Relevant Subsequence Detection with Sparse Dictionary Learning

Sam Blasiak[1], Huzefa Rangwala[1], and Kathryn B. Laskey[1]

George Mason University, Fairfax, VA 22030, USA
sblasiak@masonlive.gmu.edu, rangwala@cs.gmu.edu, klaskey@gmu.edu

Abstract. Sparse Dictionary Learning has recently become popular for discovering latent components that can be used to reconstruct elements in a dataset. Analysis of sequence data could also benefit from this type of decomposition, but sequence datasets are not natively accepted by the Sparse Dictionary Learning model. A strategy for making sequence data more manageable is to extract all subsequences of a fixed length from the original sequence dataset. This subsequence representation can then be input to a Sparse Dictionary Learner. This strategy can be problematic because self-similar patterns within sequences are over-represented. In this work, we propose an alternative for applying Sparse Dictionary Learning to sequence datasets. We call this alternative Relevant Subsequence Dictionary Learning (RS-DL). Our method involves constructing separate dictionaries for each sequence in a dataset from shared sets of relevant subsequence patterns. Through experiments, we show that decompositions of sequence data induced by our RS-DL model can be effective both for discovering repeated patterns meaningful to humans and for extracting features useful for sequence classification.

1 Introduction

Sparse Dictionary Learning has recently become popular for discovering latent components that can be used to reconstruct elements in a dataset. It has seen particular success in computer vision where it has been incorporated into solutions for problems in image reconstruction, in-painting, and classification [1–6].

Sparse Dictionary Learning's success in computer vision makes it a promising candidate as an algorithm for discovering patterns in sequence data. Sequence data, however, is not natively accepted by the Sparse Dictionary Learning model: sequences can be of variable length and patterns within sequences are not associated with a fixed set of indices. These patterns can occur at any point within a sequence and can be repeated multiple times. A strategy for making sequence data more manageable is to extract all subsequences of length K from the original dataset and use these as input to a Sparse Dictionary Learning algorithm. This strategy poses a problem, however, because self-similar patterns within sequences are over-represented.

In this work, we propose an alternative to this standard subsequence dataset approach, which we call Relevant Subsequence Dictionary Learning (RS-DL).

Our method involves constructing separate dictionaries for each sequence in a dataset from shared "relevant subsequence patterns." This structured dictionary can be used to pick out shared information from sets of sequences and can be learned using standard optimization methods. An important contribution of our work is in showing how to efficiently run the LARS algorithm given our relevant subsequence dictionary structure.

To show the utility of the RS-DL model, we run experiments on several types of sequence data. Running our algorithm on synthetic sets of sequences with discrete-valued elements, continuous electrocardiogram data, and text datasets, we show that our RS-DL model is effective for discovering repeated patterns meaningful to humans (also called motifs). We also show that RS-DL is effective for classification. To do so, we use RS-DL to extract features from time-series data and show that these features can reduce classification error compared to standard methods.

2 Background: Sparse Dictionary Learning

Sparse Dictionary Learning is a method of decomposing a dataset into the product of a dictionary matrix and a sparse vector of coefficients. Here we represent the N dataset vectors as $x_{1:N}$, with the n^{th} vector given by $x_n \in \mathbb{R}^d$, the dictionary matrix as $W \in \mathbb{R}^{d \times C}$, and the set of N sparse vectors of coefficients as $\alpha_{1:N}$, $\alpha_n \in \mathbb{R}^C$. The number of dictionary columns, C, is chosen beforehand. The Sparse Dictionary Learning objective is typically defined as follows:

$$f(x_n; W) = \min_{\alpha_n} \underbrace{\frac{1}{2}||x_n - W\alpha_n||_2^2}_{\text{loss}} + \underbrace{\lambda \psi(\alpha_n)}_{\text{sparsity-inducing term}} \quad (1)$$

where ψ is a regularization function, typically an L_1 norm.

There has been a significant amount of research to develop efficient algorithms for solving the Sparse Dictionary Learning problem [3]. These algorithms typically consist of repeating two optimization steps. In the first step, a linear regression problem with the sparsity-inducing regularization term is solved to compute $\alpha_n = \min_{\alpha_n} ||x_n - W\alpha_n||_2^2 + \lambda\psi(\alpha_n)$ given the current value of the dictionary, W, for each example in the dataset. Common algorithms to perform this task include pursuit algorithms [7], Least Angle Regression (LARS) [8], coordinate-wise descent methods [9], and proximal methods [10].

In the second step, the value of the dictionary, W, is updated given the current minimum values of α_n. As with methods for optimizing with respect to the α's, it is possible to use any of a number of different methods to minimize with respect to the dictionary. These methods include K-SVD [7] (which also updates the α terms), stochastic gradient methods [6], and solutions of the dual problem (for a constrained dictionary) [5], among others.

Sparse Dictionary Learning is similar to other decomposition techniques like Principal Component Analysis (PCA). PCA decomposes elements of a dataset into linear combinations of vectors from an orthogonal basis. Sparse Dictionary

Learning differs from PCA in two important respects. First, dictionary columns are non-orthogonal, and second, the sparsity inducing regularization term forces only a small number of columns to be used for reconstruction. These characteristics can be advantageous compared to PCA because the sparsity inducing term allows the dictionary to include more columns that the dimensionality of the vector being reconstructed [3]. This "overcomplete" representation allows a large number of patterns to be found in the data but only a small number of these patterns are used to reconstruct each data element.

3 Relevant Subsequence Dictionary Learning

We propose an approach, which we call Relevant Subsequence Dictionary Learning (RS-DL)[1], to extend Sparse Dictionary Learning to the domain of sequences. Sequences differ from more-standard vector representations in that they can vary in length across a single dataset, and patterns within sequences can occur at any position rather than being associated with a fixed set of indices. To account for these characteristics of sequence data, RS-DL constructs dictionaries from C different subsequence dictionary components, each of length K. We refer to these constituent components as "relevant subsequence patterns" and indicate these patterns by the two-dimensional array, $\mathbf{v}$, of size $C \times K$, where $v_{c,k}$ is a value associated with the k^{th} position in the c^{th} relevant subsequence pattern.

Unlike standard Sparse Dictionary Learning, RS-DL constructs a separate dictionary, W_n, for each sequence, x_n, in a dataset by positioning relevant subsequence parameters, $v_{c,:}$, so that they cover all possible subsequence starting positions. Positions in dictionary columns that are not given by relevant subsequence parameters are set to zero.

Figure 1 shows how the array of constituent relevant subsequence patterns, $\mathbf{v}$, is used to construct W_n, the dictionary associated with sequence x_n. Table 1 gives descriptions of all parameters in the RS-DL formulation. After building the dictionaries, W_n, we are left with an objective very similar to that of standard Sparse Dictionary Learning:

$$f(x_{1:N}; \mathbf{v}) = \sum_{n=1}^{N} \min_{\alpha_n} \frac{1}{2} ||x_n - W_n \alpha_n||_2^2 + \lambda |\alpha_n|_1 \tag{2}$$

To optimize with respect to this objective, we employ a stochastic gradient descent procedure where sequences are received by the learner in random order. The learner alternatively solves first for α_n, then takes a gradient step with respect to the array, $\mathbf{v}$, in a similar manner to existing Sparse Dictionary Learning optimization algorithms. For the optimization step with respect to α_n, we apply a variation of the Least Angle Regression (LARS) [8] algorithm. The LARS algorithm requires computing a number of matrix products involving W_n. However, computing these matrix products directly would be inefficient, as each W_n matrix is of size $O(T_n) \times O(CT_n)$, where T_n is the length of the n^{th} sequence.

[1] We have made code available at http://cs.gmu.edu/∼sblasiak/RS-DL.tar.gz

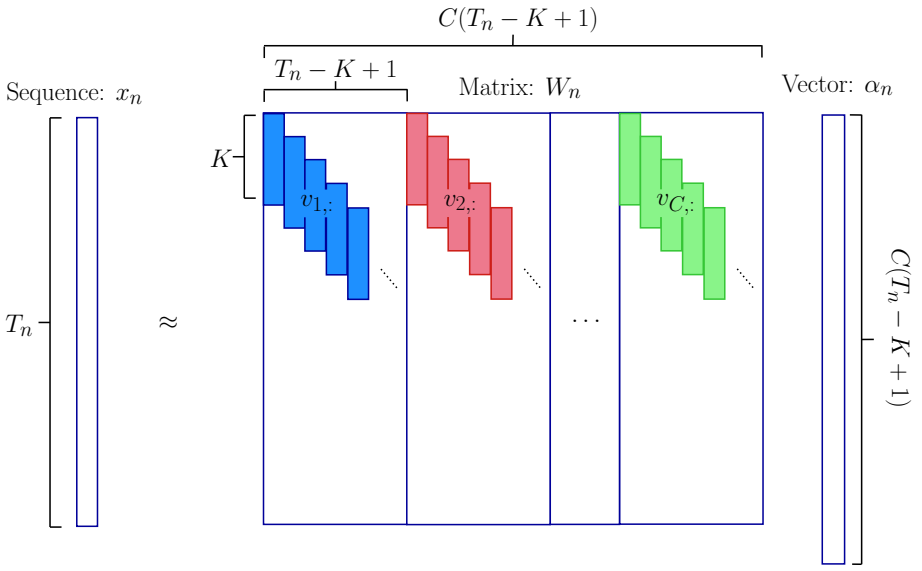


Fig. 1. The figure above illustrates the Relevant Subsequence Dictionary Learning setup. The matrix W_n is constructed from the weights $v_{c,k}$ in C blocks so that the relevant subsequence patterns given by each $v_{c,:}$ are arranged to create dictionary elements (columns of W_n) that cover every K length subsequence of the sequence x_n (illustrated in blue). White areas of the W_n matrix are set to zero. The vector α_n is L_1-regularized to select a small number of dictionary columns associated with positioned relevant subsequences patterns. The α_n-weighted sum of these positioned relevant subsequences patterns approximates x_n.

To improve performance, we can take advantage of the sparse construction of each W_n, allowing these products to be computed more quickly, as we describe in the next section.

After computing each new value of α_n, the RS-DL algorithm takes a single stochastic gradient step in $\mathbf{v}$: $\mathbf{v}^{i+1} \leftarrow \mathbf{v}^i - \left(\frac{\gamma}{i+1}\right) \frac{\partial \frac{1}{2}||x_n - W_n^i \alpha_n||_2^2}{\partial \mathbf{v}}$, where γ is a learning rate term. We found empirically that, for RS-DL, this single stochastic gradient step is often faster than solving for $\mathbf{v}$ after accumulating information from a batch of α_n's as in Mairal et. al. [3].

3.1 Efficiently Running the LARS Algorithm with RS-DL

RS-DL involves constructing dictionaries, W_n of size $O(T_n) \times O(CT_n)$, many of whose entries are set to zero. If not carefully handled, this large, sparse matrix can cause the RS-DL training algorithm to operate inefficiently. The LARS algorithm constitutes a major substep in RS-DL training and requires a number of computations involving W_n. Efficiency of these computations can be considerably improved by taking W_n's sparse construction from elements of the array $\mathbf{v}$ into account. Three LARS computations involving the dictionaries, W_n are (i) the matrix-matrix product $(W_n)_{\mathcal{A}}^{\top} (W_n)_{\mathcal{A}}$, (ii) the matrix-vector product

Table 1. Relevant Subsequence Dictionary Learning parameters

Parameter	Definition
M	The size of the alphabet for discrete sequences. We omit the M parameter when dealing with continuous-valued sequences.
$\mathbf{x}_{1:N}$	A set of N observed sequences. Individual sequences, x_n, can be of variable length. Discrete sequences are expanded to M concatenated sequences of T_n indicator variables.
T_n	The length of the n^{th} sequence.
$\alpha_{1:N}$	A set of N vectors. Each α_n vector is of length $C\left(T_n - K + 1\right)$.
W_n	A weight matrix of size $T_n \times C\left(T_n - K + 1\right)$ created from elements of the array $\mathbf{v}$.
$\mathbf{v}$	An array of values used to construct dictionary elements. $v_{c,k}$ is associated with the k^{th} position in the c^{th} relevant subsequence pattern.
λ	The L_1 regularization parameter associated with each α_n.

$(W_n)_{\mathcal{A}}\,\omega_{\mathcal{A}}$, and (iii) the matrix-vector product $W_n^\top u$, where $\mathcal{A}$ indicates an active set of columns (the number of non-zero components of α), $(W_n)_{\mathcal{A}}$ indicates a matrix constructed from this active set, $\omega_{\mathcal{A}}$ is a vector of length $|\mathcal{A}|$, and u is a vector of length T_n. Below, $t(i)$ indicates the index of the start of the subsequence associated with the i^{th} column of W_n (see Figure 1), $c(i)$ indicates the relevant subsequence position associated with the i^{th} column of W_n, and $\mathrm{sgn}(i)$ indicates the sign of the correlation between the i^{th} matrix column, $(W_n)_i^\top$, and the current residual: $\mathrm{sgn}\left((W_n)_i^\top \left(x_n - (W_n)_{\mathcal{A}}^\top \alpha_n\right)\right)$.

The matrix-matrix product, $X = (W_n)_{\mathcal{A}}^\top (W_n)_{\mathcal{A}}$, can be computed as follows:

$$X_{ij} = \begin{cases} \sum_{k=0}^{\max(K-t(j)+t(i),0)} \mathrm{sgn}\left(i\right) v_{c(i),k} \mathrm{sgn}\left(j\right) v_{c(j),t(j)-t(i)+k} & t(j) \geq t(i) \\ \sum_{k=0}^{\max(K-t(i)+t(j),0)} \mathrm{sgn}\left(i\right) v_{c(i),t(i)-t(j)+k} \mathrm{sgn}\left(j\right) v_{c(j),k} & t(j) < t(i) \end{cases} \quad (3)$$

This matrix-matrix product has an overall complexity of $O(|\mathcal{A}|^2 K)$. However, the full product does not need to be computed at each LARS iteration. Rather, as additional columns are added to the active set, we update a stored Cholesky decomposition of $(W_n)_{\mathcal{A}}^\top (W_n)_{\mathcal{A}}$, at a cost of $O(|\mathcal{A}|K)$ for each update (updates involve computing a single column of the product in Equation 3), plus $O(|\mathcal{A}|^2)$ for a back-substitution operation.

We compute the matrix-vector product, $\mathbf{x} = (W_n)_{\mathcal{A}}\,\omega_{\mathcal{A}}$, incrementally as the weighted sum of components of $\mathbf{v}$:

$$\mathbf{x}^{(1:i)} = \sum_{k=1}^{K} \mathbf{x}_{t(i)+k}^{(1:i-1)} + \mathrm{sgn}\left(i\right) \left(\omega_{\mathcal{A}}\right)_i v_{c(i),k} \quad (4)$$

where $\mathbf{x}^{(1:i)}$ indicates the sum up to the i^{th} term, and $i \in [1 \ldots |\mathcal{A}|]$. This matrix-vector product has an overall complexity of $O(|\mathcal{A}|K)$.

Finally, we compute the matrix-vector product, $\mathbf{x} = W_n^\top u$, as follows:

$$\mathbf{x}_i = \sum_{k=1}^{K} v_{c(i),k} u_{t(i)+k} \quad (5)$$

with an overall complexity of $O(CT_nK)$.

For each LARS iteration, we must also compute CT_n correlations between each column of the matrix, W_n, and the current residual at a cost of K each.

These are computed in the same way as the matrix-vector product in Equation 5. In most of out experiments, we restrict $|\mathcal{A}|$ to values less than or equal to C. Thus, each LARS iteration has a complexity of $O(CT_nK)$ when $|\mathcal{A}|$ is small, which can be a significant reduction from $O(CT_n^2)$. However, with no restrictions on the size of the active set, $|\mathcal{A}|$ can potentially grow to CT_n. In this case, complexity is eventually dominated by back-substitution operations involving the incrementally-updated Cholesky decomposition of $(W_n)_{\mathcal{A}}^{\top}(W_n)_{\mathcal{A}}$ at a cost of up to $O(C^2T_n^2)$ per iteration.

3.2 Modifications to RS-DL

The procedure for constructing the RS-DL dictionary (Figure 1) is applicable only to sequences with continuous-valued elements. To allow RS-DL to find decompositions of sequences of discrete symbols, we first transform each original sequence into M separate binary sequences, where elements of the m^{th} binary sequence indicate if the symbols in the original sequence are equal to the m^{th} symbol in the alphabet. These M binary sequences are then concatenated to obtain the input sequence to RS-DL. Dictionary construction must also be modified for discrete sequences. In this case, $\mathbf{v}$ becomes a three-dimensional constituent array, where $v_{c,k,m}$ is associated with the m^{th} symbol of the k^{th} position in the c^{th} relevant subsequence. Separate dictionaries are constructed for each of the M possible symbols using $v_{c,:,m}$ for the c^{th} relevant subsequence pattern associated with the m^{th} constructed binary sequence. These M dictionaries are then stacked vertically to create a composite dictionary.

It is also possible to use RS-DL to find decompositions of multi-variate sequences. To do so, we rearrange each multivariate sequence as a concatenation of univariate sequences. We then create a stack of M dictionaries as we did to create the dictionary for discrete sequences.

Another modification of the basic RS-DL algorithm includes appending a column to the dictionary whose entries are set to a constant value. This addition has the effect of including a bias term whose magnitude varies depending on the associated α_n term. This bias term is useful for modeling time series datasets where the amplitudes of major trends that occur in individual sequences are offset by varying amounts. We employ this bias term in all experiments conducted on time series sequences. A similar strategy can also be employed to capture linear trends.

Finally, we can modify the LARS algorithm so that, rather than finding an L_1-regularized solution for α, it finds solutions with one or fewer non-zero α terms associated with each of the C relevant subsequence patterns. Although the L_1 regularization is no longer enforced in this case, sparsity is maintained in a similar manner to the L_0-regularized[2] version of LARS[8].

[2] The "L_0 norm" [7] is a pseudo-norm that counts the number of non-zero components in a vector, i.e., $||\mathbf{x}||_0 = \sum_i \mathbb{I}(x_i \neq 0)$.

4 Relationship to Hidden Markov Models

A form of the Factorial Hidden Markov Model, which we describe later in this section, shares characteristics of RS-DL. To understand Factorial Hidden Markov Models, one must first understand the basic Hidden Markov Model (HMM), which defines a probability distribution over sequences. The HMM assumes that each symbol in a sequence is generated from a mixture distribution. Mixture components are indexed by "hidden states" in the HMM. The Markov property holds over these hidden states, meaning that the value of the hidden state indexing the observation at time point t depends only on the value of the hidden state associated with time point $t-1$.

The Profile HMM (pHMM) [11] is an HMM with specific restrictions on transitions and emissions. In Profile HMMs, hidden states are divided into three types: *Match* states, which describe important sequence elements, *Insert* states, which model noise, and *Delete* states, which do not emit a symbol and allow the model to skip a *Match* or *Insert* state. Emission distributions from the pHMM's *Match* hidden states capture an archetypal sequence or sequence fragment, and the likelihood of an observed sequence under a pHMM can be viewed as a measure of distance to the archetypal sequence encoded in the model. Blasiak et. al. [12] defined a simplified version of the pHMM, called the Simplified Local pHMM (SL-pHMM), which generates observed sequences using a contiguous sequence of *Match* states surrounded by *Insert* states. This structure simplifies the pHMM in a convenient way, as the only information needed to encode the model's entire hidden state configuration is the position of the first *Match* state.

The Factorial HMM [13] extends the basic HMM by postulating that the distribution over sequence elements depends on hidden states from multiple, parallel HMMs. If SL-pHMM factors are used, then the resulting Factorial SL-pHMM, with Gaussian emission distributions, operates very similarly to RS-DL.

Figure 2 shows an example configuration of *Match* hidden states in a Factorial SL-pHMM. This hidden state configuration leads to the same additive composition of parameters used to represent symbols of an observed sequence in RS-DL. The primary differences between RS-DL and the Factorial SL-pHMM lie in how the parameters of each model are constrained. In the Factorial SL-pHMM, the model's hidden states can be encoded in a vector, $\alpha_n^{(FHMM)}$ of length $C(T_n - K + 1)$, where C indicates the number of factors in the model, T_n is the length of the sequence, and K is the number of *Match* states in the SL-pHMM. Because the hidden state sequence in the SL-pHMM only allows a single start position for each chain of *Match* states, encoding the positions of initial *Match* hidden states requires that $\alpha_n^{(FHMM)}$ be constrained as $\alpha_{n,i}^{(FHMM)} \in \{0, 1\}$ and $\sum_{t=0}^{T_n-K} \alpha_{n,c(T_n-K+1)+t}^{(FHMM)} = 1 \quad \forall c \in [1 \dots C]$. In contrast, the α_n-vectors in RS-DL are not explicitly constrained but are instead subject to L_1 regularization. Substituting an L_1 regularizer in RS-DL for the binary constraint in the Factorial SL-pHMM is advantageous, because it converts the combinatorial optimization problem associated with the MAP solution over hidden state configurations of the Factorial SL-pHMM into one that is more-easily solvable.

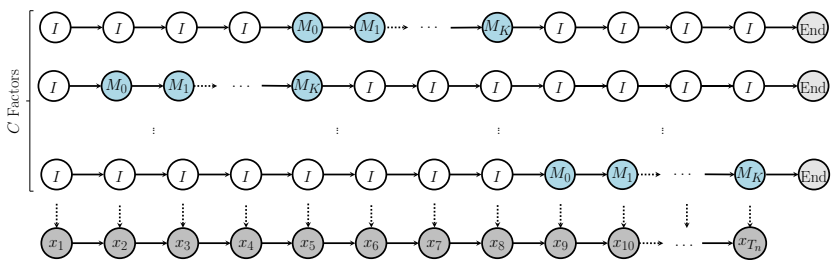


Fig. 2. The diagram above illustrates example hidden state assignments for the Factorial SL-pHMM. Sequences of SL-pHMM *Match* states are indicated by blue nodes with the text "M_k," indicating the k^{th} *Match* state. *Insert* states are indicated by white-colored nodes with the text "I." SL-pHMM transition probabilities are defined so that only a single sequence of *Match* states per individual SL-pHMM can occur. For a Factorial SL-pHMM with Gaussian emission distributions, hidden states are associated with different weights which are summed over the C constituent SL-pHMMs (vertically in the diagram) to obtain the mean parameter used to generate the appropriate observed sequence element (in gray).

5 Experiments

We evaluate Relevant Subsequence Dictionary Learning using two types of measurements. First, we expect RS-DL to find meaningful subsequences within a dataset. This task is also referred to as "motif finding" [14, 15] (other authors [16] use a different definition of the term "motif"). We quantitatively assess motif finding on a synthetic dataset consisting of discrete sequences where the ground truth motif positions are known. We also qualitatively assess motif finding results on sets of both time-series and text sequences to verify that RS-DL can pick out portions of a sequence meaningful to humans. In the next sections we make a distinction between the terms "relevant subsequence pattern" and "motif". We use "relevant subsequence pattern" to indicate the pattern encoded in RS-DL parameters, and "motif" to denote subsequences selected from a dataset because of their association with a particular relevant subsequence pattern.

We also test RS-DL in sequence classification. We hypothesize that if RS-DL can discover informative subsequences with no access to label information, then these subsequence features will be effective for classification. In these experiments, RS-DL features are input to a one-nearest-neighbor classifier to isolate the effect of different feature representations.

5.1 Datasets

We employ four types of datasets to evaluate our algorithm. To evaluate the ability of RS-DL to discover known motifs, we generated a synthetic dataset of discrete-valued sequences containing three predefined subsequences. We also assessed motif finding ability using a set of continuous-valued ECG sequences[3]

[3] http://www.cs.ucr.edu/∼eamonn/discords/ECG_data.zip

and the Associated Press (AP) dataset[4], consisting of English language text. We assessed classification ability using only continuous-valued sequences. These included both a synthetic dataset, which we call the "Bumps" dataset, and datasets from the University of California Riverside (UCR) Time Series Classification Database [17].

5.2 Finding Motifs in Synthetic Sequences

To verify basic motif finding abilities of RS-DL, we constructed a synthetic dataset, allowing us to control the location and frequency of motifs. The synthetic dataset consisted of 20 sequences, generated to contain up to three non-overlapping motifs. These motifs consisted of 5 repetitions of "a," "r," or "n," with a 10% chance at each motif position for a motif character to be replaced by a character generated uniformly from the full sequence alphabet of 20 possible characters. Non-motif sequence elements were chosen uniformly at random from the full 20-characters alphabet. Sequence lengths were generated uniformly at random from a range of 25 to 75.

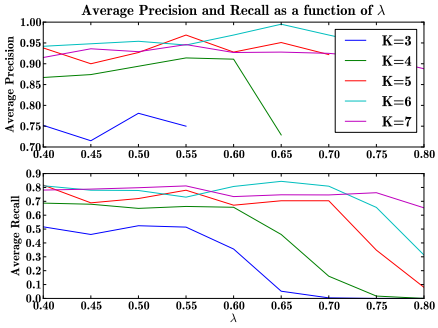
To explore the behavior of the RS-DL model, we ran a number of experiments, varying the values of K, the length of the relevant subsequence pattern, from 3 to 7, and the values of λ, the L_1-regularization parameter, from 0.4 to 0.8 in steps of 0.05. We configured the algorithm to use at most one of each relevant subsequence pattern to reconstruct each sequence.

Figure 3a shows graphs of the average precision and recall associated with motifs recovered by the RS-DL algorithm over 20 trials for each configuration of K and λ. We counted a ground truth motif as "discovered" if its start position was within $\lfloor K/2 \rfloor$ of the motif returned by the RS-DL algorithm. To verify the upper limit of algorithm performance and to confirm the trend that ground truth motifs were associated with larger values of α than false positive motifs, we counted motifs as "not found" if their associated α values were below 0.25. Motifs were extracted by taking the subsequence of length K at the position of an associated non-zero component of the α vector.

Figure 3b, shows the output of the run of the RS-DL algorithm with the lowest mean-squared error (MSE) out of 20 random initializations. Columns in the figure display both values of α and motifs selected from the dataset sequence. Different dataset sequences are associated with different rows in the figure. In this run, low values of α are consistently associated with incorrectly discovered motifs (in red), and, out of four possible relevant subsequence patterns given by the model, only three are used, which is consistent with the ground truth.

Figure 3a shows that both precision and recall tend to increase as the value of K increases. In addition, the figure shows that if we set λ to a value that is too high, both precision and recall are degraded. This behavior, when varying λ, occurs because at high λ levels, the model becomes too sparse, reducing the number of motifs returned. In this case, we do not see a corresponding increase in precision because sparsity is only enforced in the number of relevant

[4] http://www.cs.princeton.edu/~blei/lda-c/ap.tgz



(a)

α	Motif 1	α	Motif 2	α	Motif 3	α	Motif 4
0.41	aaaaa	0.00	—	0.44	rrrrv	0.44	nnnnn
0.32	asaaa	0.00	—	0.42	rrryr	0.18	yrnnl
0.33	laaaa	0.00	—	0.00	—	0.36	nnnen
0.41	aaaaa	0.00	—	0.53	rrrrr	0.00	—
0.41	aaaaa	0.00	—	0.53	rrrrr	0.29	nrnkn
0.17	safra	0.00	—	0.10	anrqv	0.44	nnnnn
0.26	eaapa	0.00	—	0.21	rgqrt	0.08	qrtnd
0.41	aaaaa	0.00	—	0.53	rrrrr	0.44	nnnnn
0.41	aaaaa	0.00	—	0.53	rrrrr	0.44	nnnnn
0.41	aaaaa	0.00	—	0.00	—	0.07	lenma
0.41	aaaaa	0.00	—	0.20	rsgpr	0.09	aaten
0.08	vsnca	0.00	—	0.53	rrrrr	0.44	nnnnn
0.41	aaaaa	0.00	—	0.43	rnrrr	0.44	nnnnn
0.41	aaaaa	0.00	—	0.53	rrrrr	0.44	nnnnn
0.17	vafka	0.00	—	0.32	rrvry	0.08	famln
0.33	aaaea	0.00	—	0.00	—	0.18	nwtwn
0.41	aaaaa	0.00	—	0.53	rrrrr	0.37	nrnnn
0.41	aaaaa	0.00	—	0.43	rrrrn	0.44	nnnnn
0.08	lytia	0.00	—	0.43	rtrrr	0.44	nnnnn
0.08	nigpa	0.00	—	0.08	lkrei	0.36	fnnnn

(b)

Fig. 3. (a) The graphs in the left-hand figure depict precision and recall over 20 runs of the RS-DL model on a synthetic dataset. As the L_1-regularization term, λ, increases, fewer motifs are returned, leading to a drop in both recall and precision. As the length of the relevant subsequence patterns increase, precision and recall tend to increase. **(b)** Recovered α coefficients (left side of Figure b) and associated subsequences (right) from a low-error run (the run with the smallest MSE out of 20 random initializations) of the RS-DL algorithm on the synthetic dataset. The low-error run gave a precision of 0.7 (with an α cutoff of 0) and a recall of 1.0. The number of relevant subsequences patterns, C, in the model was set to four, while the number of ground truth motifs was three. Consistent with the ground truth, the model only used three relevant subsequences patterns to reconstruct the data. Incorrectly discovered motifs are depicted in red.

subsequences patterns used to reconstruct a sequence. Also from Figure 3a, the best precision scores were near 1.0, occurring with $K = 6$ and $\lambda = 0.65$ and filtering motifs with α coefficients less that 0.25. This result contrasts with top precision values of 0.5 (not shown in the figure) when the α filtering level is set to zero. The reason for this trend is illustrated in Figure 3b, where motifs associated with small α coefficients also tend to be less correlated with core relevant subsequence patterns.

To avoid low recall solutions it is possible to rerun the model for a number of trials with initial relevant subsequence patterns, $\mathbf{v}$, drawn from a standard Normal distribution. Because the RS-DL problem is non-convex, the optimization algorithm will converge to different areas in the parameter space depending on initial parameter settings. We found that, with our synthetic dataset, low-MSE runs consistently produced recall values of 1.0 (see Figure 3b). Selecting a low-MSE run also allows us to better take advantage of RS-DL's sparsity. For instance, if we set C, the number of relevant subsequence patterns, to 4, larger than the 3 ground truth motifs in our dataset, then low-MSE solutions return only 3 discovered motifs (higher error solutions find a fourth motif with noisy parameters).

5.3 Motifs in Time-Series Data

To show that RS-DL can pick out patterns meaningful to humans in continuous-valued sequences, we trained it on a single sequence of electrocardiogram (ECG)

data, containing 3750 datapoints. The ECG sequence consists of a recording of electrical signals from the human heart measured at the surface of the skin. A plot of the signal (Figure 4) contains repeated patterns easily identifiable to humans. The ECG sequence also contains an anomalous motif, which, like the main set of patterns in the sequence, is easily identified by humans. We ran the RS-DL algorithm on the sequence with the L_1 regularization term, λ, set to .1 and the length of the relevant subsequence pattern, K, set to 150, and C, the number of relevant subsequence patterns, set to 15. In Figure 4, we plotted the relevant subsequence patterns learned by RS-DL associated with the largest 50 regression coefficients, α. Each pattern in the plot (top three graphs) consists of 150 values of the relevant subsequence pattern given by the constituent vector, $\mathbf{v}$, multiplied by its corresponding α coefficient. Summing over all of these plotted subsequences gives the approximate sequence reconstructed by RS-DL (bottom plot in green, offset by -1). The original sequence is also shown in the figure (bottom plot in blue). The MSE between reconstructed sequence and the original sequence was 0.98. As expected, the figure demonstrates how the relevant subsequence patterns in the upper graphs are strongly correlated with the human-perceptible patterns from the original sequence in the bottom graph. Another interesting property of the RS-DL decomposition shown in Figure 4 relates to the sparsity of the model. Only a three (6, 10, and 11) out of fifteen possible relevant subsequence patterns account for the main patterns in the sequence while additional patterns are responsible for increasingly fine-grained approximations. This type of behavior is similar to commonly-used orthonormal bases, such as the DCT basis, which consist of low frequency components that capture major trends, while high-frequency basis elements capture finer-grained variations. Another characteristic of the RS-DL solution is that the anomalous portion of the sequence is associated with a different relevant subsequence pattern (Relevant Subsequence Pattern 10) than the common ECG pattern. This characteristic shows how RS-DL can be used not only to find positions of recurrent patterns but also to distinguish between pattern types.

5.4 Motifs in Text Data

As an additional test of RS-DL's motif-finding ability, we trained the model on the Associated Press (AP) corpus. We preprocessed the corpus by removing words that occurred more than 500 times or in fewer than three documents. We then removed documents containing fewer than 10 words. The processed corpus size was 2213 documents. Finally, to make processing the text dataset tractable, rather than representing each word as a large binary vector (which would typically have a length of at least 10,000), we used the "word embedding" representation from Collobert et. al. [18]. These word embeddings are vectors in $\mathbb{R}^{50}$ and were constructed so that the Euclidean distance between a pair of vectors is be small if the meanings of the associated words are similar.

Figure 5 shows the top 15 examples, as ordered by the absolute value of the associated α coefficient, of the top four relevant subsequence patterns (out of $C = 10$ total possible relevant subsequence patterns) learned from a run of

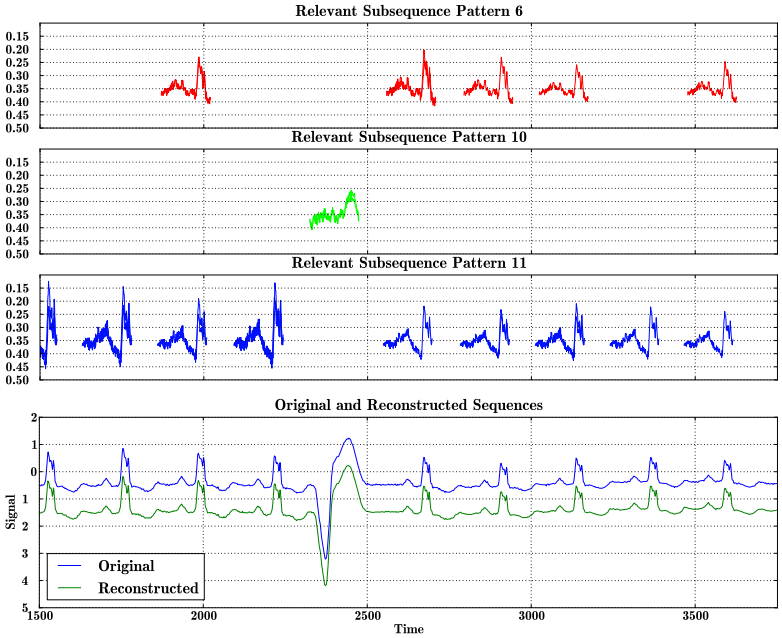


Fig. 4. A plot of the relevant subsequences patterns (upper plots) associated with the largest 50 coefficients of the vector α that were learned by RS-DL to approximate the ECG sequence (bottom plot, blue line). Only 3 out of the 15 possible relevant subsequence patterns appear in this set of 50. Relevant subsequence patterns learned by RS-DL are strongly associated with human-identifiable patterns in the sequence. The figure also shows that the approximation learned by RS-DL (bottom plot, green line) is very similar to the original sequence with an MSE of 0.98. The RS-DL approximation is offset by -1 on the y-axis to aid in presentation.

the RS-DL algorithm. Unlike text processing methods that treat words independently, RS-DL preserves the order of words within each document (minus words removed in the document preprocessing step). As the columns of five-word groups in the figure show, RS-DL, in minimizing reconstruction error over sequences of word embeddings, is capable of finding and grouping together meaningful sequences of words within the text. In the figure, all columns of discovered motifs share internally consistent semantic themes. Moreover, these themes tend to center around phrases containing important nouns. For instance, Motif 1 includes organization-related phrases like "product safety commission defended", "public health system plagued", and "environmental protection agency banned". Motifs 2 and 4 contain phrases including a person and occupation description such as "defense attorney thomas e. wilson", "district attorney william h. ryan", and "secretary james a. baker" in Motif 2 and "attorney michael rosen," "education secretary william bennett," and "assistant district attorney ted stein." Motif 3 is centered on organizations and concepts like "natural resources," "public services," and "tough economic conditions."

Motif 1, max $\|\alpha\| = 5.357$					Motif 2, max $\|\alpha\| = 1.659$				
product	safety	commission	defended	record	defense	attorney	thomas	e.	wilson
community	service	act	provides	community	district	attorney	william	h.	ryan
standards	computer	industry	announced	technology	coalition	secretary	james	a.	baker
intelligence	support	ship	passed	black	assistant	secretary	john	h.	kelly
flight	training	manuals	brought	date	treasury	secretary	james	a.	baker
public	service	employees	received	raise	treasury	secretary	james	a.	baker
public	health	system	plagued	months	district	judge	william	t.	hart
foreign	debt	economy	warned	loss	navy	secretary	james	h.	webb
center	health	promotion	issued	warning	washington	secretary	james	a.	baker
korea	security	force	placed	armed	letter	secretary	james	a.	baker
u.n.	security	council	passed	use	washington	secretary	james	a.	baker
emergency	fund	funds	provided	embassies	date	secretary	james	a.	baker
environmental	protection	agency	banned	chemical	announced	secretary	james	a.	baker
forces	law	order	opened	fire	campaign	chairman	james	a.	baker
justice	information	organization	conducted	survey	assistant	attorney	stephen	a.	mansfield

Motif 3, max $\|\alpha\| = 1.635$					Motif 4, max $\|\alpha\| = 1.628$				
workers	discrimination	foreign	workers	impact	education	university	attorney	michael	rosen
authority	conduct	foreign	policy	direct	debt	education	secretary	william	bennett
detainees	paying	legal	fees	work	hearing	assistant	attorney	john	carroll
assets	include	private	loans	trade	assistant	district	attorney	ted	stein
workers	give	local	unions	grievances	assistant	immigration	commissioner	james	kennedy
program	financed	foreign	aid	assets	white	assistant	attorney	richard	roberts
bills	raise	environmental	protection	agency	questioning	assistant	attorney	john	carroll
planning	aid	foreign	organizations	perform	controversy	press	secretary	david	beckwith
penalties	sought	corporate	executives	release	talks	treasury	secretary	james	brady
problem	finding	foreign	aid	money	deputy	assistant	attorney	john	martin
policies	managing	public	affairs	front	phone	treasury	secretary	james	baker
law	banned	agricultural	products	coal	hearing	district	judge	edward	davis
proposal	noted	natural	resources	issue	list	assistant	attorney	william	reynolds
money	available	public	services	whose	ruling	district	judge	edward	davis
reasons	tough	economic	conditions	particular	acting	prime	minister	ben	jones

Fig. 5. The figure above shows motifs discovered by the RS-DL model in the Associated Press corpus. It lists the top 15 motifs by α coefficient of the top four (out of ten possible) relevant subsequence patterns. Motifs found by RS-DL have, in general, captured sets of semantically coherent phrases. Motifs 1 and 2 contain phrases including organization and noun/concept phrases while Motifs 2 and 4 contain phrases including a person and occupation descriptions.

5.5 Classification Experiments

To assess whether features derived by RS-DL are useful for classification, we compared the performance of these features on both a synthetic dataset of our own design and five UCR Time Series datasets that satisfied the underlying assumptions of our model. Because RS-DL selects subsequences, we do not expect features from the algorithm to be effective for classification when discriminative information between sequence categories lies in global trends over an entire sequence or if the order of different patterns within a sequence is highly correlated with its category. Similarly, because RS-DL is a sparse regression algorithm, we expect relevant subsequence patterns to be matched to high-magnitude areas of dataset sequences. Therefore, if dataset sequences contain large-magnitude areas (e.g. spikes in an ECG sequence), but discriminative information found elsewhere in the sequence, we do not expect RS-DL features to be effective for classification.

With these assumptions in mind, we generated a set of continuous sequences, which we call the "Bumps" dataset[5] (see Figure 6c). Each sequence in this dataset

[5] This dataset can be found at http://cs.gmu.edu/∼sblasiak/RS-DL.tar.gz

Table 2. Classification results using RS-DL features on the UCR Time Series datasets. The "Sequence", "DTW", and "RS-DL" columns give error rates from the one-nearest-neighbor algorithm using the Euclidean distance between sequences, Dynamic Time Warping scores, and Euclidean distance between RS-DL features respectively. RS-DL features improved the classification error for all three datasets.

Dataset	# Categories	# Train	# Test	Sequence	DTW	RS-DL
Bumps (synthetic)	2	50	50	0.460	0.140	**0.056**
CBF	3	30	900	0.148	**0.030**	0.108
Coffee	2	28	28	0.250	0.214	**0.171**
DiatomSizeReduction	4	16	306	0.065	0.042	**0.028**
ECGFiveDays	2	23	861	0.203	0.249	**0.095**
TwoLeadECG	2	23	1139	0.253	0.073	**0.035**

contains two large magnitude bumps placed at random and without overlap. In the negative category one out of the two bumps in each sequence contains a divot. We also selected five datasets from the UCR Time Series database that conform to the underlying assumptions about RS-DL: CBF, Coffee, DiatomSizeReduction, ECGFiveDays, and TwoLeadECG. These datasets consist of sequences that contain large magnitude patterns occurring in all or nearly all sequences, satisfying the assumptions needed for RS-DL to extract useful features.

We ran RS-DL with randomly initialized $\mathbf{v}$ arrays for ten trials on all sequences in both the training and test sets, excluding label information, for each dataset. For all experiments, we set $C = 10$, K to 30% of the sequence length, and $\lambda = 3.0$. We also enabled the restriction on the LARS algorithm (see Section 3.2) where only a single relevant subsequence pattern of each type was used. For each sequence, we created feature vectors by concatenating the subsequences associated with each relevant subsequence pattern. Table 2 shows a comparison of classification errors using the one-nearest-neighbor algorithm on (i) features given by treating sequences as vectors in Euclidean space, (ii) Dynamic Time Warping (DTW)[6] [19] distances between sequences, and (iii) Euclidean distance between RS-DL feature vectors. As assessed by McNemar's test [20], RS-DL features reduce classification error over raw sequence vectors with p-values of less than 0.014 for all datasets. For all datasets except for the CBF dataset, RS-DL features improved on the classification error over DTW. Here, all results were significant with p-values of less than 0.01, except for the Coffee dataset, where RS-DL's improvement over DTW was significant with a p-value of 0.17.

Figure 6 shows examples of positive and negative category sequences from three of the classification datasets. In each case, RS-DL features lead to improved time-series category prediction by isolating large-magnitude trends in subsequences shared across the set of sequences (as shown in the upper portions of each plot in the figure). Constructing features from these isolated subsequences aligns these major subsequence trends, allowing minor variations that occur between the positive and negative sequence categories to be more-easily distinguished. When variations in the general trend are highly correlated with a

[6] DTW scores were computed in R using http://dtw.r-forge.r-project.org/

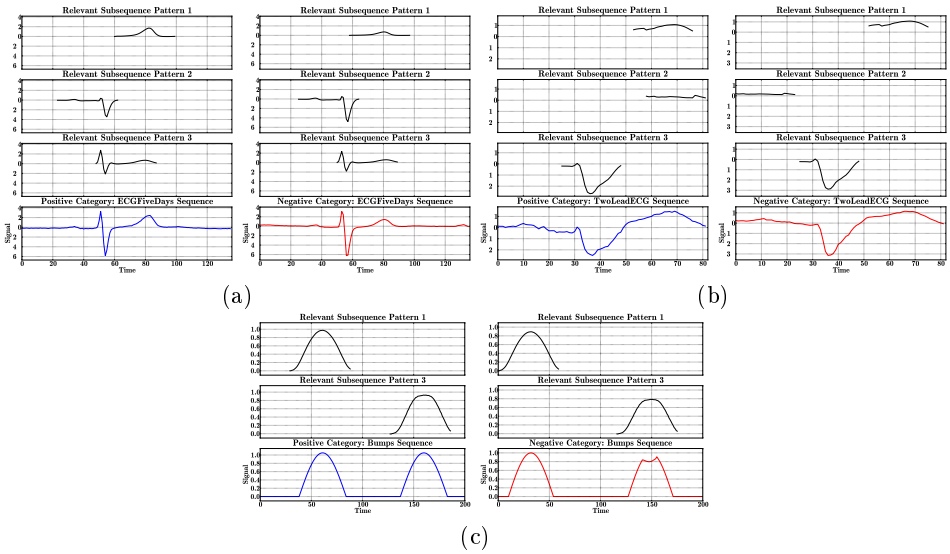


Fig. 6. Figures **a**, **b**, and **c** above show the top two (by α value) relevant subsequence patterns that approximate positive (bottom blue) and negative category (bottom red) sequences in the ECGFiveDays, TwoLeadECG, and our synthetically-generated Bumps datasets respectively. For each of these datasets, RS-DL features improve classification performance by picking out similarly shaped subsequences from different dataset categories. Classification performance improves because class distinctions occur in minor variations in the major trends captured by RS-DL. After processing by RS-DL, these minor variations can more easily be distinguished by standard classification algorithms.

category label, then the feature isolation provided by RS-DL can lead to more accurate classification.

6 Conclusions

In this paper, we have presented Relevant Subsequence Dictionary Learning, a novel method for adapting Sparse Dictionary Learning to discover interesting subsequence patterns across sets of sequences. RS-DL is related to standard statistical models over sequences through a version of the Factorial HMM with specially formulated restrictions on transition probabilities. In a series of experiments, we have shown that RS-DL can discover useful information across a variety of sequence domains. In addition, as demonstrated on time-series data, sequence features extracted using RS-DL can improve sequence classification performance.

Acknowledgments

This work was supported by NSF grant IIS-0905117 and NSF Career Award IIS-1252318 to HR.

References

1. Mairal, J., Bach, F., Ponce, J., Sapiro, G., Zisserman, A.: Discriminative learned dictionaries for local image analysis. In: Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on, IEEE (2008) 1–8
2. Mairal, J., Leordeanu, M., Bach, F., Hebert, M., Ponce, J.: Discriminative sparse image models for class-specific edge detection and image interpretation. In: Proceedings of the European Conference on Computer Vision (ECCV). (2008)
3. Mairal, J., Bach, F., Ponce, J., Sapiro, G.: Online learning for matrix factorization and sparse coding. The Journal of Machine Learning Research **11** (2010) 19–60
4. Yang, J., Yu, K., Gong, Y., Huang, T.: Linear spatial pyramid matching using sparse coding for image classification. In: Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on, IEEE (2009) 1794–1801
5. Lee, H., Battle, A., Raina, R., Ng, A.Y.: Efficient sparse coding algorithms. Advances in neural information processing systems **19** (2007) 801
6. Boureau, Y., Bach, F., LeCun, Y., Ponce, J.: Learning mid-level features for recognition. In: Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on, IEEE (2010) 2559–2566
7. Aharon, M., Elad, M., Bruckstein, A.: K-svd: Design of dictionaries for sparse representation. Signal Processing, IEEE Transactions on **54**(11) (2006) 4311–4322
8. Efron, B., Hastie, T., Johnstone, I., Tibshirani, R.: Least angle regression. The Annals of statistics **32**(2) (2004) 407–499
9. Friedman, J., Hastie, T., Höfling, H., Tibshirani, R.: Pathwise coordinate optimization. The Annals of Applied Statistics **1**(2) (2007) 302–332
10. Beck, A., Teboulle, M.: A fast iterative shrinkage-thresholding algorithm for linear inverse problems. SIAM Journal on Imaging Sciences **2**(1) (2009) 183–202
11. Eddy, S.R.: Profile hidden markov models. Bioinformatics **14**(9) (1998) 755
12. Blasiak, S., Rangwala, H., Laskey, K.B.: A family of feed-forward models for protein sequence classification. In: Machine Learning and Knowledge Discovery in Databases. Springer (2012) 419–434
13. Ghahramani, Z., Jordan, M.I.: Factorial hidden markov models. Machine learning **29**(2-3) (1997) 245–273
14. Bailey, T.L., Elkan, C.: Fitting a mixture model by expectation maximization to discover motifs in biopolymers. vectors **1** 2
15. Buhler, J., Tompa, M.: Finding motifs using random projections. Journal of computational biology **9**(2) (2002) 225–242
16. Mueen, A., Keogh, E., Zhu, Q., Cash, S., Westover, B.: Exact discovery of time series motifs. In: Proc. of 2009 SIAM International Conference on Data Mining: SDM. (2009) 1–12
17. Keogh, E., Xi, X., Wei, L., Ratanamahatana, C.A.: The ucr time series classification/clustering homepage (2011)
18. Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., Kuksa, P.: Natural language processing (almost) from scratch. The Journal of Machine Learning Research **12** (2011) 2493–2537
19. Sakoe, H., Chiba, S.: Dynamic programming algorithm optimization for spoken word recognition. Acoustics, Speech and Signal Processing, IEEE Transactions on **26**(1) (1978) 43–49
20. Dietterich, T.G.: Approximate statistical tests for comparing supervised classification learning algorithms. Neural computation **10**(7) (1998) 1895–1923