A new Kernel to use with discretized Temporal Series

F.J. Cuberos[a],L. González[b], F. Velasco[b], J.A. Ortega[c] and C. Angulo[d]

[a] Dept. Planificación-Radio Televisión de Andalucía, Seville (Spain)
[b] Dept. Applied Economics I, University of Seville (Spain)
[c] Dept. Data Processing Languages and Systems, University of Seville (Spain)
[d] GREC Research Group, Technical University of Catalonia, Vilanova i G. (Spain)
{luisgon, velasco}@us.es, fjcuberos@rtva.es, ortega@lsi.us.es, cangulo@esaii.upc.es

Abstract

In this paper a new kernel, from statistical learning theory, is proposed to work with symbols chains (words) obtained from a discretization procedure of a continuous feature. Meanwhile the exact definition of the discretization is not strictly necessary, it must ever exist either, a distance or a similarity measure between symbols in a certain alphabet (a set of symbols).

The proposed kernel is a generalization of a dot product in a vector space, not necessarily provided of any mathematical structure, that will allows to establish a similarity measure between objects of the alphabet.

This kernel is applied on a set of television shares obtained from the seven main television stations in Andalusia (Spain). A comparative study for classification purposes is done, and the associated parameter selection is studied.

1 Motivation

Automated processing and knowledge extraction from data is an important task to be performed by machine learning algorithms. Hence, it is possible the generation of classification rules from class-labelled examples. Instances can be described by a set of numerical, nominal, or continuos features. Several of these algorithms are expressly designed for handling numerical or nominal data; other algorithms performs better with discrete-values features, despite they can also handle continuous features [Kurgan and Cios, 2004]. Meanwhile a certain number of algorithms developed in the machine learning community focus on learning from nominal feature spaces, real-world classification includes patterns with continuous features where such algorithms can not be applied, unless the continuous features are firstly discretized. Discretization is the process of transforming a continuous attribute into a finite number of intervals associated with a discrete, numerical value –a number, symbol or letter. It is the usual approach for learning tasks that use mixed-mode –continuous and discrete)- data. Discretization process is developed in two stages: given the range of values for the continuous attribute, first it is found the number of discrete intervals; then, the width or boundaries for the intervals [González and Gavilán, 2001; Dougherty *et al.*, 1995; Kurgan and Cios, 2004; Macskassy *et al.*, 2003]. Discretization should generate a little number of possible symbols for the continuous attribute in order to avoid a slow and ineffective process of inductive machine learning [Catlett, 1991].

In [Macskassy *et al.*, 2003] was shown than even on purely numerical-valued data, results for text classification on the derived text-like representation outperforms the more naive numbers-as-tokens representation and, more importantly, is competitive with mature numerical classification methods such as C4.5, Ripper and SVM. The most straightforward way is to treat each number that a feature may take on as a distinct "word", and proceed with the use of a text classification method using the combination of true words and tokens-for-numbers words. However, this makes the numbers 1 and 2 as dissimilar as the numbers 1 and 100 –all three values are unrelated tokens to the classification methods. It would be desirable an approach to applying text-classification methods problems with numerical-valued features so that the distance between such numerical values is able to be discerned by the classification method. The approach considered in this paper is to translate every number into a set of intervals such that closer are two values, more similar will be the sets. This is done by finding a set of landmarks or split-points within the feature's range of legitime values by analyzing the values that the feature is observed to take on among the training example. Most of the methods translating a continuous feature into symbols –letters– in order to deal with texts –letters chains– lose part of their efficient since they are not designed for this end. The kernel proposed in this paper is specifically designed to work with letters chains coming from a discretization process of a continuous feature and it highlights the properties of these features. To cope the effectiveness of this kernel, it will be used on words from a dictionary where a distance exists between letters of the alphabet. The kernel was firstly proposed to compare among time series that had been converted into symbol chains –words– [Cuberos *et al.*, 2003; 2004]. Thus, the similarity measure between words quantified a distance between original time series. Figure 1 shows an example of a partial typified curve with their derivative values and the assigned label to each transition between adjacent values.

The rest of this paper is structured as follows: first, it is

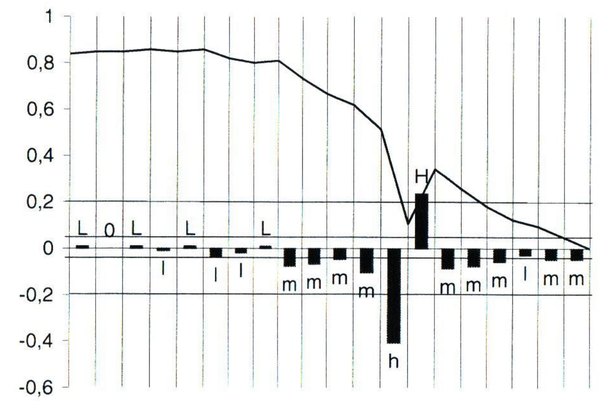


Figure 1: Sample of translation from a time serie to letters chain.

defined both, a kernel and a distance between finite intervals. Distance is used to define a real function measuring the similarity between two words and if words have the same length, this function is Kernel because it accomplish the Mercer condition. Later, one example about classification rules is developed. Finally, the conclusions and ideas for future works are enumerated.

To use traditional feature-vector-based learning methods, one could treat the presence or absence of a word as a Boolean feature and use these binary-values features. However, the use of a text-classification system on this is a bit more problematic, in the most straight-forward approach each number would be considered a distinct token and treated as a word. This paper presents an alternative approach for the use of text classification methods with numerical-valued features in which the numerical features are converted into bag-of-words features, thereby making them directly usable by text classification methods.

For many years the focus of the machine learning community has been on numerical and discrete-valued classification tasks, over the last decade there has also been considerable attention to text-classification problems [Sebastiani, 2002]. Typically such methods are applied by treating the presence or absence of each word as a separate Boolean feature. This is a commonly performed either directly, by generating a large number of such features, one for each word, or indirectly, by the use of set-valued features, in which each text-valued field of the examples is viewed as a single feature whose value for an example is the set of words that are present in that field for this example.

For many years the focus of the information retrieval community has been primarily on retrieval tasks, here, too, the last decade has seen a significant increase in interest in the use of such methods for text-classification tasks. The most common techniques use the retrieval engine as the basis for a distance metric between examples, for either direct use with nearest-neighbor methods, or based on the closely relates Rocchio relevance feedback technique, for use after creating a summary “document” for each class and retrieving the nearest one.

Many algorithms developed in the machine learning community focus on learning to nominal feature spaces [Dougherty *et al.*, 1995]. However, many real-world classification tasks exist that involve continuous features where such algorithms could not be applied unless the continuous features are first discretized. Continuous variable discretization has received significant attention in the machine learning community only recently.

This drawbacks can be overcome by using a discretization algorithms as a front-end for the learning algorithm.

2 Interval distance from a kernel

In essence, the goal in the construction of kernel functions is to guarantee the existence of an application ϕ, defined from the working set, $\mathcal{X}$ to a vectorial space endowed with a dot product, $\mathcal{F}$. From this function ϕ the kernel function is defined, denoted $k(\cdot,\cdot)$, over pairs of elements of the working set as the dot product of their transformations into the feature space, $k(\cdot,\cdot) = \langle\phi(\cdot),\phi(\cdot)\rangle_{\mathcal{F}}$, where $\langle\cdot,\cdot\rangle$ is denoted a dot product. The kernel function $k(\cdot,\cdot)$ let us establish similarities between the original elements from their transformed ones, so a distance between the points in the input space can be defined. It must be considered, when elaborating a similarity and distance measure, that the ϕ application must be able to highlight the essential characteristics of the initial set of elements.

Following the ideas presented in [González *et al.*, 2004b], let $\mathcal{I} = \{(c-r, c+r) \subset I\!R : c \in I\!R, r \in I\!R^+\}$ be the family of all the open intervals contained in the real line[(1)] of finite dimension. A function $\phi_1 : \mathcal{I} \rightarrow I\!R^2$ is defined as: $\phi_1(I) = P(c,r)^t$ and the kernel k and a distance d_1^2 between intervals are:

$$k(I_1,I_2) \quad = \quad \begin{pmatrix} c_1 & r_1 \end{pmatrix} S \begin{pmatrix} c_2 \\ r_2 \end{pmatrix}$$

$$d_1^2(I_1,I_2) \quad = \quad \begin{pmatrix} \Delta c & \Delta r \end{pmatrix} S \begin{pmatrix} \Delta c \\ \Delta r \end{pmatrix}$$

where $I_1 = (c_1 - r_1, c_1 + r_1)$, $I_2 = (c_2 - r_2, c_2 + r_2)$, $\Delta c = c_2 - c_1$ and $\Delta r = r_2 - r_1$, and P must be a non singular matrix ($S = P^t P$). Thus, the discretization of a continuous feature in symbols, usually letters, representing different intervals, allows us to use as a distance between symbols, the distance defined between intervals as it will be showed.

3 Kernel

From this point, we always consider that the symbols are letters $(A, B, \cdots)$ because the ordinal scale is reflected in the alphabetical order. Let $\mathcal{A} = \{A_1, A_2, \cdots, A_\ell\}$ be an alphabet of ℓ letters and let $\mathcal{P}$ be a set of the words that can be built from this alphabet. Let $P1 = P1_1 P1_2 \cdots P1_n$

[(1)] In default, we are working with open intervals, but it is posible to translate the study to closed intervals naturally.

and $P2 = P2_1 P2_2 \cdots P2_m$ be words from $\mathcal{P}$ where $P1_i$, $P2_j \in \mathcal{A}$ and $n \geq m$. A map K_λ is defined as:

$$K_\lambda(P1,P2) = \max\left\{\sum_{i=1}^{m} \lambda^{d^2(P1_{i+k},P2_i)},\ k = 0,\cdots,n-m\right\}$$

where $0 < \lambda < 1$ and $d(\cdot,\cdot)$ is a distance between letters.

Nota 3.1 *If words have the same length n, then:*

$$K_\lambda(P1,P2) = \sum_{i=1}^{n} \lambda^{d^2(P1_i,P2_i)}$$

Nota 3.2 *The kernel $K_\lambda(\cdot,\cdot)$ is a radial basis function (R.B.F.) since it is defined like a function of a distance, $f(d(P1,P2))$.*

Propiedad 1: *For all $P1,\ P2 \in \mathcal{P}$ and $0 < \lambda_1 < \lambda_2 < 1$ then: $K_{\lambda_1}(P1,P2) \leq K_{\lambda_2}(P1,P2)$.*

Propiedad 2: *For all $P1,\ P2 \in \mathcal{P}$ and $0 < \lambda < 1$ then: $K_\lambda(P1,P2) \leq m$. This upper bound is attained: If $P2 = P1_1 P1_2 \cdots P1_m$, then $K_\lambda(P1,P2) = m$.*

Propiedad 3: *Let $r = \max_{ij} d(A_i, A_j)$, with $A_i, A_j \in \mathcal{A}$. For all $P1,\ P2 \in \mathcal{P}$ and $0 < \lambda < 1$ then: $m\lambda^{r^2} \leq K_\lambda(P1,P2)$. This lower bound is attained: Let $A = A_i$ and $B = A_j$ be such that $d(A,B) = r^2$. If $P1 = AA\cdots A$ and $P2 = BB\cdots B$ with size of $P1$, n, and size of $P2$, m, then $K_\lambda(P1,P2) = m\lambda^{r^2}$.*

Thereby, for all $0 < \lambda < 1$:

$$m\,\lambda^{r^2} \leq K_\lambda(P1,P2) \leq m, \quad \forall P1,\ P2 \in \mathcal{P}$$

Propiedad 4: *Let $\mathcal{A}$ be an alphabet obtained from a discretization process of a continuous feature and $\mathcal{P} = \{P_1 P_2 \cdots P_n,\ P_i \in \mathcal{A}\}$ the set of all the words having length n, then*

$$K_\lambda(P1,P2) = \sum_{i=1}^{n} \lambda^{d^2(P1_i,P2_i)}$$

is a Kernel.

The proof can be found in [González *et al.*, 2003]. discretization process of a continuous feature into intervals. We consider the distance between intervals previously defined and a map ϕ from $\mathcal{A}$ to $I\!R^2$ is defined in the following way:
Each interval $(c-r\ ,\ c+r)$ from the discretization process is denoted by a letter from alphabet $\mathcal{A}$. Thus we consider the map ϕ_1 define from $\mathcal{I}$ to $I\!R^2$ and consider the composition between this and the intervals distance. A new map is defined $\phi : \mathcal{A} \longrightarrow I\!R^2$ such that:

$$\phi(A) = P\begin{pmatrix} c \\ r \end{pmatrix}, \quad \forall A \in \mathcal{A}$$

where P is a 2×2 matrix. It is defined a map $k_1 : \mathcal{A} \times \mathcal{A} \longrightarrow I\!R$ such that: $k_1(A,B) = \langle\phi(A),\phi(B)\rangle$. It is a kernel function by construction, because it is a dot product. Therefore: $d^2(A,B) = \langle\phi(A)-\phi(B),\phi(A)-\phi(B)\rangle$ is a pseudo-distance between words

Applying Corollary 3.13 (page 43) in [Cristianini and Shawe-Taylor, 2000], instead of the exponential function e^{-x} to the function $\left(\frac{1}{\lambda}\right)^{-x}$, with $0 < \lambda < 1$, it is true that the map $k_2 : \mathcal{A} \times \mathcal{A} \longrightarrow I\!R$ defined in the following way: $k_2(A,B) = \lambda^{||\phi(A)-\phi(B)||^2} = \lambda^{d^2(A,B)}$ is a kernel function. Now, if we consider the map $K_\lambda : \mathcal{P} \times \mathcal{P} \longrightarrow I\!R$ such that

$$K_\lambda(P1,P2) = \sum_{i=1}^{n} \lambda^{d^2(P1_i,P2_i)}$$

using the proposition 3.12 in [Cristianini and Shawe-Taylor, 2000], it is true that $K_\lambda(\cdot,\cdot)$ is a kernel function. ■

Tthe λ parameter measure the importance that $K_\lambda(\cdot,\cdot)$ give to matching symbols versus the comparison of different symbols. For coincident symbols the value is always 1.

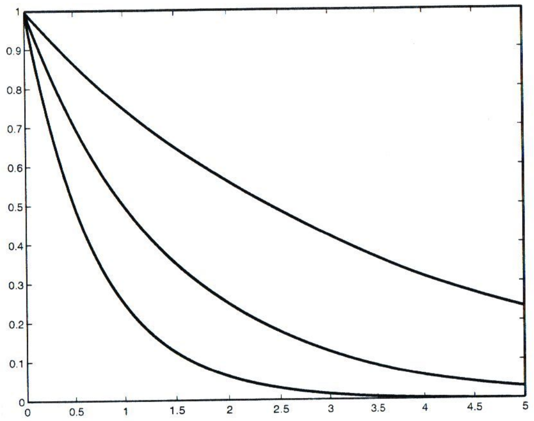


Figure 2: Graph representation of function $f(x) = \lambda^x$ for several values of λ.

In this language each word has a meaning since it represents a whole interval of values. For this reason, we should ask ourselves which are the characteristics we want to take into account in each word of the language to be able to extract meaning from them. The function $K_\lambda(\cdot,\cdot)$ considers the following ones: i) The order of the letters into each word, ii) Comparison letter by letter, and iii) The size of the words

3.1 Example

We think that it is very interesting to study an easy example before to see an implementation more complicated. Let be a table that quantifies the difference between letters:

$d(\cdot,\cdot)$	A	B	C	D	E
A	0	1	2	3	4
B	1	0	1	2	3
C	2	1	0	1	2
D	3	2	1	0	1
E	4	3	2	1	0

thus, the difference between letters is equal to the number of necessary jumps to pass from a letter to another letter.

The number of neccesary jumps to pass from the word $ACDBEAE$ to the word $EABEDCA$ (The letters are the same in different orden. The representation is in figure 3) is computed as follow.

$$\begin{array}{ccccccccc} A & C & D & B & E & A & E & & \\ 4 & 2 & 2 & 3 & 1 & 2 & 4 & \longrightarrow & \text{18 jumps} \\ E & A & B & E & D & C & A & & \end{array}$$

If the first word is retained and one unit is reduced to the difference between every letter form both words, a new word is obtained $DBCDEBB$:

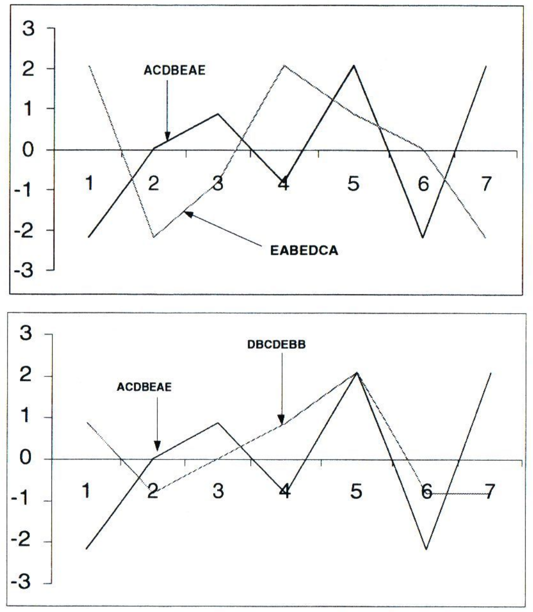


Figure 3: Representation of two words containing the same letters in a different orden. Every label represent the center of the class intervals computed with the CUM method.

that is nearer to the first word, computing 11 jumps.

$$\begin{array}{ccccccccc} A & C & D & B & E & A & E & & \\ 3 & 1 & 1 & 2 & 0 & 1 & 3 & \longrightarrow & \text{11 jumps} \\ D & B & C & D & E & B & B & & \end{array}$$

But must be noted that the ordinal scale $A < B < C < D < E$ come from the labelling of class intervals, so the difference between labels can be obtained from the distance between intervals. The CUM method, applied over the data set in [González *et al.*, 2003] from a continuos variable, produce the intervals:

Lower Bound	Upper Bound	Label	Center	Radius
-3.1844	-1.1911	A	-2.8770	0.9966
-1.1911	-0.4619	B	-0.8265	0.3646
-0.4619	0.4982	C	0.0182	0.4801
0,4982	1.2639	D	0.8811	0.3828
1.2639	2.9047	E	2.0843	0.8204

Let be $A = I$ (identity matrix) in the interval kernel previously proposed, then this scale is obtained[(2)]:

$d(\cdot,\cdot)$	A	B	C	D
B	1.5008			
C	2.2656	0.8525		
D	3.1296	1.7077	0.8684	
E	4.2757	2.9463	2.0940	1.2803

So the distance between the words $ACDBEAE$ and $EABEDCA$, previously 16 (jumps), as the sum of the distances between every pair of letters in the same position, is:

$$\begin{array}{ccccccccc} \text{A} & \text{C} & \text{D} & \text{B} & \text{E} & \text{A} & \text{E} & & \\ 4.2757 & 2.2656 & 1.7077 & 2.9463 & 1.2803 & 2.2656 & 4.2757 & \rightarrow & 19.017 \\ \text{E} & \text{A} & \text{B} & \text{E} & \text{D} & \text{C} & \text{A} & & \end{array}$$

and between the words $ACDBEAE$ y $DBCDEBB$:

$$\begin{array}{ccccccccc} \text{A} & \text{C} & \text{D} & \text{B} & \text{E} & \text{A} & \text{E} & & \\ 3.1296 & 0.8525 & 0.8684 & 1.7077 & 0 & 1.5008 & 2.9463 & \rightarrow & 11.005 \\ \text{D} & \text{B} & \text{C} & \text{D} & \text{E} & \text{B} & \text{B} & & \end{array}$$

that previously was 10.

3.2 Words of different size

Let be two words, $EBCDACDE$ and $ABAECD$, of different size. We study the similarities fixing the longest word and is compared with the shortest word :

$$\begin{array}{cccccccccc} A & B & B & E & C & D & & & & \\ 4 & 0 & 1 & 1 & 2 & 1 & & & \longrightarrow & \text{9 jumps} \\ E & B & C & D & A & C & D & E & & \\ & A & B & B & E & C & D & & & \\ & 1 & 1 & 2 & 4 & 0 & 0 & & \longrightarrow & \text{8 jumps} \\ E & B & C & D & A & C & D & E & & \\ & & A & B & B & E & C & D & & \\ & & 2 & 2 & 1 & 2 & 1 & 1 & \longrightarrow & \text{9 jumps} \\ E & B & C & D & A & C & D & E & & \end{array}$$

i.e., the distance between all the subwords from the longest word and the shortest word, maintaining the order of the words.

3.3 Generalized similarity

Let $P1$ and $P2$ be two words of the same length n from the set $\mathcal{P}$. In the definition of similarity between words, $K_\lambda(P1, P2) = \sum_{i=1}^{n} \lambda^{d^2(P1_i, P2_i)}$, all the letters have the same interest. It is possible to generalize this similarity by weighting each letter in such a form that the sum of the weights is equal to n.

Let $w_1, w_2, \cdots, w_n \in I\!R$ be scalar numbers accomplishing $w_i \geq 0$ and $\sum_{i=1}^{n} w_i = n$. The generalized similarity can be defined in two different ways:

$$K_\lambda^1(P1, P2) = \sum_{i=1}^{n} \lambda^{w_i \cdot d^2(P1_i, P2_i)}$$

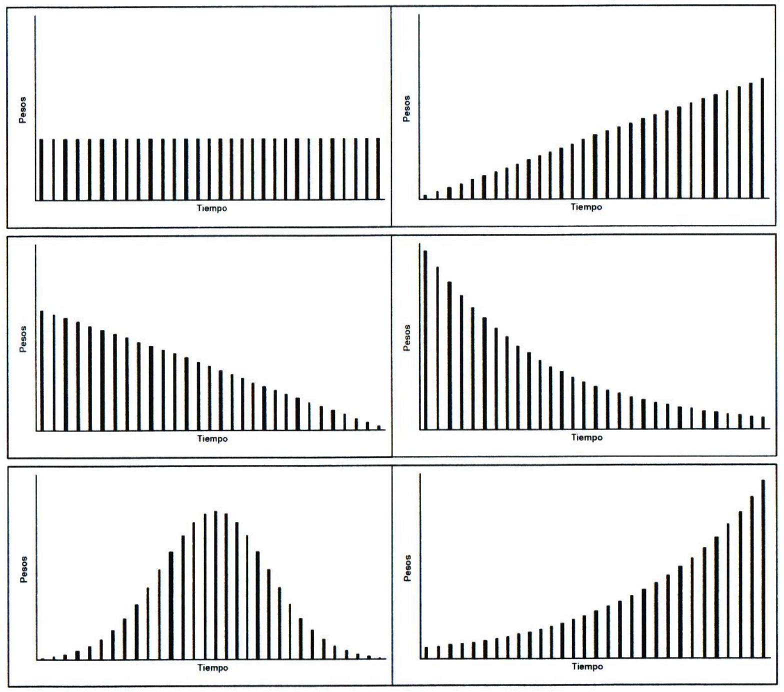


Figure 4: Different weighting shapes to build generalized similarities.

$$K_\lambda^2(P1, P2) = \sum_{i=1}^{n} w_i \cdot \lambda^{d^2(P1_i, P2_i)}$$

It is no difficult to proof that both are kernels[(3)], however the second one have a more intuitive meaning for the weights. Also, using to properties of the exponential function we have:

$$K_\lambda^1(P1, P2) = \sum_{i=1}^{n} \lambda^{w_i \cdot d^2(P1_i, P2_i)} = \sum_{i=1}^{n} w_i' \cdot \lambda^{d^2(P1_i, P2_i)}$$

where $w_i' = \lambda^{(w_i - 1)\, d^2(P1_i, P2_i)}$. Although no necessary is true that $\sum_{i=1}^{n} w_i' \neq n$. For this we propose as generalize similarity the function $K_\lambda^2(\cdot,\cdot)$.

In the figure 4, it can be observed several examples of weighting. In the left upper figure the same weight (unitary weight) is given to every letter in the word (the original definition). In the upper right figure a major importance is given to the last letters following a arithmetic progression, this situation can be useful to understand if the values of two stock shares can be similar. If we are analyzing a dynamic system from the behavior of a motor, the sense of similarity can be different given major importance to the transitory state against the stationary, and that can be achieved with the weights shown in the middle left figure. Analogously can be obtained the same or reverse situation with decreasing arithmetic progression (middle right) or increasing (lower right). Other possibility is shown in the lower right figure with the weights following a discretized normal model, where major importance is given to the letters in the middle of the word.

4 Implementation

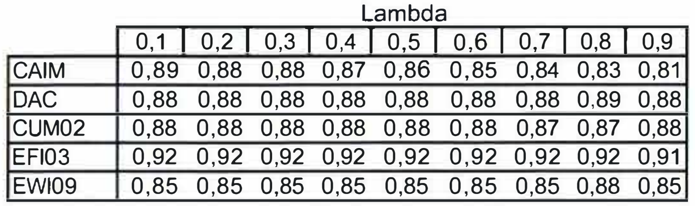
An example about classification rule is developed. Data to be considered is a set of television shares from the seven main television stations in Andalusia, Spain. It has been provided by Canal Sur Televisión and it has been collected from [TNS Audiencia de Medios, 2003]. Time series represent the average share for 15 minutes blocks, so the daily series are 96 elements length.

We are going to use several discretization methods and will see that the results are good in all them. A variety of discretization methods can be found in the literature. From the unsupervised algorithms: equal interval width, equal frequency interval, k-means clustering or unsupervised MCC; to supervised algorithms like $ChiMerge$, $CADD$, $1RD$, $D-2$ or maximum entropy. An extensive list can be found in [Kurgan and Cios, 2004].

The methods to be evaluated in this work are: 1) $Equal\,Width\,Intervals$ or EWI, 2) $Equal\,Frequency\,Intervals$ or EFI, 3) $CAIM$ (Class-Attribute Interdependence Maximization) [Kurgan and Cios, 2004], 4) $ChiSplit$ [González *et al.*, 2004a], 5) CUM

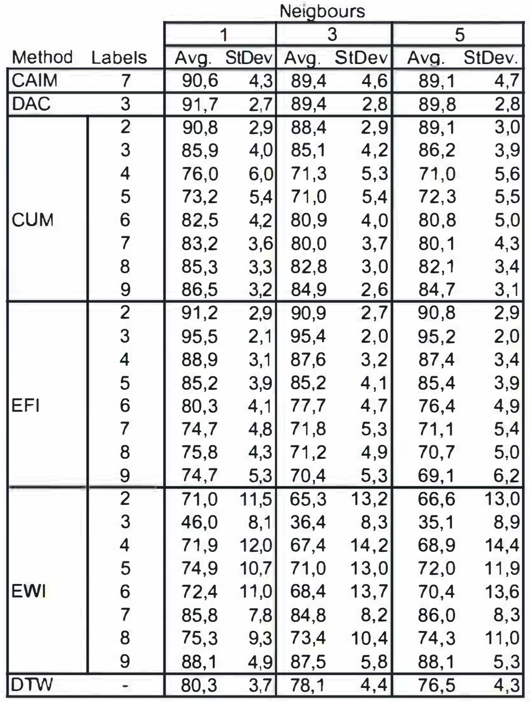
[(2)] $d(P, P) = 0$ for all letter P.

[(3)] The sum and the product of kernels is a kernel [Cristianini and Shawe-Taylor, 2000].



	Lambda								
	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,8	0,9
CAIM	0,89	0,88	0,88	0,87	0,86	0,85	0,84	0,83	0,81
DAC	0,88	0,88	0,88	0,88	0,88	0,88	0,88	0,89	0,88
CUM02	0,88	0,88	0,88	0,88	0,88	0,88	0,87	0,87	0,88
EFI03	0,92	0,92	0,92	0,92	0,92	0,92	0,92	0,92	0,91
EWI09	0,85	0,85	0,85	0,85	0,85	0,85	0,85	0,88	0,85

Figure 7: Percentage of correct identifications in Work Set for each method vs. value of λ



		Neigbours					
		1		3		5	
Method	Labels	Avg.	StDev	Avg.	StDev	Avg.	StDev.
CAIM	7	90,6	4,3	89,4	4,6	89,1	4,7
DAC	3	91,7	2,7	89,4	2,8	89,8	2,8
CUM	2	90,8	2,9	88,4	2,9	89,1	3,0
	3	85,9	4,0	85,1	4,2	86,2	3,9
	4	76,0	6,0	71,3	5,3	71,0	5,6
	5	73,2	5,4	71,0	5,4	72,3	5,5
	6	82,5	4,2	80,9	4,0	80,8	5,0
	7	83,2	3,6	80,0	3,7	80,1	4,3
	8	85,3	3,3	82,8	3,0	82,1	3,4
	9	86,5	3,2	84,9	2,6	84,7	3,1
EFI	2	91,2	2,9	90,9	2,7	90,8	2,9
	3	95,5	2,1	95,4	2,0	95,2	2,0
	4	88,9	3,1	87,6	3,2	87,4	3,4
	5	85,2	3,9	85,2	4,1	85,4	3,9
	6	80,3	4,1	77,7	4,7	76,4	4,9
	7	74,7	4,8	71,8	5,3	71,1	5,4
	8	75,8	4,3	71,2	4,9	70,7	5,0
	9	74,7	5,3	70,4	5,3	69,1	6,2
EWI	2	71,0	11,5	65,3	13,2	66,6	13,0
	3	46,0	8,1	36,4	8,3	35,1	8,9
	4	71,9	12,0	67,4	14,2	68,9	14,4
	5	74,9	10,7	71,0	13,0	72,0	11,9
	6	72,4	11,0	68,4	13,7	70,4	13,6
	7	85,8	7,8	84,8	8,2	86,0	8,3
	8	75,3	9,3	73,4	10,4	74,3	11,0
	9	88,1	4,9	87,5	5,8	88,1	5,3
DTW	-	80,3	3,7	78,1	4,4	76,5	4,3

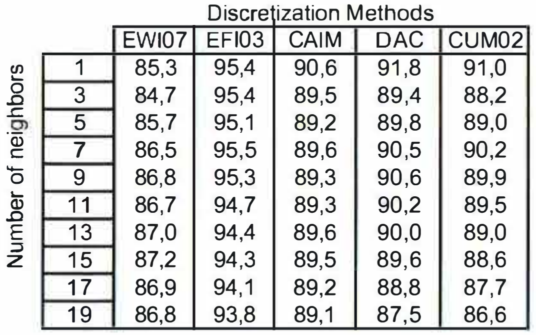
Figure 5: Identification Average (%) and Standard Deviation in Test Subset (200 Draws) vs. Number of neighbours

[González and Gavilán, 2001], and 6) DTW [Sakoe and Chiba, 1978].

In the following step several related task are accomplished: i) The discretization methods are applied over the learning subset producing a set of landmarks, ii) The landmarks are used as the limits of intervals and a symbol is assigned to each one, and iii) the series are translated into symbol chains.

The series are labelled with the name of the corresponding television station. We have selected the first 32 Wednesdays of year 2003 ($32 \cdot 7 = 224$ series) as the input set of series. Other 20 Wednesdays are used as work set (140 series) to be predicted.

In the Equal Width, Equal Frequency and CUM methods, the user must specify the number of intervals to be computed.



	Discretization Methods				
Number of neighbors	EWI07	EFI03	CAIM	DAC	CUM02
1	85,3	95,4	90,6	91,8	91,0
3	84,7	95,4	89,5	89,4	88,2
5	85,7	95,1	89,2	89,8	89,0
7	86,5	95,5	89,6	90,5	90,2
9	86,8	95,3	89,3	90,6	89,9
11	86,7	94,7	89,3	90,2	89,5
13	87,0	94,4	89,6	90,0	89,0
15	87,2	94,3	89,5	89,6	88,6
17	86,9	94,1	89,2	88,8	87,7
19	86,8	93,8	89,1	87,5	86,6

Figure 6: Identification Average (%) in Test Subset vs. Number of Neighbors

As no rule for a optimal value exist, all those methods will be calculated from 2 to 9 intervals. All the methods are applied to the learning subset and a list of interval boundaries are obtained. A letter in alphabetical order is assigned to each interval.

The learning system evaluates[(4)] the number of successful identifications on the test subset using the k-neighbors algorithm for each discretization method. The application of the presented methodology achieves a 95% correct identification rate for the work set series, 133 over 140. The best discretization method for this data set was $Equal\ Frequency\ Interval$ with 3 labels. Figure 5 shows the average percentage and variance for all the methods in 200 draws for 1, 3 and 5 neighbors.

In Figure 5 can be observed that, although the discretizaciôn methods build the intervals following different approaches, except for some anomalous case, the results are similar, that is, the kernel is very robust in front of the discretization methods.

Another question is about the influence of the number of neighbors, in the k-neighbors algorithm, in the results. Figure 6 represents the average identification for all the discretization methods with the odd values of k from 1 to 19. It shows that results are not improved with higher values of k when K_λ is used.

With respect to the parameter λ used in the kernel, it does not affect significantly to the average of correct identification.

[(4)] A complete study can be found in [Cuberos *et al.*, 2004].

Figure 7 shows that only the $CAIM$ method is affected by the variance of λ.

5 Conclusions and future work

In this paper a new similarity function for symbol chains has been defined, generating in some cases a kernel. This function measures similarities between words of a dictionary when a distance measure between symbols is defined.

In the near future, we will focus on the extension of this methodology to time series with multiple attributes and another kind of data. At the same time, we will use new data sets to extend its validation.

Finally, it must be mentioned that this kernel have certain implications in the type of considered similarity that will be studied in future investigations. The low influence of the λ parameter in identification tasks must be argued too.

Acknowledgements

This work was partially supported by the the Junta de Andalucía grants ACC-265-TIC-2001 and ACPAI-2003/014. Moreover, it has been partly supported by the Spanish Interministerial Committee of Science and Technology by means of the programs DPI2001-4404-E and TIC2003-09174-C02-01.

References

[Catlett, 1991] J. Catlett. On changing continuous attributes into ordered discrete attributes. In *Proceeding European working session on learning*, pages 164–178, 1991.

[Cristianini and Shawe-Taylor, 2000] N. Cristianini and J. Shawe-Taylor. *An introduction to Support Vector Machines and other kernel-based learning methods.* Cambridge University press 2000, 2000.

[Cuberos *et al.*, 2003] F.J. Cuberos, J.A. Ortega, F. Velasco, and L. González. ●SI. Alternative Labelling and Noise Sensitivity. In *17 International Workshop on Qualitative Reasoning*, 2003.

[Cuberos *et al.*, 2004] F.J. Cuberos, J.A. Ortega, F. Velasco, and L. González. A methodology for qualitative learning in time series. In *18 International Workshop on Qualitative Reasoning*, 2004.

[Dougherty *et al.*, 1995] J. Dougherty, R. Kohavi, and M. Sahami. Supervised and unsupervised discretization of continuous features. In A. Preditis and S. Russell, editors, *Machine learning: Proceeding of the twelfth international conference.* Morgan Kaufmann, 1995.

[González and Gavilán, 2001] L. González and J.M. Gavilán. Una metodología para la construcción de histogramas. Aplicación a los ingresos de los hogares andaluces. *XIV Reunión ASEPELT-Spain*, 2001.

[González *et al.*, 2003] L. González, F.J. Cuberos, F. Velasco, and J.A. Ortega. Un núcleo entre literales. Tech. Report 02, Dept. of Applied Economy I, University of Seville (Spain)., 2003.

[González *et al.*, 2004a] L. González, F.J. Cuberos, F. Velasco, and J.A. Ortega. DAC: A discretization algorithm. Tech. Report 12, Dept. of Applied Economy I, University of Seville (Spain)., 2004.

[González *et al.*, 2004b] L. González, F. Velasco, C. Angulo, J.A. Ortega, and F. Ruiz. Sobre núcleos, distancias y similitudes entre intervalos. *Inteligencia Artificial*, (23):111–117, june 2004.

[Kurgan and Cios, 2004] L. Kurgan and K.J. Cios. Caim discretization algorithm. *IEEE Transactions on Knowledge and Data Engineering*, 16(2):145–153, 2004.

[Macskassy *et al.*, 2003] A.A. Macskassy, H. Hirsh, A. Banerjee, and A. Dayanik. Converting numerical calssification into text classification. *Artificial Inteligence*, (143):51–77, 2003.

[Sakoe and Chiba, 1978] H. Sakoe and S. Chiba. Dymnamic programmin algorithm optimization for spoken word recognition. *IEEE Trans. on Acoustincs, Speech and Signal Proc.*, ASSP(26), 1978.

[Sebastiani, 2002] F. Sebastiani. Machine learning in automated text categorization. *ACM Computational Surveys*, (34):1--47, 2002.

[TNS Audiencia de Medios, 2003] TNS Audiencia de Medios. A service of Sofres AM company. www.sofresam.com, 2003.