Seleção de Dados para LVQ através de Aprendizado Exaustivo

Rodrigo T. Peres
rperes@ele.puc-rio.br

Carlos E. Pedreira
carlosp@centroin.com.br

Departamento de Engenharia Elétrica – PUC-RIO

Resumo

Neste artigo é proposta uma metodologia baseada em aprendizado exaustivo para seleção de dados em LVQ. Treinar um modelo com um subconjunto dos dados apropriadamente selecionado a partir do conjunto de treinamento pode ser uma estratégia interessante, e, possivelmente, produzir uma melhora no resultado de generalização. O objetivo do algoritmo proposto é buscar a eliminação de dados ruidosos causando o mínimo de dano possível ao restante da amostra. Aqui se considera um ruído do tipo 'contaminação da amostra', ou seja, uma inversão do sinal de saída. A idéia central é submeter os dados a uma experimentação exaustiva de modelos e buscar aqueles que sejam relevantes, tendo como base a correlação entre o seu erro e o erro do restante dos dados. Feita a seleção dos dados, a atualização dos protótipos é realizada com um subconjunto dos dados originalmente disponíveis. Experimentos numéricos foram realizados com dados controlados e reais, e os resultados obtidos foram muito interessantes, mostrando claramente a potencialidade do método proposto.

1. Introdução

Em um contexto supervisionado, o processo de classificação de padrões pode ser dividido em quatro etapas (não necessariamente nesta ordem): Escolha e especificação de um modelo; estimação dos parâmetros deste modelo; escolha dos atributos (ou variáveis de entrada) e seleção dos dados utilizados durante o treinamento. Todas essas etapas vêm sendo abordadas na literatura, de forma separada ou conjuntamente, com maior ou menor intensidade [17], [11], [4], [6], [2], [9], [5].

O problema de seleção de dados, assunto central deste artigo, pode ser definido como uma seleção inteligente de um subconjunto da amostra com a finalidade de melhor ajustar os parâmetros (fase de treinamento), visando melhorar a performance de generalização (fora-da-amostra). Este artigo tem como objetivo abordar esse assunto no contexto de Quantização Vetorial por Aprendizado, que na seqüência denotaremos através da amplamente utilizada sigla em inglês, LVQ. Contribuições prévias (não restritas a LVQ) envolvendo seleção de dados podem ser encontradas em [17], [11], [4], [6], [2], [15], [16]. No contexto de LVQ, este tópico foi previamente abordado em [15], [16], [7]. A abordagem dada em [7], estabeleceu uma janela fixa em torno do plano médio entre dois protótipos para o modelo LVQ 2.1. Exemplos de técnicas envolvendo aprendizado ativo podem ser encontradas em [17] e [4].

Anterior a LVQ está o conceito de Quantização Vetorial (denotada pela sigla em inglês, VQ) [14], [13]. A principal idéia de VQ é formar uma aproximação quantizada dos dados, usando um número finito de vetores protótipos. LVQ deve ser visto como uma versão supervisionada de VQ, onde a partir de uma quantização vetorial, utiliza-se o conjunto de dados de forma iterativa para posicionar os protótipos. Em seguida, classifica-se cada um dos dados usando a regra do vizinho mais próximo. Isto é, um vetor de entrada pertencerá ao agrupamento associado ao protótipo que estiver a menor distância deste, elegida uma métrica adequada. Geralmente é interessante associar mais de um protótipo a um mesmo grupo.

A principal contribuição deste artigo é a proposição de um algoritmo para seleção de dados em LVQ através de aprendizado exaustivo. A idéia central consiste em, para cada vetor do conjunto de treinamento, realizar sorteios aleatórios de modelos que serão usados para testar o próprio vetor e o restante dos dados. Se o erro do vetor for pouco correlatado com a taxa geral de erro, ou seja, com o erro do restante do conjunto, significa que este vetor pode causar um superajuste do modelo, prejudicando a generalização. Neste caso, o vetor deve ser eliminado do treinamento.

O artigo está organizado da seguinte forma: Na seção 2 apresenta-se o algoritmo básico de LVQ. Na seção 3, propõe-se um algoritmo para seleção de dados em LVQ. Os experimentos numéricos são mostrados na seção 4. A seção 5 conclui o artigo com as observações finais.

2. O Algoritmo LVQ

O algoritmo de LVQ, proposto por Kohonen [7], tornou-se amplamente conhecido pela sua eficiência e facilidade de implementação e tem sido largamente utilizado em problemas de classificação e reconhecimento de padrões.

A idéia central é selecionar um número fixo de vetores-protótipo que servem de referenciais para cada classe. Esses vetores são atualizados durante a fase de aprendizado com base nos dados. Esta atualização é baseada no algoritmo de gradiente descendente [19], que, na verdade, se funda no clássico método de aproximação estocástica [18], [10].

2.1. LVQ 1

Entre as diversas possíveis variações do esquema LVQ, vamos utilizar neste artigo o esquema de atualização conhecido por LVQ 1 [8]. Seja uma amostra x apresentada ao algoritmo pela n-ésima vez, o algoritmo básico do LVQ 1 é o seguinte:

$m_c(n+1) = m_c(n) + \alpha(n)\left[x(n) - m_c(n)\right]$, se x e m_c pertencem a mesma classe,

$m_c(n+1) = m_c(n) - \alpha(n)\left[x(n) - m_c(n)\right]$, se x e m_c pertencem a classes diferentes,

$m_i(n+1) = m_i(n)$, para i $\neq$ c,

onde $0 < \alpha(n) < 1$.

Dado um conjunto de m protótipos e um vetor de entrada x, a classe de x será:

$$c = \arg\min_i \{ \|x - m_i\| \}$$

3. Seleção de Dados Através de Aprendizado Exaustivo

Nesta seção, será desenvolvida, do ponto de vista metodológico, a principal contribuição deste artigo. Deseja-se um procedimento que seja capaz de eliminar o ruído, permitindo que os protótipos sejam ajustados de acordo com a real estrutura dos dados.

Considera-se ruído, conforme definido a seguir, como 'contaminação da amostra', ou seja, uma inversão do sinal de saída.

Definição 1: Seja D um conjunto de treinamento com n classes diferentes. Os subconjuntos $R_1,...,R_n$ de D, são ditos 'ruídos' das classes 1,...,n respectivamente, se cada $x_j \in R_i$, i = 1,...,n possui a saída de uma classe k, k = 1,...,n, $k \neq i$.

Os ruídos são, portanto, aqueles pontos que possuem a saída de uma classe diferente da que pertencem.

Aprendizado exaustivo significa a utilização de modelos aleatórios. Um algoritmo de seleção de dados baseado em aprendizado exaustivo foi anteriormente proposto em um contexto razoavelmente geral em [12]. O objetivo é utilizar um número grande de modelos aleatórios para determinar se um vetor de entrada deve ou não pertencer ao conjunto de treinamento.

Uma forma de se medir o sucesso da generalização de um modelo [12], é através do erro quadrático entre erro de treinamento e erro de teste, ou seja:

$$E[(erro\_treinamento - erro\_teste)^2]$$

Uma boa generalização está associada a um modelo para o qual o erro durante a fase de treinamento seja uma boa estimativa para o erro na generalização. Se houver vetores cujo erro não seja um bom indicador da taxa geral de erro, há um superajuste do modelo ao conjunto de treinamento em detrimento de uma boa generalização. Pode-se dizer que os erros destes vetores são pouco correlatados com o erro do restante do conjunto. Sob ponto de vista de aprendizado exaustivo, esta abordagem é analisada em [12].

Considere um conjunto de treinamento D com n amostras. Para cada vetor de entrada $x_i \in D$, calcula-se o erro médio de D – $\{x_i\}$ da seguinte forma:

$$\frac{1}{n-1}\sum_{j=1}^{n-1} erro(x_j), \text{ para } j \neq i,$$

ou seja, o erro do conjunto de treinamento com exceção do vetor x_i. Esse erro é chamado de erro-fora-da-amostra. Paralelamente, calcula-se o erro de x_i (erro-na-amostra). Ao medir a correlação entre esses dois erros, mede-se o quanto o erro de um ponto está correlatado com o erro geral [12]. Esta correlação pode ser escrita como:

$$\rho(D) = corr[erro\_na\_amostra, erro\_fora\_da\_amostra]$$

onde $\rho(D)$ representa o cálculo da correlação entre o erro de cada vetor e o erro de todo o conjunto. É claro que o interesse está no resultado de cada vetor específico x_i. Se $\rho(x_i) = 0$, então o erro de x_i não informa nada sobre o erro geral. Se $\rho(x_i) < 0$, então o modelo que classifica x_i corretamente tende a ter um erro de teste maior do que os modelos que cometem um erro na classificação de x_i. Na verdade, ρ reflete a dificuldade de classificação dos vetores de entrada do conjunto de treinamento.

3.1. Uma Proposta para Seleção de Dados em LVQ

Nesta seção, se formaliza a metodologia proposta para LVQ. De uma forma geral, o algoritmo se divide em três etapas:

1) Quantização Vetorial: realiza-se uma quantização vetorial das classes associando um número de protótipos a cada uma dessas. Um vetor de uma classe é associado a um dos protótipos dessa classe através da regra do vizinho mais próximo, ou seja, este vetor pertence à região do protótipo mais próximo a ele. Assim, se constrói uma partição dentro de cada classe;

2) Aprendizado Exaustivo: Para cada dado, sorteiam-se protótipos aleatórios, e, a cada sorteio, calcula-se o erro-na-amostra e o erro-fora-da-amostra como mostrado no início da seção 3. A correlação entre estes erros é então calculada;

3) Selecionam-se os vetores que possuam correlação positiva entre seu erro e o erro do restante do conjunto. Estes serão os vetores selecionados para participarem do treinamento.

A primeira etapa requer o estabelecimento da quantidade de protótipos atribuídos a cada uma das classes. Existem algumas heurísticas direcionadas a este problema na literatura, mas não há uma boa solução consensual. Uma possibilidade é explorar uma possível relação entre o numero de protótipos por grupo e as variâncias das densidades de probabilidade das classes, como sugerido em [10].

Aqui, usa-se como número total de protótipos $\sqrt{n}$, onde n é o número de amostras disponíveis para treinamento. Desta forma, quando $n \to \infty$, $\sqrt{n} \to \infty$ e $\lim \frac{\sqrt{n}}{n} \to 0$, ou seja, assintoticamente, o número de protótipos aumenta, porém, numa razão muito menor se comparado ao número de dados. Os protótipos convergem, então, para as médias das regiões [10]. Em seguida, utilizando a variância de cada classe, busca-se uma divisão deste número total de protótipos pelas classes. Define-se variância total de uma classe como a soma das variâncias em todas as dimensões, e.g., em um problema em $\Re^2$, a variância total será $V_x + V_y$. Em um problema com k classes, tem-se o vetor V_T das variâncias totais, $V_T = [V_{T_1} V_{T_2} ... V_{T_k}]$, onde V_{Ti} é a variância total da i-ésima classe. Seja $V_{T\max}$ = max(V_T) (a maior variância total entre todas as classes). Normaliza-se V_T, dividindo cada V_{Ti} por $V_{T\max}$. O número de protótipos por classe é dado, então, proporcionalmente as variâncias normalizadas.

Estabelecido o número de protótipos por classe, torna-se necessário determinar a localização dos mesmos. Para isto, foi utilizado o LBG (Linde-Buzo-Gray) [13], algoritmo clássico de quantização vetorial que localiza as posições dos protótipos com o critério de minimizar a distorção média.

Para uma classe hipotética A com j protótipos, existe uma partição $A_1, A_2, ..., A_j$ de A tal que, o algoritmo LBG associa cada A_i a um protótipo m_i. Além disso, para cada $a_i \in A_i$, $d(a_i, m_i) < d(a_i, m_k), k \neq i$, onde 'd' é uma métrica (em geral Euclidiana).

Note que os protótipos associados às regiões são 'representantes' destas regiões, atribuídos pelo algoritmo LBG.

Em seguida, procede-se a fase de aprendizado exaustivo. No caso de LVQ, pode-se associar o sorteio de modelos a um sorteio aleatório de protótipos. Este sorteio é feito entre os vetores de D. É importante lembrar que as classes foram particionadas durante a etapa 1. Assim, para uma classe A particionada em regiões disjuntas $A_1, A_2, ..., A_j$, a cada rodada, deve haver um sorteio de protótipo por região.

Considere o vetor $x_i \in D$. Sorteia-se um número suficientemente grande de protótipos para as regiões de cada partição. A cada rodada de sorteio, aplica-se a regra do vizinho mais próximo para efeito de classificação. Verifica-se, então, o erro médio para D – { x_i }, o restante do conjunto de treinamento, denotado por erro-fora-da-amostra (erro_FA). Paralelamente, o erro-na-amostra (erro_NA) também é calculado para x_i. Se a classe do protótipo mais próximo coincidir com a classe de x_i, atribui-se ao erro valor zero, caso contrário, valor 1. O erro-fora-da-amostra também é calculado da mesma forma. A correlação entre os dois erros é então determinada após um número grande de sorteios. Este procedimento é repetido para todos os dados, e, ao final desta etapa, cada x_i possui um valor de correlação associado. O vetor de correlação é denominado ρ.

$$\rho(D) = \frac{E[(erro\_NA)(erro\_FA)] - E[erro\_NA]E[erro\_FA]}{\sqrt{Var[erro\_NA]}\sqrt{Var[erro\_FA]}}$$

Após o cálculo de ρ, é necessário que se estabeleça um critério de corte nesse vetor para selecionar os dados que farão parte do treinamento. Assim, o valor de correlação especificado para este corte foi zero, ou seja, eliminam-se todos os vetores de entrada cujo erro possua correlação negativa com o erro geral. Este critério foi adotado porque a correlação negativa entre o erro de um vetor de entrada e o erro do restante do conjunto indica que este vetor pode estar causando um superajuste do

modelo ao conjunto de treinamento, prejudicando a generalização.

3.2. Formalização do Algoritmo

Seja D = $\{x_i, y_i\}_{i=1}^{N}$ o conjunto de treinamento composto por N pares {dado, rótulo} e z a quantidade de sorteios de protótipos a serem realizados.

Passo 1: Aplicar heurística apresentada na seção 3.1 para determinar a quantidade total e a quantidade de protótipos por classe.

Passo 2: Através do algoritmo LBG (descrito na seção 3.1), determinar a localização de cada protótipo, dividindo cada classe em uma partição.

Passo 3: Para cada $\{x_i, y_i\}_{i=1}^{N}$, fazer:

Enquanto número do sorteio $\leq$ z

Sortear um protótipo por região de cada partição;

Calcular erro x_i e erro médio do restante do conjunto, atribuindo erro zero se a classe coincidir com a classe do protótipo mais próximo e erro um caso contrário.

Calcular $\rho(x_i)$.

Passo 4: Estabelecer o corte do vetor de correlação ρ no ponto zero. Um novo conjunto de treinamento D^* é estabelecido, onde $D^* = \{ x_i \in D \mid \rho(x_i) \geq 0\}$

4. Experimentos Numéricos

Nesta seção, são apresentados os resultados numéricos. O primeiro experimento foi feito com um banco de dados gerado para esta finalidade, e o segundo com um banco de dados pertencente ao repositório da UCI[♣].

Os protótipos são atualizados através do LVQ 1. A taxa de aprendizado foi fixada em 0.01 e os dados foram apresentados uma única época para todos os experimentos realizados.

Para o cálculo do vetor de correlação ρ foram considerados 100 sorteios aleatórios de protótipos, pois esse número pareceu uma boa solução de compromisso entre a qualidade dos resultados e as possibilidades computacionais.

4.1. Um Exemplo Sintético

Considera-se um problema de duas classes, com 250 amostras de cada classe, separadas pela função cosseno. Em cada classe, 5% de ruído (13 amostras) foi gerado (invertendo-se o desfecho). Na figura 1 as classes estão, respectivamente, em verde e amarelo e os ruídos em vermelho.

Na tabela 1, está a quantidade de ruído reconhecido pelo método, após o corte do vetor ρ em zero (Passo 4, apresentado na subseção 3.2). Na tabela 2, apresenta-se a taxa de acerto dentro-da-amostra (500 dados) e fora-da-amostra (1856 dados) sem seleção de dados e após o corte em zero.

Embora a taxa de acerto fora-da-amostra tenha tido uma melhora apenas razoável após a seleção de dados, na tabela 1 vê-se claramente a potencialidade do método para reconhecimento de ruído.

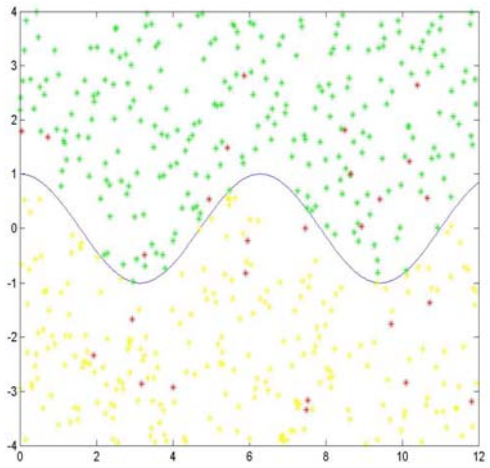

Figura 1. Dados simulados com ruído – Duas classes (verde e amarelo) e ruído (vermelho).

	'Ruído' Detectado
Classe 1	10 (77%)
Classe 2	12 (92%)

Tabela 1. Quantidade de 'ruído' detectado pelo algoritmo.

	Sem seleção	Corte em Zero
Dentro-da-amostra	92.2%	98.2%
Fora-da-amostra	95%	95.4%

Tabela 2. Taxa de acerto de classificação - (relativo a 4.1).

♣ http://www.ics.uci.edu/~mlearn/MLRepository.html

4.2. O banco Íris

O banco de dados Íris[♣] é comumente usado como benchmark na literatura de classificação de padrões. Criado por Fisher [3], o banco contém três classes (Íris Setosa, Íris Versicolour e Íris Virginica) com cinqüenta instâncias cada, e quatro atributos de entrada. Uma particularidade interessante deste banco é a pouca quantidade de dados disponíveis para treinamento, são apenas 150 instâncias. Desta forma, se ilustra que este procedimento de seleção de dados pode ser interessante não apenas para grande quantidade de dados.

O resultado está apresentado na tabela 3. A etapa 1 ilustra o processo sem seleção de dados (150 dados) com o erro fora-da-amostra estimado usando a técnica de leave-one-out [1]; a etapa 2 com o corte em zero (147 dados) e erro fora-da-amostra usando leave-one-out; finalmente, na etapa 3, foi realizado leave-one-out para os 147 dados obtidos com o corte em zero e em seguida testou-se nos outros três dados.

	Etapa 1	Etapa 2	Etapa 3
Taxa de Acerto	96.67 % (5 erros)	97.96 % (3 erros)	96.67 % (5 erros)

Tabela 3. Taxa de acerto de classificação - (relativo a 4.2).

5. Conclusão

Neste artigo foi proposto um algoritmo para seleção de dados em LVQ através de aprendizado exaustivo. A idéia central é sortear protótipos aleatórios entre vetores do próprio conjunto de treinamento, com a finalidade de determinar a correlação entre o erro de um vetor e o erro do restante do conjunto. Esta correlação mostrou ser um bom indicador na decisão de retirar um determinado vetor do conjunto de treinamento. A quantização vetorial realizada antes do aprendizado exaustivo foi fundamental para o sucesso do algoritmo, pois sortear um vetor para cada região dentro da partição de cada classe como protótipo, garante que os vetores desta partição estarão 'protegidos' no sentido que sempre existirá um protótipo da mesma classe próximo a ele. Com isso, o número de erros para estes vetores ao longo do aprendizado exaustivo acaba sendo pequeno, garantindo sua participação no treinamento. Em contrapartida, um ruído do tipo 'contaminação da amostra', que consiste na inversão do sinal de saída de um vetor, estará exposto a mais erros, já que este se encontra em uma partição de outra classe, diminuindo a correlação entre seu erro e o erro geral e levando a eliminação deste vetor do treinamento.

A metodologia proposta é computacionalmente exaustiva, mas sua aplicação já se mostra perfeitamente viável com os computadores atuais. Os resultados apresentados comprovam a capacidade do algoritmo em reconhecer ruídos do tipo 'contaminação na amostra'. Está em curso investigação com o objetivo de produzir melhoras significativas nos resultados de classificação (a melhora obtida até o momento é apenas discreta). Neste sentido, acredita-se que etapas preliminares, como a fase de quantização vetorial, serão de crucial importância.

Referências

[1] Duda, Hart, Stork (2001) In John Wiley & Sons, Inc. (Ed.) "Pattern Classification", Second Edition, Wiley-Interscience.

[2] Faraway, J.J. (1990). "Sequential design for the nonparametric regression of curves and surfaces", Proceedings of the 22nd Symposium on the Interface between Computing Science and Statistics, Springer, 104-110.

[3] Fisher, R.A. (1950). "The use of multiple measurements in taxonomic problems", Annual Eugenics, 7, Part II, 179-188 7, 1936.

[4] Hasenjäger, M., Ritter, H. e Obermayer, K. (1999). "Active Learning in Self-Organizing Maps", In Oja E. & Kaski S. editors, Kohonen Maps, pp 57-70, Elservier.

[5] Haykin, S. (1999). "Neural Networks, a Comprehensive Foundation", 2rd ed., Prentice Hall, US.

[6] Hwang, J. N., Choi, J. J., Oh S. e Marks II R. J. (1991) "Query-based learning applied to partially trained multi-layer perceptrons", IEEE Trans. Neural Networks (Jan), Vol.2, nº1, pp.131-136.

[7] Kohonen, T. (2001). "Self-Organizing Maps", Springer-Verlag.

[8] Kohonen, T., Hynninen, J., Kangas, J., Laaksonen, J., Torkkola, K. (1996). "LVQ PAK- The Learning Vector Quantization Program Package", Report A30 (Jan), Helsinki University of Technology, Laboratory of Computer and Information Science.

[9] Kwak, N., Chong-Ho Choi, (2002). "Input Feature Selection by Mutual Information Based on Parzen Window", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol 24, issue 14, pp.1667-1671.

[♣] http://www.ics.uci.edu/~mlearn/MLSummary.html

[10] La Vigna, A. (1989). “Nonparametric Classification using LVQ”, Ph.D. Dissertation, University of Maryland – 90-1, 1989.

[11] MacKay D. J. C. (1992). “Information-based objective functions for active data selection”, Neural Computation (Jul), Vol. 4, issue 4, pp. 590-604.

[12] Nicholson, A. (2002). “Generalization Error Estimates and Training Data Valuation”, Ph.D. Dissertation, California Institute of Technology, US.

[13] Y. Linde, A. Buzo, and R. M. Gray, (1980) “An algorithm for vector quantizer design”, *IEEE Trans. Commun.*, vol. COM-28, pp. 84–95.

[14] R. M. Gray, (1984) “Vector quantization”, *IEEE ASSP Mag.*, vol. 1, pp. 4–29.

[15] Pedreira, C.E. (no prelo). “Learning Vector Quantization with Training Data Selection”, IEEE Transactions on Pattern Analysis and Machine Intelligence.

[16] Pedreira, C.E. (2004). **“**Uma Metodologia de Quantização Vetorial Com Escolha Seletiva De Dados”, XV Congresso Brasileiro de Automática (CBA2004), Gramado, RS.

[17] Plutowski, M. e White, H. (1993). “Selecting Concise Training Sets from Clean Data”, IEEE Transactions on Neural Networks, Vol. 4, nº2, pp 305-318.

[18] Robins, H. & Monro, S. (1951) “A Stochastic Aproximation Method”, Annals of Mathematical Statistics, 22, 400-407.

[19] Luenberger, D.G. (1984), “Linear and Nonlinear Programming”, Second Edition, AddisonWesley.