Métodos de vizinhos mais próximos em classificação multirrótulo

Pedro Mariano Sousa Bezerra , Andrea Carolina Peres Kulaif

Laboratório de Bioinformática e Computação Bioinspirada (LBiC)
Departamento de Engenharia de Computação e Automação Industrial (DCA)
Faculdade de Engenharia Elétrica e de Computação (FEEC)
Universidade Estadual de Campinas (Unicamp)
Caixa Postal 6101, 13083-970 – Campinas, SP, Brasil

pmariano@dca.fee.unicamp.br,deia.cpk@gmail.com

Abstract – This article presents a comparison of the performance of nearest-neighbors-based approaches to solving multi-label classification problems. A transformation approach for the kNN method is compared with two adapted versions for multi-label problems, the ML-kNN method, and a modified version which seeks to explore the correlations within the labels. The results for the Yeast Data Set showed that the ML-kNN performed better in the task.

Keywords – Multi-label classification, nearest neighbors, problem transformation

1. Introdução

Métodos de classificação multirrótulo tem sido explorados nas mais diversas áreas, como categorização de textos e músicas, bioinformática e diagnósticos médicos. Qualquer cenário em que uma observação possa ser atribuída a mais de uma classe pode se encaixar nessa categoria de problemas.
Muitas abordagens foram propostas na literatura. Algumas buscam utilizar métodos de classificação tradicional com adaptação do problema. Outras foram desenvolvidas de forma a considerar as características do cenário multirrótulo. Independentemente da origem da abordagem, todas tentam lidar com um dos principais desafios desse problema, que se encontra no fato de que cada amostra pode estar associada a um número diferente de classes.
Este trabalho tem como objetivo principal comparar os resultados obtidos por técnicas de classificação multirrótulo baseadas em abordagens de vizinhos mais próximos. Foi utilizado o método *Binary Relevance* de transformação do problema e aplicado o classificador *k-Nearest Neighbors* (kNN) para comparação com o algoritmo *Multi-Label k-Nearest-Neighbors* (ML-kNN), que é uma versão adaptada do algoritmo kNN para o problema multirrótulo. Também foi proposta uma modificação do algoritmo ML-kNN, de forma a considerar a correlação entre os rótulos dos dados de treinamento para a classificação de novas amostras.
Este trabalho está organizado da seguinte maneira: a Seção 2 apresenta o problema de classificação multirrótulo; a Seção 3 expõe os métodos utilizados; os resultados estão contidos na Seção 4; por fim, a Seção 5 apresenta as conclusões acerca dos resultados obtidos no trabalho.

2. Problema de classificação multirrótulo

Problemas tradicionais de classificação representam a tarefa de relacionar um rótulo l para cada amostra de dados, dentre dois ou mais rótulos possíveis pertencentes ao conjunto de classes. No cenário multirrótulo, cada amostra do conjunto de dados estará associada a um subconjunto de rótulos $Y \subseteq \mathcal{Y}$ [1]. São utilizados, então, classificadores que sejam capazes de definir esses subconjuntos de rótulos, que podem variar de tamanho, para novas amostras, com base em um conjunto de dados dos quais os rótulos são conhecidos.
No contexto de classificação multirrótulo, os métodos podem ser classificados entre as seguintes abordagens:

Métodos de transformação do problema: o problema é dividido em múltiplos problemas de classificação binária, aplicando um classificador diferente para cada rótulo, real ou transformado (como no caso da abordagem *Label Powerset*), de forma a definir se a amostra está ou não associada ao mesmo. Este método não explora a correlação entre os rótulos. Entre os tipos de abordagens de transformação do problema, estão:
Binary Relevance (BR), aplica $L = |\mathcal{Y}|$ classificadores binários, um para cada rótulo, para decidir quais estão associados à amostra apresentada. Ao final, o resultado é formado pela união da saída dos classificadores que tiveram uma resposta positiva;
Classifier chains (CC), utiliza a regra de *Bayesian chain*, onde, dado um conjunto com L rótulos, são treinados L classificadores, sendo que o primeiro classificador é treinado somente com os padrões de

entrada, e para os demais são utilizados os padrões de entrada e os classificadores já treinados;
Label Powerset (LP), cada combinação de rótulos possível para o conjunto de rótulos do problema é transformada em uma classe, e aplicam-se métodos de classificação multiclasse. A saída do classificador indica uma classe que representa uma combinação dos rótulos. A relação do número de rótulos L com o número de classes no problema transformado é de 2^L, o que pode representar um problema quando tem-se uma grande quantidade de rótulos.

Métodos de adaptação de algoritmos: esta abordagem refere-se à adaptação de algoritmos de aprendizado de máquina existentes para o cenário multirrótulo, como os algoritmos Adaboost.MH e Adaboost.MR propostos em [2], que são extensões para classificação multirrótulo do algoritmo AdaBoost [3].

3. Metodologia

Neste trabalho, comparamos o desempenho de métodos para classificação multirrótulo adotando as abordagens de transformação do problema e de adaptação de algoritmos. Os métodos adotados baseiam-se nas técnicas de vizinhos mais próximos. O primeiro método adotado é uma abordagem de transformação do problema do tipo *Binary Relevance*, na qual um classificador kNN (*k-nearest neighbors*, ou k vizinhos mais próximos) é construído para determinar a saída binária de cada rótulo de forma independente.
O segundo método adotado consiste no *Multi-Label* kNN (ML-kNN) [4], uma adaptação do kNN para classificação multirrótulo. A ideia por trás deste algoritmo consiste em atribuir um rótulo l a uma amostra t baseada na maximização da probabilidade *a posteriori* $P(H_b^l|E_j^l)$ de a amostra possuir o rótulo l dado que um número j de amostras dentre os k vizinhos mais próximos a t também possuem esse rótulo. Para tal, é necessário calcular as probabilidades *a priori* $P(H_b^l)$ de ocorrência dos rótulos nas amostras de treinamento, onde $b = \{0, 1\}$, H_1^l representa o evento de t possuir o rótulo l e H_0^l de não possuir tal rótulo. Também calculam-se as probabilidades condicionais $P(E_j^l|H_b^l)$, onde E_j^l representa o evento de t possuir j dentre os seus k vizinhos mais próximos com rótulo l. Por fim, escolhe-se o argumento b que maximiza a probabilidade conjunta $P(H_b^l, E_j^l) = P(H_b^l)P(E_j^l|H_b^l)$, que é equivalente à maximização da probabilidade *a posteriori*. O pseudo-código do ML-kNN é apresentado na Fig. 1, onde T é o conjunto de m amostras de treinamento, x_i é a i-ésima amostra de treinamento, s é um parâmetro de suavidade, $\mathcal{Y}$ é o conjunto de rótulos, $\overrightarrow{y}_t(l)$ é a componente de um vetor binário que representa se a amostra t possui o rótulo l, $N(t)$ é o conjunto de k vizinhos mais próximos a t e $\overrightarrow{C}_t(l)$ é o número de vizinhos mais próximos a t que possuem o rótulo l.

$[\vec{y}_t, \vec{r}_t]$=ML-KNN(T, K, t, s)

%Computing the prior probabilities $P(H_b^l)$

(1) **for** $l \in \mathcal{Y}$ **do**

(2) $P(H_1^l) = (s + \sum_{i=1}^{m} \vec{y}_{x_i}(l))\,/(s \times 2 + m)\,; \;\; P(H_0^l) = 1 - P(H_1^l);$

%Computing the posterior probabilities $P(E_j^l|H_b^l)$

(3) **Identify** $N(x_i),\ i \in \{1, 2, \ldots, m\}$**;**

(4) **for** $l \in \mathcal{Y}$ **do**

(5) **for** $j \in \{0, 1, \ldots, \mathrm{K}\}$ **do**

(6) $c[j] = 0; \;\; c'[j] = 0;$

(7) **for** $i \in \{1, 2, \ldots, m\}$ **do**

(8) $\delta = \vec{C}_{x_i}(l) = \sum_{a \in N(x_i)} \vec{y}_a(l);$

(9) **if** $(\vec{y}_{x_i}(l) == 1)$ **then** $c[\delta] = c[\delta] + 1;$

(10) **else** $c'[\delta] = c'[\delta] + 1;$

(11) **for** $j \in \{0, 1, \ldots, \mathrm{K}\}$ **do**

(12) $P(E_j^l|H_1^l) = (s + c[j])/(s \times (\mathrm{K} + 1) + \sum_{p=0}^{\mathbf{k}} c[p]);$

(13) $P(E_j^l|H_0^l) = (s + c'[j])/(s \times (\mathrm{K} + 1) + \sum_{p=0}^{\mathbf{k}} c'[p]);$

%Computing $\vec{y}_t$ and $\vec{r}_t$

(14) **Identify** $N(t)$**;**

(15) **for** $l \in \mathcal{Y}$ **do**

(16) $\vec{C}_t(l) = \sum_{a \in N(t)} \vec{y}_a(l);$

(17) $\vec{y}_t(l) = \arg\max_{b \in \{0,1\}} P(H_b^l)P(E^l_{\vec{C}_t(l)}|H_b^l);$

(18) $\vec{r}_t(l) = P(H_1^l|E^l_{\vec{C}_t(l)}) = (P(H_1^l)P(E^l_{\vec{C}_t(l)}|H_1^l))/P(E^l_{\vec{C}_t(l)})$
$= (P(H_1^l)P(E^l_{\vec{C}_t(l)}|H_1^l))/(\sum_{b \in \{0,1\}} P(H_b^l)P(E^l_{\vec{C}_t(l)}|H_b^l));$

Figura 1. Pseudo-código do ML-kNN

Em ambos os métodos apresentados, a decisão de atribuir um rótulo a uma determinada amostra é realizada de forma independente entre os rótulos. Assim, os métodos não exploram possíveis correlações existentes entre os rótulos que podem contribuir para melhorar o desempenho do classificador. Com este intuito, propomos um terceiro método, baseado no ML-kNN, com uma abordagem gulosa para atribuição dos rótulos. Para uma dada amostra t, inicialmente são atribuídos um subconjunto de rótulos $Y \subseteq \mathcal{Y}$ tal que a probabilidade $P(H_1^l)$ calculada entre os vizinhos de t para todo $l \in Y$ é máxima. Esta probabilidade é utilizada para propósitos

de ranqueamento dos rótulos em métricas de avaliação. Dado o subconjunto de rótulos $Y \subseteq \mathcal{Y}$ que foram atribuídos a t, os rótulos seguintes a serem adicionados à amostra são aqueles que maximizam a probabilidade $P(H_1^l | \overrightarrow{y}_t = Y)$, ou seja, calculada entre os vizinhos de t que possuem os rótulos de Y. Este processo é repetido até que a probabilidade de atribuir rótulos à amostra seja inferior a um valor limiar p definido. Dessa forma, a decisão de atribuir um novo rótulo depende dos que já foram atribuídos, numa tentativa de explorar a correlação entre rótulos.

4. Experimentos Computacionais

Os classificadores apresentados na seção anterior foram submetidos ao conjunto de dados *Yeast* [5], o qual é bem conhecido na literatura de classificação multirrótulo [6]. O conjunto de dados é formado por dados de fungos da espécie *Saccharomyces cerevisiae*, popularmente conhecidos como levedura de cerveja. Cada amostra possui 103 atributos contendo informações sobre expressões gênicas de microarranjos e perfis filogenéticos, e o conjunto possui 2417 amostras de genes. Cada gene está associado a um conjunto de rótulos que identifica as suas classes funcionais. Este conjunto de dados é reconhecidamente difícil de ser tratado, como apontado em [7]. O conjunto de classes funcionais possui uma estrutura hierárquica com 4 níveis de profundidade. Neste trabalho, assim como feito na literatura [4], apenas as classes funcionais pertencentes ao primeiro nível hierárquico são consideradas, resultando em um conjunto com 14 rótulos.

Para todos os métodos, realizamos validação cruzada do tipo k-*fold*, com 10 pastas de validação, e busca unidimensional para determinação do parâmetro k. Como proposto em [4], adotamos $s = 1{,}0$ e variamos k no intervalo $[8, 12]$ para o ML-kNN. Os mesmos valores de k foram avaliados para o kNN com a abordagem *Binary Relevance*. Para a versão adaptada do ML-kNN, adotamos $p = 0{,}5$ e $k \in \{8, 12, 20, 30, 50\}$.

4.1. Métricas de avaliação

Para avaliar o desempenho dos métodos, utilizamos quatro métricas comumente adotadas na literatura de classificação multirrótulo [4]:

Hamming Loss: Essa métrica calcula a média dos erros cometidos, ou seja, quantas vezes uma amostra foi associada a um rótulo equivocadamente ou não foi associada a um rótulo que deveria, normalizada pelo número total de rótulos. O valor ideal é 0, de forma que, quanto menor o seu valor, melhor a classificação.

$$hloss = \frac{1}{N} \sum_{i=1}^{N} \frac{|Y_i \Delta Z_i|}{|L|} \tag{1}$$

onde N representa o número de amostras de teste, Y_i apresenta o conjunto de rótulos associados à amostra i, Z_i indica a predição realizada pelo método de classificação, Δ representa a diferença simétrica entre dois conjuntos e L é o número total de rótulos.

Coverage: Após formar o ranking dos rótulos com maior probabilidade de estarem associados a uma amostra segundo o classificador, essa métrica avalia quantas posições da lista, em média, é necessário percorrer para encontrar todos os rótulos reais da amostra. Quanto menor o valor, melhor.

$$coverage = \frac{1}{N} \sum_{i=1}^{N} \max_{y \in Y_i} rank(x_i, y) - 1 \tag{2}$$

onde x_i é a i-ésima amostra de teste, y é um rótulo e $rank(x_i, y)$ representa o ranking de rótulos em ordem decrescente de probabilidade de estarem associados à amostra x_i segundo o classificador.

Ranking Loss: Avalia a fração média de pares de rótulos que foram ordenados inversamente no ranking para uma amostra. O valor ideal é 0, ou seja, quanto menor o valor, melhor o desempenho do método.

$$rloss = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{|Y_i||\overline{Y_i}|} |(y1, y2) | f(x_i, y_1) \leq f(x_i, y_2), (y_1, y_2) \in Y_i \times \overline{Y_i}| \tag{3}$$

onde $f(x_i, y_j)$ representa a probabilidade indicada pelo classificador de a amostra x_i estar associada ao rótulo y_j, e $\overline{Y_i}$ é o conjunto complementar de Y_i.

Average Precision: Avalia a fração média de rótulos que realmente estão associados à i-ésima amostra e que tiveram colocação superior no ranking a um determinado rótulo $y \in Y_i$. Quanto maior o valor da métrica, melhor o desempenho.

$$avgprec = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{|Y_i|} \sum_{y \in Y_i} \frac{|y' | rank(x_i, y') \leq rank(x_i, y), y' \in Y_i|}{rank(x_i, y)} \tag{4}$$

4.2. Resultados

Os resultados obtidos para o kNN com *Binary Relevance*, para o ML-KNN e para a adaptação proposta para este algoritmo podem ser visualizados nas Tabelas 1, 2 e 3. As tabelas mostram o valor médio e desvio padrão para cada uma das métricas apresentadas na seção anterior com relação às 10 execuções para cada valor de k. Os valores em negrito destacam a configuração para a qual cada método obteve o melhor desempenho em cada métrica avaliada. A adaptação do ML-kNN também foi testada para valores diferentes valores de p, que não mostraram desempenho superior ao valor de $p = 0{,}5$ correspondente aos resultados da Tabela 3.

Analisando qualquer das quatro métricas, é possível verificar um desempenho superior do método ML-kNN em relação às outras abordagens. Podemos observar que, por se tratar de uma solução que foi desenvolvida pensando no problema de classificação multirrótulo, apresenta vantagens em relação à abordagem de transformação do problema, *Binary Relevance*.

Ao mesmo tempo, apesar de também lidar com o problema de classificação multirrótulo, a adaptação do ML-kNN não apresentou melhora em relação ao método original. A adaptação procura explorar a correlação existente entre os possíveis rótulos para definição do subconjunto de rótulos para cada amostra. Uma possível causa para este desempenho ruim é o fato de que uma abordagem gulosa é adotada para atribuição dos rótulos, ou seja, os rótulos são decididos de acordo com a melhor opção em cada instante, e essas decisões não são revistas após a inclusão de novos rótulos; assim, o conjunto de rótulos obtido ao final não necessariamente é o que melhor explora a correlação entre todos os rótulos. Apesar disso, a adaptação proposta do ML-kNN ainda obtém um melhor desempenho do que o kNN com *Binary Relevance* para as métricas de *Coverage, Ranking loss* e *Average precision*.

Tabela 1. Resultados para *k*NN com *Binary Relevance*

k	Hamming loss	Coverage	Ranking loss	Average precision
8	0.1993+/-0.01058	8.18174+/-0.29506	0.22069+/-0.01484	0.73513+/-0.01387
9	0.19901+/-0.01035	8.0911+/-0.2957	0.21481+/-0.01573	0.73782+/-0.01553
10	0.19691+/-0.00912	8.01045+/-0.28508	0.2091+/-0.01497	0.74168+/-0.01592
11	**0.19614+/-0.00993**	7.9513+/-0.27263	0.20495+/-0.01435	0.74423+/-0.0166
12	0.19617+/-0.00947	**7.87807+/-0.24556**	**0.20114+/-0.0136**	**0.7472+/-0.01493**

Tabela 2. Resultados para ML-*k*NN

k	Hamming loss	Coverage	Ranking loss	Average precision
8	0.19617+/-0.00865	7.28552+/-0.24709	0.16845+/-0.01195	0.76308+/-0.01322
9	0.19478+/-0.00822	7.2851+/-0.28057	0.16747+/-0.01202	0.76438+/-0.01351
10	**0.19446+/-0.00848**	7.2959+/-0.26784	0.16781+/-0.01251	0.76469+/-0.01622
11	0.19502+/-0.0077	**7.26197+/-0.26185**	**0.16722+/-0.01244**	**0.76504+/-0.01673**
12	0.19499+/-0.00816	7.27028+/-0.23141	0.16808+/-0.0114	0.76476+/-0.0153

Tabela 3. Resultados para adaptação do ML-*k*NN

k	Hamming loss	Coverage	Ranking loss	Average precision
8	0.20799+/-0.01004	8.18174+/-0.29506	0.22069+/-0.01484	0.73513+/-0.01387
12	0.20524+/-0.00993	7.87807+/-0.24556	0.20114+/-0.0136	0.7472+/-0.01493
20	0.20135+/-0.00808	7.58554+/-0.2293	0.18523+/-0.01077	0.75685+/-0.01512
30	**0.19934+/-0.00876**	7.46179+/-0.21704	0.17719+/-0.00894	**0.75988+/-0.01301**
50	0.20214+/-0.00551	**7.34092+/-0.1956**	**0.17356+/-0.00982**	0.75725+/-0.01301

5. Conclusões

Neste trabalho, apresentamos uma análise comparativa de desempenho de métodos baseados em vizinhos mais próximos para problemas de classificação multirrótulo. As abordagens dos métodos utilizados para resolver este tipo de problema se dividem em: transformação do problema e adaptação de algoritmos. Aqui, avaliamos três abordagens: o método kNN com a transformação do tipo *Binary Relevance* e duas adaptações deste algoritmo para classificação multirrótulo: o método ML-kNN e uma versão modificada, que busca explorar a correlação existente entre os rótulos. Os métodos foram submetidos à base de dados *Yeast*, muito utilizada na literatura de classificação multirrótulo, que contém dados de fungos da espécie *Saccharomyces cerevisiae*, popularmente conhecidos como levedura de cerveja.

De forma geral, para as métricas de avaliação consideradas, o ML-kNN se mostrou superior aos outros dois métodos. Por se tratar de uma adaptação do algoritmo kNN para problemas de classificação multirrótulo, era esperado que o ML-kNN obtivesse um melhor desempenho do que uma abordagem de transformação com o kNN. Os resultados da versão modificada do ML-kNN proposta neste trabalho não superaram os resultados da versão original, possivelmente devido ao fato de que uma abordagem gulosa é adotada para a atribuição dos rótulos.

Os métodos de vizinhos mais próximos possuem como vantagem a sua simplicidade de implementação. No contexto de classificação multirrótulo, estes métodos apresentam resultados competitivos [4]. Como perspectivas futuras, pode-se estudar como técnicas de aprendizado multitarefa [8] poderiam ser incorporadas nestes métodos, de forma a melhor explorar as dependências existentes entre as tarefas

de atribuição dos rótulos.

Referências

[1] Grigorios Tsoumakas and Ioannis Katakis. Multi-label classification: An overview. *Int J Data Warehousing and Mining*, 2007:1–13, 2007.

[2] Robert E. Schapire and Yoram Singer. Boostexter: A boosting-based system for text categorization. *Machine Learning*, 39(2):135–168, 2000.

[3] Y. Freund and Schapire R.E. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

[4] Min-Ling Zhang and Zhi-Hua Zhou. ML-KNN: A lazy learning approach to multi-label learning. *Pattern recognition*, 40(7):2038–2048, 2007.

[5] Mulan: A Java library for multi-label learning. `http://mulan.sourceforge.net/datasets-mlc.html`. Acessado em: 01/07/2019.

[6] Andrew Kachites McCallum. Multi-label text classification with a mixture model trained by em. In *AAAI 99 Workshop on Text Learning*, 1999.

[7] André Elisseeff and Jason Weston. A kernel method for multi-labelled classification. In *Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic*, NIPS'01, pages 681–687, Cambridge, MA, USA, 2001. MIT Press.

[8] Rich Caruana. Multitask learning. *Mach. Learn.*, 28(1):41–75, July 1997.