Técnicas de Aprendizaje Automático para la Clasificación de Series*

Juan José Rodríguez[1] y Carlos J. Alonso[2]

[1] Lenguajes y Sistemas Informáticos
Universidad de Burgos, Spain
jjrodriguez@ubu.es
[2] Grupo de Sistemas Inteligentes
Departamento de Informática
Universidad de Valladolid, Spain
calonso@infor.uva.es

Resumen En este trabajo se resumen la tesis del mismo título [21] realizada por el primer autor bajo la dirección del segundo. En la misma se desarrollan técnicas para la inducción de clasificadores sobre series. En primer lugar, se definen predicados que caracterizan propiedades importantes de las series, junto con métodos que permiten su selección. Hay dos tipos de predicados, los que consideran el valor de alguna función en un intervalo y los que tienen en cuenta alguna medida de disimilitud entre series.
Para obtener clasificadores precisos se combinan los mismos mediante boosting. Se presenta un método para utilizar los clasificadores cuando sólo se dispone de series incompletas, ya que hay valores de las mismas que se conocerán en el futuro. Los predicados seleccionados mediante boosting pueden usarse en árboles de decisión para obtener clasificadores más comprensibles o con Máquinas de Vectores Soporte, buscando clasificadores más precisos.
La validación experimental ha considerado, entre otros, conjuntos de datos sobre el reconocimiento de hablantes, de dígitos manuscritos, de signos del lenguaje de sordos y de fallos en procesos industriales continuos.

1. Introducción

El problema considerado es la inducción de clasificadores sobre series. Es una tarea de clasificación supervisada con la diferencia de que los valores de los atributos están ordenados e indexados por el tiempo. Aunque es posible utilizar cualquier método de inducción considerando como atributos de entrada combinaciones de atributos e instantes (e.g., el atributo x_7 es el valor de x en el instante 7), existen métodos para la construcción de clasificadores específicos para este tipo de datos, entre los que se encuentran los desarrollados en esta tesis. Las técnicas más consolidadas en este dominio son los Modelos Ocultos de Markov [14], las Redes de Neuronas para series [8] y el Alineamiento Dinámico

* Este trabajo ha sido financiado por el proyecto del MCyT DPI2002–01809.

Temporal [5]. Algunas aproximaciones recientes, dentro de los métodos simbólicos en aprendizaje automático son [13], [7] y [10]. En estas tres aproximaciones se seleccionan características de las series que se utilizan en, por ejemplo, árboles de decisión.

El origen de esta línea de investigación está en el desarrollo de un sistema basado en conocimiento para la diagnosis en línea de un proceso industrial [3]. Una de las características del sistema era que permitía a los usuarios la modificación de las reglas del sistema. El formato de las reglas venía dado por el lenguaje utilizado por los operarios del proceso, incluyendo condiciones del tipo, "en la ultima media hora una determinada variable ha tenido siempre valores altos y alguna vez ha tenido un valor muy alto". Así, la motivación original fue obtener este tipo de reglas utilizando técnicas de aprendizaje automático.

El resto del trabajo se organiza como sigue. La sección 2 presenta brevemente los predicados definidos así como cuál es el proceso seguido para su selección. El uso de estos predicados con distintos modos de clasificación se discute en la sección 3. La sección 4 introduce los conjuntos de datos más interesantes en los que se han aplicado las técnicas desarrolladas. Para finalizar, las conclusiones se presentan en la sección 5

2. Predicados

2.1. Definición

La figura 1 muestra los distintos tipos de predicados considerados.

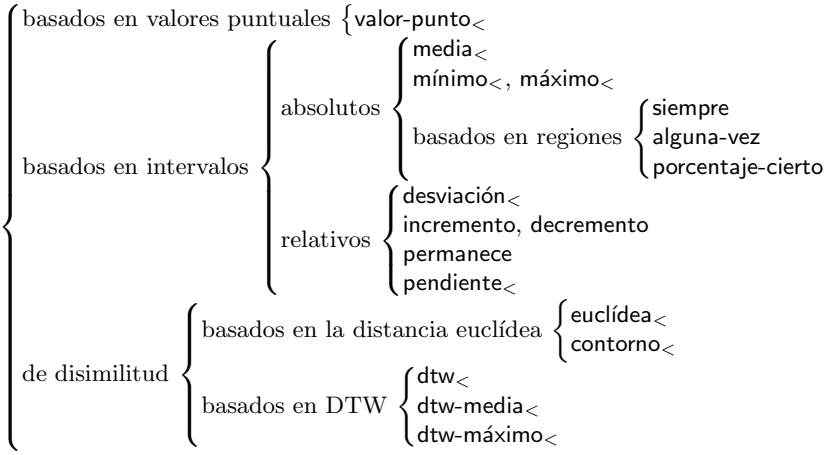

Figura 1. Tipos de predicados

Predicado puntual. Sólo considera un valor puntual:

- valor-punto(Ejemplo, Variable, Punto, Umbral)

Para el Ejemplo se toma el valor indicado por Punto (e.g., la posición 50) de la serie indicada por la Variable (e.g., x) y se compara con el Umbral.

Este predicado se introduce como criterio base de comparación, ya que es el tipo de decisiones que aparecería en un sistema de inducción de los denominados "atributo valor", como pueden ser los árboles de decisión, que están formados por comparaciones de atributos con valores. Por ejemplo, la decisión $x_{50} <$Umbral.

Predicados basados en intervalos. Consideran alguna propiedad que se cumple en un intervalo de una serie. Dentro de este tipo de predicados se hace una distinción entre *absolutos* y *relativos*, en función de si la propiedad establecida sobre el intervalo depende de los valores absolutos o de los relativos.

Bastantes de estos predicados (media, mínimo, máximo, desviación y pendiente) calculan una función sobre los valores del intervalo y comparan el resultado con un umbral:

- función-intervalo$_<$(Ejemplo, Variable, Inicio, Fin, Umbral)

Los predicados incremento y decremento comparan los valores extremos de un intervalo para comprobar si se ha producido un incremento o decremento mayor que un umbral. El predicado permanece comprueba que todos los valores de un intervalo permanecen en un rango cuya amplitud no sea mayor que el umbral.

Dentro de los predicados absolutos se encuentran los basados en *regiones*, donde una región es un rango en el dominio de valores de la variable. Se puede comprobar si la variable siempre o alguna vez dentro del intervalo se encuentra en la región. El predicado porcentaje-cierto permite exigir que la variable se encuentre en la región en al menos un determinado porcentaje.

Predicados de disimilitud . Estos predicados calculan algún tipo de medida de la disimilitud entra dos series y comparan el resultado con un umbral:

- función-disimilitud$_<$(Ejemplo, Referencia, Variable, Umbral)

Es decir, se calcula la función-disimilitud, para una determinada Variable, entre el Ejemplo que se quiere clasificar con otro ejemplo de Referencia y se compara el resultado con el Umbral. La función de disimilitud más habitual es la euclídea, pero para series es más adecuado utilizar Alineamiento Dinámico Temporal (DTW) [11].

2.2. Selección de Predicados.

El proceso de selección de literales lo que hace es tomar como entrada ejemplos de dos clases y seleccionar un predicado tomando como criterio la capacidad del mismo para discriminar entre esas dos clases. Este proceso se detalla, además

de en [21], en [4,17,18] para los predicados basados en intervalos y en [19] para los basados en disimilitud.

La mayoría de los predicados comparan el valor de alguna función con un umbral. Por tanto, el punto de partida es el proceso seguido para seleccionar un umbral para un atributo numérico al construir, por ejemplo, un árbol de decisión.

Dado que lo que se quiere es seleccionar umbrales para funciones en vez de atributos, la aproximación más directa sería considerar todas los posibles combinaciones de valores de los parámetros de las funciones. La evaluación de la función para una combinación de valores se puede considerar como un nuevo atributo, para el que se puede seleccionar el umbral. Esta aproximación no se sigue por el elevado número de combinaciones de valores posibles.

En el caso de predicados basados en intervalos la primera simplificación consiste en no considerar todos los posibles intervalos (que son $O(n^2)$ si n es la longitud de la serie), sino sólo aquellos cuya longitud sea potencia de 2 (que son $O(n \lg n)$). Por otro lado, se reutiliza el esfuerzo desarrollado para evaluar un predicado. Así, para evaluar un predicado cuyo intervalo sea de un tamaño $2i$, se tienen en cuenta los valores que se calcularon en su momento para evaluar los dos predicados de tamaño i consecutivos cuya unión forma el de tamaño $2i$. De este modo, el coste de evaluar un predicado no depende del tamaño del intervalo.

Para los predicados basados en disimilitud lo que se hace es restringir los ejemplos que se pueden utilizar como referencia a una muestra aleatoria de todos los ejemplos de entrenamiento.

3. Métodos de Clasificación

3.1. Boosting

Para poder discriminar entre distintas clases de series, en especial si el número de tipos es mayor que dos, es necesario combinar varios predicados: el mecanismo seleccionado para realizar esta combinación es el método denominado *boosting* [23], ya que una de sus propiedades es la posibilidad de obtener un clasificador fuerte a partir de un método de construcción de clasificadores débiles. En nuestro caso los predicados son los clasificadores débiles.

El funcionamiento de *boosting* se base en asignar un peso a cada ejemplo (inicialmente el mismo para todos). En cada iteración se construye un clasificador *base* (también denominado *débil*), teniendo en cuenta la distribución de pesos. Entonces los pesos de cada ejemplo se reajustan, en función de si el clasificador base lo clasificó o no correctamente. A la hora de clasificar, el resultado se obtiene mediante voto ponderado de los clasificadores base.

El cuadro 1 muestra un ejemplo de clasificador. Se corresponde con un problema de 3 clases. Está formado por 10 clasificadores base. La primera columna muestra el literal. Para cada clase hay otra columna, con el peso asociado al literal para esa clase.

Para clasificar un nuevo ejemplo, se calcula un peso para cada clase y se asigna al ejemplo la clase con mayor peso. Inicialmente el peso de cada clase es

Cuadro 1. Ejemplo de un clasificador obtenido mediante boosting, para un conjunto de datos con 3 clases. Por cada literal, se asocia un peso a cada clase.

Literal	Clase 1	Clase 2	Clase 3
desviación$_<$(E, x, 63, 126, 1.813266)	-0.399084	-0.421169	1.037954
desviación$_<$(E, x, 48, 111, 1.889214)	-0.232020	-0.153754	0.606268
media$_<$(E, x, 38, 101, 0.770881)	0.096480	0.072766	0.715047
¬media$_<$(E, x, 30, 33, 3.349725)	0.746371	-1.641331	0.664797
media$_<$(E, x, 55, 58, 4.280890)	-0.906306	0.352215	0.471577
desviación$_<$(E, x, 3, 34, 1.653625)	-0.334412	2.162192	-0.114016
¬media$_<$(E, x, 49, 52, 5.508630)	0.743638	0.517645	-0.195655
desviación$_<$(E, x, 25, 56, 1.107998)	0.634717	0.540074	0.000214
¬media$_<$(E, x, 44, 47, 3.720549)	0.484320	-0.932335	-0.155742
¬desviación$_<$(E, x, 26, 33, 3.355703)	-0.149486	0.090624	0.657721

0. Se evalúa cada literal. Si es cierto, el peso de cada clase se ve incrementado con el peso asociado al literal y la clase. Si es falso en vez de incrementar se decrementa.

3.2. Clasificación Temprana

Esta tarea consiste en cómo obtener una clasificación aproximada cuando no se dispone de series completas, sino de un fragmento inicial de las mismas. La necesidad de este tipo de clasificación surgió al intentar utilizar los clasificadores obtenidos en un sistema de diagnóstico. Si el proceso es crítico, es necesario que se dé la señal de alarma tan pronto como sea posible. La solución no es tan simple como entrenar el clasificador con series más cortas porque se han de utilizar series lo suficientemente largas como para que aparezcan los distintos comportamientos.

Para abordar este problema sólo se han considerado los predicados basados en intervalos, puesto que los de disimilitud necesitan de series completas. Si se intenta evaluar un predicado sobre una serie incompleta puede que el intervalo se refiera a valores ya conocidos, con lo que se puede evaluar a cierto o a falso. Si el intervalo se refiere a valores futuros, en general el valor del predicado es desconocido (aunque en ocasiones, por ejemplo con siempre y alguna-vez es posible saber el valor del predicado sin disponer de todos los valores en el intervalo).

Los clasificadores obtenidos con boosting son combinaciones de predicados. Si se excluyen de la combinación los predicados cuyo valor no se conoce, se sigue teniendo una combinación de (menos) predicados que se puede utilizar como clasificador, aunque lógicamente tendrá una menor precisión.

Dado que cuando se dispone de fragmentos cortos de series no va a ser posible obtener clasificadores precisos, se considera que la salida del clasificador no sea sólo una clase, sino una ordenación de las clases. De este modo, según se van conociendo más valores de las series se pueden ir descartando aquellas clases que ocupen las últimas posiciones.

3.3. Uso de Características Seleccionadas con Boosting

Se puede considerar que boosting realiza dos tareas. Por un lado permite obtener un conjunto de clasificadores diversos utilizando un único método de aprendizaje, mientras que por otro establece una forma de combinar dichos clasificadores. Sin embargo, es posible combinar los clasificadores de algún otro modo. En concreto, se pueden considerar los clasificadores base como nuevas características y utilizar otro método de aprendizaje sobre dichas características. Dado que la mayoría de los predicados calculan el valor de alguna función y comparan el resultado con un umbral, se puede considerar como característica el valor de la función en vez del valor del predicado.

En concreto, se exploran dos posibilidades. En primer lugar, árboles de decisión [16], con el objetivo de obtener clasificadores más comprensibles (normalmente a costa de la precisión) que los obtenidos con boosting. La figura 2 muestra un árbol de decisión con este tipo de características. En segundo, las Máquinas de Vectores Soporte, SVM [6], en este caso buscando una mejora en la precisión.

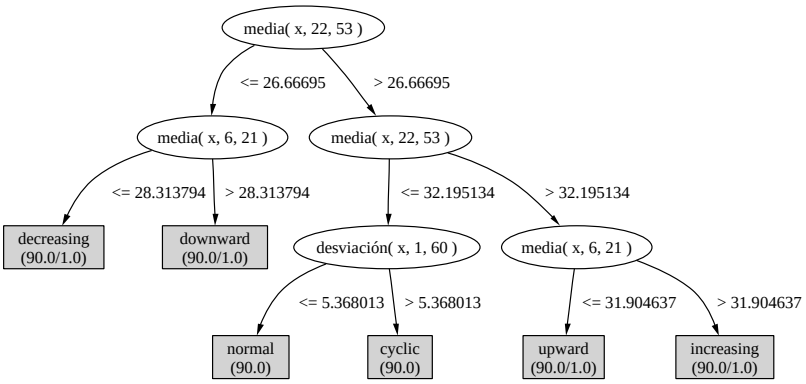

Figura 2. Árbol de decisión con las características media y desviación

3.4. Redes RBF

Tomando como partida el sistema de clasificación de series basado en boosting de literales sobre distancias, se propone un método alternativo para la construcción de redes de funciones de base radial, RBF [20,22]. Dado un literal, sobre un ejemplo podemos tener dos resultados, cierto o falso, que para el algoritmo de boosting se consideran +1 y -1. No obstante, el método de boosting puede trabajar con clasificadores base que devuelven valores reales, denominados grados de confianza. En este caso, la cantidad de información intercambiada entre el algoritmo de aprendizaje base y el algoritmo de boosting es mayor, puesto que

la salida del clasificador base no esta limitada a positivo o negativo, sino que es un número real que indica el grado de confianza.

El método seguido para obtener grados de confianza continuos a partir de literales consiste en transformar estos en funciones de base radial, RBF. Dado un literal disimilitud$_{\leq}$(Ejemplo, Referencia, Variable, Umbral), la función de base radial seleccionada es

$$h(x) = 2\exp\left(-\left(\frac{d_V(x,c)}{u}\right)^2 \ln 2\right) - 1$$

donde x es el Ejemplo, c el ejemplo de Referencia, u el Umbral y $d_V()$ es la función de disimilitud para la Variable V.

Boosting produce como resultado una combinación lineal de los clasificadores base. Si éstos son RBF, entonces la combinación es una Red RBF.

4. Aplicaciones

4.1. Conjuntos de Datos

El cuadro 2 muestra las características de los conjuntos de datos considerados. Para cada conjunto se muestra el número de clases, cuál es la longitud de las series, cuantas variables hay y de cuantos ejemplos se dispone. También se indica cuantos ejemplos se utilizan para entrenamiento y cuantos para test, ya que en algunos conjuntos esta división esta preestablecida. En otro caso se utiliza validación cruzada (abreviado VC en el cuadro), normalmente con 10 grupos, salvo para aquellos conjuntos de datos en los que la norma es utilizar otro número. Por último, se indica de qué tipo es el conjunto de datos. Los hay artificiales y reales. En este dominio es habitual el uso de conjuntos artificiales como banco de pruebas. También hay un conjunto de datos que se ha obtenido por simulación.

A continuación se describen brevemente algunos de los conjuntos de datos:

- *Disparos.* El dominio de este problema es el de la vigilancia por cámaras [15], ya que se trata de discriminar entre dos tipos de movimientos. Por un lado, sacar una pistola de la cartuchera, apuntar y guardar la pistola. Por otro, simular un disparo con el dedo. En ambos casos las posiciones iniciales y finales son las mismas. Los datos disponibles se corresponden a la posición del centroide de la mano derecha del actor.
- *Vocales Japonesas.* A pesar de su nombre es un problema de reconocimiento de locutor [12]. Se trata de discriminar entre 9 hablantes masculinos.
- *Dígitos.* Se trata de discriminar entre los distintos dígitos [1]. Los datos se obtuvieron utilizando una tableta digitalizadora. Los datos de entrenamiento y los de test se corresponden a dígitos escritos por distintas personas.
- *Auslan.* Es un conjunto de datos sobre reconocimiento de gestos pertenecientes al lenguaje utilizado por personas sordas [10]. Los datos se obtuvieron con un guantes de realidad virtual. Hay dos variantes, en las que se utilizaron distintos equipos. En la segunda los datos son de mayor calidad: hay un guante para cada mano en vez de solo uno, más medidas, mejor resolución...

Cuadro 2. Características de los conjuntos de datos.

	Clases	Longitud	Variables	Ejemplos	Entrenamiento / Validación	Tipo
CBF	3	128	1	798	VC 10	Artificial
CBF trasladado	3	128	1	5000	1000 / 4000	Artificial
Gráficas de Control	6	60	1	600	VC 10	Artificial
Ondas Triangulares	3	21	1	5000	300 / 5000	Artificial
Ondas con Ruido	3	40	1	5000	300 / 5000	Artificial
Dos patrones	4	128	1	5000	1000 / 4000	Artificial
Trace	16	[268...394]	4	1600	800 / 800	Artificial
Sonar	2	60	1	208	104 / 104	Real
Ionosfera 1	2	34	1	351	200 / 151	Real
Ionosfera 2	2	17	2	351	200 / 151	Real
Disparos	2	150	2	200	VC 10	Real
Vocales	11	10	1	990	528 / 462	Real
Vocales Japonesas	9	[7...29]	12	640	270 / 370	Real
Dígitos	10	8	2	10992	7494 / 3498	Real
Auslan / Nintendo	95	[17...149]	8	1900	VC 5	Real
Auslan / Flock	95	[45...136]	22	2565	VC 5	Real
Diagnosis	14	300	11	280	VC 10	Simulado

- *Diagnosis*. Se trata de identificar posibles fallos en un sistema dinámico de un proceso industrial [18], [2].

4.2. Experimentos

Para los conjuntos de datos anteriores se consideraron los siguientes experimentos:

- *Boosting.* Se consideraron los distintos tipos de predicados por separado, si bien algunos se agruparon por semejanza (incremento y decremento, siempre y alguna-vez, media y desviación). Se construyeron clasificadores formados por 100 predicados.
- *Clasificación Temprana.* En este caso sólo se consideraron dos combinaciones, por un lado valor-punto y por otro media y desviación.
- *Árboles.* Se utilizo el método de construcción de árboles basado en C4.5 disponible en WEKA [24]. Se consideraron las siguientes configuraciones: 1) los datos originales, 2) con las características media y desviación, 3) con la característica dtw y 4) con las tres características anteriores.
- *SVM.* Se consideraron kernels lineales y gaussianos. Como características se consideraron 1) los datos originales, 2) media y desviación y 3) dtw. Como herramientas se utilizaron tanto WEKA como LIBSVM [9].
- *Redes RBF.* En este caso se consideraron todos los predicados de disimilitud, comparando los resultados con los obtenidos cuando se utilizaba boosting con este tipo de predicados.

5. Conclusiones

La conclusión principal, a la luz de los resultados experimentales obtenidos, es que los métodos desarrollados son una alternativa válida para este tipo de problemas.

Con independencia de las características del conjunto de datos, se han considerado clasificadores formados por 100 predicados. En general se obtiene resultados competitivos con respecto a otras aproximaciones, siempre y cuando el número de clases sea moderado. Cuando el número de clases es grande, los resultados obtenidos con boosting pueden mejorar notablemente utilizando como método de combinación las SVM. Es posible que se pudieran mejorar los resultados de boosting con más predicados, pero dado que el tiempo necesario para obtener el clasificador y para clasificar depende del número de predicados, es importante obtener los mejores resultados con los predicados que se dispone.

Los buenos resultados no se deben tan solo a los métodos de boosting y SVM, como demuestran los resultados obtenidos en el primer caso por el predicado valor-punto y en el segundo utilizando los datos originales.

Los clasificadores obtenidos con boosting son adecuados para su uso en la tarea de clasificación temprana. En ciertas áreas de aplicación esta capacidad es imprescindible. Un aspecto destacable es que esta tarea se aborda utilizando los clasificadores de un modo distinto, y que no se tiene en cuenta para nada en el proceso de inducción. Por otro lado, cabe la posibilidad de que se pudieran mejorar los resultados en clasificación temprana si ésta influyera en el proceso de inducción.

Los clasificadores obtenidos con boosting no se consideran tan comprensibles como los obtenidos por otros métodos. Por eso, se ha incluido la posibilidad de utilizar los predicados seleccionados mediante boosting con árboles de decisión. Los resultados son favorables a esta aproximación frente a otros métodos que construyen árboles de decisión para series. Cuando se tienen muchas clases, los árboles pierden comprensibilidad. En este caso se puede construir un árbol para cada clase.

No obstante, hay que indicar que los clasificadores obtenidos con boosting no son cajas negras y la dificultad de su comprensión se debe principalmente a su tamaño. Para aliviar este problema, siempre y cuando sea suficiente con hacerse una idea de qué es lo que hace el clasificador, se puede: considerar cada clase por separado, considerar sólo los primeros predicados (ya que también forman un clasificador), y/o considerar sólo los predicados de más peso.

Se ha propuesto un método novedoso para la construcción de redes RBF, basado en convertir los predicados de disimilitud en RBF y en boosting. Normalmente los métodos para construir este tipo de redes funcionan en dos fases. En la primera se seleccionan los centros y los radios, mientras que en la segunda se ajustan los pesos. Por el contrario, el método propuesto es incremental, ya que en cada iteración se selecciona el centro y el radio de una RBF y se calculan sus pesos. Una de las características destacables del método es que se puede utilizar con cualquier función de disimilitud, sea o no una distancia. Esto es de particular relevancia cuando se trabaja con series porque permite utilizar

DTW. Los resultados obtenidos con este método para obtener redes RBF no son sistemáticamente mejores que los obtenidos con boosting y los predicados de disimilitud correspondientes, pero para varios conjuntos de datos las mejoras son importantes.

Referencias

1. Fevzi Alimoglu and Ethem Alpaydin. Methods of combining multiple classifiers based on different representations for pen-based handwriting recognition. In *5th Turkish Artificial Intelligence and Artificial Neural Networks Symposium (TAINN 96)*, 1996. `http://www.cmpe.boun.edu.tr/ alimoglu/tainn96.ps.gz`.
2. Carlos Alonso, Juan J. Rodríguez, and Belarmino Pulido. Enhancing consistency based diagnosis with machine learning techniques. In *Conference of the Spanish Association for Artificial Intelligence (CAEPIA'2003)*, Lecture Notes in Artificial Intelligence. Springer, 2004.
3. Carlos J. Alonso González and Juan J. Rodríguez Diez. A graphical rule language for continuous dynamic systems. In Masoud Mohammadian, editor, *Computational Intelligence for Modelling, Control and Automation*, volume 55 of *Concurrent Systems Engineering Series*, pages 482–487, Amsterdam, Netherlands, 1999. IOS Press.
4. Carlos J. Alonso González and Juan J. Rodríguez Diez. Time series classification by boosting interval based literals. *Revista Iberoamericana de Inteligencia Artificial*, 11:2–11, 2000.
5. D.J. Berndt and J. Clifford. Finding patterns in time series: a dynamic programming approach. In U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, pages 229–248. AAAI Press /MIT Press, 1996.
6. Nello Cristianini and John Shawe-Taylor. *An Introduction to Support Vector Machines and other kernel-based learning methods*. Cambridge University Press, 2000.
7. Pierre Geurts. *Contributions to decision tree induction: bias/variance tradeoff and time series classification*. PhD thesis, Department of Electrical Engineering and Computer Science, University of Liège, Belgium, 2002. `http://www.montefiore.ulg.ac.be/ geurts/thesis.html`.
8. Ernst Haselsteiner. *Time Series Classification Using Adaptive Dynamic Targets*. PhD thesis, Technischen Universitat Gratz, 2000. `http://citeseer.nj.nec.com/405214.html`.
9. Chih-Wei Hsu, Chih-Chung Chang, and Chih-Jen Lin. A practical guide to support vector classification. `http://www.csie.ntu.edu.tw/ cjlin/papers/guide/guide.pdf`.
10. Mohammed Waleed Kadous. *Temporal Classification: Extending the Classification Paradigm to Multivariate Time Series*. PhD thesis, The University of New South Wales, School of Computer Science and Engineering, 2002. `http://www.cse.unsw.edu.au/ waleed/phd/`.
11. Eamonn Keogh. Exact indexing of dynamic time warping. In *28^{th} International Conference on Very Large Data Bases*, 2002.
12. Mineichi Kudo, Jun Toyama, and Masaru Shimbo. Multidimensional curve classification using passing-through regions. *Pattern Recognition Letters*, 20(11–13):1103–1111, 1999. `http://ips6.main.eng.hokudai.ac.jp/research/pattern/paper/mine_prpVI.ps`.

13. Robert T. Olszewski. *Generalized Feature Extraction for Structural Pattern Recognition in Time-Series Data*. PhD thesis, Computer Science Department, Carnegie Mellon University, 2001. `http://reports-archive.adm.cs.cmu.edu/anon/2001/abstracts/01-108.html`.
14. L. Rabiner and B. Juang. An introduction to hidden markov models. *IEEE Magazine on Accoustics, Speech and Signal Processing*, 3(1):4–16, 1986.
15. Chotirat Ann Ratanamahatana and Eamonn Keogh. Making time-series classification more accurate using learned constraints. In *SIAM International Conference on Data Mining (SDM'04)*, 2004.
16. Juan J. Rodríguez and Carlos J. Alonso. Interval and dynamic time warping-based decision trees. In *19th Annual ACM Symposium on Applied Computing, Special Track on Data Mining*, 2004.
17. Juan J. Rodríguez, Carlos J. Alonso, and Henrik Boström. Learning first order logic time series classifiers: Rules and boosting. In Djamel A. Zighed, Jan Komorowski, and Jan Zytkow, editors, *Principles of Data Mining and Knowledge Discovery: 4^{th} European Conference; PKDD 2000*, volume 1910 of *Lecture Notes in Artificial Intelligence*, pages 299–308, Lyon, France, September 2000. Springer.
18. Juan J. Rodríguez, Carlos J. Alonso, and Henrik Boström. Boosting interval based literals. *Intelligent Data Analysis*, 5(3):245–262, 2001.
19. Juan J. Rodríguez Diez and Carlos J. Alonso González. Applying boosting to similarity literals for time series classification. In *Proceedings of First International Workshop on Multiple Classifier Systems, MCS2000*, Lecture Notes in Computer Science. Springer-Verlag, 2000.
20. Juan J. Rodríguez Diez and Carlos J. Alonso González. Learning classification RBF networks by boosting. In Josef Kittler and Fabio Roli, editors, *Multiple Classifier Systems: second international workshop, MCS 2001*, Lecture Notes in Computer Science, pages 43–52, Cambridge, UK, July 2001. Springer.
21. Juan José Rodríguez Diez. *Técnicas de Aprendizaje Automático para la Clasificación de Series*. PhD thesis, Departemento de Informática. Universidad de Valladolid, 2004.
22. Juan J. Rodríguez Diez and Carlos J. Alonso González. Building RBF networks for time series classification by boosting. In Dechang Chen and Xiuzhen Cheng, editors, *Pattern Recognition and String Matching*, volume 13 of *Combinatorial Optimization*. Kluwer, 2002.
23. Robert E. Schapire. The boosting approach to machine learning: An overview. In *MSRI Workshop on Nonlinear Estimation and Classification*, 2002. `http://www.cs.princeton.edu/ schapire/papers/msri.ps.gz`.
24. Ian H. Witten and Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann, 1999.