Embedded Feature Selection Using Probabilistic Model-Based Optimization

Shota Saito
Yokohama National University
Kanagawa, Japan
saito-shota-bt@ynu.jp

Shinichi Shirakawa
Yokohama National University
Kanagawa, Japan
shirakawa-shinichi-bg@ynu.ac.jp

Youhei Akimoto
University of Tsukuba
Ibaraki, Japan
akimoto@cs.tsukuba.ac.jp

ABSTRACT

In machine learning, feature selection is a commonly used technique for improving the predictive performance and interpretability of a trained model. Feature selection techniques are classified into three approaches: the filter, wrapper, and embedded approaches. The embedded approach performs the feature selection process during the model training and achieves a good balance between performance and computational cost in general. In the paper, we propose a novel embedded feature selection method using probabilistic model-based evolutionary optimization. We introduce the multivariate Bernoulli distribution, which determines the selection of features, and we optimize its parameters during the training. The distribution parameter update rule is the same as that of the population-based incremental learning (PBIL), but we simultaneously update the parameters of the machine learning model using an ordinary gradient descent method. This method can be easily implemented into non-linear models, such as neural networks. Moreover, we incorporate the penalty term into the objective function to control the number of selected feature. We apply the proposed method with the neural network model to the feature selection of three classification problems. The proposed method achieves competitive performance and reasonable computational cost compared with conventional feature selection methods.

CCS CONCEPTS

• **Computing methodologies** → **Feature selection**; **Machine learning**; • **Theory of computation** → *Evolutionary algorithms*;

KEYWORDS

Feature Selection, Embedded Approach, Natural Gradient, Information Geometric Optimization, Neural Network

ACM Reference Format:
Shota Saito, Shinichi Shirakawa, and Youhei Akimoto. 2018. Embedded Feature Selection Using Probabilistic Model-Based Optimization. In *GECCO '18 Companion: Genetic and Evolutionary Computation Conference Companion, July 15–19, 2018, Kyoto, Japan.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3205651.3208227

GECCO '18 Companion, July 15–19, 2018, Kyoto, Japan

1 INTRODUCTION

In machine learning, feature selection is useful for improving the predictive performance and identifying which features are important. Feature selection techniques are generally classified into three approaches: the filter, wrapper, and embedded approaches [4]. Evolutionary computation has been mainly applied to the wrapper and filter approaches [21]. Whereas, the embedded approach is considered as having a good balance between computational cost and accuracy because it performs the selection of features during the model training. The typical method of the embedded approach is LASSO [18] and similar methods [12, 22], which use the L1 regularization for the weight coefficients in a linear model. LASSO can be only applied to the linear model in general and is difficult to be applied to the complex models such as neural networks as the feature selection method.

In this paper, we propose a novel method called the probabilistic model-based embedded feature selection (PEFS) that can easily incorporate non-linear models such as neural networks. In the PEFS, we define the multivariate Bernoulli distribution of the binary vector and use the vector for feature selection. The key technique to realize the embedded feature selection is to optimize the parameters of the distribution during the model training introduced in [17]. We formulate the objective function to be minimized as the expectation of loss function under the distribution for the feature selection and show that such formulation relates to the information geometric optimization (IGO) [14] and population-based incremental learning (PBIL) [2]. Different from the IGO and PBIL, we simultaneously optimize the parameters of machine learning model (e.g., the weight parameters of neural network) using an ordinary gradient descent method. Furthermore, we introduce the penalty term into the objective function to control the number of selected features.

To evaluate the performance of the PEFS, we apply the PEFS with a neural network model to three classification problems. From the experimental results, the PEFS shows the competitive performance and reasonable computational cost compared with some existing feature selection methods.

The contribution of this paper is as follows: (1) applying the idea introduced in [17], which simultaneously optimizes both of the weight parameters of neural network and the Bernoulli parameters based on the gradient descent[1], to the feature selection problem, and (2) incorporating the penalty term into the objective function to control the number of selected features.

[1] The framework proposed in [17] uses the Bernoulli distribution to represent the neural network structures such as the selection of layers or units and optimize the neural network structures during the training.

2 PEFS: THE PROPOSED METHOD

We consider the general problem setting of supervised learning with a given training dataset $\mathcal{D} = \{X, Y\}$. Let $X = \{x_1, \dots, x_N\}$ be the set of input vectors, $x_i \in \mathbb{R}^d$, and $Y = \{y_1, \dots, y_N\}$ be the set of target variables. In the classification problem, Y refers to the labels, whereas it refers to continuous values in the regression problem.

In the PEFS, the features to be used are selected by a random variable consisting of d-dimensional binary vector $M = (m_1, \dots, m_d)^\top$, $m_i \in \{0, 1\}$. Given a binary vector M and an input vector x, the input of the prediction model is represented by $\mathrm{diag}(M)x$, where diag represents a diagonal matrix whose diagonal elements are zero or one depending on the corresponding element of M. This means that the values of the unused features become zero. We consider the probability distribution for the random variable M as the multivariate Bernoulli distribution defined by $p(M \mid \theta) = \prod_{i=1}^{d} \theta_i^{m_i}(1-\theta_i)^{1-m_i}$, where $\theta = (\theta_1, \dots, \theta_d)^\top$, $\theta_i \in [0, 1]$ refers to the parameters of the distribution. The parameter θ_i corresponds to the probability that the i-th feature is selected, i.e., $m_i = 1$.

Let us denote the model to predict the target variable from the input vector as $\phi(X, W, M)$, and the loss function to be minimized as $\mathcal{L}(Y, X, W, M)$. We often use the mean squared errors for regression and the softmax cross-entropy losses for classification. We note that the filter approach in feature selection optimizes the model parameter W using the pre-selected binary vector M, whereas the wrapper approach searches the better binary vector M based on the validation error of the optimized model parameter W. Therefore, the wrapper approach generally requires a much computational cost. Since we treat the binary vector M as the random variable, we consider minimizing the expected loss function as $\mathcal{G}(W, \theta) = \sum_{M \in \mathcal{M}} \mathcal{L}(Y, X, W, M) p(M \mid \theta)$.

Furthermore, to reduce and control the number of selected features, we add the term with respect to (w.r.t.) the number of selected features $\sum m_k$ as the penalty. Considering the loss function with the penalty of the number of selected features, its expectation under $p(M|\theta)$ is given by

$$\mathcal{G}(W, \theta) = \mathbb{E}\left[\mathcal{L}(Y, X, W, M) + \epsilon \sum_{k=1}^{d} m_k\right] \tag{1}$$

$$= \sum_{M \in \mathcal{M}} \mathcal{L}(Y, X, W, M) p(M \mid \theta) + \epsilon \sum_{k=1}^{d} \theta_k \ , \tag{2}$$

where ϵ is the coefficient to control the influence of the penalty.

In the case of $\epsilon = 0$, the minimization of $\mathcal{G}(W, \theta)$ w.r.t. θ can be viewed as the same formulation in the information geometric optimization (IGO) [14] with the objective function $\mathcal{L}(Y, X, W, M)$. The IGO updates the parameters of the distribution toward the natural gradient direction [1], which is the steepest direction of θ w.r.t. the KL-divergence. The natural gradient $\tilde{\nabla}_\theta \mathcal{G}(W, \theta)$ is given by

$$\tilde{\nabla}_\theta \mathcal{G}(W, \theta) = \sum_{M \in \mathcal{M}} \mathcal{L}(Y, X, W, M) \tilde{\nabla}_\theta \ln p(M \mid \theta) + \epsilon \tilde{\nabla}_\theta \sum_{k=1}^{d} \theta_k \ , \tag{3}$$

where the natural gradients of the log-likelihood and the penalty are given by $\tilde{\nabla}_\theta \ln p(M \mid \theta) = \mathrm{F}^{-1}(\theta) \nabla_\theta \ln p(M \mid \theta)$ and $\tilde{\nabla}_\theta \sum_{k=1}^{d} \theta_k = \mathrm{F}^{-1}(\theta) \nabla_\theta \sum_{k=1}^{d} \theta_k$, respectively, and $\mathrm{F}^{-1}(\theta)$ denotes the Fisher information matrix of $p(M \mid \theta)$. Since we consider the Bernoulli distribution as $p(M \mid \theta)$, the natural gradients can be obtained analytically as $\tilde{\nabla}_\theta \ln p(M \mid \theta) = M - \theta$ and $\tilde{\nabla}_\theta \sum_{k=1}^{d} \theta_k = \theta(1-\theta)$.

Unlike the IGO and PBIL, minimizing $\mathcal{G}(W, \theta)$ requires optimization not only w.r.t. the distribution parameters θ but also w.r.t. the model parameters W. The gradient w.r.t. W is given by

$$\nabla_W \mathcal{G}(W, \theta) = \sum_{M \in \mathcal{M}} \nabla_W \mathcal{L}(Y, X, W, M) \ . \tag{4}$$

We note that the gradient $\nabla_W \mathcal{L}(Y, X, W, M)$ can be computed by the back-propagation when the neural network model is considered.

We realize the embedded feature selection by simultaneously optimizing θ and W using (3) and (4), respectively. In practice, the gradients (3) and (4) are approximated by Monte-Carlo method using λ samples of M drawn from $p(M \mid \theta)$.

In most practical cases, the loss $\mathcal{L}(Y, X, W, M_i)$ is approximated using mini-batch samples $\mathcal{Z} = \{(x_1, y_1), \dots, (x_{\bar{N}}, y_{\bar{N}})\}$. Referring to [17], we use the same mini-batch between different M_i to obtain the accurate rankings of losses. The approximated loss function is given by $\bar{\mathcal{L}}(Y, X, W, M_i) = \bar{N}^{-1} \sum_{(y,x) \in \mathcal{Z}} l(y, x, W, M_i)$, where $l(y, x, W, M)$ represents the loss of a datum, and $\bar{N}$ indicates the number of the mini-batch samples. Also, we transform the loss value into the ranking-based utility as done in [17] as follows:

$$\bar{\mathcal{L}}(Y, X, W, M_i) \mapsto u_i = \begin{cases} 1 & (\text{best } \lceil \lambda/4 \rceil \text{ samples}) \\ -1 & (\text{worst } \lceil \lambda/4 \rceil \text{ samples}) \\ 0 & (\text{otherwise}) \end{cases} . \tag{5}$$

With the transformation, the θ update using λ samples drawn from $p(M \mid \theta)$ reads

$$\theta^{(t+1)} = \theta^{(t)} + \eta_\theta \left(\sum_{i=1}^{\lambda} \frac{u_i}{\lambda} (M_i - \theta^{(t)}) - \epsilon \theta^{(t)} (1 - \theta^{(t)}) \right) \ , \tag{6}$$

where η_θ is the learning rate for θ. If ϵ is zero, the update rule of θ recovers the PBIL [2] which is one of the estimation of distribution algorithms. Moreover, we restrict the range of θ within $[1/d, 1-1/d]$ in order to keep the possibility of generating any binary vector.

As proposed in [17], there are two options to predict a new data using optimized θ and W. First, the binary vectors are sampled from $p(M \mid \theta)$ and the prediction results are averaged. While this prediction method can reach an accurate prediction, it is not a desirable approach from the feature selection perspective because it may use all features. Another way is to deterministically decide the binary vector M such that $m_i = 1$ if $\theta_i \geq 0.5$; otherwise, $m_i = 0$. In the experiment, we use this deterministic prediction and report the results.

3 EXPERIMENT AND RESULT

In this section, we evaluate the PEFS with the neural network model on three classification problems. The datasets used in the experiment are summarized in Table 1.

3.1 Experimental Setting

For all datasets, the neural network consisting of three fully connected hidden layers with 200 units and the softmax output layer is used. Each hidden unit performs the rectified linear unit (ReLU) [13] as the activation, followed by the batch normalization [8]. The

Table 1: The summary of datasets used in the experiment.

Dataset	No. of data (training / test)	No. of features	No. of classes	Data domain
pcmac [11]	1555 / 388	3289	2	Article
usps [7]	7439 / 1859	256	10	Image
adult [10]	36178 / 9044	104	2	Income

number of units in the input and output layers depends on the number of features and classes. The weight parameters are initialized by He’s initialization [6]. We use the cross-entropy loss function as $\mathcal{L}$. The weight parameters are optimized using Adam [9], an accelerated stochastic gradient method, with the default parameter setting, and we use the weight decay of 10^{-4}.

In the PEFS, we initialize the distribution parameters by $\theta_{\text{init}} = 0.5$ and use the learning rate of $\eta_\theta = 1/d$, where d is the number of features. The mini-batch size $\bar{N}$ is set to $\bar{N} = 128$ in the PEFS. The sample size λ is set to two[2]. We vary the coefficient ϵ from 0 to 1 in 0.1 increments to check the effect. The natural gradient corresponding to $\mathcal{L}$ is bounded in $[-0.5, 0.5]^d$ because of the utility transformation defined in (5), and the one corresponding to the penalty term is within $[0, \epsilon/4]^d$.Therefore, it is reasonable to set ϵ in $[0, 1]$ independently of dataset s. The parameter update in the training is terminated at 100,000 iterations for all datasets.

We compare the performance of the PEFS with those of existing feature selection methods. For the comparison, the minimum redundancy and maximum relevance (mRMR) [15] and ReliefF [16] were used as the filter approach; the wrapper method using particle swarm optimization (PSO), which is called ErFS in [20], was used as the wrapper approach; and LASSO with a linear model was used as the embedded approach. Filter methods select the features based on the feature ranking and train the above-mentioned neural network model using high-ranked feature subset. The fitness used in ErFS is based on the error of the randomly selected 30% validation data from the training data. We set the numbers of individuals and generations to 10 and 25, respectively, in ErFS. This setting uses smaller values than those used in [20] to reduce the computational cost because the above-mentioned neural network model is more computationally expensive than the model used in [20]. In these existing methods, the mini-batch size of the neural network training is set to $\lambda\bar{N}$, which is the same number of data samples to be used at one iteration in the PEFS, for a fair comparison. In LASSO, the linear model is optimized by the SAGA [3] with 500 iterations. We employ the different regularization factors from 1 to 10 in 1 increments. The features are selected if the trained model parameters are greater than 10^{-5} in LASSO.

3.2 Result and Discussion

Prediction Performance and Number of Features. Figure 1 shows the relation between the percentile of selected features and the test error. The result of the different numbers of selected features, from 10% to 100% in 10% increments, are reported in the filter methods. The plotted values are the average values of test errors of 10 different neural network trainings. In the PEFS, we also report the average test errors and average percentile of selected features of 10 different trainings for each setting. The results of the PSO-based wrapper method and LASSO are reported for a single run.

Table 2: Computational time (hour) of a typical single run. We report the total time of the feature selection and neural network training. Note that LASSO uses the linear model.

Dataset	Without FS	Embedded PEFS / LASSO	Filter ReliefF / mRMR	Wrapper PSO
pcmac	0.675	0.748 / 0.010	0.857 / 11.1	169
usps	0.451	0.659 / 0.027	0.549 / 1.05	114
adult	0.458	0.601 / 0.009	1.27 / 0.682	113

Comparing the PEFS and filter methods, the PEFS outperforms the filter methods for small feature subset around from 10% to 40% in the usps and adult datasets. For the pcmac dataset, the performances of the PEFS outperforms the filter method for selected features of around 50%.

Comparing the PEFS and LASSO as an embedded approach, the PEFS outperforms LASSO in the usps dataset on the same percentile of selected features. However, LASSO outperforms the PEFS and filter methods in the pcmac dataset. Interestingly, the PEFS reaches the equivalent performance with LASSO when the fewer number of features are selected in the adult dataset. We consider that this performance difference is caused by the model difference; i.e., LASSO uses the linear model. These results demonstrate that the PEFS can take good advantage of the neural network model as the embedded approach. In addition, the percentile of the selected features in LASSO is not distributed compared to the PEFS. This is because the appropriate scale of the regularization factor in LASSO depends on the dataset. In contrast, the PEFS does not require to tune the scale of ϵ by the utility transformation.

Next, we compare the test error with ErFS, the PSO-based wrapper approach. ErFS outperforms the filter methods except for the pcmac dataset. In general, the wrapper method is superior to the filter method, but it is computationally expensive when the complicated model is used. In our setting, the numbers of individuals and generations may not be sufficient to reach the fewer test errors. The performances of the PEFS and ErFS are comparable on the same percentile of selected features.

Computational Costs. We compare the total computational times of the feature selection and neural network trainings. Table 2 shows the computational times of a typical single run for each method and the case of using all features (i.e., without feature selection (FS)). For three datasets, the computational time of the PEFS was about 1.2 to 1.5 times as long as without the feature selection, while ReliefF and mRMR require more computational time as the number of training data or feature increase. The computational complexity of ReliefF and mRMR is $O(dN)$ [16] and $O(d^2)$ [19], respectively. LASSO is fast because it uses the linear model, unlike the other methods. Obviously, the PSO-based wrapper method requires significant computational cost because it needs the cycle of the model training to evaluate the feature subset.

[2] The θ update in (6) with $\lambda = 2$ and $\epsilon = 0$ recovers the update rule of the compact genetic algorithm (cGA) [5].

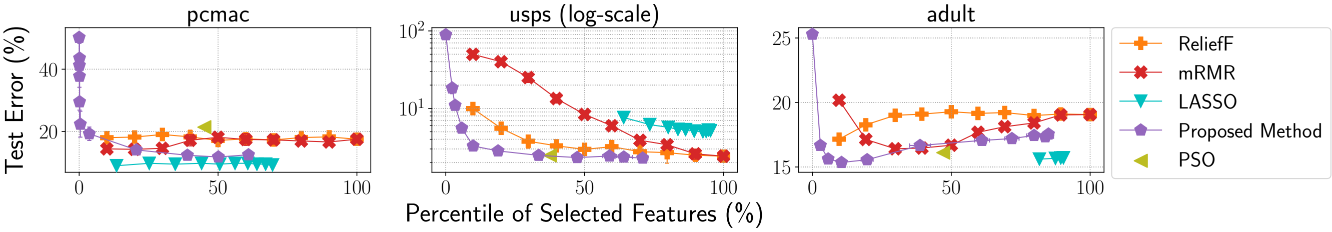

Figure 1: The relations between the percentile of selected features and the test error for each dataset. The vertical and horizontal axes represent the test error rate and the percentile of the selected features, respectively. For the mRMR, ReliefF and PEFS (proposed method), the average value and standard deviation of 10 different model trainings are plotted.

In the PEFS, the additional computational time for the feature selection does not have a significant impact because the additional computation is the computation regarding the sampling and updating of $p(M \mid \theta)$. The experimental result shows that the PEFS can select features in a reasonable computational time even when using neural network models. Moreover, the coefficient of the penalty did not affect the computational time in our implementation.

4 CONCLUSION

In this paper, we have proposed a novel embedded feature selection method using probabilistic model-based optimization, called the PEFS. The distribution for selecting features is updated by the natural gradient like IGO, while the model parameters are optimized by a usual gradient descent method simultaneously. Moreover, we have introduced the penalty based on the number of selected features. Although evolutionary algorithms have often been used for the wrapper approach, our method has shown the potential of evolutionary algorithms for embedded feature selection. We evaluated the PEFS with the neural network on three classification problems and compared the performance of our method with those of existing feature selection methods. We confirmed that introducing the penalty can control the number of selected features. The PEFS achieved the competitive test error and reasonable computational time and particularly outperformed the filter methods in the small number of selected features.

A future work can formulate the penalty under the prior distribution. Another future work can combine the PEFS with the dynamic model structure optimization technique introduced in [17], which is based on the same framework used in this paper. We can simply combine these methods and simultaneously optimize the model structure, the selected features, and the model parameters.

ACKNOWLEDGMENT

This work is partially supported by the SECOM Science and Technology Foundation.

REFERENCES

[1] Shun-ichi Amari. 1998. Natural Gradient Works Efficiently in Learning. *Neural Computation* 10, 2 (1998), 251–276.

[2] Shumeet Baluja. 1994. Population based incremental learning: A method for integrating genetic search based function optimization and competitve learning. *CMU-CS-94-163, Carnegie-Mellon Univ Pittsburgh Pa Dept Of Computer Science* (1994).

[3] Aaron Defazio, Francis Bach, and Simon Lacoste-Julien. 2014. SAGA: A Fast Incremental Gradient Method With Support for Non-Strongly Convex Composite Objectives. In *Proceedings of Advances In Neural Information Processing Systems (NIPS) 27*.

[4] Isabelle Guyon and André Elisseeff. 2003. An Introduction to Variable and Feature Selection. *Journal of Machine Learning Research* 3 (2003), 1157–1182.

[5] Georges R. Harik, Fernando G. Lobo, and David E. Goldberg. 1999. The compact genetic algorithm. *IEEE Transactions on Evolutionary Computation* 3, 4 (1999), 287–297.

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. In *2015 IEEE International Conference on Computer Vision (ICCV)*. IEEE, 1026–1034.

[7] Jonathan J. Hull. 1994. A Database for Handwritten Text Recognition Research. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 16, 5 (1994), 550–554.

[8] Sergey Ioffe and Christian Szegedy. 2015. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, Vol. PMLR37. 448–456.

[9] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*.

[10] Ron Kohavi. 1996. Scaling Up the Accuracy of Naive-Bayes Classifiers: A Decision-tree Hybrid. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD'96)*. AAAI Press, 202–207.

[11] Ken Lang. 1995. NewsWeeder: Learning to Filter Netnews. In *Machine Learning Proceedings 1995*, Armand Prieditis and Stuart Russell (Eds.). Morgan Kaufmann, San Francisco (CA), 331–339.

[12] Fan Li, Yiming Yang, and Eric P Xing. 2006. From Lasso regression to Feature vector machine. *Advances In Neural Information Processing Systems (NIPS) 18* (2006), 779–786.

[13] Vinod Nair and Geoffrey E Hinton. 2010. Rectified Linear Units Improve Restricted Boltzmann Machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML)*. 807–814.

[14] Yann Ollivier, Ludovic Arnold, Anne Auger, and Nikolaus Hansen. 2017. Information-Geometric Optimization Algorithms: A Unifying Picture via Invariance Principles. *Journal of Machine Learning Research* 18, 18 (2017), 1–65.

[15] Hanchuan Peng, Fuhui Long, and Chris Ding. 2005. Feature selection based on mutual information: Criteria of Max-Dependency, Max-Relevance, and Min-Redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 27, 8 (2005), 1226–1238.

[16] Marko Robnik-Šikonja and Igor Kononenko. 2003. Theoretical and Empirical Analysis of ReliefF and RReliefF. *Machine Learning* 53, 1/2 (2003), 23–69.

[17] Shinichi Shirakawa, Yasushi Iwata, and Youhei Akimoto. 2018. Dynamic Optimization of Neural Network Structures Using Probabilistic Modeling. *Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18)* (2018).

[18] Robert Tibshirani. 1994. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B* 58 (1994), 267–288.

[19] Zhixiang Xu, Gao Huang, Kilian Q Weinberger, and Alice X Zheng. 2014. Gradient boosted feature selection. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '14)*. ACM Press, New York, New York, USA, 522–531.

[20] Bing Xue, Mengjie Zhang, and Will N Browne. 2013. Particle swarm optimization for feature selection in classification: A multi-objective approach. *IEEE Transactions on Cybernetics* 43, 6 (2013), 1656–1671.

[21] Bing Xue, Mengjie Zhang, Will N. Browne, and Xin Yao. 2016. A Survey on Evolutionary Computation Approaches to Feature Selection. *IEEE Transactions on Evolutionary Computation* 20, 4 (2016), 606–626.

[22] Makoto Yamada, Wittawat Jitkrittum, Leonid Sigal, Eric P Xing, and Masashi Sugiyama. 2014. High-Dimensional Feature Selection by Feature-Wise Kernelized Lasso. *Neural Computation* 26, 1 (2014), 185–207.