A Supervised Time Series Feature Extraction Technique using DCT and DWT

Iyad Batal and Milos Hauskrecht
Department of Computer Science
University of Pittsbugh
{*iyad, milos*}*@cs.pitt.edu*

Abstract

The increased availability of time series datasets prompts the development of new tools and methods that allow machine learning classifiers to better cope with time series data. Time series data are usually characterized by a high space dimensionality and a very strong correlation among features. This special nature makes the development of effective time series classifiers a challenging task. This work proposes and analyzes methods combining spectral decomposition and feature selection for time series classification problems and compares them against methods that work with original time series and time-dependent features. Briefly, our approach first applies discrete cosine transform (DCT) or discrete wavelet transform (DWT) on time series data. Then, it performs supervised feature selection/reduction by selecting only the most discriminative set of coefficients to represent the data. Experimental evaluations, carried out on multiple datasets, demonstrate the benefits of our approach in learning efficient and accurate time series classifiers.

1 Introduction

Data classification is an important machine learning problem with a wide variety of applications. While the field of classification has witnessed excellent achievements in recent years, not much attention has been given to techniques for classifying time series data [3, 11]. Such classification tasks may arise in a variety of domains, such as speech and gesture recognition, intrusion detection, industrial plants monitoring, or ECG data analysis. With increasing availability of time series datasets, the development of effective time series classification algorithms is becoming more important and pressing.

The caveat of dealing with time series data is their high dimensionality and strong correlation among features. This unique structure poses a number of challenges, which have hindered the development of accurate time series classification techniques over the years. To illustrate this point, the authors in [9] experimentally showed, using a variety of datasets, that the simple one-nearest neighbor (1NN) classifier with Euclidean distance outperforms other time series classifiers proposed in the literature by a wide margin.

The objective of this work is to study techniques that automatically extract useful classification features from time series data. This task is challenging because the high feature correlations makes it difficult to develop effective feature selection techniques. Our goal is to study the classification benefits of methods that rely on selecting features from the spectral domain, and compare them to methods that work directly on time series data.

The majority of spectral feature extraction techniques for time series data proposed in the literature are unsupervised [1, 6, 16]. In this paper, we study a supervised spectral feature extraction techniques for time series classification problems. In our approach, all time series are first transformed from the time domain into a frequency domain using discrete cosine transform (DCT) [2], or into a time-frequency domain using discrete wavelet transform (DWT) [5]. This step is crucial because it helps reducing the high correlation between the features in the time domain and revealing hidden information about the series. After this transformation, dimensionality reduction is performed by using the *F-statistic* to select a small subset of the DCT or DWT coefficients which are the most important for the classification task. These discriminative spectral coefficients define a new feature space for representing the time series data and conventional machine learning methods are employed to build the classifier.

We demonstrate, using both the k-nearest neighbor (KNN) and the support vector machines (SVM) classifiers, that features selected from the spectral domain are much more effective than features selected from the time domain.

The rest of the paper is organized as follows: Section 2 explains our feature extraction methodology. We start by giving the necessary background on the DCT and DWT decompositions. Next, we present our criterion for selecting the discriminative features that will be used by the classifier. We present the experimental evaluation of our approach in section 3 and finally conclude in section 4.

978-0-7695-3926-3/09 $26.00 © 2009 IEEE
DOI 10.1109/ICMLA.2009.13

2 Methodology

2.1 Spectral time series decomposition

In most real world time series, consecutive values of a time series are usually not independent, but highly correlated. This makes it difficult to develop effective feature selection techniques that work directly on the time series data. To alleviate the problem, time series can be transformed from the time domain into another domain in order to de-correlate the time features and reveal the hidden structure of the series. In this section, we describe two popular time series transformation techniques.

2.1.1 Discrete Cosine Transform (DCT)

DCT is a Fourier-related transform similar to discrete Fourier transform (DFT), but uses only cosine functions instead of using both cosines and sins. It transforms the input signal from the time domain into the frequency domain, which highlights the periodicity of the signal. DCT has been successfully used in JPEG [17] for image compression and MPEG [8] for video compression.

DCT for a time series x of length n is defined as:

$$X_f = K(f) \sum_{i=1}^{n} x_i cos \frac{\pi f(i-0.5)}{n} \quad f = 0, ..., n-1 \quad (1)$$

Where $K(f) = \frac{1}{\sqrt{n}}$ when $f = 0$ and $K(f) = \sqrt{\frac{2}{n}}$ when $1 \leq f \leq n-1$. Multiplying by $K(f)$ makes DCT an orthonormal transformation.

It is possible to compute all DCT coefficients using $O(n\, log(n))$ operations in a way similar to the Fast Fourier Transform (FFT) algorithm.

We chose to use DCT in this work because it offers the following desirable properties compared to DFT:

1. DCT coefficients are always real numbers, as opposed to the DFT complex coefficients.
2. DCT can handle well signals with trends, whereas DFT suffers from the "frequency leak" problem when representing simple trends.
3. When successive values are highly correlated, DCT achieves better energy concentration than DFT and its performance compares closely to the optimal Karhunen-Loève transform [2].

2.1.2 Discrete Wavelet Transform (DWT)

Fourier transforms (both DFT and DCT) assume the signal to be periodic and have no temporal locality, i.e., each coefficient provides information about all time points. However, many real-world time series are not periodic. Wavelet transforms [5] offers the most recent solution to overcome this shortcoming by mapping signals into a joint time-frequency domain. DWT hierarchically decomposes the signal using windows of different sizes, hence producing multi-resolution analysis.

Haar wavelets: in our work we adopt the Haar wavelet, one of the simplest and one of the most popular wavelets. In order for the Haar decomposition to be applied, the input single must have a length that is a power of 2. This is usually done by padding zeros to the end of the time series if the length doesn't satisfy this requirement.

For an input x of length n (after the padding), the Haar coefficients are computed recursively in a bottom-up fashion as follows:

$$d_{l,i} = \frac{1}{\sqrt{2}}(s_{l-1,2i} - s_{l-1,2i+1})$$
$$s_{l,i} = \frac{1}{\sqrt{2}}(s_{l-1,2i} + s_{l-1,2i+1})$$

Where $l = 1, ..., log_2(n), \quad i = 1, ..., \frac{n}{2^l} \quad$ and $\quad s_{0,i} = x_i$.

The Haar transform of the original signal is the set of all difference values $d_{l,i}$ at every level l and every offset i (resulting in $n-1$ differences), plus the smooth component at the last level $s_{log(n),0}$. the constant $\frac{1}{\sqrt{2}}$ in the equations is used to make the transformation orthonormal. From the equation, we can see that the computational complexity of Haar decomposition is linear $O(n)$.

2.2 Feature selection

As we saw earlier, both DCT and DWT are orthonormal time series transformations. This means that if we envision the input time series of length n as a vector in an n-dimensional space, applying DCT or DWT can be seen as a rotation of the space axes. These transformations do not affect the length of the original series (according to Parseval's theorem [13]), nor the Euclidean distance between any pair of series [7]. Therefore, we can state that: *Applying KNN or SVM using all DCT or all DWT coefficients gives the exact same classification performance as applying it using all original time features.*

However, this transformation invariance is likely to vanish when subsets of features are considered. In other words, significant differences between classifiers with subsets of time features and classifiers with subsets of spectral features may arise. Applying feature subset selection is often necessary, especially for high dimensional data, to improve the classifier's accuracy. Besides, it can greatly speed up the classification time of KNN.

In this work, we adopt a univariate feature selection technique based on the *F-statistic*. The objective is to show that selecting features from spectral domains is much more effective than selecting features directly from the time domain and to show that combining spectral decomposition and feature selection results in accurate time series classifiers.

2.2.1 Feature selection using the F-statistic

In this section, we explain our approach for selecting the discriminative spectral coefficients. Our method relies on the *F-statistic* used by ANOVA.

ANOVA (ANalysis Of VAriance) is a statistical technique for comparing means of multiple groups under the null hypothesis that all means are equal. ANOVA produces an *F-statistic*, defined as:

$$F = \frac{variance\ of\ the\ group\ means}{mean\ of\ the\ within\ group\ variances} \qquad (2)$$

Essentially, the F value is a comparison of the variance amongst the different groups to the variance amongst all the individuals within those groups. If all means are equal, the F ratio should be approximately one. If the ratio is much larger than one, then the between-groups variability is larger than the within-group variability, i.e. there is significant differences between the groups.

In our context, we use the *F-statistic* to measure the importance of a specific DCT or DWT coefficient based on the values of this coefficient from all series in the training set. The groups in equation (2) represent the different time series classes. Thus, if a coefficient scores a high F value, it means that data from the different classes in terms of this coefficient can be easily separated by the classifier. Therefore, we rank all spectral coefficients according to their F values and select a small set of them be the classification features.

Note that even though ANOVA relies on the assumption that the groups are normally distributed, experiments in [10] showed that ANOVA is a relatively robust with respect to violations of the normality assumption.

When the data contains only two classes, using ANOVA is equivalent to using *t-test* and the relation is given by $F = t^2$, where *t* is the *Student's t statistic*. But when there are more than two classes, ANOVA becomes more effective than all pairwise t-tests. The reason is that ANOVA puts all the data into one F number as opposed to giving multiple t values using pairwise t-tests.

3 Experimental evaluations

In this section, we empirically examine the usefulness of different feature extraction techniques. We show that applying the feature selection step on the transformed time series is much more effective than applying it directly on the time series. We also compare our approach against classifiers that use all features (without dimensionality reduction). We report classification results using two popular classifiers: k-nearest neighbors (KNN) and support vector machine (SVM).

3.1 Classification algorithms

KNN is an instance-based learning method for classifying objects based on their closest training examples in the feature space. An object is classified by a majority vote of its neighbors, with the object being assigned to the class most common amongst its k nearest neighbors. We use the Euclidean distance as the distance measure. KNN has the advantages of being a non-parametric and non-linear classifier. However, its main disadvantage is that it is slow at classification time since it defers all computation until classification (a lazy learner).

SVM [14] is a very popular discriminative machine learning model. A special property of SVM is that it simultaneously minimizes the empirical classification error, while maximizes the geometric margin; hence known as a *maximum margin* classifier. In our work, we adopt *linear kernel* SVM. We employ the Lagrangian support vector machine algorithm [12] for solving the quadratic programming (QP) optimization problem. In order to perform multi-class classification with SVM, we adopt the *one-against-all* approach, using the continuous values of SVM decision functions (Vapnik [15]), rather than simply their signs to make the classification. This means that a data point is assigned to the class that has a decision function with the highest value. Note that this method does not leave undecided regions in the feature space.

3.2 Data

Synthetic: We chose to first experiment with a synthetic dataset because it allows us to better understand the relationship between the classification accuracy and the characteristics of the data. We defined two classes: C_1 and C_2, and generate 100 series from each as follows:

$$x_t = cos(2\pi 5t/100) + cos(2\pi 12t/100) + w_t \text{ for } x_t \in C_1$$
$$x_t = cos(2\pi 5t/100) + cos(2\pi 13t/100) + w_t \text{ for } x_t \in C_2$$

where $t = 1, .., 100$ and w_t is a Gaussian noise with standard deviation σ=2.

The most prominent frequencies for C_1 series are 5 and 12, while for C_2 series are 5 and 13. The series from the two classes look very similar when represented in the time domain since they are obscured by the noise term w_t.

CBF: The learning task is to distinguish three classes of time series: Cylinder, Bell and Funnel. This dataset contains 310 instances from each of the three classes.

Control Charts (CC): This dataset contains six different classes of control charts: normal, cyclic, increasing trend, decreasing trend, upward shift and downward shift.

Face: This is a face recognition problem. The data consists of four different individuals (3 males and 1 female), making different facial expressions. The task is to identify the person based on his head profile, which is represented as "pseudo time series".

Trace: This is a synthetic dataset designed to simulate instrumentation failures in a nuclear power plant. It contains 4 different classes and 50 instances from each class.

Dataset	Classes	Instances	Time Series length
Synthetic	2	200	100
CBF	3	930	128
Control Charts	6	600	60
Face	4	112	350
Trace	4	200	275

Table 1. The characteristics of each dataset

3.3 Experimental settings

In our experiments, we evaluate different feature extraction techniques that reduce the data dimensionality from the full time series length to a chosen parameter k. *Time_FS* / *DCT_FS* / *Haar_FS* selects the top k time features / cosine coefficients / Haar coefficients (respectively) according to their F scores. We also evaluate the performance of classifiers that use the original series, which we term as *Time series*. Please remember that using all DCT or all Haar coefficients gives the exact same classification accuracy as using all time features (Section 2.2).

We randomly select 80% of the data for training and use the other 20% for testing and repeat the sampling 10 times. All methods are tested using the exact same train/test splits and features are extracted only from the training sets, i.e., different features may be selected in the different rounds.

3.4 Results

Figure 1 shows the classification error rate on the synthetic dataset. We vary the number of features (k) from 1 to 50 (the total number of features is 100) and plot the corresponding classification error for both 1NN (top) and SVM (bottom). We also show the error when using all features as a constant line. We can see that *Time_FS* does not produce accurate classifiers because the important information is hidden in the spectral domain. On the other hand, using *DCT_FS*, both classifiers achieve good accuracies even for small number of features. The reason is that *DCT_FS* can directly spot the most discriminative coefficients to be the classification features. These coefficients correspond to cosines with frequencies 12 and 13. So by using only two *DCT_FS* features, 1NN error rate is 2.25% and SVM error rate is 1.75%. We can also see that *Haar_FS* performs well, even though the signals are purely periodic. However, *Haar_FS* requires more features to reach its optimal performance compared with *DCT_FS*.

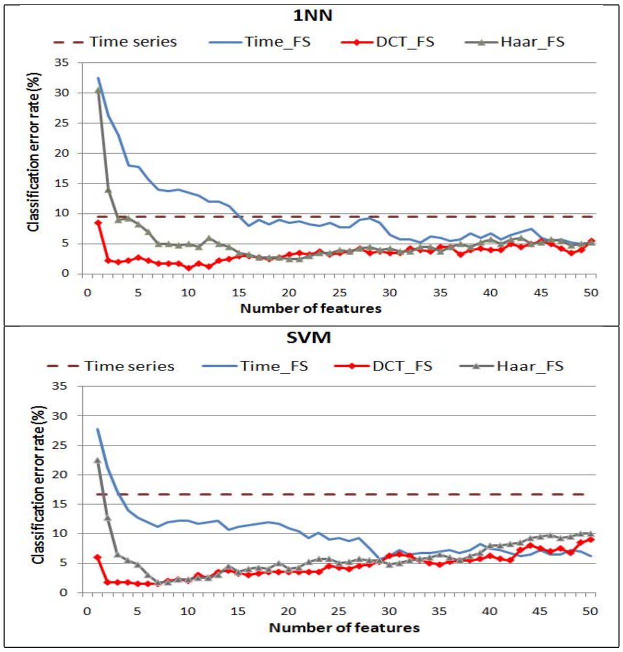

Figure 1. The classification error rate (%) of the 1NN (top) and SVM (bottom) classifiers on the synthetic data using different number of features ($k \in [1..50]$). The constant dashed line show the error using all features.

Let us examine 1NN performance when we set k (the number of features) to be 10% the length of the series. Selecting features directly from the time domain gives an error of 13.5%. On the other hand, *DCT_FS* achieves the best performance with an error of 1% and *Haar_FS* also achieves a good performance with an error of 5%. Hence, using 10% of *DCT_FS* features reduces 1NN classification error by about 89% compared with using all features (1NN error using the original series is 9.5%).

Figure 1 also shows SVM performance (bottom). By setting k to 10% the length of the series, SVM error is 2% using *DCT_FS* and 2.2% using *Haar_FS*. So using *DCT_FS* or *Haar_FS* reduces SVM classification error by about 87% compared with using all features (SVM error using the original series is 16.7%).

Table 2 shows 1NN classification errors on CBF, CC, Face and Trace datasets for $k = 10\%$ the length of the series. The text in bold shows the error rate of the best performing method on each dataset. We can see that for the CBF dataset, *DCT_FS* achieves perfect classification, as opposed to 1.24% error rate using the original time series. For CC, Face and Trace datasets, *Haar_FS* is the best performing method.

Notice that selecting features from the time domain greatly hurts 1NN classification accuracy for most datasets. For example, using *Time_FS* on the CBF dataset performs about 11 times worse than using the original time series.

From the table, we can see that using 1NN on our supervised spectral features improves the accuracy compared with using 1NN on the original series. These results are

	CBF	CC	Face	Trace
Time series	1.24	7.5	8.6	13.5
Time_FS	13.71	24.25	6.09	28.5
DCT_FS	**0**	5.1	5.2	16.1
Haar_FS	0.8	**4**	**4.1**	**11.2**

Table 2. The classification error rate (%) of the 1NN classifier. The number of features k is 10% the length of the series. The bold text shows the classification error of the best performing method on each dataset.

very interesting because 1NN on the original series has been shown to be one of the top time series classifiers [9].

Furthermore, using the reduced data greatly speeds up 1NN classification time, which is its major shortcomings. We know that the computational complexity to classify one instance using 1NN with Euclidean distance on the full data is $n * m$, where n is the length of the series and m is the number of instances in the training data. For the *DCT_FS* and *Haar_FS* methods, the transformation and feature selection steps are performed offline on the training data in order to create a new representation. At classification time, the new instance is first transformed by DCT or by Haar and the selected features are kept. Then it is compared against the reduced representation of the training data, which requires $k * m$ operations. So by reducing the dimensionality to only $k = 10\%n$, 1NN can perform about 10 times faster.

	CBF	CC	Face	Trace
Time series	4.27	17	5.91	25.25
Time_FS	9.62	30.7	5.6	33
DCT_FS	2.12	**9.8**	**5.4**	25.2
Haar_FS	**1.72**	10.1	5.45	**19.7**

Table 3. The classification error rate (%) of linear SVM. The number of features k is 10% the length of the series. The bold text shows the classification error of the best performing method on each dataset.

Table 3 shows the performance of the linear SVM classifier. We can see that by applying appropriate feature extraction, SVM performs better than when using all features (unless the extracted features are bad representative such as the *Time_FS* features). The reason is that dimensionality reduction helps preventing SVM from overfitting the data. The best performing methods are: *Haar_FS* for the CBF and Trace datasets, and *DCT_FS* for the CC and Face datasets.

4 Conclusion

In this work, we have analyzed the benefits of using spectral features for time series classification. Our framework combines spectral decomposition, using DCT or DWT, and feature selection in order to generate classification features.

We conduct our experiments on a collection of time series classification problems using two popular learning models: 1NN and SVM. The results showed that feature selection from the spectral domain is much more effective than feature selection from the time domain. The results also showed that using a small number of spectral features can improve the classification accuracy as opposed to using the complete time series. This is important because 1NN on the complete time series was shown to be one of the top performing time series classifiers [9].

References

[1] R. Agrawal, C. Faloutsos, and A. Swami. Efficient Similarity Search in Sequence Databases. In *proceedings of FODO*, 1993.

[2] N. Ahmed, T. Natarajan, and K. R. Rao. Discrete Cosine Transform. In *IEEE Transactions on Computers*, 1974.

[3] C. Antunes and A. Oliveira. Temporal data mining: an overview. In *proceedings of Workshop on Temporal Data Mining*, 2001.

[4] I. Batal, L. Sacchi, R. Bellazzi, and M. Hauskrecht. Multivariate Time Series Classification with Temporal Abstractions. In *proceedings of FLAIRS*, 2009.

[5] S. Burrus, R. Gopinath, and G. Guo. *Introduction to Wavelets and Wavelet Transform*. Prentice-Hall, Englewood Cliffs, N. J., 1997.

[6] K. Chan and A. Fu. Efficient Time Series Matching by Wavelets. In *proceedings of ICDE*, 1999.

[7] C. Faloutsos. *Searching Multimedia Databases by Content*. Kluwer Academic Publishers, Norwell, MA, first edition, 1996.

[8] D. L. Gall. MPEG: A Video Compression Standard for Multimedia Applications. *Communications of the ACM*, 34(4):46–58, 1991.

[9] E. Keogh and S. Kasetty. On the need for time series data mining benchmarks: a survey and empirical demonstration. In *proceedings of ACM SIGKDD*, 2002.

[10] R. E. Kirk. *Experimental Design: Procedures for the Behavioral Sciences*. Brooks/Cole, Pacific Grove, CA, third edition, 1995.

[11] S. Laxman and P. Sastry. A survey of temporal data mining. *SADHANA, Academy Proceedings in Engineering Sciences*, 31:173–198, 2006.

[12] O. Mangasarian and D. Musicant. Lagrangian Support Vector Machines. *Journal of Machine Learning Research*, 2001.

[13] A. Oppenheim and R. Schafer. *Digital Signal Processing*. Prentice-Hall, Englewood Cliffs, NJ, 1975.

[14] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, NY, 1995.

[15] V. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, 1998.

[16] M. Vlachos, J. Lin, E. Keogh, and D. Gunopulos. A Wavelet-Based Anytime Algorithm for K-Means Clustering of Time Series. In *proceedings of SIAM International Conference on Data Mining*, 2003.

[17] G. Wallace. The JPEG Still Picture Compression Standard. *Communications of the ACM*, 34(4):30–44, 1991.