ML-MDLText: um método de classificação de textos multirrótulo de aprendizado incremental

Marciele M. Bittencourt, Renato M. Silva, Tiago A. Almeida
Departamento de Computação (DComp)
Universidade Federal de São Carlos (UFSCar)
18052-780, Sorocaba, São Paulo, Brasil
marciele.bittencourt@gmail.com, renatoms@dt.fee.unicamp.br, talmeida@ufscar.br

***Resumo*—A classificação de textos tem sido estudada extensivamente nas últimas décadas e grande parte dos trabalhos relacionados ao tema são direcionados à classificação de rótulo único e ao aprendizado *offline*. Neste tipo de aprendizado, os documentos de texto são associados a apenas um rótulo e devem estar disponíveis com antecedência para o treinamento. Problemas reais de classificação de textos, no entanto, frequentemente envolvem instâncias multirrotuladas, que se tornam disponíveis continuamente e com padrões que mudam ao longo do tempo. Para manipular esses problemas, os classificadores idealmente deveriam ser capazes de predizer múltiplos rótulos para cada documento de texto e de atualizar o seu modelo preditivo de forma eficiente, para ser escalável mesmo com recursos de memória e tempo limitados, e ser rapidamente adaptável às mudanças nos padrões dos dados. Por isso, o aprendizado *online* e a classificação multirrótulo tem atraído grande interesse de pesquisa, uma vez que existem poucos métodos capazes de abordar os dois problemas simultaneamente e, assim, sempre é necessário recorrer às técnicas de transformação de problemas ou retreinar todo o modelo de predição quando novos documentos de texto se tornam disponíveis. Neste trabalho, é apresentado um método de classificação de textos baseado no princípio da descrição mais simples, que pode ser empregado em problemas de classificação multirrótulo sem a necessidade de transformá-los em problemas de rótulo único. Ele também apresenta a vantagem de considerar a existência de dependência entre os rótulos e de suportar o treinamento incremental naturalmente. O desempenho desse método foi avaliado empregando-o na tarefa de classificação em 15 aplicações de diferentes domínios e o resultado obtido foi comparado com os resultados de outros classificadores de referência na literatura, considerando cenários de aprendizado *offline* e *online*. Os resultados obtidos pelo método proposto são muito competitivos com os resultados obtidos pelos métodos estado-da-arte avaliados, uma vez que o ML-MDLText esteve entre as duas melhores performances em termos acurácia de subconjunto e macro F-medida em todos os experimentos de aprendizado *online*.**

Keywords*—*Classificação multirrótulo; Princípio da Descrição mais simples; Aprendizado online; Categorização de textos; Aprendizado de máquina.

- **Student level:** MSc
- **Date of conclusion:** March 27, 2020
- **Examining board members:**

1) Prof. Dr. Tiago A. Almeida (UFSCar) (Advisor).
2) Profa. Dra. Solange Oliveira Rezende (ICMC-USP).
3) Profa. Dra. Katti Faceli (UFSCar).

- **Pointer to the full dissertation:** [PDF]
- **Pointers to all publications derived from the research work:** [FILES]

Journals

1) M. M. BITTENCOURT; R. M. SILVA; T. A. ALMEIDA. ML-MDLText: an efficient and lightweight multilabel text classifier with incremental learning. *Applied Soft Computing*, Elsevier. *Artigo convidado do BRACIS'19. (Em revisão)

Conferences

1) M. M. BITTENCOURT; R. M. SILVA; T. A. ALMEIDA. ML-MDLText: A multilabel text categorization technique with incremental learning. In: *2019 8th Brazilian Conference on Intelligent Systems (BRACIS'19)*. Salvador, BA, Brasil: IEEE, 2019. p. 580–585. ISSN 2643-6256. Doi: 10.1109/BRACIS.2019.00107. URL: https://doi.org/10.1109/BRACIS.2019.00107

I. Introdução

Com os avanços recentes na tecnologia para armazenar e compartilhar dados, um grande volume de documentos de textos é gerado a todo momento. Nesse contexto, a categorização de textos automática tem se tornado a solução mais efetiva para explorar o conteúdo de toda informação em formato de texto disponível de maneira eficiente.

O conceito da classificação em muitos métodos clássicos de aprendizado de máquina segue uma abordagem *monorrótulo*, na qual é possível associar apenas uma categoria em cada documento de texto [1], [2]. Em muitas aplicações reais de categorização de textos, no entanto, o conteúdo de um documento pode ser associado a diversas categorias ou rótulos simultaneamente. Problemas como esses são conhecido como problema de classificação *multirrótulo* e contém diversas características desafiadoras [1], [2], [3]. Uma delas é que o tamanho do conjunto de rótulos associados em cada documento é um dos parâmetros que o método de classificação precisa aprender. Além disso, como há mais rótulos associados em cada documento, existem mais relacionamentos entre rótulos e documentos que devem ser aprendidos e, consequentemente, é exigido um custo computacional maior para gerar o modelo preditivo ao se comparar com a abordagem monorrótulo tradicional [4].

Alguns autores consideram que um problema de classificação multirrótulo é um conjunto de problemas de rótulo único e usam técnicas que os transformam em problemas de classificação binária ou multiclasse para empregar os métodos clássicos de classificação [3], [2].

Contudo, o emprego dessas transformações não é viável quando o número de rótulos ou de distintas combinações de rótulos (*labelsets*) é alto. Além disso, algumas características específicas do problema são ignoradas quando alguma técnica de transformação é aplicada. Por exemplo, informações de dependência ou de correlação entre os rótulos, que poderiam ser usadas para melhorar o poder preditivo do modelo, podem ser perdidas [3], [5].

Com relação a forma da construção do classificador, muitos métodos de classificação multirrótulo estabelecidos são baseados em aprendizado *offline*, no qual é exigido que um único lote de documentos rotulados seja apresentado em uma etapa de treinamento única. Diversos problemas de classificação de textos atuais envolvem o desafio de manipular um grande número de instâncias que crescem rapidamente com o tempo. Devido à dinâmica da geração acelerada desses novos textos, os padrões capturados pelas técnicas de aprendizado de máquina podem se tornar obsoletos rapidamente e recursos computacionais de memória ou tempo podem ser insuficientes para carregar e processar todos os dados disponíveis para o treinamento. Métodos indicados para problemas reais e dinâmicos devem suportar o aprendizado *online*, pois, dessa forma, o processo de treinamento pode ocorrer em diferentes momentos e de forma incremental, o que permite que o classificador seja atualizado conforme apareçam novos padrões de dados, mesmo com recursos de memória e tempo limitados [6], [7].

O aprendizado *online* tem sido muito explorado nos últimos anos, mas muitos dos estudos relacionados a esse tema são direcionados aos problemas de classificação monorrótulo [8], [9], [7], [10]. Alguns autores, como Almeida *et al.* [8], [9], de Freitas *et al.* [10] e Silva *et al.* [7], criaram métodos de classificação baseados na teoria do Princípio da Descrição mais Simples (*Minimum Description Length – MDL*) [11] que são totalmente incrementais. O princípio MDL foi proposto por Rissanen [11] e pode ser um grande aliado para resolver problemas de classificação, visto que sua estratégia de seleção de modelos evita o problema de sobreajustamento (*overfitting*). No entanto, os trabalhos que englobam esse princípio e o aprendizado *online* são direcionados a problemas binários ou multiclasse.

Para preencher as lacunas relacionadas à classificação de textos multirrótulo, este trabalho visa apresentar um método capaz de lidar com os principais desafios associados à classificação de textos e ao aprendizado multirrótulo simultaneamente. Tendo em vista as características e os trabalhos relacionados aos dois assuntos, a definição desse método foi fundamentada na seguinte pergunta de pesquisa: seria possível criar um classificador multirrótulo confiável através da adaptação de um método baseado no princípio MDL que seja naturalmente incremental e que não necessite de técnicas de transformação?

O objetivo deste trabalho é apresentar uma solução eficiente em termos de desempenho de classificação e em custo computacional para os problemas de classificação de textos multirrótulo e *online*, baseado no princípio MDL. Em resumo, as principais contribuições oferecidas neste trabalho são:

- Apresentação de um estudo sobre o problema de classificação de textos, sobre as abordagens direcionadas ao problema de classificação multirrótulo e sobre o princípio MDL, que serviu de base para a construção do método proposto;
- Definição e avaliação de um novo método de classificação de textos, baseado no princípio da descrição mais simples, que é capaz de lidar com os desafios da classificação de textos multirrótulo e *online*.
- Análise do desempenho do método proposto usando estratégias de avaliação de classificadores multirrótulo em diferentes cenários de aprendizado *online*;

O restante deste artigo está organizado como segue. A Seção II apresenta o problema da categorização multirrótulo, o aprendizado *online* e os principais estudos da literatura direcionados a esses dois assuntos. A Seção III descreve brevemente a teoria do princípio MDL e algumas de suas aplicações. A Seção IV detalha o funcionamento do método de classificação multirrótulo proposto. A Seção V apresenta a metodologia experimental utilizada para avaliar o método proposto. A Seção VI apresenta os resultados obtidos. Finalmente, a Seção VII oferece as principais conclusões e os direcionamentos para trabalhos futuros.

II. O APRENDIZADO MULTIRRÓTULO E ONLINE

No aprendizado supervisionado multirrótulo, um conjunto de associações entre exemplos e rótulos são apresentados ao algoritmo. Sendo $\mathcal{D} = \{d_1, d_2, ..., d_m\}$ um conjunto de amostras ou observações e $\mathcal{Q} = \{c_1, c_2, ..., c_q\}$, um conjunto finito com as possíveis classes ou rótulos do problema, é apresentado ao algoritmo um conjunto de documentos rotulados que pode ser expresso como $\mathcal{T} = \{(d_1,Y_1), (d_2, Y_2), ..., (d_m, Y_m)\}$, onde $Y_m \subseteq \mathcal{Q}$ [12], [13]. O subconjunto de rótulos Y_m é conhecido como *labelset* do documento d_m e o tamanho do subconjunto $|Y_m|$ corresponde ao número de rótulos associados a d_m [3]. Nesse estudo, $\mathcal{Q}^* = \{Y_1, Y_2, ..., Y_{q^*}\}$ é o conjunto com todos os q^* *labelsets* distintos observados nas amostras de treinamento.

Na classificação multirrótulo *online*, o objetivo é encontrar uma função $H : \mathcal{D} \rightarrow 2^Q$ capaz de descrever as associações em $\mathcal{T}$ e que seja facilmente atualizável caso novas associações (d_s, Y_s) se tornarem disponíveis, sem perder as características conhecidas das associações antigas. Essa função é então empregada para predizer o subconjunto de rótulos Y que melhor representam uma amostra não rotulada.

De acordo com Tsoumakas *et al.* [2], os métodos de classificação multirrótulo podem ser divididos em dois grupos: *métodos de transformação de problemas* e *métodos de adaptação de algoritmos*.

Métodos de transformação de problemas, tais como os baseados em relevância binária (BR - *Binary Relevance*) [14] ou no método pareado (PW - *Pairwise*) [15], manipulam os problemas de classificação multirrótulo como se fossem um conjunto de problemas monorrótulo e empregam métodos clássicos de aprendizado monorrótulo para a tarefa de classificação. Um inconveniente de métodos como esses é que, quando o problema contém uma grande quantidade de rótulos, um volume muito alto de classificadores são construídos para gerar a saída multirrotulada. Além disso, alguns métodos baseados na transformação BR são frequentemente criticados na literatura por ignorar correlações ou dependências

que possam existir entre os rótulos [3]. Por outro lado, os métodos de transformação baseados em *Label Powerset* (LP), diferentemente da transformação BR, podem naturalmente explorar a dependência entre os rótulos. Eles assumem que cada *labelset* distinto é uma classe e, dessa forma, o problema de classificação é transformado em um problema multiclasse. No entanto, seu desempenho é prejudicado quando existem muitos *labelsets* diferentes e *labelsets* raros.

Métodos de adaptação de algoritmos englobam abordagens monorrótulo que foram adaptadas para manipular diretamente os problemas de classificação multirrótulo sem ser necessário modificar a estrutura dos dados originais. Diversos métodos referência em tarefas de classificação, como aqueles baseados em árvores de decisão (ML-C4.5 [16] e MHT [6]), nos k-vizinhos mais próximos (ML-KNN [13], DMLkNN [17] e BRkNN-a/BRkNN-b [18]), nas máquinas de vetores de suporte (Rank-SVM [19]), em redes neurais artificiais (MMP [20], MLPP [15], [21], BP-MLL [12]) e em probabilidades (Mixture Model [22], MLNB [23]) já tiveram versões adaptadas para tarefas de classificação multirrótulo. A principal vantagem desses métodos é a capacidade de lidar com características específicas do problema de classificação multirrótulo. Contudo, muitos deles são direcionados a tarefas de aprendizado *offline*.

Dentre os poucos trabalhos que abordaram o aprendizado multirrótulo em cenários de aprendizado *online*, grande parte deles emprega o conceito das técnicas de transformações. Um dos primeiros estudos relacionado ao aprendizado multirrótulo e *online* foi conduzido por Crammer e Singer [20], que propuseram uma extensão da transformação BR com o algoritmo perceptron para gerar os protótipos de predição na manipulação de problema de *ranking* de tópicos. Já Mencía e Fürnkranz [21], [15] empregaram o perceptron em conjunto com o método de transformação PW. Segundo os autores, PW é superior ao BR, pois os subproblemas gerados são menores, o que aumenta a chance de encontrar um bom hiperplano de separação entre os rótulos e permite explorar correlações existentes entre os pares de rótulos. Qu *et al.* [24] propuseram um *ensemble* de metaclassificadores com o método de transformação BR empilhada para classificar dados multirrotulados em fluxo. Nessa proposta, os modelos antigos eram substituídos por modelos mais novos, gerados a cada *batch*-incremental através de conjuntos de dados atualizados. Como esse esquema gera novos modelos a cada processo de atualização, os métodos classificação base não precisam permitir o aprendizado *online* nessa proposta. Spyromitros-Xioufis *et al.* [25] fizeram versões melhoradas do algoritmo KNN com transformação BR para torná-lo mais eficiente. Na proposta dos autores, foi introduzido um mecanismo de parametrização de janela para manipular com o desbalanceamento na distribuição de amostras negativas e positivas de cada rótulo. Nesse trabalho, os autores utilizaram um esquema de treinamento instância-incremental, uma forma diferente da empregada por Qu *et al.* [24] e mais vantajosa para aprender com fluxo de dados.

Baseado no conceito de probabilidade e adaptação de algoritmo, Zhang *et al.* [26] propuseram um framework bayesiano de classificação multirrótulo *online* (Bayesian Online Multirrótulo Classification - BOMC). Os autores compararam esse método em um cenário incremental com o LaSVM e obtiveram resultados inferiores conforme o número de exemplos do conjunto de treinamento aumentava em problemas com poucas classes. Já Read *et al.* [27], [6] usaram *ensembles* de árvores Hoeffding (Hoeffding *tree* - HT) para desenvolver um novo método *online*, empregando a mesma versão da fórmula de entropia usada por Clare e King [16] na adaptação do C4.5. A árvore de Hoeffding difere da clássica árvore de decisão porque é considerado apenas um pequeno número de amostras na construção de cada nó da árvore, número este definido por um limite de Hoeffding. Na proposta dos autores, denominada árvores Hoeffding multirrótulo (MHT - *Multilabel* Hoeffding *Trees*), classificadores de transformação PS foram empregados nas folhas das árvores Hoeffding para melhorar o poder de predição.

III. O PRINCÍPIO MDL

O princípio MDL é um método de seleção de modelos originalmente proposto por Rissanen [11]. Esse princípio tem a prerrogativa de que o melhor modelo para explicar um dado é aquele que produz a descrição mais compacta do dado (a que obtém o menor tamanho de descrição). O princípio MDL se baseia no conceito de compressão de dados e tem raízes na complexidade de Kolmogorov [28].

O conceito de aprendizado por trás da compressão de dados é que qualquer regularidade encontrada em um dado pode ser usada para comprimí-lo, ou seja, qualquer regularidade pode ser usada para gerar uma codificação ou uma descrição do dado usando menos símbolos do que a quantidade de símbolos usada para listar o dado literalmente [29]. A complexidade de Kolmogorov de uma sequência binária é definida como o tamanho do menor programa que imprime a sequência e então, finaliza. Quanto menor a complexidade de Kolmogorov, mais regular é a sequência. Como o menor programa é também o que foi capaz de capturar mais regularidades, é possível dizer que esse programa é o que melhor representa essa sequência. Dessa forma, esse conceito também pode ser usado para definir a inferência indutiva em geral [30], [31].

A primeira versão do princípio MDL é conhecida como *crude* MDL ou MDL em duas partes. Nessa versão, dado um conjunto de modelos $\mathcal{M} = \{m_1, m_2, ..., m_t\}$ e um dado x, o modelo selecionado para representar x é aquele que minimiza a equação:

$$M_{MDL} = \arg\min_{m_j \in \mathcal{M}} [L(m_j) + L(x|m_j)], \tag{1}$$

onde $L(m_j)$ é o tamanho de descrição do modelo m_j e $L(x|m_j)$ é o tamanho da descrição de x quando codificado pelo modelo m_j. Rissanen [11] demonstrou que $L(x|m_j)$ é baseada na probabilidade condicional de x dado m_j, ou seja $L(x|m_j) = \lceil -\log_2 P(x|m_j) \rceil$. O valor de $L(m_j)$ da Equação 1 é difícil de se calcular, já o tamanho da codificação do modelo pode ser muito grande para um código e muito pequeno para outro, o que pode tornar $L(m_j)$ um valor arbitrário [29]. Para resolver esse problema, [11] propôs uma versão refinada baseada na ideia de código universal, chamada de princípio MDL *refinado*, onde um código de uma parte com tamanho $\bar{L}(x|m_j)$ é usada.

Diversos trabalhos direcionados à tarefas de classificação monorrótulo em cenários de aprendizado *online* foram propostos empregando o princípio MDL [8], [32], [7], [10]. Estudos recentes mostraram que os métodos baseados no princípio MDL naturalmente evitam o problema de *overfitting*, pois o

processo usado para a seleção de modelos busca um equilíbrio entre a complexidade de modelo e a habilidade de ajuste aos dados. Assim, eles tendem a selecionar o modelo que melhor captura os padrões dos dados e, ao mesmo tempo, não é excessivamente complexo [8], [7]. As vantagens do uso desse princípio também foram exploradas em outras aplicações de aprendizado de máquina, como na transferência de aprendizado [33], em árvores de decisão [34], aprendizado profundo [35] e modelagem de problemas de aprendizado incremental [36].

Apesar da eficiência comprovada em várias aplicações, o princípio MDL é empregado em poucos estudos relacionados ao aprendizado multirrótulo. Laghmari *et al.* [37], por exemplo, usaram o princípio MDL como estratégias de pré-corte para evitar o processo de *overfitting* em árvores de decisão binárias, aplicadas em problemas de classificação multirrótulo. Contudo, não se tem conhecimento de nenhum outro método que use o princípio MDL como principal estratégia para a classificação multirrótulo.

IV. ML-MDLTEXT

O método proposto nesse trabalho, denominado ML-MDLText, é capaz de lidar com tarefas de categorização multirrótulo em cenários de aprendizado *online*. Essa proposta é uma adaptação do MDLText [7], um método de categorização de textos multiclasse baseado no princípio MDL, que apresenta vantagens como a eficiência na classificação de textos, robustez contra sobreajustamento, escalabilidade e aplicabilidade em cenários de aprendizado *online*. Dado o seu alto desempenho apresentado recentemente por Silva *et al.* [7], foi levantada a hipótese de que uma abordagem multirrótulo adaptada desse método poderia manter suas principais vantagens e, portanto, levar a um alto desempenho na categorização, mesmo sem a necessidade de transformar o problema de classificação multirrótulo em problemas binários ou multiclasse.

Enquanto que o MDLText, proposto por Silva *et al.* [7], considera que as classes do problema são mutuamente exclusivas, os principais objetivos do ML-MDLText são (*i*) incorporar informações das ocorrências dos termos em cada rótulo individualmente, como ocorre na transformação BR e no método MDLText, e (*ii*) agregar informações relacionadas as ocorrências dos termos em cada *labelset* distinto que aparece nas amostras de treinamento, como é feito na transformação LP. Assim, o classificador usa tanto informações relacionadas às classes, como também agrega alguma dependência que pode existir entre elas para gerar a predição final.

O ML-MDLText prediz os rótulos de um dado documento não rotulado d através dos seguintes processos:

1) Inicialmente, é definida a relevância das classes através do tamanho de descrição, descrito como $L(d|c_j)$, obtido por cada classe $c_j \in \mathcal{Q}$ ao codificar d. Quanto menor o tamanho de descrição, mais relevante é a classe para d. As classes mais relevantes são candidatas a compor o *labelset* de d.
2) O tamanho do *labelset* de d é estimado por um metamodelo similar ao proposto por Tang *et al.* [38]. Esse metamodelo é construído através de metadados extraídos da base de dados multirrótulo original. Como esse metamodelo pode não ser preciso na predição do tamanho do *labelset*, é aplicada uma margem de confiança baseada em uma função gaussiana. Assim, a influência do metamodelo na definição do tamanho final do *labelset* é proporcional a essa confiança.
3) São escolhidos os *labelsets* mais prováveis de ser o conjunto de classes de d. Os *labelsets* mais prováveis são aqueles que apareceram nos dados de treinamento, que contém pelo menos uma das classes relevantes selecionadas no Processo 1 e cujo tamanho é igual ao que foi definido pelo metamodelo ou que se enquadra no intervalo definido pela função gaussiana usada no processo 2. Esses *labelsets* compõem o conjunto $\mathcal{S}$.
4) A relevância de cada *labelset* $Y_j \in \mathcal{S}$ é calculada baseada na função $L(d|Y_j)$, ou seja, o tamanho de descrição de d é calculado com relação a Y_j. O *labelset* com o menor tamanho de descrição é selecionado como o mais adequado para representar d.

Os quatro processos são descritos a seguir. Para facilitar a compreensão e a reprodução dos experimentos, o código fonte do ML-MDLText está disponível publicamente no `GitHub`[1].

A. Processo 1: definindo as classes relevantes

O tamanho de descrição de d é calculado para cada classe c_j e ela é usada para medir a relevância de c_j com respeito a d. Dessa forma, as classes são usadas como modelos na equação do tamanho de descrição do princípio MDL. O tamanho de descrição corresponde ao tamanho de codificação gerado por cada classe c_j ao codificar d e foi definido por Silva *et al.* [7] de acordo com a seguinte equação:

$$L(d|c_j) = \left[\sum_{i=1}^{|d|} L(t_i|c_j) \times K(t_i)\right] \times \hat{S}(d, c_j). \quad (2)$$

O tamanho de descrição $L(d|c_j)$ proposto por Silva *et al.* [7] é composto por três termos:

- $L(t_i|c_j)$: é o tamanho de descrição de cada termo t_i quando ele é representado por cada classe (modelo) c_j;
- $K(t_i)$: é uma função de pontuação usada para aumentar a contribuição dos termos com alta relevância no tamanho de descrição;
- $\hat{S}(d, c_j)$: é uma função de penalidade baseada na similaridade de cosseno entre d e o vetor protótipo da classe c_j.

O tamanho de descrição de cada termo $L(t_i|c_j)$ pode ser calculado pela seguinte equação, inspirada na codificação de Shannon-Fano [39]:

$$L(t_i|c_j) = \lceil -log_2 \beta(t_i, c_j) \rceil \quad (3)$$

$$\beta(t_i, c_j) = \frac{n_{c_j,t_i} + \frac{1}{|\Omega|}}{\hat{n}_{c_j} + 1}, \quad (4)$$

onde n_{c_j,t_i} é a soma dos pesos TF-IDF normalizados de t_i nos documentos de treinamento que pertencem a classe c_j;

[1] *Código fonte do ML-MDLText* na linguagem de programação Python. Disponível em http://github.com/m-bittencourt/ML-MDLText (acessado em 20/06/2020).

$\hat{n}_{c_j}$ é a soma dos pesos de todos os termos dos documentos que pertencem a classe c_j; e $|\Omega|$ é um parâmetro usado para reservar uma porção do tamanho de descrição para os termos que nunca apareceram nos documentos de treinamento da classe c_j. Nesse estudo, foi utilizado $|\Omega| = 2^{10}$ [7].

De acordo com Silva *et al.* [7], o propósito de $K(t_i)$ é associar pesos diferentes para os termos de acordo com sua importância na definição das classes. A expressão $K(t_i)$ é formulada por:

$$K(t_i) = \frac{1}{(1+\mu) - F(t_i)}, \tag{5}$$

onde μ é uma constante necessária para evitar problemas no cálculo da divisão e $F(t_i)$ é uma função de pontuação que leva em conta a frequência do termo t_i em cada classe para definir sua relevância. Nesse estudo, foi empregado $\mu = 10^{-3}$ [7]. Para calcular $F(t_i)$, Silva *et al.* [7] usou a função fatores de confidência (CF) [40] adaptada para problemas de classificação multiclasse. Mas, nesse estudo, foi utilizada uma nova versão da função CF adaptada para problemas de classificação multirrótulo que pode ser definida pela seguinte equação:

$$F(t_i) = \frac{1}{(|\mathcal{Q}^*|-1)} \times \sum_{\forall_{Y_j \in \mathcal{Q}^* \ | \ Y_j \neq Y_\tau}} \frac{\left[\frac{(SH)^2 + TH - \frac{\lambda_1}{MH}}{(MH)^2}\right]^{\lambda_2}}{1 + \left(\frac{\lambda_3}{MH}\right)}, \tag{6}$$

onde τ é o índice do *labelset* mais frequente; j é o índice do *labelset* de $\mathcal{Q}^*$; ϕ_{Y_τ,t_i} é o número de documentos que contém o termo t_i e pertencem a Y_τ; ϕ_{Y_j,t_i} é o número de documentos que contém o termo t_i e pertencem a Y_j; $SH = \phi_{Y_\tau,t_i} - \phi_{Y_j,t_i}$; $TH = \phi_{Y_\tau,t_i} \times \phi_{Y_j,t_i}$; e $MH = \phi_{Y_\tau,t_i} + \phi_{Y_j,t_i}$. Nessa equação, λ_1, λ_2 e λ_3 são constantes que ajustam a velocidade de decaimento do fator de confidência. Nesse estudo, foram usados os mesmos valores propostos por Assis *et al.* [40], que são $\lambda_1 = 0.25$, $\lambda_2 = 10$ e $\lambda_3 = 8$.

Finalmente, o $\hat{S}(d, c_j)$ da Equação 2 é uma penalidade usada para aumentar a margem de separação entre os modelos, isto é, para aumentar a diferença entre o tamanho de descrição do modelo mais provável e do menos provável. Essa penalidade é baseada na similaridade de cosseno entre o documento e o vetor protótipo do modelo c_j. Quanto menos similar for o documento e o modelo c_j, maior será a penalidade e consequentemente, maior será o tamanho de descrição necessário para representar d. O termo $\hat{S}(d, c_j)$ pode ser calculado por:

$$\hat{S}(d, c_j) = -log_2\left[\frac{1}{2} \times S(d, \overline{c}_j)\right]. \tag{7}$$

$S(d, \overline{c}_j)$ é a similaridade de cosseno entre d e o vetor protótipo do modelo c_j:

$$S(d, \overline{c}_j) = \frac{\sum_{i=1}^{|d|} \hat{w}(t_i, d|c_j) \times \overline{c}_j(t_i)}{||\hat{w}(:, d)||_2 \times ||\overline{c}_j||_2}, \tag{8}$$

onde $\hat{w}(t_i, d|c_j)$ é o peso TF-IDF de t_i nos documentos de treinamento da classe c_j; $\hat{w}(: d)$ é o peso TF-IDF de todos os termos presentes em d; $\overline{c}_j(t_i)$ é o vetor protótipo do modelo c_j de acordo com o termo t_i; e $||\overline{c}_j||_2$ é a norma do vetor protótipo da classe c_j.

Todos os parâmetros necessários para calcular $L(d|c_j)$ podem ser obtidos facilmente através dos documentos de treinamento. Eles também podem ser atualizados conforme novos documentos são apresentados e, consequentemente, $L(d|c_j)$ pode ser calculado incrementalmente de forma simples e rápida.

Baseado no tamanho de descrição obtido por cada classe, o ML-MDLText seleciona as primeiras cm classes (em ordem crescente pelo tamanho de descrição obtido) como candidatas a compor o *labelset* de saída de d. O valor de cm é definido como o menor número de posições de *ranking* que encontra pelo menos uma classe que compõe o verdadeiro *labelset* de $p^{\mathcal{T}}\%$ amostras de treinamento. Por exemplo, assumindo que as classes são ordenadas de forma crescente de acordo com o tamanho de descrição obtido em cada exemplo de treinamento. Se a primeira classe faz parte do verdadeiro *labelset* de pelo menos $p^{\mathcal{T}}\%$ das amostras de treinamento, então $cm = 1$. Caso contrário, se pelo menos uma dentre as primeiras duas classes faz parte do verdadeiro *labelset* de $p^{\mathcal{T}}\%$ das amostras de treinamento, então $cm = 2$. Caso contrário, o mesmo processo é executado considerando as primeiras três classes e assim por diante. Com cm definido no estágio de treinamento, as classes de maior relevância para d podem ser definidas formalmente por:

$$\mathcal{R} = \arg\min_{c_j \in \mathcal{Q}} [L(d|c_j)]_{1:cm}. \tag{9}$$

B. Processo 2: definindo o tamanho do labelset

Nesse processo, um metamodelo é usado para estimar o tamanho do *labelset* do documento não rotulado d [38]. Para construir esse metamodelo, uma base de dados multiclasse é extraída da base de dados multirrótulo original, cujas classes dessa nova base é o número de rótulos associados com cada documento do problema multirrótulo original, como mostra a Figura 1. Assim, a base de dados extraída contém os mesmos documentos da base original e o novo conjunto de classes do problema multiclasse é um conjunto $\mathcal{Q}' = \{a_1, a_2, ..., a_{q'}\}$, onde q' é o índice do *labelset* com maior tamanho encontrado nas amostras do conjunto multirrótulo.

Q={A, B, C, D}

Base de Dados multirrótulo	
Documento	Y
d_1	A, C
d_2	A, C, D
d_3	B, D
...	...
d_n	B

Q'={1, 2, 3, 4}

Base de dados meta multiclasse	
Documento	y
d_1	2
d_2	3
d_3	2
...	...
d_n	1

Figura 1: Extração da base de dados multiclasse do problema multirrótulo original [38], [41].

Após essa extração, um metamodelo é treinado com essa nova base multiclasse e é utilizado para predizer o tamanho do *labelset* de documentos não rotulados. Nesse estudo, foi escolhido o método de classificação *online* SGD para a construção do metamodelo, como usado por Bittencourt *et al.* [41].

Como o metamodelo pode não ser preciso na definição do tamanho do *labelset*, uma função gaussiana baseada na Equação 10 é definida para cada classe presente em $\mathcal{Q}'$, com

o intuito de considerar outros tamanhos de *labelsets* próximos ao número definido pelo metamodelo. A abertura dessa curva varia de acordo com a capacidade preditiva do metamodelo, avaliada pela F-medida [20] obtida por cada classe na predição das classes dos exemplos de treinamento:

$$G(\sigma_{n_d}, n_d, a_j) = e^{\frac{-(a_j - n_d)^2}{2\sigma^2_{n_d}}}, \tag{10}$$

$$\sigma_{n_d} = -log_2\Big[\gamma_1 \times (F\text{-}medida_{n_d})^2\Big], \tag{11}$$

onde n_d é a classe estimada pelo metamodelo (*i.e.*, o tamanho do *labelset*); a_j é a classe de índice j que pertence a $\mathcal{Q}'$; e $F\text{-}score_{n_d}$ é a F-medida obtida pelo metamodelo ao predizer a classe n_d. Nessa equação, σ_{n_d} indica a abertura da função gaussiana baseada na F-medida. Além dela, γ_1 é usada para expandir ainda mais essa abertura e foi definida de forma empírica como $\gamma_1 = 0.95$, que indica que outros tamanhos de *labelsets* devem ser considerados quando o metamodelo predizer classes com $F\text{-}medida \leq 0.95$.

A Figura 2 ilustra o comportamento das curvas gaussianas geradas para cada classe de $\mathcal{Q}'$ em uma base de dados qualquer. O eixo x representa a diferença entre as predições do metamodelo e os tamanhos verdadeiros dos *labelsets* das amostras de treinamento (classes $a_j \in \mathcal{Q}'$), correspondendo ao termo $(a_j - n_d)$ na Equação 10. O eixo y corresponde ao valor da função G (Equação 10) de acordo com a F-medida obtida na predição de cada tamanho de *labelset*.

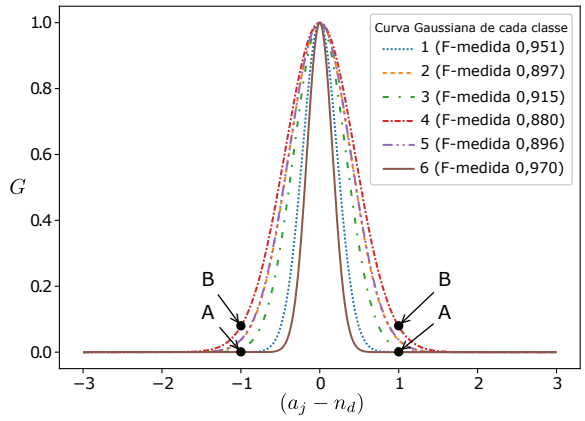


Figura 2: Funções gaussianas de cada classe em $\mathcal{Q}'$ baseada nas predições do metamodelo.

Na Figura 2, a F-medida obtida pelo metamodelo ao definir o tamanho 6 de *labelset* para as amostras foi alta ($F\text{-}medida = 0{,}970$) e, portanto, a função gaussiana do segundo gráfico chega a zero antes de alcançar algum número inteiro, como indicam os pontos A. Nesta situação, se o metamodelo apontar o tamanho 6 para um *labelset* de um documento desconhecido, por exemplo, é definido que a predição final irá conter seis rótulos. Por outro lado, o metamodelo não obteve a mesma capacidade preditiva ao estimar o tamanho 4 do *labelset* ($F\text{-}medida = 0{,}880$). Assim, sua função gaussiana retorna valores maiores que zero para tamanhos com diferença de um da estimativa do metamodelo, conforme indicam os pontos B. Isso significa que uma margem de segurança será usada para que o tamanho do *labelset* não esteja totalmente definido pela predição do metamodelo. Nesse caso, a função aponta a possibilidade da predição correta conter de 3 a 5 rótulos ($4-1$, 4 e $4+1$, respectivamente), ou seja, se o metamodelo apontar a classe 4 ao classificar um documento desconhecido, a predição final poderá conter 3, 4 ou 5 rótulos.

C. Processo 3: selecionando um grupo de labelsets

Nesse processo, é criado um conjunto $\mathcal{S}$ dos *labelsets* Y_j que apareceram nas amostras de treinamento, que contêm pelo menos uma das classes de $\mathcal{R}$ (como definido no Processo 1) e, de acordo com a quantidade de rótulos presentes no *labelset*, que devem satisfazer a desigualdade $G(\sigma_{n_d}, n_d, |Y_j|) > 0$ (com os parâmetros da curva definidos no Processo 2).

D. Processo 4: predizendo o labelset do documento d

Para identificar o conjunto de rótulos de predição do documento d, o tamanho de descrição é calculado novamente para cada *labelset* presente no grupo $\mathcal{S}$, ou seja, os *labelsets* são usados como os modelos na Equação 2 e, dessa forma, $L(d|Y_j)$ é calculado em relação a cada $Y_j \in \mathcal{S}$. Além disso, *labelsets* com tamanho muito diferente do predito pelo metamodelo são penalizados de acordo com o valor da função gaussiana. Os rótulos que estão presentes no *labelset* cujo modelo obteve o menor tamanho de descrição são escolhidos para a predição, definida pela seguinte equação:

$$Y = \underset{\forall_{Y_j \in S}}{\arg\min} \left[L(d|Y_j) \times [1 + \gamma_2(1 - G(\sigma_{n_d}, n_d, |Y_j|))]\right], \tag{12}$$

onde j é o índice do *labelsets* presente em $\mathcal{S}$ e γ_2 é o peso da penalidade máxima aplicado ao tamanho de descrição. Nessa dissertação, empiricamente, foi definido $\gamma_2 = 0{,}2$, ou seja, 20% do tamanho de descrição é aplicado como a penalidade máxima para *labelsets* com tamanho diferente do valor que foi predito pelo metamodelo.

Como as informações relacionadas aos termos, às classes e aos *labelsets* são extraídas de forma independente, o ML-MDLText também permite que novos termos e novas classes sejam apresentados ao longo do tempo. Essa é uma característica marcante e importante para cenários de aprendizado *online* em aplicações reais.

E. Análise assintótica

O pior cenário possível para o ML-MDLText executar é aquele composto por uma base de treinamento que contém todos os *labelsets* possíveis (ou seja, $|\mathcal{Q}^*| = 2^{|\mathcal{Q}|}$ e consequentemente, $|\mathcal{Q}'| = |\mathcal{Q}|$), por um valor de cm definido bem próximo de $|\mathcal{Q}|$ e por um metamodelo que tem baixo poder de predição na definição do tamanho dos *labelsets*. Essa configuração obriga avaliar todos os $2^{|\mathcal{Q}|}$ *labelsets* possíveis através do tamanho de descrição do último processo do ML-MDLText. Na prática, este cenário deve ocorrer muito raramente e, portanto, a complexidade média do ML-MDLText é bem melhor do que a apresentada no pior caso.

Na etapa de treinamento do ML-MDLText, os modelos das classes e dos *labelsets* e o metamodelo são construídos. De maneira geral, a complexidade do ML-MDLText na etapa de treinamento, no pior caso de execução, é $\mathcal{O}((2^{|\mathcal{Q}|} \times |\mathcal{T}| \times$

$\overline{n}) + f_{treinamento}(|\mathcal{T}|,\overline{n},|\mathcal{Q}|) + f_{classificacao}(|\mathcal{T}|,\overline{n},|\mathcal{Q}|))$, que varia significativamente conforme o número de classes e de *labelsets* aumenta e de acordo com a complexidade do método empregado para gerar o metamodelo. Na etapa de classificação, a complexidade do método no pior caso é $\mathcal{O}((2^{|\mathcal{Q}|} \times |d|) + f_{classificacao}(1,n,|\mathcal{Q}|))$, que também varia conforme o número de classes e de acordo com o método empregado para gerar o metamodelo.

É importante destacar que como na grande maioria dos problemas de classificação, o número de classes é bem menor do que o número de amostras e o número médio de termos, pode-se dizer que a etapa de treinamento do ML-MDLText, nesses casos, possui complexidade linear em relação ao número médio de termos e amostras, se o método de classificação usado para gerar o metamodelo também tiver complexidade linear. De maneira similar, a complexidade da etapa de classificação é linear em relação a quantidade de termos presente no documento, se o metamodelo também tiver complexidade linear nessa etapa.

V. Configuração Experimental

Para dar credibilidade aos resultados, foram realizados experimentos com bases de dados amplamente utilizadas em trabalhos relacionados [22], [42], [20], [12], [15], [38], [26], [17], [4], [43], [41]. Essas bases de dados contém diferentes características, em termos de domínio, número de documentos e classes, número de *labelsets*, cardinalidade e densidade, que podem influenciar no desempenho da classificação e são importantes para avaliar os métodos de classificação de maneira robusta. Elas são listadas na Tabela I.

Tabela I: Base de dados usadas nos experimentos.

Base de dados	m	n	q	q^*	card.	dens.	min.	max.
Reuters-ORGs	881	6.983	32	60	1,119	0,035	1	349
Reuters-places	18.798	34.169	147	889	1,212	0,008	1	12.541
RCV1-nivel1	27.974	64.286	4	15	1,168	0,292	4.150	12.752
RCV1-nivel2	27.047	61.439	54	1.239	1,425	0,026	7	4.753
RCV2-IT-nivel1	28.400	31.780	4	15	1,259	0,315	5.268	11.731
RCV2-IT-nivel2	27.954	31.176	43	356	1,267	0,029	1	4.477
RCV2-PT-nivel1	8.837	14.681	4	14	1,382	0,346	502	5.276
RCV2-PT-nivel2	8.773	14.582	37	218	1,613	0,044	1	2.937
RCV2-SP-nivel1	18.651	26.038	4	15	1,216	0,304	2.455	12.888
RCV2-SP-nivel2	18.463	25.551	44	284	1,357	0,031	2	6.475
Bibtex	7.395	1.836	159	2.856	2,402	0,015	51	1.042
Delicious	16.091	500	983	15.805	19,037	0,019	21	6.495
Enron	1.702	1.001	53	753	3,378	0,064	1	913
Medical	978	1.449	45	94	1,245	0,028	1	266
TMC2007	28.596	49.060	22	1.341	2,158	0,098	441	16.173

Descrição das bases de dados em termos de números de documentos (m). dimensão do espaço de atributos (n). número de classes (q). números de *labelsets* distintos (q^*). cardinalidade (card.). densidade (dens.). número mínimo (min.) e máximo (max.) de documentos por rótulos.

As duas primeiras bases de dados (Reuters-ORGs e Reuters-places) foram extraídas da coleção Reuters21578[2] da agência de notícias *Reuters*. As oito seguintes foram extraídas das coleções RCV1 e RCV2[3], que também contém notícias da agência *Reuters*. As demais bases de dados (Bibtex, Delicious, Enron, Medical e TMC2007) foram adquiridas através do *framework* `Mulan`[4]. Todos os procedimentos de pré-processamento efetuados nessas bases de dados está detalhado na dissertação.

[2]Coleção Reuters21578. Disponível em http://www.daviddlewis.com/resources/testcollections/reuters21578/ (Acessado em 20/06/2020).

[3]Coleções RCV1 e RCV2. Disponível em https://trec.nist.gov/data/reuters/reuters.html (Acessado em 20/06/2020).

[4]*Framework Mulan*. Disponível em http://mulan.sourceforge.net/datasets.html (Acessado em 20/06/2020).

A. Medidas e estratégias de avaliação

O método proposto foi avaliado em cenários de aprendizado *offline* e *online* com diferentes variações de *feedback*, com o intuito de analisar seu desempenho nas mais diferentes situações. Como a característica marcante do ML-MDLText é permitir o aprendizado *online*, serão detalhados neste artigo apenas os procedimentos efetuados e resultados obtidos nesse cenário de aprendizado, devido a limitação de espaço. Os resultados obtidos na etapa de classificação foram comparados com os resultados de outros métodos aplicados para a categorização nessas mesmas condições.

Para comparar o desempenho dos métodos, foram empregadas três medidas de avaliação, a *macro F-medida* (também conhecida como medida F1), a *perda de Hamming* e a *acurácia de subconjunto*, que fornecem uma visão do desempenho sob diferentes aspectos. A macro F-medida avalia a predição de cada rótulo individualmente através da F-medida e calcula a média macro sobre todos os rótulos para obter uma avaliação geral [3]. Já a acurácia de subconjunto observa a capacidade do classificador em predizer todo o labelset do documento, ou seja, é preciso predizer todos os rótulos associados ao documento para computar um acerto com essa medida. Ela é muito rigorosa, já que predições parcialmente corretas são desconsideradas. Métodos que exploram a dependência entre os rótulos de forma condicional, como a transformação LP, tentam maximizar essa medida. Em contra partida, a perda de Hamming é uma medida que considera predições parcialmente corretas em seu cálculo. Ela leva em consideração tanto os erros de predição quanto as omissões de predição, pois analisa a quantidade de vezes em que um rótulo que pertence ao documento não é predito ou vice-versa [42], [3]. Métodos que ignoram a existência de dependência entre as classes, como a transformação BR ou o ML-KNN [44], [3], tentam minimizar essa medida.

Em adição a análise de desempenho dos métodos através da observação das medidas de avaliação, foi aplicado também o teste estatístico não-paramétrico de Friedman. Esse teste verifica se a hipótese nula, de que todos os métodos são equivalentes, pode ser rejeitada [45]. Se a hipótese nula for rejeitada pelo teste de Friedman, é executada uma comparação par-a-par usando o teste *post-hoc* de Bonferroni-Dunn para identificar se existem evidências estatísticas suficientes para apontar diferenças de desempenho entre o método proposto e os outros métodos em alguma das medidas avaliadas [45]. Foi considerado $\alpha = 0.05$ para a execução dos testes estatísticos.

B. Cenários avaliados

Nos experimentos executados em cenário de aprendizado *offline*, um único lote de amostras rotuladas foi apresentado ao método na etapa de treinamento para a construção do modelo de predição. O modelo gerado foi então empregado para categorizar um conjunto de amostras e foi avaliado de acordo o seu desempenho nessa rotulação. Os resultados foram obtidos por meio de validação cruzada 5-*fold* usando a abordagem de estratificação proposta por Sechidis *et al.* [46] para problemas de classificação multirrótulo e foram comparados com outros métodos frequentemente empregados em tarefas de classificação de textos. Os outros detalhes relacionados a execução dos experimentos em cenário de aprendizado *offline* estão descritos na dissertação.

Para simular cenários de aprendizado *online*, um pequeno número de exemplos (20% dos documentos de cada classe) foi usado para o treinamento inicial. O modelo gerado foi então utilizado para categorizar um conjunto de amostras não rotuladas, através da conhecida abordagem *prequential* (também conhecida como teste-então-treina intercalado) [47]. Nessa abordagem, um documento por vez é apresentado para testar o classificador e o conjunto de rótulos preditos é armazenado para o uso posterior do cálculo de desempenho. Após algum erro no teste, o verdadeiro conjunto de rótulos do documento é apresentado e o método de classificação pode atualizar o seu modelo preditivo. A Figura 3 esquematiza o procedimento executado para a simulação de cenários de aprendizado *online*.

Para simular cenários reais, a forma com que o *feedback* é apresentado ao classificador foi variada nos experimentos, como descrito a seguir. A Figura 3 também sumariza esses três esquemas de *feedbacks* empregados.

- ***Feedback* Imediato:** após errar todo o *labelset* do documento, o método de classificação recebe imediatamente o *feedback* e atualiza seu modelo preditivo. Esse experimento simula um cenário ideal, no qual os usuários apresentam um *feedback* instantaneamente após o classificador cometer um erro.
- ***Feedback* Incerto:** apesar do classificador errar todo o *labelset* do documento, nem sempre um *feedback* é enviado. Esse experimento simula usuários que não checam todas as predições do classificador.
- ***Feedback* Atrasado:** após errar todo o *labelset* do documento, o método de classificação recebe com atraso o *feedback* para atualizar o modelo preditivo. Esse experimento simula usuários que levam um tempo para avaliar o resultado da predição.

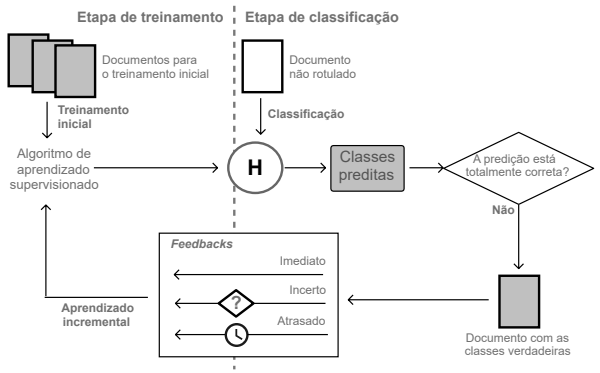


Figura 3: Esquema para a simulação de cenários de aprendizado *online* com diferentes *feedbacks* [41].

Esquema semelhante a esse foi empregado também em estudos anteriores para avaliar abordagens de aprendizado *online*, como nos trabalhos de Almeida e Yamakami [9], Cormack e Lynam [48] e Bittencourt *et al.* [41]. Nos experimentos com o *feedback* incerto e atrasado, a decisão se o *feedback* será enviado ou o tempo de espera são definidos de forma aleatória para cada documento classificado incorretamente. Nos experimentos com o *feedback* imediato, o *feedback* é enviado para todos os documentos classificados incorretamente, logo após a serem apresentados para classificação. As partições de treino e de teste são determinadas através da abordagem estratificada para dados multirrotulados proposto por Sechidis *et al.* [46]. A média de cinco execuções, variando-se a partição de treino e teste, foi utilizada para a avaliação do desempenho do método.

O método proposto foi comparado com os seguintes métodos de classificação *online*: *multinomial naïve Bayes* (M.NB) [49], *Bernoulli naïve Bayes* (B.NB) [49], gradiente descendente estocástico (*stochastic gradient descent* - SGD) [50], passivo-agressivo (*passive-aggressive* - (PA) [51] e perceptron (Perc.) [52]. Como esses métodos *baselines* não são naturalmente multirrótulo, eles foram empregados com o método de transformação BR e CC. O método proposto também foi comparado com o MLP [53], que suporta amostras multirrotuladas sem ser necessário aplicar técnicas de transformação.

Nos experimentos com o *naïve* Bayes, foi necessário utilizar o esquema de representação espaço vetorial com peso binário em todas as bases de dados, pois, esses métodos foram criados para manipular atributos discretos [53]. Para os demais métodos, foi utilizado o esquema de peso TF-IDF [54]. Como foram avaliados métodos *online* e o esquema TF-IDF necessita de informações sobre os documentos apresentados para o treinamento, o processo de conversão para peso TF-IDF também foi aplicado de forma incremental.

VI. Resultados

Nos experimentos com cenários de aprendizado *offline*, o ML-MDLText obteve desempenho relevante em macro F-medida e acurácia de subconjunto, e esteve entre os cinco melhores avaliados em um contexto geral, acima de métodos como ML-KNN, BRkNNa, *naïve* Bayes e árvores de decisão. Os métodos de transformação que empregaram o método de classificação SVM Linear obtiveram melhores pontuações do que o ML-MDLText em todas as medidas avaliadas. Esse comportamento já era esperado, visto que o SVM apresentou desempenho semelhante em comparação com o MDLText nos experimentos em aprendizado monorrótulo, como apresentado por Silva *et al.* [7] e, além disso, métodos de aprendizado *offline* geralmente obtém melhores resultados que os métodos de aprendizado *online* quando eles podem ser aplicados [55], [56]. Apesar de não ter sido apontado como o melhor dentre os conjuntos avaliados, o ML-MDLText tem como vantagem a característica de ser naturalmente multirrótulo e de permitir o aprendizado *online*. Mais detalhes sobre os resultados obtidos com os experimentos em cenários de aprendizado *offline* podem ser conferidos na dissertação.

Com relação aos experimentos executados com cenários de aprendizado *online*, os resultados obtidos pelo ML-MDLText foram ainda mais relevantes. A Tabela II apresenta as médias da macro F-medida, da perda de Hamming e da acurácia de subconjunto obtidas nas execuções de cada método e em cada base de dados em cenários de aprendizado *online* com *feedback* imediato. Para facilitar a comparação dos resultados, as medidas são apresentadas em um mapa de calor em tons de cinza, em que quanto melhor a medida, mais escuro é o tom utilizado. Além disso, a melhor medida obtida em cada base de dados está destacada em negrito.

A exploração da dependência entre os rótulos através da transformação CC não trouxe vantagens no aprendizado *online*,

Tabela II: Resultados obtidos por cada método e em cada base de dados em cenário de aprendizado *online* com *feedback* imediato.

Base de dados	Transformação BR					Transformação CC						
	M.NB	B.NB	SGD	PA	Perceptron	M.NB	B.NB	SGD	PA	Perceptron	MLP	ML-MDLText
Macro F-medida												
Reuters-ORGs	0,671	0,453	0,868	0,857	0,800	0,171	0,456	0,763	0,780	0,682	0,670	***0,873***
Reuters-places	0,123	0,059	0,634	***0,731***	0,683	0,018	0,059	0,545	0,657	0,506	0,537	0,674
RCV1-nivel1	0,851	0,815	***0,901***	0,895	0,870	0,761	0,816	0,879	0,860	0,844	0,888	0,875
RCV1-nivel2	0,337	0,258	0,488	***0,629***	0,587	0,087	0,260	0,507	0,567	0,501	0,562	0,624
RCV2-IT-nivel1	0,848	0,846	***0,892***	0,881	0,859	0,851	0,847	0,875	0,854	0,841	0,877	0,860
RCV2-IT-nivel2	0,392	0,282	0,444	0,610	0,582	0,153	0,284	0,483	0,565	0,495	0,516	***0,648***
RCV2-PT-nivel1	0,828	0,809	0,891	***0,893***	0,868	0,786	0,809	0,857	0,853	0,835	0,887	0,868
RCV2-PT-nivel2	0,405	0,316	0,534	0,585	0,581	0,231	0,317	0,524	0,551	0,491	0,541	***0,641***
RCV2-SP-nivel1	0,868	0,850	***0,901***	0,896	0,871	0,833	0,851	0,882	0,868	0,852	0,890	0,877
RCV2-SP-nivel2	0,376	0,250	0,473	0,597	0,575	0,155	0,251	0,484	0,546	0,471	0,499	***0,644***
Bibtex	0,207	0,173	0,187	0,259	0,275	0,066	0,142	0,197	0,238	0,220	0,185	***0,328***
Delicious	0,098	0,078	0,060	0,093	0,108	0,046	0,041	0,071	0,092	0,089	0,059	***0,150***
Enron	***0,235***	0,181	0,215	0,207	0,220	0,127	0,182	0,175	0,180	0,172	0,150	0,230
Medical	0,490	0,236	0,679	***0,688***	0,618	0,116	0,235	0,599	0,622	0,536	0,172	0,670
TMC2007	0,482	0,432	0,497	***0,583***	0,530	0,126	0,435	0,512	0,522	0,491	0,577	0,559
Perda de Hamming												
Reuters-ORGs	0,037	0,060	0,024	***0,022***	0,035	0,079	0,060	0,042	0,038	0,053	0,038	0,026
Reuters-places	0,007	0,009	***0,003***	***0,003***	0,004	0,008	0,009	0,004	0,004	0,006	***0,003***	0,005
RCV1-nivel1	0,078	0,098	***0,049***	0,052	0,066	0,084	0,097	0,060	0,071	0,079	0,057	0,064
RCV1-nivel2	0,026	0,027	0,012	***0,011***	0,016	0,023	0,027	0,013	0,015	0,018	0,012	0,014
RCV2-IT-nivel1	0,096	0,096	***0,065***	0,072	0,085	0,084	0,095	0,075	0,089	0,096	0,074	0,083
RCV2-IT-nivel2	0,024	0,023	0,013	***0,012***	0,016	0,022	0,023	0,014	0,016	0,019	0,013	0,014
RCV2-PT-nivel1	0,072	0,080	***0,046***	0,047	0,058	0,060	0,080	0,059	0,063	0,070	0,049	0,056
RCV2-PT-nivel2	0,035	0,037	***0,016***	***0,016***	0,022	0,030	0,037	0,020	0,021	0,026	0,017	0,020
RCV2-SP-nivel1	0,066	0,072	***0,050***	0,053	0,064	0,071	0,072	0,060	0,066	0,073	0,056	0,062
RCV2-SP-nivel2	0,021	0,022	***0,012***	***0,012***	0,015	0,018	0,022	0,013	0,015	0,017	***0,012***	0,014
Bibtex	0,060	0,065	***0,012***	***0,012***	0,021	0,014	0,102	0,014	0,015	0,021	0,013	0,017
Delicious	0,052	0,169	***0,018***	0,020	0,030	0,073	0,679	0,021	0,027	0,030	***0,018***	0,028
Enron	0,131	0,228	0,066	0,063	0,087	0,076	0,223	0,081	0,081	0,089	***0,060***	0,081
Medical	0,043	0,048	0,030	***0,026***	0,039	0,051	0,048	0,039	0,035	0,044	0,047	0,035
TMC2007	0,070	0,077	***0,058***	0,061	0,076	0,074	0,078	0,062	0,074	0,080	0,062	0,068
Acurácia de Subconjunto												
Reuters-ORGs	0,673	0,479	0,791	0,808	0,718	0,300	0,481	0,718	0,742	0,658	0,685	***0,814***
Reuters-places	0,627	0,570	0,840	***0,852***	0,779	0,570	0,573	0,803	0,805	0,743	0,809	0,805
RCV1-nivel1	0,742	0,692	***0,838***	0,828	0,790	0,743	0,696	0,814	0,784	0,764	0,816	0,821
RCV1-nivel2	0,323	0,305	0,573	***0,612***	0,525	0,198	0,307	0,579	0,552	0,493	0,585	0,604
RCV2-IT-nivel1	0,690	0,692	***0,784***	0,763	0,726	0,735	0,696	0,769	0,732	0,716	0,758	0,761
RCV2-IT-nivel2	0,416	0,413	0,599	0,640	0,571	0,314	0,420	0,630	0,611	0,563	0,621	***0,666***
RCV2-PT-nivel1	0,773	0,750	***0,852***	***0,852***	0,815	0,814	0,753	0,819	0,810	0,791	0,847	0,847
RCV2-PT-nivel2	0,516	0,472	0,698	0,713	0,643	0,421	0,477	0,673	0,664	0,618	0,699	***0,724***
RCV2-SP-nivel1	0,786	0,772	***0,834***	0,826	0,796	0,781	0,774	0,811	0,793	0,778	0,819	0,825
RCV2-SP-nivel2	0,551	0,523	0,715	0,728	0,667	0,542	0,523	0,708	0,693	0,651	0,709	***0,744***
Bibtex	0,056	0,060	0,143	0,161	0,087	0,083	0,061	0,135	0,137	0,101	0,128	***0,194***
Delicious	0,000	0,000	0,001	0,001	0,000	0,000	0,000	0,001	0,000	0,000	0,001	***0,010***
Enron	0,014	0,002	0,092	0,101	0,044	0,060	0,003	0,087	0,085	0,069	***0,106***	0,103
Medical	0,416	0,289	0,555	***0,593***	0,463	0,164	0,297	0,505	0,548	0,454	0,236	0,577
TMC2007	0,238	0,214	***0,298***	0,274	0,198	0,167	0,215	0,281	0,219	0,195	0,270	0,255

visto que os classificadores CC obtiveram resultados inferiores aos resultados obtidos pelos métodos de transformação BR, que ignoram qualquer tipo dependência, principalmente quando foi empregado o M.NB como classificador base.

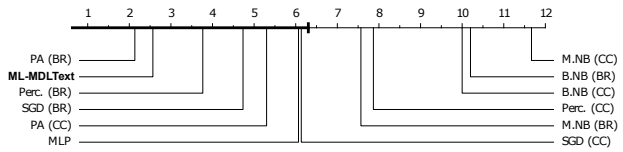
Na Tabela II, todos os métodos, inclusive o ML-MDLText, obtiveram valores de macro F-medida abaixo de 0,4 nos conjuntos de dados *Bibtex*, *Delicious* e *Enron*. Esses resultados provavelmente são consequências da alta quantidade de rótulos e da alta cardinalidade dessas bases, que acentuam ainda mais o desafio da classificação com esse tipo de aprendizado. O ML-MDLText obteve as melhores macro F-medida em seis bases de dados, inclusive na *Bibtex* e *Delicious*, que apresentam alto grau de dificuldade na classificação. PA obteve destaque em cinco base de dados, SGD em três e M.NB em uma.

Considerando a perda de Hamming, as medidas obtidas pelo ML-MDLText foram superiores aos métodos NB em todos os experimentos executados, mas inferiores aos resultados de SGD e PA. Esses resultados eram esperados, já que a transformação BR minimiza a perda de Hamming [44].

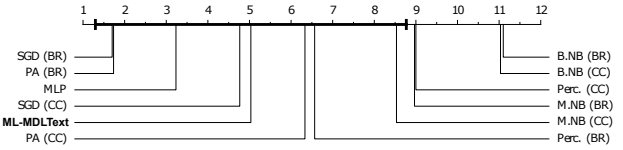
Por fim, de acordo com a acurácia de subconjunto, o método ML-MDLText obteve a melhor avaliação em seis conjuntos em um contexto geral. Embora o valor de acurácia de subconjunto de todos os métodos tenha sido relativamente baixo em *Bibtex* e *Delicious*, o ML-MDLText também obteve a melhor avaliação dentre os métodos nessas bases, que contém um número excessivamente grande de classes e de *labelsets* distintos. SGD, PA e MLP também obtiveram destaque em cinco, quatro e em uma base, respectivamente. Embora o SGD tenha obtido bons resultados em perda de Hamming, em geral, seus resultados de macro F-medida e acurácia de subconjunto foram inferiores aos resultados do ML-MDLText em muitos conjuntos de dados.

Para verificar se existem evidências estatísticas que comprovam a superioridade de algum método sobre os demais, foi executada uma análise estatística usando o teste não-paramétrico de Friedman. Com um intervalo de confiança $\alpha = 0.05$, o teste descartou a hipótese nula e indicou que existem diferenças estatísticas significativas entre os resultados obtidos em todas as medidas avaliadas. Dessa forma, foi executada uma comparação par-a-par usando o teste *post-hoc* de Bonferroni-Dunn para identificar se existem evidências estatísticas que indicam a superioridade do ML-MDLText sobre os métodos em alguma das medidas de desempenho. Nas Figuras 4(a), 4(b) e 4(c), os métodos são posicionados de acordo com seu *ranking* médio. A linha horizontal mais escura liga os métodos que não apresentaram diferenças estatísticas significativas em relação ao desempenho do ML-MDLText. Desconectados dessa linha, os métodos posicionados à direita ou à esquerda apresentaram resultados estatisticamente superiores e inferiores, respectivamente, com diferença estatística significativa.

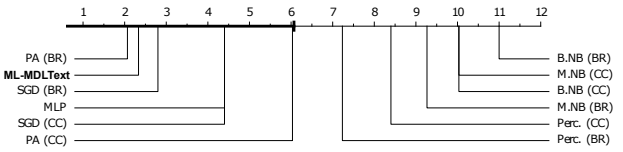
Nas Figuras 4(a), 4(b) e 4(c) é possível observar que, sem utilizar técnicas de transformação, o ML-MDLText alcançou o segundo melhor *ranking* médio em macro F-medida e em acurácia de subconjunto, logo atrás do PA. Em todas as medidas de desempenho, o teste estatístico indicou que existem evidências estatísticas para afirmar que o ML-MDLText obteve



(a) Macro F-medida.



(b) Perda de Hamming.



(c) Acurácia de subconjunto.

Figura 4: *Ranking* médio e diferença crítica calculada usando o teste post-hoc Bonferroni-Dunn para as três medidas de avaliação considerando o *feedback* imediato.

desempenho superior ao método B.NB, foi melhor que o M.NB em termos de macro F-medida e que o M.NB e Perceptron em termos de acurácia de subconjunto. Em relação à perda de Hamming, embora SGD, PA e MLP tenham obtido melhor *ranking* médio, não há diferenças estatísticas para afirmar que esses métodos foram superiores ao ML-MDLText.

De forma resumida, os resultados obtidos com o *feedback* incerto e atrasado foram bem similares aos resultados obtidos pelo *feedback* imediato e, por essa razão e pela limitação de espaço, apenas um descritivo com as diferenças encontradas são apresentadas. No *feedback* incerto, os resultados obtidos foram ligeramente inferiores, mas foram condizentes em relação aos resultados obtidos com o *feedback* imediato: o método ML-MDLText dominou as melhores macro F-medidas enquanto que SGD e o PA obtiveram os melhores valores de perda de Hamming. Esses três métodos também obtiveram os melhores resultados em relação à acurácia de subconjunto. De acordo com a macro F-medida e em comparação com o *feedback* imediato, o método PA foi o mais prejudicado nesse cenário onde nem sempre ocorre a retroalimentação para o treinamento incremental, pois ele perdeu duas das suas melhores avaliações. O ML-MDLText se mostrou mais estável e manteve suas melhores avaliações mesmo nesse cenário mais desafiante. Já no *feedback* atrasado, foi possível verificar uma piora no desempenho dos métodos Perceptron, SGD e PA com transformação CC. Provavelmente, o atraso na correção dos classificadores fez com que fosse necessário usar estimativas incorretas da cadeia de classificadores para determinar a presença ou a ausência de um determinado rótulo, o que trouxe prejuízos para a execução dos métodos de transformação CC. Na dissertação, os resultados obtidos nesses cenários são apresentados de forma mais detalhada juntamente com os testes estatísticos efetuados.

VII. Conclusão

Nesse estudo foi apresentado o ML-MDLText, um novo método de classificação de textos multirrótulo. Esse método foi projetado tendo como base as principais características de um método ideal para manipular os problemas de classificação de textos no contexto atual: deve ter a habilidade de manipular um grande volume de documentos de texto e se adequar as mudanças nos padrões encontrados nos textos, conforme novos documentos são criados. Ainda, o ML-MDLText é capaz de extrair mais conceitos do texto e oferecer uma predição com múltiplos rótulos, o que é útil em uma infinidade de aplicações.

Essa proposta é uma adaptação do método de classificação de textos MDLText para problemas multirrotulados. Enquanto que no MDLText, o cálculo do tamanho de descrição é utilizado para definir uma única classe de predição, no método proposto esse cálculo é utilizado (*i*) para definir a relevância das classes, e (*ii*) para definir a relevância dos *labelsets* e, assim, definir não apenas uma, mas um conjunto de classes de predição. Nessa abordagem multirrótulo, o número de rótulos a ser predito é estimado por um metamodelo, treinado com metadados extraídos dos dados de treinamento multirrótulo, e através de funções gaussianas, que definem o grau de confiança da saída do metamodelo. De acordo com a definição do método, a correlação entre os rótulos é explorada de forma condicional, pois a relevância do conjunto de rótulos mais apropriado é identificada levando em consideração os atributos do documento a ser rotulado.

Foi conduzida uma avaliação abrangente usando 15 bases de dados que são referências na literatura multirrótulo. O desempenho do ML-MDLText foi avaliado no tradicional cenário de aprendizado *offline* e também em simulações de cenários de aprendizado *online* reais, na qual o treinamento incremental deve ser executado para corrigir predições incorretas do classificador. Foram exploradas também diferentes possibilidades de *feedbacks* para avaliar a robustez do ML-MDLText nas mais diferentes situações. Os resultados obtidos pelo ML-MDLText foram comparados com o desempenho de outros métodos adaptados para a classificação multirrótulo, através de três medidas de avaliação específicas para o problema (macro F-medida, perda de Hamming e acurácia de subconjunto) e de testes estatísticos.

De acordo com os resultados experimentais, o ML-MDLText sempre esteve entre os dois melhores métodos em todos os cenários de aprendizado *online* quando a macro F-medida e a acurácia de subconjunto foram usadas como medidas de desempenho. Considerando a perda de Hamming, o ML-MDLText esteve entre os cinco melhores métodos nos cenários de aprendizado *online*, isso porque, muitos dos métodos empregados na comparação minimizam essa medida. Alguns métodos, como o SGD e o MLP, obtiveram uma ótima posição de *ranking* médio em perda de Hamming, no entanto ocuparam posições inferiores ao ML-MDLText em macro F-medida e acurácia de subconjunto em cenários de aprendizado *online*. O ML-MDLText obteve ainda destaque na macro F-

Medida em conjuntos de dados mais desafiadores, que contém grande quantidade de rótulos e alta cardinalidade.

Em resposta a principal pergunta de pesquisa deste trabalho, é notório que o ML-MDLText é um método de classificação multirrótulo robusto e eficiente. Além de ter obtido um desempenho notável nos experimentos, o ML-MDLText é capaz de manipular naturalmente documentos multirrotulados, sem exigir que o problema multirrótulo seja transformado em problemas de rótulo único. Portanto, ele possui um custo computacional muito menor que o PA, SGD e Perceptron empregados com transformação de problema, principalmente em tarefas com grande número de classes ou quando o classificador base é custoso computacionalmente. Ainda, o ML-MDLText mostrou ser estável e apresentou resultados competitivos com o de outros métodos avaliados em diferentes cenários e domínios de problemas de classificação de textos. Isso faz com que ele seja uma opção mais vantajosa em situações com um número elevado de amostras ou com fluxo contínuo de dados.

A seguir, são listadas algumas sugestões de trabalhos futuros:

- **Desenvolver novas opções de definição do tamanho do *labelset* de predição.** Uma sugestão de trabalho futuro seria definir e avaliar outras técnicas para determinar o tamanho do *labelset* sem o uso de metamodelo ou de classificadores auxiliares.
- **Empregar novas funções para pontuar termos.** Acredita-se que a criação de uma técnicas de atribuição de pontuação para termos (para o cálculo de K no tamanho de descrição), específica para a seleção de atributos em problemas de classificação multirrótulo, pode melhorar ainda mais o desempenho do ML-MDLText.
- **Avaliar o método proposto em problemas nos quais a dimensão do espaço de termos e de classes é variável.** Como explicado no Capítulo IV, o ML-MDLText é capaz de atualizar seu modelo preditivo conforme novas instâncias, com novos termos e novas classes, são apresentadas ao longo do tempo. Dessa forma, outro rumo de pesquisa seria explorar essa propriedade excepcional, que raramente é encontrada em métodos de classificação e é muito útil em problemas de classificação reais, dinâmicos e de larga escala.
- **Aplicar o método proposto na resolução de problemas de domínios específicos.** Outra opção de trabalho futuro seria explorar o desempenho do ML-MDLText na classificação multirrótulo em algum problema específico de classificação de textos multirrótulo, como na categorização de e-mails, na classificação de emoções, na sugestão de *tag*, no diagnóstico médico ou na rotulação de documentos e páginas da *web*, por exemplo.
- **Adaptar o método proposto para classificar documentos usando representação vetorial distribuída (*word embeddings*)**. Pela forma que o ML-MDLText foi modelado, é necessário que os documentos sejam apresentados usando o modelo *bag-of-words*. Acredita-se que o método proposto poderia apresentar resultados ainda melhores do que os apresentados neste trabalho se for adaptado para classificar documentos usando modelos de representação mais recentes, como a representação vetorial distribuída.
- **Adaptar o método proposto para tarefas de classificação hierárquicas.** Uma outra alternativa para trabalhos futuros seria adaptar o ML-MDLText para tarefas hierárquicas multirrótulo, nas quais um documento pode ter múltiplas classes organizadas hierarquicamente.

Referências

[1] A. C. P. L. F. de Carvalho and A. A. Freitas, "A tutorial on multi-label classification techniques," in *Foundations of Computational Intelligence Volume 5: Function Approximation and Classification*, A. Abraham, A.-E. Hassanien, and V. Snášel, Eds. Springer Berlin Heidelberg, Jul. 2009, pp. 177–195.

[2] G. Tsoumakas, I. Katakis, and I. Vlahavas, "Mining multi-label data," in *Data Mining and Knowledge Discovery Handbook*, O. Maimon and L. Rokach, Eds. Boston, MA: Springer US, 2010, pp. 667–685.

[3] E. Gibaja and S. Ventura, "A tutorial on multilabel learning," *ACM Computing Surveys (CSUR)*, vol. 47, no. 3, pp. 1–38, Apr. 2015.

[4] E. Alvares-Cherman, J. Metz, and M. C. Monard, "Incorporating label dependency into the binary relevance framework for multi-label classification," *Expert Systems with Applications*, vol. 39, no. 2, pp. 1647–1655, Feb. 2012.

[5] J. Zhang, C. Li, Z. Sun, Z. Luo, C. Zhou, and S. Li, "Towards a unified multi-source-based optimization framework for multi-label learning," *Applied Soft Computing*, vol. 76, pp. 425 – 435, 2019.

[6] J. Read, A. Bifet, G. Holmes, and B. Pfahringer, "Scalable and efficient multi-label classification for evolving data streams," *Machine Learning*, vol. 88, no. 1, pp. 243–272, Jul. 2012.

[7] R. M. Silva, T. A. Almeida, and A. Yamakami, "MDLText: An efficient and lightweight text classifier," *Knowledge-Based Systems*, vol. 118, pp. 152–164, Feb. 2017.

[8] T. A. Almeida, A. Yamakami, and J. Almeida, "Filtering spams using the minimum description length principle," in *Proceedings of the 2010 ACM Symposium on Applied Computing*, ser. SAC '10. New York, NY, USA: ACM, Mar. 2010, pp. 1854–1858.

[9] T. A. Almeida and A. Yamakami, "Facing the spammers: A very effective approach to avoid junk e-mails," *Expert Systems with Applications*, vol. 39, no. 7, pp. 6557–6561, Jun. 2012.

[10] B. L. de Freitas, R. M. Silva, and T. A. Almeida, "Gaussian mixture descriptors learner," *Knowledge-Based Systems*, vol. 188, pp. 1–9, 2020.

[11] J. Rissanen, "Modeling by shortest data description," *Automatica*, vol. 14, no. 5, pp. 465–471, Sep. 1978.

[12] M.-L. Zhang and Z.-H. Zhou, "Multilabel neural networks with applications to functional genomics and text categorization," *IEEE Transactions on Knowledge and Data Engineering*, vol. 18, no. 10, pp. 1338–1351, Oct. 2006.

[13] M.-L. Zhang and Z.-H. Zhou., "ML-KNN: A lazy learning approach to multi-label learning," *Pattern Recognition*, vol. 40, no. 7, pp. 2038–2048, Jul. 2007.

[14] M. R. Boutell, J. Luo, X. Shen, and C. M. Brown, "Learning multi-label scene classification," *Pattern Recognition*, vol. 37, no. 9, pp. 1757–1771, Sep. 2004.

[15] E. L. Mencía and J. Fürnkranz, "Pairwise learning of multilabel classifications with perceptrons," in *Proceedings of the 2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*. Hong Kong, China: IEEE, Jun. 2008, pp. 2899–2906.

[16] A. Clare and R. D. King, "Knowledge discovery in multi-label phenotype data," in *Proceedings of the 5th European Conference on Principles of Data Mining and Knowledge Discovery*, L. D. Raedt and A. Siebes, Eds. Springer Berlin Heidelberg, 2001, pp. 42–53.

[17] Z. Younes, F. Abdallah, T. Denœux, and H. Snoussi, "A dependent multilabel classification method derived from the k-nearest neighbor rule," *EURASIP Journal on Applied Signal Processing*, vol. 2011, no. 1, pp. 1–14, Mar. 2011.

[18] E. Spyromitros, G. Tsoumakas, and I. Vlahavas, "An empirical study of lazy multilabel classification algorithms," in *Proceedings of the 5th Hellenic Conference on Artificial Intelligence: Theories, Models and Applications*, ser. SETN '08. Syros, Greece: Springer Berlin, Heidelberg, 2008, pp. 401–406.

[19] A. Elisseeff and J. Weston, "A kernel method for multi-labelled classification," in *Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic*, ser. NIPS'01, T. G. Dietterich, S. Becker, and Z. Ghahramani, Eds. Cambridge, MA, USA: The MIT Press, 2002, vol. 1, pp. 681–687.

[20] K. Crammer and Y. Singer, "A family of additive online algorithms for category ranking," *Journal of Machine Learning Research*, vol. 3, pp. 1025–1058, Feb. 2003.

[21] E. L. Mencía and J. Fürnkranz, "Efficient pairwise multilabel classification for large-scale problems in the legal domain," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, ser. ECML PKDD '08, W. Daelemans, B. Goethals, and K. Morik, Eds. Antwerp, Belgium: Springer Berlin Heidelberg, Sep. 2008, pp. 50–65.

[22] A. K. McCallum, "Multi-label text classification with a mixture model trained by EM," in *AAAI 99 Workshop on Text Learning*, Jul. 1999.

[23] M.-L. Zhang, J. M. Peña, and V. Robles, "Feature selection for multi-label naïve Bayes classification," *Information Sciences*, vol. 179, no. 19, pp. 3218–3229, 2009.

[24] W. Qu, Y. Zhang, J. Zhu, and Q. Qiu, "Mining multi-label concept-drifting data streams using dynamic classifier ensemble," in *Advances in Machine Learning (ACML 2009)*, Z.-H. Zhou and T. Washio, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 308–321.

[25] E. Spyromitros-Xioufis, M. Spiliopoulou, G. Tsoumakas, and I. Vlahavas, "Dealing with concept drift and class imbalance in multi-label stream classification," *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence*, vol. 2, pp. 1583–1588, Jul. 2011.

[26] X. Zhang, T. Graepel, and R. Herbrich, "Bayesian online learning for multi-label and multi-variate performance measures," in *Proceedings of the 30th International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, Y. W. Teh and M. Titterington, Eds., vol. 9. Chia Laguna Resort, Sardinia, Italy: PMLR, May 2010, pp. 956–963.

[27] J. Read, A. Bifet, G. Holmes, and B. Pfahringer, "Efficient multi-label classification for evolving data streams," *2010 Working Papers*, may 2012.

[28] A. N. Kolmogorov, "Three approaches to the quantitative definition of information," *Problems of Information Transmission*, vol. 1, pp. 1–7, 1965.

[29] P. D. Grünwald, *The Minimum Description Length Principle*. The MIT Press, Mar. 2007.

[30] M. H. Hansen and B. Yu, "Model selection and the principle of minimum description length," *Journal of the American Statistical Association*, vol. 96, no. 454, pp. 746–774, Jun. 2001.

[31] T. M. Cover and P. E. Hart, "Nearest neighbor pattern classification," *IEEE Transaction on Information Theory*, vol. 13, no. 1, pp. 21–27, Jan. 1967.

[32] T. A. Almeida and A. Yamakami, "Advances in spam filtering techniques," in *Computational Intelligence for Privacy and Security*, ser. Studies in Computational Intelligence. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 199–214.

[33] P. Murena and A. Cornuéjols, "Minimum description length principle applied to structure adaptation for classification under concept drift," in *2016 International Joint Conference on Neural Network (IJCNN)*, 2016, pp. 2842–2849.

[34] M. Mehta, J. Rissanen, and R. Agrawal, "MDL-based decision tree pruning," in *Proceedings of the First International Conference on Knowledge Discovery and Data Mining*, ser. KDD'95, vol. 21, no. 2. Montréal, Québec, Canada: AAAI Press, 1995, pp. 216–221.

[35] L. Blier and Y. Ollivier, "The description length of deep learning models," in *Advances in Neural Information Processing Systems 31*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds. Curran Associates, Inc., 2018, pp. 2216–2226.

[36] P. Murena, A. Cornuéjols, and J. Dessalles, "Incremental learning with the minimum description length principle," in *2017 International Joint Conference on Neural Networks (IJCNN)*, 2017, pp. 1908–1915.

[37] K. Laghmari, C. Marsala, and M. Ramdani, "An adapted incremental graded multi-label classification model for recommendation systems," *Progress in Artificial Intelligence*, vol. 7, no. 1, pp. 15–29, Mar. 2018.

[38] L. Tang, S. Rajan, and V. K. Narayanan, "Large scale multi-label classification via metalabeler," in *Proceedings of the 18th International Conference on World Wide Web*, ser. WWW '09. New York, NY, USA: ACM, 2009, pp. 211–220.

[39] C. E. Shannon, "A mathematical theory of communication," *The Bell System Technical Journal*, vol. 27, pp. 379–423, 623–656, Jul. 1948.

[40] F. Assis, W. Yerazunis, C. Siefkes, and S. Chhabra, "Exponential differential document count – a feature selection factor for improving Bayesian filters accuracy," in *Proc. 2006 MIT Spam Conf. (SP'06)*, vol. 535, Cambridge, MA, USA, 2006, pp. 1–6.

[41] M. M. Bittencourt, R. M. Silva, and T. A. Almeida, "ML-MDLText: A multilabel text categorization technique with incremental learning," in *2019 8th Brazilian Conference on Intelligent Systems (BRACIS'19)*. Salvador, BA, Brasil: IEEE, Oct. 2019, pp. 580–585.

[42] R. E. Schapire and Y. Singer, "BoosTexter: A Boosting-based system for text categorization," *Machine Learning*, vol. 39, no. 2-3, pp. 135–168, May 2000.

[43] G. Madjarov, D. Kocev, D. Gjorgjevikj, and S. Džeroski, "An extensive experimental comparison of methods for multi-label learning," *Pattern Recognition*, vol. 45, no. 9, pp. 3084–3104, Sep. 2012.

[44] K. Dembczyński, W. Waegeman, W. Cheng, and E. Hüllermeier, "On label dependence and loss minimization in multi-label classification," *Machine Learning*, vol. 88, no. 1, pp. 5–45, jul 2012.

[45] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *Journal of Machine Learning Research*, vol. 7, pp. 1–30, Jan. 2006.

[46] K. Sechidis, G. Tsoumakas, and I. Vlahavas, "On the stratification of multi-label data," in *Proceedings of the 2011 European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD'11)*. Athens, Greece: Springer Berlin, Heidelberg, Aug. 2011, pp. 145–158.

[47] J. Gama, R. Sebastião, and P. P. Rodrigues, "On evaluating stream learning algorithms," *Machine Learning*, vol. 90, no. 3, pp. 317–346, Mar. 2013.

[48] G. V. Cormack and T. R. Lynam, "TREC 2006 spam track overview," in *TREC-2006: Fifteenth Text REtrieval Conference*. Gaithersburg, Maryland, USA: NIST Special Publication, 2006.

[49] A. McCallum and K. Nigam, "A comparison of event models for naive Bayes text classification," in *Proc. 15th AAAI Workshop on Learning for Text Categorization (AAAI'98)*. Madison, Wisconsin: AAAI Press, Jul. 1998, pp. 41–48.

[50] T. Zhang, "Solving large scale linear prediction problems using stochastic gradient descent algorithms," in *Proceedings of the 21th International Conference on Machine Learning (ICML'04)*. Banff, Alberta, Canada: ACM, Jul. 2004, pp. 116–123.

[51] K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer, "Online passive-aggressive algorithms," *Journal of Machine Learning Research*, vol. 7, pp. 551–585, Mar. 2006.

[52] Y. Freund and R. E. Schapire, "Large margin classification using the perceptron algorithm," *Machine Learning*, vol. 37, no. 3, pp. 277–296, Dec. 1999.

[53] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, no. 85, pp. 2825–2830, Oct. 2011.

[54] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Information Processing and Management*, vol. 24, no. 5, pp. 513–523, Aug. 1988.

[55] R. M. Silva, "Da navalha de occam a um método de categorização de textos simples, eficiente e robusto," Ph.D. dissertation, Faculdade de Engenharia Elétrica e de Computação, Universidade Estadual de Campinas, Campinas, SP, BR, Mar. 2017.

[56] K. Crammer, M. Dredze, and F. Pereira, "Confidence-weighted linear classification for text categorization," *Journal of Machine Learning Research*, vol. 13, no. 60, pp. 1891–1926, Jun. 2012.