CLAUDIANE MARIA OLIVEIRA

UM MÉTODO BASEADO EM REGRAS DE ASSOCIAÇÃO PARA CLASSIFICAÇÃO DE OFERTAS DE PRODUTOS DE LOJAS DE COMÉRCIO ELETRÔNICO

LAVRAS - MG

2014

CLAUDIANE MARIA OLIVEIRA

UM MÉTODO BASEADO EM REGRAS DE ASSOCIAÇÃO PARA CLASSIFICAÇÃO DE OFERTAS DE PRODUTOS DE LOJAS DE COMÉRCIO ELETRÔNICO

Dissertação apresentada à Universidade Federal de Lavras, como parte das exigências do Programa de Pós-Graduação em Ciência da Computação, área de concentração em Banco de Dados e Engenharia de Software, para a obtenção do título de Mestre.

Orientador

Dr. Denilson Alves Pereira

LAVRAS - MG

2014

Ficha Catalográfica Elaborada pela Coordenadoria de Produtos e Serviços da Biblioteca Universitária da UFLA

Oliveira, Claudiane Maria.
Um método baseado em regras de associação para classificação de ofertas de produtos de lojas de comércio eletrônico / Claudiane Maria Oliveira. – Lavras : UFLA, 2014.
77 p. : il.

Dissertação (mestrado) – Universidade Federal de Lavras, 2014.
Orientador: Denilson Alves Pereira.
Bibliografia.

1. Resolução de entidades. 2. Classificação de ofertas de produtos. 3. Regras de associação. 4. Mineração de dados. 5. Recuperação de informação. I. Universidade Federal de Lavras. II. Título.

CDD – 519.537

CLAUDIANE MARIA OLIVEIRA

UM MÉTODO BASEADO EM REGRAS DE ASSOCIAÇÃO PARA CLASSIFICAÇÃO DE OFERTAS DE PRODUTOS DE LOJAS DE COMÉRCIO ELETRÔNICO

Dissertação apresentada à Universidade Federal de Lavras, como parte das exigências do Programa de Pós-Graduação em Ciência da Computação, área de concentração em Banco de Dados e Engenharia de Software, para a obtenção do título de Mestre.

APROVADA em 31 de Outubro de 2014.

Dr. Denilson Alves Pereira	UFLA
Dr. Ahmed Ali Abdalla Esmin	UFLA
Dr. Thierson Couto Rosa	UFG

Dr. Denilson Alves Pereira

Orientador

LAVRAS - MG

2014

Dedico esta conquista aos meus pais Carlos e Luzia,
pela imensa força e por sempre acreditarem que eu seria capaz.
Ao meu amado esposo Flávio, que me apoiou plenamente durante toda esta
jornada. Acompanhou-me durante todas as madrugadas, sem dormir, em
todas as dificuldades, em todas as correrias e deadlines.

AGRADECIMENTOS

Agradeço, em primeiro lugar a Deus, pela minha vida e por estar sempre ao meu lado iluminando o meu caminho.

Ao meu esposo Flávio, pessoa que esteve plenamente ao meu lado, agradeço muito pelo carinho, pela paciência e por sua capacidade de me tranquilizar em cada correria vivida.

Aos meus pais, Carlos e Luzia, que, mesmo à distância, sempre estiveram presentes, apoiando-me para eu chegar até aqui.

Agradeço à Universidade Federal de Lavras, ao Departamento de Ciência da Computação e ao Programa de Pós- Graduação em Ciência da Computação da UFLA, pela estrutura oferecida e pela oportunidade de realização do Mestrado.

Agradeço a Capes pelo apoio financeiro que tornou possível a realização do Mestrado.

Agradeço ao meu orientador Prof. Denilson pela confiança, conselhos e orientação que possibilitaram à realização deste.

Agradeço a todos os professores, colegas, secretárias e funcionários do Departamento de Ciência da Computação da UFLA pelo conhecimento passado, pelas ajudas e pelo apoio.

Aos meus amigos e colaboradores que souberam me entender e que oferecerem o ombro amigo para a realização deste trabalho.

E, a todos que contribuíram, torceram e acreditaram em minha conquista: muito obrigada!

RESUMO

Os serviços de comparar preços são bastante utilizados pelos consumidores de lojas de comércio eletrônico. Essas lojas possibilitam aos consumidores pesquisar por produto ou categoria de produtos. O processo de identificar quais descrições em diferentes lojas correspondem ao mesmo produto é uma tarefa desafiadora para tais sistemas. Esse desafio torna-se, ainda, mais difícil, em razão da grande quantidade de produtos com descrições similares mas referente a produtos distintos. Neste trabalho, foi proposto um método que usa regras de associação para classificar ofertas de produtos de lojas de comércio eletrônico. O método proposto é de aprendizagem de máquina supervisionado, o qual usa a descrição do produto para treinar um classificador. O modelo gerado é um conjunto de regras de associação que identifica as ofertas de produtos. Os experimentos realizados, utilizando o método proposto, obtiveram os melhores resultados frente aos *baselines*.

Palavras-chave: Resolução de entidades. Classificação de ofertas de produtos. Regras de Associação. Mineração de Dados. Recuperação de Informação.

ABSTRACT

The practice of comparing prices is widely used by the customers of e-commerce stores, which enable them to research by products or category of products. The process of identifying descriptions, which are according to the same product in different stores, is a challenging task for such systems. This challenge becomes even more difficult due to great variety of products under similar description, but referring to distinct products. In this paper, we proposed a method which uses association rules to classify product offers in e-commerce stores. This is a supervised machine learning method, which uses the product description to train a classifier. The generated model is a set of association rules which identifies product offers. The experiments performed using this method were found to result in better achievements than the baselines.

Keywords: Entities resolution. Classification of product offers. Association rules. Data mining. Information retrieval.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Repositórios de dados na Web normalmente contêm referências para várias entidades do mundo real. É comum que uma mesma entidade seja rotulada de formas distintas, e uma tarefa importante das aplicações é fazer a agregação dessas variações, agrupando as diferentes descrições referentes a uma mesma entidade. Nesta seção, é apresentada a contextualização do trabalho, a proposta do projeto, a justificativa, os objetivos geral e específicos, o tipo de pesquisa e a estrutura do trabalho.

1.1 Contextualização

Os serviços de comparar preços são bastante utilizados pelos consumidores de lojas de comércio eletrônico em virtude da crescente utilização de *sites* para a compra de produtos. Esses *sites* possibilitam aos consumidores pesquisarem um determinado produto ou procurarem-no por categoria, tipo e marca. De acordo com Bilenko, Basil e Sahami (2005), um grande desafio para tais *sites* é identificar quais as descrições em diferentes lojas *online* correspondem ao mesmo produto.

Embora alguns tipos de produtos possuam identificadores únicos, como o *International Standard Book Number* (ISBN) para livros, as descrições de um mesmo produto geralmente podem ser bastante diferentes entre as várias lojas que o disponibilizam (BILENKO; BASIL; SAHAMI, 2005). Além de descrições diferentes para um mesmo produto, pode ocorrer, também, de se ter descrições muito parecidas para produtos diferentes.

Para exemplificar, na Figura 1 ilustram-se alguns resultados de ofertas da máquina de busca Google Shopping[1], para o produto *HP Photosmart*

[1] http://www.google.com/shopping

C4680. Os resultados referem-se aos produtos relacionados à pesquisa, disponíveis em várias lojas. Nesta figura, identifica-se a existência de descrições diferentes para o mesmo produto nos três primeiros resultados. Já o quarto resultado refere-se a um acessório para esse produto.

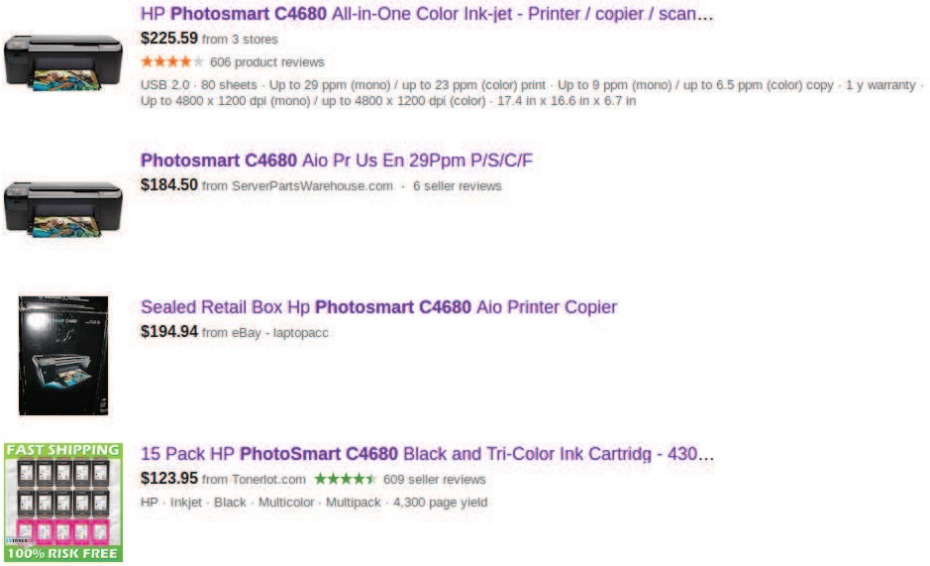

Figura 1 Ofertas do produto *HP Photosmart C4680* no Google Shopping

Ainda na Figura 1, é possível identificar que o Google Shopping realiza uma agregação de produtos, visto que o primeiro resultado dessa pesquisa refere-se ao agrupamento de 3 ofertas diferentes ($255.59 from 3 stores). Porém, o segundo e o terceiro resultados, também, referem-se ao mesmo produto do primeiro e mostram-se disponíveis separadamente. Esses três resultados, os quais deveriam ser agrupados, mostram a necessidade de melhoria no algoritmo de agrupamento do Google Shopping.

Em geral, as ofertas dos produtos possuem um código embutido em suas descrições, como, por exemplo, o código *C4680* presente em todas as

ofertas, conforme mostra a Figura 1. Porém, ainda conforme o exemplo da Figura 1, esse código é propenso a erro e pode levar a uma decisão de classificação insuficiente em virtude da associação do código a um acessório do produto (KÖPCKE et al., 2012).

Considere as ofertas de produtos na Figura 2. Trata-se de descrições muito similares para produtos distintos. Métodos tradicionais que usam medidas de similaridade de cadeias de caracteres, tais como Similaridade do Cosseno (BAEZA-YATES; RIBEIRO-NETO, 2011), Coeficiente de Similaridade de Jaccard (JACCARD, 1901), ou Distância de Edição (LEVENSHTEIN, 1966) não funcionam bem para agrupar as cadeias de caracteres desses exemplos. Tais métodos tendem a considerar similares os produtos da Figura 2 e diferentes as duas primeiras ofertas da Figura 1, que são da mesma classe.

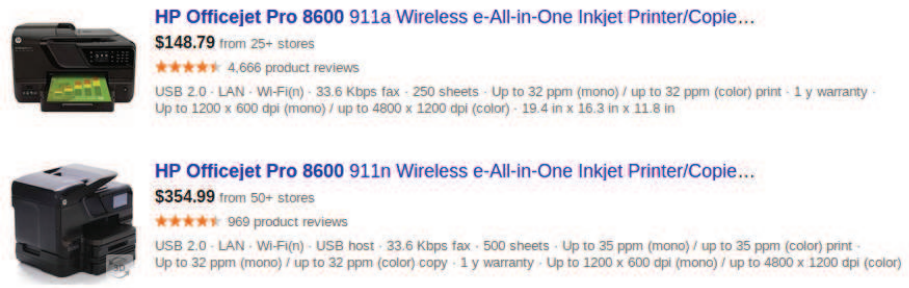

Figura 2 Ofertas de produtos similares.

Dadas várias referências a entidades, tais como descrições de produtos, o processo de identificar quais descrições referem-se à mesma entidade do mundo real é conhecido por resolução de entidades (*Entity Resolution - ER*) (BENJELLOUN et al., 2009; BHATTACHARYA; GETOOR, 2007; BILENKO et al., 2003).

1.2 Proposta deste Trabalho

Nas últimas décadas, várias pesquisas foram realizadas desenvolvendo soluções para o problema de resolução de entidades. Porém, pela grande maioria das pesquisas considera-se que as referências são formadas com base em registros estruturados. Especificamente, em ofertas de produtos, as entradas constituem-se de dados descritos em texto livre, dificultando ainda mais a identificação das entidades. Nessas descrições, frequentemente, são encontrados erros de grafia, abreviações e diferentes nomes para o mesmo atributo e, como resultado, uma mesma entidade do mundo real poderá ter múltiplas representações (CHAUDHURI; GANTI; KAUSHIK, 2006).

A proposta deste trabalho é o desenvolvimento de um método para tratar o problema de classificação de ofertas de produtos, usando regras de associação para encontrar um conjunto de palavras-chave, que distingue cada oferta de produto. Especificamente, o foco do trabalho foi em soluções para tratar dados consistindo de descrições textuais de ofertas de produtos de lojas de comércio eletrônico. As ofertas dos produtos em geral possuem um código de produto embutido nas descrições textuais que identificam unicamente cada produto. Por exemplo, na Figura 1, o código do produto pesquisado, *HP Photosmart C4680*, é *C4680*. No entanto, em algumas situações, esse código aparece em ofertas de acessórios do produto pesquisado, e têm, ainda, situações em que esse código não existe. Apesar de nem sempre um código ocorrer nas descrições, quando ocorre, ele é importante para identificar referências a entidades. Neste trabalho, buscou-se desenvolver soluções que considerem esse fato.

O método proposto é de aprendizagem de máquina supervisionada, o qual usa a descrição do produto para treinar um classificador, cujo modelo

gerado é um conjunto de regras de associação para identificar as ofertas de produtos. As regras de associação são da forma $X \rightarrow c_i$, onde X é um conjunto de palavras-chave e c_i é a classe da oferta de produto. Na fase de teste, um conjunto de palavras-chave da instância a ser testada é gerado e combinado com o antecedente das regras no modelo gerado na fase de treino.

1.3 Justificativa

A agregação de produtos (*Product Matching*) é uma variação do problema de resolução de entidades para identificar ofertas referentes ao mesmo produto. A grande quantidade de produtos com descrições muito similares, mas referente a produtos diferentes, dificulta consideravelmente a solução desse problema (KÖPCKE et al., 2012).

Embora existam diversos trabalhos propostos na literatura, (CORTEZ et al., 2011; GHANI et al., 2006; KANNAN et al., 2011a, 2011b; KOPCKE et al., 2012; KOPCKE; THOR; RAHM, 2010; NGUYEN et al., 2011), os resultados mostram que ainda existe espaço para pesquisas e proposições de melhorias no que se refere ao agrupamento de ofertas de produtos (KÖPCKE; THOR; RAHM, 2010). Assim, o foco deste trabalho foi o desenvolvimento de soluções para fazer a agregação de ofertas de produtos, especificamente em soluções para tratar dados consistindo de descrições textuais de ofertas de produtos em lojas de comércio eletrônico.

1.4 Objetivo Geral e Específicos

Objetivou-se neste trabalho propor um método, baseado em regras de associação, para a classificação de ofertas de produtos disponíveis em lo-

jas de comércio eletrônico. Para alcançar este objetivo geral, foram definidos os seguintes objetivos específicos:

1. Desenvolver estratégias que identifiquem os códigos dos produtos embutidos nas descrições textuais e que os utilizem como informação auxiliar para o processo de classificação de ofertas de produtos de lojas de comércio eletrônico.

2. Propor e avaliar o método para classificação de ofertas de produtos baseado em regras de associação.

3. Construir um ambiente de testes para realizar os experimentos.

4. Avaliar os resultados obtidos comparando-os com os outros métodos.

1.5 Tipo de Pesquisa

Esta pesquisa pode ser classificada como pesquisa exploratória quanto ao seu objetivo, experimental quanto aos procedimentos, quantitativa pela sua abordagem e em laboratório quanto ao local de execução. Quanto à sua natureza, é uma pesquisa aplicada, pois utiliza o conhecimento da pesquisa básica para gerar conhecimentos com finalidade de aplicação (WAINER, 2007). Esta pesquisa, também, pode ser classificada como *design science*, pois objetiva-se desenvolver novos conhecimentos de soluções para os problemas em estudo (AKEN, 2005).

1.6 Estrutura deste Documento

O restante deste documento está organizado da seguinte forma: um referencial teórico sobre resolução de entidades, funções de similaridades,

regras de associação, classificação de dados e classificação de produtos é apresentado no Capítulo 2; o método proposto para classificação de ofertas de produtos é apresentado no Capítulo 3; a a configuração dos experimentos é apresentada no Capítulo 4; a avaliação experimental realizada é apresentada no Capítulo 5; e a conclusão e trabalhos futuros é explicitada no Capítulo 6.

2 REFERENCIAL TEÓRICO

Neste capítulo estão descritos conceitos de resolução de entidades, funções de similaridade, mineração de dados, regras de associação, classificação de dados e, especificamente classificação de produtos, bem como trabalhos relacionados.

2.1 Resolução de Entidades

Dadas várias referências a entidades, como, por exemplo, descrição de produtos, o processo de identificar quais registros representam a mesma entidade do mundo real e agrupá-los é conhecido como resolução de entidades (*Entity Resolution - ER*) (BENJELLOUN et al., 2009; BILENKO; MOONEY, 2003). Em outras palavras, resolução de entidades é o processo de identificar grupos de registros que representam a mesma entidade no mundo real.

O problema de resolução de entidades tem sido estudado em diferentes áreas de conhecimento com nomes diferentes tais como *record linkage* (FELLEGI; SUNTER, 1969), *identity uncertainty* (PASULA et al., 2003), *citation matching* (LEE et al., 2007), *merge/purge* (HERNÁNDEZ; STOLFO, 1995), *deduplication* (CARVALHO et al., 2006) dentre outros. Especificamente o problema de resolução de entidades relacionado à oferta de produtos é referenciado por agregação de produtos (*product matching*) (KÖPCKE et al., 2012).

Esse problema é conhecido em várias aplicações. Por exemplo, duas empresas que se juntam podem querer combinar os registros de seus clientes: para um determinado cliente que fazia parte das duas empresas, poder-se-ia criar um único registro composto pelas informações conhecidas de ambas as

empresas (MENESTRINA; WHANG; GARCIA-MOLINA, 2010; WANG et al., 2012). Outra aplicação são listas de correspondência as quais podem conter várias entidades representando o mesmo endereço físico, mas cada registro com uma pequena diferença, contendo algum erro ou faltando alguma informação (BENJELLOUN et al., 2009).

A importância e a dificuldade do problema de resolução de entidades desencadeou muitas pesquisas em diferentes variações do problema, e numerosas abordagens são propostas, principalmente para dados estruturados (KÖPCKE; THOR; RAHM, 2010).

As propostas encontradas na literatura, geralmente, consideram os dados de entrada como registros estruturados divididos em atributos e utilizam alguma medida de similaridade para comparar os valores dos atributos dos registros e, com base nos valores das similaridades de cada atributo, derivam uma decisão de agregação para cada par de registros. Para tratar o problema, tem sido utilizada a abordagem de aprendizagem de máquina, com técnicas supervisionadas, as quais utilizam um conjunto de dados rotulados para construir o modelo (BILENKO; MOONEY, 2003; PASULA et al., 2003) e técnicas não supervisionadas, as quais não são necessários dados rotulados (CARVALHO; SILVA, 2003; CHAUDHURI; GANTI; MOTWANI, 2005).

Já, as abordagens, baseadas em *clustering* têm o objetivo de agrupar um conjunto de registros sem rótulos, de maneira que os registros do mesmo *cluster* sejam semelhantes entre si, de acordo com alguma medida de similaridade e os que estão em *cluster* distintos não sejam similares (HAN; KAMBER; PEI, 2011). As abordagens baseadas em classificação, objetivam-se

organizar objetos em uma determinada categoria entre diversas categorias pré-definidas (TAN; STEINBACH; KUMAR, 2009).

Outras abordagens utilizam a similaridade entre atributos de registros estruturados, pontuando a similaridade entre dois registros levando em consideração o relacionamento dos atributos (BHATTACHARYA; GETOOR, 2007). A dificuldade desta abordagem é que esses relacionamentos nem sempre estão disponíveis, dificultando a sua utilização.

Em outra abordagem, Pereira et al. (2011) propõem utilizar a Web como fonte de informação adicional para desambiguar entidades e obter nomes canônicos para elas. Em outro trabalho, foi desenvolvido uma interface para o processo de resolução de entidades interativa em dados relacionais. Esta interface permite aos usuários fazerem o uso do contexto relacional da entidade para a tomada de decisões (KANG et al., 2008).

2.2 Funções de Similaridade

No problema de classificação, normalmente são utilizadas funções de similaridade para comparar a semelhança entre duas referências. A comparação é baseada em uma medida de similaridade aplicada sobre alguns dos atributos das referências comparadas. Para atributos do tipo *string*, várias funções de similaridade foram propostas na literatura. Uma função de similaridade de *strings* faz o mapeamento de um par de *strings* s_i e s_j para um número real r. Os valores de similaridade r normalmente são dados no intervalo entre 0 e 1, em que valores próximos de 0 significam baixa similaridade e valores de r próximos de 1 significam alta similaridade. Na literatura, pode-se encontrar, também, o termo funções de distância, semelhante a funções

de similaridade, porém baixo valor significa alta similaridade (BILENKO; MOONEY, 2003; PEREIRA, 2009).

Funções de similaridade de *string* podem ser divididas em funções baseadas em caracteres e em *tokens*. Funções baseadas em caracteres usam operações de edição de caracteres, fazendo comparações de exclusões, inserções e atualização. Funções baseadas em *tokens* transformam *strings* em conjunto de *tokens*, sobre os quais ocorre o cálculo da similaridade; normalmente *token* é uma palavra (PEREIRA, 2009).

Assim, a realização do cálculo de similaridade exige uma adaptação para definir qual medida de similaridade utilizar para cada atributo da base de dados no que diz respeito a um determinado domínio de dados específico. As medidas de similaridade, baseadas em caracteres, são recomendadas para *strings* curtas, com poucas variações; enquanto medidas, baseadas em *tokens* e com representação em espaço vetorial (*vector-space*), como, por exemplo, a similaridade do cosseno, são mais apropriadas para atributos que contêm strings com mais variações (BILENKO; MOONEY, 2003).

2.2.1 Funções de Similaridade Baseadas em Caracteres

A função de similaridade baseada em caracteres mais conhecida é a Distância de Edição, *Edit Distance* ou Distância Levenshtein (LEVENSHTEIN, 1966), dentre outras existentes como Jaro (JARO, 1978) e Jaro-Winkler (WINKLER, 1999). A Distância de Edição é obtida por meio do número mínimo de operações, a saber: inserção, eliminação ou substituição de caracteres necessários para transformar uma *string* em outra.

Sejam s_i e s_j duas *strings*, $ed(s_i, s_j)$ a distância de edição entre as *strings* s_i e s_j e $mim(|s_i|, |s_j|)$ o tamanho da *string* menor (s_i ou s_j). Assim,

a similaridade baseada na distância de edição entre as *strings* s_i e s_j é dada pela Equação (1).

$$edSim(s_1, s_2) = 1 - \frac{ed(s_1, s_2)}{mim(|s_1|, |s_2|)} \tag{1}$$

2.2.2 Funções de Similaridade Baseadas em *Tokens*

O Coeficiente de Similaridade de *Jaccard* é uma função baseada em *tokens* (ou palavras), utilizada para quantificar a semelhança entre duas strings (JACCARD, 1901).

Sejam s_i e s_j duas *strings*, $|s_i \bigcap s_j|$ a cardinalidade da interseção das palavras das duas *strings*, e $|s_i \bigcup s_j|$ a cardinalidade da união das palavras das duas *strings*. O Coeficiente de Similaridade de Jaccard entre as *strings* s_i e s_j é dado pela Equação 2.

$$Jaccard(s_i, s_j) = \frac{|s_i \bigcap s_j|}{|s_i \bigcup s_j|} \tag{2}$$

A função de similaridade do Cosseno é baseada em *tokens*. Esta função é utilizada para medir a semelhança entre um par de *strings* usando o modelo vetorial. Neste modelo, as *strings* são representadas como vetores em um espaço t-dimensional, em que o t representa uma *features* obtida da *string* (BAEZA-YATES; RIBEIRO-NETO, 2011).

Seja $S = \{s_1, s_2, ..., s_{|S|}\}$ uma coleção de |S| *strings*. Seja $K = \{k_1, k_2, ..., k_{|K|}\}$ o conjunto dos |K| *tokens* distintos que aparecem na coleção S. Para cada par (k_i, s_j), $k_i \in K$ e $s_j \in S$, é associado um peso $w_{i,j} \geqslant 0$. Uma *string* s_j é representada como um vetor de pesos $\vec{s}_j = \{w_{1,j}, w_{2,j}, ..., w_{t,j}\}$. Cada $w_{i,j}$ representa a importância do *token* k_i para descrever o conteúdo semântico da *string* s_j (PEREIRA, 2009).

O grau de similaridade entre as *strings* s_i e s_j é obtido como a correlação entre os vetores $\vec{s_i}$ e $\vec{s_j}$. Essa correlação pode ser quantificada pelo cosseno do ângulo entre esses dois vetores, dado pela Equação 3. Essa medida de similaridade é conhecida como Similaridade do Cosseno.

$$sim(s_i, s_j) = \frac{\Sigma_{l=1}^{t} w_{l,i} \times w_{l,j}}{\sqrt{\Sigma_{l=1}^{t} w_{l,i}^2} \times \sqrt{\Sigma_{l=1}^{t} w_{l,j}^2}} \tag{3}$$

O peso $w_{l,j}$ pode ser calculado pela Equação 4, conforme apresentando em Baeza-Yates e Ribeiro-Neto (2011).

$$w_{l,j} = (1 + log f_{l,j}) * log \frac{|S|}{n_i} \tag{4}$$

onde a $f_{l,j}$ é a frequência do *token* k_l na *string* s_j, n_i é o número de strings de S em que k_l ocorre. O esquema utilizado para realizar os cálculos dos pesos é conhecido por TF-IDF, *term-frequency x inverse document frequency*, onde (TF) considera a frequência do termo na *string* e o (IDF) a raridade do termo na coleção.

As funções de similaridades baseadas em caracteres e *tokens* podem ser combinadas formando métodos híbridos, tais como: a similaridade *Soft TF-IDF*, descrita por Bilenko et al. (2003), ou a similaridade de Segundo Nível descrita por Monge e Elkan (1996) e, a similaridade Jaccard e Distância de Edição (FRENCH; POWELL; SCHULMAN, 2000).

2.3 Regras de Associação

Os avanços na tecnologia de coleta e armazenamento de dados permitiram às organizações acumularem grande quantidade de dados em suas

operações realizadas diariamente. Entretanto, extrair as informações úteis tem provado ser uma atividade extremamente desafiadora.

Mineração de dados é o processo de extração de informação interessante (não trivial, implícita, previamente desconhecida e potencialmente útil) de grandes bases de dados, tais como *data warehouses*, repositórios xml, etc (CHEN; HAN; YU, 1996). As técnicas de mineração de dados podem ser usadas para apoiar várias aplicações de inteligência de negócios tais como criação de perfis para clientes, vendas direcionadas, administração do fluxo de trabalho, formato de organização de loja e também em detecção de fraude (TAN; STEINBACH; KUMAR, 2009).

Mineração de dados (*data mining*), é parte do processo conhecido como *Knowledge Discovery in Database* (KDD), que é um processo geral de conversão de dados brutos em informações úteis (ZHAO; BHOWMICK, 2003). As tarefas centrais da mineração de dados são: análise de agrupamentos, modelagem previsiva, análise de associação e detecção de anomalias.

A análise de associação é utilizada para descobrir relacionamentos interessantes em determinado conjunto de dados (TAN; STEINBACH; KUMAR, 2009). O objetivo das regras de associação é extrair correlações interessantes entre padrões frequentes, estruturas de associações frequentes ou casual de uma base de dados ou repositório de dados. As regras de associação descrevem padrões de relacionamento entre os itens de uma base de dados. Uma de suas típicas aplicações é a análise de transação de compras (*market basket analysis*). Este processo examina padrões de compras de consumidores para determinar produtos que costumam ser adquiridos em conjunto (BERRY; LINOFF, 1997).

Na Tabela 1, é apresentado um exemplo de uma base de dados de transações de compras de um supermercado hipotético. Cada transação é constituída de um identificador único (TID) e pela relação de produtos adquiridos por um cliente.

Tabela 1 Exemplo de transações de compras

TID	Itens
1	{Pão, Leite}
2	{Pão, Fraldas, Cerveja, Ovos}
3	{Leite, Fraldas, Cerveja, Cola}
4	{Pão, Leite, Fraldas, Cerveja}
5	{Pão, Leite, Fraldas, Cola}

Fonte: Tan, Steinbach e Kumar (2009)

Com base neste exemplo, uma estratégia para mineração de regras de associação poderia gerar a seguinte regra: {Fraldas $\rightarrow$ Cerveja}. Esta regra é utilizada para indicar que os clientes que compram fraldas também compram cerveja (TAN; STEINBACH; KUMAR, 2009). Existe uma forte associação derivada da base que indica que pessoas que compram fraldas também compram cerveja, isto é, de quatro compras envolvendo fraldas, três delas incluíam cerveja. O exemplo ilustra uma das características mais atrativas das regras de associação: elas são expressas em uma forma muito fácil de ser compreendida (BERRY; LINOFF, 1997).

As regras de associação foram introduzidas no seguinte formato (AGRAWAL; IMIELINSKI; SWAMI, 1993). Seja $I = \{i_1, i_2, ...i_m\}$ um conjunto de m itens distintos e $D = \{t_1, t_2, ...t_{|D|}\}$ uma base de dados formada por um conjunto de transações, onde cada transação $t_i \in D$ é composta por um conjunto de itens (*itemset*), tal que $t_i \subseteq$ I. Uma regra de associação é

uma expressão na forma A $\Rightarrow$ B (A implica em B ou Se A então B), onde A e B são conjuntos disjuntos de itens, ou seja, A $\bigcap$ B $= \varnothing$.

Seja X um conjunto de itens. Uma propriedade importante de um conjunto de itens é o contador de suporte dado por σ(X), o qual se refere ao número de transações que contém um determinado conjunto de itens. Matematicamente, o contador de suporte para um conjunto de itens A pode ser declarado da seguinte maneira: $\sigma(\mathrm{X}) = |\{\ t_i \mid \mathrm{X} \subseteq t_i,\ t_i \in \mathrm{D}\}|$ (AGRAWAL; IMIELINSKI; SWAMI, 1993).

O valor de uma regra de associação pode ser medido em termos de seu suporte e confiança (TAN; STEINBACH; KUMAR, 2009). O suporte determina a frequência na qual uma regra é aplicável a um determinado conjunto de dados, dado pela Equação (5).

$$S(A \Rightarrow B) = \frac{\sigma(A \bigcup B)}{|D|} \tag{5}$$

A confiança determina a frequência na qual os itens em B aparecem em transações que contenham A, dado pela Equação (6).

$$C(A \Rightarrow B) = \frac{\sigma(A \bigcup B)}{\sigma(A)} \tag{6}$$

2.4 Classificação de Dados

Nos últimos anos, em razão do aumento da disponibilidade de documentos em formato digital e a consequente necessidade de organizá-los, despertou-se um interesse na área de classificação automática (SEBASTIANI, 2002). Classificação é a tarefa de organizar objetos em categorias pré-definidas (TAN; STEINBACH; KUMAR, 2009). Classificação, também

referenciada como categorização, tem o objetivo de aplicar rótulos aos dados correspondentes às categorias de forma automática (SEBASTIANI, 2002).

A classificação é um processo dividido em duas etapas: primeiro, com base em um conjunto de dados rotulados, é construído um modelo de classificação, o qual descreve as classes predefinidas. Na segunda etapa, o modelo construído é aplicado ao conjunto de teste, que consiste de registros com classes desconhecidas (HAN; KAMBER; PEI, 2011). Em decorrência da necessidade de dados rotulados serem fornecidos inicialmente, este processo é conhecido como aprendizagem supervisionada, ou seja, para construir a função de treinamento um conjunto de dados é fornecido como entrada (chamado de conjunto de treino). Esse conjunto de dados é composto de pares de objetos e classes, indicando a classe a que cada objeto pertence, de acordo com especialistas humanos (BAEZA-YATES; RIBEIRO-NETO, 2011).

Inicialmente, é estimada a porcentagem dos dados do conjunto de teste que foram classificados corretamente pelo classificador. O conjunto de dados de treino é uma amostra de dados independente do conjunto de dados de teste, gerados aleatoriamente. Se a porcentagem dos dados classificados é aceitável, o mesmo pode ser utilizado para classificar dados não rotulados (HAN; KAMBER; PEI, 2011).

Exemplos de algoritmos de classificação incluem classificadores de árvore de decisão, classificadores baseado em regras, redes neurais, máquinas de vetor de suporte e classificadores Bayes simples (HAN; KAMBER; PEI, 2011). Cada técnica emprega um algoritmo de aprendizagem para identificar um modelo que seja mais apropriado para o relacionamento entre o conjunto de atributos e o rótulo da classe dos dados de entrada.

Uma árvore de decisão é um método de classificação, a qual utiliza um conjunto de treino para construir as regras organizadas como caminhos em uma árvore. Cada nó interno desta árvore denota um teste em um atributo, cada ramificação representa um resultado do teste, e os nós folhas representam as classes. O nó mais alto na árvore é o nó raiz (BAEZA-YATES; RIBEIRO-NETO, 2011). Para realizar a classificação de uma instância ainda não classificada, os valores dos atributos das instâncias são testados na árvore de decisão. Um caminho é traçado com base na raiz até um nó folha, o qual mantém, a previsão de classe para a instância. As árvores de decisão podem facilmente ser convertidas em regras de classificação (HAN; KAMBER; PEI, 2011).

Classificadores Bayesianos, baseados no Teorema de Bayes, são exemplos de classificadores estatísticos. Eles podem prever a probabilidade de associação de classe com a possibilidade de uma determinada instância pertencer a uma classe particular. Classificadores Bayesianos assumem que o efeito do valor de um atributo é independente do valor de outros atributos, o que simplifica a computação envolvida e torna esse classificador simples (HAN; KAMBER; PEI, 2011).

O classificador *Suport Vector Machines* (SVM) é um método de classificação supervisionado, introduzido por Cortes e Vapnik (1995). Este método consiste em uma técnica computacional de aprendizado para problemas de reconhecimento de padrão. Introduzida por meio da teoria estatística de aprendizagem por Vapnik, essa classificação é baseada no princípio de separação ótima entre classes, tal que se as classes são separáveis, ele tenta encontrar a maior margem para separar diferentes classes de dados, de forma

que a solução escolhida separa o máximo as classes (CORTES; VAPNIK, 1995).

2.5 Classificação de Produtos

O serviço de comparar preços é muito utilizado por consumidores de lojas de comércio eletrônico. *Sites* que oferecem esse serviço recebem ofertas de produtos com base em diversas lojas de comércio eletrônico e agrupam as ofertas do mesmo produto. As ofertas categorizadas são utilizadas em várias tarefas, incluindo a navegação na taxonomia de produtos e a correspondência de ofertas de produtos a um catálogo de produtos. A fim de proporcionar uma navegação nos produtos e uma possível experiência de compras de alta qualidade aos consumidores, é de fundamental importância um classificador de ofertas de produtos eficiente.

Os *sites* de compras, normalmente mantém os produtos organizados hierarquicamente em categorias, como mostra a Figura 3 (a). Eles, também, mantém um catálogo de produtos que contém especificações detalhadas para cada produto, organizados em uma estrutura de pares de *atributo-valor*, como mostra a Figura 3 (b).

Existem trabalhos propostos na literatura para diferentes tipos de classificação de produtos, dentre eles, os trabalhos que associam as ofertas de produtos a um catálogo de produtos (KANNAN et al., 2011; NGUYEN et al., 2011). Outros que realizam a classificação de produtos em algum nível da hierarquia, como por exemplo nas classes "Roupas", "Componentes" e "Impressoras" (CORTEZ et al., 2011) e outro que realiza a classificação em um nível de ofertas produtos, por exemplo "HP Photosmart C4680" (KÖPCKE et al., 2012). Para o serviço de comparar preços é necessária

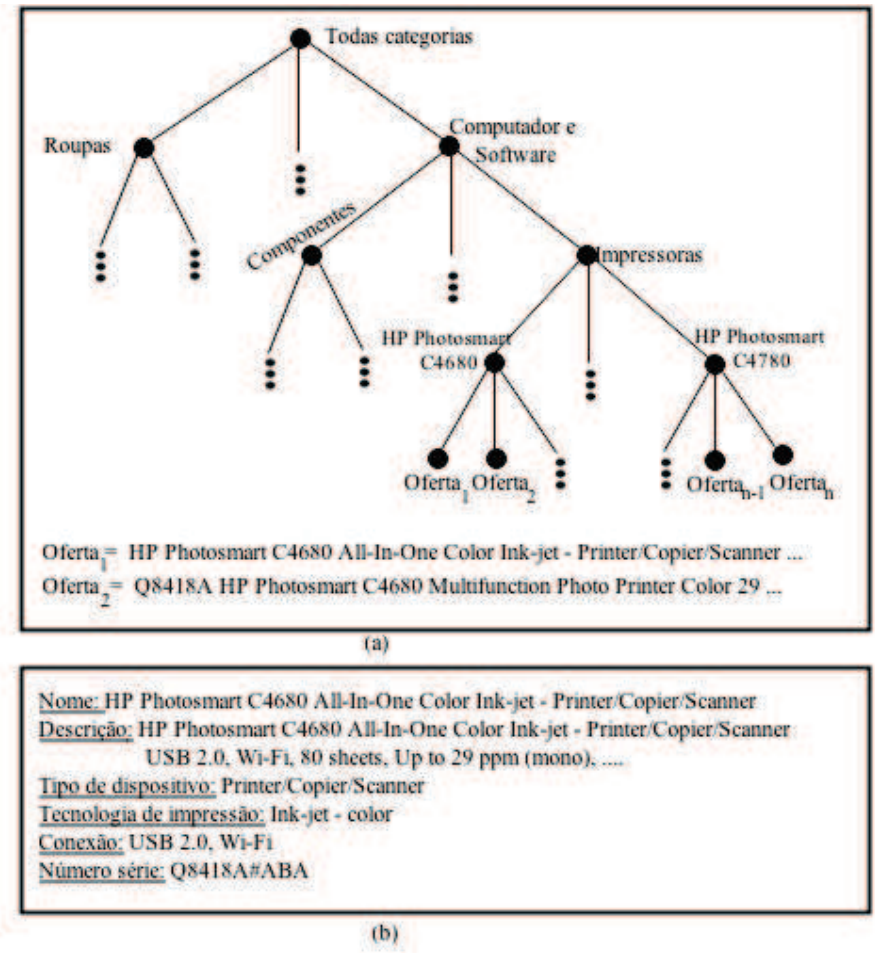

Figura 3 (a) Exemplo de uma hierarquia de produtos e (b) exemplo de pares *atributo-valor* em um catálogo de produtos.

a classificação de ofertas de produtos, que é o foco deste trabalho. Essa classificação torna-se muito mais desafiadora em razão do grande número de classes existentes.

2.6 Trabalhos Relacionados

Dada a abrangência de produtos comercializados na Web, várias técnicas e medidas de classificação foram sugeridas na tentativa de melhorar a correspondência de produtos ofertados nos *shoppings online*. Em relação a associar ofertas de produtos a um catálogo de produtos, Kannan et al. (2011) descreveram seu sistema, utilizado pela máquina de busca Bing para combinar as ofertas recebidas pelo Bing Shopping com seu catálogo de produtos. Neste trabalho, o objetivo é combinar as ofertas recebidas em forma de descrição textual com um catálogo de produtos descritos em forma de registros estruturados. Eles utilizaram uma abordagem probabilística para encontrar o produto no catálogo o qual possui maior equivalência para a oferta.

Outro trabalho desenvolvido para associar ofertas de produtos a um catálogo de produtos foi Nguyen et al. (2011), em que utilizaram também dados do Bing Shopping em seus experimentos, cujo objetivo foi identificar novos produtos no conjunto de ofertas não estruturadas e adicioná-los ao catálogo de produtos cujos atributos são estruturados. Neste trabalho, foi desenvolvida uma nova técnica escalável para o esquema de agregação de produtos que aproveita o conhecimento anteriormente obtido entre as associações das ofertas e produtos.

Relacionado a classificar produtos em uma determinada categoria da hierarquia de produtos apresentada na Figura 3, Cortez et al. (2011) apresentaram uma proposta para a categorização de produtos ofertados em shoping online, utilizando, também, uma abordagem probabilística para o problema de classificação. Os seus experimentos foram realizados em bases

de dados reais do Shopping Uol[2]. Os seus resultados não são comparáveis diretamente aos resultados obtidos neste trabalho, em virtude do nível de classificação ser diferente. Em seu trabalho, a classificação foi feita a nível de produtos em determinada categoria. E neste trabalho, a classificação foi feita em nível de oferta de cada produto. Nos experimentos realizados, foram utilizados *datasets* extraídos da mesma fonte que eles, porém a quantidade de classes é muito maior, proporcionando mais desafios no processo de classificação de ofertas de produtos.

Classificação de ofertas de produtos em categorias no nível de produtos Köpcke et al. (2012) trataram o problema de combinar ofertas de produtos com ofertas de produtos, ou seja, dados não estruturados combinados com dados não estruturados. Neste, a proposta é usar uma nova abordagem para identificar duplicatas de ofertas de produtos em *dataset* distintos. Sua abordagem preprocessa das ofertas de produtos para extrair os códigos dos produtos para a identificá-los e fazer a distinção de produtos similares. Para extração dos códigos, foram utilizadas expressões regulares que identificam os possíveis códigos candidatos presentes nos títulos e descrições dos produtos. E, para validação, utilizaram a Web como fonte de informação, em que cada código candidato foi submetido à máquina de busca onde é verificada com corretude pela relação dos resultados contendo o fabricante correspondente. Essa abordagem depende do código do fabricante presente nas descrições. Neste trabalho, não foram desenvolvidas estratégias específicas para extrair o código de produto, porém se o código aparece nas descrições ele é extraído naturalmente e gera regra contribuindo para melhorar a qualidade do método proposto.

[2]http://shopping.uol.com.br/

Outros tipos de trabalhos, Sato et al. (2011) categorizaram as páginas de produtos em função de sua informação. Utilizaram o classificador Naive Bayes e fizeram experimentos com quatro tipos de recursos para classificá-los: todas as palavras do título das páginas do produto, os nomes (substantivos) extraídos dos títulos, todas as descrições das páginas dos produtos e os nomes extraídos delas. Seus experimentos mostraram que as páginas de produtos podem ser mais corretamente classificados em função apenas dos nomes dos títulos das páginas. Caracterizando o relacionamento com este trabalho, o qual utiliza as descrições das ofertas de produtos para a classificação, essa descrição utilizada é o título da página do produto.

Ghani et al. (2006) desenvolveram uma estratégia de extrair pares atributo e valor de descrições textuais de produtos utilizando uma abordagem supervisionada, a qual requer dados rotulados para a fase de treinamento do classificador. Não diretamente relacionado a este trabalho, porém essa tarefa de extrair atributos poderia associar ao método proposto com o objetivo de melhorar a descrição do produto, removendo atributos menos interessantes para a classificação ou até mesmo contribuindo para a remoção de *tokens* ruidosos.

O trabalho proposto utiliza regras de associação, esta técnica é capaz de encontrar relacionamentos entre um conjunto de itens em um *dataset* (AGRAWAL; IMIELINSKI; SWAMI, 1993). O uso de regras de associação já foi aplicado a um problema relacionado, que é a desambiguação de nomes de autores, que afeta a qualidade dos serviços em bibliotecas digitais. Veloso et al. (2012) desenvolveram soluções para desambiguar nomes de autores por meio de regras de associação, usando dados como nomes de co-autor, título do trabalho e local de publicação, que associam as carac-

terísticas da citação ao nome do autor. Seus experimentos demonstraram que as soluções propostas foram eficazes e práticas, superando soluções supervisionadas propostas na literatura. A utilização de regras de associação também pode ser encontrada para classificar documentos em Veloso et al. (2006). Esses trabalhos são similares ao método proposto, o qual utiliza uma abordagem de aprendizagem baseada no uso de regras de associação para fazer a correspondência de ofertas de produtos.

Pereira, Silva e Esmin (2014) apresentaram um método para a desambiguação de títulos de veículos de publicação usando regras de associação. O método proposto é uma extensão deste trabalho, em que uma variante do método deles foi adaptada neste trabalho para classificar ofertas de produtos de lojas de comércio eletrônico, foi introduzido o uso do suporte mínimo para gerar regras na fase de treinamento e um novo esquema de votação em que é escolhida a melhor regra na fase de teste. Foram propostas e avaliadas novas estratégias alternativas ao algoritmo em sua versão básica e vários aspectos na aplicação do produto.

Embora existam diversos trabalhos propostos na literatura, os resultados mostram que ainda existe muito o que melhorar nas estratégias de agregação de ofertas de produtos. Köpcke et al. (2012) afirmam que, apesar da utilização de pequenos conjuntos de dados para a realização dos experimentos, soluções correntes podem chegar a apenas entre 30 e 70% de *F-measure*, medida de qualidade muito utilizada para avaliar o desempenho de sistemas (BAEZA-YATES; RIBEIRO-NETO, 2011). De acordo com Kannan et al. (2011), para que um sistema de agregação de ofertas de produtos com catálogos seja viável e colocado em funcionamento é necessário alcançar ao menos 85% de precisão, medida de qualidade também

muito utilizada na avaliação de sistemas de classificação (BAEZA-YATES; RIBEIRO-NETO, 2011).

3 MÉTODO PROPOSTO PARA CLASSIFICAÇÃO DE OFERTAS DE PRODUTOS

3.1 Formulação do Problema

A tarefa de classificação de ofertas de produtos pode ser formulada da seguinte maneira. Seja $P = \{p_1, p_2, ..., p_{|P|}\}$ um conjunto de ofertas de produtos. Cada oferta de produto p_i possui uma lista de atributos tais como: descrição, preço do produto e o nome da loja que está ofertando. Seja $C = \{c_1, c_2, ..., c_{|C|}\}$ um conjunto de classes com seus respectivos rótulos, neste trabalho, uma classe é um produto, por exemplo, a descrição *HP Officejet Pro K5400* é descrição de uma classe de produto, representada por um rótulo que identifica unicamente essa classe. O objetivo é produzir uma função de classificação que mapeia cada oferta de produto, p_i, em uma das classes pré-definidas do conjunto C.

O método proposto para resolver o problema de classificação usa a técnica de aprendizagem de máquina supervisionada. Nesse caso, é dado um conjunto de dados como entrada, denominado dados de treinamento, denotado por D, que consiste de exemplos de instâncias das quais a classe correta da oferta do produto é conhecida. Cada instância gera um conjunto de m *features* (características) $\{f_1, f_2, ..., f_m\}$. Neste trabalho, essas *features* são *tokens* (palavras) extraídos dos atributos (*strings*), neste caso, da descrição da oferta do produto. Os dados de treinamento são usados para produzir um modelo de aprendizagem utilizando regras de associação, o qual relaciona as *features* nos dados de treinamento com a classe correta da oferta de produto.

O conjunto de dados de teste, denotado por T, para o problema de classificação, consiste em um conjunto de ofertas de produtos para os quais as *features* são conhecidas enquanto a classe correta da oferta de produto é desconhecida. O modelo de aprendizagem, que é uma função que mapeia um conjunto de *features* $\{f_1, f_2, ..., f_m\}$ para a classe $c_i \in C$ é usado para predizer a classe correta da oferta de produto no conjunto de teste.

A função de aprendizagem usa um classificador associativo, que explora as associações entre os *tokens* nas ofertas de produto que identifica unicamente cada classe. Tais associações são descobertas usando regras da forma $X \rightarrow c_i$, onde $X \subseteq \{f_1, f_2, ..., f_m\}$ e $c_i \in C$. Por exemplo, a descrição da oferta de produto *Officejet J3680 All-in-One Printer, Fax, Scanner, Copier, HEWCB071A*, que pertence à classe c_1, pode produzir duas regras de associação $\{J3680\} \rightarrow c_1$ e $\{HEWCB071A\} \rightarrow c_1$, que indicam que o conjunto de *tokens* $\{J3680\}$ e o conjunto $\{HEWCB071A\}$, ambos identificam unicamente as ofertas de produtos da classe c_1.

Para produzir regras de associação que identificam unicamente cada classe, o classificador associativo aprende somente as regras que possuem 100% de confiança. De acordo com Agrawal e Srikant (1994), uma regra $X \rightarrow Y$ contido no conjunto de dados D tem confiança c, se $c\%$ das instâncias em D que contém X também contém Y. Então, o modelo gerado não contém regras $X \rightarrow c_i$ e $X \rightarrow c_j$ tal que $i \neq j$. Esta estratégia não é perfeita, pois pode não produzir regras para todas as classes. Tal situação ocorre quando os conjuntos de *tokens* em toda oferta de produto de uma classe estão contidos em algum conjunto de *tokens* de classes distintas. Nesse caso, nenhuma regra é gerada para essa classe. Buscando resolver tal situação, na fase de teste, para predizer a classe da instância a qual nenhuma regra é encontrada

no modelo de aprendizagem, o método proposto, utiliza outra estratégia que será explicada posteriormente.

3.2 Fase de Treinamento

Os dados de treinamento D podem ser fornecidos por uma fonte de dados como o banco de dados de um shopping online, contendo descrições de ofertas de produtos. Nos dados de entrada, existem uma ou mais instâncias para cada classe distinta. Cada instância de entrada em D é uma *string* descrevendo a oferta de produtos para a classe c_i, representado por $d_{j,i}$, descrição j para a classe c_i. Seja $R_{c_i}^{d_{j,i}}$ o conjunto de regras $X \rightarrow c_i$ onde $X \subseteq d_{j,i}$ ($d_{j,i}$ contém todas as *features* em X). Isto é, $R_{c_i}^{d_{j,i}}$ é composto de regras que preveem a classe c_i originadas da descrição da oferta de produto $d_{j,i}$. Seja R_{c_i} o conjunto de regras que preveem a classe c_i, e seja R o conjunto de todas as regras geradas pelo modelo de aprendizagem. Então, $R_{c_i}^{d_{j,i}} \subseteq R_{c_i} \subseteq R$.

O Algoritmo 1 mostra os passos da fase de treinamento. Ele recebe como entrada um conjunto de ofertas de produtos em que as classes são conhecidas e um suporte mínimo para gerar as regras, e retorna um conjunto de regras associativas que possuem 100% de confiança e um suporte mínimo para predizer as classes das ofertas de produtos. O primeiro passo do Algoritmo 1 (Linhas 1–6) insere *tokens* distintos das instâncias na estrutura de índice invertido (BAEZA-YATES; RIBEIRO-NETO, 2011). Essa estrutura é composta por pares chave-valor, onde a chave é um *token* e o valor é uma lista de ocorrência deste token, contendo, em cada posição, a classe c_i e a identificação específica da descrição j. A construção desta estrutura é executada pela função *InsertInvertedIndex*.

A função *Tokenize* divide a descrição em *tokens*. Antes de dividir em tokens, cada *string* é preprocessada, removendo sinais de pontuação, símbolos como ()[]{}, *stopwords* (artigos, preposições e conjunções) e convertendo letras para minúsculo. *Strings* com hífen (-) e barra (/) são processadas diferentemente. O processamento usa (-) ou (/) em duas situações, como separador e como um agregador obtendo-se três ou mais tokens. Por exemplo, para processar a *string* "DSC-W730"os seguintes *tokens* serão gerados: "dsc", "w730"e "dscw730".

O segundo passo do Algoritmo 1 (Linha 7–23) é um processo iterativo para criar regras de associação. A função *GenItemSets* gera conjuntos de itens de tamanho k (k *tokens*) usando os conjuntos de tamanho com $k-1$ obtidos da iteração anterior. Agrawal e Srikant (1994) apresentam um algoritmo para combinar o conjunto de itens. Cada conjunto de tamanho k é pesquisado no índice invertido, e se ele ocorre somente em uma classe e possui o suporte mínimo, uma regra é criada contendo o *k-itemset* como antecedente e o identificador da classe como consequente (Linha 12–17).

A função *SizeOccurrenceList* retorna o número de classes distintas em que um *k-itemset* aparece em alguma descrição de oferta de produto. Essa função pesquisa no índice invertido usando cada *token* em um *k-itemset* como chave e recupera sua lista de ocorrências. Quando $k > 1$, ela faz a operação de interseção na lista de ocorrências de cada *token* para encontrar o resultado. Se o tamanho do resultado é 1, a regra formada pelo *k-itemset* e sua classe possui 100% de confiança. O algoritmo também verifica o suporte mínimo. O suporte é definido da seguinte forma: a regra $X \rightarrow c_i$ possui suporte s em D se $s\%$ das instâncias da classe c_i em D contém X. A

Algoritmo 1: Fase de Treinamento

Require: Exemplos para Treinamento D
Require: Suporte Mínimo ms
Ensure: Conjunto de regras R

```
1:  for each instance d_{j,i} ∈ D do
2:    S_0 ← Tokenize(d_{j,i})
3:    for each token tk ∈ S_0 do
4:      InsertInvertedIndex(tk, j, c_i)
5:    end for
6:  end for
7:  R ← ∅
8:  for each instance d_{j,i} ∈ D do
9:    S_0 ← Tokenize(d_{j,i})
10:   m ← Length(S_0)
11:   for (k ← 1; k ≤ m; k++) do
12:     S_k ← GenItemSets(k, S_{k-1})
13:     for each k-itemset it ⊆ S_k do
14:       if SizeOccurrenceList(it) = 1 then
15:         // Regra com 100% de confiança
16:         if SupportRule(it → c_i) ≥ ms then
17:           InsertRule(it → c_i, R)
18:         end if
19:         RemoveItemSet(it, S_k)
20:       end if
21:     end for
22:   end for
23: end for
24: return R
```

função *SupportRule* retorna o suporte da regra. Se a regra também atende ao suporte mínimo, ela é inserida no conjunto de regras pela função *InsertRule*.

Se um *k-itemset* forma uma regra com 100% de confiança então algum *l-itemset*, $l > k$, que inclui este *k-itemset* também forma. Então, o algoritmo utiliza uma estratégia de poda, evitando combinar esse *k-itemset* na próxima iteração realizado pela função *RemoveItemSet* (Linha 19).

Exemplo 1. Na Tabela 2 ilustra-se um exemplo de dados de treinamento, onde a classe c_1 possui 4 instâncias. Na Tabela 3 mostram-se as regras geradas com base nos dados de treinamento da Tabela 2. As regras estão organizadas de acordo com o suporte mínimo usado para gerá-las. Observe que quando se aumenta o suporte, o número de regras é reduzido, tentando-se eliminar os *tokens* que não contribuem para identificar as ofertas de produtos. Note que não existe nenhuma regra para a classe c_2, uma vez que os *tokens* da Instância #5 formam um subconjunto dos tokens da Instância #6 da classe c_3. Essas duas instâncias são de classes distintas porque uma é a impressora *wirelles* e a outra não.

Tabela 2 Um exemplo de dados de treinamento

Classe	**#Inst.**	**Descrição da oferta de produto**
c_1	1	HP Officejet J3680 All-in-One Printer, Fax, Scanner, Copier
	2	Officejet J3680 All-in-One Printer, Fax, Scanner, Copier, HEWCB071A
	3	HP Officejet J3680 All-in-One - multifunction (fax / copier / Scanner)
	4	HEWCB071A HP Officejet J3680 All-in-One Printer CB071A Hewlett-Packard
c_2	5	HP Officejet Pro 8000 Printer
c_3	6	Hewlett Hp Officejet Pro 8000 Wireless Printer Cb9297a#b1h
c_4	7	HP Officejet Pro K5400 Color Printer

3.3 Fase de Teste

Os dados de teste T são compostos de um conjunto de descrições de ofertas de produtos. Pelo Algoritmo 2 mostram-se os detalhes da fase de teste para predizer a classe da descrição $d \in T$. Primeiro d, de tamanho

Tabela 3 Regras geradas com base nos dados de treinamento da Tabela 2

Classe	**#Instância**	**Regras geradas**	
		Sem suporte	**Suporte de 50%**
c_1	1	$fax \rightarrow c_1$	
		$allinone \rightarrow c_1$	
		$packard \rightarrow c_1$	$fax \rightarrow c_1$
		$multifunction \rightarrow c_1$	$allinone \rightarrow c_1$
		$j3680 \rightarrow c_1$	$j3680 \rightarrow c_1$
	2	$hewlettpackard \rightarrow c_1$	$copier \rightarrow c_1$
	3	$copier \rightarrow c_1$	$hewcb071a \rightarrow c_1$
	4	$cb071a \rightarrow c_1$	$scanner \rightarrow c_1$
		$hewcb071a \rightarrow c_1$	
		$scanner \rightarrow c_1$	
c_2	5		
c_3	6	$cb9297a\#b1h \rightarrow c_3$	$cb9297a\#b1h \rightarrow c_3$
		$wireless \rightarrow c_3$	$wireless \rightarrow c_3$
		$hewlett, pro \rightarrow c_3$	$hewlett, pro \rightarrow c_3$
		$8000, hewlett \rightarrow c_3$	$8000, hewlett \rightarrow c_3$
c_4	7	$k5400 \rightarrow c_4$	$k5400 \rightarrow c_4$
		$color \rightarrow c_4$	$color \rightarrow c_4$

m, é tokenizado e seus *k-itemsets*, $1 \leq k \leq m$, são gerados por um processo iterativo (Linhas 1–5). Segundo, cada *itemset* é pesquisado (*matched*) entre os antecedentes das regras R no modelo de aprendizagem. Todas as regras em que os antecedentes casam com algum *itemset* formam o conjunto de regras candidatas, R_d (Linhas 6–10). Terceiro, utilizando um esquema de votação, o consequente de cada regra do conjunto de regras candidatas R_d é contando, e se houver uma classe $c_i \in C$, com a maior contagem, essa é a escolhida como a classe para a descrição da oferta de produto d (Linhas 11–17).

Em caso de empate na votação para todos os *itemsets* ou nenhuma regra é encontrada em R cujo antecedente casa com o *itemset* de d, então é usada uma função de similaridade, por exemplo *Jaccard* ou Cosseno, para escolher a classe de d. Neste caso, cada descrição de oferta de produto

utilizada na fase de treinamento é comparada com a *string* d, e a classe correspondente à *string* com a maior similaridade é escolhida como a classe da descrição da oferta de produto d (Linha 19).

Algoritmo 2: Fase de Teste

Require: R, D, $d \in T$
Ensure: A classe predita de d

1: $S_0 \leftarrow$ Tokenize(d)
2: $m \leftarrow$ Length(S_0)
3: **for** ($k \leftarrow 1; k \leq m; k++$) **do**
4: $\quad R_d \leftarrow \emptyset$
5: $\quad S_k \leftarrow$ GenItemSets(k, S_{k-1})
6: $\quad$ **for each** k-itemset $it \subseteq S_k$ **do**
7: $\quad\quad$ **for each** $X \rightarrow c \subset R$ such that $it = X$ **do**
8: $\quad\quad\quad R_d \leftarrow R_d \cup X \rightarrow c$
9: $\quad\quad$ **end for**
10: $\quad$ **end for**
11: $\quad$ **for each** $r \in R_d$ in the form $X \rightarrow c$ **do**
12: $\quad\quad c.count++$
13: $\quad$ **end for**
14: $\quad pc \leftarrow c_i$ such that $c_i.count > c_j.count \; \forall j \neq i$
15: $\quad$ **if** $pc \neq \emptyset$ **then**
16: $\quad\quad$ return pc // a classe predita de d
17: $\quad$ **end if**
18: **end for**
19: return PredictBySimilarity(D, d) // empate na votação ou nenhuma regra encontrada

Exemplo 2. Na Tabela 4 ilustra-se um exemplo de três instâncias nos dados de teste, e as regras encontradas com base nos dados de treinamento da Tabela 3, que correspondem com os *itemsets* gerados pelas strings de teste. Para a Instância #1, quando o $k = 1$ todas as regras no modelo de treinamento indicam c_1 como a classe correta, e por votação ela é a classe escolhida (Linhas 14–16 do Algoritmo 2). Para a Instância #2, quando $k = 1$

Tabela 4 Um exemplo de dados de teste e as regras da Tabela 3 que combinam com os *itemsets* gerados pela instância de teste.

#Inst.	Instância de Teste	Regras encontradas no modelo	
		Sem suporte	**Suporte de 50%**
1	Hewlett-Packard - HP Officejet J3680 All-in-One	$hewlett\,packard \rightarrow c_1$ $allinone \rightarrow c_1$ $packard \rightarrow c_1$ $j3680 \rightarrow c_1$	$allinone \rightarrow c_1$ $j3680 \rightarrow c_1$
2	Hewlett-Packard OfficeJet Pro K5400 Color Inkjet Printer	$hewlett\,packard \rightarrow c_1$ $packard \rightarrow c_1$ $k5400 \rightarrow c_4$ $color \rightarrow c_4$ $hewlett, pro \rightarrow c_3$	$k5400 \rightarrow c_4$ $color \rightarrow c_4$
3	HP OfficeJet Pro 8000 Printer		

e o suporte mínimo igual a 0%, ocorre empate entre as regras das classes c_1 e c_4. Neste caso, o algoritmo não é capaz de classificar a instância de teste usando o $k = 1$ e gera os *itemsets* de tamanho $k = 2$. Para este tamanho, existe somente a regra $hewlett, pro \rightarrow c_3$, que é uma classificação incorreta para essa instância, que não pertence à classe c_3. No entanto, quando o suporte mínimo é igual a 50%, a classe c_4 é corretamente escolhida pelo algoritmo. E para a Instância #3, não existe regra no modelo de treinamento que combina com essa *string* para todos os *k-itemsets*. Nesse caso, a decisão é dada por uma medida de similaridade, como o Cosseno (Linha 19 do Algoritmo 2).

3.4 Alternativas para o Algoritmo Básico

Em adição ao algoritmo básico, composto pelas etapas apresentadas nos Algoritmos 1 e 2, foram propostas e avaliadas duas alternativas, que se constituem de pequenas variações no algoritmo básico.

Alternativa 1: limitar o número de *tokens* nas *strings* de descrição de oferta de produto. A hipótese é que regras mais interessantes sejam formadas pelos *tokens* que aparecem mais no início das *strings*, os quais usualmente descrevem o título e o código das ofertas do produto. Os *tokens* restantes usualmente descrevem suas características técnicas, que são menos importantes para identificar a oferta do produto. Algumas vezes, códigos implícitos também são colocados no final das *strings*. Desta maneira, pode-se usar somente um número fixo de *tokens* do início e alguns do final das *strings*. Tal estratégia também reduz o número de *tokens* a serem combinados para formar os *itemsets*, o que constitui uma estratégia de poda apresentada na Seção 3.5. Nos experimentos realizados, foram utilizados os 10 primeiros e os 3 últimos *tokens* de cada instância de entrada, em ambas fases treino e teste.

Alternativa 2: escolher a classe da regra contendo o *token* que ocorre mais no início da *string* de descrição de oferta do produto, em casos de empate com outras regras na fase de teste do algoritmo para os *itemsets* de tamanho igual a 1. A hipótese é que se a instância de teste contém um código implícito, tal código ocorre na maioria dos casos no início da string. Por exemplo, considere que o modelo treinado contém as regras $\{5200tn \rightarrow c_x\}$ e $\{prints \rightarrow c_y\}$. Considere também a instância *hp laserjet 5200tn printer trade compliant tp to 35 ppm prints* na fase de teste, cuja classe correta é c_x, e que para os *itemsets* de tamanho igual a 1 casam somente os *itemsets* $\{5200tn\}$ e $\{prints\}$ com as regras treinadas. Tal instância contém um código implícito $\{5200tn\}$. Para esta estratégia, o desempate entre as classes c_x e c_y seria resolvido em favor da c_x porque o *token* 5200tn ocorre antes do *token* *prints* na instância de teste.

3.5 Complexidade Computacional

A complexidade computacional de tempo do método proposto na fase de treino é dominada pelo número de *strings* de descrições de ofertas de produtos para serem treinadas e pelo número de *tokens* nestas *strings*. No pior caso, todos os tokens em cada *string* precisam ser combinados para formar os itemsets. Seja n o número de *strings* de descrição de ofertas de produtos e m a média de *tokens* presentes nestas *strings*. Então, a complexidade computacional de tempo do método é no pior caso $O(n * 2^m)$.

Com o objetivo de reduzir o tempo de execução, foram adotadas algumas estratégias de poda. A primeira delas é para evitar a geração de regras cujo antecedente é um super conjunto de um antecedente de outra regra, isto é, o algoritmo evita continuar combinando um *k-itemset* que gera regra com 100% de confiança (Linha 19 do Algoritmo 1). Por exemplo, suponha que uma instância seja formada pelos *tokens* $\{A,B,C,D,E\}$. Na Tabela 5 mostram-se os *itemsets* podados quando o 1-*itemset* $\{D\}$ forma uma regra. Os *itemsets* de tamanhos 2 a 5 em negrito não serão gerados pelo algoritmo.

A segunda estratégia de poda usa o suporte mínimo da regra. Um *k-itemset* que não ocorre em um número mínimo de descrições de ofertas de produtos de alguma classe é removido do *k-itemset*, evitando combiná-los para formar os *k+1-itemsets*. A função *GenItemSets* do Algoritmo 1 (Linha 12) pode ser modificada para incluir apenas *itemsets* que alcançam o suporte mínimo *ms* para pelo menos uma classe.

A terceira estratégia de poda limita o tamanho k dos *itemsets* gerados. A condição de parada $k \leq m$, na Linha 11 do Algoritmo 1, é alterada para $k \leq max$, onde max é o tamanho máximo de *itemsets* a serem gerados.

Tabela 5 Exemplo de estratégia de poda que não gera os *itemsets* em negrito quando o 1-*itemset* $\{D\}$ forma regra

Tamanho do k-itemset				
k=1	k=2	k=3	k=4	k=5
A	AB	ABC	**ABCD**	**ABCDE**
B	AC	**ABD**	ABCE	
C	**AD**	ABE	**ABDE**	
D	AE	**ACD**	**ACDE**	
E	BC	ACE	**BCDE**	
	BD	**ADE**		
	BE	**BCD**		
	CD	BCE		
	CE	**BDE**		
	DE	**CDE**		

Foi observado, nos experimentos, que as regras mais interessantes são encontradas utilizando os *itemsets* menores, principalmente quando as ofertas de produtos contêm um identificador implícito. Na Figura 4 ilustra-se a qualidade da classificação por meio do resultado da medida micro-F1 e o percentual de ofertas de produtos classificadas utilizando somente as regras (mais detalhes na Seção 5) de acordo com a variação do tamanho máximo do *itemset* para o *dataset Uol-eletronic*. Utilizando somente *itemsets* de tamanho um, o método proposto obtém o melhor resultado da classificação, porém o percentual de ofertas de produtos classificadas é o menor em razão das poucas regras com 100% de confiança encontradas. Aumentando-se o tamanho máximo do *itemset*, o número de regras encontradas aumenta, assim como o percentual de ofertas de produtos classificadas, porém a qualidade da classificação diminui em decorrência de regras que podem levar a uma classificação incorreta.

A quarta estratégia de poda limita o número de *tokens* nas *strings* de descrição da oferta de produto. Foi observado que as regras mais interessan-

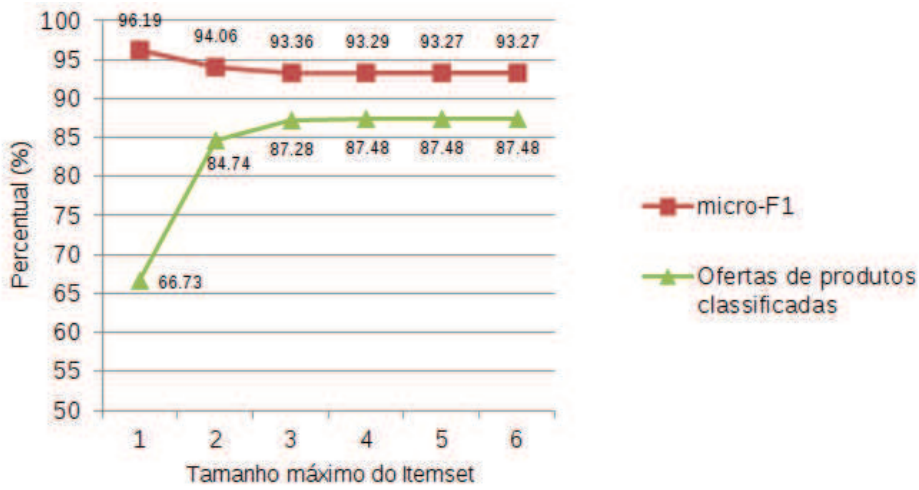

Figura 4 Evolução dos resultados da micro-F1 e o percentual de ofertas de produtos classificadas utilizando apenas as regras de acordo com a variação do tamanho máximo do *itemset*.

tes são encontradas usando os primeiros *tokens* nas *strings*, que usualmente descrevem o título da oferta de produto. Os *tokens* restantes usualmente descrevem suas características técnicas, que são menos importantes para identificar a oferta de produto. Algumas vezes, identificadores implícitos também são encontrados no final da *string*. Desta forma, pode- se utilizar apenas um número fixo de *tokens* no início e no final das *strings*. Assim, o número de tokens na string de entrada, (m), pode ser considerado uma constante, e a complexidade computacional de tempo do método torna-se linear, $O(n)$.

A mesma análise de complexidade pode ser realizada na fase de teste do algoritmo, e a terceira e a quarta estratégias de poda também são aplicáveis. Observe que a pesquisa no conjunto de regras R, no modelo de

aprendizagem, pode ser executado em $O(1)$ usando uma tabela *hash* para guardar as regras.

4 CONFIGURAÇÃO DOS EXPERIMENTOS

Neste capítulo são apresentados os software e hardware utilizados, as bases de dados de experimentos, as métricas de avaliação e os *baselines* utilizados.

4.1 Software e hardware utilizados

Para a realização dos experimentos, foram utilizadas duas máquinas com arquiteturas distintas. A *Máquina 1*, com arquitetura x86_64, com processador Intel(R) Core(TM)2 Quad de 2.66GHz, 4GB de memória RAM e sistema operacional Ubuntu 12.04.4 LTS 64-bit. E a *Máquina 2*, com a arquitetura i686, com processador Intel(R) Xeon(R) E5620 2.40GHz, 4GB de memória RAM e o sistema operacional Ubuntu 12.04.5 LTS 32-bit.

Para o desenvolvimento do método proposto e do coletor de dados para construir as bases de dados iniciais foi utilizada a linguagem de programação Java, utilizando a JDK 7 (Oracle) 1.7.0.

4.2 Bases de dados *(datasets)*

O método proposto neste trabalho foi avaliado utilizando-se diversos *datasets* distintos, como descrito a seguir:

Obtido por meio de crawling no site de e-commerce do Shoping UOL. Com base na hierarquia de disponibilização das ofertas, foi construído um xml conforme apresentado na Figura 5.

UOL dataset - Obtido por meio de *crawling* no site de e-commerce do Shoping UOL. Neste site, os dados foram rotulados manualmente por especialistas e as ofertas de produtos estão organizadas hierarquicamente.

Com base na hierarquia de disponibilização das ofertas, foi construído um xml conforme apresentado na Figura 5. Foram coletados dados de produtos que tinham no mínimo duas instâncias de ofertas de produto associadas a elas. Uma instância de oferta de produto contém os seguintes dados: a descrição da oferta do produto, o preço e a loja. Porém, para a execução deste trabalho, utilizou-se apenas a descrição da oferta do produto. Para avaliação do classificador proposto, foram utilizados os produtos e as ofertas de produtos. Na Figura 5, os produtos são identificados pelo campo *productOffer* que representa, neste trabalho, a classe do produto. As instâncias de ofertas de produto são identificadas pelo campo *offerProduct*.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<site>
   <urlSite>http://shopping.uol.com.br</urlSite>
   <category>
      <categoryName>Eletrônicos e Informática</categoryName>
      <class>
         <className>Informática</className>
         <subClass>
            <subClassName>Impressora e Multifuncional</subClassName>
            <productOffer>
                <offerId>i2129771</offerId>
                <offerDescription>Impressora HP Laserjet P1102w Laser</offerDescription>
                <offerLink>/impressora-hp-laserjet-p1102w-laser_2129771.html</offerLink>
                <offerProduct>
                    <prodId>i116675570</prodId>
                    <store>Balão da Informática</store>
                    <prodDesc>Impressora LaserJet Pro P1102, Laser Monocromática, Velocidade Impressão
                    18ppm, Memória 2MB, Conexão USB 2.0, (CE651A)</prodDesc>
                    <realPrice>R$ 424,29</realPrice>
                </offerProduct>
                <offerProduct>
                    <prodId>i1125641908</prodId>
                    <store>HP</store>
                    <prodDesc>Impressora HP LaserJet Pro P1102w Wireless com ePrint</prodDesc>
                    <realPrice>R$ 494,10</realPrice>
               </offerProduct>
            </productOffer>
      </class>
   </category>
</site>
```

Figura 5 Estrutura dos dados obtidos por meio de *crawling* no Shoping UOL.

A coleção utilizada possui um total de 421.989 ofertas de produtos, associados a 101.403 classes. Para avaliar diferentes aspectos do método proposto, esta coleção foi dividida em três *datasets*, como descrito a seguir.

UOL-electronic dataset - Formado pelas seguintes categorias de produtos: celular, telefone, eletrônicos e informática e eletrodomésticos. Neste *dataset*, as ofertas de produtos geralmente possuem um código de produto implícito em sua descrição. O *dataset* contém 9.552 ofertas de produtos associados a 2.218 classes. A média de *tokens* por instância neste *dataset* é de $12,6$ *tokens*, a maior instância possui 35 *tokens* e a menor 2 *tokens*.

UOL-non-electronic dataset - Formado pelas seguintes categorias de produtos: perfumaria e cosméticos, acessórios de moda e joias, brinquedos e jogos, esporte e fitness, e bebês e crianças. As ofertas de produtos nesta categoria geralmente não contam com código de produto implícito em sua descrição. O *dataset* contém 26.640 ofertas de produtos associados a 5.299 classes. Neste *dataset*, a média de *tokens* por instância é de $6,42$ *tokens*, a maior instância possui 28 tokens e a menor 1 *token*.

UOL-book dataset – Formado pela categoria Livros, que também não possui código implícito em sua descrição. O *dataset* contém 385.797 ofertas de produtos associados a 93.886 classes. Neste *dataset*, a média é de $5,72$ *tokens* por instância, a maior possui 33 *tokens* e a menor 1 *token*.

Printer dataset - Composto por descrições de impressoras obtidas por meio de busca no Google Shopping [3]. Este *dataset* foi utilizado por Pereira et al. (2011) e possui 2.167 descrições distintas de impressoras distribuídas em 157 classes, com uma média de $13,8$ descrições por classe. Neste *dataset*, a média de tokens por instância é de $7,5$ tokens, a maior instância possui 16 *tokens* e a menor possui 1 *token*.

[3] https://www.google.com/shopping

Abt-Buy datasets - Composto por descrições de produtos obtidos dos sites Abt [4] e Buy [5]. Estes *datasets* foram utilizados por Köpcke, Thor e Rahm (2010) para avaliar abordagens de resolução de entidades. Os *datasets* contêm respectivamente 1.081 e 1.092 instâncias. Os produtos de um *dataset* foram comparados com os produtos do outro *dataset*, resultando em 1.097 pares de produtos correspondentes utilizados para construir as classes. Uma classe representa o agrupamento dos produtos equivalentes nos *datasets*, resultando em 1.075 classes contendo duas instâncias cada, exceto 16 classes que possuem três instâncias cada. Nestes *datasets*, a média de *tokens* por instância é de 8,6 *tokens*, a maior instância possui 37 *tokens* e a menor 2 *tokens*.

Amazon-GoogleProducts datasets - Composto por descrições de produtos obtidos dos sites Amazon[6] e Google Shopping. Estes *datasets* foram utilizados por Köpcke, Thor e Rahm (2010) para avaliar abordagens de resolução de entidades. Os *datasets* contêm, respectivamente, 1.363 e 3.226 instâncias. Os produtos de um *dataset* foram comparados com os produtos do outro *dataset*, resultando em 1.300 pares correspondentes utilizados para construir as classes. Uma classe representa o agrupamento dos produtos equivalentes nos *datasets*, resultando em 1.105 classes contendo duas instâncias cada, com as seguintes exceções: 115 classes possuem 3 instâncias; 14 classes possuem 4 instâncias; 13 classes possuem 5 instâncias e apenas 1 classe possui 6 instâncias. Neste *dataset*, a média é de 7,28 *tokens* por instância, a maior instância possui 29 *tokens* e a menor 1 *token*.

[4] http://www.abt.com/
[5] http://www.buy.com
[6] http://www.amazon.com/

Na Tabela 6 mostra-se, para cada *dataset* apresentado, o número de ofertas, o número de classes e a média de *tokens* por instância.

Tabela 6 Estatística dos *datasets*.

Dataset	N^o de Ofertas	N^o de Classes	Média de *tokens*
UOL-eletronic	9.552	2.218	12,6
UOL-non-eletronic	26.640	5.299	6,42
UOL-book	385.797	93.886	5,72
Printer	2.167	157	7,5
Abt-Buy	1.097	1.075	8,6
Amazon-GoogleProducts	1.300	1.105	7,28

4.3 Métricas de Avaliação

Para avaliar o classificador proposto, foram utilizadas as métricas macro-média F_1 ($macro-F1$) e micro-média F_1 ($micro-F1$) (BAEZA-YATES; RIBEIRO-NETO, 2011). A medida F_1 é calculada baseada nos valores de precisão e revocação (recall), as quais são calculadas em relação a uma determinada classe c_p.

Precisão é a fração de todas as instâncias associadas à classe c_p pelo classificador que realmente pertence a esta classe, de acordo com o conjunto de teste, definida como

$$P_{(c_p)} = \frac{n_{f,t}}{n_f} \tag{7}$$

onde $n_{f,t}$ é o número de instâncias que ambos, o conjunto de teste e a função de classificação associaram à classe c_p, ou seja, é o número de instâncias que o classificador acertou. E n_f é o número de instâncias associadas à classe c_p pela função de classificação.

Revocação é a fração de todas as instâncias que pertencem à classe c_p de acordo com o teste que foi corretamente associado à classe c_p pelo classificador, definida como

$$R_{(c_p)} = \frac{n_{f,t}}{n_t} \tag{8}$$

onde n_t é o número de instâncias associadas à classe c_p pelo conjunto de teste, ou seja, é o número de instâncias que realmente são da classe c_p.

A medida F_1 equilibra a importância relativa da precisão e revocação, definida como

$$F_{1(c_p)} = \frac{2P_{(c_p)}R_{(c_p)}}{P_{(c_p)} + R_{(c_p)}} \tag{9}$$

A macro-F_1 corresponde à média simples de F_1 sobre todas as classes $|C|$, definida como

$$macro - F_1 = \frac{\sum_{p=1}^{|C|} F_{1(c_p)}}{|C|} \tag{10}$$

A micro-F_1 corresponde ao valor global de F_1 obtida pela precisão e revocação calculadas sobre todas as classes, como a seguir

$$P = \frac{\sum_{c_p \in C} n_{f,t}}{\sum_{c_p \in C} n_f} \qquad R = \frac{\sum_{c_p \in C} n_{f,t}}{\sum_{c_p \in C} n_t} \tag{11}$$

Após obter a precisão e revocação sobre todas as classes Equação 11, a micro-F_1 é definida como

$$micro - F_1 = \frac{2PR}{P+R} \tag{12}$$

Utilizou-se a técnica de *cross-validation* estratificada para medir os resultados experimentais, isto é, cada *dataset* foi dividido randomicamente em 10 partes, garantindo que cada classe é proporcionalmente representada em cada parte. Os experimentos foram executados 10 vezes, sendo em cada execução utilizadas 9 partes para treinamento e uma para teste. Os resultados apresentados neste trabalho são uma média dos resultados em 10 execuções. A mesma estratégia foi utilizada para os *baselines*, utilizando-se as mesmas amostras de dados para treinamento e teste durante as execuções.

4.4 *Baselines*

O método proposto foi comparado com três *baselines* baseados no *Jaccard*, no Cosseno e no SVM. No método baseado no *Jaccard*, o coeficiente de similaridade do *Jaccard* (JACCARD, 1901) é utilizado para comparar cada instância contida no *dataset* de teste com cada instância contida do *dataset* do treino. A classe da instância no dataset de treino com a maior similaridade com cada instancia testada é escolhida para ser a classe da instância testada.

No método baseado no Cosseno, utilizou-se a mesma estratégia do método baseado no *Jaccard*, exceto que a similaridade do Cosseno é utilizada para comparar as instâncias. Cada instância é representada como um vetor de *tokens*, utilizando os *tokens* distintos contidos no *dataset* de treino. Os pesos dos tokens são calculados por meio do *inverse-document-frequency* (IDF), que mede a raridade do *termo* no conjunto de treinamento (BAEZA-YATES; RIBEIRO-NETO, 2011).

No método baseado no SVM, cada produto é associado a uma classe treinada pelo classificador SVM (VAPNIK, 1995). Cada instância é repre-

sentada como um vetor de *features* em que cada *token* correspondente a uma *feature*, e o seu valor IDF representa o peso de cada *feature*. Os experimentos foram realizados utilizando o pacote LibLinear[7].

[7] http://www.csie.ntu.edu.tw/∼cjlin/liblinear/

5 AVALIAÇÃO EXPERIMENTAL

Foi executado um conjunto de experimentos para avaliar o método proposto, suas variantes e as estratégias de poda. A seguir, são descritos cada um dos experimentos e a discussão dos resultados.

5.1 Tamanho Máximo do *Itemset*

Em algumas situações nenhuma regra é encontrada no modelo de treinamento cujo antecedente corresponda a um *itemset* que ocorra na *string* de teste ou ocorre algum empate entre as classes das regras encontradas. Neste caso, o método proposto precisa utilizar outra função de similaridade para decidir a classe da instância de teste. Neste experimento, foram consideradas somente as instâncias de teste em que suas classes podem ser descobertas usando somente regras, sem a utilização de outra função de similaridade.

Na Seção 3.5, foram discutidas algumas estratégias de poda para melhorar o tempo de execução do algoritmo. A terceira estratégia apresentada foi sobre o limite do tamanho dos *itemsets* a serem gerados. Este experimento avalia tal estratégia de poda. Na Figura 4 apresenta-se o resultado da micro-F1 e o percentual de ofertas de produtos classificados utilizando apenas regras, de acordo com a variação do tamanho máximo dos *itemsets* para o dataset *UOL-electronic*. Este experimento utiliza o suporte mínimo de 30%.

Utilizando somente *itemsets* de tamanho um, o método proposto obteve o maior resultado da medida micro-F1 para a classificação, no entanto, foram classificados somente $66,73\%$ das ofertas de produtos do *dataset* utilizadas para o experimento. À medida que se aumenta o tamanho máximo dos

itemsets, o número de ofertas de produtos classificados também aumenta, mas o resultado da micro-F1 diminui em razão do fato de aumentar o número de regras encontradas que podem levar a uma classificação errada. Para *itemsets* maiores que três, ambos os resultados se mantêm estáveis. Foi observado um comportamento semelhante em experimentos com outros *datasets*, o que motivou a proposta da estratégia de poda. Ou seja, pode-se limitar o tamanho dos *itemsets* a serem gerados, ganhando eficiência, sem perder a eficácia.

Nos próximos experimentos apresentados, foi adotado o tamanho máximo dos *itemsets* igual a três.

5.2 Identificação do Código do Produto

A hipótese apresentada é que o método proposto pode identificar códigos de produtos implícitos nas descrições das ofertas. No *dataset UOL-electronic*, a maioria das descrições possui um código de produto implícito, que pode identificar sua classe, visto que a maioria dos códigos de produtos são formados por um ou dois *tokens*. Analisando a Figura 4, pode-se observar que para os *itemsets* de tamanho um e dois, o método proposto classifica até $84,74\%$ das ofertas de produtos. Observando em detalhes os resultados, notou-se que a maioria dos acertos foi em virtude das regras formadas por códigos de produtos.

No entanto, o método proposto não é capaz de classificar mais ofertas de produtos utilizando regras formadas por códigos de produtos em decorrência de *tokens* ruidosos contidos nas descrições. Neste trabalho, são denominados ruídos aqueles *tokens* que são raros na coleção de ofertas de produtos, e erroneamente considerados como identificadores de produtos

pelo método proposto. Exemplos de *tokens* ruidosos podem ser visualizados nas Tabelas 2, 3, e 4. Para suporte igual 0%, a instância de teste #2 da Tabela 4 está classificada incorretamente, em razão dos *tokens* ruidosos "hewlettpackard" e "packard", que apontam para a classe c_1, enquanto a sua classe correta é a c_4, fornecida pela regra formada pelo código de produto "k5400". Neste caso, os *tokens* ruidosos conduziram ao erro. Em outras situações, um *token* ruidoso pode levar a um empate entre classes e, neste caso, o código do produto não é capaz de classificar a oferta do produto, sendo a classificação realizada com a utilização de outros *tokens* do *itemset* com tamanho maior, ou não sendo a instância classificada pelas regras.

5.3 Suporte Mínimo

Neste experimento, foi avaliada a influência do suporte mínimo para geração de regras, na qualidade dos resultados da classificação. Na Figura 6 ilustra-se a evolução dos resultados das métricas macro-F1 e micro-F1 e a porcentagem de ofertas de produtos classificadas usando apenas as regras de acordo com a variação do suporte mínimo. Com o aumento do suporte mínimo, a qualidade da classificação também aumenta, mas o percentual de ofertas de produtos classificados diminui. Para o suporte mínimo maior que 40%, os resultados para todas as métricas permanecem estáveis ou diminuem.

Este experimento foi realizado utilizando o *dataset UOL-electronic* e o mesmo comportamento foi observado nos *datasets UOL-non-electronic*, *UOL-book* e *Printer*. Com o objetivo de manter um relacionamento entre a qualidade da classificação e o número de ofertas de produtos classificados usando apenas as regras, os experimentos sugerem o uso do suporte mínimo

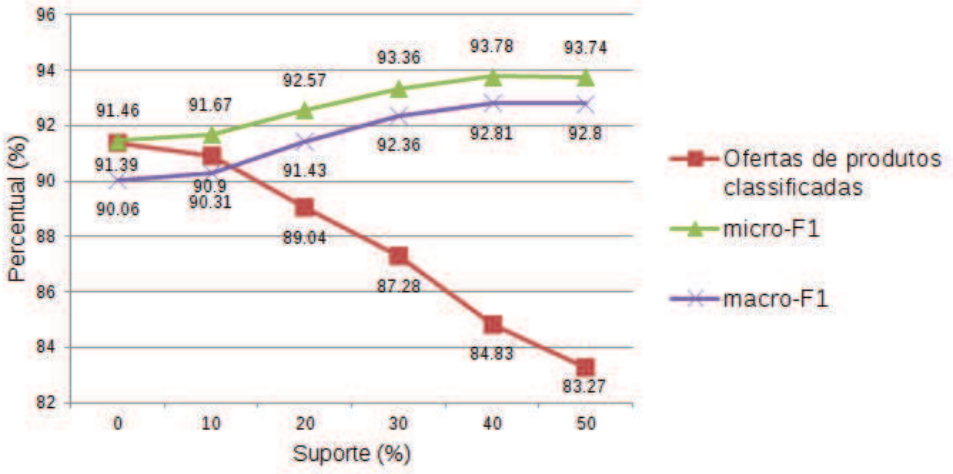

Figura 6 Evolução dos resultados macro-F1 e micro-F1 e o percentual de ofertas de produtos classificados usando apenas as regras de acordo com a variação do suporte mínimo, para o *dataset UOL-eletronic*.

em torno de 30%. Nos próximos experimentos apresentados, foi adotado o suporte mínimo igual a 30%.

O suporte mínimo também influencia na eficiência do método, uma vez que um k-itemset que não alcança um suporte mínimo, não é usado para gerar os $k+1$-itemsets. Esta é a segunda estratégia de poda discutida na Seção 3.5.

5.4 Comparação com os Baselines

Este experimento compara o método proposto com os baselines para classificação de ofertas de produtos. Para as instâncias de teste, quando o método proposto não é capaz de decidir qual sua classe usando apenas regras, utiliza-se o método baseado no SVM como função de similaridade, ou seja, utiliza-se o classificador SVM para classificar o que o método não

conseguiu classificar. Na Tabela 7 mostram-se os resultados nos *datasets* *UOL-electronic*, *UOL-non-electronic* e *Printer*, usando as métricas: macro-F1 e micro-F1. O método proposto obtém os melhores resultados em todos os *datasets* para todas as medidas. Estatisticamente, considerando um nível de confiança de 95%, pode-se afirmar que o método proposto é superior a todos os *baselines* nos *datasets* *UOL-electronic* e *UOL-non-electronic*, e empatado com o SVM no *dataset* *Printer*.

Tabela 7 Resultados comparando o método proposto com os baselines.

Método	*Uol-electronic*		*UOL-non-electronic*		*Printer*	
	MacF1	MicF1	MacF1	MicF1	MacF1	MicF1
Método Proposto	87.25	88.88	81.96	84.75	97.86	98.34
SVM	84.54	86.81	78.90	82.53	97.72	98.20
Cosseno	82.03	84.54	76.97	80.40	89.22	90.40
Jaccard	74.10	77.52	73.30	77.54	77.65	81.63

Comparando a qualidade dos resultados nos três *datasets*, o *dataset* que contém mais instância e mais classes é o que possui os piores resultados. O método proposto obtém os maiores ganhos nos resultados no *dataset* *UOL-non-electronic*, que contém mais instâncias e mais classes.

Foi avaliado também, o desempenho do método proposto para predizer a classe da oferta de produtos utilizando apenas as regras. Isto é, quando nenhuma regra é encontrada no modelo de treinamento para predizer a classe da instância de teste ou ocorre um empate para todos os *itemsets*, as instâncias foram desconsideradas no experimento. Nestes casos, pelo método proposto utiliza-se uma função de similaridade auxiliar para decidir a classe da instância. Como mostram as Figura 4 e 6, o método proposto foi capaz de classificar $87,28\%$ das instâncias de testes usando apenas regras no *dataset* *UOL-electronic* com o tamanho máximo do *itemset* igual a três e o suporte mínimo igual a 30%. As instâncias de teste classificadas pelo método

proposto usando apenas regras foi passada aos baselines para serem classificadas. Na Tabela 8 mostram-se os resultados dos datasets *UOL-electronic*, *UOL-non-electronic* e *Printer* usando as medidas, micro-F1 e macro-F1.

Tabela 8 Resultados comparando o método proposto com os *baselines* para as instâncias de teste classificadas usando somente regras no método proposto.

Método	*Uol-electronic*		*UOL-non-electronic*		*Printer*	
	MacF1	MicF1	MacF1	MicF1	MacF1	MicF1
Método Proposto	91.37	92.49	90.53	92.09	99.53	99.71
SVM	88.48	90.15	86.88	88.89	99.42	99.56
Cosseno	85.62	87.55	85.68	87.56	91.20	91.85
Jaccard	77.42	80.25	81.48	84.04	79.92	83.36

Assim como no experimento anterior, estatisticamente, o método proposto é superior a todos os baselines nos datasets *UOL-electronic* e *UOL-non-electronic*, e empatado com o SVM no dataset *Printer*. Neste experimento demonstra-se que, quando o método proposto é capaz de classificar uma instância usando regras, ele é geralmente melhor que os baselines. No entanto, o número de instâncias de teste classificadas pelas regras varia de acordo com o *dataset*. Para os datasets *UOL-electronic* e *Printer*, os quais a maioria das instâncias contém um código de produto implícito, o método proposto classifica $88,36\%$ e $93,98\%$ das instâncias de teste, respectivamente, enquanto que para o *dataset UOL-non-electronic* ele classifica $69,22\%$. Este resultado é um indicativo de que é necessário investigar formas para aumentar o número de instâncias classificadas pelas regras ou melhorar os resultados dos classificados. No próximo experimento foram investigadas outras variantes para o método proposto.

5.5 Alternativas para o Método Básico

Na Seção 3.4 foram descritas duas estratégias alternativas para o algoritmo básico. Neste experimento foram avaliadas essas alternativas. Na Tabela 9 mostram-se os resultados nos *datasets UOL-electronic*, *UOL-non-electronic* e *Printer* usando as métricas, micro-F1 e macro-F1. "Método Proposto" corresponde ao método proposto em sua versão básica, "Alternativa 1" e "Alternativa 2" correspondem às estratégias alternativas com o mesmo nome descrito na Seção 3.4.

Tabela 9 Resultados comparando o método proposto com suas estratégias alternativas.

Método	*Uol-electronic*		*UOL-non-electronic*		*Printer*	
	MacF1	MicF1	MacF1	MicF1	MacF1	MicF1
Método Proposto	87.25	88.88	81.96	84.75	97.86	98.34
Alternativa 1	86.83	88.46	81.93	84.67	97.86	98.34
Alternativa 2	87.58	89.17	81.99	84.77	97.28	97.88

Os resultados são estatisticamente similares. No entanto, a "Alternativa 1" é especialmente mais interessante quando o número de *tokens* nas descrições das ofertas de produtos é alto. Esta alternativa limita o número de *tokens* a serem combinados para gerar os *itemsets*, o que melhora o tempo de execução, sendo comprovada nos *datasets Uol-eletronic*, *Uol-non-eletronic* e *Printers* que reduziu o tempo de treinamento, como por exemplo no *dataset Uol-eletronic* o tempo reduziu de de 181,65 (s) para 29,36 (s) em média. Neste experimento, o número de *tokens* foi limitado a 13 *tokens*. A "Alternativa 2" também melhora o tempo de execução, desde que os casos de decisões de empate entre as regras seja feita usando *itemsets* de tamanho 1, evitando gerar *itemsets* maiores.

5.6 Treinamento Usando Poucas Instâncias por Classe

Köpcke, Thor e Rahm (2010) avaliaram um conjunto de abordagens de resolução de entidades para o problema de casamento de produtos. Em seu trabalho, para combinar as entidades de produtos de lojas online, nenhuma das abordagens obtiveram bons resultados. Neste experimento, foram utilizados os seus *datasets* para avaliar o método proposto e os *baselines*. Os resultados obtidos não são comparáveis com seus resultados porque a formulação do problema é diferente. Os seus *datasets* foram adaptados para a tarefa de classificação utilizada neste trabalho, como descrito na Seção 4.2, *datasets Abt-Buy* e *Amazon-Google*.

Foram executados quatro experimentos nestes *datasets*: (1) treinamento com o *dataset Abt* e teste com o *dataset Buy*, (2) treinamento com o *dataset Buy* e teste com o *dataset Abt*, (3) treinamento com o *dataset Amazon* e teste com o *dataset GoogleProducts*, e (4) treinamento com o dataset *GoogleProducts* e teste com o *dataset Amazon*. Portanto, nestes experimentos não foram utilizados a *cross-validation*. A principal característica destes experimentos é que os *datasets* de treinamento contêm em sua maioria apenas uma instância por classe. Nas Tabelas 10 e 11 mostram-se os resultados.

Tabela 10 Resultados comparando o método proposto com os baselines nos datasets Abt-Buy.

	Treino: *Abt* e Teste: *Buy*		Treino: *Buy* e Teste: *Abt*	
Método	MacF1	MicF1	MacF1	MicF1
Método Proposto	78.07	81.85	74.30	78.35
SVM	81.15	84.23	73.71	77.24
Cosseno	81.97	84.97	77.81	81.87
Jaccard	76.77	80.57	69.91	74.75

Tabela 11 Resultados comparando o método proposto com os baselines nos datasets Amazon-Google.

	Treino: *Amazon* e Teste: *Google*		Treino: *Google* e Teste: *Amazon*	
Metodo	MacF1	MicF1	MacF1	MicF1
Método Proposto	63.25	64.91	70.20	75.29
SVM	55.05	56.08	58.39	63.98
Cosseno	75.07	77.92	72.92	78.28
Jaccard	72.85	75.21	69.66	75.02

O método proposto, bem como o método baseado no SVM, não obtiveram os melhores resultados neste experimento. A principal razão é em decorrência do pequeno número de instâncias por classe para criar o modelo de treinamento. O método proposto não pode obter vantagem com o suporte mínimo porque a maioria das classes contém apenas uma instância. Estes *datasets* também contêm muitos produtos distintos, o que gerou muitas regras com *tokens* ruidosos pelo método proposto, ou seja, muitas regras são geradas por *tokens* que não identificam a classe. Um método simples como o baseado no Cosseno, obteve os melhores resultados nesses experimentos, embora tenha um tempo de execução maior que o método proposto.

5.7 Comportamento em Datasets Maiores

Foi avaliado também o comportamento do método proposto em um *dataset* maior, neste caso, utilizando o dataset *UOL-Book*. Neste experimento, foi avaliado o método proposto para predizer a classe das ofertas de produtos utilizando apenas as regras. O método foi capaz de classificar 80,43% das instâncias de testes. As instâncias de teste classificadas pelo método proposto foram passadas aos *baselines* para serem classificadas, porém ficou-se na espera de resultados por vários dias e não foi possível a execução.

Acredita-se que este acontecimento tenha sido motivado pelo tamanho da base de dados.

Na Tabela 12 mostram-se os resultados do método proposto no *dataset UOL-Book* usando as métricas, micro-F1, e macro-F1. Os resultados neste *dataset* foram melhores do que na maioria dos *datasets* analisados anteriormente. Isto deve-se ao fato das descrições dos produtos serem sucintas. Neste *dataset*, existem ofertas de produtos, que são constituídas apenas de *stopwords*, como por exemplo, o título do livro "*Quem, eu?*". Em virtude desse fato, neste experimento, as *stopwords* não foram removidas, pois resultaria em muitas ofertas sem nenhum *token*.

Tabela 12 Resultados utilizando o método proposto para classificar o dataset *UOL-Book*.

	UOL-book	
Método	MacF1	MicF1
Método Proposto	96.12	96.67

5.8 Tempo de Execução

Foi medido o tempo de execução do método proposto e dos *baselines* para classificar os datasets. O tempo de execução foi medido em cada um dos 10 *folds* do *cross-validation*, e neste trabalho, são apresentados o tempo médio destas 10 execuções e o desvio padrão. Os experimentos utilizando o *dataset UOL-non-eletronic* foram executados na *Máquina 1* e os experimentos utilizando o *UOL-Book* foi executado na *Máquina 2*, conforme apresentado na Seção 4.1. Na Tabela 13 apresentam-se os resultados.

O método proposto apresenta menor tempo total de execução que todos os baselines. Conforme apresentado na Seção 3.5, o método proposto apresenta um bom tempo de execução na prática.

Tabela 13 Tempo médio de execução em segundos (s) e o desvio padrão do método proposto dos *baselines*.

		Método Proposto		SVM		Cosseno	Jaccard
Datasets	Medidas	Treino	Teste	Treino	Teste	Teste	Teste
UOL-non-electronic	Tempo de Exec.	100,97	16,45	427,05	15,51	2.119,10	2.202,83
	Desvio	0,66	1,08	14,12	1,83	38,87	38,81
Uol-Book	Tempo de Exec.	122.402,53	17.003,89	-	-	-	-
	Desvio	956,05	154,13	-	-	-	-

Métodos baseado no SVM não são viáveis para classificar *datasets* com um número maior de classes (CRAMMER; SINGER, 2002). O SVM é baseado em decisões binárias, por exemplo, uma instância pertence ou não a uma determinada classe. Com múltiplas classes, usualmente, um classificador diferente precisa ser gerado para cada classe ou para cada par de classes (BAEZA-YATES; RIBEIRO-NETO, 2011). Nos experimentos realizados, não foi possível executar o método baseado no SVM no *dataset UOL-book*, o qual possui milhares de classes.

5.9 Dificuldades, Problemas e Limitações

Nesta seção, o objetivo é apresentar as limitações do método proposto e alguns casos em que o método falha, ilustrar exemplos e discutir as razões pelas quais as falhas ocorreram.

O método proposto possui a limitação de não resolver casos em que não encontra regras no modelo na fase de treinamento. Essa situação acontece quando a descrição de uma oferta de um produto é subconjunto de outra descrição. Por exemplo, *HP Officejet Pro 8000 Printer* refere à classe c_2 e *Hewlett Hp Officejet Pro 8000 Wireless Printer* refere-se à classe c_3 ou seja, a primeira oferta é subconjunto da segunda e referem-se a produtos distintos, conforme apresentado na Tabela 2. Neste caso, a primeira des-

crição de oferta não gera nenhuma regra, mostrando uma situação em o método proposto é limitado.

Uma das falhas encontradas no método proposto, porém minimizadas com a utilização do suporte mínimo, é em relação a tokens que possuem baixo valor classificatório, conforme mencionado anteriormente, são tokens ruídos que geram regras com 100% de confiança. Como apresentado na Tabela 3, com o suporte de 0%, são geradas várias regras originadas de tokens que são ruídos. Por exemplo, a regra $packard \rightarrow c_1$ e $hewlett\,packard \rightarrow c_1$ foram geradas e na etapa de teste, Tabela 4, as mesmas serviram de empates com as regras que classificariam a Instância #2 na classe correta.

Outra situação em que o método proposto falha é em função de problemas na própria base de dados, como, por exemplo o produto *Smartphone Samsung Galaxy S4 I9500 3G GSM Desbloqueado* pode ser identificado pelo token *I9500*, que representa o código embutido na descrição da oferta. Porém, nessa base de dados existe uma outra oferta desse mesmo produto, ou seja dessa mesma classe, *Celular Samsung Smartphone Galaxy S4 GT-I9505 Preto Box*, que se refere a outro, o qual é identificado pelo código *GT-I9505*. Nessa situação, o nosso método proposto falha na classificação, associando a instância de teste à classe errada.

Alguns casos de falhas podem ser minimizadas adicionando-se sinônimos aos tokens, como por exemplo, 16.1 megapixels, 16.1MP, 16.1 MP. Expressões dessa natureza podem ser identificados unicamente em uma determinada classe, porém eles são distintos apenas lexicograficamente, possuindo o mesmo valor classificatório para identificar uma determinada classe.

6 CONCLUSÃO E TRABALHOS FUTUROS

Os serviços de comparar preços são bastante utilizados pelos consumidores de lojas de comércio eletrônico em virtude da crescente utilização de *sites* para a compra de produtos. Uma tarefa desafiadora para esses *sites* é identificar quais as descrições em diferentes lojas de comércio eletrônico correspondem ao mesmo produto. Por esse motivo, existe uma grande quantidade de trabalhos abordando o problema de classificação de produtos.

Neste trabalho, considerou-se o problema de classificação em nível de ofertas de produtos, ou seja, a classificação não foi feita em nível de produtos em uma determinada categoria e, sim, em nível de oferta de cada produto. O método apresentado utilizou-se de regras de associação para encontrar um conjunto de palavras-chave, que difere as ofertas entre si.

Uma das principais contribuições deste trabalho é apresentar uma nova abordagem para classificação de produtos utilizando apenas a sua descrição, um atributo comumente disponível em sistemas de comércio eletrônico. O método proposto é capaz de identificar o código de produto implícito na descrição, quando eles existem, e é genérico para classificar produtos que não têm código.

A avaliação experimental realizada mostrou resultados relevantes para o problema de classificação, além de demonstrar a viabilidade de utilização do método proposto. Dentre todos os experimentos realizados, os que se esperavam melhores resultados eram os experimentos utilizando os *datasets Uol-eletronic* e *Printers*, pois nestes *datasets* as ofertas de produtos geralmente possuem um código de produto implícito em sua descrição. Apesar de nem sempre um código ocorrer nas descrições, quando existe, ele é importante para identificar referências a entidades, e neste trabalho, buscou-

se desenvolver soluções que consideram esse fato. Outros bons resultados obtidos pelo método proposto foi utilizando os datasets *Uol-non-eletronic* e *Uol-book*, pois apesar de não conterem o código de produto implícito, os dados de treinamento mostraram-se suficientemente satisfatórios para treinar o classificador no método utilizado. Os experimentos dos *datasets Abt-Buy* e *Amazon-Google* não foram os melhores para o método proposto, devido aos *datasets* de treinamento conterem em sua maioria apenas uma instância por classe, principal característica desses *datasets*, influenciando diretamente na impossibilidade de utilização do suporte mínimo para gerar regras.

As alternativas apresentadas ao método em sua versão básica são interessantes, em especial a *Alternativa 1*, a qual é capaz de reduzir consideravelmente o tempo de execução do método proposto. A estratégia de utilizar um suporte mínimo para gerar regras foi capaz de melhorar todos os resultados das métricas utilizadas na avaliação, pois permite a eliminação de *tokens* que são ruídos e reduz o espaço de combinação de itens proporcionando uma redução no tempo de execução.

Esta nova abordagem para classificação de ofertas de produtos pode contribuir também por fornecer uma visão interessante para novas pesquisas na área. Para os trabalhos futuros, um dos principais desafios é manter o modelo criado na fase de treinamento atualizado, evitando a necessidade de re-treinamento com o surgimento de novas ofertas de produtos. É interessante também investigar o desempenho do método proposto para identificar novas classes de produtos, ou seja, quando um produto não se encaixa em nenhuma das classes atuais.

REFERÊNCIAS

AGRAWAL, R.; IMIELINSKI, T.; SWAMI, A. Mining association rules between sets of items in large databases. **ACM SIGMOD Record**, New York, v. 22, n. 2, p. 207-216, June 1993.

AGRAWAL, R.; SRIKANT, R. Fast algorithms for mining association rules. In: INTERNATIONAL CONFERENCE ON VERY LARGE DATA BASES, 20., 1994, Santiago. **Proceedings…** Santiago: M. Kaufmann, 1994. p. 487-499.

AKEN, J. E. V. Management research as a design science: articulating the research products of mode 2 knowledge production in management. **British Journal of Management**, Chichester, v. 16, n. 1, p. 19-36, 2005.

BAEZA-YATES, R.; RIBEIRO-NETO, B. **Modern information retrieval:** the concepts and technology behind search. Indianapolis: A. Wesley Professional, 2011. 944 p.

BENJELLOUN, O. et al. Swoosh: a generic approach to entity resolution. **The VLDB Journal**, New York, v. 18, n. 1, p. 255-276, Jan. 2009.

BERRY, M. J. A.; LINOFF, G. **Data mining techniques:** for marketing, sales and customer support. New York: Wiley Computer, 1997. 888 p.

BHATTACHARYA, I.; GETOOR, L. Collective entity resolution in relational data. **ACM Transactions on Knowledge Discovery from Data (TKDD)**, New York, v. 1, n. 1, p. 1-36, Mar. 2007.

BILENKO, M.; BASIL, S.; SAHAMI, M. Adaptive product normalization: using online learning for record linkage in comparison shopping. In: IEEE INTERNATIONAL CONFERENCE ON DATA MINING, 5., 2005, Washington. **Proceedings…** Washington: IEEE Computer Society, 2005. p. 58-65.

BILENKO, M. et al. Adaptive name matching in information integration. **IEEE Intelligent Systems**, New York, v. 18, n. 5, p. 16-23, 2003.

BILENKO, M.; MOONEY, R. J. Adaptive duplicate detection using learnable string similarity measures. In: ACM SIGKDD INTERNATIONAL CONFERENCE ON KNOWLEDGE DISCOVERY AND DATA MINING, 9., 2003, New York. **Proceedings…** New York: ACM, 2003. p. 39-48.

CARVALHO, J. C. P.; SILVA, A. S. da. Finding similar identities among objects from multiple web sources. In: ACM INTERNATIONAL WORKSHOP ON WEB INFORMATION AND DATA MANAGEMENT, 5., 2003, New Orleans. **Proceedings…** New Orleans: ACM, 2003. p. 90-93.

CARVALHO, M. G. de et al. Learning to deduplicate. In: ACM/IEEE-CS JOINT CONFERENCE ON DIGITAL LIBRARIES, 6., 2006, New York. **Proceedings…** New York: ACM, 2006. p. 41-50.

CHAUDHURI, S.; GANTI, V.; KAUSHIK, R. A primitive operator for similarity joins in data cleaning. In: INTERNATIONAL CONFERENCE ON DATA ENGINEERING, 22., 2006, Washington. **Proceedings…** Washington: IEEE Computer Society, 2006. 1 CD-ROM.

CHAUDHURI, S.; GANTI, V.; MOTWANI, R. Robust identification of fuzzy duplicates. In: INTERNATIONAL CONFERENCE ON DATA ENGINEERING, 21., 2005, Washington. **Proceedings…** Washington: IEEE Computer Society, 2005. p. 865-876.

CHEN, M. S.; HAN, J.; YU, P. S. Data mining: an overview from a database perspective. **IEEE Transactions on Knowledge and Data Engijeering**, New York, v. 8, n. 6, p. 866-883, 1996.

CORTES, C.; VAPNIK, V. Support-vector networks. **Journal Machine Learning**, Hingham, v. 20, n. 3, p. 273-297, Sept. 1995.

CORTEZ, E. et al. Lightweight methods for large-scale product categorization. **Journal of the American Society for Information Science and Technology**, New York, v. 62, n. 9, p. 1839-1848, Sept. 2011.

CRAMMER, K.; SINGER, Y. On the algorithmic implementation of multiclass kernel-based vector machines. **The Journal of Machine Learning Research**, Edinburgh, v. 2, n. 2, p. 265-292, Dec. 2002.

FELLEGI, I.; SUNTER, A. A theory for record linkage. **Journal of the American Statistical Association**, New York, v. 64, n. 328, p. 1183-1210, 1969.

FRENCH, J. C.; POWELL, A. L.; SCHULMAN, E. Using clustering strategies for creating authority files. **Journal of the American Society for Information Science and Technology**, Chicago, v. 51, n. 8, p. 774-786, May 2000.

GHANI, R. et al. Text mining for product attribute extraction. **ACM SIGKDD Explorations Newsletter**, New York, v. 8, n. 1, p. 41-48, June 2006.

HAN, J.; KAMBER, M.; PEI, J. **Data mining:** concepts and techniques. 3rd ed. Waltham: M. Kaufmann, 2011. 744 p.

HERNANDEZ, M. A.; STOLFO, S. J. The merge/purge problem for large databases. **ACM SIGMOD Record**, New York, v. 24, n. 2, p. 127-138, May 1995.

JACCARD, P. Etude comparative de la distribution florale dans une portion des Alpes et des´ Jura. **Bulletin del la Soci´et´e Vaudoise des Sciences Naturelles**, Lausanne, v. 37, p. 547-579, 1901.

JARO, M. A. **UNIMATCH:** a record linkage system. Washington: Bureau of the Census, 1978. 275 p.

KANG, H. et al. Interactive entity resolution in relational data: a visual analytic tool and its evaluation. **IEEE Transactions on Visualization and Computer Graphics**, New York, v. 14, n. 5, p. 999-1014, 2008.

KANNAN, A. et al. Matching unstructured product offers to structured product specifications. In: ACM SIGKDD INTERNATIONAL CONFERENCE ON KNOWLEDGE DISCOVERY AND DATA MINING, 17., 2011, New York. **Proceedings…** New York: ACM, 2011. p. 404-412.

KOPCKE, H. et al. Tailoring entity resolution for matching ¨ product offers. In: INTERNATIONAL CONFERENCE ON EXTENDING DATABASE TECHNOLOGY, 15., 2012, New York. **Proceedings…** New York: ACM, 2012. p. 545-550.

KOPCKE, H.; THOR, A.; RAHM, E. Evaluation of entity resolution approaches on real-world ¨ match problems: proceedings of the VLDB Endowment. **Proceedings of the VLDB Endowment, VLDB Endowment**, Singapore, v. 3, n. 1/2, p. 484-493, Sept. 2010.

LEE, D. et al. Are your citations clean? **Communications of the ACM**, New York, v. 50, n. 12, p. 33-38, Dec. 2007.

LEVENSHTEIN, V. I. Binary codes capable of correcting deletions, insertions and reversals. **Soviet Physics Doklady**, Moscow, v. 10, n. 8, p. 707-710, 1966.

Disponível em: <http://profs.sci.univr.it/~liptak/ALBioinfo/files/levenshtein66.pdf>. Acesso em: 10 jun. 2014.

MENESTRINA, D.; WHANG, S. E.; GARCIA-MOLINA, H. Evaluating entity resolution results. **Proceedings of the VLDB Endowment, VLDB Endowment**, Singapore, v. 3, n. 1/2, p. 208-219, Sept. 2010.

MONGE, A.; ELKAN, C. The field matching problem: algorithms and applications. In: INTERNATIONAL CONFERENCE ON KNOWLEDGE DISCOVERY AND DATA MINING, 2., 1996, Portland. **Proceedings…** Portland: AAAI, 1996. p. 267-270.

NGUYEN, H. et al. Synthesizing products for online catalogs. **Proceedings of the VLDB Endowment, VLDB Endowment**, Singapore, v. 4, n. 7, p. 409-418, Apr. 2011.

PASULA, H. et al. Identity uncertainty and citation matching. In: _____. **Proceedings of the advances in neural information processing systems**. Vancouver: MIT, 2003. p. 1401-1408.

PEREIRA, D. A. **A web-based approach for entity resolution and creation of authority files**. 2009. 111 p. Tese (Doutorado em Ciência da Computação) - Universidade Federal de Minas Gerais, Belo Horizonte, 2009.

PEREIRA, D. A. et al. A generic web-based entity resolution framework. **Journal of the American Society for Information Science and Technology**, Chicago, v. 62, n. 5, p. 919-932, May 2011.

PEREIRA, D. A.; SILVA, E. E. B. da; ESMIN, A. A. A. Disambiguating publication venue titles using association rules. In: ACM/IEEE JOINT CONFERENCE ON DIGITAL LIBRARIES, 16., 2014, London. **Proceedings…** London: ACM/IEEE, 2014. 1 CD-ROM.

SATO, N. et al. Categorization of product pages depending on information on the web. In: COMPUTER SCIENCE AND SOFTWARE ENGINEERING, 8., 2011, Nakhon Pathom. **Proceedings…** Nakhon Pathom: IEEE, 2011. p. 393-398.

SEBASTIANI, F. Machine learning in automated text categorization. **ACM Computing Surveys**, New York, v. 34, n. 1, p. 1-47, Mar. 2002.

TAN, P. N.; STEINBACH, M.; KUMAR, V. **Introduction to data mining:** mineração de dados. Rio de Janeiro: Ciência Moderna, 2009. 28 p.

VAPNIK, V. N. **The nature of statistical learning theory**. New York: Springer-Velag, 1995. 314 p.

VELOSO, A. et al. Cost-effective on-demand associative author name disambiguation. **Information Processing & Management**, Oxford, v. 48, n. 4, p. 680-697, 2012. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0306457311000847>. Acesso em: 10 ago. 2014.

VELOSO, A. et al. Multi-evidence, multi-criteria, lazy associative document classification. In: ACM INTERNATIONAL CONFERENCE ON INFORMATION AND KNOWLEDGE MANAGEMENT, 15., 2006, Arlington. **Proceedings…** New York: ACM, 2006. p. 218-227.

WAINER, J. **Atualização em informática 2007**. Rio de Janeiro: PUC, 2007. 262 p. Disponível em: <https://www.sbc.org.br/bibliotecadigital/?module=Public&action=PublicationObject&subject=230&publicationobjectid=70>. Acesso em: 10 jul. 2014.

WANG, J. et al. Crowder: crowdsourcing entity resolution. **Proceedings of the VLDB Endowment, VLDB Endowment**, Singapore, v. 5, n. 11, p. 1483-1494, July 2012.

WINKLER, W. E. **The state of record linkage and current research problems**. Washington: Statistical Research Division, U.S. Census Bureau, 1999. 15 p.

ZHAO, Q.; BHOWMICK, S. S. **Association rule mining:** a survey. Singapore: Nanyang Technological University, 2003. 20 p.