Informática
XVI CONVENCIÓN Y FERIA INTERNACIONAL 2016
16TH INTERNATIONAL CONVENTION AND FAIR
Conectando sociedades
Connecting societies
La Habana, Cuba
Del 14 al 18 de marzo
Havana, Cuba
March 14th to 18th

PAQUETE PARA LA CLASIFICACIÓN DE SERIES TEMPORALES EN WEKA

A PACKAGE FOR TIME SERIES CLASSIFICATION IN WEKA

César Soto Valero[1], Mabel González Castellanos[2]

[1] Universidad Central «Marta Abreu» de Las Villas, Cuba, cesarvalero@uclv.cu, Carretera a Camajuaní km 5 ½. CP: 54830

[2] Universidad Central «Marta Abreu» de Las Villas, Cuba, mabelc@uclv.edu.cu

RESUMEN:

Las series temporales posibilitan la descripción de una gran variedad de fenómenos que transcurren a lo largo del tiempo. Los métodos que realizan análisis de series temporales usando técnicas de minería de datos son capaces de resolver múltiples problemas, superando muchas de las limitaciones presentes en los modelos estadísticos y matemáticos usados tradicionalmente en este campo. Weka es un poderoso sistema de aprendizaje automatizado, sin embargo ofrece muy pocas posibilidades para el manejo de series temporales. En el presente trabajo se describen el diseño e implementación de un paquete para Weka con herramientas desarrolladas especialmente para la clasificación de este tipo de datos. Dichas herramientas son las siguientes: una función de distancia basada en la medida de similitud DTW, un algoritmo para la búsqueda de vecinos cercanos el cual toma ventaja de la cota inferior de Keogh durante la clasificación usando 1-NN y un filtro para la reducción de la numerosidad de series temporales. Los resultados experimentales obtenidos con el uso de estas herramientas en varios conjuntos de datos tradicionales demuestran la validez y utilidad del paquete implementado.

Palabras Clave: Series temporales, clasificación, paquete, DTW, Weka, k-NN.

ABSTRACT:

Time series makes possible the description of a wide variety of phenomena that take place over the time. The methods that perform time series analysis using data mining techniques solve many problems, overcoming many of the limitations present in statistical and mathematical models traditionally used in this field. Weka is a powerful machine learning framework, however, it lacks of tools to manage time series. In this paper we describe the design and implementation of a package for Weka with tools specially developed for the classification of time series data. Those tools are: a distance function based on the similarity measure DTW, an algorithm for finding nearest neighbors for classification using 1-NN, which takes advantage of the Keogh's lower bound, and a filter for numerosity reduction of time series data. The experimental results obtained with the use of these tools in several traditional time series datasets show the validity and usefulness of the developed package.

KeyWords: Time series, classification, package, DTW, Weka, k-NN.

1. INTRODUCCIÓN

En la vida real, la mayoría de los fenómenos que se estudian secuencialmente ordenados en el tiempo deben tomar en cuenta la compleja dinámica de cada proceso en específico, con la finalidad de entenderlo de la mejor manera posible. Una herramienta útil para alcanzar dicho objetivo es el análisis de series temporales.

Una serie temporal se define como una secuencia de n observaciones o datos x_t ordenadas cronológicamente, sobre una característica (serie univariable) o sobre varias características (serie multivariable) de una unidad observable, tomadas en diferentes momentos. Las series temporales se caracterizan fundamentalmente por la gran numerosidad de los datos que la conforman, la alta dimensionalidad y la necesidad de su constante actualización [1].

Las series temporales se estudian principalmente con el objetivo de extraer información de algún fenómeno del pasado e intentar predecir el futuro, lo cual permite descubrir características en los datos y determinar su variación a largo plazo. El aumento de la potencia de cómputo en la actualidad ha extendido la aplicación de las series temporales en muchos dominios. Por ejemplo, en economía se utilizan estas series en el control de la calidad, para estudiar índices de precios en el mercado, desempleo, producto interno bruto (PIB), índices poblacionales entre otros. Un estudio económico que muestra la correlación causal entre el consumo eléctrico y la producción económica en Australia usando series temporales se puede consultar en [2].

La minería de datos para series temporales es una contribución importante a los campos de estudio de la minería de datos y del análisis de series temporales. Los métodos utilizados en la minería de datos para series temporales son capaces de caracterizar satisfactoriamente series periódicas, no periódicas, complejas y caóticas. Estos métodos cubren las limitaciones de las técnicas estadísticas y matemáticas utilizadas tradicionalmente en su análisis (por ejemplo, el modelo ARIMA clásico [3, 4]) ya que adaptan las técnicas de la minería de datos para tratar las series como una clase especial de datos.

En los últimos años se han llevado a cabo numerosas investigaciones relacionadas con la minería de datos para series temporales, por ejemplo: la búsqueda de similitudes [1, 5], la identificación de subsecuencias [6], la reducción de su dimensionalidad [7, 8] y la segmentación [9]. En [1], con el propósito de facilitar su estudio, se determinan las siguientes tareas fundamentales de la minería de datos para series temporales: representación e indexado, clasificación, medidas de similitud, emparejamiento de subsecuencias, segmentación, visualización y descubrimiento de patrones y conglomerados.

Weka[1] es actualmente una de las plataformas para la minería de datos más populares y cuenta con un paquete dedicado específicamente a la predicción de series temporales mediante técnicas de regresión [10]. No obstante, las funcionalidades que brinda este paquete resultan insuficientes para enfrentar otras tareas relacionadas con las series temporales. En la literatura se halla un número considerable de propuestas que presentan métodos para la clasificación de series temporales utilizando algoritmos del aprendizaje automatizado. Es por esto que contrasta el hecho de que Weka aún no posea alguna herramienta con esta finalidad.

El objetivo de este trabajo consiste en implementar un paquete para Weka que brinde las siguientes facilidades para la clasificación de series temporales:

- Cálculo de la distancia entre series utilizando una medida de similitud elástica.
- Clasificación de series temporales mejorando el desempeño del algoritmo de búsqueda de vecinos más cercanos.
- Reducción de la numerosidad presente en conjuntos de datos con gran cantidad de instancias de entrenamiento.

2. CLASIFICACIÓN DE SERIES TEMPORALES

La clasificación es quizás la técnica más popular de la minería de datos. Dicha técnica asocia datos entre grupos predefinidos o clases. La mayoría de los algoritmos de clasificación asumen algún conocimiento de los datos o realizan fases de entrenamiento en sus clasificaciones.

En el dominio de las series temporales, se debe considerar un tratamiento especial atendiendo a la naturaleza de los datos que se representan. El problema de la clasificación de series temporales puede ser definido de la siguiente forma:

Dada una base de casos $\mathrm{D} = \{\mathrm{t}_1, \mathrm{t}_2, \dots, \mathrm{t}_\mathrm{n}\}$ constituida por series temporales, y un conjunto de clases $\mathrm{C} = \{\mathrm{C}_1, \mathrm{C}_2, \dots, \mathrm{C}_\mathrm{m}\}$, el problema de clasificar dichas series consiste en definir una función $\mathrm{f}: \mathrm{D} \rightarrow \mathrm{C}$ donde cada t_i es asignada a una clase correspondiente. Y una clase C_j contiene precisamente a las series asignadas en ella, por tanto $\mathrm{C}_\mathrm{j} = \{\mathrm{t}_\mathrm{i} \mid \mathrm{f}(\mathrm{t}_\mathrm{i}) = \mathrm{C}_\mathrm{j}\,,\ 1 \leq \mathrm{i} \leq \mathrm{n}, \mathrm{t}_\mathrm{i} \in \mathrm{D}\}$.

[1] Weka fue creado en la Universidad de Waikato en Nueva Zelanda, está escrito en Java y publicado bajo la Licencia Pública General de GNU.

En [11] se propone clasificar los datos de las series temporales basándose en sus propiedades locales o en sus patrones. Mientras que en [12] se desarrolló un método de representación usando descomposición ondulatoria que puede seleccionar automáticamente los parámetros para la clasificación. Además, en [13] se proponen clasificadores de series temporales semi-supervisados para los que solo son necesarios un grupo reducido de ejemplos etiquetados de aprendizaje.

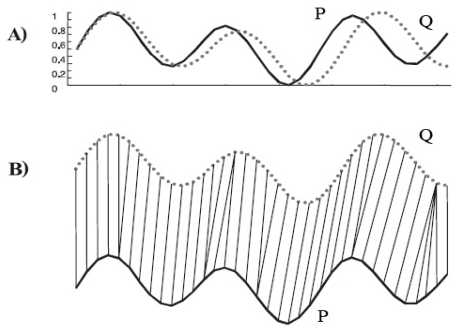
Recientemente, los investigadores se han enfocado en desarrollar clasificadores a la medida. Por ejemplo, en [14] se presenta una investigación sobre la clasificación de señales basándose en el modelado de un sistema dinámico que captura los datos para las series usando modelos de texturas Gaussianos. En [15] se estudia la adecuación de la medida de distancia DTW y de los árboles de decisión para la clasificación, mientras que en [16] se evalúa el desempeño de los clasificadores mediante la combinación de la reducción de la numerosidad usando DTW y clasificadores basados en la búsqueda del vecino más cercano.

Estudios empíricos han demostrado que el algoritmo de búsqueda de los k vecinos más cercanos, conocido como k-NN, es uno de los que mejores resultados ofrece en la clasificación de series temporales. Dada su sencillez y buen desempeño, la comunidad científica ha tomado a 1-NN[2] como algoritmo de referencia a derrotar en la mayoría de los casos, representando una buena medida de comparación para cada nuevo enfoque propuesto [17].

2.1 DTW

Las medidas de similitud tienen una gran importancia para las distintas tareas del análisis de series temporales. Dada la naturaleza numérica y continua de las series temporales, existen dos enfoques principales para el cálculo de la similitud: considerar la serie en toda su longitud, y la comparación de subsecuencias. Una de las medidas de distancias más utilizada es la euclidiana [18], y se emplea fundamentalmente en las series temporales transformadas. Sin embargo, varios estudios revelan que no siempre es la medida indicada en dominios específicos [19].

Una de las medidas de similitud más populares actualmente en el campo de las series temporales se conoce con el nombre de Distorsión Dinámica del Tiempo o DTW (acrónimo del inglés *Dynamic Time Warping*) [20]. DTW consigue un alineamiento optimal entre dos series temporales emparejándolas de forma no lineal mediante contracciones y dilataciones de las series en el eje temporal. Por consiguiente, este emparejamiento permite encontrar regiones equivalentes entre las series o hallar su similitud.



La Figura. 1 muestra como dos series P y Q son emparejadas entre sí a lo largo del tiempo usando DTW. Cada punto de la serie Q es conectado con su correspondiente punto similar en la serie P mediante una línea recta que los une. Si ambas series en la figura fuesen idénticas, todas las líneas entre ellas serían verticales pues no se necesitaría de un emparejamiento diferente para alinearlas entre sí. La distancia DTW es por lo tanto una medida de la diferencia entre dos series temporales después de que ambas hayan sido alineadas de forma óptima. Dicha medida se corresponde con la suma de las distancias entre cada par de puntos conectados en las series.

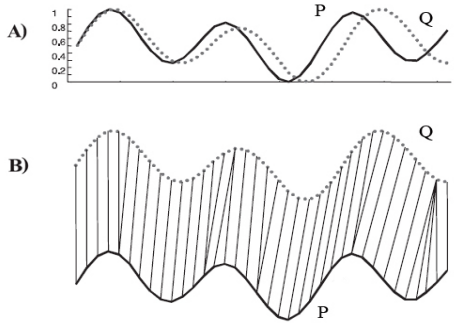


Figura. 1: A) Dos series temporales P y Q similares pero desfasadas. B) El emparejamiento entre los puntos de cada serie usando DTW.

Para alinear dos series temporales P y Q usando DTW, primeramente se construye una matriz M_{n*m}. El i-ésimo elemento de la matriz m_{ij} contiene la distancia $d(q_i, p_j)$ entre dos puntos q_i y p_j. La distancia euclidiana es la típicamente usada para calcular su alineamiento:

[2] Para el caso cuando $k = 1$, el algoritmo k-NN suele representarse como 1-NN.

$$d(q_i, p_j) = (q_i - p_j)^2 \quad (2)$$

Un camino distorsionado W constituye un conjunto continuo de elementos de la matriz M_{n*m} que definen una correspondencia entre P y Q.

$$w_k = (i_k, j_k)$$
$$W = w_1, w_2, \dots, w_k, \dots, w_K \quad (2)$$

Donde se cumple:

$$max(m, n) \leq K < m + n - 1 \quad (3)$$

Dicho camino, generado paso a paso, esta comúnmente sujeto a varias restricciones como por ejemplo: las condiciones de frontera, continuidad y monotonía. Las restricciones de frontera son $w_1 = (1,1)$ y $w_k = (m, n)$. Esto hace que el camino distorsionado comience y termine diagonalmente. Otra restricción es la de continuidad, dado un $w_k = (a, b)$, entonces $w_{k-1} = (a', b')$, donde $a - a' \leq 1$ y $b - b' \leq 1$. Esto restringe los pasos posibles en el camino a las celdas adyacentes, incluyendo la celda diagonal adyacente. Además, las restricciones $a - a' \leq 1$ y $b - b' \leq 1$ fuerzan a los puntos en W a ser monótonamente espaciados en el tiempo.

Existen un número bastante grande de caminos que satisfacen las condiciones antes mencionadas. Sin embargo, para el cálculo de DTW solo interesa el camino que minimice el costo de alineamiento, Figura. 2.

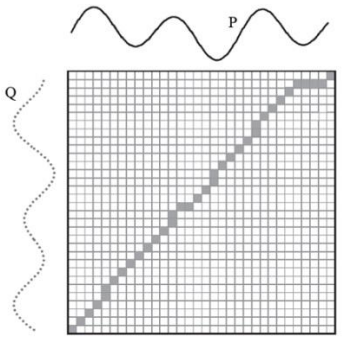


Figura. 2: Matriz M donde se forma el camino W mínimo para los alineamientos entre las series P y Q.

Este camino puede ser eficientemente hallado usando programación dinámica. Para ello se evalúa la ecuación de recurrencia (4), que define la distancia acumulada $\gamma(i, j)$ como la distancia $d(i, j)$ de la celda actual y el mínimo de las distancias acumuladas de los elementos adyacentes a ella.

$$\gamma(i, j) = d(q_i, p_j) + \min\begin{Bmatrix} \gamma(i-1, j-1), \\ \gamma(i-1, j), \gamma(i, j-1) \end{Bmatrix} \quad (4)$$

La ecuación (5) calcula el valor de DTW, generando un camino de alineamiento W tal que la distancia entre dichos elementos adyacentes es la mínima:

$$DTW(Q, P) = min\left\{\sqrt{\sum_{k=1}^{k} d(w_k)}\right\} \quad (5)$$

Donde $d(w_k)$ se define como muestra la ecuación (). Detalles sobre este tratamiento pueden verse en [21].

$$d(w_k) = d(q_{ik} - p_{ik})^2 \quad (6)$$

El cálculo de DTW completo tiene una complejidad computacional de orden cuadrático $O(n^2)$ lo cual hace que su implementación sea costosa cuando los puntos de datos de las series superan los miles (alta dimensionalidad). Diferentes métodos han sido propuestos para disminuir la complejidad temporal en el cálculo de las distancias DTW, estos pueden ser divididos en las siguientes categorías:

- **restricciones globales:** Limitar el número de celdas que son evaluadas en la matriz de costos (banda Sakoe-Chiba, paralelogramo de Itakura).
- **abstracción de los datos:** Ejecutar DTW en una reducida representación de los datos (reducción de la numerosidad).
- **indexado:** Usar funciones de acotación para reducir el número cálculos a DTW que deberán ejecutarse durante la clasificación o el agrupamiento (por ejemplo, el uso del algoritmo Keogh para la acotación inferior de DTW).

DTW ha encontrado amplia aplicación en varias disciplinas como son: minería de datos, reconocimiento de gestos, robótica, en procesos fabriles, en medicina [22], entre otros. En el reconocimiento del lenguaje oral, donde tuvo su primera aplicación, esta medida de similitud es utilizada eficientemente para determinar si dos ondas sonoras representan la misma frase en una conversación interpersonal. Esto se debe a que la duración del sonido de cada letra puede variar, pero la onda sonora en general debe tener la misma forma cuando la frase es la misma.

En minería de datos para series temporales DTW es usada comúnmente como una función de distancia y se considera la más usada actualmente para la clasificación de series temporales [23, 24]

2.2 Restricciones globales

El establecimiento de restricciones globales que controlan el subconjunto de la matriz que el camino de alineamiento es capaz de visitar, es uno de los métodos más usados en el mejoramiento de la eficiencia de DTW [25, 26]. La Figura. 3 muestra dos de las restricciones globales más usadas, el subconjunto de la matriz que el camino de alineamiento es capaz de visitar es también conocido como ventana elástica (en inglés: *warping window*).

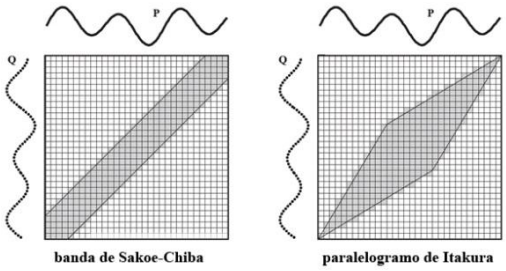


Figura. 3: Dos de las restricciones globales más utilizadas para el cálculo de DTW.

De esta forma, se restringen los índices del camino $w_k = (i,j)_k$ tal que $j-r \leq i \leq j+r$, donde r es el término que regula la elasticidad permitida para un punto dado de la serie ¡Error! No se encuentra el origen de la referencia.(normalmente r es un valor pasado como un parámetro de la función que calcula DTW). En el caso de la banda Sakoe-Chiba, r es independiente de i; para el paralelogramo de Itakura r es una función de i. Evaluaciones empíricas en numerosos conjuntos de datos han demostrado que los mejores resultados se obtienen cuando el valor de r no supera el 10% de la longitud de las series.

En [23] se lleva a cabo una evaluación empírica general de DTW comparándola con otras medidas de similitud. La Figura. 4 muestra los resultados obtenidos de la comparación para una colección de 38 conjuntos de datos de series temporales en la cual cada punto rojo representa un conjunto de datos, y los ejes de coordenadas los índices de error de cada uno. La comparación es por pares "A contra B", un punto rojo bajo la línea diagonal indica que "A" es superior a "B":

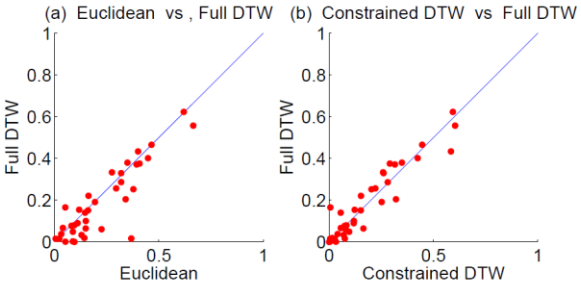


Figura. 4: Comparación entre las distancias: A) euclidiana y DTW sin restricciones globales B) DTW sin restricciones globales y DTW con un ancho de ventana r igual al 10% del conjunto de datos.

Como muestra la figura, DTW sin restricciones globales tiene un desempeño superior a la distancia euclidiana. Además, DTW con un ancho de ventana igual al 10% del conjunto de datos (el cual no tiene por qué ser el tamaño óptimo) es casi igual (e incluso ligeramente superior) a DTW sin restricciones globales. Lo anterior valida el uso de restricciones globales, en lugar de realizar el cálculo completo de DTW, reduciendo de esta forma el costo computacional empleado en su cómputo.

Por otro lado, se ha comprobado empíricamente que tanto los porcentajes de clasificación correcta como la velocidad de los cálculos amortizados dependen en gran medida del tamaño del conjunto de datos a utilizar. Esto pudiera parecer desalentador, teniendo en cuenta que el cálculo de la distancia euclidiana tiene una complejidad temporal de $O(n)$ mientras que DTW con restricciones globales tiene una complejidad de $O(nw)$, donde w es al ancho de la ventana utilizada. No obstante la complejidad amortizada de DTW durante la clasificación es realmente de $O((P*n)+(1-P)*nw)$, donde P es la fracción de cálculos de DTW podados usando la cota inferior de Keogh para la búsqueda de vecinos más cercanos en el algoritmo 1-NN, tal como veremos en el siguiente subepígrafe.

2.3 Cota inferior de Keogh para el cálculo de DTW

Otro método de probada eficacia para el mejoramiento de la eficiencia computacional del cálculo de DTW es el uso de la cota inferior de Keogh, con lo cual se evitan numerosos cálculos innecesarios de DTW. Para ello, se definen dos nuevas series a partir del valor de r:

$$U_i = max(q_{i-r}, q_{i+r})$$
$$L_i = min(q_{i-r}, q_{i+r}) \quad (7)$$

Donde U y L representan la serie superior e inferior respectivamente de una serie Q dada. Como se puede apreciar en la Figura. 5, ambas series conforman una banda que cubre totalmente la serie Q. Nótese que aunque la banda de Sakoe-Chiba tiene un ancho constante en la matriz, la banda correspondiente que envuelve la serie generalmente no tiene un espesor uniforme. En particular, dicha envolvente se hace más ancha en los puntos en los que la serie cambia más rápidamente, achicándose en sus mesetas.

Una propiedad obvia pero importante de las series U y L es la siguiente:

$$\forall_i\colon\ \mathrm{U}_i \geq q_i \geq \mathrm{L}_i \tag{8}$$

Habiendo definido las series U y L, se define la medida de acotación inferior para DTW:

$$LB\_Keogh(P,Q) = \sqrt{\sum_{i=1}^{n} \begin{cases} (c_i - \mathrm{U}_i)^2 & si\ c_i > \mathrm{U}_i \\ (c_i - \mathrm{L}_i)^2 & si\ c_i < \mathrm{L}_i \\ 0 & en\ otro\ caso \end{cases}} \tag{9}$$

Esta función puede ser visualizada como la distancia euclidiana entre cualesquiera de los puntos de la serie candidata y la envolvente más cercana a dicha serie, Figura. 5.

Luego, la cota inferior de Keogh calcula el cuadrado de la suma de la distancia euclidiana que hay entre cualquier parte fuera de la envolvente de la serie candidata $\boldsymbol{C}$, al borde ortogonal más cercano de la envolvente definida por $\boldsymbol{L}$ y $\boldsymbol{U}$. Este valor numérico es por lo tanto devuelto como una cota inferior de DTW.

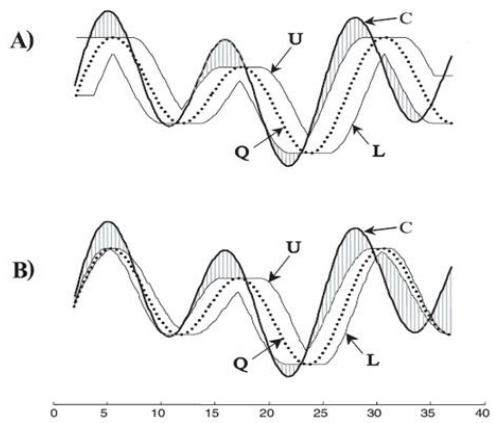


Figura. 5: Las series U y L creadas para la serie Q. A) usando la banda de Sakoe-Chiba y B) usando el paralelogramo de Itakura.

En este punto, se puede probar () [27].

$$\forall_i\colon\ \mathrm{U}_i \geq q_i \geq \mathrm{L}_i,$$
$$LBKeogh(P,Q) \leq DTW(P,Q) \tag{10}$$

Por tanto, la cota inferior de Keogh es una cota inferior de DTW, y su uso evita numerosos cálculos innecesarios. Cabe señalar que el costo computacional de obtención de esta cota inferior es de $O(n)$, o sea, un costo lineal mucho menor que el costo $O(nw)$ visto en el subepígrafe anterior.

2.4 Reducción de la numerosidad

En minería de datos, la idea de reducir la numerosidad de los datos sin perder exactitud en la clasificación ofrece numerosas ventajas [28]. Es bien conocido que si se escogen con cuidado las instancias que va a descartar un clasificador, esto puede reducir significativamente su tiempo de ejecución, a la vez que se mantiene la efectividad del clasificador (en muchos casos incluso los resultados obtenidos son mejores luego de efectuada la reducción apropiada).

Investigaciones recientes han mostrado que la utilización de DTW con restricciones óptimas, unido al uso de algoritmos que reduzcan convenientemente la numerosidad de los datos dan como resultado conjuntos de datos muy compactos y con poca o ninguna pérdida de precisión en la clasificación [13].

3. PAQUETE PARA LA CLASIFICACIÓN DE SERIES TEMPORALES EN WEKA

En este epígrafe se describen las tres extensiones implementadas para la clasificación de series temporales en Weka, las cuales fueron agrupadas en un paquete denominado timeSeriesClassification, tal como muestra la Figura. 6.

```
+-timeSeriesClassification
    +-src
    |   +-main
    |       +-java
    |           +-weka
    |               +-core
    |                   +-DTWDistance.java
    |                   +-neighboursearch
    |                       +-DTWSearch.java
    |               +-filters
    |                   +-supervised
    |                       +-instance
    |                           +-NumerosityReduction.java
    |
    |
    |
    |
    +-timeSeriesClassification.jar
    +-Description.props
    +-build_package.xml
    +-build
    +-lib
    +-doc
```

Figura. 6: Estructura del paquete timeSeriesClassification.

La compilación y construcción del paquete timeSeriesClassification se llevó a cabo usando Apache Ant en su versión 1.9.1 [29]. El paquete puede cargarse en cualquier versión de Weka posterior a la 3.7.7 [30]. A continuación se describen los pasos que se siguieron para implementar cada una las extensiones, siguiendo la metodología para extender Weka propuesta en [31].

3.1 Implementación de DTW como una función de distancia

Los métodos de minería de datos basados en k-NN son altamente sensibles a la función de distancia que utilizan [32]. A pesar de que varios algoritmos para la clasificación de series temporales han sido propuestos, incluyendo árboles de decisión [33], redes neuronales [34], redes bayesianas [34] entre otros, evaluaciones empíricas en más de 40 conjuntos de datos internacionales han mostrado que el clasificador 1-NN-DTW es superior a la mayoría de las técnicas usadas en la clasificación de series temporales [13, 23]. Por lo tanto, se implementó DTW para Weka como una función de distancia.

La implementación de DTW utiliza la banda Sakoe-Chiba como una restricción global opcional en su cálculo con un ancho de ventana r igual al porciento de la longitud de la serie empleado, Figura. 7.

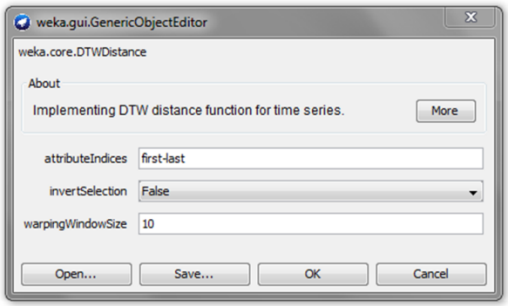


Figura. 7: Interfaz gráfica de la función de distancia DTWDistance implementada en Weka.

3.2 Algoritmo para la mejora de la búsqueda usando k-NN

Como se ha visto anteriormente (Subepígrafe 2.3), el uso de la cota inferior de Keogh ha sido adoptado en una gran cantidad de aplicaciones las cuales requieren que los cálculos de DTW se realicen de forma eficiente. Esta adopción ha facilitado el uso de DTW para la clasificación conjuntos de datos numerosos y constituidos por series temporales con una alta dimensionalidad.

El siguiente seudocódigo ilustra el algoritmo de búsqueda de vecinos más cercanos DTWSearch, implementado para la clasificación de series temporales en Weka. El algoritmo usa la cota inferior de Keogh para seleccionar las series más similares a la serie C en el conjunto de datos Q. DTWSearch está basado en la distancia DTW y ejecuta la función distanceLB para hallar la cota inferior de DTW. De esta forma se consigue ahorrar cálculos innecesarios de DTW, lo cual mejora significativamente la eficiencia del clasificador 1-NN-DTW.

```
1:  function sequentialScanLB(Q, C)
2:          bestSoFar = inf;
3:          for all sequences in dataset C do
4:                  distLB = distanceLB (C_i, B);
5:                  if(distLB < bestLB) then
6:                          trueDist = DTW (C_i, K);
7:                  if(trueDist < bestSoFar) then
8:                          bestSoFar = trueDist;
9:                          bestMatchIndex = i;
10: end function;
```

La Figura. 8 muestra la interfaz gráfica en Weka del algoritmo DTWSearch implementado.

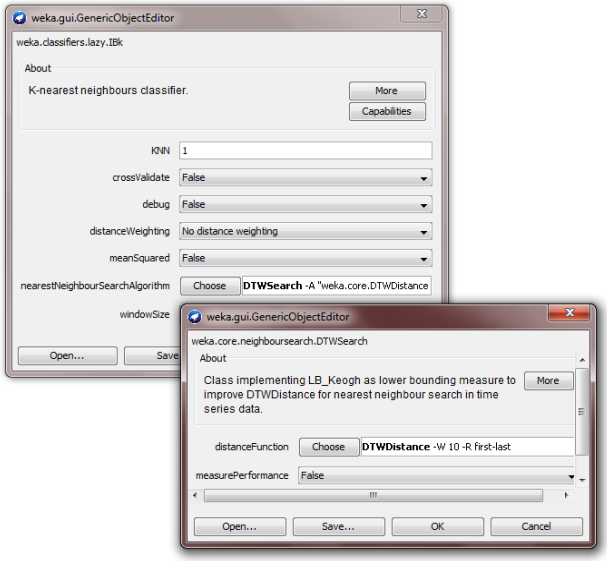


Figura. 8: Interfaz gráfica del algoritmo de búsqueda de vecinos más cercanos DTWSearch implementado en Weka.

3.3 Filtro para la reducción de la numerosidad de series temporales

Se realizó la implementación del filtro NumerosityReduction para Weka el cual puede ser utilizado en el pre procesamiento de los datos. Dicho filtro efectúa la reducción de la numerosidad de las series temporales eliminando aquellas instancias que menos aporten a la clasificación, teniendo siempre en cuenta la interdependencia entre el atributo clase y los valores de los demás atributos de la serie.

El algoritmo implementado en Weka se denomina Rank Based Reduction (RBR) y toma como base los algoritmos propuestos en [35]. La idea intuitiva de funcionamiento es simple y a la vez ingeniosa: si la eliminación de una instancia p en un conjunto de datos S no produce que otras instancias en S sean mal clasificadas, entonces p puede ser extraída de S sin que por ello se afecte la clasificación.

RBR funciona en dos etapas, las cuales denominaremos jerarquización y desempate. Primeramente se asigna una jerarquía a cada una de las series temporales (instancias), que van a ser clasificadas con 1-NN-DTWDistance, según su aporte a la clasificación de todo el conjunto de datos. Esta etapa comienza con la eliminación de todas aquellas series duplicadas (si existen), pues estas no aportan información nueva al clasificador. Luego se aplica 1-NN-DTW en todo el conjunto de datos, asignándole una menor jerarquía a aquellas series que representen información inútil ya que estas series generalmente afectan la clasificación de otras series cercanas. Para cada serie se le asigna un valor de jerarquía según la ecuación (11):

$$jerarquía(x) = \sum_j \begin{Bmatrix} 1 & si\ clase(x) = clase(x_j) \\ -2 & en\ otro\ caso \end{Bmatrix} \quad (11)$$

Donde x_j es la serie que tiene a x como su vecino más cercano. Por consiguiente, se le asigna mayor jerarquía a aquellas series que más aportan en la clasificación del resto, y aquellas que peor lo hacen obtienen valores negativos.

Si dos series tienen la misma jerarquía, entonces el empate se rompe asignándoles diferentes prioridades. La prioridad de una serie x es calculada según la siguiente ecuación ():

$$prioridad(x) = \sum_j \left(\frac{1}{(\text{DTWDistance}(x,\ x_j))^2} \right) \quad (12)$$

Donde x_j es la serie que tiene a x como su vecino más cercano y $\text{DTWDistance}(x,\ x_j)$ representa la distancia DTW entre x y x_j. El supuesto en este caso es que si una serie está demasiado alejada de su vecino más cercano, entonces este ejerce una menor influencia en la clasificación de la serie vecina. Luego, si dos series tienen la misma jerarquía, aquella con la menor prioridad será descartada primero. Notar que, gracias a que en la primera etapa se han eliminado las instancias duplicadas, se puede asegurar que el denominador de la fracción será distinto de cero.

Una vez definidas las clases que van a ser reducidas y el porcentaje que se desea reducir de las mismas (¡Error! No se encuentra el origen de la referencia.), se aplican ambas etapas del algoritmo. Finalmente se obtiene un conjunto de datos reducido el cual contiene aquellas series con un mayor valor de jerarquía.

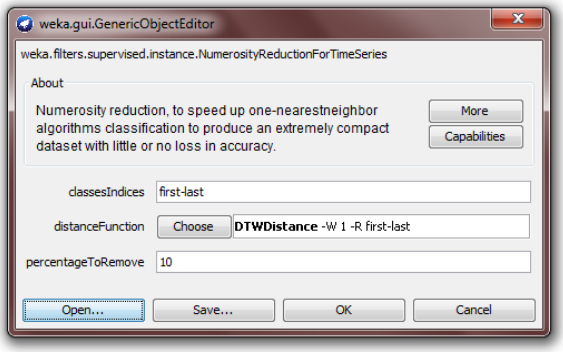


Figura. 9: Interfaz gráfica del filtro NumerosityReduc-

tion implementado en Weka.

4. RESULTADOS EXPERIMENTALES

En este epígrafe se realiza una validación experimental y estadística de las extensiones implementadas. Los resultados de los métodos son comparados con los resultados arrojados por los algoritmos equivalentes de Weka [30] y con los presentados en la literatura. Los conjuntos de datos de series temporales utilizados (entrenamiento y control) se encuentran disponibles en [36] y proceden de dominios diversos, garantizando así la fiabilidad de la experimentación.

4.1 Validación de la función de distancia implementada

A partir de los experimentos realizados se desea conocer el desempeño de la nueva función de distancia DTWDistance implementada, respecto a la distancia tradicional EuclideanDistance de Weka. DTWDistance es puesta a prueba por el algoritmo k-NN en la clasificación de diez conjuntos de datos. En todos los experimentos el clasificador 1-NN es utilizado, y la función de distancia DTWDistance implementa la banda Sakoe-Chiba como restricción global de DTW.

La Tabla I muestra los resultados obtenidos. Como se puede apreciar, para todos los conjuntos de datos el porciento de series bien clasificadas usando 1-NN con DTW como función de distancia (tanto con un ancho de ventana $r = longitud\ de\ la\ serie$ óptimo como sin restricción global de ventana) es superior, o igual en el peor de los casos, al porciento obtenido usando la distancia euclidiana de Weka.

Tabla I: Porcientos de clasificación correcta obtenidos aplicando 1-NN con EuclidianDistance y DTWDistance como función de distancia.

Conjuntos de datos	1-NN-Euclidian-Distance	1-NN-DTW con ancho de ventana r óptimo	1-NN-FullDTW (r = 100%)
Gun Point	91.3333	**92** (1)	90.6667
ECG200	87	**87** (0)	76
Yoga	83.0333	**84.1667** (2)	83.6
Coffe	75	**82.1429 (3)**	82.1429
Trace	76	99 (3)	**100**
Waffer	99.5457	**99.562** (1)	97.9883
Fifty Words	63.0769	**76.4835** (6)	69.011
Two Patterns	90.675	99.725 (4)	**100**
Fish	78.2857	82.8571 (4)	**83.4286**
CBF	85.2222	99.444 (11)	**99.6667**

Los resultados de la aplicación de la prueba no paramétrica de signos de Wilcoxon confirman las diferencias entre los resultados obtenidos con DTWDistance y EuclidianDistance. El número de diferencias positivas entre estas dos distancias es de 9, mientras que el número de diferencias negativas es igual a cero, y hubo un solo empate. Luego, la significación obtenida fue de 0.08, y por lo tanto se rechaza la hipótesis nula: la media de las diferencias de los porcientos de clasificación correcta no son equivalentes entre estas funciones de distancia. O sea, DTWDistance (con restricción global o sin ella) es superior a la función de distancia euclidiana EuclidianDistance de Weka.

4.2 Validación del algoritmo para mejora de la búsqueda k-NN

Para comprobar la eficacia de la utilización de DTWSearch como algoritmo para la búsqueda de vecinos más cercanos, se contabilizó la cantidad de cálculos de DTW que se podan durante su ejecución en comparación con el algoritmo LinearNNSearch estándar de Weka. En todos los casos las comparaciones fueron realizadas utilizando la función de distancia DTWDistance con el ancho de ventana óptimo de la Tabla I. La Figura. 10 muestra gráficamente el porciento de la cantidad de cálculos de DTW que se evitan en cada uno de los diez conjuntos de datos usando DTWSearch en lugar de LinearNNSearch.

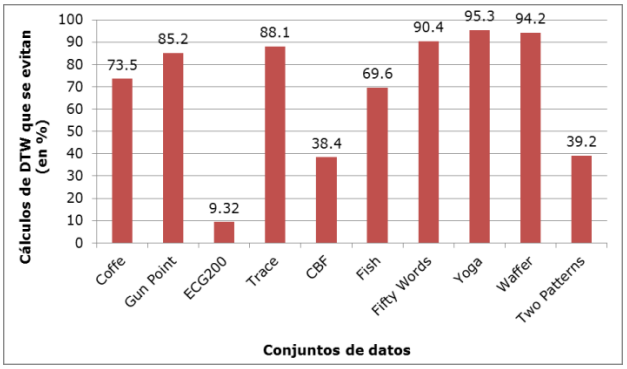


Figura. 10: Cálculos de DTW que se evitan usando DTWSearch como algoritmo de búsqueda de los k vecinos más cercanos para 1-NN-DTW.

Como se puede apreciar en la figura anterior, el número de cálculos de DTW que se evitan usando DTWSearch es significativo, sobre todo en aquellos conjuntos con gran cantidad de series de entrenamiento y control (Yoga, Waffer y Two Patterns).

4.3 Validación del filtro implementado

A continuación se desea comprobar la efectividad

del filtro NumerosityReduction implementado en Weka para la reducción de la numerosidad de series temporales. Los conjuntos de datos Two Patterns y Waffer tienen conjuntos de entrenamiento conformados por 1000 series cada uno (una cantidad significativamente mayor al resto), por esta causa, ambos van a ser seleccionados para evaluar la efectividad del filtro. Además se incluye en estos experimentos el conjunto ECG200 como ejemplo de los resultados que pueden ser obtenidos usando el filtro en un conjunto de datos menos numeroso.

En todos los casos, NumerosityReduction va a ser aplicado usando la distancia DTWDistance como función de distancia del algoritmo 1-NN, con un ancho de ventana igual al 10% de la longitud de la serie.

La Figura. 11 muestra los resultados obtenidos después de la aplicación del filtro con reducciones porcentuales ascendentes. Como se puede apreciar, para el caso de los conjuntos de datos Two Patterns y Waffer, los porcientos de clasificación del conjunto de prueba no varía significativamente hasta que se reduce hasta en un 90% del conjunto de entrenamiento. Esto demuestra que el filtro implementado es sumamente efectivo para la reducción de la numerosidad de estos dos conjuntos de datos. En el caso de ECG200, la reducción se hace efectiva hasta en un 70% de reducción del conjunto de entrenamiento, con lo cual se puede afirmar que el filtro no solo resulta ser eficaz para conjuntos de datos con gran numerosidad, sino que también es efectivo para otros no tan numerosos.

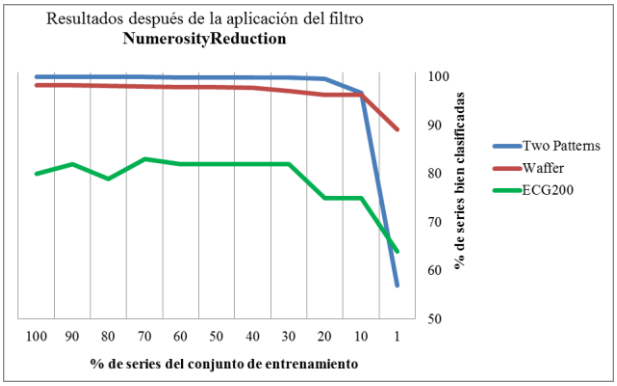


Figura. 11: Porciento de series correctamente clasificadas según el porciento de series reducidas en el conjunto de entrenamiento para los conjuntos de datos Two Patterns, Waffer y ECG200.

5. CONCLUSIONES

Las series temporales permiten describir de forma natural gran variedad de fenómenos que transcurren a lo largo del tiempo. Los métodos de minería de datos para series temporales superan muchas de las limitaciones presentes en los análisis estadísticos y matemáticos tradicionales. Atendiendo a las deficiencias de Weka para efectuar clasificación de series temporales se implementaron en forma de paquete la siguiente selección de métodos para facilitar esta tarea:

- Una función de distancia basada en DTW, la cual se comporta significativamente superior a la distancia euclidiana al efectuarse su validación en el contexto del algoritmo 1-NN.
- Un algoritmo para efectuar la búsqueda de vecinos más cercanos con 1-NN, el cual toma ventaja de la cota inferior de Keogh, con resultados que muestran de forma experimental una reducción considerable del costo computacional empleado durante la clasificación.
- Un filtro supervisado para la reducción de la numerosidad, situación muy común en el dominio de las series temporales, el cual efectúa una eliminación apropiada de aquellas instancias que entorpecen la clasificación; logrando mantener, para un menor número de instancias, la calidad de la clasificación una vez llevado a cabo el proceso de reducción.

6. REFERENCIAS

1. Fu, T.-c., *A review on time series data mining.* Engineering Applications of Artificial Intelligence, 2011. **24**(1): p. 164-181.

2. Shahiduzzaman, M., & Alam, K., *Cointegration and causal relationships between energy consumption and output: Assessing the evidence from Australia.* Energy Economics, 2012. **34**(6), p. 2182-2188.

3. Pandit, S.M. and S.-M. Wu, *Time series and system analysis with applications*. 1990: RE Krieger Publishing Company.

4. Bowerman, B.L. and R.T. O'Connell, *Forecasting and time series: an applied approach*. 1993: Belmont.

5. Povinelli, R.J. *Using genetic algorithms to find temporal patterns indicative of time series events*. 2000.

6. Faloutsos, C., M. Ranganathan, and Y. Manolopoulos, *Fast subsequence matching in time-series databases*. Vol. 23. 1994: ACM.

7. Keogh, E. *Fast similarity search in the presence of longitudinal scaling in time series databases*. 1997. IEEE.

8. Keogh, E.J. and M.J. Pazzani, *A simple dimensionality reduction technique for fast similarity search in large time series databases*, in *Knowledge Discovery and Data Mining. Current Issues and New Applications*. 2000, Springer. p. 122-133

9. Abonyi, J., et al. *Principal component analysis based time series segmentation-Application to hierarchical clustering for multivariate process data*. 2003.

10. COMMUNITY, P. *Time Series Analysis and Forecasting with Weka*. 2015 [citado el 23 de Mayo de 2015]; Disponible en: http://wiki.pentaho.com/display/DATAMINING/Time+Series+Analysis+and+Forecasting+with+Weka.

11. Geurts, P., *Pattern extraction for time series classification*, in *Principles of Data Mining and Knowledge Discovery*. 2001, Springer. p. 115-127.
12. Zhang, H., T.B. Ho, and M.S. Lin, *A non-parametric wavelet feature extractor for time series classification*, in *Advances in knowledge discovery and data mining*. 2004, Springer. p. 595-603.
13. Xi, X., et al. *Fast time series classification using numerosity reduction*. 2006. ACM.
14. Povinelli, R.J., et al., *Time series classification using Gaussian mixture models of reconstructed phase spaces.* Knowledge and Data Engineering, IEEE Transactions on, 2004. **16**(6): p. 779-783.
15. Rodríguez, J.J. and C.J. Alonso. *Interval and dynamic time warping-based decision trees*. 2004. ACM.
16. Wei, L., E. Keogh, and X. Xi. SAXually explicit images: finding unusual shapes. in Data Mining, 2006. ICDM'06. Sixth International Conference on. 2006. IEEE.
17. Grabocka, J., A. Nanopoulos, and L. Schmidt-Thieme, *Invariant time series classification*, in *Machine Learning and Knowledge Discovery in Databases*, Springer, Editor. 2012. p. 725-740.
18. Agrawal, R., et al., System and method for discovering similar time sequences in databases. 1997, Google Patents.
19. Megalooikonomou, V., et al. A multiresolution symbolic representation of time series. in Data Engineering, 2005. ICDE 2005. Proceedings. 21st International Conference on. 2005. IEEE.
20. Bernad, D.J., *Finding patterns in time series: a dynamic programming approach.* Advances in knowledge discovery and data mining, 1996.
21. Kruskall, J.B., Liberman, *The symmetric time warping algorithm: from continuous to discrete,TimeWarps,String Edits and Macromolecules.* 1983.
22. Keogh, E.J. and M.J. Pazzani. Derivative Dynamic Time Warping in SDM. 2001. SIAM.
23. Ding, H., et al., *Querying and mining of time series data: experimental comparison of representations and distance measures.* Proceedings of the VLDB Endowment, 2008. **1**(2): p. 1542-1552.
24. Ye, L. and E. Keogh. Time series shapelets: a new primitive for data mining. in Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining. 2009. ACM.
25. Ratanamahatana, C.A. and E. Keogh. Making time-series classification more accurate using learned constraints. 2004. SIAM.
26. Sakoe, H. and S. Chiba, *Dynamic programming algorithm optimization for spoken word recognition.* Acoustics, Speech and Signal Processing, IEEE Transactions on, 1978. **26**(1): p. 43-49.
27. Keogh, E., S. Lonardi, and B.Y.-c. Chiu. Finding surprising patterns in a time series database in linear time and space. In Proceedings of the Eighth ACM SIGKDD International conference on Knowledge Discovery and Data Mining. 2002. ACM.
28. Pękalska, E., R.P.W. Duin, and P. Paclík, *Prototype selection for dissimilarity-based classifiers.* Pattern Recognition, 2006. **39**(2): p. 189-208.
29. Foundation, A.S. 2013 [citado el 23 de Junio de 2015]; Disponible en: http://ant.apache.org.
30. Witten, I.H., E. Frank, and M.A. Hall, *Data Mining: Practical Machine Learning Tools and Techniques: Practical Machine Learning Tools and Techniques*. 2011: Morgan Kaufmann.
31. Héctor Matías Gonzále, L.I.A.P., *Extensiones al Ambiente de Aprendizaje Automatizado WEKA*, en *Departamento de Inteligencia Artificial*. 2005-2006, Universidad Central Marta Abreu de las Villas: Santa Clara.
32. Jiang, L., et al. Survey of improving k-nearest-neighbor for classification. in fskd. 2007. IEEE.
33. Rodríguez, J.J., C.J. Alonso, and H. Boström, *Learning first order logic time series classifiers: Rules and boosting*, in *Principles of Data Mining and Knowledge Discovery*. 2000, Springer. p. 299-308.
34. Nanopoulos, A., R. Alcock, and Y. Manolopoulos, *Feature-based classification of time-series data.* Information processing and technology, 2001: p. 49-61.
35. Wilson, D.R. and T.R. Martinez. Instance pruning techniques. in ICML. 1997.
36. Keogh, E., Zhu, Q., Hu, B., Hao. Y., Xi, X., Wei, L. & Ratanamahatana, C. A. *The UCR Time Series Classification/Clustering Homepage*. 2011; Disponible en: www.cs.ucr.edu/~eamonn/time_series_data/

7. BREVE SÍNTESIS CURRICULAR

César Soto Valero es Licenciado en Ciencia de la Computación, graduado en la Universidad Central de las Villas en el año 2013. Sus intereses investigativos incluyen aspectos de la minería de datos, el aprendizaje automatizado, la clasificación de series temporales y el análisis de grandes conjuntos de datos deportivos.