Estratégias Evolucionárias para a Mineração de Dados em Séries Temporais Univariadas

Leila Maria Vriesmann[1,2], Aurora Trinidad Ramírez Pozo[2], Alaine Margarete Guimarães[1]

[1]Departamento de Informática – Universidade Estadual de Ponta Grossa (UEPG)
84.030-900 – Ponta Grossa – PR – Brasil

[2]Departmento de Informática – Universidade Federal do Paraná (UFPR)
Caixa Postal 19.081 – 81.531-980 – Curitiba – PR – Brasil

leilavriesmann@yahoo.com.br, aurora@inf.ufpr.br, alainemg@uepg.br

Abstract. *This paper describes the use of Evolution Strategies, an Artificial Intelligence technique, in Temporal Data Mining problems more specifically for time series forecasting. Temporal data have the factor time as one of their characteristics. In this work, the Evolution Strategies have been used to map functions that describe the data. Furthermore, this process has been improved through a boosting algorithm. The techniques were evaluated and their results were compared with the results obtained by the ARMA model. Boosting and Evolution Strategies overcame the results of the ARMA model in almost all analyzed datasets. As future work comparative studies with other methodologies for events prediction will be studied.*

Resumo. *Este artigo descreve o uso de Estratégias Evolucionárias, uma técnica de Inteligência Artificial, em problemas de Mineração de Dados Temporais, mais especificamente em séries temporais. Dados temporais são aqueles que possuem o fator tempo como uma de suas características. Neste trabalho, as Estratégias Evolucionárias são utilizadas para mapear funções que descrevem os dados, as quais são aperfeiçoadas por meio de um algoritmo de boosting. Os resultados são comparados com os resultados obtidos pelo modelo ARMA. Boosting e Estratégias Evolucionárias superaram os resultados do modelo ARMA em quase todas as bases de dados analisadas. Como trabalho futuro sugerem-se estudos comparativos com outras metodologias para a predição de eventos.*

1. Introdução

A extração de padrões (ou conhecimento) de um conjunto de dados é tarefa da Mineração de Dados, a qual é a etapa principal do processo de descoberta de conhecimento em base de dados [Fayyad, Piatetsky-Shapiro e Smyth 1996]. Quando o banco de dados possui o fator tempo como uma de suas características consideradas no processo de descoberta de conhecimento, utiliza-se o termo Mineração de Dados Temporais.

Os valores de um fenômeno ao longo do tempo são chamados, segundo Kout, Vlcek e Klema (2005), séries temporais (*time series*). Séries temporais que consistem de

observações individuais armazenadas seqüencialmente com incrementos iguais de tempo são chamadas "séries temporais univariadas (*univariate time series*)" [NIST/SEMATECH e-Handbook of Statistical Methods 2006]. Séries temporais univariadas podem ser utilizadas na Mineração de Dados Temporais.

A predição de eventos é um dos objetivos mais procurados com a Mineração de Dados Temporais. Sua função é prever fatos do futuro com base em informações do passado.

A Mineração de Dados Temporais é importante para a previsão e análise do comportamento de fatos baseando-se em suas alterações ao longo do tempo. Na literatura podem-se encontrar diversos trabalhos onde o atributo que se deseja prever é discreto [Liu, Mckay e Abbass 2003] [Weiss e Hirsh 1998]. No entanto, muitos problemas do domínio real são constituídos por atributos contínuos.

O mapeamento de funções que descrevem o comportamento de um determinado atributo em uma base de dados pode ser realizado por técnicas capazes de manipular valores contínuos para, por exemplo, especificar pesos em cada atributo previsor. Nesse contexto, não existem muitas opções além das técnicas estatísticas. Almejando-se trabalhar com técnicas de Computação Evolucionárias, têm-se as Estratégias Evolucionárias (ESs – *Evolution Strategies*), que são apropriadas para a manipulação desse tipo de dado. Estratégias Evolucionárias [Bäck, Rudolph e Schewefel 1993] codificam indivíduos (soluções potenciais) na forma de variáveis reais, os quais são evoluídos ao longo das gerações. Por meio desse processo, pode-se facilmente obter uma função linear que descreve os dados.

O objetivo desse trabalho é utilizar Estratégias Evolucionárias no mapeamento de funções para a Mineração de Dados em séries temporais univariadas. No entanto, em domínios mais complexos, obter apenas uma função linear não é suficiente. As funções geradas pela Estratégia Evolucionária são, dessa maneira, aperfeiçoadas por meio de um algoritmo de Boosting, baseado no AdaBoost.R2, o qual força o algoritmo de ES a focar nos exemplos mais difíceis do conjunto de treinamento. Os resultados são comparados com o modelo ARMA.

Na Seção 2 pode-se encontrar definições de Mineração de Dados Temporais e de Séries Temporais Univariadas. A Seção 3 trata de Estratégias Evolucionárias e de Boosting.

Na Seção 4 são mostrados e comentados alguns experimentos. Finalmente, a Seção 5 traz as conclusões sobre o trabalho realizado.

2. Mineração de Dados Temporais e Séries Temporais Univariadas

A Mineração de Dados (Data Mining), segundo Fayyad, Piatetsky-Shapiro e Smyth (1996), visa extrair padrões (estados) de um conjunto de dados. Sua extensão, a Mineração de Dados Temporais, tem a capacidade de minerar atividades ao invés de somente estados e, assim, inferir relacionamentos de proximidade temporal e contextual, indicando também uma associação causa-efeito [Roddick e Spiliopoulou 2002].

Os valores de um fenômeno ao longo do tempo são chamados, segundo Kout, Vlcek e Klema (2005), séries temporais (*time series*). Séries temporais que consistem de

observações individuais armazenadas seqüencialmente com incrementos iguais de tempo são chamadas "séries temporais univariadas (*univariate time series*)" [NIST/SEMATECH e-Handbook of Statistical Methods 2006].

Uma série temporal univariada normalmente é apresentada em uma única coluna de números. O tempo, portanto, é uma variável implícita neste tipo de série temporal, uma vez que os dados são coletados em intervalos regulares de tempo e não existe a necessidade de fornecê-los explicitamente [NIST/SEMATECH e-Handbook of Statistical Methods 2006].

Dados temporais numéricos podem ser mapeados por meio de uma função linear que descreve o comportamento da variável ao longo do tempo. O mapeamento pode ser realizado por de Estratégias Evolucionárias, uma técnica de Inteligência Artificial, a qual é o assunto da próxima seção.

3. Estratégias Evolucionárias e Boosting

Estratégia Evolucionária (ES – *Evolution Strategy*) é um algoritmo da área de Inteligência Artificial onde indivíduos (soluções potenciais) são codificados por um conjunto de variáveis de valores reais, o "genoma" [Kusiak 2000].

De acordo com Rechenberg (1965), o primeiro algoritmo de ES foi apresentado em 1964 na Universidade Técnica de Berlim (TUB – *Technical University of Berlin*). A idéia era imitar os princípios da evolução orgânica em otimização de parâmetros experimental para aplicações tais como *pipe bending* ou controle de PID para um sistema não linear [Dianati, Song e Treiber 2002].

Uma ES é caracterizada pelo tamanho da população, o número de descendentes produzidos em cada geração e se a nova população é selecionada de pais e descendentes ou somente de descendentes [Kusiak 2000]. Assim, o $(\mu + \lambda)$-ES especifica que μ pais produzem λ descendentes ($\lambda > \mu$). Os descendentes competem com seus pais na seleção dos μ melhores indivíduos para a criação da próxima geração.

O primeiro passo de ES consiste na criação dos μ pais com m posições. Cada posição recebe um valor aleatório, de acordo com o domínio do problema. É realizada a recombinação de pais para gerar os filhos, os quais depois passam por um processo de mutação. Posteriormente, de acordo com o tipo de ES, os melhores entre os pais e os filhos (tipo "+") são selecionados para a criação da próxima geração. O processo se repete até que a condição de parada seja atingida. Maiores informações sobre Estratégias Evolucionárias podem ser encontradas em Dianati, Song e Treiber (2002).

Para obter uma função linear que descreve os dados, especificou-se que cada posição a_1, a_2, ..., a_m de cada vetor pai corresponde ao peso de um atributo previsor (Equação 1). O valor de m corresponde ao número de atributos previsores da base de dados. A condição de parada do algoritmo, nesse caso, é o número máximo de gerações.

$$f(x) = a_1.x_1 + a_2.x_2 + ... + a_m.x_m \quad (1)$$

Objetivando aperfeiçoar as hipóteses geradas pela ES, um algoritmo de Boosting pode ser utilizado. Boosting [Schapire 2002], um método de aprendizado de máquina, é

baseado na observação de que encontrar muitas hipóteses fracas com um classificador pode ser mais fácil do que encontrar uma única hipótese de predição com alta precisão. Para aplicar a abordagem de Boosting, começa-se com um método ou algoritmo para encontrar as hipóteses fracas. O algoritmo de Boosting chama esse algoritmo de aprendizado repetidamente, cada vez alimentando-o com um diferente subconjunto dos exemplos de treinamento (ou, mais precisamente, uma diferente distribuição ou atribuição de pesos sobre os exemplos de treinamento). Cada vez que é chamado, o algoritmo de aprendizado gera uma nova hipótese de predição fraca, e depois de muitas execuções, no caso da tarefa de classificação, o algoritmo de Boosting deve combinar essas hipóteses fracas dentro de uma única hipótese de predição que, de acordo com o esperado, será mais precisa que qualquer outra gerada anteriormente [Schapire 2002]. Nesse caso, o algoritmo para encontrar as hipóteses (funções) fracas é a ES.

Um algoritmo de Boosting que pode ser utilizado com a tarefa de encontrar funções lineares (com atributos contínuos) é o AdaBoost.R2 [Solomatine e Shrestha 2004], o qual utiliza, no seu desenvolvimento, uma função de perda L_t, ou obtida por meio de uma função linear (Equação 2), ou uma função quadrática (Equação 3) ou uma função exponencial (Equação 4). Em cada função, para cada exemplo, é calculado o erro $l_t(i)$ entre o valor predito $f_t(x_i)$ pela hipótese gerada pelo algoritmo de aprendizado fraco na iteração t e entre o valor real y_i do atributo meta, em módulo.

$$L_t(i) = |f_t(x_i) - y_i| / (\max (|f_t(x_i) - y_i|)) \quad (2)$$

$$L_t(i) = (|f_t(x_i) - y_i|/ \max (|f_t(x_i) - y_i|)) \ (|f_t(x_i) - y_i|/ \max (|f_t(x_i) - y_i|)) \quad (3)$$

$$L_t(i) = 1\text{-}\exp(-|f_t(x_i) - y_i| / (\max (|f_t(x_i) - y_i|)) \quad (4)$$

Dessa maneira, pode-se especificar pesos para cada exemplo de treinamento de uma base de dados temporal. Esses pesos, inicialmente, são iguais e, à medida que as iterações do Boosting são realizadas, os exemplos com maiores erros ficam com peso maior (de acordo com a função de perda), obrigando o algoritmo de aprendizado fraco a concentrar-se nos exemplos mais difíceis. Para que isso seja possível, a função de *fitness* (Equação 5) da Estratégia Evolucionária (algoritmo de aprendizado fraco), leva em consideração os pesos $D_t(i)$ de cada exemplo i em cada iteração t do Boosting.

$$fit = \sum_{i=1}^{m} (| f(x_i) - y_i | \ . \ D_t(i)) \ . \ m \quad (5)$$

Uma adaptação foi realizada no algoritmo AdaBoost.R2, de modo que foi construído o algoritmo ESboost (Figura 1), o qual combina Estratégias Evolucionárias e Boosting. A execução do ESboost finaliza quando se atinge o número máximo de iterações T. Na próxima seção são descritos alguns experimentos com o algoritmo.

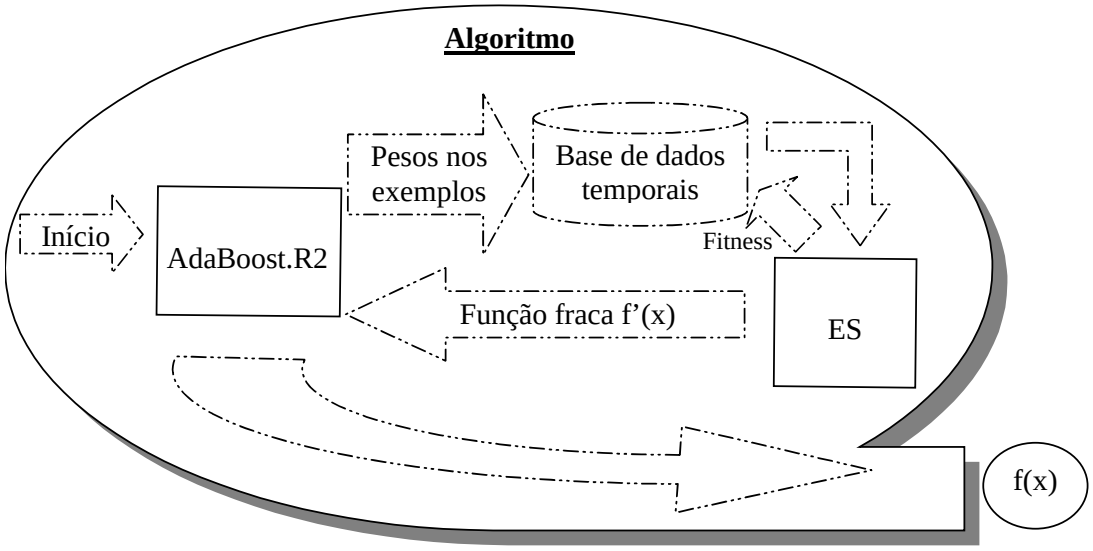

Figura 1. Algoritmo ESboost.

4. Experimentos

O algoritmo ESboost foi executado utilizando o (10+50)-ES, onde 10 indivíduos, denominados pais, geram 50 filhos, os quais competem com seus pais na criação da próxima geração. Demais parâmetros do ESboost são apresentados na Tabela 1. Maiores informações sobre os parâmetros de Estratégias Evolucionárias podem ser encontradas em Dianati, Song e Treiber (2002).

Selecionaram-se algumas séries temporais univariadas (Tabela 2), coletadas em Morretin e Toloi (2004), as quais foram divididas em 90% para treinamento e 10% para teste, de acordo com Souza, Costa e Pozo (2005). Maiores detalhes sobre as bases de dados podem ser obtidos em Morretin e Toloi (2004). Considerou-se até os 4 últimos valores das séries, ou seja, para prever um atributo no tempo, considerava-se seus 4 últimos valores apresentados. Essa metodologia e as bases de dados foram especificadas para que fosse possível a comparação com os valores apresentados por Souza, Costa e Pozo (2005) no modelo ARMA.

Tabela 1 – Parâmetros para o (10+50)-ES padrão e para o ESboost com T =100

PARÂMETROS	ESBOOST COM T = 100
Execuções do Boosting	100
Tamanho da população	10
Número de descendentes	50
Número máximo de gerações	500
Probabilidade de cruzamento	0.6
Tipo de recombinação do ES	Global
Recombinação das variáveis	Discreta
Probabilidade de mutação	1

Os resultados foram fornecidos de acordo com a seguinte medida de erro (quanto menor o erro, melhor o resultado), a qual foi utilizada por Souza, Costa e Pozo (2005) no modelo ARMA:

$$RMS = \frac{\sqrt{\sum_{i=1}^{n} \left(f(x_i) - y_i \right)^2}}{n} \tag{6}$$

onde

n é o número de exemplos de teste

$f(x_i)$ é o valor esperado para y_i

e y_i é o valor real do atributo meta

Tabela 2 – Séries temporais univariadas

BASES DE DADOS	CONJUNTO TOTAL (100%)	TREINAMENTO (90%)	TESTE (10%)
Atmosfera	365	329	36
Bebida	187	169	18
Consumo	154	139	15
Fortaleza	149	135	14
Icv	114	102	12
Ipi	187	169	18
Lavras	384	346	38
Manchas	176	159	17

Os resultados do (10+50)-ES (Estratégias Evolucionárias sem uso de Boosting) e do ESboost obtidos nas séries temporais univariadas da Tabela 2 foram comparados com o modelo ARMA [Box, Jenkins 1976], que é a combinação de modelos auto-regressivos (AR) e de médias móveis (MA). O modelo ARMA é comumente utilizado na tarefa de regressão, e serve como base para comparar com outros modelos.

Na Tabela 3 têm-se os melhores valores obtidos, e na Tabela 4 têm-se as médias e desvios padrão. Os valores das tabelas variam de acordo com o domínio do atributo que se deseja prever em cada base de dados.

As comparações dos resultados obtidos por meio de ESboost, de ES sem Boosting, e do modelo ARMA foram realizadas com base na diferença absoluta (da) por meio da Equação 5.

$$da\ (As - Ap) = \frac{média\ (As) - média\ (Ap)}{\sqrt{\frac{dp\ (As)^2 + dp\ (Ap)^2}{2}}} \tag{7}$$

onde

As é o algoritmo padrão

Ap é o algoritmo proposto

Média(As) é a média obtida para o algoritmo padrão

Média(Ap) é a média obtida para o algoritmo proposto

dp(As) é o desvio padrão obtido para o algoritmo padrão

dp(Ap) é o desvio padrão obtido para o algoritmo proposto

Tabela 3 – Melhor RMS de 10 execuções em séries temporais univariadas

BASE DE DADOS	ARMA		ESBOOST COM T =100		
		(10+50)-ES	LINEAR	QUADRÁTICA	EXPONENCIAL
Atmosfera	ARMA (4,3) = 1,06	0,9901	0,9864	**0,9486**	0,9624
Bebida	ARMA (4,3) = 3,44	3,9576	3,6323	**3,1841**	3,3689
Consumo	ARMA (4,3) = 2,99	3,4665	**2,8759**	3,7501	3,6548
Fortaleza	ARMA (1,0) = 176,12	178,1100	174,2040	163,6180	**157,6720**
Icv	ARMA (1,0) = 36,50	6,5137	6,4734	**5,8958**	6,9867
Ipi	ARMA (3,2) = 4,04	2,6542	**2,5152**	2,5279	2,5310
Lavras	ARMA (3,2) =**12,03**	15,0261	13,8986	22,0506	14,4811
Manchas	ARMA (2,1) = 6,27	**3,4074**	3,6379	8,5569	3,8482

Em relação aos melhores valores obtidos, somente na base de dados *Lavras* o modelo ARMA supera todos os algoritmos executados. Nas bases *Atmosfera*, *Icv* e *Ipi* todos os algoritmos de ESboost e o (10+50)-ES se comportaram melhor que o modelo ARMA.

Comparando os resultados obtidos por ES sem Boosting, denominado (10+50)-ES na Tabela 4, com o ESboost , observou-se que o último apresentou melhor desempenho na maioria das bases de dados temporais em pelo menos uma das funções de perda. Uma exceção a essa regra foi com a base de dados *Manchas*.

Tabela 4 – Média e desvio padrão do RMS de 10 execuções em séries temporais univariadas

BASE DE DADOS			ESBOOST COM T =100					
	(10+50)-ES		LINEAR		QUADRÁTICA		EXPONENCIAL	
	Média	Desvio Padrão	Média	Desvio Padrão	Média	Desvio Padrão	Média	Desvio Padrão
Atmosfera	0,9905	0,0004	0,9872	0,0005	0,9982	0,0344	**0,9717**	0,0058
Bebida	3,9931	0,0208	3,7223	0,0584	**3,4042**	0,1783	3,4246	0,0325
Consumo	3,4879	0,0336	**3,1938**	0,1543	4,7505	0,5053	3,8621	0,1559
Fortaleza	178,3926	0,1007	174,9947	0,5140	164,3455	0,3143	**159,0095**	0,7840
Icv	8,2304	1,0498	7,4290	0,5214	**6,8763**	0,7853	7,5798	0,4471
Ipi	2,6729	0,0127	**2,5207**	0,0057	2,5605	0,0511	2,5425	0,0072
Lavras	15,0337	0,0040	**13,9615**	0,0551	26,7077	1,6520	23,5049	5,3182
Manchas	**3,4297**	0,0188	3,6690	0,0329	9,2623	0,3862	4,0661	0,1822

5. Conclusão

Este trabalho propôs o uso de Estratégias Evolucionárias na tarefa de regressão para a Mineração de Dados Temporais. O algoritmo de ES foi aplicado em dados temporais (séries temporais univariadas).

O (10+50)-ES permitiu que as melhores soluções sobrevivessem ao longo das gerações. Objetivando aperfeiçoar os resultados de ES, aplicou-se o algoritmo AdaBoost.R2 ao algoritmo de mapeamento de uma função linear com Estratégias Evolucionárias. Isso permitiu que exemplos difíceis fossem focados.

Comparando o ESboost com as execuções do ES sem Boosting, observou-se que na maioria dos casos houve uma melhora pelo menos com uma das funções de perdas. Os resultados do algoritmo ESboost superaram os resultados do modelo ARMA em quase todas as séries temporais univariadas. Somente na base de dados Lavras, o desempenho foi pior, provando que o ESboost obtém bons resultados em séries temporais univariadas.

Alguns trabalhos futuros poderão constituir-se da aplicação de outras técnicas de aprendizado de máquina no lugar de ES e de estudos comparativos com outras metodologias para a predição de eventos.

References

Bäck, T., Rudolph, G. and Schewefel, H.P. (1993) “Evolutionary programming and evolution strategies: similarities and differences”. In: Annual Conference on Evolutionary Programming, 2., San Diego, La Jolla, 1993. p. 11-22.

Box, G.E.P. and Jenkins, G. M. (1976) Time series analysis: forecasting and control. Ed. Rev. San Francisco: Holden-Day.

Dianati, M., Song, I. and Treiber, M. (2002) An introduction to genetic algorithms and evolution strategies.

Fayyad, U., Piatetsky-Shapiro, G. and Smyth, P. (1996) “The KDD process for extracting useful knowledge from volumes of data”. In: Communications of the ACM, v.39 n.11, p.27-34.

Kout, J., Vlcek, T. and Klema, J. (2005) “Predictive System for Multivariate Time Series”. In: Automa International - Trade Journal for Industrial Automation. n. 11, p. 6-8.

Kusiak, A. (2000) “Evolutionary computation and data mining”. In: SPIE Conference on Intelligent System and Advanced Manufaturing, SPIE, Vol.4192, 2000, Boston, MA. B. Gopalakrishnan and Agunasekaran, nov. 2000. p.1-10.

Liu, B., Mckay, B. and Abbass, A. (2003) Improving genetic classifiers with a boosting algorithm. CEC 2003, p.2596-2602.

Morretin, P. A. and Toloi, C.M.C. (2004) Análise de Séries Temporais. Ed. Edgard Blucher.

NIST/SEMATECH e-Handbook of Statistical Methods. (2006) Disponível em: http://www.itl.nist.gov/div898/handbook/, mar.

Rechenberg, I. (1965) Cybernetic solution path of an experimental problem. Royal Aircraft Establishment, Library translation nº. 1122, Farnborough, Hants., UK, ago.

Roddick, J.F. and Spiliopoulou, M. (2002) “A Survey Of Temporal Knowledge Discovery paradigms and methods”. In: IEEE Transactions on Knowledge and Data Engineering, v.14, n.4, p. 750-767, jul./ago.

Schapire, R.E. (2002) The boosting approach to machine learning: an overview. In: MSRI Workshop on Nonlinear Estimation and Classification.

Solomatine, D.P. and Shretha, D. (2004) “AdaBoost.RT: a Boosting Algorithm for Regression Problems”. In: IEEE International Join Conference on Neural Networks, p.1163-1168.

Souza, L.V., Costa, E. and Pozo, A.T.R. (2005) “Análise da Capacidade da Programação Genética na Previsão de Séries Temporais”. In: Congresso de Métodos Numéricos em Engenharia, 2005, Granada.

Weiss, G.M. and Hirsh, H. (1998) “Learning to predict rare events in event sequences”. In: Int'l Conf. Knowledge Discovery and Data Mining (KDD '98), 4., R. Agrawal, P. Stolorz, and G. Piatetsky-Shapiro, eds., p. 359-363.