Classificação de Dados Através de Árvores de Perceptrons Geradas por Evolução Diferencial

Alan R. R. Freitas[1], **Rodolfo A. Lopes** [1], **Rodrigo C. P. Silva** [1], **Frederico G. Guimarães**[2]

[1]Programa de Pós-Graduação em Engenharia Elétrica
Universidade Federal de Minas Gerais
Av. Antônio Carlos 6627, 31270-901, Belo Horizonte, MG, Brasil

[2]Departamento de Engenharia Elétrica
Universidade Federal de Minas Gerais
Av. Antônio Carlos 6627, 31270-901, Belo Horizonte, MG, Brasil

{alandefreitas,rodolfo.ufop,rcpsilva}@gmail.com, fredericoguimaraes@ufmg.br

Resumo. *Classificação é uma das mais importantes tarefas de Mineração de Dados devido à sua capacidade preditiva e aplicação em diferentes áreas. No contexto de classificação, as Árvores de Perceptron (AP) têm sido aplicadas em diversos casos com bons resultados. Neste artigo é apresentado um algoritmo de evolução diferencial para a evolução de Árvores de Perceptron. Além disso, apresenta-se neste artigo o conceito de legitimidade que é utilizado para redução do custo de avaliação das soluções, que consome muito tempo de processamento. Testes comparativos utilizando diferentes bases de dados retiradas do repositório da UCI foram realizados. Eles indicam que a abordagem proposta é competitiva e muito promissora quando comparadas com outras técnicas de classificação tradicionais.*

Abstract. *Classification is one of the most important tasks in Data Mining due its predictive capacity and application in different fields. In this context, Perceptron Decision Trees (PDT) have been applied in some cases with good results. This paper presents a Differential Evolution algorithm that evolves Perceptron Decision Trees. Furthermore, we present the concept of legitimacy which is used to reduce the costs of solution evaluation, which consumes much processing time. Comparative tests utilizing different data sets collected from the UCI repository were conducted. It suggests that the proposed approach is competitive and very promising when compared with other traditional approaches for classification.*

1. Introdução

Devido à sua capacidade preditiva e aplicabilidade em diferentes áreas, classificação é uma das mais importantes tarefas em mineração de dados [de Campos Merschmann and Plastino 2010]. A tarefa consiste em encontrar um modelo que descreva e distingua classes de dados [Han et al. 2011]. O modelo é construído baseado na análise de um conjunto de dados de treinamento, que é usado para examinar características de uma nova instância (para a qual a classe é desconhecida) e atribuí-la uma classe.

Neste contexto, Árvores de Perceptrons (AP) foram utilizadas em vários problemas práticos de classificação com bons resultados [Bennett et al. 1998, Bennett and Mangasarian 1992b, Murthy et al. 1994]. Elas são árvores de decisão cujos nós testam uma combinação linear de atributos dividindo o espaço de atributos por hiperplanos [Bennett et al. 2000].

Neste artigo, mostra-se uma abordagem baseada em Evolução Diferencial (DE) para gerar AP. DE é um algoritmo evolucionário importante que funciona com uma população de soluções e iterativamente procura soluções de qualidade (Seção 3). Assim, DE funcionará com uma população de AP iterativamente minimizando seus erros de classificação.

Ao descrever a metodologia deste trabalho, introduzem-se as seguintes contribuições:

- Um algoritmo baseado em Evolução Diferencial para evoluir Árvores de Perceptrons (Seção 5)
- Uma abordagem para representação e manipulação de AP no contexto de soluções do DE (Seção 5.1-5.2)
- Um estratégia para reposição de soluções (Seção 5.3), na qual os piores classificadores dão lugar a novos
- Um método para controlar a legitimidade das avaliações (Seção 5.4) que faz a avaliação mais legítima à medida que se faz necessário

Testes comparativos com classificadores conhecidos são apresentados na Seção 6. Os resultados mostram que mesmo sendo necessário um estudo mais extensivo, o algoritmo proposto é muito promissor para construir modelos de classificação precisos (Seção 7).

2. O Problema de Classificação

Classificação consiste em predizer o valor de um atributo meta especificado pelo usuário (classe) baseado em valores dos outros atributos, chamados de atributos de predição [Freitas 2003]. No exemplo de classificação de um filme, o atributo alvo pode ser o *Gênero do Filme*, tendo os valores (classes) "ação" ou "drama", enquanto os atributos de predição podem ser o título do filme, os atores e a trilha sonora.

Na tarefa de classificação, quer-se construir um modelo que mapeia os atributos de predição às classes com precisão. Para este propósito, os dados sendo minerados são normalmente divididos em dois conjuntos de dados mutuamente excludentes, o conjunto de treinamento e o conjunto de teste. Na primeira fase, chamada de treinamento, o algoritmo de classificação constrói um modelo acessando os valores dos atributos e classes de cada exemplo do conjunto de treinamento.

Uma vez que o processo de treinamento está completo e o algoritmo construiu o modelo, sua capacidade de previsão é avaliada no conjunto de teste, que não foi visto durante o treinamento. Assim, se a performance do modelo construído é boa o suficiente, ele será utilizado para classificar novas instâncias cujas classes são desconhecidas.

3. Evolução Diferencial

Evolução Diferencial (DE) [Storn and Price 1997] é um importante Algoritmo Evolucionário (AE) desenvolvido para melhorar a qualidade de soluções ao longo de várias

iterações. No DE, novas soluções candidatas são criadas pela combinação de um pai com outros indivíduos da mesma população. Então, esta solução candidata substitui o pai se tiver um melhor valor.

Através do processo iterativo básico de ED como a descrita abaixo, espera-se que uma solução satisfatória seja encontrada:

- Gerar várias soluções $\mathbf{x}$ em posições aleatórias no espaço de busca
- Inicializar a probabilidade de cruzamento CR e o peso diferencial F
- Até que um critério de parada seja atingido, os seguintes passos são repetidos:
  - Para cada solução $\mathbf{x}$:
    - Escolher três outras soluções $\mathbf{a}$, $\mathbf{b}$ e $\mathbf{c}$ da população que são diferentes de $\mathbf{x}$.
    - Escolher um índice aleatório R entre 1 e o número de variáveis do problema.
    - Para gerar uma nova solução $\mathbf{y}$, para cada posição i da solução:
      - Gerar um valor real r entre 0 e 1 com distribuição uniforme
      - Se $r < CR$ ou $i = R$, $\mathbf{y}_i = \mathbf{a}_i + F(\mathbf{b}_i - \mathbf{c}_i)$, senão $\mathbf{y}_i = \mathbf{x}_i$
    - Se $\mathbf{y}$ é melhor que $\mathbf{x}$, $\mathbf{x}$ é substituído por $\mathbf{y}$
- Retornar a melhor solução conhecida

Os valores F, CR e o tamanho da população devem ser definidos pelo usuário. Dada uma formulação apropriada do problema de classificação de dados, o DE pode então evoluir regras de classificação para o conjunto de testes.

4. Árvores de Perceptron

Como mencionado na Seção 2, a tarefa de classificação envolve a geração de um modelo. Uma abordagem comum para classificadores é construir uma árvore binária de classificação. Nestas árvores, cada nó interno indica um teste sobre um atributo, cada ramo indica um resultado possível do teste e cada nó folha corresponde a uma classe. Desta maneira, os testes associados com cada nó são equivalentes a hiperplanos paralelos aos eixos do espaço de entradas [Bennett et al. 2000].

Neste trabalho, utiliza-se uma abordagem diferente para definir cada nó da árvore. Nesta abordagem, os nós internos testam uma combinação linear dos atributos. Com a equação $\mathbf{w}\mathbf{x} + \theta = z$, onde $\mathbf{w}$ são escalares de peso, $\mathbf{x}$ são os atributos dos dados e θ é uma constante, pode-se dividir o espaço de atributos em hiperplanos, que não são necessariamente paralelos aos eixos. Com esta abordagem, definimos uma AP.

Assim, o espaço de atributos pode ser dividido considerando todos os atributos dos dados a cada passo, diferentemente da árvore binária convencional. Caso se precise considerar apenas o atributo i na AP, o vetor $\mathbf{w}$ deve ser constituído de zeros, com exceção da posição $\mathbf{w}_i$, onde seu valor deve ser 1.

Vários autores empregaram conceitos similares para agregar vários classificadores lineares ou perceptrons em uma árvores empregando diferentes nomes [Bennett and Mangasarian 1992a, Bennett and Mangasarian 1994, Breiman 1984, Brodley and Utgoff 1995], incluindo AP [Bennett et al. 2000].

5. Metodologia

Nesta seção, descreve-se como DE foi utilizada para evoluir AP e o procedimento utilizado para validar nossa abordagem, com comparações com outros classificadores conhecidos baseados em árvores.

5.1. Representação de um Classificador

Em nosso método, cada AP é uma árvore binária completa com classificadores lineares nos nós internos. Cada um destes é definido por um vetor de pesos $\mathbf{w}$ de tamanho n e uma constante θ, sendo n o número de atributos do problema de classificação. Assim, o número de classificadores em uma AP é $2^{(h-1)} - 1$, sendo h uma variável de controle definida como a profundidade da árvore.

Além dos classificadores, cada AP contém $2^{(h-1)}$ nós folha, que contêm uma classificação possível para cada amostra. Estes nós são definidos no espaço discreto.

Assim, uma AP completa pode ser definida por matrizes de tamanho $n \times 2^{h-1} - 2$ para os pesos, $1 \times 2^{h-1} - 1$ para as constantes e $1 \times 2^{h-1}$ para os nós folha. Este é o tamanho de cada solução empregada pelo DE.

Enquanto o tamanho do indivíduo é $O(2^{h-1} - 1)$, o custo de avaliação é ainda $O(h)$ pois apenas um caminho possível é pesquisado na AP. Para h igual a 3, 5 e 7, por exemplo, as AP têm tamanho de representação 3, 15 e 63, respectivamente.

Cada uma das $2^{h-1} - 1$ colunas das matrizes define um classificador. Depois de utilizar um classificador linear na posição i de n, o classificador na posição $2i$ é usado se a saída do classificador é < 0 ou o classificador $2i + 1$ é usado caso contrário.

Como o erro total é a medida de comparação entre as AP da população, o valor de função objetivo de cada uma das soluções é definida como o número de classificações errôneas, incluindo falsos positivos e falsos negativos. Esta função objetivo, naturalmente, deve ser minimizada pelo DE.

5.2. Operadores

Operadores simples, como mostrados na Seção 3 foram empregados para a geração de novas soluções. Valores de F e CR foram definidos como valores aleatórios entre 0.4 e 1 e entre 0.9 e 1, respectivamente [Storn and Price 1997]. Estes valores são alterados a cada iteração de uma geração.

Para os nós folha, uma abordagem discreta deve ser utilizada para o cruzamento das soluções. A abordagem utilizada [Prado et al. 2010] é inspirada na ideia de que o DE utiliza um vetor de diferença para alterar a solução. Contudo, agora este vetor representa movimentos de troca entre duas posições possíveis.

Fazendo isto, o movimento descrito como $\mathbf{b}_i - \mathbf{c}_i$ é apenas executado se $\mathbf{a}_i = \mathbf{b}_i$. Se $\mathbf{a}_i = \mathbf{c}_i$, o movimento $\mathbf{c}_i - \mathbf{b}_i$ é aplicado. Em um terceiro caso possível, onde $\mathbf{a}_i \neq \mathbf{b}_i$ e $\mathbf{a}_i \neq \mathbf{c}_i$, nenhum movimento é aplicado.

Neste caso, o valor F é utilizado para definir se a operação deve acontecer. A operação ocorre se um número aleatoriamente gerado é menor que F.

5.3. Reposição de Indivíduos

Como a reposição de indivíduos é feita um a um, pode ocorrer que algumas soluções estejam estagnadas no espaço de busca. Para evitar este problema, um novo operador de sobrevivência dos indivíduos foi criado para sempre manter novos candidatos na população.

A cada geração, novos indivíduos aleatórios são gerados para substituir os $x\%$ piores indivíduos. Mais ainda, os indivíduos com uma taxa de acerto menor que $1/n_{classes}$ também são substituídos.

5.4. Legitimidade

Nas gerações iniciais, quando todos os indivíduos ainda são completamente aleatórios, não são necessárias várias comparações para perceber que alguns são melhores que os outros. Na maior parte dos casos, com o emprego dos classificadores em menos de 10 amostras é possível definir claramente como algumas AP são melhores que outras. O mesmo não ocorre após várias gerações, quando a maior parte das soluções classificam as amostras com uma baixa taxa de erro.

Com isto em mente, e o alto custo de avaliar soluções, propõe-se uma abordagem para reduzir o tempo gasto para avaliar soluções nas primeiras gerações.

A idéia chave é que as soluções iniciais não precisam ser avaliadas com a mesma legitimidade que as soluções nas últimas gerações, onde uma análise mais refinada é necessária para distinguir boas soluções. O valor l define a legitimidade utilizada na avaliações dos indivíduos. Na geração com legitimidade l, apenas l amostras são testadas na avaliação de cada solução.

O valor inicial deste parâmetro foi definido como o dobro do número de atributos de cada amostra. Após cada geração, o valor l é atualizado de acordo com a equação $l = \lceil \min(N, l * \alpha) \rceil$, onde N é o número de amostras disponíveis para treinamento e α é a taxa de aumento na legitimidade a cada geração.

O parâmetro α foi definido como $1 + \delta/\sigma$, onde δ é a velocidade de legitimização, definida como 10^{-6}, e σ é o desvio padrão do valor de função objetivo das soluções. Assim, quando as soluções são ainda muito diversas, o parâmetro α é menor.

Alguns indivíduos podem ocasionalmente ter um valor de função objetivo melhor do que lhe é devido em casos onde a legitimização é baixa. Por esta razão, quando o valor de função objetivo é calculado, o valor de legitimidade utilizado para o obter o valor de função objetivo é também guardado. Assim, se o indivíduo não é substituído após t gerações e o valor de legitimidade já aumentou, o indivíduo é reavaliado com o novo valor de legitimidade l. O valor t foi definido como 10.

6. Resultados

Nesta seção, apresenta-se os resultados computacionais. Nossa abordagem usando a técnica de legitimidade foi comparada com outros classificadores baseados em árvores bem conhecidos: J48 [Quinlan 1993], BFTree [Shi 2007] e RandomTree [Frank et al. 2005]. Os resultados destes métodos foram obtidos através do uso do *Weka Framework*[1] [Frank et al. 2005].

[1]Disponível para download em http://www.cs.waikato.ac.nz/~ml/weka/index_downloading.html .

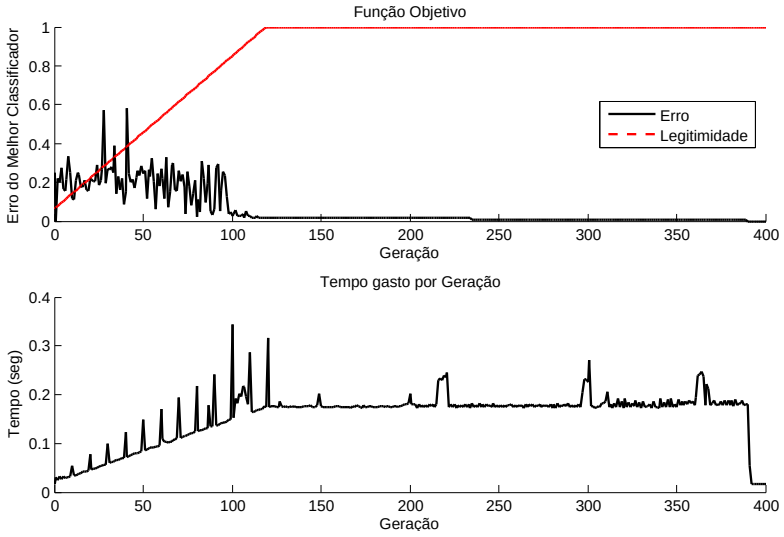

Figure 1. Controle de Legitimidade

Para as comparações, oito bases de dados foram retiradas do repositório UCI[2]. Suas características são descritas na Tabela 1. Elas foram divididas em dois conjuntos, 85% para o treinamento e 15% para teste. É importante notar que para comparações justas entre os métodos, os mesmos conjuntos foram utilizados por todos.

Table 1. Descrição das Bases de Dados

Base de Dados	Instâncias	Atributos	Classes
Breast Tissue	106	10	6
Vertebral Column	310	7	3
Ecoli	336	8	8
Glass Identification	214	10	6
Hill-Valey	1212	101	2
Iris	150	5	3
Libras Movement	360	91	15
Sonar vs. Rocks	208	61	2

A abordagem proposta tem a seguinte configuração, definida em uma série de testes preliminares:

- Tamanho da população : 50 indivíduos
- Número máximo de gerações: 2000
- Altura da AP (h): 5
- Taxa de reposição de indivíduos: 0.1

Devido à natureza estocástica de nosso método, os resultados relacionados a ele apresentam um resumo de 30 execuções independentes. Na Tabela 2, o teste de kruskall-wallis (com nível de significância $\alpha = 0.05$) foi utilizado para comparar os resultados médios obtidos por nosso método com os resultados obtidos por outros. Como pode ser visto, a abordagem proposta foi significantemente melhor que a J48 e a RandomTree na

[2]Disponível em http://archive.ics.uci.edu/ml/

maior parte das bases de dados. O método apresentou também resultados competitivos quando comparados com a BFTree.

Table 2. Comparação entre AP evoluídas com DE e outras técnicas

Base de Dados	J48	BFTree	RandomTree
Melhores Resultados	4	3	5
Piores Resultados	3	3	3
Resultados Iguais	1	2	0

A Tabela 3 apresenta os resultados obtidos por cada um dos métodos. Para nossa abordagem, que é chamada de AP pela simplicidade, apresentam-se os piores resultados (Mín), melhores resultados (Máx), a média e o desvio padrão de 30 execuções. O melhor resultado da abordagem proposta foi melhor em 5 de 8 bases de dados. Os outros métodos tiveram os melhores resultados em apenas uma base de dados cada. Isto revela o potencial do método proposto, contudo, mais estudo e esforço deve ser empregado para reduzir a variância da qualidade das soluções.

Table 3. Resultados Comparativos - Taxa de Acerto

Base	AP-Mín	AP-Média(std)	AP-Máx.	J48	BFTree	R.Tree
Breast T.	0.3750	0.5292 (0.0910)	0.6875	0.625	**0.8125**	0.75
V. Column	0.7660	0.8411 (0.0294)	**0.9149**	0.851	0.851	0.8297
Ecoli	0.6863	0.7673 (0.0430)	**0.8627**	0.745	0.7647	0.6666
Glass I.	0.4242	0.5808 (0.0690)	0.6667	0.6969	0.7575	**0.7878**
Hill-V.	0.9341	0.9945 (0.0140)	**1.0**	0.5109	0.5769	0.6318
Iris	0.8261	0.9435 (0.0305)	**0.9565**	0.8695	0.8695	0.8695
Libras M.	0.1852	0.3185 (0.0616)	0.4074	**0.7222**	0.6111	0.6666
Sonar vs. R.	0.6563	0.7729 (0.0614)	**0.9063**	0.5312	0.625	0.7187
		Média	**0.8002**	0.6939	0.7335	0.7400

A Tabela 4 mostra o erro obtido pelo método na fase de treinamento. Pode-se observar que nas bases onde os resultados são fracos, o método teve dificuldades para minimizar os erros durante a fase de treinamento, terminando o processo com um erro alto. Talvez o critério de parada de 2000 gerações tenha sido muito apertado nestes casos.

Table 4. Treinamento - Taxa de Erro

Base de Dados	Erro Mínimo	Erro Médio (Desvio Padrão)	Erro Máximo
Breast Tissue	0.2222	0.2952 (0.0473)	0.4111
Vertebral Column	0.1065	0.1343 (0.0130)	0.1559
Ecoli	0.0982	0.1304 (0.0307)	0.2105
Glass Identification	0.2099	0.3098 (0.0472)	0.4199
Hill-Valley	0.0	0.0054 (0.0133)	0.0602
Iris	0.0	2.6247e-004 (0.0014)	0.0079
Libras Movement	0.5327	0.5874 (0.0289)	0.6438
Sonar vs. Rocks	0.0511	0.0977 (0.0219)	0.1591

7. Conclusão

Este trabalho apresenta uma abordagem para construir modelos de classificação baseados em Árvores de Perceptrons. Para o treinamento do modelo, utilizou-se uma adaptação do algoritmo de Evolução Diferencial que funciona com ambos os tipo de variáveis, contínuas e discretas, presentes no modelo. Uma vez que a avaliação das soluções tem alto custo no procedimento apresentado, este trabalho também apresentou o conceito de legitimidade, que permite uma redução no tempo de avaliação das soluções nas primeiras gerações.

Os resultados indicam que a abordagem proposta tem um bom potencial para construir modelos precisos e as melhores soluções encontradas, considerando todas as execuções do método, obtiveram os melhores resultados na maior parte das bases de dados.

Apesar do sucesso das melhores soluções encontradas entre as replicações, o teste estatístico revelou que o desempenho médio do método é apenas competitivo quando comparado com outros classificadores importantes. Além disto, os resultados obtidos durante a fase de treinameto mostraram que o método teve dificuldades para minimizar o erro em algumas bases de dados.

Para melhorar o desempenho médio e para tratar o problema de minimização do erro, pode-se empregar um conjunto de validação. O conjunto de validação, pode ser extraído do conjunto de treinamento e funcionará como conjunto de teste dentro do treinamento. O conjunto de validação pode aumentar a capacidade de generalização dos classificadores, melhorando a precisão e pode ser utilizado para dar apoio ao critério de parada e para o mecanismo de reposição, melhorando a minimização do erro.

Em suma, a abordagem proposta parece ser muito promissora. Porém, estudos mais profundos devem ser realizados para melhorar a robustez do método e mais comparações com outros algoritmos de classificação também devem ser feitos para uma validação mais apropriada do método.

Agradecimentos

O presente trabalho foi realizado com o apoio financeiro da CAPES, CNPq e FAPEMIG - Brasil.

References

Bennett, K., Cristianini, N., Shawe-Taylor, J., and Wu, D. (2000). Enlarging the margins in perceptron decision trees. *Machine Learning*, 41(3):295–313.

Bennett, K. and Mangasarian, O. (1992a). Robust linear programming discrimination of two linearly inseparable sets. *Optimization methods and software*, 1(1):23–34.

Bennett, K. and Mangasarian, O. (1994). Multicategory discrimination via linear programming. *Optimization Methods and Software*, 3(1-3):27–39.

Bennett, K. P., Auslender, L., Wu, D., and Ave, S. (1998). On support vector decision trees for database marketing. Technical report, Department of Mathematical Sciences Math Report No. 98-100, Rensselaer Polytechnic Institute.

Bennett, K. P. and Mangasarian, O. L. (1992b). Multicategory discrimination via linear programming. *Optimization Methods and Software*, 3:27–39.

Breiman, L. (1984). *Classification and regression trees*. Chapman & Hall/CRC.

Brodley, C. and Utgoff, P. (1995). Multivariate decision trees. *Machine Learning*, 19(1):45–77.

de Campos Merschmann, L. H. and Plastino, A. (2010). Hisp-gc: A classification method based on probabilistic analysis of patterns. *JIDM*, 1(3):423–438.

Frank, E., Hall, M., Holmes, G., Kirkby, R., Pfahringer, B., Witten, I. H., and Trigg, L. (2005). Weka. In Maimon, O. and Rokach, L., editors, *Data Mining and Knowledge Discovery Handbook*, pages 1305–1314. Springer US. 10.1007/0-387-25465-$X_6$2.

Freitas, A. A. (2003). Advances in evolutionary computing. chapter A survey of evolutionary algorithms for data mining and knowledge discovery, pages 819–845. Springer-Verlag New York, Inc., New York, NY, USA.

Han, J., Kamber, M., and Pei, J. (2011). *Data Mining: Concepts and Techniques*. The Morgan Kaufmann Series in Data Management Systems. Elsevier.

Murthy, S. K., Kasif, S., and Salzberg, S. (1994). A system for induction of oblique decision trees. *Journal of Artificial Intelligence Research*, 2:1–32.

Prado, R. S., Pedrosa Silva, R. C., Guimarães, F. G., and Neto, O. M. (2010). Using differential evolution for combinatorial optimization: A general approach. In *Proceedings of the 2010 IEEE International Conference on Systems, Man and Cybernetics (SMC)*, pages 11–18.

Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning (Morgan Kaufmann Series in Machine Learning)*. Morgan Kaufmann, 1 edition.

Shi, H. (2007). Best-first decision tree learning. Master's thesis, University of Waikato, Hamilton, NZ.

Storn, R. and Price, K. (1997). Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization*, 11(4):341–359.