Tratamento do Volume de Dados Armazenados em Ambiente de Aprendizado Baseado em Instâncias

Luis André Claudiano, Maria do Carmo Nicoletti
Faculdade de Campo Limpo Paulista - FACCAMP
Campo Limpo Paulista - SP

{laclaudi@uol.com.br, carmo@cc.faccamp.br

Abstract. *This paper presents an ongoing research work in the area of instance-based learning (IBL). Algorithms that implement IBL frequently have to deal with the problem of deciding which data instances (and their description) to store for use during the generalization phase. Storing too many instances requires large memory, which can contribute to slowing down the execution speed. So strategies that allow a reduction in the storage requirements, without reducing the performance of programs implementing such algorithms, should be considered. The paper presents the main topics of the research conducted so far; a few strategies proposed in the literature, for reducing the storage volume associated with IBL, will be the main research topic to be pursued.*

Resumo. *Este artigo apresenta um trabalho de pesquisa em andamento na área de aprendizado baseado em instâncias (ABI). Algoritmos que implementam ABI têm que lidar com o problema de decidir quais instâncias de dados (e suas descrições) armazenar para uso durante a fase de generalização. O armazenamento de muitas instâncias requer uma memória de tamanho considerável, fato que pode contribuir para tornar mais lenta a execução do programa que implementa tal algoritmo. Assim sendo, estratégias devem ser consideradas que permitam a redução em exigências de armazenamento, sem que isso provoque redução da performance de tais programas. O artigo descreve os principais tópicos de pesquisa abordados até o momento; algumas poucas estratégias encontradas na literatura, para a redução do volume de armazenamento associado com o ABI, serão abordadas na continuação da pesquisa.*

1. Introdução

Aprendizado de Máquina (AM) é uma das muitas subáreas da área de pesquisa denominada Inteligência Artificial (IA). O chamado *aprendizado indutivo de máquina* (AIM) é o modelo de AM mais popular e mais bem sucedido e tem sido implementado utilizando inúmeras técnicas e algoritmos (ver [Mitchell 1997]). Para viabilizar AIM é mandatório que um conjunto de instâncias, que representam os conceitos a serem aprendidos, esteja disponível. Esse conjunto de instâncias é denominado *conjunto de treinamento*. Conjuntos de treinamento geralmente são descritos por vetores de pares atributo−valor_de_atributo e, dependendo da situação, de uma classe associada (que indica qual conceito a instância em questão representa). A classe de cada instância que participa do conjunto de treinamento é, na maioria dos casos, determinada por um especialista humano da área de conhecimento descrita pelos dados.

Dentre os muitos modelos de aprendizado supervisionado (e.g., simbólico, neural, etc.), encontra-se aquele chamado de *aprendizado baseado em instâncias* (ABI) (também conhecido como aprendizado preguiçoso (*lazy learning*)). Como comentado em [Mitchell

1997] algoritmos que implementam ABI, via de regra, simplesmente armazenam as instâncias de treinamento e adiam o processo de 'generalização' até o momento em que uma nova instância, de classe desconhecida, precisa ser classificada. Uma vantagem desse tipo de aprendizado é que, ao invés de generalizar o conceito, considerando todo o conjunto de treinamento disponível, estima o conceito localmente, para cada nova instância a ser classificada. Uma das desvantagens de algoritmos baseados em instâncias é o custo da classificação, quando o conjunto de treinamento é volumoso e envolve, também, um grande número de atributos, uma vez que todo o processamento requerido acontece na fase de classificação, dado que a fase de aprendizado consiste simplesmente no armazenamento do conjunto de treinamento.

Algoritmos que implementam o aprendizado baseado em instâncias são, via de regra, inspirados no algoritmo *Nearest Neighbour* (NN) [Cover & Hart 1967] ou em alguma de suas variantes [Hart 1968] [Gates 1972]. A *descrição do conceito baseada em instâncias* inclui o conjunto de instâncias armazenadas e, possivelmente, alguma informação relativa ao desempenho de tais instâncias, durante processos de classificação que já aconteceram. A família de algoritmos conhecida como *Instance-based Learning* (IBL) apresentada e avaliada em [Aha *et al.* 1991] [Aha 1992] agrupa um conjunto de algoritmos que implementam o ABI. Os algoritmos da família IBL procuram contornar alguma das limitações associadas aos algoritmos de IBI que, como apontado em [Breiman *et al.* 1984], tem como inconvenientes:

- são classificadores computacionalmente caros uma vez que armazenam todas as instâncias de treinamento;
- são intolerantes a atributos com ruído;
- são intolerantes a atributos irrelevantes;
- são sensíveis à escolha da função de similaridade;
- não existir maneira natural ou simples de se trabalhar com atributos que tenham valores nominais ou, então, atributos que, por alguma razão, não têm valores associados;
- fornecerem pouca informação útil com relação à estrutura dos dados.

Este artigo descreve o trabalho sendo realizado, que tem como contexto os algoritmos de ABI e, como foco, a investigação de esquemas que buscam tratar/contornar um dos principais inconvenientes no uso de tais algoritmos i.e., aquele decorrente da necessidade de armazenamento de todas as instâncias de treinamento. Uma das formas de lidar com o alto volume de instâncias armazenadas é por meio das chamadas técnicas de redução. Para tanto, o trabalho se propõe a investigar empiricamente alguns algoritmos que implementam técnicas de redução, particularmente os seis algoritmos propostos em [Wilson & Martinez 2000]. A investigação buscará comprovar (ou não) a real contribuição de tais estratégias, quando agregadas a um conjunto de algoritmos que implementam ABI.

2. O Nearest Neighbor (NN) & Variantes

No que segue alguns algoritmos de ABI que são objeto da pesquisa sendo conduzida são abordados com relação a várias de suas características.

O algoritmo mais básico de ABI é aquele conhecido como *vizinho mais próximo* (*Nearest Neighbor* – NN). Tal algoritmo assume que todas as instâncias armazenadas correspondem a pontos no espaço N-dimensional R^N e, então, uma função de distância é

usada para determinar qual instância, dentre aquelas armazenadas, é a mais próxima de uma dada nova instância (a ser classificada). Uma vez determinada a instância mais próxima, sua classe é atribuída à nova instância considerada. Via de regra a determinação do(s) vizinho(s) mais próximo(s) é feita por meio do cálculo da distância; a distância euclidiana é a mais popular entre as distâncias utilizadas para esse fim.

Considere que uma instância arbitrária x seja descrita pelo vetor N-dimensional de valores de atributos notado por $<a_1(x),a_2(x),\ldots,a_N(x)>$ ($a_r(x)$ representa o valor do r-ésimo atributo da instância x, $1 \leq r \leq N$). A distância euclidiana entre duas instâncias x_i e x_j é mostrada na Eq. (1).

$$d(x_i, x_j) = \sqrt{\sum_{r=1}^{N} \left(a_r(x_i) - a_r(x_j)\right)^2} \qquad (1)$$

O processo de classificação de uma nova instância pode ser estendido considerando um número maior de vizinhos mais próximos (da instância a ser classificada), o que dá origem à versão do algoritmo conhecida como *k vizinhos mais próximos* (k-NN), cujo algoritmo, para o aprendizado de funções com valores discretos i.e., f: $R^N \rightarrow V$, em que $V=\{v_1,\ldots,v_s\}$ é apresentado na Figura 1 (extraído de [Mitchell 1997]).

Algoritmo de treinamento:

para cada exemplos de treinamento (x,f(x)), inclua o exemplo na lista *exemplos_treinamento*

Algoritmo de classificação:

- seja x_q uma instância a ser classificada,
  - sejam $x_1,\ldots,x_k$ as k instâncias da lista *exemplos_treinamento* mais próximas de x_q
  - retorne

$$\hat{f}(x_q) \leftarrow \operatorname{argmax}_{v \in V} \sum_{i=1}^{k} \delta(v, f(x_i))$$

em que $\delta(a,b)=1$ se a=b, caso contrário, $\delta(a,b)=0$

Figura 1. Algoritmo K-NN para aproximação de uma função com valores discretos, f: $R^N \rightarrow V$.

Considere que *exemplos_treinamento* seja $\{(x_1, v_1), (x_2,v_1), (x_3,v_2), (x_4,v_3), (x_5,v_1)\}$ ou seja, com cinco instâncias x_1, x_2, x_3, x_4, x_5 representando 3 classes distintas i.e., v_1, v_2 e v_3. A função f associa a cada instância sua respectiva classe ou seja, $f(x_1) = v_1$, $f(x_2) = v_1$, $f(x_3) = v_2$, etc. Suponha que x_6 seja uma instância a ser classificada e que x_1, x_2 e x_4 sejam as k=3 instâncias mais próximas de x_6. Seguindo o algoritmo da Figura 1, uma soma deve ser conduzida para cada uma das classes associadas às k=3 instâncias existentes, ou seja, para v_1 e v_3.

Para a classe v_1 soma-se, para cada uma das instâncias próximas de x_6, os valores $\delta(v_1,f(x_1)) + \delta(v_1,f(x_2)) + \delta(v_1,f(x_4)) = \delta(v_1,v_1) + \delta(v_1,v_1) + \delta(v_1, v_3) = 1+1+0=2$.

Para a classe v_3 soma-se, para cada uma das instâncias próximas de x_6, os valores $\delta(v_3,f(x_1)) + \delta(v_3,f(x_2)) + \delta(v_3,f(x_4)) = \delta(v_3,v_1) + \delta(v_3,v_1) + \delta(v_3, v_3) = 0+0+1=1$.

Dentre as duas classes $v \in \{ v_1, v_3\}$, a que teve o maior valor de soma foi a classe v_1 e, portanto, tal classe é atribuída à instância x_6.

O algoritmo descrito na Figura 1 pode ser facilmente adaptado para a aproximação de funções com valores contínuos. Para fazer isso o algoritmo deve calcular o valor médio dos k exemplos de treinamento mais próximos, ao invés de calcular o seu valor mais comum. Mais precisamente, para aproximar uma função contínua f: $R^N \rightarrow R$, a fórmula final do algoritmo descrito na Figura 1 deve ser substituída por Eq. (2).

$$\hat{f}(x_q) \leftarrow \frac{\sum_{i=1}^{k} f(x_i)}{k} \qquad (2)$$

Quando do cálculo da distância entre instâncias, o k-NN utiliza todos os atributos que descrevem a instância. Este fato contrasta, de certa forma, com outros métodos de aprendizado automático como, por exemplo, o aprendizado de regras de decisão (ou árvores de decisão). Algoritmos que constroem árvores de decisão utilizam medidas (via de regra, a entropia) para identificar aqueles atributos que têm maior relevância para caracterizar os conceitos sendo aprendidos e, desta maneira, nem sempre o conjunto inteiro de atributos que descreve o conjunto de instâncias, comparece na expressão induzida do conceito. Essas medidas tendem a descartar atributos que são irrelevantes (para a caracterização do conceito).

Na literatura podem ser encontrados inúmeras propostas de variantes do NN (e k-NN) (ver [Bhatia & Vandana 2010]), que buscam contornar algum(ns) dos problemas apontados na Seção 1. Com esse mesmo objetivo e, também, para melhor caracterizar e tornar algoritmos de ABI mais robustos, Aha e colaboradores conduziram um estudo descrito em [Aha *et al.* 1991, Aha 1992] em que vários algoritmos de ABI foram propostos, sendo o IB1 o primeiro e o mais simples deles; os que o seguem i.e. IB2, IB3, IB4 e IB5, tratam de mecanismos para contornar alguns dos problemas evidenciados em domínios reais, tais como tratamento de ruídos nos dados, tratamento de atributos irrelevantes e a introdução de novos atributos.

Wilson & Martinez em [Wilson & Martinez 2000] em um contexto de ABI, apresentam seis algoritmos, caracterizados como de redução, que se propõem à remoção de instâncias da descrição do conceito, com vistas à redução do volume de armazenamento.

A pesquisa até agora conduzida permitiu delinear o conjunto de algoritmos de ABI que será o foco inicial da pesquisa, i.e., aquele constituído por NN, k-NN, IB1, IB2, IB3, IB4 e IB5, a medida que a pesquisa progrida poderão ser considerados outros algoritmos, tendo por motivação o uso de técnicas de redução. A pesquisa buscará evidenciar empiricamente a efetividade de cada um dos 6 algoritmos de redução propostos em [Wilson & Martinez 2000], quando agregado(s) a cada um dos algoritmos do conjunto escolhido. Avaliações de desempenho dos algoritmos de redução em conjuntos de dados artificiais e reais (extraídos do Repositório da UCI [Lichman 2013]) serão conduzidas após a implementação dos algoritmos em um sistema computacional.

Referências

[Aha *et al.* 1991] Aha, D. W.; Kibler, D.; Albert, M. K. (1991) Instance-based learning algorithms, Machine Learning, v. 6, pp. 37–66.

[Aha 1992] Aha, D. W. (1992) Tolerating noisy, irrelevant and novel attributes in instance-based learning algorithms, International Journal of Man-Machine Studies, 36, 267–287.

[Angiulli 2007] Angiulli, F. (2007) Fast condensed nearest neighbor, in: Proc. of The 22nd ICML, 5-11.

[Angiulli 2007] Angiulli, F. (2007) Condensed nearest neighbor data domain description, Fast condensed nearest neighbor, IEEE Transactions on Pattern Analysis and Machine Intelligence, v. 29, n. 10, 1746–1758.

[Bhatia & Vandana 2010] Bhatia, N.; Vandana, A. (2010) Survey of nearest neighbor techniques, International Journal of Computer Science and Information Security, v. 8, no. 2, 2010, 302-305.

[Breiman *et al.* 1984] Breiman, L.; Friedman, J.; Stone, C. J.; Olshen, R. A. (1984) Classification and regression trees, CRC Press.

[Cover & Hart 1967] Cover, T. M.; Hart, P. E. (1967) Nearest neighbor pattern classification, IEEE Transactions on Information Theory, v. IT-13, no. 1, 21–27.

[Gates 1972] Gates, G.W. (1972) The reduced nearest neighbor rule, IEEE Transactions on Information Theory, v. 18, no. 3, 431–433.

[Hart 1968] Hart, P. E. (1968) The condensed nearest neighbor rule, IEEE Transactions on Information Theory, v. 14, 515–516.

[Lichman 2013] Lichman, M. (2013) UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

[Mitchell 1997] Mitchell, T. M (1997) Machine Learning, USA: McGraw-Hill.

[Wilson & Martinez 2000] Wilson, D. R.; Martinez, T. R. (2000) Reduction techniques for instance-based learning algorithms, Machine Learning, v. 38, 257–286.