cada *bag* usando os métodos de selecção da instância protótipo foram suficientes para classificar corretamente os *bags*. Nos conjuntos de dados de imagens, apenas no dataset Fox foram atualizadas quatro instâncias protótipo usando MILTree-SI e duas instâncias protótipo usando MILTree-Med, a fim de melhorar o modelo de classificação multi-instância.

Além disso, aproximadamente três a oito *bags* (representados por suas instâncias protótipo) foram utilizados para atualizar o modelo de classificação nos conjuntos de dados Musk e nos conjuntos de dados de imagens, respectivamente. Estes *bags* foram identificados utilizando o espaço de projeção dos *bags* de MILTree, a fim de aumentar o desempenho do classificador. MILTree ajudou os usuários a identificar visualmente a classe que tem mais *bags* erroneamente classificados, com o intuito de conhecer qual classe precisa de novos *bags* para fortalecer seu modelo de classificação.

Na tabela 5.3 são apresentadas as acurácias alcançadas dos métodos MILTree-SI e MILTree-Med antes de atualizar as instâncias protótipo dos *bags* ou agregar novos *bags* ao modelo de classificação multi-instância com a ajuda do layout MILTree (colunas 2 e 4), e as acurácias alcançadas depois de fazer as atualizações nos modelos (colunas 3 e 5). Esses resultados demostram que a inserção do usuário para a correção ou atualização das instâncias protótipo dos *bags*, seleção de novos *bags* os quais sejam relevantes (ver Seção 4.2.3) e outras análises sobre os dados multi-instancia usando o layout MILTree realmente ajudam a criar modelos mais robustos e com resultados promissores na classificação multi-instância. Os valores em negrito, na Tabela 5.3, indicam o modelo que obteve a melhor acurácia em cada método, comparando a acurácia obtida no modelo inicial (sem atualizações) com o novo modelo (com atualizações).

A acurácia da classificação, a precisão e a sensitividade, nos novos modelos (modelos atualizados), para ambos testes, são apresentados na Tabela 5.4. Nele se pode observar que o método MILTree-Med tem melhor desempenho em todos os conjuntos de dados *benchmarks*, exceto em Musk1 e Fox, onde ambos os métodos têm igual desempenho. Os valores em negrito, na Tabela 5.4, indicam o método que obteve a melhor acurácia em cada conjunto de dados.

Na Tabela 5.5, comparamos MILTree-SI e MILTree-Med, utilizando os resultados obtidos com o novo modelo (com atualizações das instâncias protótipo), com o desempenho de 9 algoritmos MIL na literatura: quatro métodos *baselines*, tais como EM-DD (Zhang e Goldman, 2001), DD-SVM (Chen e Wang, 2004), mi-SVM(Andrews et al., 2003) e

	MILTree-Med		MILTree-SI	
Dataset	modelo ini.	modelo novo.	modelo ini.	modelo novo.
Musk1	73.9	**83.2**	73.9	**83.2**
Musk2	85.4	**91.8**	73.5	**85.4**
Elephant	76.8	**83.1**	77.4	**81.4**
Fox	64.9	**72.7**	62.6	**72.7**
Tiger	76.0	**83.0**	72.7	**82.9**

Tabela 5.3: Comparação das acurácias obtidas nos modelos iniciais (sem atualizações) com os novos modelos (com atualizações) usando os métodos de seleção de instâncias protótipo MILTree-Med e MILTree-SI nos *benchmarks* e o layout MILTree.

	MILTree-Med			MILTree-SI		
Dataset	Acurác.	Prec.	Sensitiv.	Acurác.	Prec.	Sensitiv.
Musk1	**83.2**	83.2	81.7	**83.2**	82.4	81.7
Musk2	**91.8**	91.4	91.4	85.4	84.4	84.3
Elephant	**83.1**	81.7	81.6	81.4	79.4	79.4
Fox	**72.7**	68.3	68.3	**72.7**	68.3	68.3
Tiger	**83.0**	82.0	81.4	82.9	83.4	81.4

Tabela 5.4: Resultados da classificação usando MILTree-Med e MILTree-SI nos *benchmarks*.

MI-SVM(Andrews et al., 2003), e quatro métodos focados na seleção de instâncias como MILES (Chen et al., 2006), MILIS (Fu et al., 2011), MILSIS (Yuan et al., 2012) e MILD-B (Li e Yeung, 2010), e um dos métodos mais recentes chamado de MILDE (Amores, 2015). A melhor acurácia se destaca em negrito.

Os métodos propostos MILTree-SI e MILTree-Med são muito competitivos com relação aos outros métodos, em particular MILTree-Med. Nos conjuntos de dados Musk2, Fox e Elephant, a acurácia média da classificação usando MILTree-Med é de 91,8%, 72,7% e 83,0% respectivamente, que são estatisticamente melhores do que os outros algoritmos MIL na literatura. Além disso, MILTree-Med é muito competitivo com os métodos MIL do estado da arte e melhor do que eles em relação à performance média global, de 82,8%.

Métodos	Musk1	Musk2	Elephant	Fox	Tiger	Avg.
MILTree-Med	83.2	**91.8**	83.1	**72.7**	**83.0**	**82.8**
MILTree-SI	82.3	85.4	81.4	72.7	82.9	81.1
EM-DD	84.8	84.9	78.3	56.1	72.1	75.2
MI-SVM	77.9	84.3	73.1	58.8	66.6	72.1
mi-SVM	87.4	83.6	80	57.9	78.9	77.6
DD-SVM	85.8	91.3	83.5	56.6	77.2	79.0
MILD-B	88.3	86.8	82.9	55.0	75.8	77.8
MILIS	88.6	91,1	-	-	-	-
MILES	86.3	87.7	84.1	63.0	80.7	80.4
MILSIS	**90.1**	85.6	81.8	66.4	80.0	80.9
MILDE	87.1	91.0	**85**	66.5	83.0	82.5

Tabela 5.5: Comparação entre MILTree-SI/MILTree-Med e os métodos relacionados a partir da literatura sobre os conjuntos de dados benchmarks. Os valores em negrito indicam o método que obteve o melhor desempenho em cada conjunto de dados.

5.4 Classificação de Imagens

Descrição dos conjuntos de imagens:

Classificação de imagens refere-se à rotulagem de imagens em categorias pré-definidas. Para esse experimento utilizamos o conjunto de dados de imagens Corel 1000, publicado pela Corel Corporation que contém 1000 imagens (Chen et al., 2006). Ele contém 10 subcategorias que representam temas de interesse distintos. Cada subcategoria contém 100 imagens.

Uma vez que o conjunto de dados Corel não foi planejado para problemas de aprendizado multi-instância, o conjunto Corel foi adaptado para cenários MIL através da segmentação das imagens em regiões no trabalho de Chen e Wang (2004). Dessa forma, as imagens representam os *bags* e as regiões de uma imagem representam as instâncias. Cada instância é representada por um vetor de características, a qual consiste de nove características. Três delas são a média das componentes de cor LUV, outras três representam a raiz quadrada da energia nas bandas de alta frequência da transformada Wavelet e, as três últimas representam as componentes de forma das regiões da imagem normalizadas com a inércia da ordem 1, 2 e 3. Como resultado, cada região (instância) numa imagem é um vetor de características que representa as propriedades de cor, textura, e forma da região. O

conjunto de dados Corel para MIL esta publicamente disponível no repositório DDSVM[3]. Na segunda coluna das Tabelas 5.7 e 5.8 é apresentado o número médio de instâncias por *bag* para cada categoria.

Para criar os conjuntos binários, compostos por *bags* positivos e *bags* negativos, escolhemos uma categoria como a classe positiva e selecionamos uniformemente 100 imagens a partir das categorias restantes para criar a classe negativa como é realizado em (Andrews et al., 2003). O mesmo processo é seguido para cada categoria. Finalmente, temos 10 sub-conjuntos de dados binários, onde cada sub-conjunto é composto por 100 imagens positivas (*bags* positivos) e 100 imagens negativas (*bags* negativos).

Experimentos:

Dois testes foram realizados. O primeiro teste usa MILTree-SI, e o segundo teste utiliza MILTree-Med. Ambos os utilizam o layout MILTree, para atualizar as instâncias protótipo e para identificar novos *bags* que possam ajudar na criação de modelos de classificação multi-instância mais robustos.

Para ambos testes, dividimos o conjunto de dados em cerca de 20% de conjunto de treino e 80% como conjunto de teste. Note que nossos métodos apenas precisam de uma pequena quantidade de *bags* como conjunto de treinamento para obter um bom desempenho, devido ao uso da visualização MILTree, a qual guia o usuário na escolha do conjunto de treino relevante. Maiores detalhes sobre este processo são apresentados na Seção 4.2.1.

Na tabela 5.6 são apresentadas as acurácias alcançadas usando os métodos MILTree-SI e MILTree-Med antes de atualizar as instâncias protótipo dos *bags* ou agregar novos *bags* ao modelo de classificação multi-instância com a ajuda do layout MILTree (colunas 2 e 4), e as acurácias alcançadas depois de fazer as atualizações nos modelos (colunas 3 e 5). Segundo os resultados, em ambos os métodos a acurácia na classificação com os modelos atualizados obtem melhores resultados que os modelos iniciais, nos quais o usuário só fez a seleção do conjunto de treino usando o layout MILTree, mas não as atualizações nas instâncias protótipo, nem as atualizações dos modelos com novas instâncias ou *bags* (ver Seção 4.2.3). Os valores em negrito, na Tabela 5.6, indicam o modelo que obteve a melhor acurácia em cada método, comparando a acurácia obtida no modelo inicial (sem atualizações) com o novo modelo (com atualizações).

[3]http://www.cs.olemiss.edu/∼ychen/ddsvm.html

	MILTree-Med		MILTree-SI	
Dataset	modelo ini.	modelo novo.	modelo ini.	modelo novo.
Categoria0	72.2	**76.0**	63.4	**68.1**
Categoria1	74.5	**79.1**	71.7	**75.8**
Categoria2	70.5	**79.0**	71.8	**76.5**
Categoria3	90.1	**91.3**	70.5	**72.9**
Categoria4	76.8	**78.4**	82.1	**83.7**
Categoria5	78.0	**81.9**	80.4	**80.9**
Categoria6	89.7	**89.1**	87.9	**89.1**
Categoria7	79.5	**84.2**	80.0	**80.0**
Categoria8	80.2	**81.2**	73.8	**75.2**
Categoria9	78.3	**81.3**	74.8	**75.7**

Tabela 5.6: Comparação das acurácias obtidas nos modelos iniciais (sem atualizações) com os novos modelos (com atualizações), usando os métodos de seleção de instâncias protótipo MILTree-Med e MILTree-SI nos subconjuntos de imagens do Corel-1000 (categorias) e o layout MILTree.

A acurácia da classificação, a precisão e a sensitividade nos novos modelos (modelos atualizados), para ambos testes são apresentados na Tabela 5.7 e na tabela 5.8. Em ambas as tabelas *Proto* indica o número de instâncias protótipo que foram atualizados no espaço de projeção de *bags* ou no espaço de projeção de instâncias. *AddBags* indica o número de *bags* que foram incluídos no conjunto de treinamento a partir do espaço de projeção de *bags*. *AddProto* indica o número de instâncias protótipo B_{iy} incluídas no conjunto de treinamento apartir dos *bags* de treinamento. Observe que quando agregamos um *bag*, o que realmente estamos agregando é só a instância protótipo B_{ix} desse *bag*.

AddBags só está incluído na segunda tabela, a qual apresenta os resultados de MILTree-SI. Isso porque só no método MILTree-SI é recomendável incluir as instâncias protótipo B_{iy} no modelo de classificação ao invés de retirar os B_{ix} para incluir B_{iy} (o qual é feito em MILTree-Med). Isso ocorre porque, no método MILTree-SI, as instâncias protótipo B_{ix} e B_{iy} são semelhantes, diferente de MILTree-Med, que seleciona instâncias protótipo B_{ix} e B_{iy} com características muito diferentes. Portanto, agregar instâncias protótipo B_{iy} em MILTree-SI é mais útil do que em MILTree-Med.

A tabela 5.9 apresenta a acurácia alcançada por diferentes métodos MIL na literatura, incluindo EM-DD (Zhang e Goldman, 2001), mi-SVM(Andrews et al., 2003), MI-SVM(Andrews et al., 2003), DD-SVM(Chen e Wang, 2004) e um método mais atual chamado de SMILES(Xiao et al., 2014). Podemos ver que os métodos propostos MILTree-SI e MILTree-Med em geral tem maior acurácia na classificação dos diferentes conjuntos de

		Medidas				
Categoria ID	inst/bag	Acurác.	Prec.	Sensitiv.	Proto	AddBags
Categoria0	4.84	76.0	72.7	72.7	7	0
Categoria1	3.54	79.1	78.7	76.7	3	2
Categoria2	3.1	79.0	79.2	76.7	6	3
Categoria3	7.59	91.3	90.9	90.9	4	0
Categoria4	2.00	78.4	79.7	76.0	4	0
Categoria5	3.02	81.9	83.2	80.3	4	1
Categoria6	4.46	89.1	88.7	88.5	1	0
Categoria7	3.89	84.2	83.6	82.9	6	2
Categoria8	3.38	81.2	79.5	79.3	1	1
Categoria9	7.24	81.3	81.3	79.4	1	0

Tabela 5.7: Resultados da classificação usando MILTree-Med sobre o Corel-1000.

		Medidas					
Categoria ID	inst/bag	Acurác.	Prec.	Sensitiv.	Proto	AddProto	AddBags
Categoria0	4.84	68.1	62.2	62.1	0	11	2
Categoria1	3.54	75.8	74.9	72.7	3	0	0
Categoria2	3.1	76.5	73.3	73.3	3	0	2
Categoria3	7.59	72.9	68.9	68.6	0	12	0
Categoria4	2.00	83.7	85.0	82.3	4	0	1
Categoria5	3.02	80.9	80.8	79.0	0	14	0
Categoria6	4.46	89.1	88.6	88.5	1	0	1
Categoria7	3.89	80.0	78.3	77.9	0	6	0
Categoria8	3.38	75.2	71.7	71.7	0	0	2
Categoria9	7.24	75.7	72.4	72.4	0	6	0

Tabela 5.8: Resultados de classificação usando MILTree-SI sobre o conjunto de dados Corel-1000.

dados. Em particular, o método MILTree-Med superou todos os outros métodos em 7 dos 10 conjuntos.

MILTree-Med é muito competitivo com EM-DD, MI-SVM e DD-SVM os quais estão baseados na seleção da mais representativa instância dentro dos *bags* para a construção de um classificador. Por outro lado, MILTree-Med também é muito competitivo com mi-SVM e SMILES que são métodos que utilizaram todas as instâncias contidas num *bag* para criar um classificador.

CatID	EM-DD	mi-SVM	MI-SVM	DD-SVM	SMILES	MILTree-SI	MILTree-Med
Cat0	68.7	71.1	69.6	70.9	72.4	68.14	**76.0**
Cat1	56.7	58.7	56.4	58.5	62.7	75.8	**79.1**
Cat2	65.1	67.9	66.9	68.6	69.6	76.5	**79.0**
Cat3	85.1	88.6	84.9	85.2	90.1	72.9	**91.3**
Cat4	96.2	94.8	95.3	96.9	**96.6**	83.7	78.4
Cat5	74.2	80.4	74.4	78.2	80.5	80.9	**81.9**
Cat6	77.9	82.5	82.7	77.9	83.3	89.1	**89.1**
Cat7	91.4	93.4	92.1	94.4	**94.7**	80.3	84.2
Cat8	70.9	72.5	67.2	71.8	73.8	75.2	**81.2**
Cat9	80.2	84.6	83.4	84.7	**84.9**	75.8	81.3

Tabela 5.9: Comparação entre MILTree-SI/MILTree-Med e os métodos relacionados na literatura sobre o Corel Dataset.

Os valores em negrito indicam o método que obteve o melhor desempenho em cada conjunto de dados.

5.5 Desempenho para Classificação MultiClasse

Nesta Seção voltamos nossa atenção para o desempenho de MILTree-Med e MILTree-SI utilizando o layout MILTree para apoiar o usuário em resolver problemas de classificação multiclasse. Os métodos *baseline*, tais como EM-DD (Zhang e Goldman, 2001), mi-SVM (Andrews et al., 2003), MI-SVM (Andrews et al., 2003), DD-SVM (Chen e Wang, 2004), são originalmente propostos para classificação binária. No entanto, alguns deles, como DD-SVM (Chen e Wang, 2004) e MILES (Chen et al., 2006), foram adaptados para classificação multi-classe usando decomposição *one-against-all*. Utilizando a mesma estratégia, os métodos propostos MILTree-Med e MILTree-SI também trabalham para classificação multiclasse. Assim, o problema de classificação multiclasse é decomposto em uma série de problemas de classificação binários, separando cada classe das classes restantes. Fazemos isso de forma automática, ou seja, nosso algoritmo pode receber uma base de dados multiclasse e internamente realizar a decomposição *one-against-all*.

Para fins de observar o desempenho no contexto multiclasse, testamos MILTree-Med e MILTree-SI sobre os conjuntos de dados de imagens Corel-1000 e Corel-2000, e sobre o conjunto de dados de texto Biocreative.

Corel-1000 e Corel-2000

O conjunto de dados COREL contém 2000 imagens obtidas a partir de 20 categorias diferentes, com 100 imagens em cada classe. Detalhes da segmentação e extração de características foram mencionados na Seção 5.4. A Tabela 5.10 mostra a quantidade total de *bags* e instâncias dos dois conjuntos de imagens. Para nossos experimentos, realizamos dois testes. O primeiro teste usa apenas as primeiras 10 categorias do conjunto de dados. Esse conjunto é chamado Corel-1000. O segundo conjunto de dados usa o conjunto total de imagens das 20 categorias e é chamado de Corel-2000. Note que, na Seção 5.4, Corel-1000 é tratado como um problema binário, e por isso tinha outra configuração para realizar os testes. A Figura 5.1 apresenta algumas imagens exemplo das 20 categorias e seus correspondentes resultados de segmentação.

Dataset	*Bags*	Instâncias	Dimensões
Corel-1000	1000	4306	9
Corel-2000	2000	7947	9

Tabela 5.10: Conjuntos de texto Biocreative. Quantidade total de *bags* e instâncias por cada categoria.

Dois testes foram realizados para o conjunto Corel-1000 e dois testes também foram realizados para o conjunto Corel-2000. O primeiro teste usa MILTree-Med, e o segundo teste utiliza MILTree-SI. Ambos utilizam o layout MILTree para a seleção do conjunto de treinamento.

Nos quatro testes, o conjunto de dados foi dividido em aproximadamente 30% de dados de treinamento e 70% de dados de teste. O conjunto de dados de treinamento foi selecionado usando MILTree no espaço de *bags*.

Com relação à atualização de instâncias protótipo, para o conjunto Corel-1000 usando MILTree-Med foram atualizados no total 111 *bags*, os quais foram selecionados facilmente utilizando a árvore *InstancesPrototypes ClassMatch*. Usando o MILTree-SI, foram identificados 136 *bags* com instâncias protótipo inadequadas, porém, só 37 deles (localizadas nos extremos da árvore) foram escolhidas e agregadas ao modelo inicial porque os *bags* localizados no meio da árvore não contribuiam para a melhoria do modelo. Note que em conjuntos de dados maiores encontra-se maior quantidade de *bags* com instâncias protótipo inadequadas, nesses casos a funcionalidade de atualização de instâncias protótipo da árvore *InstancesPrototypes ClassMatch* se mostra mais útil.

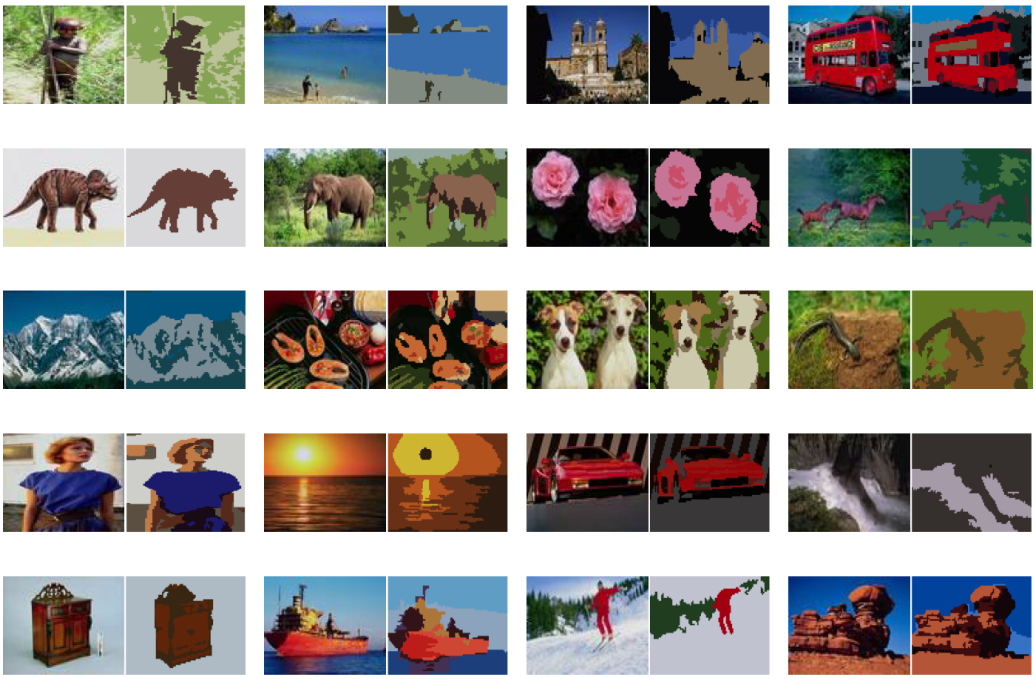


Figura 5.1: Imagens selecionadas aleatoriamente a partir de 20 categorias do conjunto de dados Corel e seu correspondente resultado na segmentação. As regiões segmentadas são mostradas na sua cor representativa.

No caso do conjunto Corel-2000 usando MILTree-Med, foram atualizados no total 336 *bags*, os quais foram selecionados facilmente utilizando a árvore *InstancesPrototypes ClassMatch*. Usando o MILTree-SI, foram agregados ao modelo inicial 387 instâncias protótipo alternativas, as quais foram identificadas utilizando a árvore *InstancesPrototypes ClassMatch*. Note que em ambos os testes, em Corel-1000 e Corel-2000, a árvore *InstancesPrototypes ClassMatch* ajudou o usuário tanto na identificação de *bags* com inadequadas instâncias protótipo como na seleção de qual desses *bags* (os localizados nos extremos da árvore) são melhores candidatos para atualizar suas instâncias protótipo de maneira que melhore os resultados do classificador.

A Tabela 5.11 apresenta a acurácia obtida na classificação por MILTree-Med e MILTree-SI, incluindo os resultados de DD-SVM (Chen e Wang, 2004), MILES (Chen et al., 2006), MILIS (Fu et al., 2011) e MILDE (Amores, 2015) conforme relatado nos artigos originais, e os resultados de MI-SVM e mi-SVM como relatado em Fu et al. (2011). A partir da tabela, podemos ver que MILTree-Med e MILTree-SI são mais competitivos para a classificação de imagens multiclasse que outros métodos na literatura devido à eficiente seleção visual de *bags* de treinamento, e a seleção de instâncias protótipo para cada *bag*.

Método	Corel-1000	Corel-2000
MILTree-Med	**93.1**	**93.9**
MILTree-SI	90.3	**93.9**
MI-SVM	75.1	54.6
mi-SVM	76.4	53.7
DD-SVM	81.5	67.5
MILIS	83.8	70.1
MILES	82.3	68.7
MILDE	-	74.8

Tabela 5.11: Comparação entre MILTree-SI/MILTree-Med e os métodos relacionados na literatura sobre os conjuntos de dados Corel-1000 e Corel-2000. Os valores em negrito indicam o método que obteve o melhor desempenho em cada conjunto de dados.

Biocreative

Biocreative é um problema de categorização de texto, contendo 1623 documentos (artigos) extraído de revistas biomédicas, divididos em 3 categorias ou *Gene Ontology* (GO) chamados: "Components", "Processes" e "Functions" (Ray e Craven, 2005b). A Figura 5.2 apresenta um exemplo de documento de texto biocreative. No contexto de MIL, cada documento é considerado um *bag* e os parágrafos do artigo são as instâncias. No total temos 1623 *bags* e 34569 instâncias. A Tabela 5.12 apresenta detalhes sobre este conjunto de dados. A tarefa é decidir em qual categoria deve ser anotado um documento. Como entrada, temos o número de documentos e cada parágrafo descrito por um vetor de características, o qual representa a frequência de ocorrência das palavras e estatísticas que captam alguns aspectos da interação proteína-categoria GO, tais como a distância média entre as menções da proteína e o texto relacionado com a categoria GO.

Protein: Mitochondrial 28S ribosomal protein S14
Article: PUBMED ID 10938081
...Three of the four currently identified mammalian mitochondrial small subunit ribosomal proteins that have prokaryotic homologs (S7, S10, and S14) are located in the head of the small subunit...

Figura 5.2: Um exemplo de um documento (artigo) do conjunto de dados biocreative. Adaptado de (Ray e Craven, 2005b).

Dataset	*Bags*	Instâncias	Dimensões
Components	423	9104	200
Functions	443	9387	200
Processes	757	25181	200

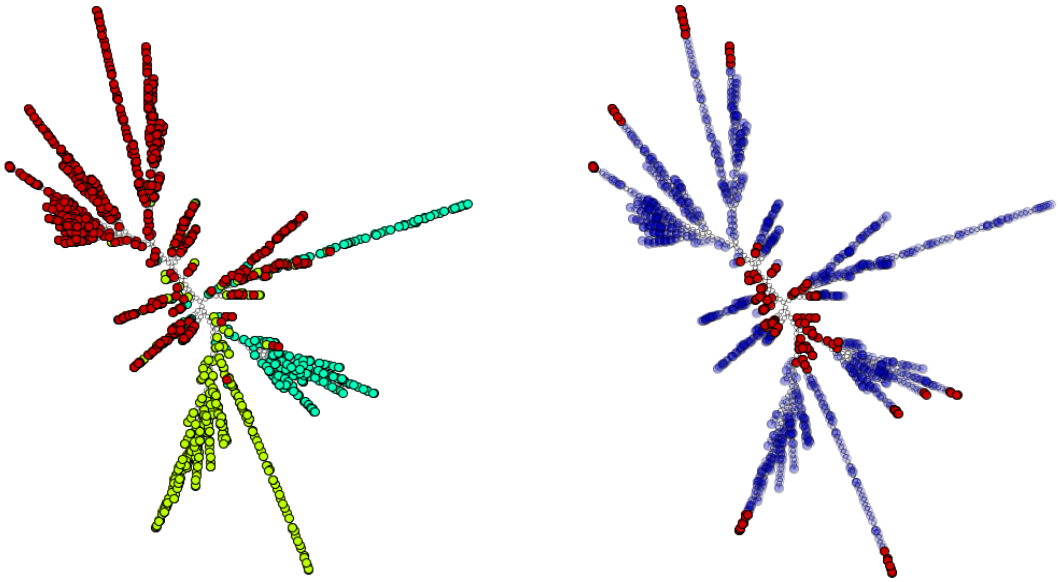
Tabela 5.12: Conjuntos de texto Biocreative. Quantidade total de *bags* e instâncias por cada categoria.

Dois testes foram realizados. O primeiro teste usa MILTree-SI, e o segundo teste utiliza MILTree-Med. Ambos utilizam o layout MILTree para a seleção do conjunto de treinamento. Para ambos testes, dividimos o conjunto de dados em cerca de 10% de conjunto de treinamento e 90% como conjunto de teste. Nossos métodos apenas precisaram de 10% de conjunto de treinamento para obter bons resultados, devido ao uso da visualização MILTree, a qual guia ao usuário na escolha do conjunto de treino relevante. Maiores detalhes para escolher conjunto de treinamento relevante são descritos na Seção 4.2.1.

Em ambos os experimentos, nenhum *bag* com instâncias protótipo inadequadas foi identificado. Isso quer dizer que as instâncias escolhidas como instâncias protótipo pelos métodos de seleção MILTree-SI e MILTree-Med foram as mais adequadas, refletindo-se isso nos resultados da classificação. Além disso, a boa separação das categorias "Components", "Processes" e "Functions" no espaço de projeção de *bags* da MILTree foram determinantes para guiar o usuário na escolha de *bags* representativos de cada categoria. A Figura 5.3 apresenta a projeção dos *bags* no espaço de *bags* da MILTree. Note que a boa escolha do conjunto de treinamento está diretamente ligada à qualidade do modelo de classificação. A Tabela 5.13 mostra a acurácia da classificação obtidos por MILTree-Med e MILTree-SI.

Com o intuito de comparar nossos resultados com os métodos *baseline*: DD (Dietterich et al., 1997), EM-DD (Zhang e Goldman, 2001), e MI-SVM (Andrews et al., 2003), foi utilizada a ferramenta *Weka*[4]. Como DD, EM-DD e MI-SVM só trabalham com classificação binária, foram criados três conjuntos de dados binários: "ComponentsData", "ProcessesData" e "FunctionsData". O estratégia utilizada para criar estes conjuntos binários foi o seguinte: O conjunto binário "ComponentsData" é formado por *bags* positivos (documentos da categoria "Components") e *bags* negativos (documentos das categorias "ProcessesData" e "FunctionsData"). Essa mesma estratégia foi utilizada para criar os conjuntos binários "ProcessesData" e "FunctionsData". A Tabela 5.13 apresenta as acurácias média (entre os 3 conjuntos binários) da classificação de *10-fold cross-validation*

[4]http://www.cs.waikato.ac.nz/ml/weka/documentation.html



(a) **(b)**

Figura 5.3: Espaço de projeção de *bags* da MILTree para o conjunto de textos Biocreative usando MILTree-SI, com a projeção do seu *ground truth*(a), e o conjunto de amostras selecionadas para treinamento (b). As árvores MILTree geradas em (a) e (b) usam a distância Euclidiana.

alcançada por esses métodos. Assim, nessa mesma Tabela 5.13 pode-se observar que os métodos propostos MILtree-Med e MILTree-SI, com ajuda do layout MILTree, obtêm maiores valores de acurácia na classificação do conjunto de texto Biocreative do que os métodos *baseline*.

Método	Biocreative
MILTree-Med	**99,1**
MILTree-SI	96.3
MI-SVM	90,9
EM-DD	91.0
DD	90.9

Tabela 5.13: Comparação das acurácias na classificação entre MILTree-SI/MILTree-Med e os métodos MIL *baselines* para o conjunto de dados Biocreative. O valor em negrito indica o método que obteve o melhor desempenho.

5.6 Considerações Finais

Nesse capítulo foram apresentados os resultados dos métodos de seleção de instâncias e as discussões sobre os experimentos executados. Os experimentos mostraram que a projeção dos dados multi-instâncias usando a árvore multi-escala MILTree e os métodos de seleção de instâncias protótipo MILTree-SI ou MILTree-Med melhoram a qualidade dos modelos de classificação criados e por tanto melhoram os resultados da classificação. O ponto principal é que esta abordagem permite que o usuário interfira na seleção das instâncias protótipo dos *bags* por meio do layout MILTree (desde o espaço de projeção de *bags* ou de instâncias), permitindo corrigir ou atualizar a instância protótipo dos *bags* que têm instâncias protótipo inadequadas (*bags* com instâncias protótipo que não são as mais representativas entre todas as instâncias do *bag*). De uma forma geral os métodos propostos MILTree-Med e MILTree-SI obtiveram melhores resultados comparáveis com os métodos *baseline*. No próximo Capítulo são apresentadas as conclusões e limitações dos métodos, bem como uma discussão sobre trabalhos futuros.

Capítulo

6

Conclusões

6.1 Conclusões

A visualização MILTree é uma contribuição importante para a mineração visual de dados sobre cenários de aprendizado múltipla instância. A MILTree tem uma estrutura intuitiva, que está relacionada com os dados MIL. Além de apoiar a compreensão visual do conjunto de dados, a nossa abordagem também utiliza métodos de seleção de instâncias protótipo que lidam com problemas multi-classe: o MILTree-SI tenta descobrir as instâncias mais representativas em ambos os *bags* positivos e negativos, onde *bags* negativos também podem ter instâncias positivas; o MILTree-Med usa um algoritmo de agrupamento para particionar as instâncias sem rótulo, tentando encontrar grupos positivos e negativos com o intuito de identificar as instâncias protótipo. Os métodos de seleção de instâncias aliados com a visualização foram capazes de melhorar os resultados dos métodos MIL do estado da arte usando procedimentos automáticos, além de ser também flexível, permitindo ao usuário escolher instâncias manualmente.

A fim de melhorar o modelo de classificação, foi adotada uma metodologia de mineração visual, que apoiada na visualização MILTree e os métodos MILTree-Med e MILTree-SI, permitem ao usuário participar de todas as etapas do processo de classificação multi-instância, tais como a seleção de conjunto de treinamento, criação, atualização e validação do modelo de classificação multi-instância.

Os métodos propostos consistentemente superaram os métodos MIL do estado da arte na grande maioria dos conjuntos de dados multi-instância. Os resultados mostram claramente que a análise visual de dados de múltiplas instâncias através do uso da MILTree contribuíram para melhorar a seleção das instâncias protótipo e criar modelos de classificação mais precisos. A visualização de dados multi-instância também ajudou na seleção de conjuntos de treinamento representativos já que a MILTree mantem as propriedades da árvore NJ com relação ao posicionamento dos dados no layout. Além disso, a análise visual dos resultados da classificação permite adotar determinadas estratégias para atualizar o modelo e, portanto, obter melhores resultados na classificação de múltiplas instâncias.

6.2 Limitações

Embora os resultados apresentados neste trabalho sejam promissores, existem algumas limitações com relação a utilização da proposta. A primeira limitação refere-se à exploração manual de grandes quantidades de *bags* com instâncias protótipo não adequadas. Embora o usuário tenha a opção de escolher a atualização automática das instâncias protótipo com as instâncias protótipo alternativas usando a árvore *InstancePrototypes ClassMatch*, ou fazendo a exploração só dos *bags* mais afastados do centro da NJ (o qual é recomendável nesses casos), nem sempre isso representaria grandes melhorias na classificação. Nesse caso é necessária a exploração visual dos *bags* na procura de instâncias protótipo. Essa é uma limitação que precisa ser melhor estudada e entendida.

Apesar da nossa abordagem conseguir lidar com grandes quantidades de instâncias agrupando-lhes em *bags* e fazendo o análise a partir da projeção de *bags* e instâncias em dois níveis, a abordagem tem dificuldades com a interação de grandes quantidades de *bags*.

6.3 Trabalhos Futuros

Um aspecto interessante da abordagem proposta é que ela consegue classificar tanto *bags* como também instâncias, e por isso os usuários, ao explorar um *bag*, conseguem identificar as instâncias que foram erroneamente classificadas (assumindo que todas deveriam pertencer à mesma classe do *bag*). No entanto, como o objetivo do aprendizado multi-instância é a classificação dos *bags*, a análise final da classificação só é feita sobre os *bags*. Porém, a abordagem poderia ser estendida para classificação multi-instânces multi-label. Nesse cenário, também se deveriam estudar e analizar os resultados da classificação no nível das instâncias e finalmente associar um *bag* a vários rótulos.

Considerando a importância da seleção correta de instâncias protótipo dentro dos *bags*, e aproveitando que a MILTree permite explorar visualmente as instâncias dos *bags* no espaço de projeção das instâncias, um trabalho futuro pode ser a procura de estratégias para mostrar visualmente o objeto associado a uma instância (como a região segmentada de uma imagem, o parágrafo de um texto, etc.) de forma eficiente. Embora no contexto de aprendizado multi-instâncias não se conhecem os rótulos das instâncias, essa funcionalidade ajudaria o usuário em tarefas de rotulamento de dados multi-instância e consequentemente na validação de instâncias protótipo de um *bag*.

Pretende-se investigar novos métodos de extração de instâncias a partir de textos (*bags*), uma vez que não se tem muitos conjuntos de dados de texto multi-instância atualmente, e pelos experimentos feitos acredita-se que a abordagem se mostra promissora no cenário de classificação de textos.

Referências Bibliográficas

Amores, J. Milde: multiple instance learning by discriminative embedding. *Knowledge and Information Systems*, v. 42, n. 2, p. 381–407, 2015.

Andrews, S.; Tsochantaridis, I.; Hofmann, T. Support vector machines for multiple-instance learning. In: *Advances in Neural Information Processing Systems 15*, MIT Press, 2003, p. 561–568.

Belattar, K.; Mostefai, S. Cbir using relevance feedback: Comparative analysis and major challenges. In: *Computer Science and Information Technology (CSIT), 2013 5th International Conference on*, 2013, p. 317–325.

Chan, W. Y.; Qu, H.; Mak, W. H. Visualizing the semantic structure in classical music works. *Visualization and Computer Graphics, IEEE Transactions on*, v. 16, n. 1, p. 161–173, 2010.

Chang, C.-C.; Lin, C.-J. Libsvm: A library for support vector machines. *ACM Trans. Intell. Syst. Technol.*, v. 2, n. 3, p. 27:1–27:27, 2011.

Chavarro, A.; Camargo, J.; Gonzalez, F. Visualizing multimodal image collections. In: *Image, Signal Processing, and Artificial Vision (STSIVA), 2013 XVIII Symposium of*, 2013, p. 1–6.

Chen, C. *Information visualization: Beyond the horizon.* Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006.

Chen, G.; Wang, T.; Herrera, P. Relevance feedback in an adaptive space with one-class svm for content-based music retrieval. In: *Audio, Language and Image Processing, 2008. ICALIP 2008. International Conference on*, 2008, p. 1153–1158.

Chen, Y.; Bi, J.; Wang, J. Miles: Multiple-instance learning via embedded instance selection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, v. 28, n. 12, p. 1931–1947, 2006.

Chen, Y.; Wang, J. Z. Image categorization by learning and reasoning with regions. *J. Mach. Learn. Res.*, v. 5, p. 913–939, 2004.
Disponível em http://dl.acm.org/citation.cfm?id=1005332.1016789

Chen, Y.; Wang, J. Z.; Geman, D. Image categorization by learning and reasoning with regions. *Journal of Machine Learning Research*, v. 5, p. 913–939, 2004.

Cherkassky, V. The nature of statistical learning theory . *Neural Networks, IEEE Transactions on*, v. 8, n. 6, p. 1564–1564, 1997.

Choo, J.; Lee, H.; Kihm, J.; Park, H. ivisclassifier: An interactive visual analytics system for classification based on supervised dimension reduction. In: *Visual Analytics Science and Technology (VAST), 2010 IEEE Symposium on*, 2010, p. 27–34.

Cox, T.; Cox, M. *Multidimensional scaling*. Monographs on statistics and applied probability. Chapman & Hall/CRC, 2001.

Cuadros, A. M.; Paulovich, F. V.; Minghim, R.; Telles, G. P. Point Placement by Phylogenetic Trees and its Application to Visual Analysis of Document Collections. *IEEE Symposium on Visual Analytics Science and Technology*, p. 99–106, 2007.

Dalhuijsen, L.; Velthoven, L. V. Musicalnodes: The visual music library. In: *Proceedings of the 2010 International Conference on Electronic Visualisation and the Arts*, EVA'10, Swinton, UK, UK: British Computer Society, 2010, p. 232–236 (*EVA'10*,).
Disponível em http://dl.acm.org/citation.cfm?id=2227180.2227214

Dietterich, T. G.; Lathrop, R. H.; Lozano-Perez, T.; Pharmaceutical, A. Solving the multiple-instance problem with axis-parallel rectangles. *Artificial Intelligence*, v. 89, p. 31–71, 1997.

Dou, W.; Yu, L.; Wang, X.; Ma, Z.; Ribarsky, W. Hierarchicaltopics: Visually exploring large text collections using topic hierarchies. *Visualization and Computer Graphics, IEEE Transactions on*, v. 19, n. 12, p. 2002–2011, 2013.

Eler, D.; Nakazaki, M.; Paulovich, F.; Santos, D.; Oliveira, M.; Neto, J.; Minghim, R. Multidimensional visualization to support analysis of image collections.

In: *Computer Graphics and Image Processing, 2008. SIBGRAPI '08. XXI Brazilian Symposium on*, 2008a, p. 289–296.

Eler, D.; Paulovich, F.; Oliveira, M.; Minghim, R. Coordinated and multiple views for visualizing text collections. In: *Information Visualisation, 2008. IV '08. 12th International Conference*, 2008b, p. 246–251.

Eler, D. M.; Nakazaki, M. Y.; Paulovich, F. V.; Santos, D. P.; Andery, G.; Oliveira, M. F.; Batista Neto, J.; Minghim, R. Visual analysis of image collections. *The Visual Computer*, v. 25, n. 10, p. 923–937, 2009.

Faceli, K.; Lorena, A. C.; Gama, J. M. P.; Carvalho, A. C. P. L. F. *Inteligência Artificial - Uma Abordagem de Aprendizado de Máquina*. LTC, 2011.

Foody, G. M.; Mathur, A. The use of small training sets containing mixed pixels for accurate hard image classification: Training on mixed spectral responses for classification by a {SVM}. *Remote Sensing of Environment*, v. 103, n. 2, p. 179 – 189, 2006.
Disponível em http://www.sciencedirect.com/science/article/pii/S0034425706001350

Fruchterman, T. M. J.; Reingold, E. M. Graph drawing by force-directed placement. *Softw. Pract. Exper.*, v. 21, n. 11, p. 1129–1164, 1991.

Fu, Z.; Robles-Kelly, A. An instance selection approach to multiple instance learning. In: *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, 2009, p. 911–918.

Fu, Z.; Robles-Kelly, A.; Zhou, J. Milis: Multiple instance learning with instance selection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, v. 33, n. 5, p. 958–977, 2011.

Ghani, S.; Kwon, B. C.; Lee, S.; Yi, J. S.; Elmqvist, N. Visual analytics for multimodal social network analysis: A design study with social scientists. *Visualization and Computer Graphics, IEEE Transactions on*, v. 19, n. 12, p. 2032–2041, 2013.

Heimerl, F.; Koch, S.; Bosch, H.; Ertl, T. Visual classifier training for text document retrieval. *Visualization and Computer Graphics, IEEE Transactions on*, v. 18, n. 12, p. 2839–2848, 2012.

HOFFMAN, P. E.; GRINSTEIN, G. G. Information visualization in data mining and knowledge discovery. cáp. A Survey of Visualizations for High-dimensional Data Mining, San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., p. 47–82, 2002. Disponível em http://dl.acm.org/citation.cfm?id=383784.383790

HUANG, Y.; ZHANG, W.; WANG, J. Multiple instance learning with correlated features. In: *Information Science and Technology (ICIST), 2012 International Conference on*, 2012, p. 441–446.

JOIA, P.; PAULOVICH, F.; COIMBRA, D.; CUMINATO, J.; NONATO, L. Local affine multidimensional projection. *Visualization and Computer Graphics, IEEE Transactions on*, v. 17, n. 12, p. 2563–2571, 2011.

JOLLIFFE, I. T. *Principal component analysis*. New York: Springer, 2002.

KEIM, D.; KRIEGEL, H.-P. Visualization techniques for mining large databases: a comparison. *Knowledge and Data Engineering, IEEE Transactions on*, v. 8, n. 6, p. 923–938, 1996.

LANGLEY, P. *Elements of machine learning*. Morgan Kaufmann Publishers Inc., 1996.

LI, W.-J.; YEUNG, D.-Y. Mild: Multiple-instance learning via disambiguation. *Knowledge and Data Engineering, IEEE Transactions on*, v. 22, n. 1, p. 76–89, 2010.

LICHMAN, M. UCI machine learning repository. 2013. Disponível em http://archive.ics.uci.edu/ml

LUXBURG, V. U.; SCHOLKOPF, B. Statistical Learning Theory: Models, Concepts, and Results. In: *Handbook of the History of Logic Vol. 10: Inductive Logic*, v. 10, Amsterdam, Netherlands: Elsevier North Holland, p. 651 – 706, 2011. Disponível em http://www.sciencedirect.com/science/publication?issn=18745857volume=10

MARON, O.; LOZANO-PÉREZ, T. A framework for multiple-instance learning. In: *Advances in Neural Information Processing Systems*, MIT Press, 1998, p. 570–576.

MARON, O.; RATAN, A. L. Multiple-instance learning for natural scene classification. In: *In The Fifteenth International Conference on Machine Learning*, Morgan Kaufmann, 1998, p. 341–349.

MARTINS, R. M.; ANDERY, G. F.; HEBERLE, H.; PAULOVICH, F. V.; ANDRADE LOPES, A.; PEDRINI, H.; MINGHIM, R. Multidimensional projections for visual analysis

of social networks. *Journal of Computer Science and Technology*, v. 27, n. 4, p. 791–810, 2012.

Mason, L.; Baxter, J.; Bartlett, P.; Frean, M. Boosting algorithms as gradient descent. In: *In Advances in Neural Information Processing Systems 12*, MIT Press, 2000, p. 512–518.

Migut, M.; Worring, M. Visual exploration of classification models for risk assessment. In: *Visual Analytics Science and Technology (VAST), 2010 IEEE Symposium on*, 2010, p. 11–18.

Minghim, R.; Paulovich, F. V.; Andrade Lopes, A. Content-based text mapping using multi-dimensional projections for exploration of document collections. *Visualization and Data Analysis*, v. 6060, 2006.

Mitchell, T. M. *Machine learning*. 1 ed. New York, NY, USA: McGraw-Hill, Inc., 1997.

Nguyen, Q.; Huang, M. L. A space-optimized tree visualization. *IEEE Symposium on Information Visualization, 2002. INFOVIS 2002.*, v. 2002, p. 85–92, 2002.

Paiva, J. *Técnicas computacionais de apoio à classificação visual de imagens e outros dados*. Tese de Doutoramento, Universidade de São Paulo, 2013.

Paiva, J.; Florian, L.; Pedrini, H.; Telles, G.; Minghim, R. Improved similarity trees and their application to visual data classification. *Visualization and Computer Graphics, IEEE Transactions on*, v. 17, n. 12, p. 2459–2468, 2011.

Paiva, J.; Schwartz, W.; Pedrini, H.; Minghim, R. An approach to supporting incremental visual data classification. *Visualization and Computer Graphics, IEEE Transactions on*, v. 21, n. 1, p. 4–17, 2015.

Paiva, J. G. S.; Schwartz, W. R.; Pedrini, H.; Minghim, R. Semi-supervised dimensionality reduction based on partial least squares for visual analysis of high dimensional data. *Comp. Graph. Forum*, v. 31, n. 3pt4, p. 1345–1354, 2012.

Paulovich, F. *Mapeamento de dados multi-dimensionais integrando mineração e visualização*. Tese de Doutoramento, Universidade de São Paulo, 2008.

Paulovich, F.; Minghim, R. Hipp: A novel hierarchical point placement strategy and its application to the exploration of document collections. *Visualization and Computer Graphics, IEEE Transactions on*, v. 14, n. 6, p. 1229–1236, 2008.

Paulovich, F. V.; Eler, D. M.; Poco, J.; Botha, C. P.; Minghim, R.; Nonato, L. G. Piecewise laplacian-based projection for interactive data exploration and organization. In: *Proceedings of the 13th Eurographics / IEEE - VGTC Conference on Visualization*, EuroVis'11, Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2011, p. 1091–1100 (*EuroVis'11*,).

Paulovich, F. V.; Nonato, L. G.; Minghim, R.; Levkowitz, H. Least square projection: A fast high-precision multidimensional projection technique and its application to document mapping. *IEEE Transactions on Visualization and Computer Graphics*, v. 14, n. 3, p. 564–575, 2008.

Paulovich, F. V.; Oliveira, M. C. F.; Minghim, R. The Projection Explorer: A Flexible Tool for Projection-based Multidimensional Visualization. *XX Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI 2007)*, p. 27–36, 2007.

Pipanmaekaporn, L. Feature discovery in relevance feedback using pattern mining. In: *Computer and Information Science (ICIS), 2013 IEEE/ACIS 12th International Conference on*, 2013, p. 301–307.

Ray, S.; Craven, M. Supervised versus multiple instance learning: an empirical comparison. In: *Proceedings of the 22nd international conference on Machine learning*, ICML '05, 2005a, p. 697–704 (*ICML '05*,).

Ray, S.; Craven, M. Supervised versus multiple instance learning: An empirical comparison. In: *Proceedings of 22nd International Conference on Machine Learning (ICML-2005*, ACM Press, 2005b, p. 697–704.

Reynolds, D. A.; Quatieri, T. F.; Dunn, R. B. Speaker verification using adapted gaussian mixture models. In: *Digital Signal Processing*, 2000, p. 2000.

Rüger, S. Putting the user in the loop: Visual resource discovery. In: Detyniecki, M.; Jose, J.; Nürnberger, A.; Rijsbergen, C., eds. *Adaptive Multimedia Retrieval: User, Context, and Feedback*, v. 3877 de *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, p. 1–18, 2006.

Saitou, N.; Nei, M. The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Molecular biology and evolution*, v. 4, n. 4, p. 406–25, 1987.

Shen, C.; Jiao, J.; Yang, Y.; Wang, B. Multi-instance multi-label learning for automatic tag recommendation. In: *Systems, Man and Cybernetics, 2009. SMC 2009. IEEE International Conference on*, 2009, p. 4910–4914.

Shen, Z.; Ma, K.-L.; Eliassi-Rad, T. Visual analysis of large heterogeneous social networks by semantic and structural abstraction. *IEEE Transactions on Visualization and Computer Graphics*, v. 12, n. 6, p. 1427–1439, 2006.

Sivakamasundari, G.; Seenivasagam, V. Different relevance feedback techniques in cbir: A survey and comparative study. In: *Computing, Electronics and Electrical Technologies (ICCEET), 2012 International Conference on*, 2012, p. 1115–1121.

Soukup, T.; Davidson, I. *Visual Data Mining: Techniques and Tools for Data Visualization and Mining* . John Wiley y Sons, 2002.
Disponível em http://www.amazon.com/Visual-Data-Mining-Techniques-Visualization/dp/0471149993

Tejada, E.; Minghim, R.; Nonato, L. G. On improved projection techniques to support visual exploration of multidimensional data sets. *Information Visualization*, v. 2, p. 218–231, 2003.

Tenenbaum, J. B.; Silva, V.; Langford, J. C. A global geometric framework for nonlinear dimensionality reduction. *Science*, v. 290, p. 2319–2323, 2000.

Torgerson, W. Multidimensional scaling of similarity. *Psychometrika*, v. 30, n. 4, p. 379–393, 1965.

Torrens, M.; Arcos, J.-l. Visualizing and exploring personal music libraries. In: *In Proceedings of the 5th International Conference on Music Information Retrieval (ISMIRŠ04*, 2004, p. 421–424.

Venkatesan, R.; Chandakkar, P.; Li, B.; Li, H. Classification of diabetic retinopathy images using multi-class multiple-instance learning based on color correlogram features. In: *Engineering in Medicine and Biology Society (EMBC), 2012 Annual International Conference of the IEEE*, 2012, p. 1462–1465.

Vijayakumar, S.; D'Souza, A.; Schaal, S. Incremental online learning in high dimensions. *Neural Computation*, v. 17, n. 12, p. 2602–2634, 2005.

Viola, P.; Platt, J. C.; Zhang, C. Multiple instance boosting for object detection. In: *In NIPS 18*, MIT Press, 2006, p. 1419–1426.

Wang, Y.; Zhang, C.; Wang, Z. Rate distortion multiple instance learning for image classification. In: *Image Processing (ICIP), 2013 20th IEEE International Conference on*, 2013, p. 3235–3238.

Wold, H. Partial least squares. *Encyclopedia of Statistical Sciences*, v. 6, p. 581–591, 1985.

Xiao, Y.; Liu, B.; Cao, L.; Yin, J.; Wu, X. Smile: A similarity-based approach for multiple instance learning. In: *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, 2010, p. 589–598.

Xiao, Y.; Liu, B.; Hao, Z.; Cao, L. A similarity-based classification framework for multiple-instance learning. *Cybernetics, IEEE Transactions on*, v. PP, n. 99, p. 1–1, 2013.

Xiao, Y.; Liu, B.; Hao, Z.; Cao, L. A similarity-based classification framework for multiple-instance learning. *Cybernetics, IEEE Transactions on*, v. 44, n. 4, p. 500–515, 2014.

Xu, P.; Wu, Y.; Wei, E.; Peng, T.-Q.; Liu, S.; Zhu, J. J.; Qu, H. Visual analysis of topic competition on social media. *IEEE Transactions on Visualization and Computer Graphics (Proceedings of IEEE VAST 2013*, v. 19, n. 12, 2013.

Xu, X.; Li, B. Evaluating multi-class multiple-instance learning for image categorization. In: Yagi, Y.; Kang, S.; Kweon, I.; Zha, H., eds. *Computer Vision – ACCV 2007*, v. 4844 de *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, p. 155–165, 2007a.

Xu, X.; Li, B. Multiple class multiple-instance learning and its application to image categorization. *Int. J. Image Graphics*, 2007b.

Xu, Y.; Hong, W.; Chen, N.; Li, X.; Liu, W.; Zhang, T. Parallel filter: A visual classifier based on parallel coordinates and multivariate data analysis. In: Huang, D.-S.; Heutte, L.; Loog, M., eds. *Advanced Intelligent Computing Theories and Applications. With Aspects of Artificial Intelligence*, v. 4682 de *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, p. 1172–1183, 2007.

Xu, Y.-Y.; Shih, C.-H. Image classification via multiple-instance decision-based neural networks. In: *Image and Graphics (ICIG), 2013 Seventh International Conference on*, 2013, p. 394–399.

Yang, J.; Fan, J.; Hubball, D.; Gao, Y.; Luo, H.; Ribarsky, W.; Ward, M. Semantic image browser: Bridging information visualization with automated intelligent image analysis. In: *Visual Analytics Science And Technology, 2006 IEEE Symposium On*, 2006, p. 191–198.

Yuan, L.; Liu, S.; Huang, Q.; Liu, J.; Tang, X. Salient instance selection for multiple-instance learning. In: Huang, T.; Zeng, Z.; Li, C.; Leung, C., eds. *Neural Information Processing*, v. 7665 de *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, p. 58–67, 2012.

Zafra, A.; Gibaja, E.; Ventura, S. Multiple instance learning with multiobjective genetic programming for web mining. In: *Hybrid Intelligent Systems, 2008. HIS '08. Eighth International Conference on*, 2008, p. 513–518.

Zhang, B.; Wang, Y.; Wang, W. Multiple-instance learning from multiple perspectives: Combining models for multiple-instance learning. In: *Applications of Computer Vision (WACV), 2012 IEEE Workshop on*, 2012a, p. 481–487.

Zhang, K.-B.; Orgun, M.; Shankaran, R.; Zhang, D. Interactive visual classification of multivariate data. In: *Machine Learning and Applications (ICMLA), 2012 11th International Conference on*, 2012b, p. 246–251.

Zhang, M.-L.; Zhou, Z.-H. Multi-instance clustering with applications to multi-instance prediction. *Applied Intelligence*, v. 31, n. 1, p. 47–68, 2009.

Zhang, Q.; Goldman, S. A. Em-dd: An improved multiple-instance learning technique. In: *In Advances in Neural Information Processing Systems*, MIT Press, 2001, p. 1073–1080.

Zhang, Q.; Goldman, S. A.; Yu, W.; Fritts, J. E. Content-based image retrieval using multiple-instance learning. In: *in Proc. 19th International Conference on Machine Learning*, Morgan Kaufmann, 2002, p. 682–689.

Zhao, J.; Collins, C.; Chevalier, F.; Balakrishnan, R. Interactive exploration of implicit and explicit relations in faceted datasets. *Visualization and Computer Graphics, IEEE Transactions on*, v. 19, n. 12, p. 2080–2089, 2013.

Zhou, Z.-h.; Zhang, M.-l. Multi-instance multi-label learning with application to scene classification. In: *In Advances in Neural Information Processing Systems 19*, 2007a.

Zhou, Z.-H.; Zhang, M.-L. Solving multi-instance problems with classifier ensemble based on constructive clustering. *Knowledge and Information Systems*, v. 11, n. 2, p. 155–170, 2007b.

Zhu, L.; Zhao, B.; Gao, Y. Multi-class multi-instance learning for lung cancer image classification based on bag feature selection. In: *Fuzzy Systems and Knowledge Discovery, 2008. FSKD '08. Fifth International Conference on*, 2008, p. 487–492.