Adapting Supervised Feature Selection Methods for Clustering Tasks

Eduardo R. Hruschka, Catholic University of Santos, R. Carvalho de Mendonça, 144, Santos SP 11070-906, Brazil; E-mail: erh@unisantos.br
Thiago F. Covões, Catholic University of Santos, R. Carvalho de Mendonça, 144, Santos SP 11070-906, Brazil; E-mail: tcovoes@lsin.unisantos.br
Estevam R. Hruschka, Jr., Federal University of São Carlos, Brazil; E-mail: estevam@dc.ufscar.br
Nelson F. F. Ebecken, Federal University of Rio de Janeiro, Brazil; E-mail: nelson@ntt.ufrj.br

ABSTRACT
In this paper, we elaborate on how feature selection methods traditionally used in classification problems can be adapted for clustering problems, assuming that the number of clusters is not known a priori. Computational complexity of each described algorithm is provided. Empirical results in six bioinformatics datasets illustrate that the adaptation of four well-known supervised methods for feature selection (correlation-based, consistency-based, wrapper of k-NN classifier, and C4.5) can be useful for clustering tasks.

1. INTRODUCTION
Successful data mining applications depend on several factors. The availability of suitable feature selection methods is one of such factors. Feature selection involves choosing a subset of original variables (attributes) by eliminating the redundant, uninformative, and noisy ones. This issue has been broadly investigated in supervised learning tasks for which datasets with many features are available, like in text mining and gene expression data analysis. Under this perspective, there are many potential benefits of feature selection like, for instance [3]: facilitating data visualization and understanding, reducing the measurement and storage requirements, reducing training and utilization times, and defying the curse of dimensionality. Many of these benefits can also be achieved in unsupervised learning (clustering). However, most of the existing supervised methods for feature selection rely on assessing how well some features discriminate among a set of predefined classes. These classes are not available in clustering tasks, in which one seeks to identify a finite set of categories (clusters) to describe a given dataset. In this sense, it is difficult to assess the relevance of a subset of features for describing classes that are not known a priori. Since the optimal number of clusters and the optimal feature subset are inter-related, the feature selection task becomes even more challenging when the number of clusters is unknown [7], which is our assumption in this work.

Although many algorithms for clustering have been proposed in the literature, relatively little work has been done on feature selection for clustering [2][7]. Most clustering methods assume that all features are equally important [2]. However, some features may be more important than others for inducing clusters. In these cases, feature selection methods can be useful. Liu and Yu [9] provide a comprehensive survey of feature selection algorithms for classification and clustering. In brief, there are two fundamentally different approaches for feature selection [8]: wrapper and filter. The former evaluates the subset of selected features using criteria based on the results of clustering algorithms, i.e., the clustering method is wrapped into the feature selection procedure. The latter involves performing feature assessments based on intrinsic properties of the data. These properties are presumed to affect the performance of the clustering algorithm, but they are not a direct measure of its performance, i.e., the feature set is filtered without considering the clustering algorithm that will be ultimately used. In general, filters are less computationally expensive than wrappers, which may be superior in relation to the quality of the clusters found. It is also possible to combine filters and wrappers, obtaining hybrid approaches. Doing so, one can expect to have a reasonable tradeoff between efficiency (computational effort) and efficacy (partition quality). In this sense, we here elaborate on how feature selection methods traditionally used in classification problems can be adapted for clustering tasks.

2. CLUSTERING ALGORITHM
We assume that *clustering* involves the partitioning of a set **X** of instances into a collection of mutually disjoint subsets $\mathbf{C}_i$ of **X**. Formally, let us consider a set of N instances $\mathbf{X}=\{\mathbf{x}_1,\mathbf{x}_2,...,\mathbf{x}_N\}$ to be clustered, where each $\mathbf{x}_i \in \Re^\rho$ is a vector consisting of ρ measurements. The instances must be clustered into non-overlapping groups $\mathbf{C}=\{\mathbf{C}_1,\mathbf{C}_2,...,\mathbf{C}_k\}$ where k is the number of clusters, such that:

$$\mathbf{C}_1 \cup \mathbf{C}_2 \cup ... \cup \mathbf{C}_k = \mathbf{X}\ ,\ \mathbf{C}_i \neq \varnothing, \text{ and } \mathbf{C}_i \cap \mathbf{C}_j = \varnothing \text{ for } i \neq j. \qquad (1)$$

After partitioning the dataset, instances that belong to the same cluster should be more similar to each other than instances that belong to different clusters. Therefore, it is necessary to devise means of evaluating similarities between instances. This problem is often tackled indirectly, i.e. distance measures are used to quantify dissimilarities between instances. Several dissimilarity measures can be used for clustering tasks. We here use the Euclidean distance.

The simplified silhouette [5] is used for estimating the number of clusters. Before describing the *simplified silhouette*, let us introduce the silhouette proposed in [6]. Consider an instance i belonging to cluster **A**. So, the average dissimilarity of i to all other instances of **A** is denoted by $a(i)$. Now let us take into account cluster **C**. The average dissimilarity of i to all instances of **C** will be called $d(i,\mathbf{C})$. After computing $d(i,\mathbf{C})$ for all clusters $\mathbf{C} \neq \mathbf{A}$, the smallest one is selected, i.e. $b(i) = \min d(i,\mathbf{C}),\ \mathbf{C} \neq \mathbf{A}$. This value represents the dissimilarity of i to its neighbor cluster, and the silhouette $s(i)$ is:

$$s(i) = \frac{b(i) - a(i)}{max\{a(i)\ b(i)\}} \qquad (2)$$

The higher $s(i)$ the better the assignment of instance i to a given cluster. In addition, if $s(i)$ is equal to zero, then it is not clear whether i should have been assigned to its current cluster or to a neighboring one. Finally, if cluster **A** is a singleton, then $s(i)$ is not defined and the most neutral choice is to set $s(i) = 0$ [6]. The average of $s(i)$ over $i = 1,2,...,N$ can be used as a criterion to assess the quality of a given partition. Doing so, the best data partition is achieved when the silhouette is maximized.

The original silhouette [6] depends on the computation of all distances between instances, leading to a computational cost of $O(N^2)$, which is often not sufficiently efficient for real-world applications. To circumvent this limitation, a *simplified silhouette* can be used. The simplified silhouette (SS) [5] is based on the computation of distances between instances and cluster centroids. More specifically, the term $a(i)$ of Eqn. (2) becomes the dissimilarity of instance i to its corresponding cluster (**A**) centroid. Similarly, instead of computing $d(i,\mathbf{C})$ as the average dissimilarity of i to all instances of **C**, $\mathbf{C} \neq \mathbf{A}$, distances between i and the centroid of **C** are computed. While these modifications reduce the computational cost from $O(N^2)$ to $O(N)$, empirical results [5] suggest that the partition quality may be not significantly affected.

The computation of the original silhouette [6], as well as of its *simplified* version [5], only depends on the partitions found. Therefore, such silhouettes can be applied to assess partitions found by several clustering algorithms. We adopt the *k*-means algorithm to obtain partitions to be evaluated by the simplified silhouette (SS). Roughly speaking, *k*-means is designed to minimize the sum of distances between instances and nearest centroids. From the SS criterion viewpoint, good partitions are also obtained when this minimization is suitably performed, as well as when the clusters are well separated. Thus, although other clustering algorithms could be used, our approach favors a synergy between *k*-means and SS. In particular, we perform multiple runs of *k*-means (for different values of *k*) and then choose the best obtained partition according to the SS value. It is also known that *k*-means may get stuck at suboptimal solutions for a given *k*. To alleviate this limitation, one can perform multiple runs of *k*-means for a fixed *k*. Fig. 1 summarizes the sampling strategy here used to find data partitions. The *correct* number of clusters *k** is automatically estimated by means of the simplified silhouette.

Provided that the computational cost of *k*-means is $O(t \cdot k \cdot \rho \cdot N)$, where *t* is the number of iterations, the overall computational cost of the sampling strategy for *k*-means (Fig. 1) is estimated as $O((k_{max} - k_{min} + 1) \cdot np \cdot (t \cdot k \cdot \rho \cdot N))$. In what concerns the minimum and maximum number of clusters (k_{min} and k_{max}, respectively) let us assume a scenario in which domain knowledge is not available. In such a scenario, searching for a solution in a suitable subset of the search space in terms of *k* is desirable. To that end, a rule of thumb [11] involves choosing values for *k* from the set $\{2,...,N^{1/2}\}$. Then, the resultant overall computational cost of the used sampling strategy for *k*-means is estimated as $O(\rho \cdot N^2)$ - assuming that the number of assessed partitions, *np*, and the number of *k*-means iterations, *t*, are significantly less than *N*. From this point of view, if domain knowledge regarding *k** (or a range for its probable values) is available and $k^* << N$, then the resultant overall computational cost of the used sampling strategy for *k*-means is estimated as $O(\rho \cdot N)$.

So far, we have described a procedure to find data partitions (the *correct* number of clusters *k** is estimated according to the SS criterion), assuming that a set of ρ features is provided. In the next section, we discuss how feature selection methods traditionally used in classification problems can be adapted for clustering problems.

3. METHODS FOR FEATURE SELECTION

Most of the commonly used supervised methods for feature selection rely on assessing how well some features discriminate among a set of predefined classes, which are not known in clustering problems. Therefore, supervised feature selection methods cannot be directly applied in these problems. However, the difficulty originated from the lack of information concerning the classes can be circumvented by assuming that a set of clusters can be modeled as being a set of different classes. Doing so, supervised methods can be adapted for selecting features in clustering tasks. In our work, we hypothesized that supervised methods can be used to determine the relevant features that model a set of clusters obtained by *k*-means. In this sense, a simple alternative involves running *k*-means for all available features and then select the relevant ones according to their importance to the found clusters. Although this approach is reasonable, the optimal number of clusters and the optimal feature subset are often inter-related [7]. Since we assume that the number of clusters is not known a priori, i.e., it is estimated by the silhouette-based clustering method (Fig. 1), we believe that an exploratory approach is better suited for our purposes.

An exhaustive search for all the possible feature subsets is often computationally intractable – the order of the search space is $O(2^{\rho})$. Thus, sequential search algorithms (e.g. forward, backward, and bidirectional) are widely used. Forward methods tend to be particularly problematic when supervised methods are adapted for clustering problems, mainly because it is difficult for them to select feature subsets that are good *copredictors* of the clusters if none of these *copredictors* is a good *predictor* of the clusters by itself [1]. In these cases, backward selection can succeed because it works by eliminating features rather than successively adding them. However, backward methods are often less efficient than forward methods. Although the approach here used does not strictly conform to the commonly used categorization for sequential selection, it can be viewed as a backward method, because it removes features from an initially complete set, as presented in the algorithm illustrated in Fig. 2. This algorithm was designed for exploring the interactions between clustering results (i.e., number of clusters and corresponding partitions) and selected features. As discussed in Section 2, the computational complexity of step 2 is estimated as $O(\rho \cdot N^2)$ when $k_{min}=2$ and $k_{max}=N^{1/2}$ – assuming that domain knowledge is not available to set *k*. Let us call *I* the number of iterations of the algorithm depicted in Fig. 2. Thus, with a little notation abuse, the overall computational cost of this algorithm is $O(I \cdot [\rho \cdot N^2 + M])$, where *M* is the computational cost of each particular feature selection used in step 3. In the following, we briefly describe four well-known supervised feature selection methods that can be used in this step.

3.1 Correlation Feature Selection (CFS)

Correlation-based feature selection (CFS) [4] evaluates subsets of features by means of a heuristic that considers the usefulness of individual features for predicting the class (in our case cluster labels represent the classes) along with the level of inter-correlation among them. Let $\bar{r}_{cf}$ be the average feature-class correlation and $\bar{r}_{ff}$ be average feature-feature inter-correlation. The $Merit_s$ of a feature subset S containing ρ_s features is given by:

$$Merit_s = \frac{\rho_s \cdot \bar{r}_{cf}}{\sqrt{\rho_s + \rho_s \cdot (\rho_s - 1) \cdot \bar{r}_{ff}}} \qquad (3)$$

The numerator estimates the prediction capability of the features in S in relation to the cluster, whereas the denominator indicates the redundancy level among them. The correlation between two discrete random variables (features) X and Y is computed according to the Symmetrical Uncertainty (SU):

$$SU = 2 \cdot \frac{H(Y) - H(Y|X)}{H(Y) + H(X)} \qquad (4)$$

Figure 1. Sampling strategy for k-means

```
1. Choose $k_{min}$, $k_{max}$, and np.
2. SSV ← -1;   / SSV = Simplified Silhouette Value /
3. For each $k \in \{k_{min},...,k_{max}\}$ do:
    3.1 Generate np random initial partitions of instances into k nonempty clusters;
    3.2 Run k-means for each initial partition generated in step 3.1, and compute its corresponding simplified silhouette. Let the best obtained value be BOV;
    3.3 If (BOV>SSV) then {
            SSV←BOV;
            $k^* \leftarrow k$;
            Hold the corresponding partition for k*.
        }
4. Return SSV and its corresponding data partition for k*.
```

Figure 2. Adapting supervised feature selection for k-means

```
Let the complete feature set be C, and the feature subsets obtained in two consecutive iterations be S' and S''. The algorithm for feature selection can be summarized as:
1. S' ← C;  / initially all features are used to get partitions /
2. Run the adopted sampling strategy for k-means considering the feature subset S';
3. S'' ← {feature subset achieved from supervised selection, for which the clusters of the current partition are considered classes};
5. If (S''=S') then:
    5.1 Hold the current partition and the respective feature subset;
    5.2 Stop.
   Else (S' ← S'' & go to Step 2);
```

where *H(Y)* and *H(Y/X)* are given by equations (5) and (6) respectively:

$$H(Y) = -\sum_i p(y_i) \log_2 p(y_i) \qquad (5)$$

$$H(Y|X) = -\sum_j p(x_j) \sum_i p(y_i|x_j) \cdot \log_2 p(y_i|x_j) \qquad (6)$$

where $p(x_j)$ is the probability for values of *X*. Numeric features are discretized before computing correlations, and a forward selection technique is used to search for the feature subset. The complexity of CFS can be estimated as $M=O(\rho^2 \cdot N)$, then the total computational cost of the algorithm depicted in Fig. 2 is estimated as $O(I \cdot [\rho \cdot N^2 + \rho^2 \cdot N])$ when CFS is used in step 3.

3.2 Consistency-based Evaluation (CBE)

Liu and Setiono [10] propose that two instances are inconsistent if they match except for their class labels. Assuming that *NC* is the number of distinct combinations of feature values for a given subset of features *S*, the consistency of this subset can be measured by:

$$Consistency_S = 1 - \frac{\sum_{i=1}^{NC} (|D_i| - |M_i|)}{N} \qquad (7)$$

where $|D_i|$ and $|M_i|$ are the number of occurrences of the *i*th feature value combination and the cardinality of the majority class, respectively. *N* is the number of dataset instances. Numeric features are discretized before computing the consistency of a given set of features *S*, and a forward selection technique is used to search for the subset of features. The overall computational cost of the algorithm depicted in Fig. 2 is estimated as $O(I \cdot [\rho \cdot N^2 + \rho^2 \cdot N])$ when CBE is used in step 3.

3.3 Wrapper of k-NN Classifier (W-KNN)

This method uses the *k*-Nearest Neighbor (*k*-NN) classifier [1] as the target learning algorithm to assess the quality of subsets of attributes by means of a forward selection based search. Each feature subset is evaluated according to the *k*-NN accuracy achieved in cross-validation procedure. The *k*-NN classifier was chosen because it is a distance-based method. Since the clustering method used in this work is also based on the computation of distances between instances, a synergy between feature selection and clustering can be favored. The incorporation of W-KNN in the algorithm summarized Fig. 2 leads to a computational cost estimated as $O(I \cdot \rho^3 \cdot N^2)$.

3.4 C4.5 Decision Tree

The *C4.5* [12] classifier performs feature selection as part of a decision tree building process, in which a subset of features is selected according to an information gain criterion. We assume that the features selected by *C4.5* can also be interesting to model the clusters of a given partition. More precisely, features selected by *C4.5* will form *S´´* in step 3 of the algorithm in Fig. 2. The overall computational cost of the resultant algorithm is estimated as $O(I \cdot \rho \cdot N^2)$.

3.5 Computational Complexity Summary

It is expected that the number of iterations (*I*) of the algorithm depicted in Fig. 2 is significantly less than both ρ and *N*. Indeed, the experimental results to be reported in the next section somehow support this claim. Considering that this assumption is reasonable, Table 1 summarizes the overall computational costs of the feature selection methods here used (2nd column). In the last column we provide corresponding estimates for applications in which domain knowledge is available to set the number of clusters (see discussion in Section 2 for further details).

Table 1. Time complexity summary

Method	k_{min}=2 and k_{max}=$N^{1/2}$	Domain knowledge
CFS	$O(\rho \cdot N^2 + \rho^2 \cdot N)$	$O(\rho^2 \cdot N)$
CBE	$O(\rho \cdot N^2 + \rho^2 \cdot N)$	$O(\rho^2 \cdot N)$
W-KNN	$O(\rho^3 \cdot N^2)$	$O(\rho^3 \cdot N^2)$
C4.5	$O(\rho \cdot N^2)$	$O(\rho \cdot N \cdot log_2 N)$

4. EMPIRICAL EVALUATION

The assessment of clustering accuracy often requires datasets for which the clusters are a priori known. We have performed experiments in six bioinformatics datasets [13]. Five datasets, here called Bio1, Bio2, Bio3, Bio4, and Bio5, are composed of 400 genes (instances) described by 20 measurements (features). These are formed by synthetic data with error distributions derived from real-world data, and contain six approximately equal-sized clusters. In addition, we have used a real-world dataset (*yeast galactose*) that is composed of 20 measurements and 205 genes. In this dataset, the expression patterns reflect four functional categories (clusters). The datasets used in the experiments reported here take into account four repeated measurements. From now on, the correct clusters will be called classes, whereas the term cluster will refer to each group of similar instances found by a clustering algorithm. However, although the class corresponding to each gene is known a priori, this information was not used in the clustering process. Thus, the quality of the partitions obtained can be assessed by verifying the degree for which the obtained clusters match the classes. Since the assessed approaches have systematically found a number of clusters that is approximately equal to the correct ones, a simple and intuitive way of evaluating the accuracy of the obtained partitions involves applying some measure of classification quality like the class error (CE), which is the percentage of the instances misclassified in relation to the total number of instances.

In all experiments, we have set k_{min}=2, k_{max}=$N^{1/2}$, and *np*=20, following the elaboration described in Section 3. Tables 2-7 summarize the average results obtained in five runs of each method. In such tables, *I*, ρ^*, *k**, *CE*, and *CT* stand for number of iterations, number of selected features, estimated number of clusters, class error and computing times (in a Pentium IV, 3 GHz CPU, 1 GB RAM), respectively. Small variances have been observed for all the assessed aspects. Accordingly, only subtle differences were observed in relation to the found partitions. Finally, the last line of each table refers to the results obtained from the sampling strategy for *k*-means without feature selection.

For most of the assessed methods, the subsets of selected features are significantly smaller than the complete feature sets. In brief, good data partitions were obtained in most of the performed experiments, suggesting that the assessed methods can be useful for clustering gene expression data. CFS presented considerably worse results in most of the performed experiments. CBE, W-KNN, and C4.5, by their turn, have presented similar results, though significantly different results have been obtained for particular datasets. For instance, C4.5 has shown a better performance than both CBE and W-KNN in terms of *CE* in Bio2, although selecting more features. On the other hand, both CBE and W-KNN have provided

Table 2. Dataset Bio1

Method	*I*	ρ^*	*k**	CE (%)	CT(s)
CFS	1	17	6	0.00	3.5
CBE	1	2	6	0.50	2.7
W-KNN	1	2	6	0.00	27.7
C4.5	1	2	6	0.00	2.7
All features	-	20	6	0.00	2.0

Table 3. Dataset Bio2

Method	*I*	ρ^*	*k**	CE(%)	CT(s)
CFS	1	19	5	16.75	3.1
CBE	1	2	4	33.75	2.1
W-KNN	1	2	4	33.25	25.0
C4.5	1	4	5	16.75	2.1
All features	-	20	5	16.75	1.4

Table 4. Dataset Bio3

Method	*I*	ρ*	*k**	CE(%)	CT(s)
CFS	1	18	6	0.00	2.8
CBE	1	2	6	0.25	2.0
W-KNN	1	2	6	0.00	28.8
C4.5	1	4	6	0.00	2.1
All features	-	20	6	0.00	1.4

Table 5. Dataset Bio4

Method	*I*	ρ*	*k**	CE(%)	CT(s)
CFS	1	11	4	33.50	2.6
CBE	1	2	4	33.50	2.1
W-KNN	1	2	4	33.50	25.6
C4.5	1	4	4	33.50	2.0
All features	-	20	4	33.50	1.5

Table 6. Dataset Bio5

Method	*I*	ρ*	*k**	CE(%)	CT(s)
CFS	1	1	6	0.25	1.8
CBE	1	1	6	0.25	1.9
W-KNN	1	2	6	0.00	39.1
C4.5	1	2	5	17.50	1.9
All features	-	20	4	33.50	1.4

Table 7. Dataset yeast

Method	*I*	ρ*	*k**	CE(%)	CT(s)
CFS	1	1	3	7.80	1.2
CBE	1	1	3	7.80	1.1
W-KNN	2	1	2	47.80	13.0
C4.5	1	2	3	7.32	1.2
All features	-	20	3	7.31	0.8

better partitions than C4.5 in Bio5. Finally, W-KNN has not performed well in *yeast*. In brief, CBE has shown a good tradeoff between clustering quality and computational effort required to achieve it.

To conclude, some results deserve further attention. The features selected in Bio5 allowed finding better partitions than those obtained by all features, illustrating how the removal of redundant and/or irrelevant features may even promote the improvement of the clustering process. Second, one or two features were enough to provide good partitions for *yeast*.

5. CONCLUSIONS

We have described how feature selection methods traditionally used in classification problems can be adapted for clustering problems. Analyses in terms of time complexity have been undertaken for all the studied methods. Also, empirical results in six bioinformatics datasets illustrated the performance of the assessed methods, which in general have provided good data partitions, while reducing the number of features. The results obtained by the consistency-based evaluation [10] suggest that it is promising for applications in which computational efficiency is a central issue.

6. ACKNOWLEDGMENTS

We are grateful to the Brazilian Research Agencies CNPq, FAPESP, and FAPERJ for their financial support.

7. REFERENCES

[1] Cover, T. and Hart, P., Nearest neighbor pattern classification, IEEE Transactions on Information Theory v.13, pp 21–27, 1967.

[2] Dash, M., Liu, H., Feature Selection for Clustering, Proc. of the Pacific-Asia Conf. on Knowledge Discovery and Data Mining, LNCS 1805, pp. 110-121, 2000.

[3] Guyon, I., Elisseeff, A., An Introduction to Variable and Feature Selection, Journal of Machine Learning Research 3, pp. 1157-1182, 2003.

[4] Hall, M. A., Holmes, G., Benchmarking Attribute Selection Techniques for Discrete Class Data Mining, IEEE Transactions on Knowledge and Data Engineering, v. 15, n.3, 2003.

[5] Hruschka, E. R., de Castro, L. N., Campello, R. J. G. B., Evolutionary Algorithms for Clustering Gene-Expression Data, Proc. of the 4th IEEE Int. Conf. on Data Mining 2004, Brighton, UK, pp. 403-406, 2004.

[6] Kaufman, L., Rousseeuw, P. J., Finding Groups in Data – An Introduction to Cluster Analysis, Wiley Series in Probability and Mathematical Statistics, 1990.

[7] Law, M.H.C., Figueiredo, M.A.T., Jain, A.K., Simultaneous Feature Selection and Clustering Using Mixture Models, IEEE Transactions on Pattern Analysis and Machine Intelligence, 26(9), pp. 1154-1166, 2004.

[8] Liu, H. and Motoda, H., Feature Selection for Knowledge Discovery and Data Mining. Kluwer Academic, 1998.

[9] Liu, H., Yu, L., Toward Integrating Feature Selection Algorithms for Classification and Clustering, IEEE Transactions on Knowledge and Data Engineering, 17(3), 1-12, 2005.

[10]Liu, H., Setiono, R., A probabilistic approach to feature selection: A filter solution", Proc. of the 13th Int. Conf. on Machine Learning, pp. 319-327, Morgan Kaufmann, 1996.

[11]Pal, N.R., Bezdek, J. C., On Cluster Validity for the Fuzzy c-Means Model, IEEE Transactions on Fuzzy Systems, v. 3, n. 3, 370-379, 1995.

[12]Quinlan, R., C4.5: Programs for Machine Learning, Morgan Kaufmman, 1993.

[13]Yeung, K.Y., Medvedovic, M., Bumgarner, R.E., Clustering gene-expression data with repeated measurements, Genome Biology, 4(5), article R34, 2003.