CLASSIFICAÇÃO MULTI-RÓTULO HIERÁRQUICA AUTOMÁTICA DE DOCUMENTOS TEXTUAIS

GUSTAVO HENRIQUE ORAIR

CLASSIFICAÇÃO MULTI-RÓTULO HIERÁRQUICA AUTOMÁTICA DE DOCUMENTOS TEXTUAIS

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação do Instituto de Ciências Exatas da Universidade Federal de Minas Gerais como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação.

ORIENTADOR: WAGNER MEIRA JR.

Belo Horizonte, Minas Gerais

Julho de 2009

Orair, Gustavo Henrique
063c Classificação Multi-rótulo Hierárquica Automática de documentos textuais / Gustavo Henrique Orair. — Belo Horizonte, Minas Gerais, 2009
xxiii, 97 f. : il. ; 29cm

Dissertação (mestrado) — Universidade Federal de Minas Gerais
Orientador: Wagner Meira Jr.

1. Computação - Teses. 2. Recuperação da Informação - Teses. I. Orientador. II. Título.

CDU 519.6*73(043)

UNIVERSIDADE FEDERAL DE MINAS GERAIS
INSTITUTO DE CIÊNCIAS EXATAS
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

FOLHA DE APROVAÇÃO

Classificação Multi-rótulo Hierárquica de Documentos Textuais

GUSTAVO HENRIQUE ORAIR

Dissertação defendida e aprovada pela banca examinadora constituída pelos Senhores:

PROF. WAGNER MEIRA JÚNIOR - Orientador
Departamento de Ciência da Computação - UFMG

PROF. MARCOS ANDRÉ GONÇALVES - Co-orientador
Departamento de Ciência da Computação - UFMG

PROF. LEONARDO CHAVES DUTRA DA ROCHA
Departamento de Ciência da Computação - UFSJ

PROFA. GISELE LOBO PAPPA
Departamento de Ciência da Computação - UFMG

Belo Horizonte, 29 de julho de 2009.

Dedico este trabalho primeiramente à sociedade brasileira, principal financiadora deste trabalho.

A meus pais, José Orair da Silva e Vera Lúcia Rocha Silva que me educaram e ensinaram os valores a seguir que permitiram chegar a este degrau. À Andréia pela dedicação na travessia. A meus irmãos pelo exemplo de competência e inteligência que sempre persegui. A meu sobrinho Henrique que com certeza representou um novo alento.

Aos professores Wagner Meira Júnior e Mohammed Zaki pela orientação e oportunidade de desenvolver um trabalho tão interessante e desafiador, contribuindo para uma sólida formação além da rica experiência internacional vivida.

Agradecimentos

O fim de uma jornada é sem dúvida o momento ideal para agradecermos às pessoas que nos acompanharam durante a jornada, porém muitas que se participaram desta jornada desde o início podem acabar injustamente não mencionadas. Estes, sabem quem são e espero que sinceramente me perdoem pela omissão de seus nomes.

Agradeço a meus orientadores, Wagner Meira Jr. e Marcos André Gonçalves, por sempre me instigarem à busca. Sem dúvida, o papel de orientador, não é uma tarefa fácil e não se resume a apenas à transferência de conhecimento. Agradeço por terem tido a coragem de assumir um papel tão importante neste projeto.

Agradeço à sociedade brasileira, em particular ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) e à Fundação de Desenvolvimente da Pesquisa (FUNDEP), pelo investimento e suporte.

Agradeço à minha família, mãe Vera, pai José, irmãos Guilherme e Rodrigo, sem dúvida foi no convívio familiar que aprendi a buscar sempre o melhor e é nele que nos apoiamos nos momentos mais difíceis. A meu sobrinho Henrique, que me ensinou a reviver as pequenas coisas da vida dando novo ânimo a minha vida. Agradeço a minha namorada Andréia, sempre companheira, pelo amor e dedicação, que suportou a falta de tempo, o afastamento das viagens, os estresses do trabalho, sempre com perseverança, pela motivação e apoio nos momentos necessários.

Agradeço a meu co-orientador Mohammed Zaki, por ter me convidado à visita ao Rensselaer Polytechnic Institute e aos grandes amigos que fiz neste curto período de 4 meses de trabalho, Hilmi Yildrim, Saeed Salem e Mohammad Al Hasan. Sem dúvida, no convívio com pessoas tão diferentes, tenho certeza que aprendi bastante não só na área técnica quanto como experiência de vida. Adquiri bastante maturidade nas diversas conversas, tutoriais, palestras vividas em terras estado-unidenses. Em especial agradeço a Mohammad Al Hasan pelas diversas discussões sempre instigantes e extremamente valiosas.

Não poderia deixar de agradecer ao Fábio Soares Figureiredo, pelas diversas discussões, o grande auxílio ao desenvolvimento do trabalho, e pelo seu espírito sempre

instigador que sem dúvida me nortearam.

Agradeço aos vários amigos do DCC e do Speed com quem convivi neste tempo de aprendizado e que compartilharam vários dos problemas do dia a dia, Carlos, Fireman, Hawks, George, Coutos, Sachetto, Charles, Arlei, Macambira, Hélio, Fhmourão, Antônio, David dentre vários outros não aqui nomeados mas não menos importantes.

Agradeço aos amigos do Gueto do Cabral pelas inúmeras quintas feiras tão importantes para descontrair, Roberto, Rui, Logan, Farrer, Sílvio, Turmalina, PL, Luis, Vitor, Gush e outros.

Aos amigos do bairro, Lucas, Matheus, Miquim, Vinny, Whisky, Marcelim, Pedro Paulo, Lelê, Gegê, Pedro, GuiDugli, Pablo, Artur, Júlia, Laura, Adriano, Júnia, Yasmim e vários outros. Aos SomeNozes, Farley, Jorge, Rieves, Lud, Mimis, Tati, Flavinha, Nhega, Fabi e vários outros. Ao pessoal do Sebrae, Tonhão (de novo), Luisa, Japonês, Peruca, Forrest, Dri, Cavalo, Amorim, Miri, Jejé e vários outros.

Por fim, a todos os demais que mesmo não estando pessoalmente nomeados aqui, sabem que fizeram parte deste processo.

“ O sucesso nasce do querer, da determinação e persistência em se chegar a um objetivo.
Mesmo não atingindo o alvo, quem busca e vence obstáculos, no mínimo fará coisas admiráveis.”
(José de Alencar)

Resumo

A quantidade de informações armazenadas em bases de dados de documentos textuais aumenta cada vez mais. Esse crescimento demanda métodos automáticos para organização destes dados. Neste contexto, o estudo da classificação automática de textos tem merecido bastante atenção tanto no meio acadêmico quanto no mercado.

A maioria dos trabalhos sobre a classificação estuda o desenvolvimento de técnicas de classificação de textos em que existem um número limitado de classes e a dependência entre as classes não é expressiva. Existem vários cenários de aplicação relevantes em que estas premissas não são válidas. Para solucionar tais problemas, um novo tópico de pesquisa, a Classificação Multi-rótulo Hierárquica (HMC) vem sendo continuamente estudado mas ainda representa um grande desafio para a área.

Nos problemas de HMC, o conjunto de classes tende a ser muito maior e estas estão organizadas segundo uma estrutura hierárquica. Os métodos tradicionais, além de ignorar o conhecimento existente nesta estrutura, degeneram o desempenho tanto se o número de classes é expressivo quanto se existe interdependência entre estas classes.

Neste trabalho realizamos um extensivo estudo da literatura, desenvolvemos um arcabouço, o MASSIFICA, para o desenvolvimento e análise de métodos e propomos um algoritmo baseado em regras de classificação postergada para o problema de HMC. O *Massifica* foi utilizado para a avaliação do desempenho do algoritmo proposto e de sistemas de classificação a partir de classificadores base tradicionais baseando-se tanto na arquitetura plana quanto na arquitetura *top-down*. Apresentamos os resultados em um cenário de aplicação importante de classificação de atividades econômicas de empresas.

Por fim, realizamos uma discussão dos principais desafios e como as diferentes soluções resolvem ou falham na presença destes desafios. Concluímos que o novo algoritmo proposto, apesar de apresentar um desempenho inferior nos primeiros níveis da hierarquia, consegue um desempenho competitivo principalmente nos níveis mais profundos da hierarquia, em que, em geral, as classes são raras e existe menor quantidade de informação.

Abstract

The amount of information stored in text databases is steadily increasing. As such, demand for automated techniques to organize this data also continues to grow. In this context, academic and industry research has been focused on the study of automatic text classification.

Most work on text classification studies the development of techniques in which there are a limited number of classes and dependencies between them is not significant. There are several relevant application scenarios in which these assumptions are not valid. To solve these problems, a new research topic, the Multi-label Hierarchical Classification (HMC) has received more attention but still represents a major challenge for the area.

In HMC problems, the set of classes is likely to be much greater and, as such, they are hierarchically structured. Classic methods, in addition to ignore the existing structure knowledge, have their performance degradated if the number of classes is too large or interdependence between the classes exists.

In this work we perform an extensive literature study, present a framework targeting development and analysis of HMC algorithms, the MASSIFICA, and propose a lazy classification rule-based algorithm suitable for HMC problems. *MASSIFICA* was used as benchmark to evaluate performance of a proposed algorithm against well known base classifiers based on both flat architecture and structured database (top-down) architectures. We also present results applied to a real application scenario: classification of companies' economic activities.

Finally, we discuss challenges and how different solutions react to these challenges. We conclude that the new algorithm, despite having a lower performance in the first hierarchical levels, can perform competitively, particularly in the deeper levels of the hierarchy, which in general classes are uncommon and less information is provided.

Lista de Figuras

Lista de Tabelas

Sumário

Capítulo 1

Introdução

A quantidade de informações armazenadas em bases de dados vem aumentando cada vez mais. Esse crescimento inviabiliza a compreensão e descoberta de conhecimento e informações úteis de forma trivial. Surge a necessidade de formas automatizadas de análise das informações, assim, cresce a importância da *Mineração de Dados* e da *Descoberta de Conhecimento*. Esses campos têm demandado não somente algoritmos mais eficientes, como também diferentes soluções para novos problemas que surgem ou que passam a demandar mais recursos ou maior eficiência.

Um dos problemas mais estudados recentemente na área de Mineração de Dados, Aprendizagem de Máquina e Inteligência Artificial é o problema da classificação automática de textos. Dada a complexidade deste problema, uma solução proposta é o desenvolvimento de técnicas de aprendizagem de máquina de forma que dados alguns exemplos, tais técnicas construam um modelo para realizar a predição de novos documentos desconhecidos.

Várias técnicas de aprendizagem de máquina já foram propostas para a Classificação Automática de documentos. Hofmann et al. [2003] argumenta em seu trabalho que existem três direções para a melhora do estado da arte de tais técnicas: o desenvolvimento de melhores algoritmos, a obtenção de melhor representação de atributos e/ou a utilização de conhecimento adicional. A última dessas direções, a utilização de conhecimento adicional tem se mostrado uma boa direção de pesquisa, e inclui a exploração de interdependências entre documentos[Veloso et al., 2006], de hierarquias de classes [Lima et al., 1998; Wu et al., 2005] dentre outros. Neste sentido, Manning et al. [2008] cita que a busca por melhor desempenho dos sistemas de classificação, muitas vezes, não pode se concentrar apenas no estudo de diferentes métodos de aprendizagem de máquina, mas também em discutir as diferentes características dos documentos textuais e do sistema de classificação. Manning et al. [2008] afirmam ainda que é extremamente frequente

alcançar maiores ganhos de desempenho explorando-se as características específicas do domínio.

O problema da classificação na presença de um número maior de classes, tanto manual quanto automática, costuma complicar o problema sensivelmente. Para contrapor esta complexidade, é comum a criação de uma ontologia, taxonomia ou hierarquia de classes. Diante disto, torna-se necessário o desenvolvimento de técnicas de classificação que incorporem a hierarquia durante o aprendizado, uma vez que ignorar a existência de inúmeras classes e a semântica e/ou as dependências entre as classes pode diminuir significativamente a acurácia das técnicas de classificação.

O presente trabalho objetiva o estudo das diferentes técnicas de categorização de textos em problemas em que o número de classes é extenso, e por esta razão, existe uma hierarquia de classes. Este problema é conhecido como Classificação hierárquica de textos.

1.1 Objetivo

Neste trabalho pretende-se estudar os algoritmos de classificação hierárquica de textos.

Os principais objetivos deste trabalho são:

1. Apresentação de uma extensa revisão bibliográfica sobre o problema da Classificação Automática Hierárquica Multi-rótulos de Documentos;

2. Identificação das principais dificuldades inerentes aos problemas de HMC;

3. Apresentação do estado da arte das diferentes técnicas aplicadas ao problema de HMC;

4. Proposta de uma técnica desenvolvida especificamente para os problemas de HMC;

5. Avaliação e validação das técnicas e de diferentes propostas por meio de métricas como acurácia e eficiência.

1.2 Cenários de Aplicação

Existem inúmeros exemplos de aplicações modernas para o problema da HMC que vão desde a categorização de textos (organização de diretórios de páginas, organização de artigos de um jornal, de documentos médicos, de artigos científicos, categorização de

músicas), classificação de funções de proteínas, previsão de funções de genes (functional genomics) até a classificação semântica de cenários (semantic scene classification).

- Sugestão automática de rótulos Recomendação automática de títulos favoritos em redes sociais [Tatu et al., 2008].
  - Bookmarks
  - Bibtex
- Anotação
  - Anotação/Predição automática de função genética
- Indexação automática de palavras-índices Existem vários contextos em que é necessário a atribuição automática de palavras-índice a um documento. Essas palavras índice geralmente pertencem a um vocabulário controlado de palavras que definem o tema ou objetivo que pretende-se qualificar o documento. Assim, uma das alternativas viáveis é considerar cada palavra do vocabulário controlado com um rótulo válido.
- Organização de Documentos
  - WIPO-alpha
  - Yahoo Directories
  - Newspaper articles IPTC Code (International Press and Telecomunication Code)
  - E-mail management
  - IPC (International Patent Classification)
  - WordNet (base de conhecimento linguístico da linguagem inglesa)
  - MEsh (MEdical Subject Headings) - vocabulário controlado para publicações de artigos e livros na Medicina
  - ICD (Internation Code of Diseases) - código de doenças internacional utilizado para categorizar artigos médicos
  - Diretórios da ACM - base de dados de artigos da área de computação
- Classificação de atividades econômicas (ISIC/NAICS/CNAE)

  Uma outra nova interessante aplicação é a Classificação da Atividade Econômica de uma empresa de forma automática. Uma empresa para se constituir deve

elaborar um Contrato Social que descreve seu objeto social, isto é, descreve as atividades que a empresa realizará. Com base nesse objeto social o governo, nas diversas esferas, municipal, estadual e federal, precisa classificar a empresa de acordo com a tabela *CNAE subclasses*. Essa tabela descreve as atividades da empresa e um código específico de cada atividade e possui uma estrutura hierárquica onde os códigos de atividades são cada vez mais discriminativos. Pretende-se, dado um objeto social de uma empresa, classificar de forma automática qual ou quais são os códigos de atividade econômica daquela empresa.

1.3 Motivação das Hierarquias

As hierarquias são normalmente desenvolvidas quando há necessidade de organizar uma série de conceitos. Desenvolver uma estrutura hierárquica para auxiliar o entendimento, apresentação ou organização de vários conceitos é uma alternativa natural quando depara-se com um número muito grande de conceitos. A hierarquia facilita as tarefas de encontrar, analisar ou apresentar os conceitos

Um exemplo claro de como uma hierarquia de conceitos facilita o entendimento é o sucesso da especialização de classes possibilitada pelas linguagens de programação orientada a objetos como Java ou C++. A possibilidade de definir as classes por meio de especializações facilita o desenvolvimento da programação genérica. De fato, esta é uma das formas mais simples de implementar e se ensinar a programação genérica, uma vez que a intuição da geração e organização de conceitos por meio de hierarquias é muito comum e facilmente compreendida. A necessidade da criação de hierarquias para auxiliar na organização de documentos é facilmente encontrada em muitas aplicações, como por exemplo, em repositórios de páginas web, repositório de artigos, repositório de e-mails, bases de dados de patentes dentre outros inúmeros exemplos. A organização hierárquica permite ao usuário realizar diferentes enfoques em detalhes específicos a cada nível ou à medida em que se navega nesta hierarquia. Além disto, a própria estrutura hierárquica carrega uma relação clara de especialização entre os seus nodos que pode ser importante nas tarefas a serem realizadas.

Neste trabalho procura-se estudar como os sistemas de classificação automática de textos podem explorar as hierarquias manualmente desenvolvidas por especialistas. Estas hierarquias são geralmente construídas de forma totalmente independente da tarefa de classificação automática e são motivadas pela necessidade dos profissionais de organizar melhor os documentos. Hierarquias são encontradas sem muita dificuldade como nas páginas do diretório Yahoo Liu et al. [2005], nos diretórios de documen-

tos médicos como Medline Rak et al. [2005], dentre vários outras. As hierarquias são construídas definindo-se uma ontologia de conceitos que representam conhecimentos específicos do domínio de classificação dos documentos. Por exemplo, um repositório de notícias de um jornal podem ser categorizadas segundo seus tópicos ou segundo o setor da indústria sobre o qual a notícia relaciona-se, por esta razão, os especialistas constroem diferentes hierarquias cada uma refletindo conceitos relevantes a cada sistema de categorização e as relações entre as classes possuem semântica diferenciada de acordo com o objetivo de cada sistema de classificação[Lewis et al., 2004].

Os sistemas de classificação de textos podem se beneficiar da hierarquia de inúmeras formas diferentes. Assim, as hierarquias podem ser exploradas pelos sistemas de classificação automática em várias etapas, como pré-processamento, aprendizagem, pós-processamento, apresentação e avaliação. Por exemplo, caso a classificação em um domínio seja muito difícil, ao invés de tentar classificar os documentos nas classes mais profundas e neste caso incorrer em um alto risco de realizar previsões com baixa acurácia, podemos optar por classificar os documentos apenas nos níveis mais altos da hierarquia, e apresentar esta classificação para os especialistas como um auxílio na tarefa de classificação manual.

Existem uma série de algoritmos que propõem a construção de hierarquias automaticamente, tais técnicas se baseiam em algum conceito intrínseco de organização dos documentos para a criação das hierarquias. O estudo destas técnicas não está no escopo de nosso trabalho.

Capítulo 2

Fundamentos teóricos

2.1 O Problema da Classificação

A classificação de documentos é a tarefa de assinalar um conjunto de documentos D a um conjunto pré-definido de categorias C. A decisão de assinalar um documento d_j a uma classe c_i é representada atribuindo-se um rótulo l_i ao documento d_j. A categorização de textos é a tarefa de classificar documentos *textuais*. O enfoque deste trabalho é a classificação de documentos textuais e portanto, usaremos de forma análoga ao longo deste trabalho os termos *classificação* e *categorização*.

Definição 1 *A classificação* único-rótulo *é a tarefa de assinalar, para cada documento* $d_j \in D$*, um rótulo e somente um rótulo* c_i*,* $c_i \in C$ *e* $|C| > 1$*. Se* $|C| = 2$ *o problema é chamado de classificação binária, enquanto se* $|C| > 2$ *temos o problema de classificação* multi-classe[1].

Alguns problemas de classificação de documentos existe uma grande dificuldade na decisão de uma única classe ou categoria a que pertence o documento. Isto pode acontecer porque existe ambiguidade nos textos, os documentos podem conter vários assuntos ou mesmo assuntos multidisciplinares. Diante da necessidade de representar a ambiguidade ou multidisciplinaridade de assuntos foi formulado o problema da classificação *multi-rótulo*[2]. A classificação *multi-rótulo* permite que mais de um rótulo possa ser assinalado a cada documento:

[1] multi-categoria

[2] Diferentes termos são utilizados para designar este mesmo conceito como "qualquer-de classificação" (*"any-of classification"*) ou classificação multivalorada (*"multivalue classification"*).

Definição 2 *Classificação* multi-rótulo *("multi-label"): Dado um conjunto de documento D e um conjunto de classes C, a tarefa na classificação* multi-rótulo *é assinalar um conjunto de rótulos $C'_i \in C$ a cada documento $d_j \in D$.*

2.1.1 Aprendizado de Máquina

Um classificador precisa decidir uma função de assinalamento que, dados os documentos, consiga decidir a quais classes o documento pertence. A tarefa do classificador é portanto descobrir a função de assinalamento, chamada de *função objetivo* $\Phi : \mathcal{D} \times \mathcal{C} \rightarrow \mathbf{R}$, onde $\mathcal{D}$ é o conjunto de documentos e $\mathcal{C}$ define o conjunto pré-definido de classes. Em muitos problemas de classificação esta função objetivo é facilmente obtida e modelada. Por exemplo, imagine uma base de dados do jogo da velha (*tic tac toe*) com todas as entradas de jogos que faltam apenas uma única rodada para ser finalizada. Um jogador pode facilmente definir a decisão correta e portanto é possível modelar a função objetivo modelando-se todas as possibilidades.

Em problemas de classificação mais complexos, descobrir, definir ou modelar tal função pode ser uma tarefa extremamente difícil e/ou complexa, principalmente caso a função seja desconhecida a priori. Para solucionar estes casos foram propostas técnicas de aprendizagem de máquina para a classificação de documentos. Os três tipos principais de aprendizagem de máquina são:

- Aprendizagem não-supervisionada

  Modela um conjunto de entradas em que documentos já classificados não estão disponíveis;

- Aprendizagem de máquina semi-supervisionada

  Combina-se tanto documentos já assinalados quanto documentos não assinalados para a aprendizagem;

- Aprendizagem supervisionada

  A aprendizagem de máquina supervisionada consiste da aprendizagem de uma função aproximada da função objetivo a partir de uma base de dados de treino com documentos já assinalados. A base de dados de treino consiste de objetos de entrada (tipicamente vetores) e o resultado/saída. Se a função realiza o mapeamento para uma série de rótulos (valores categóricos) como resultado denominamos esta tarefa de classificação, se o resultado são valores contínuos denominamos como uma regressão. Para predizer uma função que realize este

mapeamento, a fase de aprendizagem precisa generalizar a partir dos dados atuais (documentos de treino) para situação não vistas de forma razoável. Essa generalização é conhecida por enviesamento indutivo (*"inductive bias"*).

Definição 3 *O viés de um algoritmo de aprendizado é o conjunto de premissas que um aprendiz se utiliza para predizer as saídas dadas as entradas apresentadas [Mitchell, 1980].*

A seguir apresentaremos alguns conceitos importantes focados na aprendizagem de máquina supervisionada resumida na figura 2.1.

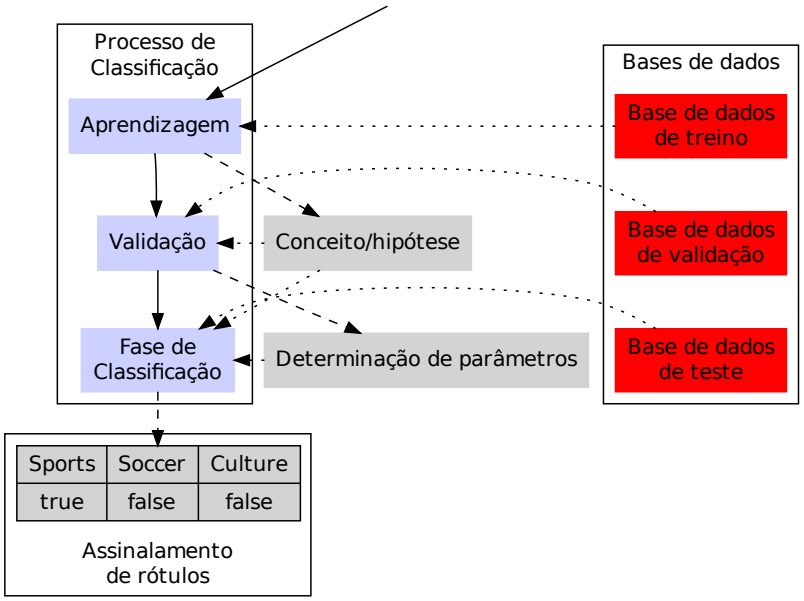


Figura 2.1: As fases do processo de classificação supervisionado

Na figura 2.1 apresentamos uma visão geral do processo de aprendizagem e classificação de rótulos. Este processo inicia-se na *fase de aprendizagem* em que uma *base de dados de treino* alimenta um algoritmo que com base nos documentos assinalados desta base constrói-se um *conceito*. Esta fase ainda inclui o pré-processamento em que os documentos são transformados em estruturas a serem utilizadas pelos algoritmos de aprendizagem de máquina. Posteriormente, o processo inicia-se a *fase de validação* em que é utilizada uma *base de dados de validação* para se decidir quais os parâmetros a serem utilizados no modelo, resultando-se ao final desta fase, na *determinação dos*

parâmetros do modelo. O processo então finaliza-se com a *fase de classificação* em que são realizados os *assinalamentos dos rótulos*.

Nas próximas seções apresentaremos o processo de aprendizagem de forma mais detalhada. Na seção 2.2 apresentaremos a definição do conceito que é construído por um algoritmo de aprendizagem de máquina, na seção 2.3 apresentamos as três fases que constituem o processo de classificação e discutimos alguns conceitos inerentes ao processo.

2.2 Construção de Conceito

Em algoritmos de classificação baseados em aprendizagem supervisionada, a tarefa é encontrar uma aproximação para uma função de assinalamento chamada de *função objetivo* desconhecida $\Phi : \mathcal{D} \times \mathcal{C} \rightarrow \mathbf{R}$, onde $\mathcal{D}$ é o conjunto de documentos e $\mathcal{C}$ define o conjunto pré-definido de classes. Tradicionalmente, o resultado R da função objetivo é um conjunto de valores booleanos indicando se o documento pertence ou não a cada classe, ou seja, $R = \{V, F\}$, onde o valor V representa a decisão de assinalar o documento $D_j \in \mathcal{D}$ à classe $C_i \in \mathcal{C}$ e o valor F representa a decisão de não assinalamento do documento $D_j \in \mathcal{D}$ à classe $C_i \in \mathcal{C}$. Alternativamente, o resultado R pode ser um conjunto de números reais que exprimam a probabilidade e/ou a confiança de que o documento $D_j \in \mathcal{D}$ pertença às classes $C_i \in \mathcal{C}$ [3]. A função objetivo Φ também pode ser chamada de conceito objetivo. A função aproximada $\phi : \mathcal{D} \times \mathcal{C} \rightarrow \mathbf{R}$ é chamada de *classificador* ou *hipótese* $\mathcal{H}$. Essa função objetivo precisa ser o mais coincidente possível à função objetivo Φ.

Em geral as técnicas diferem-se apenas quanto aos conceitos construídos pelo algoritmo, como árvores de decisão, vetores de suporte (*"support vectors"*), redes bayesianas, redes neuronais, dentre outros. Por esta razão, os algoritmos são discriminados com base nos conceitos que constroem.

2.3 Conjunto de Treinamento, validação e teste

A aprendizagem da aproximação da função objetivo Φ é realizada indutivamente, com base em um conjunto de treinamento composto de exemplos de documentos que foram previamente classificados. Esse conceito produzido deve ser consistente com os dados de treinamento.

[3] Neste segundo caso é necessária a definição de um limiar de assinalamento que mapeará um limite para a transformação do resultado finalmente no conjunto de valores booleanos $\{V, F\}$ de decisões de assinalamento.

Muitos dos conceitos construídos dependem de diversos parâmetros, como por exemplo, um limiar de assinalamento. A definição destes parâmetros faz parte da fase de indução ou aprendizagem. O ajuste de parâmetros realizado avaliando-se a base de treinamento pode gerar problemas sérios como o superajuste (*"overfitting"*) que será discutido na seção 2.7.1. Portanto, alguns algoritmos de aprendizagem demandam que se defina uma base de dados independente da *base de dados de treinamento*, a *base de dados de validação*. Os ajustes durante a *fase de validação* são realizados através de métricas de desempenho como a acurácia do modelo. Diversos valores de parâmetros são avaliados e a *determinação dos valores dos parâmetros* é realizada pela seleção dos valores que apresentarem melhor resultado segundo as métricas de desempenho.

Por fim, definimos a *base de testes* que, com base em métricas de desempenho, serve para mensurarmos a efetividade de cada técnica. As diversas métricas de desempenho que podem ser utilizadas serão discutidas na seção 2.4.2. Vale notar que existem trabalhos em que a própria base de testes é utilizada para a validação do modelo/conceito construído, tal metodologia é equivocada pois tendencia as métricas de desempenho uma vez que o correto assinalamento da base de testes é utilizado na *determinação dos parâmetros*. Assim, os resultados das métricas podem refletir um *superajuste* sobre a base de teste que não corresponde ao desempenho real da técnica.

Em geral, distinguimos as diferentes fases do processo de classificação em três fases distintas: aprendizagem, validação e classificação/assinalamento. Estas fases cumprem papéis diferentes durante o processo e têm o esforço computacional também bastante diferenciados. Cabe ressaltar que alguns algoritmos podem suprimir algumas destas fases, como os algoritmos de aprendizagem postergada em que a fase de treinamento é suprimida (seção 2.6.6).

2.4 Avaliação de classificadores

As técnicas de aprendizagem possuem diferentes características e pontos fortes e fracos distintos. Para que se possa estudar e comparar diferentes técnicas e métodos são necessários processos sistemáticos de avaliação de classificadores.

Nesta seção apresentaremos os principais métodos de amostragem e as principais medidas de qualidade da classificação que podem ser utilizadas para a avaliação do desempenho das técnicas de classificação.

2.4.1 Métodos de amostragem para avaliação de classificadores

Na seção 2.3 vimos a necessidade de se dividir a base de dados em base de treinamento, de validação e de teste. Veremos a seguir os diferentes métodos de amostragem que definem como é realizada esta divisão da base de dados.

- Amostragem Aleatória

  Neste método a base de dados de teste e de validação são definidas por amostragem aleatória. Primeiramente, constrói-se a base de testes por amostragem aleatória sem repetição. Em geral, a base de testes é definida com cerca de 10% do tamanho da base de dados total. Posteriormente, caso necessário, determina-se a base de validação também por amostragem aleatória sem repetição dos documentos que não compõem a base de testes. Os demais documentos não amostrados comporão a base de treinamento.

  Preferencialmente, várias execuções do método de amostragem aleatórias são realizadas definindo diversas bases de treinamento, validação e teste e modelos são construídos para cada amostragem. Cada execução produz diferentes valores das métricas de desempenho e métodos estatísticos são utilizados com base nas diferentes amostragens para determinar se um algoritmo é ou não superior a outro. Um problema deste método é que a definição do número de diferentes amostragens necessárias para obter resultados estatisticamente significantes pode ser difícil e o número de reamostragens necessário pode ser proibitivo.

- Validação cruzada (*"cross-validation"*)

  A validação cruzada é o método mais encontrado na literatura e se destaca por sua simplicidade e robustez. Neste método, subdivide-se a base de dados em k partições mutuamente exclusivas. Várias execução são realizadas em que cada execução utilizará uma única partição desta base como base de testes. modelos e utilizadas k diferentes bases de testes. Ou seja, a cada repetição da classificação uma parte diferente

  Recomenda-se também que a divisão em k diferentes conjuntos seja feita com base em amostragem aleatória, desta forma, evita-se que alguma tendência ou ordem previamente existente na base de dados interfira nos resultados. Note que caso haja necessidade de uma base de dados de validação, permite-se que a definição desta base seja feita por cada algoritmo de forma autônoma.

- *"Leave-one-out"*

O método *"leave-one-out"* é um caso especial da validação cruzada em que $k = |D|$. Ou seja, são gerados $|D|$ diferentes modelos e a cada modelo gerado testa-se o algoritmo de classificação com um único documento como base de testes. Este processo é extremamente custoso, apresenta complexidade computacional elevada, e portanto, só é viável em bases de dados pequenas. De fato, este método é mais utilizado quando percebe-se que a base de dados é tão pequena que a retirada de documentos da base de treino impacta significativamente o desempenho do algoritmo pois a quantidade de informação existente na base de treino torna-se insuficiente para uma boa aproximação da função objetivo Φ.

2.4.2 Medidas de qualidade da Classificação

Nesta seção, apresentaremos as principais medidas de classificação utilizadas na literatura para a avaliação dos classificadores tradicionais (único rótulo). Essas medidas são apresentadas de uma forma mais abrangente para que já possam ser utilizadas para a avaliação de classificadores multi-rótulo [Godbole & Sarawagi, 2004]. Seja Y_j o conjunto de rótulos reais assinalados do documento de teste d_j, $Y_j \subset C$, e Z_j é o conjunto de rótulos previstos pelo classificador para o documento d_j, $Z_j \subset C$ [4].

- Acurácia A acurácia mede a quantidade de rótulos cuja ocorrência ou não ocorrência foi prevista corretamente.

$$A(\mathcal{H}, D) = \frac{1}{|D|} \sum_{j=1}^{|D|} \frac{|Y_j \Delta Z_j|}{|C|} \tag{2.1}$$

  Onde Δ é a diferença simétrica entre os dois conjuntos e corresponde à operação XOR em lógica booleana.

- Precisão

  A precisão de um classificador mede a percentagem de rótulos previstos pelo classificador Z_i que são de fato rótulos corretos [5].

$$P(\mathcal{H}, D) = \frac{1}{|D|} \sum_{j=1}^{|D|} \frac{|Y_j \cap Z_j|}{|Z_j|} \tag{2.2}$$

- Revocação

[4] Na classificação único rótulo, $|Y_j| = 1$ e $|Z_j = 1|$.

[5] Em modelos de classificação único-rótulo a acurácia e a precisão serão medidas com valores idênticos.

Revocação (*"recall"*) mede a quantidade de rótulos previstos corretamente dentre todos os rótulos corretos.

$$R(\mathcal{H}, D) = \frac{1}{|D|} \sum_{i=1}^{|D|} \frac{|Y_i \cap Z_i|}{|Y_i|} \tag{2.3}$$

- Medida F

  A precisão e a revocação captam duas características diferentes e complementares em um sistema de classificação. Caso um sistema seja avaliado apenas pela sua precisão, pode-se construir um classificador que apenas assinale rótulos extremamente confiáveis, assim o número de rótulos corretos deste sistema de classificação será pequeno e portanto a sua revocação será muito baixa. Por outro lado, caso o sistema foque apenas na revocação, um classificador tenderá assinalar rótulos mesmo com *confiança* o que resultaria em um sistema com baixa precisão e alta revocação. Assim, o desenvolvimento de um bom sistema de classificação deve considerar este compromisso entre precisão e revocação. Por esta razão, propôs-se a utilização da medida F *("F-measure")* para refletir este compromisso.

$$F_\beta = \frac{(\beta^2 + 1) \times P \times R}{(\beta^2 \times P) + R}, \beta \in [0, +\infty] \tag{2.4}$$

- F-1

  A medida F é extremamente utilizada na literatura, em geral esta medida é utilizada com $\beta = 1$, o que corresponde à média harmônica entre a precisão e a revocação:

$$F_1 = \frac{2 \times R \times P}{R + P} \tag{2.5}$$

  Note que pode-se aumentar a importância da precisão utilizando valores de β menores ou aumentar a importância da revocação utilizando valores de β maiores. Em particular se $\beta = 0$ então $F_0 = P$ e se $\beta = \infty$ então $F_\infty = R$.

- Micro e Macro médias

  Muitas das medidas apresentadas acima são aplicadas a cada classe separadamente. É necessário então sumarizar tais medidas para cada classe de forma a avaliar o sistema como um todo. Existem duas formas de sumarização por médias, as medidas sumarizadas por médias Micro (*"Micro-averaging"*) atribuem às classes um peso proporcional à quantidade de documentos que possuem. Na

sumarização Macro (*"Macro-averaging"*) a média é calculada sem considerar o tamanho das classes, isto é, cada categoria recebe o mesmo peso. A diferença principal entre as duas formas de sumarização reside no favorecimento do desempenho nas classes frequentes ou infrequentes. Em geral, os sistemas de classificação conseguem melhor desempenho nas classes frequentes, como as médias Micro consideram o peso estabelecido de acordo com a frequência da classe, esta sumarização tende a privilegiar as classes frequentes. Porém, as classes menos frequentes costumam ser mais difíceis de classificar e um bom desempenho nestas classes tende a ser mais relevante que um bom desempenho em classes frequentes, mais fáceis. Por esta razão, em geral quando temos classes desbalanceadas a sumarização por *médias Macro* pode ser até mais importante que a sumarização por *médias Micro*.

2.5 Pré-processamento de Documentos

A categorização de textos é uma área da computação que dado um documento formado por palavras em linguagem natural deve-se assinalar rótulos representando a quais classes tais documentos pertencem. Para isso, em geral é necessário que as palavras sejam transformadas em uma série de atributos ou em um vetor para finalmente se realizar a classificação. Ou seja, os documentos textuais precisam passar por uma fase de pré-processamento para serem preparados e transformados em uma representação adequada para ser utilizada pelos algoritmos de classificação. Na figura 2.2 podemos ver um esquema das principais fases do *pré-processamento de documentos* que detalharemos nas seções 2.5.1 a 2.5.4.

2.5.1 Determinando o vocabulário de termos

A primeira fase do pré-processamento consiste de determinar quais os termos dos textos que comporão o modelo de classificação. Esta fase é composta de três processos principais: análise léxica, remoção de *"stopwords"* e normalização de variações morfológicas. Durante a *análise léxica* é executado o processo de *tokenização*, uma das primeiras atividades a serem executados em qualquer mineração de textos. Esse processo consiste em a partir da análise de uma sequência de caracteres, separar tais caracteres em símbolos, chamados de *"tokens"*. Neste processo, pode-se se for o caso, descartar caracteres dispensáveis como pontuação, *"tags html"*, dentre outros.

Após a *tokenização*, o próximo passo é a *remoção de "stopwords"*. As palavras cujo significado seja irrelevante para o contexto, como conjunções, artigos e preposições, são

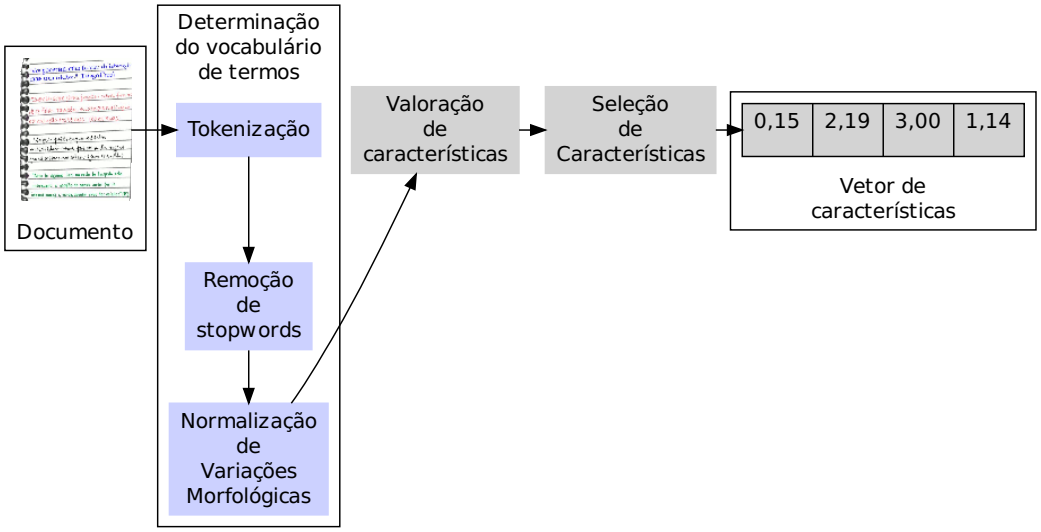


Figura 2.2: Principais fases do Pré-processamento de documentos

removidas. Esta atividade normalmente parte de uma lista pré-definida destas palavras a serem removidas. Vale lembrar que esta fase pode influir nas fases posteriores do processo de classificação, tanto diminuindo a carga computacional destas fases, quanto removendo possíveis ruídos e imprecisões nos modelos que serão gerados.

A *Normalização de Variações Morfológicas* pode constituir uma etapa importante do processo ou mesmo ser suprimida em alguns sistemas. Nesta etapa, também chamada de radicalização, lematização ou *"stemming"*, as palavras ou *"tokens"* encontrados no documento textual são transformados em seus radicais, lemas ou "stems". O resultado deste processo é que palavras derivadas do mesmo radical, por exemplo, sejam identificadas nos processos subsequentes como referentes ao mesmo termo ou tendo o mesmo significado.

2.5.2 O método *"bag of words"*

O método convencional de representação de textos para as tarefas de mineração é conhecido como *"bag of words"* (BOW). Esta *"sacola de palavras"* consiste de uma representação em que importa-se com a ocorrência das palavras mas ignora-se a ordem em que as palavras ocorrem no texto. Neste método cada termo t_k pertencente ao vocabulário corresponde a uma característica f_k e cada documento d_j é transformado em um vetor de características $\vec{f_j}$. Alternativamente, o vetor de características pode

ser gerado baseado em n-gramas ou sequências [Figueiredo, 2008].

2.5.3 Valoração de características

Em um documento textual, diferentes termos carregam uma semântica maior que outros termos. Assim, precisa-se identificar quais os termos potencialmente mais relevantes de um documento, esta tarefa é realizada pela *valoração de características*. Para cada termo t_k atribuímos um peso/valor f_k ao vetor de características correspondente à ocorrência do termo t_k no documento.

Existem diferentes métodos de *valoração de características*. A alternativa mais simples corresponde à atribuição de valores binários em que o valor das características é determinado como $\{0, 1\}$ onde $f_k = 1$ representa a ocorrência uma ou mais vezes do termo t_k no documento e $f_k = 0$ a não ocorrência do termo. Métodos mais complexos foram propostos para atender à necessidade de mensurar o poder discriminativo dos termos nos documentos. Um dos métodos mais utilizados é conhecido por $TF \times IDF$, neste método consideram-se dois fatores para a determinação da importância de um termo: a frequência do termo no documento (TF) e a raridade com que o termo ocorre na coleção de documentos (IDF).

$$TF \times IDF = TF(t_k, d_j) \times \log \frac{|D|}{DF(t_k)}$$

onde $TF(t_k, d_j)$ define o número de ocorrências do termo t_k no documento d_j, $DF(t_k)$ denota o número de documentos em que o termo t_k ocorre e $|D|$ representa o número total de documentos de treino na coleção.

Note que a valoração e características deve ser realizada considerando-se como a coleção de documentos, apenas a base de treinamento. Não deve-se utilizar a coleção completa de documentos para a valoração pois neste caso se estaria incluindo no modelo a ser construído conhecimento existente também na base de testes o que poderia tendenciar a avaliação em prol por exemplo de modelos com menor capacidade de generalização (a ser discutido na seção 2.7.1).

A importância e influência dos diferentes métodos de valoração de características para os sistemas de classificação não serão detalhadas. O leitor pode consultar um interessante trabalho em que é realizada a comparação de modelos de classificação *"naive bayes"* que se diferenciam pelos métodos de valoração de características [Mccallum & Nigam, 1998; Schneider, 2003].

2.5.4 Seleção de Características

A seleção de características é um processo que consiste na seleção de um subconjunto das características mais informativas que ocorrem na base de treinamento e apenas este subconjunto é então considerado durante o processo de classificação. O principal objetivo desta seleção é a eliminação de características que não sejam informativas e que causem ruído para o modelo de classificação o que pode aumentar a acurácia dos classificadores. Uma característica "poluidora" é uma característica que ao ser adicionada à representação do documento diminui a acurácia do modelo. Esta degradação pode ocorrer caso esta característica "poluidora" seja utilizada para uma generalização incorreta a partir de uma especificidade da base de treinos que não se aplica a novos dados[6]. Além da limpeza da base de dados com a remoção de *"ruído"*, a seleção de características também simplifica o modelo de classificação. Alguns autores ressaltam que a seleção de características é um método capaz de substituir um modelo de classificação mais complexo por um modelo de classificação mais simples, de forma a obter um conceito/hipótese menos suscetível ao superajuste, principalmente se existe limitada quantidade de dados de treinamento [Manning et al., 2008]. A seleção de características ainda diminui o tamanho do vocabulário a ser aplicado ao processo de aprendizagem e assinalamento o que diminui o esforço computacional destes processos. Em classificadores cujo processo de aprendizagem é pesado computacionalmente, este fator é particularmente mais relevante.

A seleção de características é portanto um dos processos mais importantes para os sistemas de classificação e influi direta e sensivelmente tanto no desempenho quanto na eficiência destes sistemas. Inclusive, é comum que diferentes trabalhos apresentem resultados experimentais diversos utilizando os mesmos algoritmos e as mesmas bases de dados e esta divergência é resultado de diferentes métodos de seleção de características aplicados [Chen et al., 2007]. Em particular, os algoritmos *Naive Bayes* baseados no modelo Bernoulli são extremamente sensíveis a características ruidosas e portanto ao se aplicar este modelo recomenda-se realizar uma efetiva seleção de características que melhorará significativamente o desempenho do sistema de classificação [Mccallum & Nigam, 1998; Manning et al., 2008].

[7].

[6] O superajuste é discutido na seção 2.7.1.

[7] Muitos autores realizam a seleção de características utilizando toda a base de dados, este processo não é considerado correto.

2.5.4.1 Critérios para a medida de utilidade

A seleção de características precisa encontrar as características mais informativas. Para isso, é necessário definir uma medida de utilidade $U(t_k)$ para cada característica/termo t_k que indique quais os termos mais informativos. Posteriormente, seleciona-se k melhores termos segundo as medidas de utilidade $U(t_k)$.

A seguir apresentamos uma lista não-exaustiva dos principais critérios utilizados na literatura como medidas de utilidade. Considere:

- $P(t_k)$ a probabilidade de ocorrência do termo t_k na coleção;
- $P(\neg t_k)$ a probabilidade de não ocorrência do termo t_k na coleção;
- $P(c_i)$ a probabilidade de ocorrência da classe c_i na coleção;
- $P(\neg c_i)$ a probabilidade de não ocorrência da classe c_i na coleção;
- $P(t_k, c_i)$ a probabilidade de ocorrência simultânea do termo t_k e da classe c_i em um documento;
- $P(t_k|c_i)$ a probabilidade de ocorrência do termo t_k dada a ocorrência da classe c_i.
- Frequência dos termos nos documentos

$$\mathrm{F}(t_k, c_i) = P(t_k|c_i) \tag{2.6}$$

- Estatística X^2 (*"chi-square"*)

  A estatística X^2 foi originalmente proposta para testar a independência entre dois eventos A e B. Este critério foi adaptado para o processo de seleção de características considerando-se como evento A a ocorrência do termo t_k e evento B a ocorrência da classe c_i. Esta medida se baseia na diferença da probabilidade ocorrida e da probabilidade esperada.

  Seja $E(k,i) = P(t_k) \times P(c_i)$, a probabilidade da ocorrência simultânea do termo t_k e da classe c_i, $E(\neg k, \neg i)$=$(1 - P(t_k)) \times (1 - P(c_i))$, a probabilidade da não ocorrência simultânea do termo supondo independência entre os eventos e $N(k,i) = P(t_k, c_i)$.

$$X^2(t_k, c_i) = \sum_{a \in \{t_k, \neg t_k\}} \sum_{b \in \{c_i, \neg c_i\}} \frac{(N(a,b) - E(a,b))^2}{E(a,b)} \tag{2.7}$$

- Informação mútua

  A informação mútua mede o quanto a informação da presença ou ausência de um termo t influi na probabilidade de ocorrência de um rótulo l_i correspondendo à classe c_i. Formalmente:

$$MI(t_k, c_i) = \log \frac{P(t_k, c_i)}{P(t_k) \times P(c_i)} \tag{2.8}$$

- Entropia cruzada A entropia cruzada entre duas distribuição de probabilidade mede a quantidade de informação média necessária para identificar um evento dado um conjunto de possibilidades.

$$\mathrm{H}(p, q) = P(t_k) \sum_i P(c_i|t_k) \times \log \frac{P(t_k, c_i)}{P(t_k, c_i)} \tag{2.9}$$

- *Odds Ratio*

$$\mathrm{OR}(t_k, c_i) = \frac{P(t_k|c_i) \times (1 - P(t_k|\neg c_i))}{(1 - P(t_k|c_i)) \times P(t_k|\neg c_i)} \tag{2.10}$$

Uma vez determinado o critério a ser utilizado como medida de utilidade de um certo termo t_k para uma classe c_i, precisamos sumarizar globalmente tais valores para um conjunto de classes C. Isto é, deve-se considerar a utilidade do termo para cada classe c_i e decidir quais os termos a serem utilizados.

Os métodos mais tradicionais são:

- Somatório O critério de sumarização é realizado pelo somatório das medidas das utilidades do termo para cada classe.

$$U(t_k) = \sum_{c_i \in C} U(t_k, c_i) \tag{2.11}$$

- Soma ponderada O critério de sumarização considera a frequência de cada classe ao realizar uma soma ponderada.

$$U(t_k) = \sum_{c_i \in C} U(t_k, c_i) \times P(c_i). \tag{2.12}$$

- Função Máximo O critério de sumarização é realizado pela função máximo, isto é, a utilidade do termo t_k é estabelecida como a maior medida de utilidade para

quaisquer classe c_i.

$$U(t_k) = \max_{c_i \in C} U(t_k, c_i) \tag{2.13}$$

2.6 Métodos de Classificação

Nesta seção apresentaremos os principais métodos de classificação tradicionais existentes. A maioria dos demais métodos de classificação são variações e/ou adaptações destes métodos.

2.6.1 Classificadores probabilísticos

Os modelos probabilísticos baseiam-se em um modelo condicional:

$$p(c_i|t_1, \ldots, t_n)$$

ou seja, estima-se a probabilidade sobre uma variável de classe c_i, condicionais a uma série de características variáveis $t_1, \cdots, t_n$.

O principal representante do grupo de classificadores probabilísticos é o classificador *Naive Bayes*, um dos métodos mais simples de classificação existentes. O teorema de Bayes expressa a probabilidade condicional (probabilidade a posteriori) de uma hipótese $H = p(c_i|t_1, \ldots, t_n)$ em termos da probabilidade apriori de $H = P(c_i)$, a probabilidade a priori de $E = p(t_1, \ldots, t_n)$ e a probabilidade condicional de E dado que H ocorreu $p(t_1, \ldots, t_n|c_i)$.

$$p(c_i|t_1, \ldots, t_n) = \frac{p(c_i)\ p(t_1, \ldots, t_n|c_i)}{p(t_1, \ldots, t_n)} \tag{2.14}$$

Este cálculo baseado em tabelas de probabilidade pode ser intratável. Estas probabilidades são calculadas com base no Teorema de Bayes[8]:

$$\begin{aligned} p(c_i, t_1, \ldots, t_n) &= p(c_i)\ p(t_1, \ldots, t_n|c_i) \\ &= p(c_i)\ p(t_1|c_i)\ p(t_2, \ldots, t_n|c_i, t_1) \\ &= p(c_i)\ p(t_1|c_i)\ p(t_2|c_i, t_1)\ p(t_3, \ldots, t_n|c_i, t_1, t_2) \\ &= p(c_i)\ p(t_1|c_i)\ p(t_2|c_i, t_1)\ p(t_3|c_i, t_1, t_2)\ \ldots p(t_n|c_i, t_1, t_2, t_3, \ldots, t_{n-1}) \end{aligned}$$

[8]O denominador da equação pode ser ignorado pois é um fator que se aplicará a todas as classes.

Este classificador se baseia no teorema de Bayes que assume "fortemente" a independência entre os termos (*"Naive Bayes assumption"*), ou seja, a ocorrência ou ausência de um termo não possui nenhuma relação com a ocorrência ou ausência de outros termos:

$$p(t_i|c_i, t_j) = p(t_i|c_i)$$

Supondo a independência condicional entre os termos ((*"naive bayes assumption"*), podemos simplificar o cálculo das probabilidades:

$$p(c_i, t_1, \ldots, t_n) = p(c_i)\ p(t_1|c_i)\ p(t_2|c_i)\ p(t_3|c_i)\ \cdots = p(c_i) \prod_{k=1}^{n} p(t_k|c_i). \tag{2.15}$$

Tais modelos são facilmente construídos, uma vez que se dependem apenas do cálculo das probabilidade a priori das classes $p(c_i)$ e das distribuições de probabilidade $p(t_k|c_i)$.

Podemos encontrar na literatura dois principais modelos de algoritmos Naive Bayes que se distinguem quanto à definição de evento que tais modelos assumem: modelo de eventos de multivariáveis de Bernoulli e o modelo de eventos multinomial. O modelo de eventos multinomial incorpora a frequência dos termos no documento. Segundo este modelo o documento representa uma sucessão de eventos em que cada evento é a ocorrência de um termo no documento. A premissa **forte** (*"naive"*) de Bayes é aplicada ao assumir-se que a probabilidade de cada ocorrência de um termo em um documento independe do contexto do texto e da posição do termo no documento. Sendo assim, cada documento d_j é representado como uma distribuição multinomial de termos constituído de $|d_j|$ eventos independentes (essa representação é correspondente à representação "bag of words" de documentos).

Apesar de que estes modelos assumem a independência entre os termos o que não é válido na maioria dos problemas de classificação, estes modelos conseguem obter resultados competitivos comparados com os de outros classificadores mais complexos principalmente quando auxiliados por uma boa seleção de características. Existem análises aprofundadas que mostram algumas razões teóricas que justificam a aparente inesperada eficácia dos classificadores *"naive bayes"* [Zhang, 2005]. Outra vantagem dos classificadores *"bayesianos"* é que requerem uma pequena quantidade de documentos de treino para estimar os parâmetros (médias e variâncias das variáveis) necessárias para a classificação, principalmente porque se assume a independência entre as variáveis.

2.6.2 Classificadores baseado em instância

O classificador kNN, um dois mais simples algoritmos de classificação, é baseado em instâncias (*"instance-based"*). O assinalamento dos rótulos aos documentos é decidido com base nas classes às quais pertencem os objetos vizinhos mais próximos na base de treinamento. Este algoritmo encontra os k documentos mais similares e realiza uma votação de acordo com as classes às quais pertencem estes k documentos para a decisão de assinalamento das classes ao documento de teste. Diferentes definições de distância/similaridade entre os objetos podem ser utilizadas. Em geral, a similaridade do cosseno é utilizada na categorização de textos enquanto enquanto em problemas com características numéricas é utilizada a distância Euclidiana, porém pode-se utilizar definições de distância alternativas que tenham um significado consistente com a aplicação específica. A seleção do valor do parâmetro k e da métrica de distância podem ser escolhidas durante a fase de validação.

A votação pode ser realizada por maioria simples ou pela ponderação dos votos de acordo com a similaridade dos documentos, ou seja, os documentos mais similares terão um peso maior durante a votação. Em geral, as técnicas que se baseiam na votação ponderada por similaridade tende a apresentar melhor acurácia. Existe também uma versão probabilística do algoritmo kNN em que as probabilidades de assinalamento a uma classe c_i são estimadas como a proporção dos k vizinhos mais próximos do documento que pertencem à classe c_i. Esta versão probabilística é facilmente adaptada para realizar classificações multi-label [Zhang & Zhou, 2007].

Considere as regiões no espaço em que o conjunto dos k vizinhos mais próximos são os mesmos, estas regiões subdividem o espaço em polígonos convexos em que a classificação é a mesma.

2.6.3 Classificadores Lineares

Os classificadores lineares decidem o assinalamento de rótulos com base em uma combinação linear das características $\vec{f}$ de um documento d_j. Em geral, estas características são utilizadas para estimar uma medida de probabilidade do documento pertencer à certa classe c_i e o documento é assinalado caso esta *probabilidade* exceda um certo limiar de assinalamento.

- Rocchio

  O classificador Rocchio se baseia na definição de *centróides*, pontos representantes das classes, que são definidos como o centro de massa ou a médias dos valores dos documentos que pertencem à classe. O assinalamento do documento é decidido

avaliando-se a distância/similaridade das características $\vec{f}$ do documento d_j com o centróides da classe c_i. Seja $\mathcal{D}_\rangle$ o conjunto dos documentos da base de treinamento que pertencem à classe c_i, o centróide da classe c_i pode ser então calculado por:

$$\vec{\mu}(c_i) = \frac{1}{|\mathcal{D}_i|} \sum_{d_j \in \mathcal{D}_i} \vec{v}(d_j) \tag{2.16}$$

Note que com essa abordagem são definidas fronteiras, hiperplanos, em que a distância a dois centróides é a mesma. Assim como o classificador kNN, o cálculo das distâncias/similaridades entre um documento e os centróides das classes podem se basear em diferentes métricas de distância e em bases de documentos textuais geralmente baseiam-se na similaridade do cosseno.

- Support Vector Machines

  Vários classificadores lineares buscam encontrar um hiperplano que separam os objetos de diferentes classes. Alguns algoritmos procuram por qualquer hiperplano que satisfaça à condição de divisão, outros buscam definir o melhor hiperplano segundo algum critério. A teoria sobre SVM foi primeiramente introduzida Os classificadores SVM, introduzido por Cortes & Vapnik [1995], baseiam-se no critério de que o melhor hiperplano a ser encontrado deve manter a maior distância possível entre quaisquer objetos de cada uma das classes. A intuição destes algoritmos é de que a distância entre o hiperplano e o objeto mais próximo deste hiperplano, chamada de *margem*, deve ser a maior possível de forma a maximizar a *generalização* do modelo e minimizar o superajuste (seção 2.7.1). Para calcular o hiperplano, o algoritmo encontra os objetos que estão mais próximos à fronteira e suas coordenadas são utilizadas para a definição do hiperplano. Tais objetos são chamados de *vetores de suporte* (*"support vectors"*) pois apenas as suas posições importam para a determinação do hiperplano a ser traçado. Os demais objetos, que não estão próximos da fronteira, não influem durante a fase de aprendizagem. Na figura 2.3 exemplificamos uma base de treinamento, possíveis hiperplanos que separam as classes e o hiperplano que maximiza a margem de separação. Também foram destacados os *vetores de suporte* de cada classe, assim como a margem entre os vetores enaltecendo que o hiperplano definido é aquele em que a margem é a maior possível.

  Seja um conjunto de objetos da base de treinamento dado por:

$$\mathcal{D} = \{(\mathbf{x}_i, y_i) | \mathbf{x}_i \in \mathbb{R}^d, y_i \in \{-1, 1\}\}_{i=1}^n \tag{2.17}$$

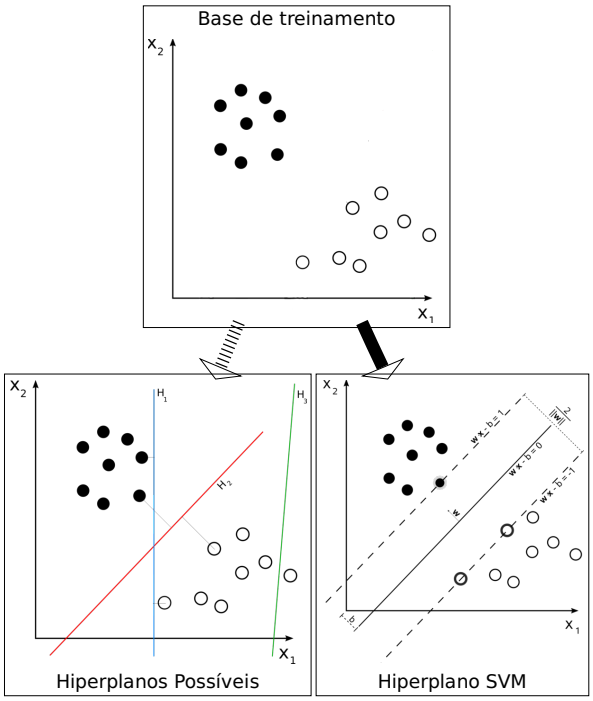


Figura 2.3: O Classificador SVM

onde $y_i = \{-1, 1\}$ em que $y_i = 1$ indica que o objeto $\mathbf{x}_i$ pertence à classe e $y_i = -1$ indica a ausência do rótulo da classe [9]. Chamamos os documentos de treino em que $y_i = 1$ de positivos e $y_i = -1$ de negativos. Precisa-se encontrar o hiperplano que separa os documentos positivos e negativos da base de treinamento. Para a definição da *função hipótese* γ precisamos definir um vetor normal $\vec{w}$ ao hiperplano, o vetor de pesos, e um parâmetro de interseção b que determina o posicionamento do hiperplano em relação ao ponto origem:

$$f(\vec{x}) = sign(\vec{w}^T \vec{x} + b), \tag{2.18}$$

Precisamos então encontrar os valores $\vec{w}$ e b que maximizam a margem. Ou seja, maximizar a distância entre dois hiperplanos paralelos que separam os documentos de diferentes classes da base de treinamento:

$$\vec{w}^T \vec{x} - b = +1$$

e

$$\vec{w}^T \vec{x} - b = -1.$$

[9] A formulação do problema tradicional é de uma classificação binária.

A distância entre os dois hiperplanos é $\frac{2}{\|\vec{w}\|}$. Precisa-se minimizar $\|\vec{w}\|$ respeitando as restrições de que não deve haver documentos encontrados entre os dois hiperplanos nem documentos mal classificados. Se a base de treinamento é linearmente separável, os hiperplanos respeitarão estritamente a condição de separação dos documentos de diferentes classes. Para respeitar as restrições adicionamos as seguintes condições ao problema:

$$y_i(\vec{w}^T\vec{x_i} - b) \geq 1, \quad \forall i| \quad 1 \leq i \leq n. \qquad (1) \qquad (2.19)$$

Integrando a função objetivo e as restrições em uma só formulação, formulamos o seguinte problema de otimização:

Minimize (baseado nas variáveis $\vec{w}, b$):

$$\|\vec{w}\|$$

sujeito a $(\forall i = 1, \ldots, n)\ y_i(\vec{w}^T\vec{x_i} - b) \geq 1$.

Este formulação é de um problema de otimização quadrática para o qual existem vários algoritmos para a sua resolução. Existem também vários algoritmos mais eficientes para resolver problemas específicos em que se existe a mesma estrutura deste problema de otimização quadrática.

As diversas extensões do algoritmo SVM não serão tratadas nesta dissertação, como o SVM multi-classe, SVM não-linear, SVM com margem flexível (*soft-margin classifier*).

2.6.4 Classificadores baseado em árvores de decisão

Os algoritmos baseados em regras de classificação buscam encontrar uma combinação de características que induz à decisão de assinalamento de um rótulo. Estes algoritmos são populares pois a decisão sobre o assinalamento é decidido baseado em regras de classificação que são facilmente compreensíveis e gerenciadas pelo usuário. De fato um algoritmo baseado em regras de classificação pode ser desenvolvido de forma que a fase de aprendizagem de máquina é suprimida, o especialista então com conhecimento sobre o domínio de aplicação constrói uma série de regras que ao serem aplicadas determinam as regras de classificação. Tais modelos explicativos facilitam a tarefa de classificação já que permite a apresentação ao usuário das características do modelo que motivaram a classificação e muitas vezes permitem ao usuário agir ativamente na construção ou restrição de regras para adaptar a particularidades do problema.

Os primeiros algoritmos propostos baseados em regras de classificação foram aqueles que constroem uma árvore de decisão durante a fase de aprendizagem. Uma árvore de decisão é uma árvore binária $\mathcal{A}$ em que cada nodo interno é rotulado por uma condição de decisão a ser tomada baseada em características dos documentos. O processo de decisão acontece avaliando-se as características $\vec{f}$ de um documento de teste percorrendo-se uma árvore de decisão em que as características do documento são utilizadas para decidir como se dará a navegação na árvore (i.e. qual o ramo da árvore a ser seguido) até que se atinja uma folha da árvore. As folhas desta árvore representam o assinalamento de rótulos referente às classes $c_i \in C$.

Os algoritmos baseados em árvores de decisão constroem a árvore de decisão durante o treinamento de acordo com diferentes critérios para separação dos dados em cada nodo. Estes algoritmos geralmente constroem a árvore segundo uma estratégia recursiva de divisão-e-conquista em que a árvore é construída particionando os nodos em seus filhos baseado em um teste e este processo de divisão-e-conquista é repetido para cada nodo-filho. O processo de particionamento hierárquico para caso ou não seja viável particionar um nodo ou uma discriminação perfeita seja atingida (i.e., todos os documentos pertencem à mesma classe). O cerne destes algoritmos reside na seleção das características (i.e., quais características devem ser incluídas na decisão de um certo nodo). Existem diversos critérios de seleção de características como os baseados em entropia Veloso [2007].

2.6.5 Algoritmos de classificação associativa

Os algoritmos de classificação associativa foram propostos como evolução dos algoritmos baseados em árvores de decisão. Enquanto as árvores de decisão fazem uma busca heurística local gulosa pelas regras (decisões) analisando as características mais promissoras, os algoritmos de classificação associativa realizam uma busca global pelas regras que satisfazem algumas restrições de qualidade. Tais algoritmos se diferenciam pelas restrições de qualidade impostas durante a seleção das regras.

Uma regra de classificação, também chamada de regra de associação de classe, é uma regra da forma:

$$r = E \implies l_i \tag{2.20}$$

tal que E é o antecedente da regra e representa um subconjunto de características, $E \in F$, e l_i é o consequente da regra r, $l_i \in C$, i.e. o rótulo l_i corresponde ao assinalamento a uma das classes do conjunto de classes C. Uma regra é válida em um documento de treino d_j se $E \in \vec{f}_i$, i. e. d_j possui todas as características de E e $l_i \in c_i$, i. e. o documento d_j foi assinalado o rótulo l_i. Em geral, uma regra de classificação

pode ser avaliada segundo diferentes critérios de relevância. Os principais critérios utilizados são confiança e suporte. Seja $P(E)$ a probabilidade de um documento d_j possuir as características de E. $P(l_i)$ a probabilidade de um documento ser assinalado à classe l_i. Podemos definir o suporte σ de uma regra como:

Definição 4

$$\sigma(E \implies l_i) = P(E \cap l_i) \tag{2.21}$$

O suporte de uma regra $E \implies l_i$ é $\sigma(E \implies l_i)$, ou seja, a proporção dos documentos de treino em que tanto E quanto l_i ocorre. A confiança de uma regra $E \implies l_i$ é definida por:

Definição 5

$$\theta(E \implies l_i) = \frac{P(E \implies l_i)}{P(E)} \tag{2.22}$$

A confiança reflete a certeza que temos de que dada a ocorrência do antecedente, a probabilidade de que o consequente da regra deverá ocorrer.

Enquanto o suporte σ garante que uma regra frequentemente se aplica aos documentos da base de dados, a confiança define a força da regra. O número de regras pode ser extenso e cresce exponencialmente em relação ao número de características $|\vec{f}|$. Por esta razão, estes algoritmos focam apenas nas regras mais relevantes, i. e. regras fortes e frequentes. Assim, tais algoritmos ignoram as regras cujo σ_{min} e θ_{min} não sejam superados Veloso [2007].

Os algoritmos que se baseiam apenas na melhor regra encontrada podem não realizar a melhor decisão. Isso acontece porque a generalização fica comprometida levando ao superajuste (conceitos a serem discutidos nas seção 2.7.1). Algoritmos foram propostos para realizar a predição baseado em todas as regras geradas coletivamente. Para isso, é necessário um método de votação. O método de votação considera cada regra CAR cujo consequente é c_i como um voto ponderado com base na qualidade da regra (segundo um critério escolhido, e.g. confiança da regra) ao assinalamento à classe c_i. Os votos são então ponderados para cada rótulo l_i de acordo com a definição 6.

Definição 6 *Seja R_i o conjunto de regras válidas que superam os os limiares σ_{min} e θ_{min} e cujo consequente é l_i, a função real que define a ponderação das regras é:*

$$S(C_i) = \sum_{r \in R_i} \sigma(r) \times \theta(r) \tag{2.23}$$

Apesar de termos apresentado a ponderação de votos com a soma das regras, diferentes autores sugerem diferentes formas de ponderação das regras tais como utilizar a média

ao invés da soma, utilizar outros critérios de regra ao invés da confiança, considerar o suporte $\sigma(l_i)$ dentre outras diversas alternativas. Note que o classificador bayesiano Bernoulli pode ser considerado um classificador que gera apenas regras de tamanho 2 ("naive bayes assumption") e que possui um critério de votação específico.

2.6.6 Algoritmos Postergados

Os algoritmos de classificação postergados são aqueles em que a fase de treino é suprimida. A ideia destes algoritmos é postergar a construção do conceito até que um documento de teste seja conhecido. Com base nas características deste documento de teste. os algoritmos então constroem conceitos especializados para a instância de teste em questão. Os mais conhecidos algoritmos de aprendizagem postergada são os algoritmos baseados em instância (seção 2.6.2) já que o conceito é induzido a partir de evidências dos documentos considerados os vizinhos mais próximos do documento de teste.

Um outro grupo de algoritmos postergados que vem ganhando popularidade recentemente são os algoritmos de classificação associativa postergada. Este grupo de técnicas foi utilizada com sucesso em diferentes cenários de aplicação como classificação multi-evidência [Veloso et al., 2006], classificação multi-rótulo [Veloso et al., 2007] e classificação calibrada [Veloso et al., 2008].

Os algoritmos de classificação associativa tradicionais realizam uma busca global pelas regras diferentemente da busca gulosa dos algoritmos de árvores de decisão. Um novo conceito introduzido pelos algoritmos de classificação associativa postergada é que a estratégia de indução do conceito é realizada em um contexto sob demanda. Desta forma, a criação de regras inúteis para a classificação é evitada e consegue-se definir parâmetros mais flexíveis de acordo com o documento de teste a ser classificado.

Tais técnicas primeiramente realizam uma projeção da base de treinamento de acordo com as características do documento de teste. O treinamento, i.e., a geração das regras de classificação, é realizado sobre a base de dados resultante desta projeção. Com isso, constrói-se um conceito especializado para cada documento de teste em que são consideradas suas evidências.

2.6.7 Comitê de classificadores

Existe uma série de algoritmos que tendem a melhorar o desempenho dos sistemas através da utilização de um comitê constituído de mais de um classificador (esta área é conhecida por diversos nomes como *"committee classifiers"*, *"ensemble classifiers"*,

multiple classifiers systems" ou *"mixture of experts"*) para resultar em um sistema de classificação integrado mais correto, preciso e acurado. Intuitivamente a ideia é que um comitê de classificadores pode apresentar acurácia superior a todos os demais classificadores utilizados individualmente.

Um comitê é composto por diferentes sistemas de classificação:

1. Comitês de reamostragem

   Os comitês de reamostragem são formados por diferentes hipóteses produzidas por meio da aprendizagem com base em diferentes conjuntos da base de treinamento. A aprendizagem, em geral, é realizada com uma única técnica de classificação base, i.e. um único algoritmo de classificação.

2. Comitês de Múltiplos parâmetros

   Os algoritmos de aprendizagem em geral necessitam da execução de uma fase de validação para a escolha dos melhores parâmetros a serem utilizados durante a classificação. Uma alternativa à seleção dos melhores parâmetros é a construção de um comitê formado pelo mesmo sistema de classificação base executando-se com diferentes parâmetros.

3. Comitês de múltiplos sistemas de classificação base

   Uma outra alternativa mais complexa é a constituição de um comitê utilizando diferentes sistemas de classificação com características diferentes. Uma vantagem deste tipo de comitês é que pode-se criar o comitê com classificadores de características muito diferentes tornando o sistema mais robusto. Por exemplo, o comitê pode ser escolhido selecionando-se diferentes algoritmos com *"compromissos viés-variância"* variados (ver sobre na seção 2.7.2).

2.6.7.1 Comitês de classificadores baseado em reamostragem

Os principais trabalhos de comitês de classificadores encontrados são comitês de reamostragem [Kotsiantis & Pintelas, 2004; Dietterich, 2000]. Os comitês de reamostragem trabalham particularmente muito bem com algoritmos de aprendizagem instáveis, i.e. algoritmos em que pequenas mudanças na base de treinamento resultam em sensíveis mudanças nas predições realizadas. Isto acontece porque, se as predições dos algoritmos não divergem muito, a agregação das várias decisões será muito similar às decisões individuais resultando um ganho inexpressivo. Por esta razão, algoritmos considerados instáveis, tais como os baseado em árvores de decisão, regras de classificação ou redes neuronais são mais utilizados em comitês enquanto algoritmos baseados em regressão

linear e vizinhos mais próximos, mais estáveis, não são tão utilizados [Kotsiantis & Pintelas, 2004].

Dentre as técnicas de comitês baseado em reamostragem se destacam as técnicas de *"bagging"* ou *"boosting"* que se distinguem pelo método de amostragem da base de treinamento de cada classificador. O conceito *"bagging"* é originado da expressão *"bootstrap aggregating"* que representa o método pela *agregação* de modelos gerados de amostras diferentes. *"Bagging"* consiste, portanto, na criação de um comitê baseado em várias hipóteses induzidas de um subconjunto da base de treinamento. Estes subconjuntos são gerados por uma amostragem aleatória com substituição de $|D|$ documentos da base de dados. Cada amostra, *"bootstrap"*, pode conter duplicatas assim como não conter todos os documentos da base de dados. Estatisticamente, em média, cada amostra terá 63,2% dos documentos representados e conterá muitos documentos múltiplas vezes repetidos [Kotsiantis & Pintelas, 2004]. Após a indução das várias hipóteses um método de votação é utilizado para agregar as diferentes predições em uma predição única predição final do comitê. Note que se a base de treinamento é pequena, os ganhos do comitê não conseguirão compensar a queda de acurácia dos modelos individuais, pois cada modelo será construído a partir de uma base de treinamento ainda menor [Kotsiantis & Pintelas, 2004].

"Boosting" é uma técnica de reamostragem iterativa sequencial que atribui diferentes pesos aos documentos da base de treino. Os pesos são atribuídos de acordo com a acurácia do classificador para cada documento em específico na iteração anterior, aumentando-se os pesos para os documentos mal classificados. Assim, esta técnica constrói problemas de aprendizagem cada vez mais difíceis, concentrando os esforços dos classificadores consecutivos nos documentos que ainda não foram corretamente classificados. A fase de agregação dos comitês é realizada por meio de uma votação ponderada em que os pesos são definidos de acordo com a acurácia de cada modelo apurada na base de treino.

Os algoritmos de *"boosting"* são considerados superiores aos de *"bagging"* em bases de dados sem ruídos. Entretanto, resultados empíricos indicam que as técnicas de *"bagging"* são menos sensíveis a ruído e superam as técnicas de *"boosting"* em bases de dados com ruído [Kotsiantis & Pintelas, 2004]. Uma outra vantagem das técnicas de *"boosting"*, e em particular do Adaboost, é que o modelo não tende ao superajuste rapidamente [Dietterich, 2000][10].

As técnicas baseadas em comitês exigem a construção de vários modelos para posterior votação ou integração e algumas técnicas de classificação base são proibitivas

[10] O super ajuste é explicado na seção 2.7.1

pelo alto custo que se demanda. Por este motivo, a construção dos comitês utilizam sistemas de classificação-base mais simples, como o algoritmo C4.5, um algoritmo eficiente, considerado instável, de árvores de decisão.

2.7 Análise de algoritmos de Aprendizagem

Selecionar um algoritmo de classificação é uma parte vital de um sistema de classificação destinado a resolver um problema de categorização de textos específico. Para resultar em um bom sistema de classificação, é necessário que o analista conheça algumas das características da base de dados e como cada algoritmo de classificação se comporta diante destas características. Por esta razão é importante entender os diferentes fenômenos existentes em um sistema de classificação.

Nesta seção introduziremos alguns fundamentos importantes para a análise e comparação dos sistemas de classificação.

2.7.1 Superajuste (*"Overfitting"*)

A aprendizagem supervisionada prediz uma função de assinalamento com base em um conjunto de treino e assumem que a mesma distribuição encontrada na base de treinamento será encontrada na base de teste. Os algoritmos de aprendizagem procuram uma aproximação da função de assinalamento real que precisa ser *"generalizada"* para ser aplicável a documentos/situações inéditas. Caso se aprenda um modelo extremamente complexo, este modelo pode não ser *"generalizável"*, pois o modelo pode ter sido extremamente **ajustado** para o conjunto de treino, chamamos este fenômeno de superajuste (*"overfitting"*). Um algoritmo de aprendizado pode se ajustar muito bem a um conjunto de treino e não incorrer em erros ao se classificar tal conjunto, mas não necessariamente este modelo será efetivo ao se classificar o conjunto de teste onde documentos inéditos devem ser classificados.

O superajuste acontece quando a generalização ou enviezamento indutivo é *"contaminado"* por uma propriedade acidental do conjunto de treino, como um erro, variância natural do modelo ou ruídos inseridos neste conjunto de documentos.

2.7.2 O compromisso viés-variância

O compromisso viés-variância (*"bias-variance trade-off"* ou *"bias-variance dillema"*) é um dos fatores mais importantes para a escolha do algoritmo apropriado para um problema de classificação específico. O compromisso viés-variância foi introduzido ini-

cialmente no domínio regressão estatística, baseando-se na função de erro quadrático como a função de erro (*"loss function"*) [Geman et al. [1992] apud. Valentini & Dietterich [2004]]. O objetivo do estudo deste dilema é entender quais características de um método induz uma classificação incorreta. Existem outras funções de erro possíveis e diferentes decomposições destes erros, como *"0/1-loss"*, dentre outras.

Nestas decomposições, busca-se evidenciar duas componentes que constituem o erro [11]. O viés (*"bias"*) representa o erro sistemático ou a tendenciosidade do modelo, enquanto a variância representa a variabilidade de um estimador. Apesar de existirem outros componentes em outros modelos de decomposição, apresentaremos o modelo de erro quadrático que já nos permite entender estas duas componentes principais, viés e variância.

Nestes modelos assume-se que existe uma incerteza inerente ao assinalamento e avalia-se o quão bem o classificador estima a probabilidade condicional $P(c_i|d)$ de um documento d pertencer a uma classe c_i, dado que a base de treinamento e de testes pertencem à distribuição $P(\langle d, c_i\rangle)$. O objetivo desta classificação é encontrar uma função hipótese γ tal que $\gamma(d)$ esteja o mais próximo possível da probabilidade real $P(c|d)$ avaliada sobre todo o conjunto de dados $\mathcal{D}$. Para sumarizar esta medida utilizamos o erro quadrático médio (*"mean squared error"*):

$$\text{MSE}(\gamma) = E_d[\gamma(d) - P(c|d)]^2 \tag{2.24}$$

onde E_d é a média de $P(d)$.

Um classificador γ deve portanto minimizar $\text{MSE}(\gamma)$ para uma distribuição $P(\langle d, c_i\rangle)$.

$$\begin{aligned}
\text{erro-de-aprendizagem}(\Gamma) &= E_{\mathcal{D}}[\text{MSE}(\Gamma_{\mathcal{D}})] && (2.25)\\
&= E_{\mathcal{D}}E_d[\Gamma_{\mathcal{D}}(d) - P(c_i|d)]^2 && (2.26)\\
&= E_d[\text{viés}(\Gamma, d) + \text{variância}(\Gamma, d)] && (2.27)\\
\text{viés}(\Gamma, d) &= [P(c_i|d) - E_{\mathcal{D}}\Gamma_{\mathcal{D}}(d)]^2 && (2.28)\\
\text{variância}(\Gamma, d) &= E_{\mathcal{D}}[\Gamma_{\mathcal{D}}(d) - E_{\mathcal{D}}\Gamma_{\mathcal{D}}(d)]^2 && (2.29)
\end{aligned}$$

Segundo este modelo, o viés (*"bias"*) é portanto a diferença quadrática entre $P(c_i|d)$, a real probabilidade condicional de d pertencer à classe c_i, e $\Gamma_{\mathcal{D}}(d)$, a média das predições realizadas pelo classificador sobre a base de dados de treinamento. O viés

[11] Pode haver outras componentes dependendo do modelo de decomposição.

é alto se o método de aprendizagem realiza classificações consistentemente erradas. O viés é considerado baixo se:

1. as classificações estão consistentemente corretas;

2. modelos gerados a partir de variadas bases de treinamento gera classificações erradas em diferentes documentos;

3. modelos gerados a partir de variadas bases de treinamento geram classificações corretas ou incorretas nos mesmos documentos, mas na média, próximas de 0.

Se qualquer uma destas três hipóteses é válida, então $E_{\mathcal{D}}\Gamma_{\mathcal{D}}(d)$, a média sobre todos os documentos da base de treinamento, está próxima de $P(c|d)$.

Métodos lineares como *Rocchio* e *Naive Bayes* apresentam um alto viés para problemas não-lineares porque só podem modelar um único tipo de fronteiras de classe, um hiperplano linear. Assim, Se um modelo gerativo $P(d)$ apresenta fronteiras de classe com complexidade não-linear, o viés será alto porque um grande número de objetos será consistentemente classificado incorretamente. Um especialista que defina que utilizará um algoritmo linear para predizer a classificação em uma base de dados reconhecidamente não-linear **envieza** o algoritmo a tratar o problema como se fosse linearmente tratável e o induz a cometer erros sistemáticos.

Os métodos não-lineares, como por exemplo o *kNN*, em geral, apresentam baixo viés. No algoritmo *kNN*, as fronteiras de decisão podem variar sensivelmente com uma pequena alteração da distribuição dos documentos na base de treinamento. Este fato resulta em que cada documento possui uma chance de ser classificado corretamente para algumas bases de dados e incorretamente para uma base de dados apenas um pouco diferente. A predição média $E_{\mathcal{D}}\Gamma_{\mathcal{D}}(d)$ fica portanto mais próximo de $P(c|d)$ e o viés é menor que nos métodos lineares.

A variância é a variação na previsão dos classificadores: a média da diferença quadrática entre $\Gamma_{\mathcal{D}}(d)$ e a sua média $E_{\mathcal{D}}\Gamma_{\mathcal{D}}(d)$. A variância é grande se diferentes bases de treino $\mathcal{D}$ induzem classificadores $\Gamma_{\mathcal{D}}$ extremamente diferentes e pequena se a base de treinamento participa fracamente nas decisões de assinalamento que $\Gamma_{\mathcal{D}}$ realiza, sejam corretas ou incorretas. Variância mensura quão inconsistentes as decisões são, não importando se as mesmas são corretas ou incorretas.

Os métodos lineares apresentam pequena variância porque bases de treinamento pouco divergentes produzem decisões baseadas em hiperplanos muito similares. Os métodos não-lineares apresentam alta variância, assim tais algoritmos são mais sensíveis a ruído. Isto ocorre porque erros de classificação em documentos de teste similares a

um documento com ruído na base de treinamento serão cometidos. Assim, diferentes amostragens da base de treinamento podem resultar em predições muito diferentes [12].

Os métodos de aprendizagem que apresentam alta variância são mais suscetíveis ao superajuste. Isto acontece pois ao capturar as características reais da base de treinamento, tais métodos são suscetíveis a "aprender" também o ruído. Também pode-se pensar na variância como a complexidade do modelo, ou seja, o nível de detalhamento na caracterização da base de treinamento. Assim, caso exista maior detalhamento, é possível modelar-se mais detalhes de grão-fino e consequentemente a variância será alta.

Nosso objetivo é selecionar um método de aprendizagem que minimize o erro que é constituído de duas componentes que não podem ser simultaneamente minimizados. Ao compararmos dois métodos Γ_1 e Γ_2, na maioria dos casos, estaremos analisando o dilema de que um método apresenta maior viés e menor variância enquanto o outro apresenta menor viés e maior variância. A escolha por um método de aprendizagem em detrimento de outro não se reduz simplesmente à escolha de qual método produz classificadores mais confiáveis nas bases de treinamento (baixa variância) ou os métodos que podem aprender decisões de fronteiras mais difíceis (baixo viés), a escolha depende que analisemos os méritos tanto do viés quanto da variância no domínio de aplicação. A análise aprofundada do compromisso viés-variância é portanto essencial na seleção dos classificadores a utilizar em um problema específico.

Sob o ponto de vista do compromisso viés-variância, pode-se acreditar que as classes nos problemas de classificação de textos em geral são complexas e aparentemente não podem ser modeladas linearmente, assim, parece surpreendente que muitos dos algoritmos mais conhecidos de classificação de textos são lineares, sendo muitos deles são os mais efetivos em muitas aplicações, como os SVMs lineares e/ou *"regularized linear regression"*. Acontece que em espaços com alta dimensionalidade como os encontrados na categorização de textos, o potencial de separação das fronteiras por uma função linear aumenta acentuadamente, o que torna os modelos lineares extremamente poderosos. Ainda assim, métodos de aprendizagem não-lineares podem modelar fronteiras de decisão mais complexas que um hiperplano, mas isto também os tornam mais sensíveis a ruído. Métodos não-lineares algumas vezes são superiores se existe considerável base de treinamento, mas essa superioridade não se aplica a todos os casos.

[12] Na seção 2.6.7 chamamos essa característica de instabilidade.

Capítulo 3

Trabalhos Relacionados

Neste capítulo, apresentamos vários trabalhos existentes na área de classificação de documentos e mais especificamente na categorização de textos que resolvem o problema de **HMC**. Primeiramente, na seção 3.1 apresentamos várias técnicas aplicáveis à classificação de documentos multi-rótulo plana e na seção 3.2 apresentamos as técnicas para o problema de **HTC** Posteriormente, apresentamos alguns desafios e problemas que surgem ao tratar o problema mais amplo da classificação multi-rótulo hierárquica. Apresentaremos também as técnicas existentes para a solução destes novos problemas.

3.1 Classificação Multi-Rótulo

Nesta seção apresentamos os problemas de classificação plana (não-hierárquica) multi-rótulo. Apresentamos os principais trabalhos algoritmos sobre classificação multi-rótulo na subseção 3.1.1 e em seguida, na subseção 3.1.2, apresentamos as medidas de classificação multi-rótulo utilizadas na avaliação de sistemas de classificação multi-rótulo.

3.1.1 Aprendizagem multi-rótulo

Nesta subseção apresentamos os principais trabalhos de técnicas de classificação multi-rótulo.

A maioria dos problemas multi-rótulo assumem a presença de poucas classes de treino que não demandam uma organização em forma de ontologias ou hierarquias.

A generalidade do problema de classificação multi-rótulo deixa claro que este problema é muito mais difícil do que o problema da classificação tradicional (único-rótulo). Os primeiros trabalhos nesta área foram propostas para tratar do problema de ambiguidade durante a categorização de textos. A literatura ainda considera esta

uma área nova [Zhang & Zhou, 2007]. Interessante ressaltar que alguns trabalhos de classificação multi-rótulo foram executados em uma base de dados hierárquica multi-rótulo porém classificando-se apenas no primeiro nível da hierarquia, como por exemplo em Zhang & Zhou [2007], cujos experimentos foram realizados no primeiro nível da hierarquia da base Yeast (esta base de dados possui quatro níveis de profundidade) [1] e em diferentes ramos do diretório de páginas Yahoo.

Grande parte dos trabalhos de classificação multi-rótulo, principalmente os primeiros se concentraram em apenas realizar a transformação de uma base de dados multi-rótulo em uma base de dados único-rótulo. Tsoumakas & Katakis [2007] apresentam um interessante resumo das diferentes técnicas existentes sobre os algoritmos de classificação multi-rótulo. Para isso ele divide tais algoritmos justamente entre dois diferentes grupos principais:

1. Algoritmos baseados em transformação do problema: Métodos que transformam a classificação multi-rótulo em um ou mais problemas de classificação único rótulo ou de regressão.

2. Algoritmos baseados em adaptação algorítmica: Algoritmos que estendem algoritmos de aprendizagem de máquina para suportarem as bases de dados multi-rótulo diretamente.

3.1.1.1 Transformações do problema de classificação Multi-rótulo

Tsoumakas & Katakis [2007] apresenta as diferentes transformações do problema multi-rótulo para um ou mais problemas único-rótulo. Tais transformações são discutidas para facilitar a análise dos diferentes algoritmos posteriormente.

Doc.	Indústria	Agricultura	Comércio	Serviços
1	X			X
2			X	X
3	X			
4		X	X	

Tabela 3.1: Exemplo de assinalamento multi-rótulo

Os primeiros trabalhos de classificação em bases de dados multi-rótulo ainda consideravam que um documento pertencer a mais de uma classe era de certa forma considerada um ruído. Tais abordagens propunham então ignorar este ruído e realizar a

[1] http://mips.gsf.de/proj/yeast/catalogues/funcat

classificação único rótulo. Seguindo esta linha de raciocínio, a primeira transformação, chamada por Tsoumakas & Katakis [2007] de PT1, constitui-se de arbitrariamente ou aleatoriamente selecionar apenas um dos rótulos dos documentos multi-rótulo e sumariamente descartar os demais rótulos do documento. Uma segunda transformação, PT2, considera que por serem ruídos, tais documentos dever ser retirados da base de dados e não devem ser considerados em nenhuma fase do sistema de classificação. Estas abordagens descartam uma quantidade considerável de informação além de não considerarem a multi-rotularidade como um fenômeno natural. Suponha que exista uma base de dados cujos documentos pertencem às classes como descritas na tabela 3.1. Apresentamos as bases de dados resultante da transformação PT1 (tabela 3.2(a)) e PT2 (tabela 3.2(b)).

Doc.	Indústria	Agricultura	Comércio	Serviços
1				X
2			X	
3	X			
4		X		

(a) PT1

Doc.	Indústria	Agricultura	Comércio	Serviços
3	X			

(b) PT2

Tabela 3.2: Transformações do problema multi-rótulo que consideram multi-rotularidade como ruído

Doc.	Comércio	Ind. & Serv.	Com. & Serv.	Agr. & Com.
1		X		
2			X	
3	X			
4				X

Tabela 3.3: Transformação em conjunto potência

A transformação em conjunto potência (TCP) é uma das mais populares transformações e consiste em considerar cada possível subconjunto de rótulos do problema multi-rótulo original como um único rótulo no problema transformado. Então, realiza-se a classificação único rótulo no problema transformado $H : X \to P(L)$, onde $P(L)$ é o conjunto potência do conjunto de rótulos L original. Apesar de conseguir tratar de alguma forma a correlação entre rótulos, essa transformação, em geral, gera um número

muito grande de classes no problema transformado e provavelmente um número muito pequeno de exemplos por classe. Esta esparsidade em geral degrada a eficiência do algoritmo principalmente se o número de classes originalmente já for expressivo. Por exemplo, em um problema originalmente com 100 classes, o número de rótulos possíveis ao realizar essa transformação seria de $2^{100} \sim 10^{30}$. Esta alternativa já foi utilizada em vários trabalhos como por exemplo Boutell et al. [2004] dentre outros. O resultado da transformação do problema original neste problema é apresentado na tabela 3.3.

Doc.	Indústria	¬Indústria
1	X	
2		X
3	X	
4		X

(a) Subproblema Indústria

Doc.	Agricultura	¬Agricultura
1		X
2		X
3		X
4	X	

(b) Subproblema Agricultura

Doc.	Comércio	¬Comércio
1		X
2	X	
3		X
4	X	

(c) Subproblema Comércio

Doc.	Serviços	¬Serviços
1	X	
2	X	
3		X
4		X

(d) Subproblema Serviços

Tabela 3.4: Transformações do problema multi-rótulo em vários problemas de classificação binária

Uma-Classe-Contra-Resto (TUCCR) é a transformação mais comum encontrada na literatura, porque é semelhante à adaptação utilizada para a classificação único-rótulo multi-classe. Como grande parte dos algoritmos de aprendizagem foram tradicionalmente desenvolvidos para tratar o problema da classificação binária, pode-se realizar a aprendizagem de $|C|$ classificadores binários $H_c : X \to \{c_i, \neg c_i\}$, um para cada rótulo $c_i \in C$. Para cada classificação binária, um documento é *positivo* (rotulado *verdadeiro*) se o documento pertence à classe c ou negativo ($falso$) caso contrário. Por fim, a decisão de assinalamento é o conjunto união de todos os rótulos positivos:

$$\mathcal{H}(x) = \cup_{c_i \in C} c_i : H_{ci}(x) = c_i \tag{3.1}$$

Um dos problemas da transformação TUCCR é que, possivelmente, o número de documentos que não possuem os rótulos será muito maior que o número de documentos positivos, levando a um desbalanceamento. O resultado desta transformação é apresentado na tabela 3.4.

Doc.	Classe
1	Indústria
1	Serviços
2	Comércio
2	Serviços
3	Agricultura
4	Agricultura
4	Comércio

Tabela 3.5: Transformações do problema multi-rótulo por replicação de atributos

Uma alternativa diferente é a transformação por replicação de atributos (TRA) que consiste na transformação de cada documento multi-rótulo em vários documentos único-rótulo, um para cada rótulo c_i a que o documento foi assinalado. Posteriormente, aplica-se um algoritmo de classificação que produz um grau de certeza e assinala-se os rótulos cujo grau de certeza seja maior que um certo limiar. O resultado da transformação TRA é apresentado na tabela 3.5.

3.1.1.2 Adaptação algorítmica do problema de classificação multi-rótulo

Nesta seção apresentamos os principais algoritmos especificamente desenvolvidos para os problemas de classificação multi-rótulo. Estas técnicas em geral foram propostas para lidar fortemente com a ambiguidade e dependência entre as classes de forma não estruturada, ou seja, não é conhecida nem informada, a priori, nenhuma relação estruturada de dependência entre as classes. Geralmente tais técnicas assumem que existe bastante informação para todas as classes e portanto os seus modelos devem também aprender a relação entre estas classes.

Uma das primeiras e principais propostas de classificação multi-rótulo é o ML-kNN [Zhang & Zhou, 2007] que propõe uma abordagem baseada em vizinhos mais próximos para o problema da classificação multi-rótulo. Baseado no conjunto de vizinhos mais próximos e no ganho de informação dos rótulos destes vizinhos, ou seja, no número de vizinhos pertencentes a cada possível classe, os rótulos são determinados com base no princípio MAP (*"maximum a posteriori principle"*).

McCallum [1999] propôs um algoritmo bayesiano para classificação multi-rótulo. Este algoritmo define um modelo gerativo probabilístico que assume que cada rótulo gera uma certa distribuição de diferentes palavras. Nesta intuição, os algoritmos multi-rótulos são portanto modelos "misturados" constituídos de palavras geradas por diferentes distribuições (existe uma distribuição para cada rótulo válido). Um algoritmo *EM* (*"expectation-maximization"*) é utilizado para inferir do conjunto de treino os pesos

da "mistura" e as distribuições das palavras em cada componente da mistura.

Godbole & Sarawagi [2004] apresenta dois métodos para melhora de classificadores SVM aplicados com a transformação TUCCR. A ideia é que se estenda a base de dados original com $|C|$ características adicionais contendo as predições dos classificadores binários. Para a classificação de um novo documento, os classificadores binários da primeira iteração e os rótulos previstos são acrescentados às características originais formando-se um meta-documento. Este documento é então classificado novamente em busca de novos rótulos. Esta extensão leva em consideração as dependências entre os diferentes rótulos. Tsoumakas & Katakis [2007] argumenta que este método é uma especialização da aplicação do método Stacking Wolpert [1992], um método para a combinação de múltiplos classificadores.

3.1.2 Medidas de classificação Multi-Rótulo

Muitos algoritmos de classificação são algoritmos de otimização que buscam minimizar ou maximizar uma função objetivo. Estes algoritmos são propostos de forma a minimizar o erro mensurado por uma medida de qualidade da classificação. Grande parte dos algoritmos possuem uma fase de validação que decide com base nas medidas de erro os melhores parâmetros a serem utilizados no sistema de classificação. Por esta razão, o entendimento das diferentes medidas de erros existentes e como estas medidas se comportam têm uma importância fundamental no desenvolvimento destes algoritmos.

A medida *"one-error"* foi uma das primeiras medidas aplicadas a problemas de classificação multi-rótulo [Schapire & Singer, 1998]. Esta medida é uma medida simples que reflete o erro comum quando aplicado a sistemas de classificação único-rótulo adaptado para problemas multi-rótulo. É considerada a ocorrência de um erro, segundo esta medida, se a classe mais provável, segundo a hipótese de um classificador, não está dentre as classes corretas deste documento multi-rótulo.

Seja um classificador $\mathcal{H}(x) : X \rightarrow Y$ que assinala um único documento x escolhendo $\mathcal{H}(x_i) = argmax_{r \in Y} f(x, y)$. Dado um conjunto de documentos assinalados $S =< (x_1, Y_1), \cdots, (x_m, Y_m) >$ em que Y_i é o conjunto dos rótulos corretos do documento x_i:

$$\text{one-err}_s(\mathcal{H}) = \frac{1}{m} \sum_{i=1}^{m} [\mathcal{H}(x_i) \notin Y_i] \tag{3.2}$$

Schapire & Singer [1998] se propõe um algoritmo de *"boosting"* para problemas de categorização de textos multi-rótulo. Duas versões diferentes do algoritmo *Adaboost* foram apresentadas, tais versões se diferem pela medida de erro que é minimizada durante as iterações. Enquanto a primeira versão busca a minimização do *"hamming-*

loss" (Adaboost.MH), a segunda versão busca a minimização do *"ranking-loss"* (Adaboost.MR).

A medida *"hamming-loss"* foi proposta para que fosse possível comparar dois subconjuntos de rótulos. Esta medida calcula a distância *"hamming"* entre o vetor de assinalamento $\vec{Z}$ o vetor correto de rótulos $\vec{Y}$.

$$\text{hloss}(H, D) = \frac{1}{|D|} \sum_{i=1}^{|D|} \frac{|Y_i \Delta Z_i|}{|C|} \tag{3.3}$$

Onde Δ representa a diferença simétrica entre os dois vetores.

A versão Adaboost.MR foi proposta para minimizar a medida de *"ranking-loss"*. Esta medida foi proposta a partir da necessidade de se comparar problemas de classificação que se baseiam em um *"ranking"*. Espera-se que um um algoritmo capaz de gerar um *ranking* de rótulos, deve posicionar em suas primeiras posições os rótulos corretos (reais) do documento em questão. Portanto, a hipótese $f : \mathcal{D} \times \mathcal{C} \rightarrow \mathbf{R}$, onde $\mathbf{R}$ é um *ranking* dos rótulos mais prováveis. Comumente, os algoritmos geram uma função f que assinala um valor real para cada rótulo c_i, o posicionamento no ranking é então inferido através da ordenação real de $f(d, c_i)$. Sendo assim, c_1 antecede c_2 se $f(d, c_1) > f(d, c_2)$. Seja um exemplo (x, Y), esta medida se concentra apenas na ordem relativa de pares *cruciais* c_0 e c_1 tal que $c_0 \notin Y$ e $c_1 \in Y$. Consideramos que f equivoca na ordenação de um par *crucial* $f(x, c_1) \leq f(x, c_0)$, ou seja, a função f é ineficaz em apresentar o rótulo c_0 antes do rótulo c_1.

$$\text{rloss}(f) = \frac{1}{|D|} \sum_{i=1}^{|D|} \frac{1}{|Y_i||\bar{Y}_i|} |\{(y_1, y_2) | f(x_i, y_1) \leq f(xi, y_2), (y_1, y_2) \in Y_i \times \bar{Y}_i\}| \tag{3.4}$$

A cobertura (*"coverage"*) é um conceito trazido da área de recuperação de informação. Essa medida calcula até que posição no ranking das classes previstas pelo classificador precisamos percorrer na média até encontrar todas as classes corretas dos documentos. Na área de Recuperação de Informação, a cobertura reflete a precisão de um sistema de classificação quando se assume que a revocação é perfeita. Formalmente, definimos a cobertura de f com respeito a $S =< (x_1, Y_1), \cdots, (x_m, Y_m) >$ como:

$$cobertura_S(\mathcal{H}) = \frac{1}{m} \sum_{i=1}^{m} \max_{l \in Y_i} rank_f(x_i, l) - 1 \tag{3.5}$$

Algumas medidas tradicionais de classificação único-rótulo podem ser adaptadas para tratar o caso multi-rótulo. Nesta adaptação, a abordagem mais comum é que

diferentemente da medida de classificação único-rótulo em que cada documento recebe um peso único, estabelecemos que seja um documento $d_i = (X_i, C'_i)$, e C'_i o subconjunto de C dos rótulos assinalados ao do documento d_i, transformamos todo par X_i, c_i como um assinalamento. Com essa abordagem, calculamos a micro e macro acurácia, precisão, revocação, F1, dentre outras como na classificação tradicional único-rótulo (ver seção 2.4.2). Note que esta abordagem pode ser vista como a transformação do problema multi-rótulo em um problema único-rótulo e sua posterior avaliação. Esta transformação seria a TUCCR ou TRA, tais transformações são discutidas na seção 3.1.1.1.

3.2 Classificação Hierárquica

Nesta seção, apresentamos a definição do problema, as medidas de qualidade de classificação baseada em hierarquias, as vantagens destas medidas sobre as medidas de qualidade de classificação não-hierárquicas. Concluímos a seção com as principais dificuldade enfrentadas nos problemas de HMC.

3.2.1 Definição do Problema

Nesta seção, definimos conceitos fundamentais para a formulação do problema de classificação hierárquica. São também apresentados os relacionamentos e propriedades destes conceitos com os documentos textuais e as classes existentes.

Uma hierarquia $\mathcal{H} = (\mathbf{N}, \mathbf{A})$ é definida como uma grafo acíclico direcionado. Esse grafo consiste de um conjunto $\mathbf{N}$ de nodos e um conjunto $\mathbf{A}$ de pares ordenados, as arestas, tal que $(N_p, N_f) \in A \subseteq \{N \times N\}$. A aresta define um relacionamento direto de descendência entre o nodo pai N_p e o nodo filho N_f. A descendência é uma relação transitiva que define nodos antecedentes e descendentes. Assume-se a existência de um nodo *raiz* N_r, este nodo [2] não possui um antecedente direto (pai). Os nodos que não possuem descendentes são denominados nodos folhas, os demais nodos (não-folhas nem raiz) são chamados de nodos internos.

Cada classe C_i do problema de categorização está relacionada com um nodo N_i desta hierarquia $\mathcal{H}$. Então temos uma relação de um para um entre o conjuntos de nodos e o conjunto de classes$N \equiv C$ (em alguns casos podem ser criados nodos adicionais na hierarquia como por exemplo o nó raiz). Por simplificação, ao mencionarmos um nodo N_i poderemos utilizar diretamente a classe C_i relacionada a este nodo.

[2] A definição de nodo ou nó nesta dissertação é a mesma e ambos os termos podem ser utilizados sem nenhuma diferença de significado.

Os documentos D de uma hierarquia $\mathcal{H}$ contém um conteúdo textual e podem ser assinalados a uma ou mais classes. As classes a qual pertencem um documento d_i são chamadas de rótulos de um documento $\mathcal{L} = \{C_1 \dots C_l\}$. Os documentos são representados por um vetor de características $\vec{d_i} =< d_1, \dots, d_{|T|} >$ (ver seção 2.5).

As arestas na hierarquia representam um relação de constituição, "é parte de". Um documento assinalado a uma classe C_i descendente de C_j, também é considerada implicitamente como pertencente à classe C_j. As hierarquias virtuais são aquelas em que os documentos D só podem ser assinalados explicitamente a classes-folha.

3.2.2 Medidas de qualidade da Classificação baseadas em hierarquias

Uma adequada avaliação de classificadores em sistemas de classificação hierárquica precisa considerar a hierarquia. As medidas de qualidade das classificações planas não conseguem distinguir diferentes erros porque ignoram as relações existentes entre as classes da hierarquia. Estas hierarquias foram desenvolvidas de forma específica para categorização, desconsiderar tais relações significa desconsiderar as relações entre as classes definidas explicitamente por especialistas do problema. Por exemplo, uma classificação errada em que o nodo correto está muito próximo na hierarquia e em níveis bem mais profundos deve ser considerado um erro menos crítico que se o nodo correto está em um caminho da árvore distinto desde o primeiro nível da hierarquia. Ou seja, a proximidade na hierarquia deve ser considerada como uma proximidade no conceito semântico entre as classes e essa proximidade precisa estar representada na métrica de avaliação.

Antes de apresentar medidas de qualidade para problemas de HMC, apresentaremos o conceito de consistência hierárquica proposto por Kiritchenko [2005].

Definição 7 *Consistência Hierárquica: Seja um assinalamento C_i de um documento $d_i \in D$ tal que C_i é um subconjunto de rótulos $C_i \subseteq C$. Este assinalamento é considerado consistente dada uma hierarquia $\mathcal{H}$, se C_i inclui todos os conjuntos de antecessores $\forall c_k \in C_i$, se $c_k \in C_i$ e $c_j \in Antecessores(c_k)$, então $c_j \in C_i$.*

Definição 8 *Requisito para consistência hierárquica: Qualquer assinalamento produzido por um sistema de classificação hierárquica precisa ser consistente segundo a hierarquia de classes definida.*

3.2.2.1 Requisitos de uma medida de avaliação de classificação hierárquica

As medidas de qualidade em problemas de **HMC** devem considerar a estrutura hierárquica para a comparação entre distintas classificações. Com o objetivo de avaliar e propor uma nova medida de qualidade, Kiritchenko [2005] primeiro define os principais critérios necessários a uma medida de qualidade de classificação hierárquica:

Seja *H-Score*$(c_1|c_2)$ o valor da métrica de avaliação hierárquica ao assinalar um documento $d \in D$ à classe c_i dado que c_j a classe correta, *H-dist*$(c_i|c_j)$ o *comprimento do caminho mínimo não-direcionado* [3] entre os nodos c_i e c_j e $nivel(x)$ o comprimento do caminho mínimo da raiz ao nodo x:

1. Classificação parcialmente correta

   Uma boa medida de avaliação deve discernir classificações parcialmente corretas. Os assinalamentos a nodos que pertençam a uma mesma subárvore são preferíveis a nodos totalmente não relacionadas. Na figura 3.1(a) apresentamos uma hierarquia de atividades econômicas para a classificação de empresas segundo sua atividade econômica. No exemplo, vemos dois distintos assinalamento de uma empresa que exerce a atividade de "Extração de Minério de ferro". O primeiro assinalamento classificou a empresa como do setor de "Indústrias Extrativas", o segundo assinalamento classificou como setor de "pesca e/ou aquicultura". Intuitivamente, nota-se que o primeiro assinalamento é mais correto que o segundo. Ao analisarmos a hierarquia vemos que isto se reflete no fato de que a primeira classificação, diferentemente da segunda, foi correta pelo menos em alguns níveis mais generalizados (menos profundos) da hierarquia.

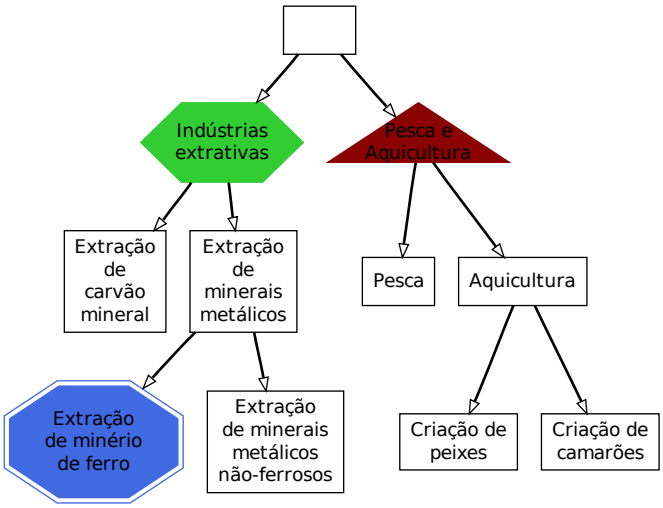
   Definição 9 *Para qualquer documento* $(d, c_i) \in D \times C$,
   se $Antecessores(c_j) \cap Antecessores(c_i) \neq \emptyset$ *e* $Antecessores(c_k) \cap Antecessores(c_0) = \emptyset$, *então* *H-Score*$(c_j|c_i) >$ *H-Score*$(c_k|c_i)$.

2. Discriminação de erros por distância

   Dados dois assinalamentos, o assinalamento que estiver a uma maior distância do assinalamento correto, deve ser considerado um pior erro que um assinalamento a uma menor distância.

   a) Um assinalamento que classificou corretamente um nível abaixo que outro deve ter melhor pontuação (*"score"*) Na figura 3.1(b), uma empresa do setor de "Extração de minério de ferro" foi classificado, no primeiro assinalamento

[3] As arestas da árvore são consideradas não-direcionadas.



(a) Classificação parcialmente correta

(b) Discriminação de erro por distância

(c) Discriminação de erro por distância

Figura 3.1: Diferenciação de erros em hierarquias. Em azul: a classe correta (sempre a classe G), em verde: um assinalamento com erro brando, em vermelho: assinalamento com erro pior

como uma empresa do setor de "Indústrias Extrativas" enquanto em um segundo assinalamento foi classificada como do setor de "extração de minerais metálicos". O segundo assinalamento, portanto, especificou melhor qual o tipo de atividade da empresa e essa característica se reflete na hierarquia já que a classe "extração de minerais metálicos" está mais próxima da classe

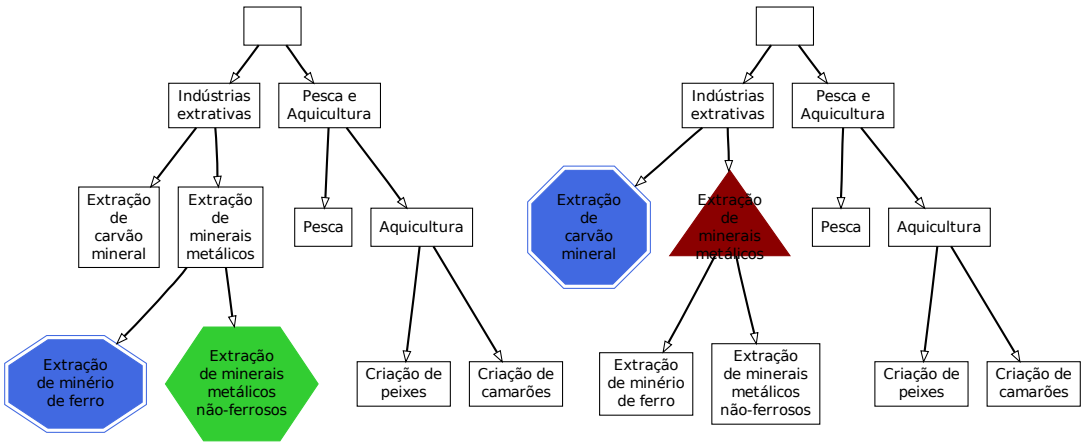


Figura 3.2: Discriminação de erro por profundidade. Em azul: a classe correta, em verde: um assinalamento com erro brando, em vermelho: assinalamento com pior erro

"Extração de minério de ferro" que a classe "Indústrias Extrativas".

b) Um assinalamento que aprofundou incorretamente na hierarquia deve ser considerado um pior erro que um assinalamento que foi realizado em um nodo antecessor.

Na figura 3.1(c), os assinalamentos na classe "Extração de carvão mineral" e outro em "Indústrias Extrativas", sendo a classe correta "Extração de minério de ferro". Uma vez que a empresa é de "extração de minério de ferro", classificá-la como de "Indústrias Extrativas" é correto mas é genérico. Já ao classificar a empresa como do setor de "extração de carvão mineral", o assinalamento incorreu diretamente em um erro. Portanto o primeiro assinalamento é preferível ao segundo pois está mais próximo que a classe correta.

Definição 10 *Para qualquer documento* $(d, c_i) \in D \times C$,
se $c_j = Antecessores(c_k)$ *e* $H\text{-}dist(c_j, c_i) > H\text{-}dist(c_k, c_i)$,
então $H\text{-}Score(c_j|c_i) < H\text{-}Score(c_k|c_i)$

3. Discriminação de erros por profundidade

Este requisito exige que se penalize erros nos níveis mais altos da hierarquia pois quanto maior a profundidade na hierarquia, mais específicos os conceitos para distinguir as classificações e portanto mais "aceitáveis" são os erros.

Na figura 3.2, um assinalamento na classe "Extração de minerais metálicos não-ferrosos" de uma empresa de "extração de minério de ferro" é um assinalamento mais aceitável que um assinalamento para a classe "extração de minerais metálicos" a uma empresa de "extração de carvão mineral".

Definição 11 *Para quaisquer documentos* $(d, c_1), (d, c_2) \in D \times C$, *se* $H\text{-}dist(c_1, c_1') = H\text{-}dist(c_2, c_2')$, $nível(c_1) = nível(c_2) + \delta$, $nível(c_1') = nível(c_2') + \delta$ *e* $\delta > 0, c_1 \neq c_1', c_2 \neq c_2'$, *então* $H\text{-}Score(c_1'|c_1) > H\text{-}Score(c_2'|c_2)$

Todos estes requisitos, possuem um conceito semântico consistente quando aplicado a uma hierarquia específica desenvolvida por especialistas. Ou seja, respeitam o conhecimento dos especialistas inserido na estrutura hierárquica. Mesmo um não-especialista da área de classificação de atividades econômicas consegue concluir várias relações intrínsecas entre as atividades econômicas apenas analisando a estrutura hierárquica da figura 3.1.

3.2.2.2 Medidas de avaliação de classificação hierárquica

Kiritchenko [2005] propõe uma extensão de medidas tradicionais de avaliação da classificação para o problema de Classificação Multi-Rótulo Hierárquica como precisão, revocação e medida F. Estas definições baseiam-se no conceito de consistência hierárquica (definição 7). A extensão das medidas tradicionais é realizada por uma fase anterior de transformação da base de dados e posteriormente realiza-se o cálculo das medidas da mesma forma que as definições originais. Esta transformação consiste da explicitação dos rótulos implicitamente assinalados.

Definição 12 *Expansão de rótulos baseada em consistência hierárquica: Seja o documento* (d_j, C_i), $d_j \in D$, $C_i \subseteq C$ *que foi assinalado a um subconjunto de rótulos* $C_i' \subseteq C$, *a expansão destes subconjuntos de rótulos* C_i *e* C_i' *é realizada definindo-se os novos subconjuntos* $\hat{C}_i e \hat{C}_i'$ *adicionando-se todos os rótulos antecessores implícitos:* $\hat{C}_i = \{\cup_{c_k \in C_i} Antecessores(c_k)\}$, $\hat{C}_i' == \{\cup_{c_l \in C_i'} Antecessores(c_l)\}$.

Após a expansão dos rótulos corretos e assinalados, as métricas de precisão, revocação e medida F são calculadas sobre os subconjuntos expandidos e chamadas de hP (*"hierarchical Precision"*) hR (*"hierarchical recall"*). A medida F calculada com base em hP e hR é chamada de hF_β. Kiritchenko [2005] demonstra que a medida hF1 satisfaz todos os requisitos da avaliação hierárquica (seção 3.2.2.1). Além disto, a medida é simples, de fácil generalização, consistente e discriminativa.

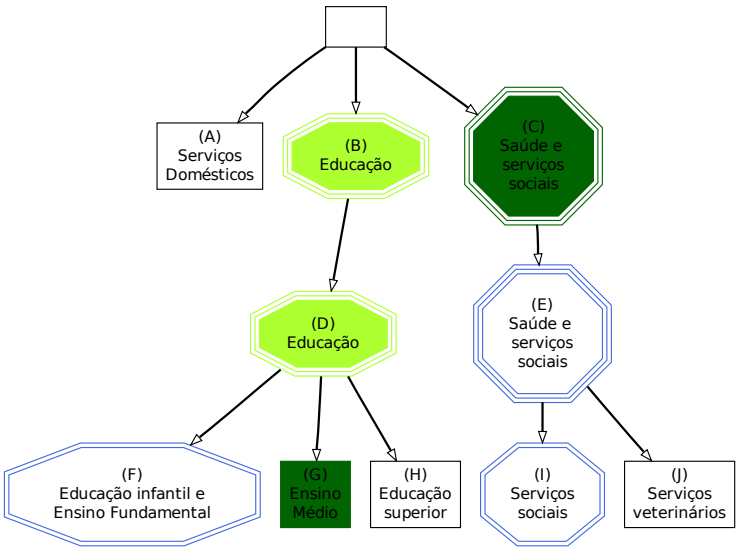


Figura 3.3: Expansão de rótulos baseada em consistência hierárquica

Na figura 3.3, podemos ver uma empresa que foi classificada como do setor de "saúde e serviços sociais" e "ensino médio", i. e. $C'_i = \{C, G\}$ e a classificação correta seria de uma empresa de "educação infantil e ensino fundamental" e "serviços sociais", i.e. $C_i = \{F, I\}$. A expansão dos rótulos corretos e assinalados (previstos) resultam nos subconjuntos $\hat{C}'_i = \{B, C, D, G\}$ e $\hat{C}_i = \{B, C, D, E, F, I\}$. O conjunto de rótulos assinalados corretamente é $\hat{C}_i \cap \hat{C}'_i = \{B, C, D\}$, portanto, o número de rótulos assinalados no subconjunto expandido é $|\hat{C}'_i| = 4$, o número de rótulos corretos no subconjunto expandido é $|\hat{C}_i| = 6$ e o número de rótulos corretamente assinalados $|\hat{C}_i \cap \hat{C}'_i| = 3$. Sendo assim, $hP = \frac{|\hat{C}_i \cap \hat{C}'_i|}{|\hat{C}'_i|} = \frac{3}{4}$, $hR = \frac{|\hat{C}_i \cap \hat{C}'_i|}{|\hat{C}_i|} = \frac{3}{6}$ e $hF_1 = \frac{hP.hR}{hP+hR} = \frac{\frac{3}{4}.\frac{3}{6}}{\frac{3}{4}+\frac{3}{6}} = \frac{3}{10}$.

Cesa-Bianchi et al. [2006] apresenta uma nova medida de erro baseada em uma hierarquia, *"h-loss"*. Essa medida considera que devemos penalizar apenas o primeiro erro cometido durante o caminhamento da hierarquia [Cesa-Bianchi et al., 2006]. Ou seja, uma vez que um classificador comete um erro de assinalamento em um nodo mais alto da hierarquia, os erros de classificação de nodos descendentes deste nodo não devem ser considerados (ignora-se a propagação de erros). Propõe-se então um algoritmo linear que se baseia nessa medida. Ressalta-se aqui que os autores não explicam porque esta medida é válida.

Existem ainda medidas que se baseiam em um "ranking". Todavia, tais medidas são complicadas de serem utilizadas, pois é difícil comparar a certeza de assinalamento entre classes com diferentes níveis de dificuldade [Granitzer, 2003].

3.2.3 Dificuldades do Problema de HMC

Os problemas de HMC (classificação hierárquica multi-rótulo) possuem várias dificuldades em comum. Nesta seção apresentamos as principais características destes problemas e porque quais as dificuldades inerentes à classificação nestes cenários.

Em geral, as hierarquias nos problemas de **HMC** são geradas quando o número de classes $|C|$ é acentuado de forma que uma hierarquia que permita encapsular detalhes das categorias em cada nível facilita o entendimento e análise das diversas classes existentes. Um grande número de classes acarreta em uma série de dificuldades para os algoritmos de aprendizagem de máquina:

1. Dificuldade inerente da classificação

   Note que em uma classificação único-rótulo, um classificador aleatório tem uma chance de $1/|C|$ de acertar. Em problemas de classificação tradicional o número típico de classes varia entre 2 e 10 e assim, um classificador aleatório possui chances expressivas de acertar. Nos problemas de classificação hierárquica, em geral, o número de classes pode variar desde centenas a centenas de milhares de classes, a decisão portanto é muito mais difícil, por exemplo, a chance de acerto do classificador aleatório se torna bastante inexpressiva. Sendo um problema de classificação multi-rótulo, a dificuldade é ainda maior.

2. Dificuldade de obtenção de bases de dados para treinamento

   Essa dificuldade inerente da classificação se traduz também na classificação manual. Se considerarmos que qualquer algoritmo de aprendizagem precisa induzir e generalizar uma função de distribuição encontrada na base de treinamento, a dificuldade de se obter uma base de treinamento confiável é considerável. Muitas destas bases reais precisam ser classificadas manualmente para comporem a base de treinamento. Uma abordagem alternativa é um esquema de criação de bases de dados semiautomática. Isto é, dada um subconjunto da base já classificada, utilizam-se métodos de classificação para proporem novos rótulos para documentos não classificados. Se a predição desta classe é extremamente confiável, assinala-se o rótulo, senão um especialista é utilizado para assinalar os rótulos mais incertos Lewis et al. [2004]. Note que essa abordagem apesar de prática pode gerar bases de dados "viciadas" em relação ao método de aprendizagem auxiliar utilizado durante a fase de classificação semi-manual.

   Além da dificuldade de assinalamento de um documento em específico, pois existe o número muito grande de classes. É necessário também que obtenhamos um número significativo de documentos representantes de cada classe para que possamos

aprender sobre o conjunto de treino. Tipicamente, na literatura, a quantidade de informação necessária pode variar bastante de algoritmo para algoritmo mas utilizam-se em geral de 100 a 1000 documentos para cada classe, a existência de apenas 10 documentos por classe é considerada um cenário em que técnicas especiais precisam ser consideradas [Toutanova et al., 2001]. Se assumirmos que os documentos são assinalados a um pequeno número de rótulos, podemos supor que precisaríamos de cerca de $1000 \times |C|$ documentos durante a fase de treinamento. Para um problema com cerca de 1000 classes, isso significa que precisamos de ao menos 10^6 documentos. Mesmo assim, estamos pressupondo que existe balanceamento entre as classes, ou seja, o número de documentos por classe não varia muito, o que na verdade é bastante raro.

Temos dois grandes subproblemas originados da necessidade de grandes bases de dados de treinamento, teste e validação. Primeiramente, esta base de dados deve ser classificada manualmente. Desnecessário dizer que o trabalho de construção de uma base de dados significativa para este tipo de problema é impraticável para ser realizada por um pequeno número de especialistas. Isso nos leva a um segundo problema, um grande número de especialistas são necessários para se gerar as bases de dados. Este fato pode levar a diferentes políticas de assinalamento por parte de diferentes analistas, acrescido do fato de que a classificação manual também é um ponto crítico, as bases de dados manuais resultantes possuem uma quantidade considerada de erros, inconsistências dentre outros problemas.

Desta forma, as bases de dados utilizadas para o treinamento dos algoritmos de classificação, costumam apresentar uma qualidade menor e é necessário que os sistemas de classificação e sistemas de avaliação da qualidade da classificação considerem este fator.

3. Suscetibilidade a ruídos, superajuste

   Como mencionamos anteriormente, existe grande dificuldade inerente à classificação dos problemas de HMC e com isso a qualidade das bases de dados fica comprometida. Neste contexto, tornam-se necessários que os modelos aprendidos desta base de dados sejam robustos de forma a não sofrer do superajuste (seção 2.7.1).

4. Diferentes níveis de especialização dos conceitos

   À medida em que nos aprofundamos na hierarquia encontramos classes com cada vez maior granularidade e mais específicas. Com isso, a proximidade semântica e a quantidade de termos em comum entre as classes é maior. Assim, existem menos

termos discriminativos entre as classes e a diferenciação destas classes pode ser bastante complicada. Além disto, diferentes termos podem ser discriminativos a cada nível da hierarquia. Esta variada especialização além da alta especialização nos níveis mais profundos da hierarquia é um desafio complicado principalmente se um método de seleção de características inadequado for utilizado em fases anteriores à classificação (ver seção 3.3.1).

5. Desbalanceamento de classes e tendenciosidade

   A base de dados hierárquica em geral é extremamente complexa, enquanto existe quantidade abundante de documentos para uma classe, outras classes podem possuir uma quantidade inexpressiva de documentos. Por exemplo, em um sistema de classificação de atividades econômicas, a quantidade de empresas que são condomínios, cerca de 10% das empresas constituídas em Vitória e Belo Horizonte, é muito alta. Por outro lado, a quantidade de empresas de "extração de materiais metálicos não-ferrosos" é muito pequena.

   Este desbalanceamento nos induz a dois problemas, primeiramente muitos dos algoritmos tradicionais assumem um certo balanceamento entre as classes. Não raro, encontramos trabalhos na literatura em que as classes mais raras são desconsideradas na avaliação dos sistemas. Um exemplo desta prática é a base de dados da ACM de artigos da área de computação, a grande maioria dos trabalhos de classificação avaliam a qualidade de seus sistemas no primeiro nível da hierarquia de assuntos da ACM. O mais interessante é que existem no total 12 classes diferentes, como 4 destas classes são raras, muitos trabalhos utilizam a base de dados apenas do primeiro nível excluindo-se as 4 classes mais raras [Veloso et al., 2007]. Isso nos mostra que o desbalanceamento é um problema crítico já nos primeiros níveis e tende a se agravar ainda mais à medida que nos aprofundamos em uma hierarquia.

6. Quantidade insuficiente de dados e raridade de classes

   Como mencionado acima, os problemas de HMC precisam lidar com a raridade das classes e a falta de quantidade de informação já que obter uma quantidade de dados abundante é muitas vezes impraticável nos sistemas reais [Liu et al., 2005; Manning et al., 2008].

   Por essa razão, é necessário que o sistema de classificação consiga induzir uma boa função/hipótese para uma série de classes raras com base em uma quantidade muito pequena de informação. Para atendermos a este requisito, podemos propor a utilização de algoritmos com alta variância e baixo viés. Porém, isto nos leva a

um paradoxo, um outra grande problema das bases de dados dos problemas de HMC é a existência inerente de ruído. Evitar que este ruído induza os sistemas ao superajuste, faz com que necessitemos de um sistema de classificação com alto viés e baixa variância. Obviamente, encontrar um modelo que satisfaça tais compromissos antagônicos é uma tarefa não-trivial.

7. Alta demanda de poder computacional

   Os algoritmos de aprendizagem e classificação precisam lidar com um número muito grande de classes, muitos destes algoritmos não escalam bem em relação ao número de classes, o que pode ser crítico. Outros métodos necessitam que sejam induzidos múltiplos conceitos, um para cada nodo da hierarquia a até múltiplos conceitos para cada nodo da hierarquia ou combinações de nodos (seção 3.1.1.1). Conforme já mencionamos, as bases de dados de problemas de **HMC** precisam ter um tamanho muito maior que as bases de dados da classificação tradicional, para que um número razoável de documentos de cada classe na base de treinamento exista. Além disto, os problemas de **HMC** apresentam níveis diferentes de especialização de conceitos à medida que se aprofunda na hierarquia. Esta especialização variada do assunto por nível da hierarquia, gera a necessidade de um número mais acentuado de características que os métodos tradicionais já que a classificação dos nodos mais altos da hierarquia em geral demandam termos mais genéricos enquanto os nodos mais profundos da hierarquia necessitam de termos bem mais específicos. As técnicas de classificação para os problemas de **HMC** precisam escalar tanto em relação ao número de classes, ao tamanho do vocabulário (quantidade de características) e ao número de documentos na base de treinamento e teste, estes três fatores tendem a ser críticos em problemas de **HMC**, assim, a escalabilidade de tais algoritmos se torna um fator crítico.

3.3 Explorando a hierarquia

Os algoritmos de classificação hierárquica podem ser organizados de acordo com a fase em que o conhecimento da hierarquia é utilizado. Obviamente, a hierarquia pode ser explorada em mais de uma fase.

1. Pré-processamento e seleção de características baseada em hierarquias

2. Transformações do problema (expansão de rótulos, replicação de documentos dentre outros)

3. Aprendizagem/Classificação

4. Pós-processamento

A seguir apresentaremos as principais abordagens de técnicas que se aplicam em cada fase.

3.3.1 Pré-processamento baseado em hierarquias

Uma das principais tarefas para um bom sistema de classificação é a representação correta dos documentos. Em particular é comum ignorar-se a ordem dos termos e utilizar a representação ‘‘bag-of-words”. A seleção de características ‘‘feature selection” é um processo essencial para um bom sistema de classificação de textos que objetiva a seleção das características mais informativas e a remoção das características que apenas geram ruído para o modelo em questão.

Manning et al. [2008] afirma que a complexidade do aprendizado depende não somente do classificador a ser utilizado mas também de vários outros aspectos, sendo um deles a seleção de características. Em sua discussão sobre viés-variância, ele destaca que um classificador a princípio mais complexo, como um classificador polinomial quadrático composto de dois parâmetros, pode ser menos “poderoso” que um classificar linear com 10.000 dimensões. Com isto, ele pondera que a análise da complexidade não deve focar apenas no algoritmo de aprendizado, mas também em fases importantes como a seleção de características.

Em uma classificação hierárquica a necessidade e complexidade da tarefa de seleção de características fica clara (conforme discutimos na seção 3.2.3. Naturalmente os conceitos representados por cada categoria, nodo da árvore, tendem a se aprofundar de mais genéricos para cada vez mais especializados. Portanto, quanto mais profundo um nodo está na hierárquica tende-se a uma maior especialização de seu assunto. Por esta razão, os termos que são relevantes para a diferenciação entre nodos de níveis mais profundos da hierarquia são em geral bem diferentes dos termos relevantes para a diferenciação dos nodos de níveis mais altos.

Além disso, um dos objetivos da seleção de características é a simplificação do modelo, diminuindo sua complexidade e dificuldade de computação. Uma vez que muitas destas técnicas precisam apresentar uma boa escalabilidade para serem efetivamente utilizadas em bases de dados reais de HMC, a redução da quantidade de características torna-se ainda mais necessária no contexto de classificação hierárquica pois vários modelos podem ser necessários para a classificação ou mesmo os modelos podem ter um

número muito grande de classes, causando um aumento intenso da carga computacional necessária para a aprendizagem.

Os métodos de seleção de características podem ser aplicados de forma global ou local. Os métodos globais selecionam as características que diferenciam os termos entre todas as classes que compõem a hierarquia enquanto os métodos locais em geral selecionam as características avaliando uma parte especializada da árvore hierárquica, por exemplo, uma subárvore.

3.3.2 Classificação Hierárquica

Existem diversos algoritmos sobre classificação hierárquica. Estes algoritmos podem ser divididos em dois grandes grupos globais (*"big-bang"*) ou locais (*"top-down"*) [Sun & Lim, 2001; Kiritchenko, 2005].

Definição 13 *Categorização de Textos Hierárquica Global: Um método é chamado de global se a construção do conceito do classificador é realizado uma única vez para a discriminação de* ***todas*** *as categorias em uma hierarquia.*

Segundo Kiritchenko [2005], os métodos globais diferem das abordagens planas pois de alguma forma elas consideram os relacionamentos entre as categorias. Em nosso trabalho sempre que referenciarmos a classificação plana, estamos referenciando a classificação a definição de classificação global.

Definição 14 *Categorização de Textos Hierárquica Local: Um método de aprendizagem é chamado de local se para cada nodo de uma hierarquia é realizada uma construção do conceito do classificador.*

Os métodos locais geralmente se baseiam em uma estratégia *"top-Down"* de classificação em que primeiramente se classifica o documento nas categorias nos níveis mais altos da hierarquia, e recursivamente navega-se nesta hierarquia realizando as decisões nas categorias nos níveis mais profundos. Esta intuição de tais técnicas está intrinsecamente ligada à forma como os especialistas realizam a classificação manual. Na classificação manual, apenas os filhos de um nodo são analisados por vez, o que encapsula os detalhes e permite que a cada nível ou nodo da árvore se concentre nos detalhes específicos deste nível/nodo. A mesma estratégia é utilizada para os métodos automáticos, o problema de **HMC** é subdivido em subproblemas em que cada subproblema gerado possui um número de classes $|C|$ bem menor que o número de classes totais. Em aprendizagem de máquina é conhecido experimentalmente que quanto maior o número de classes de um problema, maior sua dificuldade inerente. Além disso, uma vez

classificado nos níveis mais altos baseando-se em problemas mais simples, é possível ignorar uma série de classes nos níveis mais baixos da hierarquia onde as classificações tendem a ser mais difíceis (seção 3.2.3).

Note que os modelos Top-Down, chamados de **PAM** (*"pachinko allocation machine"*), tomam decisões que não podem ser reconsideradas. O maior problema enfrentado nestes modelos é a propagação de erros, se um documento não é classificado em uma classe antecessora não é possível se recuperar deste erro e conseguir reclassificá-lo.

3.3.2.1 Classificação Hierárquica Global

Nesta seção apresentamos os principais trabalhos de Classificação Hierárquica Global (**CHG**).

McCallum et al. [1998] se baseia na hierarquia para executar um método chamado de *shrinkage of classes* para obter melhores estimativas dos parâmetros de um modelo probabilístico bayesiano. O termo *"shrink"* se refere ao efeito de "encolhimento dos parâmetros", isto é, o método de *"shrinkage"* executado suaviza as diferenças entre os parâmetros a priori estimados de nodos próximos da hierarquia. A intuição por trás deste algoritmo é que no caminho a qualquer nodo, os parâmetros do modelo não deveriam ser alterados bruscamente. Assim, os parâmetros são "suavizados" evitando alterações bruscas entre um nodo pai e um nodo filho.

Toutanova et al. [2001] estende a abordagem de McCallum et al. [1998] e mostra que um "modelo hierárquico misturado" (*"hierarchical mixture model"* consegue melhorar o desempenho dos algoritmos principalmente quando as classes possuem poucos documentos de treino.

3.3.2.2 Classificação Hierárquica Local

Sun & Lim [2001] propõem um dos primeiros algoritmos *"top-down"* SVM. Esta abordagem se difere da abordagem de Dumais & Chen [2000] por separar um classificador local e um classificador de subárvore. Assim, o modelo possui dos modelos de treinamento distintos, o primeiro para tomar a decisão se o documento deve ser assinalado a um certo nodo da hierarquia e o segundo se o documento deve continuar o percorrimento na árvore com a navegação "top-down". Este trabalho argumenta que a exploração da hierarquia por Dumais & Chen [2000] não apresenta melhoras significativas.

Sun [2004] defende que um dos maiores problemas dos algoritmos de classificação *"top-down"* é a propagação de erros e mais especificamente, a "blocagem" (*"blocking"*). A "blocagem" se refere aos documentos equivocadamente rejeitados nos níveis mais altos da hierarquia que não chegam a ser classificados nos níveis mais profundos da

hierarquia pois não há reconsideração de decisões. Sun [2004] propõe três diferentes métodos para lidar com a blocagem utilizando o algoritmo proposto em Sun & Lim [2001]:

1. Método Redução de limiar (TRM, *"Threshold Reduction method"*);

2. *"Restricted Voting Method"* (RVM);

3. Método multiplicativo estendido (EMM, *"Extended Multiplicative Method"*).

TRM se baseia no princípio da técnica de redução de limiar. Esta técnica utiliza valores de limiar "brandos" que permitam que mais documentos passem para os níveis inferiores da hierarquia. RVM enfrenta o problema de blocagem permitindo que os classificadores de nível mais profundo acessem os documentos antes que eles sejam rejeitados pelo nodo pai. Assim, duranta a decisão de assinalamento a um nodo, todos os documentos que foram considerados por seu nodo "avô" podem ser utilizados. O método EMM, multiplicativo estendido, foi derivado do método multiplicativo proposto por Dumais & Chen [2000]. EMM associa um classificador local para cada folha e um classificador subárvore para cada nodo interno. Seja c_n um nodo folha no nível n com nodo pai c_{n-1}. Um documento d_j é assinalado à classe c_n se $P(c_n|d_j) \times P(c_{n-1}.s|d_j) \geq \theta_{c_{n(n-1)}}$, onde $\theta_{c_{n(n-1)}}$ é um limiar. Similarmente, d_j pode ser aceito por um classificador subárvore associado à classe c_{n-1} se $P(c_{n-1}.s|d_j) \times P(c_{n-2}.s|d_j) \geq \theta_{c_{(n-1)(n-2)}}$. Os autores mencionam que este método pode ser aplicado para mais níveis, mas os experimentos se limitaram apenas a dois níveis de profundidade.

Sun [2004] conclui que **RVM** apresentou o melhor desempenho dentre as três estratégias, porém, cabe ressaltar que os experimentos foram realizados em uma hierarquia com apenas três níveis de profundidade. O autor força a utilização de um mesmo limiar para todos os nodos de um mesmo nível da árvore, esta abordagem possui problemas que discutiremos mais a frente (5) mas este problema possui incidência limitada em hierarquias menores como as utilizadas neste trabalho.

Em um estudo sobre a escalabilidade de algoritmos SVM, Liu et al. [2005] realizou um estudo experimental baseando-se em todo o diretório Yahoo! e o algoritmo SVM "top-down" apresentou resultados modestamente superiores à classificação plana. O foco deste trabalho estava na escalabilidade, e neste quesito, a estratégia local "top-down" foi muito superior à classificação plana.

3.3.3 Pós-processamento baseado em hierarquias

Muitos algoritmos baseados em hierarquia, independente da arquitetura de classificação, podem realizar métodos de pós-processamento. Algumas arquiteturas de sistemas de classificação **HMC** dependem que seja realizado uma fase de pós-processamento. Primeiramente, abordamos uma técnica muito simples de pós-processamento, esta técnica é um método muito simples para satisfazer o requisito de consistência hierárquica (def. 8). Este método consiste da expansão dos rótulos assinalados como proposto por Kiritchenko [2005] (def. 12). Definimos este pós-processamento com o nome de *consistência hierárquica*.

Na literatura, encontramos dois trabalhos em que o cerne está em explorar a hierarquia como pós-processamento: Bade et al. [2006] e Punera & Ghosh [2008]. Bade et al. [2006] propõem um método de pós-processamento baseado em uma técnica conhecida por **EUM** *"Expected Utility Maximizatin"*. Segundo esta abordagem, assume-se que um classificador $H : D \times C$ é construído realizando predições e gerando resultados como valores reais que expressam a confiança no assinalamento deste rótulo, posteriormente, os autores definem uma função de utilidade que se baseia na hierarquia do problema. Um método de otimização é aplicado para maximizar a utilidade esperada e a predição é realizada com base no resultado desta nova fase. Bade et al. [2006] mencionam que seu método de pós-processamento é a aplicação da classificação sensível a custo (*"cost-sensitive classification"*) em que a matriz de custos é determinada pela estrutura hierárquica das classes. Apesar disto, os autores preferem modelar o problema como "Expected Utility Theory" (EUT) onde as ações correspondem às previsões e os estados às classes, porque essa abordagem permite interessantes extensões. A motivação desta ideia é a de um usuário buscando localizar uma informação em uma hierarquia, assim, a utilidade máxima é a de assinalar um nodo ao qual pertence o documento e a utilidade decresce à medida em que a distância na hierarquia entre o nodo correto e o nodo previsto aumenta $H - dist(\hat{c}, c)$. Na tabela 3.6 é mostrada a matriz de utilidade, em princípio esta matriz corresponde à matriz de custos da classificação sensível a custos. Os autores ainda estendem as medidas de precisão, revocação e acurácia integrando-se a função de utilidade (equação 3.6) nestas medidas.

$$util(\hat{c}|c) = \begin{cases} exp(\frac{-\text{H-dist}(\hat{c},c)}{2}) & \text{se } \hat{c} \in \text{Antecessores}(c) \text{ ou } \hat{c} = c, \\ 0 & \text{senão} \end{cases} \tag{3.6}$$

Kiritchenko [2005] argumenta que também tentou aplicar técnicas sensíveis a custo. A estratégia de estabelecer que quanto maior a distância entre duas classes

	ρ_1	ρ_2	$\cdots$	$\rho_{\|C\|}$
	c_1	c_2	$\cdots$	$c_{\|C\|}$
c_1	u_{11}	u_{12}	$\cdots$	$u_{1\|C\|}$
c_2	u_{21}	u_{22}	$\cdots$	$u_{2\|C\|}$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
$c_{\|C\|}$	$u_{\|C\|1}$	$u_{\|C\|2}$	$\cdots$	$u_{\|C\|\|C\|}$

Tabela 3.6: Matriz de utilidades

na hierarquia maior o custo do erro foi aplicada para o preenchimento da matriz de custos. Foram utilizadas duas técnicas de classificação sensível a custos, o *MetaCost* [Domingos, 1999] e o C5.0, uma versão comercial do algoritmo C4.5 que possui algumas funcionalidades adicionais, como custos de classificações incorretas variáveis. Experimentalmente, Kiritchenko executou os dois algoritmos na base de dados "20 newsgroups" e argumentou que os resultados usando C5.0 apesar de terem sido levemente superiores à classificação com base em custos uniformes, não foram satisfatórias. A autora intui que os baixos ganhos foram causados pelas probabilidades das previsões muito mal calibradas geradas pelos algoritmos baseados em árvores de decisão. Já o algoritmo MetaCost apresentou resultados muito inferiores ao classificador base não-sensível a custos.

Punera & Ghosh [2008] propõem um método de pós-processamento para suavizamento das saídas dos classificadores como um problema regressão de uma árvore isotônica regularizada (*"regularized isotonic tree regression problem"*) e apresenta um método de solução ótima baseada em programação dinâmica. Neste trabalho, propõem-se duas propriedades de restrições de monotonicidade da classificação separadas em dois distintos cenários, no primeiro cenário a hierarquia é virtual e no segundo cenário a hierarquia é não-virtual. Aqui apresentamos as definições apenas do segundo cenário, a diferença é que na classificação virtual a restrição é *estrita* enquanto na classificação não-virtual a restrição é *branda*.

Definição 15 *Monotonicidade de classificação branda (relaxed classification monotonicity) O score de um classificador para um nodo da árvore é sempre maior ou igual que o score do classificador de seus nodos filhos.*

Definição 16 *Dado os scores do classificador $x(.)$, encontre os scores suavizados $y(.)$ que minimize*

$$\sum_{c \in C} w_c \times |x(c) - y(c)| + \sum_{c \in C, u_i = Filho(c)} \gamma_c \times (y(c) - max\{y(u_i)\}) \tag{3.7}$$

restrito à Monotonicidade de classificação branda (def. 15), onde w_c e γ_c são os pesos e as penalidades específicas para cada nodo.

3.4 Estado da arte

A seguir apresentamos os principais trabalhos na área assim como as principais características e limitações das mesmas.

Steinbruch [2006] propõe dois diferentes algoritmos utilizando como método de classificação básico modelos bayesianos multinomiais para tratar multi-rótulo. A primeira proposta se baseia na transformação TCP (tabela 3.3). A segunda proposta é semelhante à proposta de Godbole & Sarawagi [2004]. Para a classificação de um novo documento, realizam-se iterações em que os rótulos previstos nas primeiras iterações são acrescentados às características originais. Este documento é então classificado novamente em busca de novos rótulos. Esta extensão leva em consideração as dependências entre os diferentes rótulos. [Veloso et al., 2007] propôs esta mesma estratégia aplicada a algoritmos de classificação associativa postergada.

Cesa-Bianchi et al. [2006] apresenta um algoritmo, **H-RLS** (*"Hierarchical Regularized Least Squares"*), que incrementalmente constrói um classificador de limiar linear para cada nodo da hierarquia. Este algoritmo "top-down" é então comparado com algoritmos baseado em Perceptrons (arquitetura plana e Top-down), SVM (arquitetura plana e Top-Down) e uma versão plana S-RLS (Regularized Least Squares).

Mathias Seeger explora a hierarquia para diminuir o tempo de execução dos algoritmos lineares[Seeger, 2008]. Kiritchenko [2005] utiliza uma estratégia global e o algoritmo Adaboost.MH para seus experimentos. Primeiramente, ela defende que um único limiar de assinalamento não pode ser utilizado quando se tratam problemas de classificação com muitas classes. Como sua abordagem é global, ela estende o algoritmo Adaboost.MH para utilizar um limiar para cada classe separadamente.

Juho Rousu propôs um método baseado no SVM para explorar a hierarquia[Rousu et al., 2006].

Capítulo 4

Bases de Dados

Neste capítulo apresentamos a base de dados que utilizamos em nossos resultados experimentais. Uma caracterização e discussão sobre esta base de dados é realizada. Como discutido na seção 3.2.3 uma das tarefas mais complicadas para a análise de algoritmos para o problema de HMC é a obtenção de bases de dados confiáveis e extensas. Em geral, tais problemas precisam de um número de documentos muito maior que os problemas de classificação tradicionais exigindo um esforço manual grande para o desenvolvimento de bases de dados confiáveis para o estudo dos algoritmos. Por conta desta dificuldade, a literatura na área de técnicas de *HMC* ainda carece de um *benchmark* bem estabelecido.

4.1 CNAE

Nesta dissertação focamos a aplicação das técnicas de *HMC* em uma nova aplicação, a classificação automática de atividades econômicas. Nesta aplicação, deseja-se determinar quais possíveis atividades econômicas, reconhecidas e regulamentadas no país, uma empresa exerce dada uma descrição textual das atividades a serem executadas por esta empresa.

Este problema existe na maioria dos países do mundo, cada país determina as atividades econômicas regulamentadas, os códigos de atividades econômicas geralmente estão organizadas de forma hierárquica com base em um padrão internacional de classificação, o International Standard Industrial Classification of All Economic Activities (ISIC [1]). No Brasil, foi desenvolvida uma tabela específica com base na tabela ISIC chamada de CNAE. Outros países utilizam tabelas bem parecidas com a CNAE que também aderem ao padrão ISIC como é o caso do *NAICS - North American Industry*

[1] http://unstats.un.org/unsd/cr/registry/regcst.asp?Cl=17

Classification System[2] utilizado nos Estados Unidos. Os estudos de classificação automática de atividades econômicas é portanto altamente aplicável em outros países e representa um problema extremamente importante e desafiador.

O governo, nas diversas esferas precisa classificar tais empresas segundo a sua atividade econômica. É o código de atividade econômica que é utilizado em várias instâncias durante a fase de vida da empresa para a definições de vários quesitos que permeiam o seu processo produtivo. Por exemplo, as instituições financeiras se baseiam no código de atividade econômica principal para definirem o ramo de atividade da empresa e consequentemente, o risco inerente a seu financiamento. São os códigos de atividade econômica também que definem se uma empresa precisa de licença ambiental ou não. Vários impostos e taxas incidem ou não a uma empresa com base no seu código de atividade econômica. As várias instituições de estatística se baseiam no código de atividade econômica da empresa para extraírem estimativa de renda, produção dentre outros. Todos os estudos que diferenciam os diferentes setores da economia se baseiam em tal informação para definirem se uma empresa pertence a qual setor da economia. Fica evidente portanto que a correta classificação das empresas segundo a CNAE é de extrema importância fiscal, econômico e financeira.

Estima-se que a cada ano cerca de 300 mil empresas são criadas ou alteram o seu contrato social exigindo que uma nova classificação das atividades econômicas seja realizada [de Souza et al., 2007]. A dificuldade, quantidade e complexidade desta tarefa motiva-nos a estudar técnicas automáticas de classificação que podem constituir tanto uma ferramenta de auxílio aos analistas, como ferramentas não supervisionadas ou de auditoria.

Recentemente foram iniciados os estudos de técnicas de classificação de atividades econômicas [de Souza et al., 2007]. Estes trabalhos apresentam resultados em uma base de dados muito pequena em que uma quantidade muito pouco expressiva de atividades econômicas ocorrem na base de dados, a avaliação de técnicas neste subconjunto tão pequeno não consegue demonstrar a complexidade real do problema. Além disto, estes trabalhos realizam uma classificação plana e não exploram a hierarquia expressa na tabela CNAE. As tabelas ISIC, CNAE e NAICS são resultado de um extenso trabalho de especialistas, ignorar esta hierarquia significa ignorar informações extremamente importantes que podem ser vitais ao se realizar a classificação em bases de dados maiores. Até onde sabemos ainda não existe o estudo de técnicas de classificação hierárquica multi-rótulo para este problema.

A tabela *CNAE-subclasses* é definida por uma hierarquia com 5 níveis de pro-

[2] www.census.gov/naics/

fundidade chamados respectivamente de seção, divisão, grupo, classe e subclasse. As empresas precisam ser classificadas segundo os códigos das subclasses a que pertencem, isto é, o quinto nível da hierarquia. Não é possível realizar a classificação em atividades econômicas a não ser no nível de subclasses, portanto a CNAE é uma hierarquia *virtual* com cerca de 1200 subclasses [3].

Esta base de dados foi disponibilizada pela Receita Federal do Brasil e possui 3281 descrições de atividades econômicas de empresas da cidade de Vitória - Espírito Santo e cerca de 85000 descrições de atividades econômicas de empresas da cidade de Belo Horizonte - Minas Gerais assim como os códigos de atividades econômicas manualmente classificados por analistas da Receita Federal. Note que mesmo a tarefa de classificação manual é extremamente complexa e suscetível a erro. Esta base de dados foi primeiramente tratada, este pré-processamento não será detalhado, mas inclui, dentre outros, a retirada de códigos inexistentes, códigos de atividade inseridos nas descrições dos textos e a correção ortográfica de palavras.

Lematização não foi aplicada à base de dados o que torna o vocabulário bem mais extenso. Posteriormente uma seleção de características global foi aplicada sobre a base de dados selecionando apenas os 5000 termos mais relevantes segundo o critério $max(X^2)$ (ver seção 2.5.4). Foram então ignorados todos os documentos que não possuíam nenhum termo após a fase de seleção de características. A base de dados resultante possui 87789 documentos e 1003 códigos de atividades econômicas.

4.2 Dificuldades do problema de classificação de atividades econômicas

Nesta subseção apresentamos uma caracterização da base de Dados CNAE demonstrando que a maioria dos problemas comuns aos problemas de HMC apresentados na seção 3.2.3 também são inerentes ao problema de classificação de atividades econômicas com base na tabela CNAE-Subclasses. Ao longo desta subseção apresentaremos as principais dificuldades do problema de classificação de atividades econômicas mostrando como estes problemas impactam os resultados das técnicas de classificação.

Na figura 4.1 mostramos o número de documentos (i.e., empresas) por classe da base de dados.

1. Dificuldade inerente da classificação

[3] Em nossas bases de dados utilizamos a versão 1.1 da tabela CNAE-subclasses.

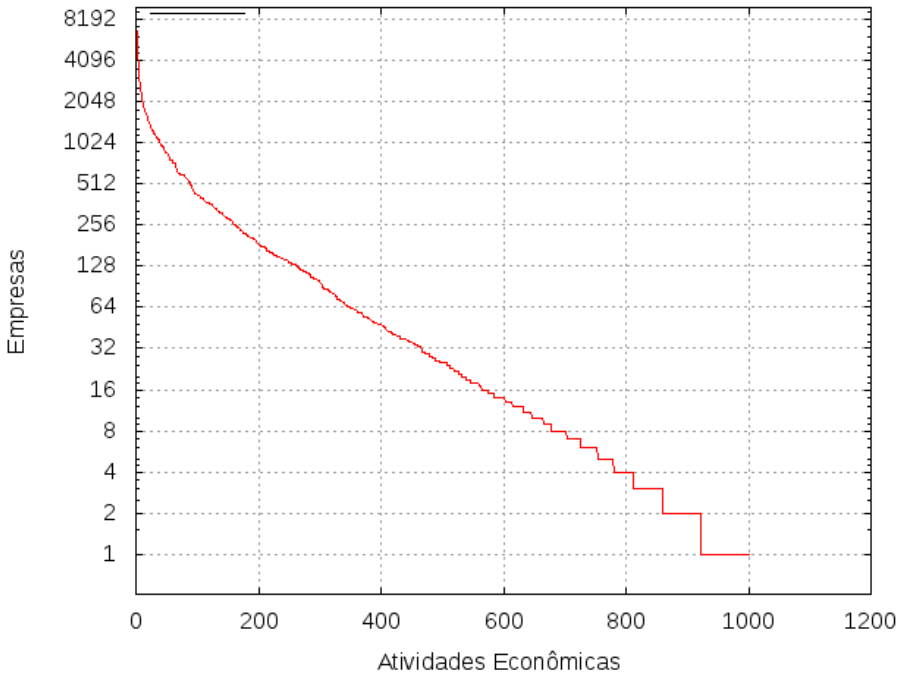


Figura 4.1: Número de empresas por classe

O problema da classificação de atividades econômicas com base na tabela CNAE-subclasses precisa classificar as empresas em cerca de 1200 códigos de atividades econômicas possíveis. Sendo que as empresas podem possuir múltiplos atividades econômicas e portanto múltiplos códigos de atividade econômica, fica portanto clara a dificuldade inerente da classificação.

2. Dificuldade de obtenção de bases de dados para treinamento

   Conforme citamos na seção 4.1 a produção de uma base de treinamento é por si só uma tarefa extremamente difícil. A base de dados CNAE foi tratada e limpa. O alto custo de criação de grandes bases de dados torna também difícil garantir que a base de dados está correta, assim como todos os assinalamentos são precisos. O compromisso entre criar bases de dados com quantidade de documentos suficientes para o treinamento e utilizar bases de dados bem validadas se tornam concorrentes. Na figura 4.1 vemos que cerca de 50% das classes do último nível ocorrem em menos que 15 documentos, o que nos indica que esta base de dados deveria ser maior para permitir que os algoritmos tenham informação suficiente para inferir modelos confiáveis. Não obstante, existem claros exemplos de assinalamentos inconsistentes ou incompletos, como por exemplo, uma empresa cuja descrição da atividade econômica é sucintamente "perfura-

ção de poços artesianos"está classificada com uma única atividade econômica "*i*nstalações hidráulicas, sanitárias e de gás" (4543801) mesmo existindo a atividade econômica "*p*erfuração e construção de poços de água" (4529205). Note que em nosso tratamento desta base de dados não alteramos estes assinalamentos uma vez que consideramos que devíamos nos basear nos assinalamentos dos especialistas e que assinalamentos incorretos como o do exemplo, são naturais à complexidade da tarefa.

Na figura 4.1 vemos o desbalanceamento existente no último nível da base de dados da CNAE. Enquanto temos uma única classe que ocorre 8552 (condomínios) temos 357 classe que ocorrem em apenas 10 documentos ou menos. Claramente o nível de informação entre estas classes varia sensivelmente.

3. Suscetibilidade a ruídos, superajuste

   Como mencionamos anteriormente, existe grande dificuldade inerente à classificação dos problemas de HMC. Com isso, a qualidade das bases de dados fica comprometida. Neste contexto, torna-se necessários que os modelos aprendidos desta base de dados sejam robustos de forma a não sofrer do superajuste(2.7.1).

   Uma dificuldade em especial ocorre porque o número de documentos de cada classe pode ser pequeno e assim o modelo facilmente acabará sendo induzido por ruídos, por exemplo, a atividade econômica "*p*erfuração e construção de poços de água" foi assinalada a apenas 25 empresas portanto o compromisso entre aprender um modelo de uma base de dados que possui poucos exemplos e não sofrer de superajuste podem ser antagônicos.

4. Alta especialização dos conceitos

   Conforme mostramos no exemplo a seguir à medida em que se aprofunda na hierarquia, as atividades econômicas passam a ser cada vez mais similares tornando a diferença semântica entre as classes mais tênue o que pode levar inclusive especialistas do domínio a incorrer em erros. Quanto mais se aprofunda na hierarquia, mais especializados os conceitos (atividades econômicas) e mais difícil discernir tais conceitos o que dificulta tanto a classificação manual quanto automática.

5. Desbalanceamento de classes e tendenciosidade

   A base de dados hierárquica em geral é extremamente complexa. Enquanto existe quantidade abundante de documentos para uma classe, outras classes podem possuir uma quantidade muito pequena de exemplos.

O desbalanceamento das classes é outro desafio na classificação hierárquica. Inclusive este desbalanceamento tende a se atenuar à medida que se aprofunda na hierarquia. Na figura 4.1 vemos que 8552 do total de 87789 empresas são apenas condomínios e portanto possuem uma única atividade econômica. Ao mesmo tempo, 428 atividades econômicas de um total de 1002 são assinaladas a 15 ou menos empresas. Vemos o comportamento de cauda longa no número de ocorrência de cada classe do último nível da hierarquia.

6. Quantidade insuficiente de dados e raridade de classes

   Como mencionado acima o desbalanceamento faz com que mesmo em posse de uma base de dados relativamente grande e custosa de ser produzida manualmente ainda resulta-se em uma base de dados em que a quantidade de classes raras com informação insuficiente para a construção de modelos de classificação confiáveis é alta. Mesmo com uma base de dados com 87789 empresas temos que quase metade das classes ocorre em 15 ou menos documentos. Note que na literatura considera-se que na ocorrência de tão poucos exemplos de classes, deve-se utilizar algoritmos especiais que consigam aprender Toutanova et al. [2001]; Liu et al. [2005]; Manning et al. [2008] com um número tão pequeno de exemplos.

7. Alta demanda de poder computacional

   A base de dados CNAE possui 1002 atividades econômicas, a hierarquia resultante possui 1784 classes. Dependendo do algoritmo aplicado o número de modelos gerados pode ser ainda maior. Por esta razão, vários algoritmos e abordagens existentes não escalam para o problema em questão e portanto não podem ser utilizadas. Por exemplo, o número de modelos a serem criados pelos algoritmos que se baseiam na transformação em conjunto potência (3.1.1) é muito grande se tornando proibitiva esta abordagem. Algoritmos para a classificação automática da base de dados CNAE precisam escalar tanto em relação ao número de documentos, ao número de classes e ao número de termos do vocabulário.

Experimentos preliminares foram executados na base de dados CNAE. Para estes experimentos preliminares, foram selecionados aleatoriamente 10% da base de dados para compor uma base de teste e os demais documentos foram selecionados para treino. As técnicas iniciais executadas não escalaram para o problema [Veloso et al., 2007]. Na figura 4.2 apresentamos resultados preliminares em que pode-se ver uma tendência clara de correlação entre a quantidade de documentos disponíveis na base de dados de treino e o número de assinalamentos corretos para a classe em questão.

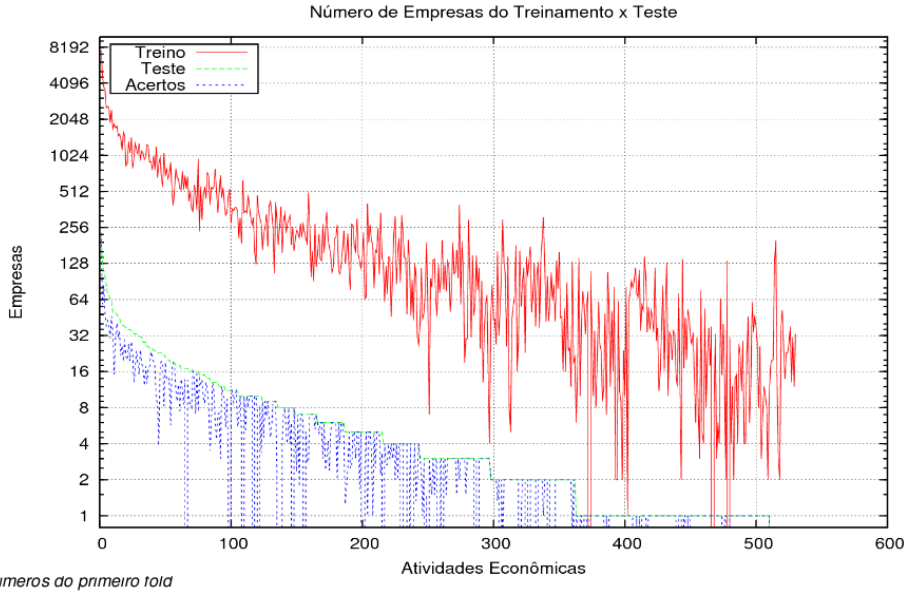


Figura 4.2: Número de empresas na base de treino, teste e número de assinalamentos corretos.

Capítulo 5

Metodologia

Neste capítulo apresentamos uma nova técnica para o problema de HMC. Esta técnica se baseia na abordagem de classificação associativa postergada, apresentaremos as bases de dados utilizadas para a experimentação, apresentaremos as medidas de qualidade da classificação a serem utilizadas e os algoritmos baseline utilizados para a comparação.

5.1 O Arcabouço Massifica

Para a execução e análise dos algoritmos foi desenvolvido um arcabouço, o Massifica, para aplicação das técnicas de classificação hierárquica. Este arcabouço permite que se faça a integração de diferentes algoritmos de aprendizagem e se executem tais algoritmos de acordo com diferentes arquiteturas de classificação.

A nossa proposta de algoritmo (seção 5.2) será comparada com duas técnicas de classificação bastante eficientes, o Adaboost e o SVM. As implementações utilizadas como referência foram a biblioteca ICSI Boosting, uma versão *open source* que utiliza classificadores fracos baseados em árvore de decisão de único nível (*"decision stumps"*) [1] e SVM-Perf [Joachims, 2006], um pacote que implementa uma versão eficiente de SVM e que pode ser treinado em tempo linear. As três técnicas estão integradas ao arcabouço Massifica.

5.1.1 Arquiteturas de classificação

O arcabouço Massifica foi implementado com diferentes arquiteturas de navegação na estrutura de árvore hierárquica.

- Plana Todas as Categorias

[1] http://code.google.com/p/icsiboost/

Nesta arquitetura é realizada uma única classificação plana com todos os nodos da hierarquia.

- Arquitetura Plana Categorias Folhas

  Nesta arquitetura é realizada uma única classificação em que o conjunto C', as classes utilizadas durante a aprendizagem plana, é definido como todos os nodos folhas da hierarquia $\mathcal{H}$. Note que esta arquitetura é especialmente válida para os problemas de HMC em que a hierarquia é virtual.

- Plana nível a nível

  Na arquitetura plana nível a nível, são induzidos n modelos onde n representa a profundidade a árvore hierárquica. Assim cada modelo C'_i é constituído de todos os nodos presentes no nível i da árvore hierárquica.

- TOP-DOWN

  Na arquitetura Top-Down é realizado o procedimento Top-Down (Pachinko Allocation Machine) como descrito na seção 3.3.2.2.

Realizaremos a classificação da base de dados CNAE com os três diferentes algoritmos apresentados utilizando as arquiteturas "top-down" e *plana categorias-folhas* (arquitetura apropriada para hierarquias virtuais).

Uma vez que os métodos baseados na arquitetura *plana-categorias-folhas* não satisfazem o requisito de consistência hierárquica (def. 8), implementamos o método de pós-processamento baseado em consistência hierárquica (ver seção 3.3.3) para satisfazer tal requisito e tornar as técnicas assim comparáveis segundo as medidas apresentadas na seção 3.2.2.2.

Posteriormente, dividimos a base de dados em 10 partes iguais de forma aleatória para se realizar a validação cruzada (*"ten-fold cross validation"*) (ver seção 2.4.1). Os parâmetros dos algoritmos SVM e Adaboost foram selecionados por tentativa e erro manualmente. Já o parâmetro confiança da técnicas MLAC foi selecionado da seguinte forma. Redividiu-se a base de dados de treinamento em 10 partes iguais de forma aleatória, utilizando uma destas partes como validação. Note que este processo foi implementado no arcabouço Massifica e como a base de dados de validação é selecionado com base uma amostra aleatória, diferentes parâmetros podem resultar de diferentes execuções do MLAC, isso acontece em particular nas classes mais raras pois o modelo é muito mais sensível.

5.2 Classificação Associativa Postergada para HMC

Nesta seção apresentamos uma nova técnica de classificação associativa postergada, os problemas inerentes à aplicação destas técnicas aos problemas de HMC e as soluções implementadas em nossa nova técnica.

As técnicas de classificação associativa postergada constroem o conceito apenas quando se apresenta o documento de teste. Assim, deve-se construir um conceito (i. e., as regras de classificação) para cada documento de teste. Por esta razão tais algoritmos demandam bastante carga computacional. Diante deste fato, a escalabilidade se apresenta como um dos grandes desafios para a viabilidade de tais técnicas. A solução proposta nestas técnicas se baseia na criação de uma cache das regras geradas que podem ser compartilhadas entre os diferentes conceitos a serem construídos. Esta cache minimiza o problema uma vez que a maior parte da carga de trabalho de tais técnicas está na construção destas regras.

Em nossos primeiros trabalhos com esta técnica, concluímos que em bases de dados com um número muito pequeno de classes, a utilização de uma cache de regras consegue contornar quase que praticamente o problema, porém, ao aplicarmos tais técnicas em bases de dados com um número maior de classes o "cacheamento" se torna muito mais complicado necessitando uma refinação da solução do problema.

A primeira técnica que experimentamos foi a CLAC, uma técnica associativa postergada que propõe a projeção progressiva baseada em rótulos (*progressive label focusing*)[Veloso et al., 2007]. Toda vez que um rótulo é selecionado para um documento de teste, uma nova projeção precisa ser gerada e novas regras construídas dada a nova projeção em questão. Essas reprojeções sucessivas se mostraram inviáveis na aplicação desta técnica em bases de dados com um número maior de classes.

A inviabilidade se apresenta por razões diversas:

- Construção de múltiplos conceitos para cada documento de teste

  Esta implementação demanda que sejam realizadas sucessivas reprojeções das bases de dados, após cada reprojeção, é realizada a construção do conceito da nova base de dados projetada,

- Complicação do mecanismo de cache

  A necessidade de que se permita que um rótulo previsto seja inserido como antecedente de uma regra de classificação complica bastante o mecanismo de cache implementado nestes algoritmos.

- Alta demanda de carga computacional

A construção dos diferentes conceitos e a necessidade de manutenção de uma cache de regras é bastante dificultada nos problemas de HMC já que o número de classes é muito grande o que demanda caches bem maiores que o usual.

- Esparsidade dos dados

  Como mencionado na seção 3.2.3, as bases de dados HMC possuem classes extremamente raras. Ao realizar as projeções sucessivas a tendência é que tais classes tenham um número ainda menos de documentos representantes. Sendo assim, o mecanismo de projeções sucessivas para a exploração da dependência entre as classes só consegue ser efetivo quando existe quantidade abundante de informação. Uma seleção da base de dados em cima de uma base de dados já esparsa complica ainda mais o problema.

Posteriormente, trabalhamos com a aplicação de uma variante do algoritmo ILAC [Veloso et al., 2007]. Novamente, alguns problemas surgiram para a aplicação de tais técnicas aos problemas de HMC. As técnicas de classificação associativa costumam ter dois principais parâmetros para considerar uma regra válida, a confiança e o suporte. Uma das maiores dificuldades na aplicação destas técnicas é a escolha destes parâmetros. Na aplicação destas técnicas em problemas de HMC diferentes parâmetros podem ser necessários para a aplicação em cada nodo da árvore (quando da utilização da abordagem Top-down), além disso, na existência de um número muito grande de classes e de um desbalanceamento acentuado entre as classes, pode-se fazer necessária a utilização de vários limiares de assinalamento (um para cada classe). No caso da classificação associativa, a confiança das regras, em sua tese de doutorado, Kiritchenko encontrou problemas similares Kiritchenko [2005] e propôs então a utilização de diferentes limiares de assinalamento para cada classe.

5.2.1 Seleção de parâmetros

Uma outra importante limitação da aplicação das técnicas de classificação associativa, é que os parâmetros a serem utilizados podem alterar significativamente o comportamento do algoritmo. Assim, o desempenhos destes algoritmo é extremamente sensível a uma boa seleção de parâmetros, em particular, os principais parâmetros utilizados, confiança e suporte podem ser alterados para inclusive alterar o compromisso viés-variância do algoritmo. Por exemplo, caso o algoritmo seja executado com um valor de suporte muito alto, o algoritmo tende a ter alto viés e baixa variância quando comparado com o mesmo algoritmo executado com um valor de suporte baixo.

Assim, estas técnicas demandam que um especialista faça uma triagem na base de dados, alguns experimentos iniciais, e então escolha os parâmetros a serem utilizados. Ou seja, a fase de validação dos parâmetros é uma fase crítica e sensível e que demanda um conhecimento especialista tanto da base de dados quanto das técnicas de classificação associativa.

Nos problemas de HMC, a utilização de um especialista é proibitiva já que o número de parâmetros existentes pode ser muito grande, por exemplo, conforme mostramos nas seções anteriores, pode ser necessária a realização de uma fase de validação e diferentes parâmetros para diferentes níveis da hierarquia e por exemplo, um determinado valor de parâmetro pode ser necessário para cada nodo da hierarquia. Sendo assim, demanda-se o desenvolvimento de técnicas automáticas de escolha dos parâmetros, i.e., uma fase de validação automática. Para contornar este problema, desenvolvemos um método *"Hill Climbing"* para a escolha automática do parâmetro confiança semelhante ao aplicado por Coenen & Leng [2007].

Em geral, para o parâmetro suporte, ao invés de um valor relativo é utilizado um valor absoluto, isto é, o número de documentos de treino em que a regra é válida. Esta abordagem é adequada em problemas em que não existe um grande desbalanceamento entre as classes. Porém, como mostrado anteriormente na seção 3.2.3, os problemas de HMC em geral apresentam grande desbalanceamento entre as classes, neste caso a utilização de um suporte relativo em relação ao tamanho das classes se mostrou mais adequado. Isto é, o número de documentos necessários para que a regra seja válida depende do número de documentos de treino que foram assinalados à classe em questão.

Na prática, a utilização de um suporte relativo ao invés de um suporte absoluto faz com que o algoritmo seja mais flexível já que permite realizar predições com menor indício de classes raras e aumenta a necessidade de indícios para as classes mais frequentes. Podemos entender este processo como uma flexibilização da decisão de assinalamento de acordo com a frequência da classe, fazendo com que as predições de classes frequentes tenham maior viés e as predições de classes mais raras tenham maior variância.

Este algoritmo resultante das diversas adaptações necessárias para o problema de HMC, será chamado de M-LAC.

5.2.2 Funcionalidades extras implementadas

Além das diferenças já citadas na técnica em questão foram implementadas importantes funcionalidades apresentadas a seguir:

- Leitura de arquivo compactado

Muitas das bases de dados são extremamente grandes. Pode ser útil então manipular os arquivos de entrada das bases de dados (treino, validação e teste) de forma compactada. Por isso, a técnica foi implementada permitindo a leitura no formato de arquivos gzip[2].

- Implementação da cache em arquivos mapeados em memória

  A utilização de técnicas de mapeamento de memória facilitam a implementação e melhoram o desempenho das aplicações uma vez que a memória é mapeada diretamente em disco e fica a cargo do sistema operacional a gerência da memória destes dados para a memória principal. Para mais detalhes sobre a vantagem do mapeamento de arquivos em memória o leitor pode buscar em Jr et al. [1987].

- Remoção de Stopwords

  A remoção de stopwords foi implementada diretamente no algoritmo. Isto pode facilitar o processo por parte do usuário, uma vez que este pode deixar a cargo do algoritmo a remoção das stopwords.

- Seleção de Características

  Os principais critérios utilizados na literatura: ganho de informação, Odds-Ratio e X^2 foram implementados [3].

A nossa técnica de classificação associativa é sensível a ruídos, uma forma de enfrentar este problema é com técnicas de seleção de características e remoção de stopwords conforme mencionado na seção 3.2.3.

5.3 Medidas de qualidade da classificação

Em nosso trabalho focamos as medidas de qualidade propostas por Kiritchenko, hP, hR e hF1 (seção 3.2.2), pois são simples e focam diretamente na hierarquia desenvolvida por especialistas para a análise das diferentes técnicas e atendem aos requisitos da seção 3.2.2.1. Ressalta-se que muitas vezes vários artigos apresentam uma série de métricas diferentes e não conseguem resumir de forma consistente o desempenho das técnicas diante de uma quantidade mais extensa de medidas.

Apresentaremos as Micro e Macro médias das três métricas hP, hR e HF1. Para a análise dos diferentes desempenhos das métricas nos diferentes níveis da hierarquia apresentaremos tais médias considerando apenas os nodos de um nível da hierarquia

[2] http://www.gzip.org

[3] O estudo das diferentes técnicas de seleção de características não está no escopo deste trabalho.

e no geral. As diferenciações das métricas Macro e Micro permite-nos diferenciar o desempenho em classes raras e frequentes e os resultados nos diferentes níveis esclarece como o desempenho se degrada à medida que se aprofunda na hierarquia, seja pela propagação de erros dos modelos "top-down", seja pela alta especialização dos conceitos (conforme discutido na seção 3.2.3).

Capítulo 6

Resultados Experimentais

Neste capítulo apresentamos os resultados experimentais da execução de diferentes algoritmos que se diferenciam pelo algoritmo-base utilizado, o algoritmo M-LAC, Adaboost e SVM e pela arquitetura de classificação, Plana ou Top-Down. Posteriormente mostraremos com maior profundidade os resultados obtidos na execução do Adaboost Top-Down.

As medidas de qualidade apresentadas são a Acurácia, Micro média F1 ($\mu F1$) e Macro Média F1 ($MF1$). Estas medidas são apresentadas como propostas por Kiritchenko na coluna G da tabela 6.3. Além disto, apresentamos tais medidas calculando as médias nível a nível, isto é, na coluna do nível i, são calculadas as médias de todos os nodos da hierarquia que pertencem ao nível i. Estas medidas adicionais nível por nível permitem que se avalie os diferentes desempenhos das técnicas à medida que se aprofunda na hierarquia.

Para a execução destes algoritmos, a primeira dificuldade é a seleção de parâmetros. Na seção 5.3 mostramos que uma boa medida da qualidade da classificação precisa apresentar um valor maior possível da métrica Macro Média F1 ($MF1$) indicando que a técnica consegue assinalar com qualidade classes raras. Em particular, focaremos principalmente na maximização da medida Macro Média F1 ($MF1$) tanto geral, como proposto por Kiritchenko, quanto no último nível da hierarquia, i.e. nível 5.

O algoritmo M-LAC que propomos não demanda que façamos a seleção manual de um limiar já que conforme explicado na seção 5.2.1 é utilizado um método "hill climbing" para a decisão dos parâmetros.

O algoritmo SVM foi executado com diferentes compromissos entre erro de treinamento e margem em uma base de dados separada.

Nas tabelas 6.1 e 6.2 apresentamos os resultados das métricas para diferentes

limiares do algoritmo SVM e do algoritmo Adaboost tanto na arquitetura Plana quanto Top-Down. Os limiares escolhidos como a melhor execução para a comparação entre os algoritmos foram enaltecidos em negrito.

Arquitetura Plana						
Limiar	Geral			Nível 5		
	Acc (%)	μF1 (%)	MF1 (%)	Acc (%)	μF1 (%)	MF1 (%)
-0,85	79,2	53,2	21,3	69,3	40,6	18,6
-0,90	73,4	57,5	**22,0**	62,6	44,9	**19,3**
-0,95	63,1	60,3	21,9	52,0	48,3	19,2
Arquitetura Top-Down						
-0,50	82,3	61,7	26,2	71,7	47,8	22,8
-0,70	73,9	64,3	31,0	60,3	50,9	27,3
-0,80	65,2	63,9	**32,1**	49,1	50,2	**28,4**
-0,90	49,3	60,3	27,8	30,9	44,2	23,3

[a] Acc=Acurácia, μF1=Micro Average F1, MF1=Macro Average F1
[b] G=Geral

Tabela 6.1: Seleção de parâmetros do algoritmo SVM

Arquitetura Plana						
Afinidade	Geral			Nível 5		
	Acc (%)	μF1 (%)	MF1 (%)	Acc (%)	μF1 (%)	MF1 (%)
-0,001	71,9	60,6	24,6	59,3	50,9	21,2
-0,002	65,8	63,2	26,4	52,3	52,6	22,8
-0,003	58,1	63,4	**26,7**	44,3	52,2	**23,3**
-0,004	49,3	61,7	26,0	36,1	49,6	22,6
Arquitetura Top-Down						
-0,001	70,1	68,6	34,6	59,0	58,3	31,7
-0,002	58,0	67,0	**35,2**	45,7	56,5	**32,1**
-0,003	43,7	61,6	32,8	31,6	50,2	29,7

[a] Acc=Acurácia, μF1=Micro Average F1, MF1=Macro Average F1
[b] G=Geral

Tabela 6.2: Seleção de parâmetros do algoritmo Adaboost

Após a seleção dos parâmetros para todos os algoritmos e arquiteturas, cada algoritmo foi executado utilizando-se a validação cruzada de 10 partes (*ten-fold cross validation*). Os resultados obtidos estão resumidos nas tabelas 6.3 e 6.4 a apresentam

as métricas com a confiança de 95%. Primeiramente analisamos os resultados obtidos com a arquitetura Top-Down. Ao comparar os três algoritmos Top-Down vemos claramente que, segundo as métrica Micro F1 ($\mu F1$) e Macro F1 ($MF1$), nos primeiros níveis da hierarquia os algoritmos Adaboost e SVM são extremamente competitivos e superiores ao algoritmo M-LAC. Fica claro que a técnica M-LAC não foi desenvolvida para sistemas Top-Down o que a torna não competitiva na classificação principalmente nos primeiros níveis da hierarquia. Além disso, vemos que apesar de ser competitivo nos primeiros níveis da hierarquia, o SVM Top-Down degrada seu desempenho à medida que se aprofunda a hierarquia e com isso apresenta um desempenho (31,8%) quanto à métrica $MF1$ cerca de 10% inferior à técnica Adaboost Top-Down (34,9%).

Tais resultados indicam que o SVM apresenta melhores resultados quando a informação é abundante durante o treino e se a quantidade de informação passa a ser mais escassa o desempenho do sistema diminui sensivelmente. Além disso, não se pode ignorar a propagação de erros nos sistemas Top-Down, a técnica Adaboost Top-Down apresentou revocação maior que o SVM Top-Down desde os primeiros níveis da hierarquia. Como no sistema Top-Down a decisão de não assinalamento em um nível impede que os documentos sejam avaliados no próximo nível, A decisão de não assinalamento nos primeiros níveis da técnica SVM Top-Down fizeram com que muitos documentos não chegassem aos níveis mais profundos da hierarquia o que influiu na baixa revocação deste algoritmo nestes níveis. Assim, apesar de apresentar uma precisão bem próxima da técnica Adaboost Top-Down. Com isso, podemos concluir que em um sistema Top-Down a melhor configuração que apresente o melhor μF1 ou MF1 em um determinado nível pode não ser o mais adequado pois esta configuração pode apresentar uma propagação de erros maior e penalizar o desempenho nos níveis mais profundos da hierarquia. Estas considerações são semelhantes às apresentadas por Sun [2004], em que se ressalta que os sistemas Top-Down sofrem do problema de propagação de erros, uma vez que os erros cometidos pelos classificadores nos níveis mais altos da hierarquia não são recuperáveis a menos que se incorpore mecanismos de recuperação de erros. Uma das técnicas utilizadas é a redução dos limiares, esta redução em geral diminui a precisão e aumenta a revocação dos sistemas de classificação, note que nestas bases de dados o aumento da revocação acaba refletindo na maior quantidade de documentos que atingem os níveis mais profundos da hierarquia.

Ao compararmos a técnica M-LAC Plana e Top-Down vemos um resultado interessante, a precisão do sistema com a arquitetura plana é sistematicamente maior que a precisão na arquitetura Top-Down. Isto quer dizer que o método de assinalamento implícito que garante a consistência hierárquica apresentou uma precisão maior que o método de assinalamento explicitamente pelo algoritmo M-LAC. Portanto, o algoritmo

Arquitetura	Nível	M-LAC Plana			SVM Plana			Adaboost Plana		
		MP (%)	MR (%)	MF1 (%)	MP (%)	MR (%)	MF1 (%)	MP (%)	MR (%)	MF1 (%)
Plana	G	$34,3 \pm 0,6$	$42,0 \pm 0,7$	$34,0 \pm 0,5$	$25,3 \pm 0,3$	$21,3 \pm 0,3$	$21,7 \pm 0,3$	$24,6 \pm 0,4$	$29,6 \pm 0,4$	$25,9 \pm 0,3$
	1	$53,6 \pm 1,4$	69 ± 3	$56,3 \pm 1,4$	60 ± 2	$48,2 \pm 1,4$	$52,5 \pm 1,6$	58 ± 2	$55,8 \pm 2,1$	57 ± 2
	2	$38,4 \pm 0,8$	$53,2 \pm 1,5$	$41,1 \pm 0,8$	$39,1 \pm 1,0$	$31,7 \pm 0,7$	$34,2 \pm 0,8$	$39,5 \pm 1,0$	$40,4 \pm 1,3$	$39,5 \pm 1,1$
	3	$34,7 \pm 1,0$	$45,2 \pm 1,2$	$35,4 \pm 0,9$	$31,4 \pm 0,5$	$24,9 \pm 0,4$	$26,3 \pm 0,4$	$30,0 \pm 0,8$	$33,7 \pm 0,8$	$31,0 \pm 0,7$
	4	$33,9 \pm 0,8$	$41,6 \pm 1,0$	$33,8 \pm 0,7$	$25,1 \pm 0,3$	$20,9 \pm 0,3$	$21,4 \pm 0,3$	$25,0 \pm 0,4$	$30,2 \pm 0,3$	$26,4 \pm 0,3$
	5	$33,7 \pm 0,6$	$40,0 \pm 0,4$	$32,7 \pm 0,4$	$22,2 \pm 0,3$	$19,3 \pm 0,3$	$19,1 \pm 0,3$	$21,2 \pm 0,4$	$26,9 \pm 0,4$	$22,7 \pm 0,3$
Top-Down	G	$23,5 \pm 0,4$	$26,0 \pm 0,8$	$21,4 \pm 0,3$	$35,0 \pm 0,7$	$33,6 \pm 0,5$	$31,8 \pm 0,5$	$34,8 \pm 0,4$	$38,2 \pm 0,5$	$34,9 \pm 0,4$
	1	63 ± 2	54 ± 2	55 ± 2	$72,7 \pm 1,5$	$61,4 \pm 1,3$	$65,2 \pm 1,3$	$66,1 \pm 1,8$	63 ± 2	64 ± 2
	2	$45,5 \pm 1,9$	$36,7 \pm 1,2$	$36,4 \pm 1,0$	$60,1 \pm 1,7$	$46,7 \pm 0,9$	$50,1 \pm 1,0$	$50,7 \pm 1,2$	$50,4 \pm 1,0$	$49,5 \pm 1,0$
	3	$28,7 \pm 0,7$	$28,6 \pm 0,8$	$25,2 \pm 0,3$	$42,8 \pm 0,9$	$37,3 \pm 0,7$	$37,5 \pm 0,6$	$39,3 \pm 0,7$	$41,7 \pm 0,8$	$39,1 \pm 0,5$
	4	$22,8 \pm 0,5$	$25,1 \pm 0,9$	$20,8 \pm 0,4$	$35,3 \pm 0,9$	$33,7 \pm 0,5$	$32,1 \pm 0,6$	$35,0 \pm 0,5$	$38,3 \pm 0,7$	$35,0 \pm 0,5$
	5	$20,2 \pm 0,4$	$24,5 \pm 0,8$	$18,9 \pm 0,3$	$30,3 \pm 0,7$	$31,1 \pm 0,5$	$28,3 \pm 0,5$	$31,8 \pm 0,3$	$35,9 \pm 0,5$	$32,1 \pm 0,4$

[a] MP=Média macro da Precisão, MR= Média macro da Revocação, MF1=Média macro de F1

Tabela 6.3: Resultado das técnicas na base de dados CNAE

Arquitetura	Nível	M-LAC Plana		SVM Plana		Adaboost Plana	
		μF1 (%)	MF1 (%)	μF1 (%)	MF1 (%)	μF1 (%)	MF1 (%)
Plana	G	$66,5 \pm 0,1$	$34,0 \pm 0,5$	$57,5 \pm 0,3$	$21,7 \pm 0,3$	$63,2 \pm 0,1$	$25,9 \pm 0,3$
	1	$79,8 \pm 0,2$	$56,3 \pm 1,4$	$74,4 \pm 0,2$	$52,5 \pm 1,6$	$78,1 \pm 0,2$	57 ± 2
	2	$73,8 \pm 0,2$	$41,1 \pm 0,8$	$67,4 \pm 0,3$	$34,2 \pm 0,8$	$71,7 \pm 0,2$	$39,5 \pm 1,1$
	3	$67,0 \pm 0,2$	$35,4 \pm 0,9$	$58,5 \pm 0,3$	$26,3 \pm 0,4$	$64,2 \pm 0,2$	$31,0 \pm 0,7$
	4	$61,7 \pm 0,2$	$33,8 \pm 0,7$	$50,7 \pm 0,4$	$21,4 \pm 0,3$	$57,3 \pm 0,1$	$26,4 \pm 0,3$
	5	$56,6 \pm 0,1$	$32,7 \pm 0,4$	$44,8 \pm 0,4$	$19,1 \pm 0,3$	$52,1 \pm 0,1$	$22,7 \pm 0,3$
Top-Down	G	$52,9 \pm 0,1$	$21,4 \pm 0,3$	$63,9 \pm 0,2$	$31,8 \pm 0,5$	$67,4 \pm 0,2$	$34,9 \pm 0,4$
	1	$69,3 \pm 0,2$	55 ± 2	$82,5 \pm 0,2$	$65,2 \pm 1,3$	$82,2 \pm 0,2$	64 ± 2
	2	$61,3 \pm 0,3$	$36,4 \pm 1,0$	$75,5 \pm 0,2$	$50,1 \pm 1,0$	$75,7 \pm 0,2$	$49,5 \pm 1,0$
	3	$51,3 \pm 0,2$	$25,2 \pm 0,3$	$64,4 \pm 0,2$	$37,5 \pm 0,6$	$67,6 \pm 0,2$	$39,1 \pm 0,5$
	4	$46,9 \pm 0,3$	$20,8 \pm 0,4$	$56,4 \pm 0,2$	$32,1 \pm 0,6$	$61,3 \pm 0,2$	$35,0 \pm 0,5$
	5	$43,2 \pm 0,2$	$18,9 \pm 0,3$	$50,0 \pm 0,2$	$28,3 \pm 0,5$	$57,1 \pm 0,2$	$32,1 \pm 0,4$

[a] μF1=Média Micro de F_1, MF1=Média macro de F_1

Tabela 6.4: Medidas F_1 das técnicas na base de dados CNAE

M-LAC através do assinalamento implícito explorou a hierarquia para uma melhora significativa de seu desempenho nos primeiros níveis da hierarquia. Fica claro que o sistema Top-Down M-LAC sofre da propagação de erros já que os documentos não assinalados no primeiro nível não foram considerados nos níveis subsequentes.

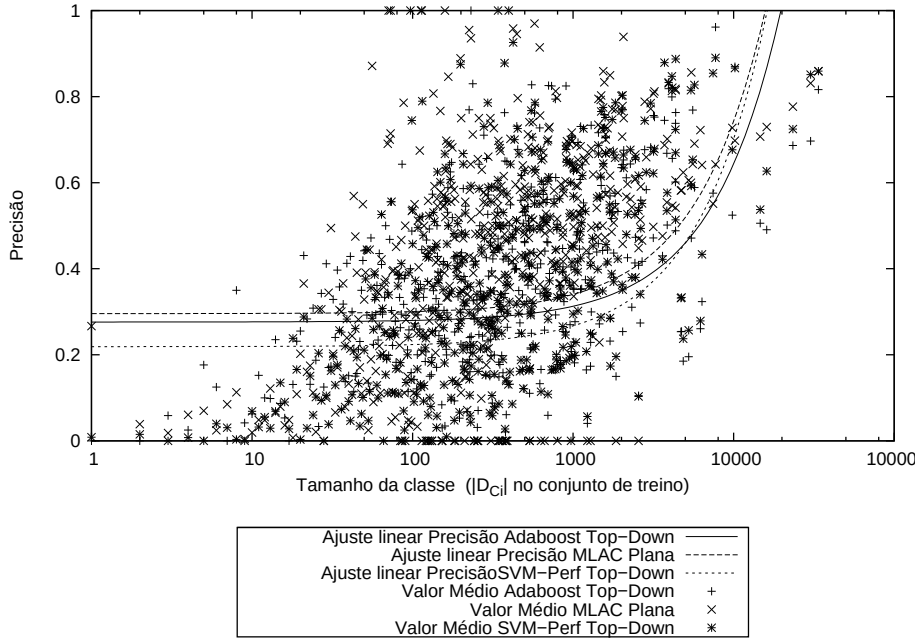
Na tabela 6.4, podemos ver o desempenho dos algoritmos segundo as métricas $\mu F1$ e $MF1$, percebemos que os algoritmos SVM e Adaboost na arquitetura plana apresentam resultados piores que na arquitetura Top-Down principalmente nas classes raras já que a diferença do desempenho segundo $MF1$ é bem maior que a diferença de desempenho segundo a métrica $\mu F1$.

As duas técnicas que prevaleceram sobre as demais foram a técnica M-LAC Plana e a técnica Adaboost Top-Down, apresentando respectivamente a $\mu F1_G$ de 34,0% e 34,9%, esta diferença não é significativa já que os intervalos de confiança 95% se sobrepõem. Podemos ver na tabela 6.4 que a técnica Adaboost Top-Down é superior à técnica M-LAC Plana segundo a métrica $\mu F1$ porém que segundo a métrica $MF1$ esta superioridade não se sustenta, fica claro portanto que o desempenho do algoritmo Adaboost Top-Down é superior nas classes mais frequentes e nos primeiros níveis da hierarquia, porém inferior nas classes raras e nos últimos níveis da hierarquia. Em particular, a técnica M-LAC Plana obteve a $MF1_5$ de 32,7% enquanto a técnica Adaboost Top-Down obteve 32,0%.

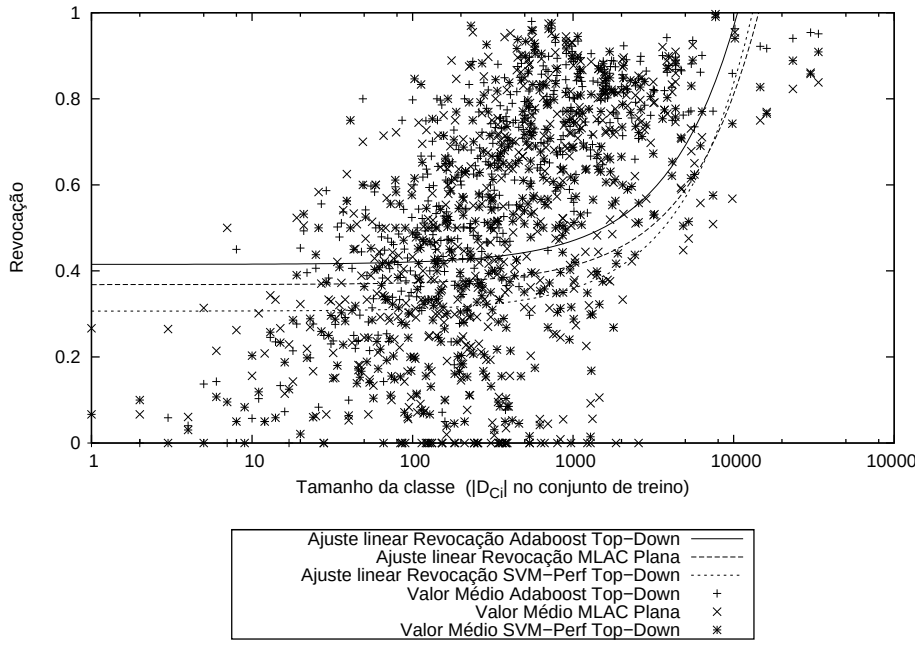
Nas figuras 6.1 e 6.2 comparamos as três principais técnicas dos resultados anteriores: SVM Top-Down, Adaboost Top-Down e M-LAC plana enaltecendo os valores para diferentes tamanhos de classes. Vemos pelo ajuste linear que o desempenho dos algoritmos SVM decaem sensivelmente à medida em que o número de documentos das classes caem. Além disso, a Medida F apresenta resultados muito semelhantes nas classes mais frequentes, ou seja, quando há bastante informação de treino, as técnicas divergem muito pouco quanto a seu desempenho, sendo assim a diferenciação das técnicas é muito mais importante nas classes menores. Apesar do ajuste linear indicar um comportamento pode-se ver que os pontos estão bastante dispersos isto mostra a complexidade da análise do desempenho dos sistemas de classificação hierárquica.

Como pode se ver a acurácia dos sistema decai muito nos níveis mais profundos da hierarquia. Esse fenômeno já foi explicitados por outros trabalhos na literatura [Granitzer, 2003],pois os sistemas Top-Down se degradam por causa da propagação de erros. Esse efeito é ainda maior pois a classificação já é naturalmente dificultada à medida em que se aprofunda na hierarquia pois as classes representam conceitos mais similares, o problema do desbalanceamento se agrava assim como existe menor quantidade de informação por classe.

Na figura 6.4 vemos que o desempenho dos sistemas de classificação possui uma



(a) Precisão



(b) Revocação

Figura 6.1: Precisão e Revocação por número de documentos de treino

correlação altíssima com a quantidade de informação existente com a classe em questão (neste caso o número de documentos de treino). Podemos ver que para as classes que possuíam o número de documentos de treino superior a 500, o desempenho do sistema de classificação foi muito superior e à medida que o número de documentos

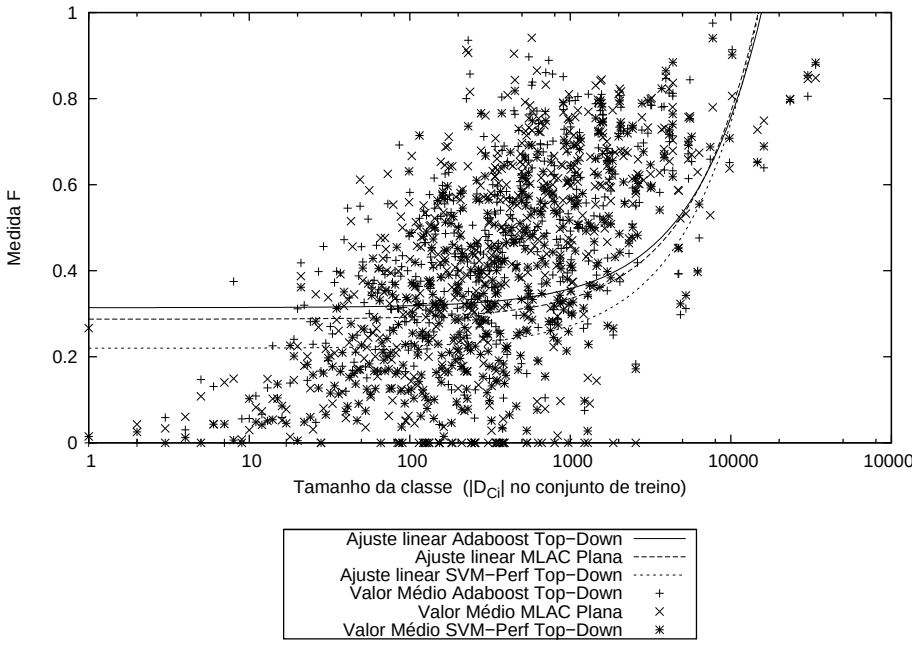
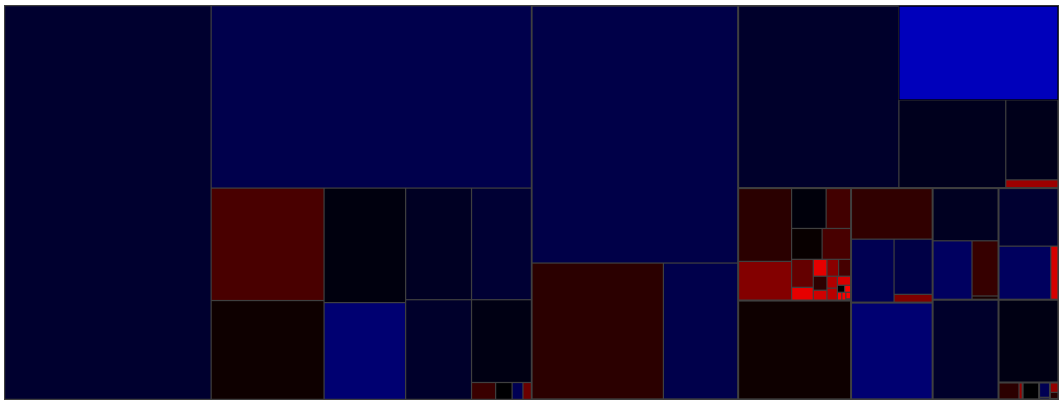
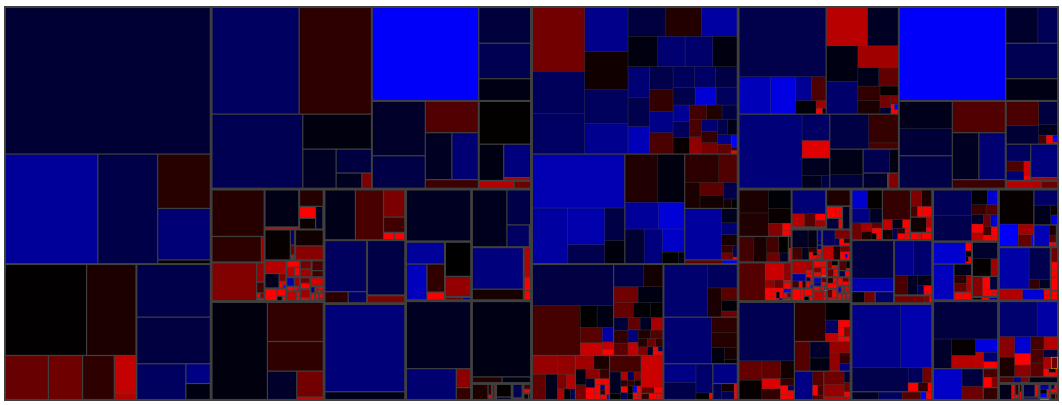


Figura 6.2: Medida F_1 por número de documentos de treino

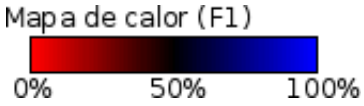
de treino diminui consequentemente o desempenho também diminui drasticamente. Vemos portanto que não basta analisarmos a profundidade da classe na árvore como também a quantidade de informação de treino existente para a classe em questão.



(a) 1^o nível da base CNAE usando Adaboost (b) 2^o nível da base CNAE usando Adaboost



(c) 3^o nível da base CNAE usando Adaboost (d) 4^o nível da base CNAE usando Adaboost



(e) Mapa de Calor

Figura 6.3: Sistema de classificação Adaboost Top-Down

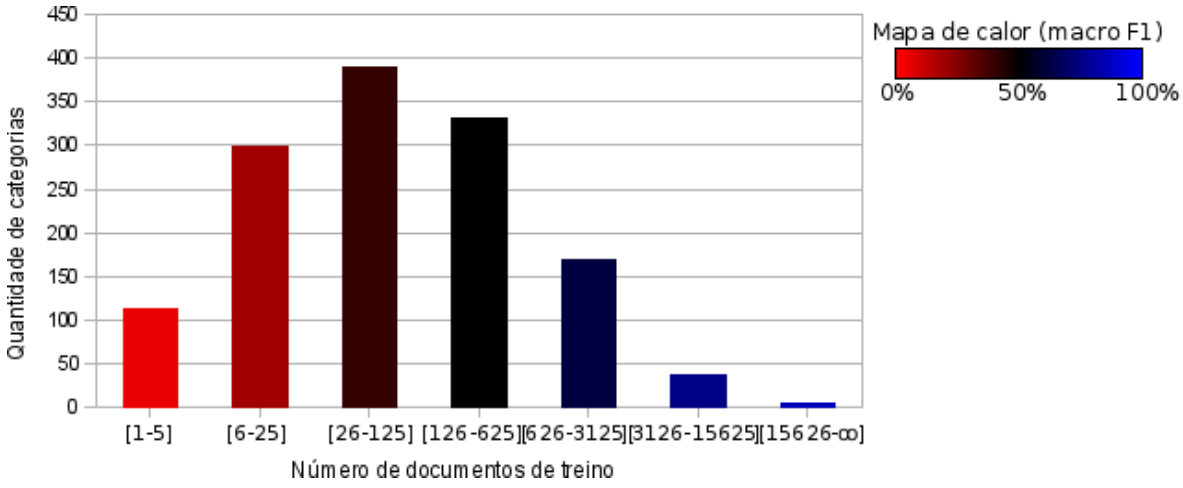


Figura 6.4: Histograma da medida F do Top-Down Adaboost por número de documentos de treino

Capítulo 7

Discussão

Primeiramente, os resultados apresentados mostraram que a própria avaliação de sistemas de classificação para os problemas de Classificação Hierárquica Multi-rótulo é muito mais complexa que a avaliação dos sistemas de classificação tradicionais. Esta avaliação deve ter em consideração quais são as principais características do sistema de classificação e deve-se obter métricas capazes de demonstrar como os sistemas de classificação estão atendendo estas características. Mostramos que as métricas utilizadas em nossos resultados experimentais conseguem extrair importantes características dos sistemas de classificação.

Na avaliação das métricas visivelmente vimos que a classificação se torna cada vez mais difícil à medida que se aprofunda na hierarquia, isto acontece por diversas razões tais como os conceitos semânticos das classes se tornam mais próximos, a quantidade de informação de treino se torna menos abundante e o número de classes aumenta. Esta alteração das características da classificação demanda diferentes características dos sistemas de classificação e portanto algoritmos que apresentam um bom resultado nos níveis menos profundos da hierarquia podem apresentar resultados ruins sob as circunstâncias de classificação dos níveis mais profundos da hierarquia, neste ambiente, algoritmos que consigam se adequar automaticamente às diferentes características da classificação se tornam algoritmos bem promissores para os problemas de HMC.

Durante a análise dos resultados obtidos vimos que mesmo algoritmos com arquitetura plana podem utilizar a hierarquia para decidir diferentes limiares de assinalamento e isto já seria suficiente para melhorar os sistemas de assinalamento.

Podemos ver pelos resultados que o nosso algoritmo MLAC plano conseguiu apresentar resultados competitivos. Porém, para classes maiores o algoritmo desenvolvido tende a apresentar superajuste e assim apresenta uma precisão bem menor que dos demais algoritmos. Por esta razão, este algoritmo quando executado sob uma arquitetura

Top-Down apresentou resultados muito aquém dos demais. Este algoritmo é uma má opção se as classificações nos primeiros níveis da hierarquia são importantes.

Esta solução, apesar de não apresentar bom desempenho diretamente não sofre dos problemas comuns enfrentados pelos sistemas de classificação Top-Down (PAM) e em particular da propagação de erros.

Pudemos ver pelos resultados que o Adaboost conseguiu um excelente desempenho tanto na arquitetura plana quanto Top-Down comparados aos sistemas de classificação utilizando outros classificadores-base. Isso ocorre porque em geral os sistemas de boosting/bagging conseguem diminuir os riscos da classificação e se adequarem aos diferentes compromissos viés-variância enfrentados. Portanto tal algoritmo é muito promissor para os problemas de HMC. Cabe ressaltar que o fase de boosting torna o algoritmo robusto enquanto o classificador fraco utilizado baseado em *decision stumps* tem um poder discriminativo bem relevante.

Por fim, vimos que para a decisão de qual o classificador utilizar está fortemente embasada no compromisso viés-variância. Além disso, os problemas de HMC tendem a apresentar um compromisso viés-variância muito complexo que pode se alterar completamente de acordo com a altura da árvore. Essa tendência nos induz a propor que diferentes classificadores adequados a diferentes compromissos viés-variância sejam utilizados em diferentes níveis da hierarquia e que dificilmente um único classificador base com configurações semelhantes se adeque bem às diferentes classificações dos métodos locais apresentados.

Capítulo 8

Conclusão

Neste trabalho apresentamos um detalhado estudo da literatura. Este estudo teve o objetivo de delinear as diferentes estratégias existentes para a solução dos problemas de HMC (Classificação Multi-rótulo Hierárquica). e permitiu que definíssemos uma taxonomia de tais algoritmos inclusive tentando unificar alguns termos diferentes apresentados na literatura que constituem técnicas semelhantes. Esta diferenciação dificulta o estudo e comparação das diferentes técnicas.

Apresentamos o resultado experimental em uma nova base de dados de atividades econômicas. Essa base de dados é extremamente importante em diferentes contextos para as entidades governamentais como a Secretária da Receita Federal, o IBGE dentre outros.

Neste trabalho, foi implementado uma técnica de classificação associativa postergada preparada para a classificação em bases de dados com um número alto de classes. Além disso, esta técnica foi incrementada com várias funcionalidades adicionais tanto para a melhora da eficiência quanto da efetividade da técnica, como a seleção de características segundo diversos critérios. O trabalho ainda fez parte de um projeto que desenvolveu um novo arcabouço, o Massifica, para a comparação de técnicas de classificação HMC. Este arcabouço permite a rápida comparação de diferentes algoritmos de classificação utilizando variadas estratégias de navegação. Além do nosso algoritmo, os algoritmos Adaboost e SVM-Perf utilizados nesta dissertação já estão integrados a este arcabouço.

Concluímos que apesar de que vários trabalhos de classificação apresentarem resultados impressionantes quanto ao desempenho, estes trabalhos focam apenas no primeiro nível de uma hierarquia e que à medida que se tenta classificar em níveis mais profundos a acurácia e desempenho diminui drasticamente. Mais ainda, classificadores muito eficientes e portanto considerados superiores quando aplicados no primeiro da

hierarquia não necessariamente serão os classificadores com melhor desempenho em níveis mais baixos da hierarquia uma vez que a complexidade da classificação aumenta consideravelmente tanto pela irregularidade da informação, pela semelhança entre as classes que tende a aumentar, a falta de informação sobre classes críticas dentre outros vários fatores. Realizamos a análise de três principais algoritmos utilizando tanto a arquitetura plana quanto a arquitetura Top-Down. Mostramos que o compromisso viés-variância é um dos principais fatores a ser analisado para o desenvolvimento de algoritmos para o problema de HMC. Além disso, muitos algoritmos de classificação tradicional e classificação multi-rótulo se baseiam em premissas que não são válidas neste tipo de problema tornando as estratégias aplicadas por estes algoritmos ineficientes.

Como trabalhos futuros, pretende-se realizar um estudo mais apurado das diferentes técnicas de seleção de características assim como estudar como tais técnicas se relacionam com os diferentes algoritmos de classificação existentes. É necessária a extensão deste estudo para outras bases de dados HMC. Outra estratégia promissora é o estudo de técnicas de *boostraping* específica para os problemas de HMC, uma vez que a geração de bases de dados confiáveis e populosas ainda é um dos problemas principais a serem tratados. Pretendemos estender o estudo da adequação do compromisso viés-variância, e como este estudo pode ser utilizada sob demanda para a utilização de diferentes algoritmos.

Concluímos ainda a necessidade na área de "benchmarks" mais realistas e complexos para o avanço da área de classificação já que um número extenso de trabalhos é desenvolvido utilizando bases de dados extremamente comportadas e pequenas. Sob estas bases, algoritmos supostamente superiores podem apresentar resultados muito piores na prática. É necessário maior discernimento do quão expressivos e/ou específicos são os ganhos realizados de várias técnicas de classificação conforme já se discute na literatura [Hand, 2006].

Referências Bibliográficas

Bade, K.; Hullermeier, E. & Nurnberger, A. (2006). Hierarchical classification by expected utility maximization. *Data Mining, IEEE International Conference on*, 0:43–52.

Boutell, M. R.; Luo, J.; Shen, X. & Brown, C. M. (2004). Learning multi-label scene classification. *Pattern Recognition*, 37(9):1757–1771.

Cesa-Bianchi, N.; Gentile, C. & Zaniboni, L. (2006). Incremental algorithms for hierarchical classification. *J. Mach. Learn. Res.*, 7:31--54.

Chen, W.; Yan, J.; Zhang, B.; Chen, Z. & Yang, Q. (2007). Document transformation for multi-label feature selection in text categorization. Em *ICDM '07: Proceedings of the 2007 Seventh IEEE International Conference on Data Mining*, pp. 451--456, Washington, DC, USA. IEEE Computer Society.

Coenen, F. & Leng, P. (2007). The effect of threshold values on association rule based classification accuracy. *Data Knowl. Eng.*, 60(2):345--360.

Cortes, C. & Vapnik, V. (1995). Support-vector networks. *Mach. Learn.*, 20(3):273--297.

de Souza, A.; Pedroni, F.; Oliveira, E.; Ciarelli, P. M.; Henrique, W. F. & Veronese, L. (2007). Automated free text classification of economic activities using vg-ram weightless neural networks. *Intelligent Systems Design and Applications, International Conference on*, 0:782–787.

Dietterich, T. G. (2000). Ensemble methods in machine learning. Em *MCS '00: Proceedings of the First International Workshop on Multiple Classifier Systems*, pp. 1--15, London, UK. Springer-Verlag.

Domingos, P. (1999). Metacost: a general method for making classifiers cost-sensitive. Em *KDD '99: Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 155--164, New York, NY, USA. ACM.

Dumais, S. T. & Chen, H. (2000). Hierarchical classification of Web content. Em Belkin, N. J.; Ingwersen, P. & Leong, M.-K., editores, *Proceedings of SIGIR-00, 23rd ACM International Conference on Research and Development in Information Retrieval*, pp. 256--263, Athens, GR. ACM Press, New York, US.

Figueiredo, F. S. (2008). Extração de Características para Classificação Automática de Textos[portuguese]. Dissertação de mestrado, Departamento de Ciência da Computação, Universidade Federal de Minas Gerais, Belo Horizonte, Brasil.

Geman, S.; Bienenstock, E. & Doursat, R. (1992). Neural networks and the bias/variance dilemma. *Neural Comput.*, 4(1):1--58.

Godbole, S. & Sarawagi, S. (2004). Discriminative methods for multi-labeled classification. Em Dai, H.; Srikant, R. & Zhang, C., editores, *PAKDD*, volume 3056 of *Lecture Notes in Computer Science*, pp. 22–30. Springer.

Granitzer, M. (2003). Hierarchical text classification using methods from machine learning. Dissertação de mestrado, Graz University of Technology, Graz, Austria.

Hand, D. J. (2006). Classifier technology and the illusion of progress. *Statistical Science*, 21:1.

Hofmann, T.; Cai, L. & Ciaramita, M. (2003). Learning with taxonomies: Classifying documents and words.

Joachims, T. (2006). Training linear svms in linear time. Em *KDD '06: Proceedings of the 12th ACM SIGKDD: International Conference on Knowledge Discovery and Data Mining*, pp. 217--226, Philadelphia, PA, USA. ACM Press.

Jr, A. T.; Rashid, R. F.; Young, M. W.; Golub, D. B.; Thompson, M. R.; Bolosky, W. & Sanzi, R. (1987). A unix interface for shared memory and memory mapped files under mach. Em *In Proceedings of the Summer 1987 USENIX Conference*, pp. 53--67.

Kiritchenko, S. (2005). *Hierarchical Text Categorization and Its Application to Bioinformatics*. Tese de doutorado, Faculty of Graduate and Postdoctoral Studies, Ottawa, Canada.

Kotsiantis, S. B. & Pintelas, P. E. (2004). Combining bagging and boosting. *IJCI '04: Proceedings of the International Journal of Computational Intelligence*, 1(4):324–333.

Lewis, D. D.; Yang, Y.; Rose, T. G. & Li, F. (2004). Rcv1: A new benchmark collection for text categorization research. *J. Mach. Learn. Res.*, 5:361--397.

Lima, L. R. S.; Laender, A. H. F. & Ribeiro-Neto, B. A. (1998). A hierarchical approach to the automatic categorization of medical documents. Em *CIKM '98: Proceedings of the seventh international conference on Information and knowledge management*, pp. 132--139, New York, NY, USA. ACM Press.

Liu, T.-Y.; Yang, Y.; Wan, H.; Zeng, H.-J.; Chen, Z. & Ma, W.-Y. (2005). Support vector machines classification with a very large-scale taxonomy. *SIGKDD Explor. Newsl.*, 7(1):36--43.

Manning, C. D.; Raghavan, P. & Schtze, H. (2008). *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA.

McCallum, A. (1999). Multi-label text classification with a mixture model trained by em.

Mccallum, A. & Nigam, K. (1998). A comparison of event models for naive bayes text classification. Em *AAAI Workshop on Learning for Text Categorization*.

McCallum, A. K.; Rosenfeld, R.; Mitchell, T. M. & Ng, A. Y. (1998). Improving text classification by shrinkage in a hierarchy of classes. Em Shavlik, J. W., editor, *Proceedings of ICML-98, 15th International Conference on Machine Learning*, pp. 359--367, Madison, US. Morgan Kaufmann Publishers, San Francisco, US.

Mitchell, T. M. (1980). The need for biases in learning generalizations. Relatório técnico, Department of Computer Science, Rutgers University.

Punera, K. & Ghosh, J. (2008). Enhanced hierarchical classification via isotonic smoothing. Em *WWW '08: Proceeding of the 17th international conference on World Wide Web*, pp. 151--160, New York, NY, USA. ACM.

Rak, R.; Kurgan, L. & Reformat, M. (2005). Multi-label associative classification of medical documents from medline. Em *ICMLA '05: Proceedings of the Fourth International Conference on Machine Learning and Applications*, pp. 177--186, Washington, DC, USA. IEEE Computer Society.

Rousu, J.; Saunders, C.; Szedmak, S. & Shawe-Taylor, J. (2006). Kernel-based learning of hierarchical multilabel classification models. *Journal of Machine Learning Research*, 7:1601--1626.

Schapire, R. E. & Singer, Y. (1998). Improved boosting algorithms using confidence-rated predictions. Em *COLT' 98: Proceedings of the eleventh annual conference on Computational learning theory*, pp. 80--91, New York, NY, USA. ACM.

Schneider, K.-M. (2003). A comparison of event models for naive bayes anti-spam e-mail filtering. Em *EACL '03: Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics*, pp. 307--314, Morristown, NJ, USA. Association for Computational Linguistics.

Seeger, M. W. (2008). Cross-validation optimization for large scale structured classification kernel methods. *J. Mach. Learn. Res.*, 9:1147--1178.

Steinbruch, D. (2006). Um estudo de algoritmos para classificação automática de textos utilizando naive-Bayes[portuguese]. Dissertação de mestrado, Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, Brasil.

Sun, A. (2004). Blocking reduction strategies in hierarchical text classification. *IEEE Trans. on Knowl. and Data Eng.*, 16(10):1305--1308. Senior Member-Ee-Peng Lim and Member-Wee-Keong Ng and Fellow-Jaideep Srivastava.

Sun, A. & Lim, E.-P. (2001). Hierarchical text classification and evaluation. Em *ICDM '01: Proceedings of the 2001 IEEE International Conference on Data Mining*, pp. 521--528, Washington, DC, USA. IEEE Computer Society.

Tatu, M.; Srikanth, M. & D'Silva, T. (2008). Rsdc'08: Tag recommendations using bookmark content. Em *Proceedings of ECML PKDD Discovery Challenge (RSDC08)*, pp. 96--107.

Toutanova, K.; Chen, F.; Popat, K. & Hofmann, T. (2001). Text classification in a hierarchical mixture model for small training sets. Em Paques, H.; Liu, L. & Grossman, D., editores, *CIKM '01: Proceedings of the tenth international conference on Information and knowledge management*, pp. 105--113, New York, NY, USA. ACM Press.

Tsoumakas, G. & Katakis, I. (2007). Multi-label classification: An overview. Em *International Journal of Data Warehousing and Mining*, pp. 1--13. Idea Group Publishing.

Valentini, G. & Dietterich, T. G. (2004). Bias-variance analysis of support vector machines for the development of svm-based ensemble methods. *J. Mach. Learn. Res.*, 5:725--775.

Veloso, A.; Jr., W. M. & Zaki, M. J. (2008). Calibrated lazy associative classification. Em de Amo, S., editor, *SBBD*, pp. 135–149. SBC.

Veloso, A.; Meira, Jr., W.; Cristo, M.; Gonçalves, M. & Zaki, M. (2006). Multi-evidence, multi-criteria, lazy associative document classification. Em *CIKM '06: Proceedings of the 15th ACM international conference on Information and knowledge management*, pp. 218--227, New York, NY, USA. ACM.

Veloso, A.; Meira, Jr., W.; Gonçalves, M. & Zaki, M. (2007). Multi-label lazy associative classification. Em *PKDD 2007: Proceedings of the 11th European conference on Principles and Practice of Knowledge Discovery in Databases*, pp. 605--612, Berlin, Heidelberg. Springer-Verlag.

Veloso, A. A. (2007). *On-demand associativa Classification.* Tese de doutorado, Departamento de Ciência da Computação, Universidade Federal de Minas Gerais, Belo Horizonte, Brasil.

Wolpert, D. H. (1992). Stacked generalization. *Neural Networks*, 5:241--259.

Wu, F.; Zhang, J. & Honavar, V. (2005). Learning classifiers using hierarchically structured class taxonomies. Em *SARA 2005: Proceedings of the Symposium on Abstraction, Reformulation, and Approximation*, volume 3607, pp. 313--320, Edinburgh, Berlin. Springer-Verlag.

Zhang, H. (2005). Exploring conditions for the optimality of naïve bayes. *IJPRAI*, 19(2):183–198.

Zhang, M.-L. & Zhou, Z.-H. (2007). Ml-knn: A lazy learning approach to multi-label learning. *Pattern Recogn.*, 40(7):2038--2048.