Técnicas de Aprendizaje para la Identificación de Fallos en Sistemas Dinámicos*

Carlos J. Alonso[1] , Juan José Rodríguez[2] , Belarmino Pulido[1] y Oscar Prieto[1]

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

Resumen En este trabajo se presenta una propuesta para la diagnosis de sistemas dinámicos complejos utilizando técnicas de aprendizaje automático. La diagnosis se considera como un problema de clasificación de series que se aborda con métodos desarrollados en el grupo de investigación. Estos métodos se basan en utilizar predicados que capturan propiedades interesantes de las series y la combinación de estos predicados mediante boosting. Se incluyen los resultados obtenidos sobre una planta piloto en la que se dispone de 11 variables observables y se consideran 14 modos de fallo distintos.

1. Introducción

La diagnosis de sistemas dinámicos complejos es un tema de investigación abierto, para el que se han usado una gran variedad de técnicas [2]. Los problemas de más complejidad aparecen en sistemas dinámicos, en los que hay un gran número de componentes, un conjunto pequeño de variables observables (en comparación con las variables presentes en el sistema), con modelos que no se conocen bien y que son difíciles de estimar, con interacciones entre componentes que tampoco se conocen con exactitud y donde la presencia de sistemas de control puede ocultar la presencia de fallos. Dentro de este tipo de problemas, los procesos continuos industriales son un ejemplo típico.

La identificación de fallos se puede considerar como un problema de reconocimiento de patrones, donde éstos son series de los valores históricos de las variables observables del sistema. En este trabajo se presenta una propuesta para abordar este problema utilizando técnicas de aprendizaje para la clasificación de series. Dichas técnicas están basadas en la definición de un conjunto de predicados que capturan características interesantes de las series, y en el uso del método boosting para combinar estos predicados. Otros trabajos que utilizan técnicas de aprendizaje automático para la diagnosis de sistemas dinámicos son [4,14,12,13,10].

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

En el caso de estudio las técnicas de aprendizaje actuarán en combinación con un sistema de diagnóstico basado en modelos. En concreto, el sistema realiza la detección y localización de fallos mediante técnicas de diagnóstico basado en consistencia apoyándose en posibles conflictos [7]. Este tipo de técnicas requieren de ayuda adicional para abordar la identificación, una vez que se ha producido la detección. Nuestra propuesta [1] es utilizar clasificadores que establezcan en línea un ránking de los modos de fallo más probables en función de los datos disponibles hasta el momento. Hay que tener en cuenta que este hecho complica significativamente la tarea, al no disponer de la descripción completa de un modo de fallo para su identificación, dado que sólo dispondremos de los síntomas que se hayan manifestado hasta el momento actual.

El resto del artículo se organiza como sigue. La sección 2 presenta la planta considerada. Una propuesta para abordar este tipo de problemas mediante técnicas de aprendizaje supervisado se presenta en la sección 3, incluyendo los resultados experimentales para la planta. La sección 4 se dedica a la *clasificación temprana*, que considera el problema de cómo intentar clasificar la situación actual cuando dicha clasificación depende de valores que se van a generar en el futuro. Finalmente, la sección 5 presenta las conclusiones más relevantes.

2. La Planta

La figura 1 muestra un esquema de la planta piloto considerada. Está formada por 4 tanques $\{T_1 \dots T_4\}$, 5 bombas $\{P_1 \dots P_5\}$ y dos controladores PID que actúan sobre las bombas P_1, P_5 para mantener los niveles de los tanques $\{T_1, T_4\}$ cercanos a los valores especificados por el punto de operación. Para calentar el fluido de los tanques $\{T_2, T_3\}$ se usan dos resistencias $\{R_2, R_3\}$.

Se dispone de las siguientes medidas:

- Niveles de los tanques T_1 y T_4: LT1, LT4.
- Valores de los controladores PID de las bombas P_1 y P_5: PI.1.control, PI.4.control.
- Flujo de entrada al tanque T_1: FT1.
- Flujo de salida de los tanques T_2, T_3 y T_4: FT2, FT3, FT4.
- Temperatura en los tanques T_2, T_3 y T_4: TT2, TT3, TT4.

Es decir, hay 11 variables distintas.

2.1. Modos de Fallo

Se consideran los siguientes modos de fallo:

- Fugas en los tanque: 5 modos. Se consideran los 4 tanques, pero en el tanque $T1$ se consideran como modos de fallo distintos las fugas pequeñas y las grandes.
- Bloqueos en tuberías: 5 modos. Se consideran bloqueos en las dos salidas de $T1$, en la salida derecha de $T3$ y $T4$ y en la salida izquierda de $T2$.

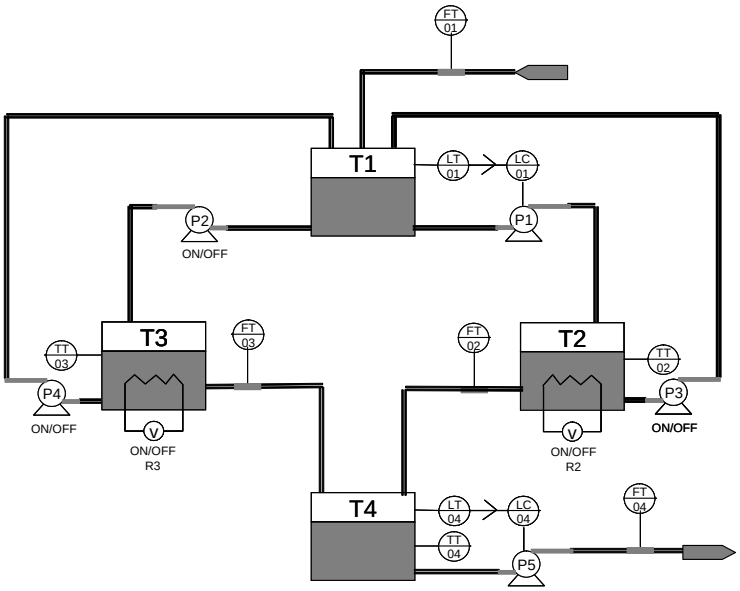

Figura 1. Esquema de la planta piloto.

- Fallos en las bombas: 3 modos. Se corresponden con las bombas $P1$, $P2$ y $P5$.
- Fallo en resistencia: 1 modo. Para la resistencia $R2$.

Lo que hace un total de 14 modos de fallo distintos. El cuadro 1 muestra cuáles son estos modos de fallo.

Para cada modo de fallo se realizaron 20 simulaciones. En cada una de ellas se simula el proceso durante 15 minutos de tiempo real. El periodo de muestreo es de 1 segundo, por lo que la longitud de las series es de 900. Dado que una resolución tan fina no es necesaria y que el tiempo de los métodos de inducción depende de la longitud de la serie se comprimieron las series a una longitud de 300.

La figura 2 muestra un ejemplo de cada modo de fallo. En cada fila se muestra un ejemplo de un modo de fallo distinto. En cada columna se muestra una de de las variables. Todas las gráficas de una mismo tipo de magnitud (e.g., temperaturas) se muestran a la misma escala.

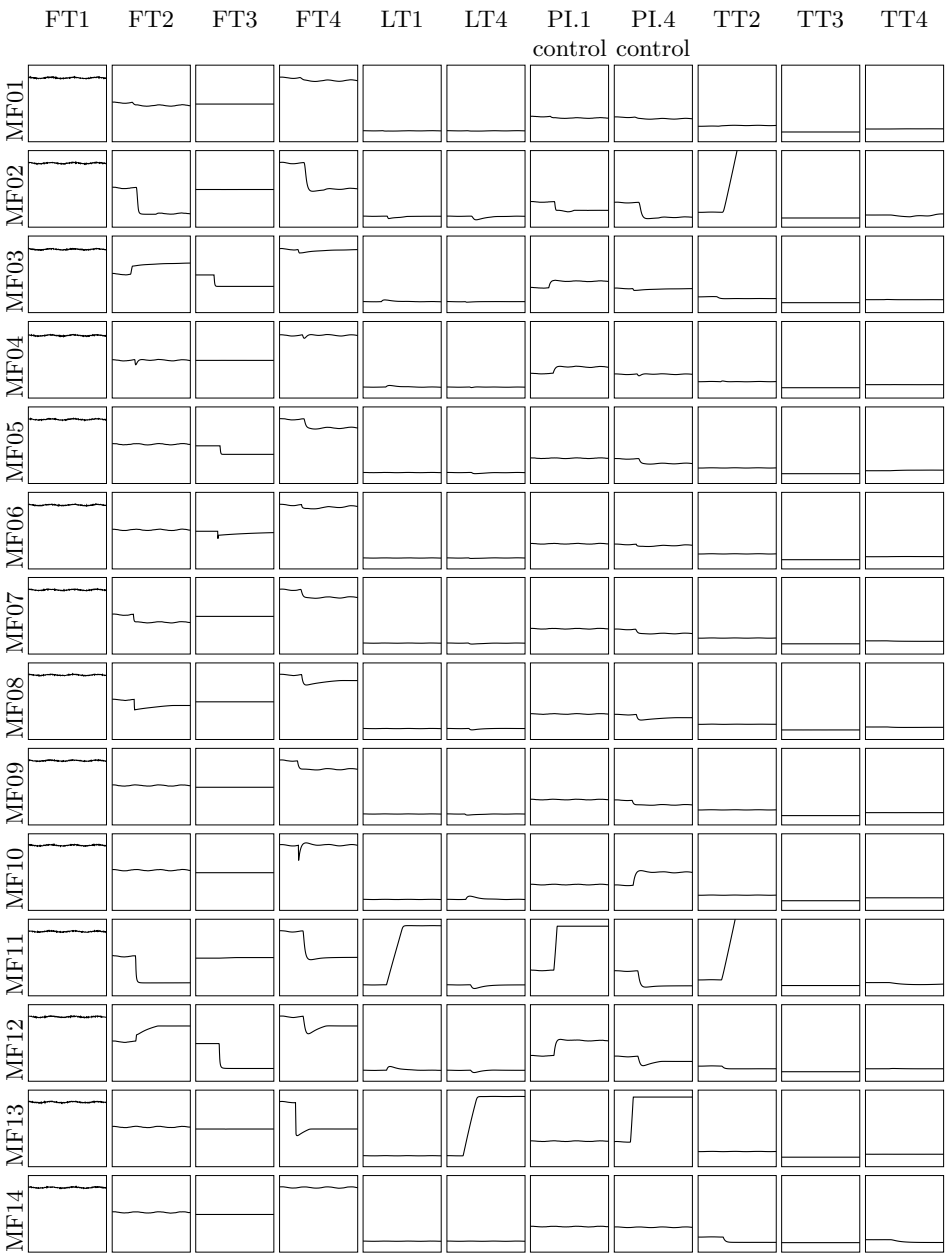

Figura 2. Ejemplos de los modos de fallo. En cada fila se muestra un ejemplo de un modo de fallo distinto. En cada columna se muestra una de de las variables.

Cuadro 1. Modos de fallo considerados.

Identificador	Componente	Tipo
MF01	T1	Fuga pequeña en T1
MF02	T1	Fuga grande en T1
MF03	T1	Bloqueo tubería T1 (salida izquierda)
MF04	T1	Bloqueo tubería T1 (salida derecha)
MF05	T3	Fuga en T3
MF06	T3	Bloqueo tubería T3 (salida derecha)
MF07	T2	Fuga en T2
MF08	T2	Bloqueo tubería T2 (salida izquierda)
MF09	T4	Fuga en T4
MF10	T4	Bloqueo tubería T4 (salida derecha)
MF11	P1	Rotura / pérdida de rendimiento
MF12	P2	Rotura / pérdida de rendimiento
MF13	P5	Rotura / pérdida de rendimiento
MF14	R2	Rotura resistencia en tanque T2

3. Aprendizaje Automático para la Identificación de Fallos

3.1. Descripción del Método

El método propuesto para obtener clasificadores para este tipo de datos se basa en definir una serie de predicados que capturen propiedades interesantes de las series. La figura 3 muestra los distintos tipos de predicados considerados.

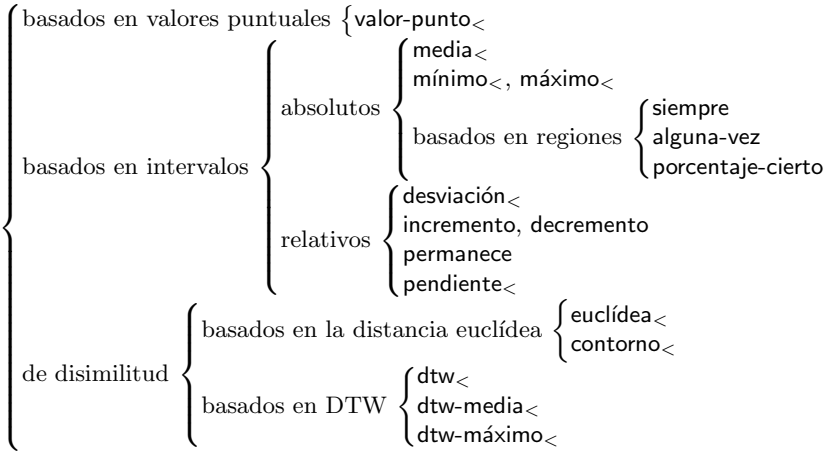

Figura 3. Tipos de predicados

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., la temperatura TT4) 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 TT4[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) [6].

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.

Por cuestiones de espacio no es posible describir como se seleccionan los predicados. Este proceso se detalla en [8] para los predicados basados en intervalos y en [9] para los basados en disimilitud.

Boosting. Para poder discriminar entre distintas clases de series, en especial si el número de tipos es mayor que 2, es necesario combinar varios predicados: El mecanismo seleccionado para realizar esta combinación el método denominado *boosting* [11], ya que una de sus propiedades es la posibilidad de obtener un clasificador fuerte a partir de un método de obtención de clasificadores débiles. En nuestro caso los predicados son los clasificadores débiles.

3.2. Resultados con Boosting

Se utilizó validación cruzada estratificada con 10 grupos. Los clasificadores están formados por 100 predicados.

El cuadro 2 y la figura 4 muestra los resultados obtenidos utilizando boosting de literales. Se consideran los distintos tipos de predicados por separado, aunque hay algunos tipos que se agrupan por afinidad (mínimo con máximo, incremento con decremento y siempre con alguna-vez). Por otro lado, media y desviación también se agrupan debido a que para calcular el segundo valor es necesario calcular el primero.

Los mejores resultados se corresponden con dtw y sus variantes. El siguiente mejor resultado es el obtenido por los predicados media y desviación.

Cuadro 2. Resultados obtenidos utilizando boosting de literales. Se muestra el error en función del número de literales.

	10	20	30	40	50	60	70	80	90	100
valor-punto	30.36	13.21	8.57	4.64	5.71	5.71	4.29	3.93	3.21	3.57
media, desv.	27.14	13.93	10.00	7.50	5.00	2.50	2.50	2.50	2.50	2.50
mín., máximo	26.43	9.64	7.50	6.43	3.93	2.14	3.57	3.21	2.86	2.86
siempre, alg.	37.86	17.50	10.71	7.50	6.07	6.07	5.71	5.71	5.36	5.71
porc.-cierto	36.43	14.64	10.36	8.57	7.14	6.07	5.36	5.71	5.36	5.00
incr. y decr.	28.21	11.79	9.29	6.07	6.07	5.00	4.29	3.93	3.57	3.93
permanece	40.00	18.93	15.36	11.79	11.79	7.14	7.50	7.14	7.14	6.79
pendiente	26.07	7.14	3.93	3.21	3.21	2.86	2.86	3.21	2.50	2.86
euclídea	37.50	15.00	8.93	5.71	3.21	2.86	2.86	2.14	2.86	2.86
contorno	35.00	13.93	10.00	4.29	5.00	3.93	3.93	2.86	2.50	2.86
dtw	36.79	12.14	5.00	4.29	3.21	2.14	2.14	2.14	2.50	1.79
dtw-media	38.57	16.79	6.43	4.29	4.29	2.14	2.86	3.21	2.50	2.50
dtw-mín-máx	37.14	10.00	6.07	5.36	3.93	3.21	3.21	2.50	2.14	1.79
media, desv., pendiente	27.14	7.86	3.93	2.86	2.86	2.50	2.14	2.14	2.14	1.79
porc.-cierto, incr., decr., permanece	32.86	10.36	8.57	3.57	3.21	3.21	3.21	3.21	3.21	2.86

Combinaciones de Predicados. Para intentar mejorar los resultados obtenidos con este tipo de predicados se pueden considerar combinaciones de predicados. En concreto, las combinaciones aquí consideradas son:

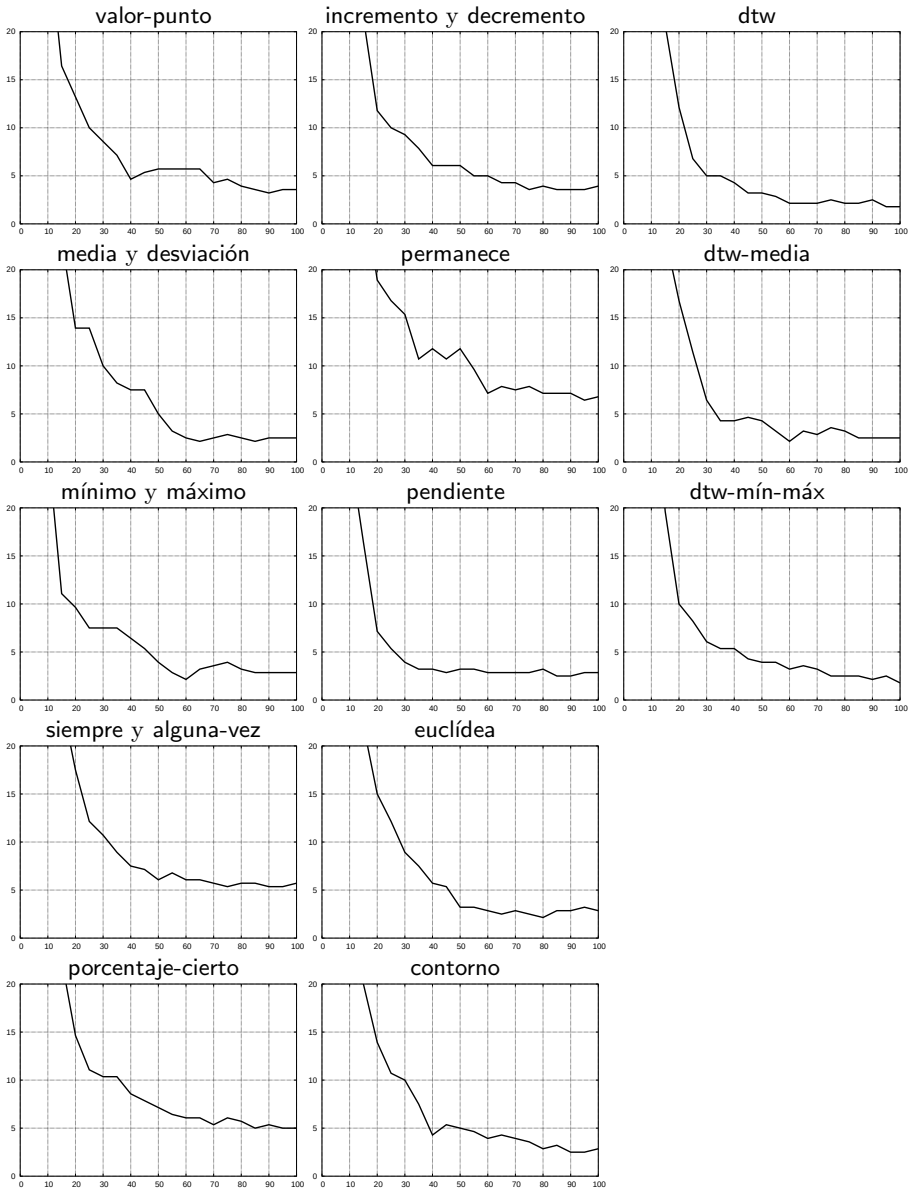

Figura 4. Resultados obtenidos utilizando boosting de literales. Se muestra el error en función del número de literales.

- media, desviación y pendiente.
- porcentaje-cierto, incremento, decremento y permanece.

En cada combinación se consideran predicados absolutos y relativos, para intentar capturar distintos tipos de características.

Al final del cuadro 2 y en la figura 5 se muestran los resultados con estas combinaciones de clasificadores. Los resultados con las combinaciones son mejores (aunque las diferencias son leves) que los obtenidos con los distintos tipos de predicados por separado.

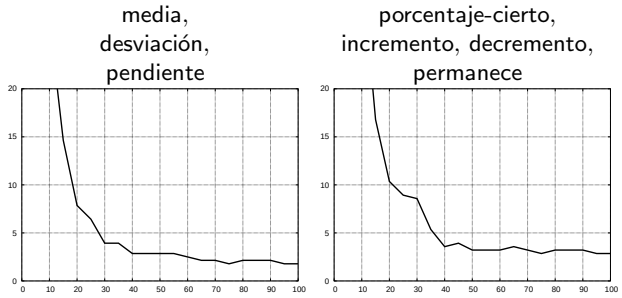

Figura 5. Resultados obtenidos utilizando boosting con varios tipos de literales simultáneamente para el conjunto de datos *Diagnosis*.

3.3. Resultados con Otros Métodos

Con el fin de tener alguna indicación respecto de la dificultad del problema, se incluyen algunos resultados para este conjunto de datos obtenidos con otros métodos. Salvo que se indique expresamente lo contrario, estos resultados se obtuvieron utilizando WEKA [15]:

- 16 % con un clasificador bayesiano *naïve*.
- 15 % utilizando la técnica del vecino más cercano.
- 6.07 % utilizando J48, la versión del método de inducción de árboles de decisión C4.5 disponible en WEKA.
- 7.86 % utilizando Máquinas de Vectores Soporte (SVM) [3] con kernel lineal. Se utilizaron los valores por defecto de los distintos parámetros.
- 6.79 % utilizando SVM con kernel gaussiano. En este caso la implementación utilizada fue LIBSVM [5]. Esta herramienta incorpora un método, basado en validación cruzada, para seleccionar los valores de los dos parámetros más importantes.

4. Clasificación Temprana

4.1. Descripción de la Tarea

La tarea que denominamos "clasificación temprana" consiste en clasificar series incompletas, ya que hay valores de las mismas que no se conocen porque se van a generar en el futuro. Supongamos que se quiere evaluar si se ha producido un fallo hace 5 minutos. Como los clasificadores se han entrenado con series de 15 minutos, hasta dentro de 10 minutos no se dispondrá de series de longitud suficiente. Sin embargo, es útil intentar identificar el fallo tan pronto como sea posible. Al intentar clasificar con series incompletas la precisión va a ser indudablemente peor que cuando se dispone de series completas, pero la clasificación temprana es útil en tanto en cuanto según se van conociendo valores se pueden ir descartando fallos.

La técnica utilizada para abordar esta tarea se basa en primer lugar en considerar que la evaluación de un predicado puede dar lugar, además de los valores cierto y falso, al resultado "desconocido". Esta aproximación es válida para predicados basados en intervalos, pero no para los de disimilitud. Si un intervalo de un predicado se refiere a valores ya conocidos de la serie entonces se podrá evaluar a cierto o falso, mientras que si se refiere a valores todavía desconocidos su evaluación dará como resultado el valor desconocido.

En segundo lugar, los clasificadores están formados por combinaciones de predicados. Si de dicha combinación se eliminan aquellos predicados cuya evaluación produce el resultado desconocido, se sigue teniendo una combinación de (menos) predicados, que se puede seguir utilizando como clasificador.

El método de boosting al clasificar no produce como resultado sólo cuál es la clase asignada por el clasificador sino que a cada clase le asigna un valor de confianza. Estas confianzas se pueden utilizar para ordenar las clases. Esta ordenación es útil en clasificación temprana porque aunque el clasificador no sea capaz de determinar exactamente la clase correcta, si es capaz de colocarla entre las primeras, esa información es valiosa.

Error de Posición. Sea un clasificador que dispone de varios intentos para clasificar un ejemplo. Si un clasificador es capaz de identificar la clase al segundo intento y otro sólo identifica la clase en el último intento, no parece que sean igual de buenos. Sin embargo, bajo el criterio habitual el error de ambos es del 100 %. Para intentar capturar esta diferencia se define el *error de posición*, que para un ejemplo es número de clases que aparecen por delante de la correcta en la salida del clasificador, divido por el número de clases menos uno. Así, el error de posición es un valor entre 0 y 1 (si se multiplica por 100 se puede considerar un porcentaje). Es 0 si la clase correcta es la primera y 1 si es la última. El error de posición de un conjunto de datos es la media de los errores de posición de los ejemplos en ese conjunto de datos. Para evitar confusiones, a lo que se suele denominar error lo denominaremos *error de clasificación.*

4.2. Resultados

La figura 6 muestra los resultados obtenidos para clasificación temprana con distintas combinaciones de predicados. Si bien el uso de la media, desviación y pendiente proporcionaba los mejores resultados cuando se clasificaban series completas, los resultados con estos predicados en clasificación temprana no son mejores que los obtenidos con valor-punto. No obstante, sí que se mejoran los resultados de este predicado con los predicados porcentaje-cierto, incremento, decremento y permanece.

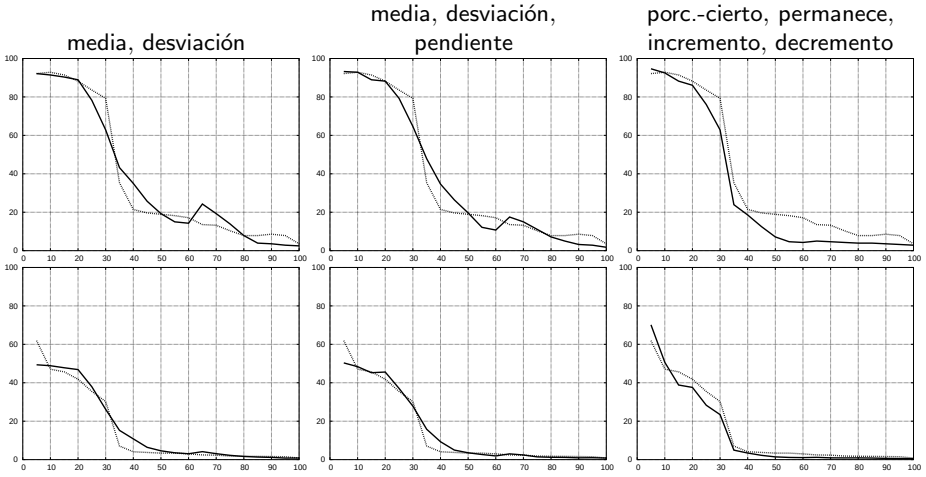

Figura 6. Resultados con clasificación temprana para el conjunto de datos *Diagnosis*. Las gráficas superiores muestran el error de clasificación mientras que las inferiores muestran el de posición, en ambos casos en función del porcentaje disponible de la serie. En línea discontinua se muestra, en todas las gráficas, los resultados con valor-punto.

5. Conclusiones

En este trabajo se ha presentado una aplicación del Aprendizaje Automático, la identificación de fallos en la diagnosis de sistemas dinámicos en procesos continuos industriales. En concreto, se ha considerado una planta piloto en la que se dispone de 10 variables y en la que hay 14 modos de fallo distintos. Los resultados demuestran la bondad de los métodos desarrollados frente a un conjunto de técnicas estándar y da idea de la complejidad del problema. Esta complejidad (que viene dada por colecciones de datos temporales, con ruido y distintos puntos de ocurrencia del fallo) se acrecienta si se quiere hacer algo más que diagnosis *post-mortem*. Cuando se requiere, como es el caso, combinar con

otro sistema que haga la detección y localización, y proponer un conjunto de fallos más probables en línea, es necesario abordar la clasificación temprana.

Referencias

1. Carlos Alonso, Juan J. Rodríguez, and Belarmino Pulido. Enhancing consistency based diagnosis with machine learning techniques. In *Current Topics in Artificial Intelligence: 10th Conference of the Spanish Association for Artificial Intelligence*, volume 3040 of *Lecture Notes in Artificial Intelligence*, pages 312–321. Springer, 2004.
2. Karthik Balakrishnan and Vasant Honavar. Intelligent diagnosis systems. *Journal of Intelligent Systems*, 8, 1998.
3. Nello Cristianini and John Shawe-Taylor. *An Introduction to Support Vector Machines and other kernel-based learning methods*. Cambridge University Press, 2000.
4. C. Feng. Inducting temporal fault diagnostic rules from a qualitative model. In S. Muggleton, editor, *Inductive Logic Programming*. Academic Press, 1992.
5. 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`.
6. Eamonn Keogh. Exact indexing of dynamic time warping. In *28^{th} International Conference on Very Large Data Bases*, 2002.
7. B. Pulido and C. Alonso GonzÃ¡lez. Possible conflicts: a compilation technique for consistency-based diagnosis. *IEEE Transactions On Systems, Man, and Cybernetics, Part B. Special Issue on Diagnosis of Complex Systems: bridging the methodologies of the FDI and DX communities*, in press, 2004.
8. Juan J. Rodríguez, Carlos J. Alonso, and Henrik Boström. Boosting interval based literals. *Intelligent Data Analysis*, 5(3):245–262, 2001.
9. 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.
10. Davide Roverso. Fault diagnosis with the aladdin transient classifier. In *System Diagnosis and Prognosis: Security and Condition Monitoring Issues III, AeroSense2003, Aerospace and Defense Sensing and Control Technologies Symposium*, 2003. `http://www.ife.no/media/1167_Aladdin_AeroSense_OR14.pdf`.
11. 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`.
12. Derek Sleeman, F. Mitchell, and R. Milne. Applying KDD techniques to produce diagnostic rules for dynamic systems. Technical Report AUCS/TR9604, Department of Computing Science. University of Aberdeen, 1996. `ftp://ftp.csd.abdn.ac.uk/pub/reports/tr9604.ps`.
13. Antonio J. Suárez, Pedro J. Abad, J.A. Ortega, and Rafael M. Gasca. Diagnosis progresiva en el tiempo de sistemas dinámicos. In *IV Jornadas de ARCA, Sistemas Cualitativos y Diagnosis, JARCA'02*, 2002.
14. V. Venkatusugramanian and K. Chan. A neural network methodology for process fault diagnosis. *AIChE J.*, 35:1993–2001, 1995.
15. Ian H. Witten and Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann, 1999.