On the relationship between distance and generalisation

V. Estruch C. Ferri J. Hernández-Orallo M.J. Ramírez-Quintana

July 12, 2006

1 Preliminaries

The most common metric spaces (e.g. the real numbers using the absolute difference for distance) normally have some properties (e.g. completeness) upon which new useful concepts and operations can be established (e.g. to compute the limit of a sequence). Therefore, if we aim to define a distance-based generalisation operator, we could wonder whether the metric spaces we are working with should satisfy certain conditions.

In this line, we propose the following reasoning: given two objects, x and y, we notice that a concept z is more general than x and y if z somehow collects the common features of x and y. Additionally, z is likely to represent other elements besides x and y. For instance, imagine that x is a regular pentagon and y is an equilateral triangle. Then, z could be the set of all the of regular polygon with n sides. It is helpful to see that many generalisations hide a transformation which can be used to gradually convert x into y and vice-versa. Namely, the pentagon of our example can be transformed into a regular triangle by means of two basic steps. First, we convert the pentagon into a square (the square is a particular case of z) by removing one of its sides and then we do the same over the square. Methods which have to perform a search in a metric space to find a good generalisation would benefit from this property. From this observation, it seems that the metric spaces we are interested in have to preserve some kind of "continuity" in order to express proper generalisations. Let us see that both concepts ("continuity" and "gradual transformations") are related.

From a mathematical point of view, the metric spaces employed in structured learning are usually discrete (discontinuous). For instance, the space of lists of symbols using the edit distance. But if we require the space of lists to be mathematically continuous, this would require to map lists into elements belonging to a continuous space (e.g. $\mathcal{R}^n$). This strategy sounds similar to how kernel methods work []. The problem is that defining a generalisation operator in this continuous space may not have an easy interpretation in the original data space. Hence, it seems to be better, at least for the sake of the comprehensibility of the extracted patterns, to work on the original space.

Our proposal is as follows: we intuitively consider that an object is continuous when it is not made of separated pieces, i.e. it is not divided. The concept of "naturally divisible" can be easily handled in a continuous space. For instance, if we observe Figure 1, it is intuitively clear that the square $X = [0,1] \times [0,1]$ is made of only one piece. However, it does not happen the same with the object $Y = Y1 \cup Y2 \cup Y3$. It is impossible to go, in a continuous way, from a point in Y_1 to a point in Y_2 without "stepping" outside $Y1$, $Y2$ and $Y3$. Necessarily, we have to jump from one "piece" to another.

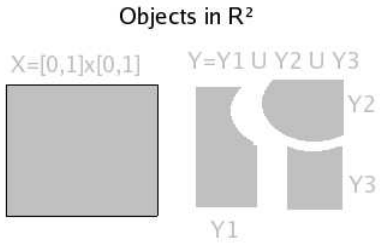

Figure 1: **(Left picture)** The closed square is not a divided object. We can connect any two points by means of a continuous path without leaving the object. **(Right picture)** In this case, the object is made of three separated pieces ($\{Y1, Y2, Y3\}$). There does not exist a continuous path included in Y which connects two points belonging to different components.

Let us see what happens with discrete spaces. By definition a discrete metric space is divided because it is made up of separated pieces, its elements and, between them, there is nothing. But if the generalisations we are looking for implicitly hide a "gradual transformation" , we need a not divided space. The trick will consist of revisiting the concept of "divided" in the following way. Two parts are divided if they are further than the shortest step which is possible in the space. For example, let us consider the finite metric spaces (X, d) and (Y, d) depicted in Figure 2. Although both spaces are discrete, the distribution of their elements is completely different. The elements of X are grouped into clusters whereas the elements of Y are not. The clusters will play the role of the separated pieces in the left figure. Note that it is impossible to "travel" from one element to another belonging to a different cluster taking minimum length steps. A longer step is always needed. It contrasts with space Y, where these minimum steps are always possible.

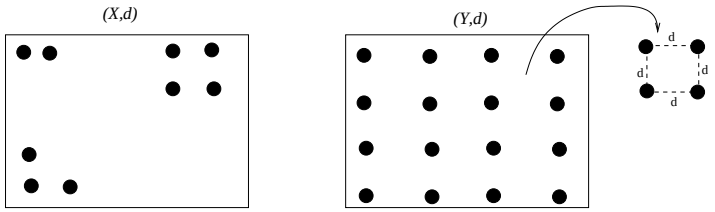

Figure 2: **(Left picture)** The elements of X are grouped into clusters. **(Right picture)** The elements of Y are uniformly distributed.

The concept of a "gradual transformation" is more intuitive in Y since equally significant changes can convert one element into another. In other words, spaces like Y have the advantage of leading to more reasonable generalisations. Let us illustrate this claim with the example in Figure 3. We want generalise the elements a and b. If these elements belong to X, not too much intuitive generalisations will be obtained because the way in which the elements of X are distributed. Note that a generalisation tries to stand out what properties two elements have in common. In this case, a generalisation should include some elements close to a (e.g. $a1$), others close to b (e.g. $b1$) and

those which let one go from a to b step by step. But latter do not exist and, thus, it is not possible gradually to go from a to b, there is no notion of middle point and it is more difficult to find the similar traits between a and b. However, this problem does not occur in space Y. We can always find a sequence of elements, quite close one to each other, to go from a to b. For example, in Figure 3, the most basic sequence is formed by a, c and b. Therefore, we are focusing on those metric spaces whose elements allow gradual transformations (in other words, the elements are "uniformly distributed") since generalisations can be performed more easily, and as we will see, the extracted patterns can be more regular and more comprehensible. The following definitions let us formalise this property.

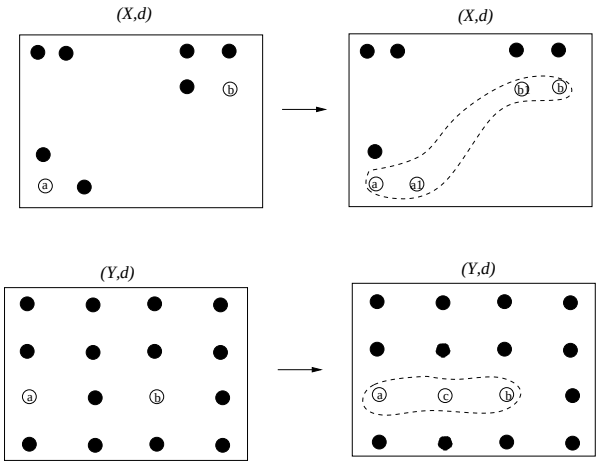

Figure 3: Generalisation of the elements a and b. **(Top-left picture)** The elements belong to different clusters. **(Top-right picture)** A generalisation should include the elements a, $a1$, b and $b1$. **(Bottom-left picture)** Now, the elements a and b belong to Y. **(Bottom-right picture)** Intuitively, the intermediate element c permits to directly connect a and b. Therefore, these elements could be included by a generalisation.

Definition 1 (Infimum Distance) *Let (X, d) be a metric space. We define the infimum distance of X, and we denote it by $I(X)$, as*

$$I(X) = inf\{d(x, y) : \forall x, y \in X, x \neq y\},$$

where inf stands for the infimum of a set. That is, $I(X)$ is the greatest lower bound of X not necessarely belonging to it.

Note that if (X, d) is a continuous metric space then $I(X) = 0$. However, the reciprocal is not true. There are discrete metric spaces where $I(X) = 0$, as we show in Example 2.

Definition 2 (Locally-connected) *Let (X, d) be a metric space, we say that X is a locally-connected metric space if for all $\epsilon > I(X)$ and for all x belonging to X, then $B(x, \epsilon) - \{x\} \neq \emptyset$, where $B(x, \epsilon) = \{y \in X : d(x, y) < \epsilon\}$.*

Intuitively, it means that every element belonging to X has at least one "neighbour" at a distance equal to or slightly greater than $I(X)$. Note that this definition is not still sufficient to say that the elements of X are "uniformly distributed". Next, two examples are given in order to illustrate the need for Definitions 3 to 6.

Example 1 *Let (X_1, d) and (X_2, d) be two finite metric spaces where $X_1 = \{x_1, x_2, x_3, x_4\}$, $X_2 = \{x_1, x_2, x_3\}$ and $d(\cdot,\cdot)$ is defined as follows:*

$d(\cdot,\cdot)$	x_1	x_2	x_3	x_4
x_1	0	δ	2δ	2δ
x_2	δ	0	2δ	2δ
x_3	2δ	2δ	0	δ
x_4	2δ	2δ	δ	0

Table 1: Definition of the distance function $d(\cdot,\cdot)$

Note that, $I(X_1) = I(X_2) = \delta$. Clearly, (X_2, d) is not a locally-connected metric space because $B(x_3, \delta) = \{x_3\}$. Nevertheless, (X_1, d) is a locally-connected metric space although it has two "clusters" $\{x_1, x_2\}$ and $\{x_3, x_4\}$.

Example 2 *Let (X, d) be a metric space where $X = \Sigma^*$ ($\Sigma = \{a, b\}$) and d is defined as follows*[1]*. Given to words $x = x_1 \cdots x_n$ and $y = y_1 \cdots y_m$ ($n, m > 0$) belonging to X,*

$$d(x,y) = \begin{cases} 0 & \text{, if } x{=}y \\ \frac{1}{2^i} & \text{, if } x_j = y_j \ (j = 1, \ldots, i) \text{ and } x_{i+1} \neq y_{i+1} \end{cases}$$

The infimum distance of X is equal to zero since for any $\epsilon > 0$, we can always find two words x and y belonging to X such that $d(x,y) < \epsilon$. The space (X, d) is not locally connected.

The elements of X_2 in example 1 and the elements of X in example 2 are not uniformly distributed and Definition 2 captures this. However, as we have mentioned, Definition 2 is not restrictive enough to ensure that the elements of a metric space are uniformly distributed (as the previous space (X_1, d) in Example 1). Thus, we need a harder condition. Having this in mind, we introduce the concept of δ-path.

Definition 3 (δ-path) *Let (X, d) be a metric space and let δ be a real number greater than zero. We will say that a finite sequence of elements $P = \{x_i\}_{i=0}^{n>0}$ belonging to X is a δ-path if $d(x_i, x_{i+1}) \leq \delta$ for all $0 \leq i \leq n-1$. We can represent this sequence as $x_0 \to x_1 \to \ldots \to x_n$*

In what follows, we will say that the elements x and y are δ-path connected when there exists a δ-path $P = \{x_i\}_{i=0}^{n>0}$ such that $x_0 = x$ and $x_n = y$. Intuitively, it means that we can "transform" x into y (through P) by means of successive changes which must be equal or smaller than δ.

Example 3 *Let (X, d) be a metric space where X is the set of finite words over the two-symbol alphabet $\Sigma = \{a, b\}$ and d the distance function defined in* **??** *(edit distance). If $x = aab$ and $y = bba$, then we can find a sequence according to Definition 3, such that:*

$$x = aab \to ab \to b \to bb \to bba = y,$$

[1] This distance is a slight variation of the fractal distance [].

and this sequence is a 1*-path.*

The idea from this example is that we can set $\delta = I(X)$ and hence we have the definition of a path with minimum steps. However, this idea would only work with those metric spaces X such that $I(X) > 0$. In order to cover every metric space, it will be enough to state what a 0-path is:

Definition 4 (0-path) *Let (X, d) be a metric space with $I(X) = 0$. We will say that a sequence of points P belonging to X is a 0-path, if P is the image of a continuous function γ defined over the closed interval $[0, 1]$ if the space is continuous or over the rational number in $[0, 1]$ otherwise, such that $\gamma([0, 1]) = P$.*

Summing up, a 0-path is just the well-known concept of a curve (indiscrete or discrete). That is, given two points x and y belonging to X, we will say that x and y are 0-path connected, if there exists a continuous function γ such that $\gamma(0) = x$ and $\gamma(1) = y$ (see Figure 4).

For the sake of simplicity, in what follows we call δ-path to every path with $\delta \geq 0$. When it is needed, we explicitly distinguish between $\delta > 0$ and $\delta = 0$.

A useful property concerning a δ-path, and that we will need in the next section, is its length.

Definition 5 (Length of a δ-path) *Let (X, d) be a metric space and let $P = \{x_i\}_{i=0}^{n>0}$ be a δ-path in X, then the length of P ($L(P)$) is defined as follows*[2]*:*

$$L(P) = \begin{cases} \sum_{i=0}^{n-1} d(x_i, x_{i+1}), \text{ if } \delta > 0 \\ sup\{\sum_{i=0}^{n-1} d(\gamma(t_i), \gamma(t_{i+1})) : \forall n \in N \text{ and } 0 = t_0 < t_1 < \cdots < t_n = 1\}, \text{ if } \delta = 0 \end{cases}$$

For instance, all paths in Figure 4 have length 5.
We are now ready to define the metric spaces we will work with.

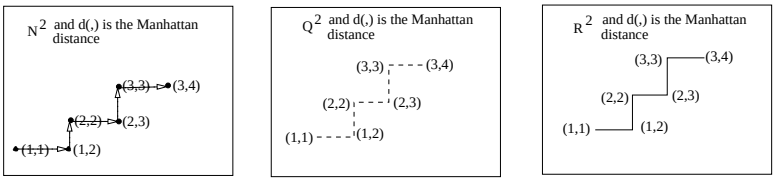

Figure 4: **(Left picture)** A 1-path (a finite collection of points) connecting the elements $(1, 1)$ and $(3, 4)$. **(Centre picture)** A 0-path (a discrete curve) connecting the elements $(1, 1)$ and $(3, 4)$. **(Right picture)** A 0-path (an indiscrete curve) connecting the elements $(1, 1)$ and $(3, 4)$.

Definition 6 (Globally connected[3]**)** *Let (X, d) be a locally-connected metric space. We will say that X is a globally-connected metric space, if for every pair of items x and y belonging to X, they are δ-path connected with $\delta \geq I(X)$.*

[2] *sup* stands for the supremum of a set, that is, its least upper bound.

[3] Note that there exist some peculiar distance functions in the sense that the return value is a tuple of numbers instead of one single number (see e.g. the distance function defined in **??**). Definition 6 can be easily adapted to take these cases into account. For instance, as the image set of the mentioned distance function is total ordered and contains a minimum element, then $I(X)$ is just this minimum ($I(X) = (1, -1)$).

Example 4 *The metric space used in the Example 3 is a* globally-connected *space. Immediately,* $I(X) = 1$*, and given two finite words* $x = x_1 \dots x_n$ *and* $y = y_1 \dots y_m$ $(n, m > 0)$*, we can find a trivial* 1*-path connecting them:*

$$x = x_1 \cdots x_n \to x_2 \cdots x_n \to \ldots \to x_n \to x_n y_1 \to y_1 \to \ldots \to y_1 \cdots y_{m-1} \to y = y_1 \cdots y_m$$

For indiscrete metric spaces, Definition 6 matches the well-known topological definition of a *path-connected* metric space. As the metric spaces we will work with will be always *globally-connected*, for the sake of simplicity, we will simply refer to them as *connected* metric spaces.
Before going on, it is enlightening to see that some "dense" spaces are not necessarily connected. For instance, the following example shows that "strange" spaces are discarded by Definition 6.

Example 5 *The metric space introduced in Example 2 is not connected. Given any two words* x *and* y *belonging to the space, it is impossible to find a continuous function* γ *such that* $\gamma(0) = x$ *and* $\gamma(1) = y$*. From a practical point of view, this latter distance is quite useful if we pursue to distinguish those words beginning by a or by b. Fortunately, although it is out of the scope of this paper, it can be redefined in such a way that the elements of the space* (X, d) *are uniformly distributed.*

Summing up, Definition 6 means that the space is made of one piece, as we discussed at the beginning. This idea is key to introduce the concept of connected sets in discrete metric spaces, i.e. the extension of the idea to groups of objects in the space.

Definition 7 (Connected set) *Let* (X, d) *be a connected metric space. We will say that* $Y \subset X$ *is a connected set, if for every pair of elements* x *and* y *belonging to* Y *there exists a* δ*-path* P *connecting them such that* $\delta = I(X)$ *and* P *is included in* Y*.*

Informally speaking, a connected set is a set, i.e. a group of elements from a connected space, which is made of one piece.

Example 6 *Let us illustrate what a connected set is in a discrete space* Y*. For this purpose, we will take the space depicted in the Figure 3. As we can appreciate in the left picture (see figure below), the set* A *is connected since for every pair of elements belonging to* A*, there exists at least one* δ*-path with* $\delta = I(Y)$ *(dashed lines) included in* A *connecting them. However, the set* B *on the right is disconnected. The elements* x *and* y *are not* δ*-path connected with* $\delta = I(Y)$*.*

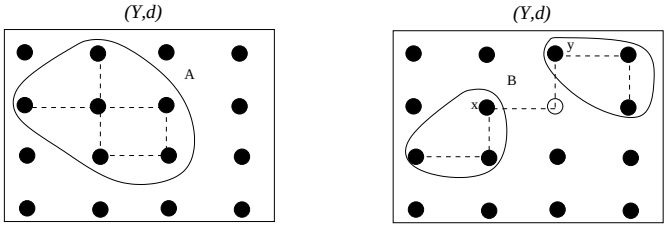

Figure 5: **(Left picture)** The set A is connected since its elements are δ-path connected. **(Right picture)** Nevertheless, the set B is disconnected because we would need an extra element (for instance, z) in one of the two sets X or T.

2 Generalisation operators

In our approach, connected sets play a fundamental role in order to define a generalisation operator since a generalisation of a group of elements will be just a connected set containing them. First, in this section we introduce a generalisation operator for a pair of items (binary generalisation operator) and then, we will extend it for n-ary operators, that is, operators taking more than two input elements. In advance, we will say that, in this latter case, the generalisation will be a connected set covering the group of items as well.

Intuitively, the (binary) generalisation of two items of a connected metric space (X, d) could be extensionally defined as a connected set that contains both items. But, from a comprehensibility point of view, we must highlight that our concept of generalisation should be associated to a family of patterns $\mathcal{L}$, where each pattern represents a set in the connected metric space (X, d). Note that a generalisation which is just expressed as a set of elements (e.g. $\{2, 4, 6, \ldots\}$)is much less useful than a generalisation which has an associated pattern (e.g. $\{x : odd(x)\}$). That is, a pattern $h \in \mathcal{L}$ will be an intensional (and hopefully "comprehensible") manner of denoting the set of all the elements in X which are covered by h. Additionally, viewing patterns as the sets they denote, we can use the well-known set operations. For example, we can say that a pattern h_1 is included in a pattern h_2 if the set which represents h_1 is included in the set which represents h_2, or we can say that an element $x \in X$ belongs to the pattern h, if x is covered by h. Note that the same set can have several patterns which denote it. Depending on our pattern language $\mathcal{L}$ we will be able to express some sets as generalisations but some others not. For instance, if our family of patterns in the metric space $\mathcal{R}^2$ is made of all the possible squares of size 1×1, the concept of a 2×2 square can not be expressed. An important reason to introduce the notion of $\mathcal{L}$ is because not all the connected sets in one given space X will usually have an intuitive pattern associated to them (just figure out all the possible connected sets in $\mathcal{R}^2$). For this purpose, $\mathcal{L}$ will be defined according to the problem to be solved, and very specially, on the kind of patterns the user can understand. Nevertheless, in the worst case, $\mathcal{L}$ can always be defined as 2^X if we do not have any representation bias. So, instead of extensional generalisation operators, we will work with intensional operators.

Definition 8 (binary generalisation operator) *Let (X, d) be a connected metric space and let $\mathcal{L}$ be a pattern language. An binary generalisation operator Δ is a function $\Delta : X \times X \to \mathcal{L}$ such that:*

$$\forall x, y \in X, \exists h \in \mathcal{L} : \Delta(x, y) = h,$$

where $x, y \in h$ and h is a connected set.

Therefore, a binary generalisation operator simply maps pairs of items into patterns representing connected sets.

So far, we have not still mentioned anything concerning the"shape" of the sets computed by Δ. For instance, the generalisation of two points belonging to $\mathcal{R}^2$ could be something as simple as a straight line or something as complicated as the most intricate curve connecting them. But, from a comprehensibility point of view, it might be more interesting to compute regular-shaped sets (see Left Figure 6). Therefore, it is convenient to established some conditions about the "shape" of these sets. For this purpose, we distinguish among *proper*, *hard-proper* and *improper* generalisations:

Definition 9 (*proper* generalisation) *Let Δ be a binary generalisation operator defined in a connected metric space (X, d). We will say that Δ is a proper generalisation if, for every pair of*

*elements x and y ($x \neq y$) belonging to X, $\Delta(x, y)$ includes **some** $I(X)$-path P connecting x and y such that $d(x, y) = L(P)$.*

Definition 10 (*hard-proper* generalisation) *Let Δ be a binary generalisation operator defined in a connected metric space (X, d). We will say that Δ is a hard-proper generalisation if, for every pair of elements x and y ($x \neq y$) belonging to X, $\Delta(x, y)$ includes **all** the $I(X)$-paths P connecting x and y such that $d(x, y) = L(P)$.*

For instance, if we consider the plane $\mathcal{R}^2$ with the Manhattan distance[4] and $\mathcal{L}$ some sufficiently expressive family of patterns, then given two points A and B belonging to $\mathcal{R}^2$, all the 0-paths P connecting A and B such that $L(P) = d(A, B)$ are included (among other sets) in a square whose two of its opposite vertices are A and B. If $\Delta(A, B)$ gives this square, we can say that Δ is a *hard-proper* generalisation. On the contrary, if $\Delta(A, B)$ gives something smaller e.g. a triangle whose two of its vertices are A and B, then Δ is not a *hard-proper* generalisation (see Right Figure 6).

Definition 11 (*improper* generalisation) *Let Δ be a binary generalisation operator defined in a connected metric space (X, d). We will say that Δ is an improper generalisation operator if it is not proper, that is, given any two elements x and y ($x \neq y$) belonging to X, then $\Delta(x, y)$ does not include **any** $I(X)$-path P such that $d(x, y) = L(P)$.*

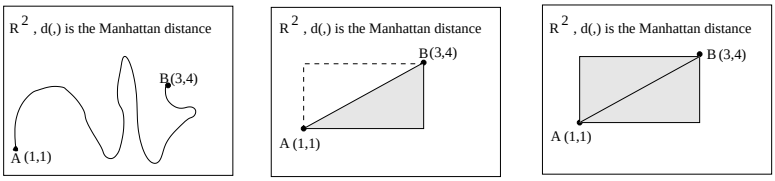

Figure 6: Generalising the elements $A(1, 1)$ and $B(3, 4)$. **(Left picture)** This curve is an *improper* generalisation because its length is greater than $d(A, B)$. **(Middle picture)** This triangle is a *proper* generalisation of the elements A and B. Note that the dashed line is a 0-path whose distance is equal to $d(A, B)$ and it is not covered by the triangle. **(Right picture)** This square is a *hard-proper* generalisation of the elements A and B.

In order to define n-ary distance-based generalisation operators, the concept of "nerve" of a set of elements E is needed. In this way, a nerve of E, denoted by $N(E)$, is simply a connected[5] graph whose nodes are the elements belonging to E.

Definition 12 ***(Distance-based n-ary generalisation operator)*** *Let (X, d) be a connected metric space and let $\mathcal{L}$ be a pattern language. Given a generalisation operator Δ, we will say that Δ is a (proper or hard) distance-based generalisation operator if, for every $E \subseteq X$, there exists a nerve $N(E)$ such that,*

- ***(proper)** For every pair of elements x, y in E such that they are directly linked in $N(E)$, $\Delta(E)$ includes **some** $I(X)$-path P connecting x and y such that $d(x, y) = L(P)$.*

[4]Given two points $A(a_1, a_2)$ and $B(b_1, b_2)$, the Manhattan distance is defined as $d(A, B) = |a_1 - b_1| + |a_2 - b_2|$.
[5]Here, theterm connected refers to the well-known property for graphs.

- ***(hard)*** *For every pair of elements x, y in E such that they are directly linked in $N(E)$, $\Delta(E)$ includes* ***all*** *$I(X)$-path P connecting x and y such that $d(x, y) = L(P)$.*

Definition 12 can be difficult to understand. Note that when $E = \{x, y\}$, this latter definition matches definitions of proper or hard-proper binary generalisation operator.
Another issue related to the generalisation operator is to determine when it performs the least general generalisation (lgg). It will be an important issue if we want the generalisations to "fit" a group of elements as much as possible. Despite the lgg is a widely studied concept in the field of *Inductive Logic Programmming* (ILP), it does not happen the same when the data is not described by means of logical predicates. Thus, the following observations are in some way an attempt to extend the notion of lgg for different sorts of data, not only first-order predicates. The process is similar to the one employed in ILP. First, we will use a criterion to say, given two generalisations G_1 and G_2 of a set of elements E, which one is less general. Unlike ILP, this criterion will be based on the underlying metric space. Then, note that, automatically, this criterion induces an order relation in the set of all the possible generalisations of E, where the lgg will be just the bottom element of this ordered set.

Note that the lgg concerns the generalisations of a set of elements E, only saying nothing about the generalisation operator Δ. In other words, it is possible that Δ computes the lgg for E, but not necessarily for other sets. This issue will be treated in further sections where we will define some Δ generalisation operators for several sort of data, such that Δ performs always the lgg for every set E.

In order to formalise our proposal, we might utilise the inclusion operation between sets ($\subset$) as a "mechanism" to compare how general two generalisations are. That is, viewing generalisations as connected sets, we would say that the generalisation G_1 is less general than the generalisation G_2, if $G_1 \subset G_2$. However, if we do that, several details must be taken into account:

- The minimal generalisation: in addition, we should substitute the adjective "least" by the adjective "minimal" because we could find in the pattern language $\mathcal{L}$ two non comparable lgg. For instance, if $\mathcal{L}$ is the family of all the rectangles in $\mathcal{R}^2$, the two rectangles depicted in Figure 7 (right) could be lgg. In what follows, we will use the term minimal generalisation (mg).

- Flexibility:

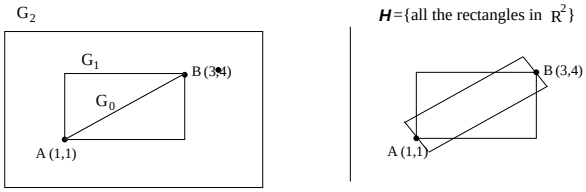

Figure 7: **(Left picture)** Several *hard-proper* generalisations of the elements A and B. The closed figure G_2 (the square) is a *hard-proper* generalisation of the elements A and B. **(Right picture)** Two different minimal generalisations of the elements A and B considering all the rectangles in $\mathcal{R}^2$ as the language pattern.

Unfortunately, if we consider only the inclusion to compare how general two generalisations are, an important problem arises. Again, let us consider $\mathcal{R}^2$ with the Euclidean distance and $\mathcal{L}$ as the set of all the rectangles in $\mathcal{R}^2$. Then, the generalisation of n points belonging to $\mathcal{R}^2$ would be something as simple as a rectangle containing them (see left Figure 8). However, in some contexts, it would be preferable to obtain some slightly more elaborated generalisations. For instance, the one depicted on the right of Figure 8. Note that, in this case, a more expressive pattern language is needed ($\mathcal{L}'$ is the set of all the rectangles in $\mathcal{R}^2$ along with their finite unions). But the more complex a generalisation is, the less intelligible it is and, the higher the chance of overfitting is.

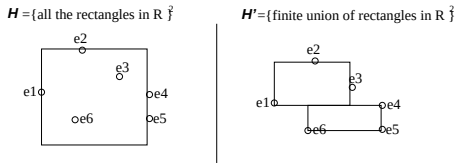

Figure 8: **(Left picture)** A naive generalisation of the elements $\{e_1, \ldots, e_6\}$. **(Right picture)** A more elaborated generalisation considering a more expressive family of patterns.

Now, a second problem is that the mg could not exist. Consider two elements A and B in the pattern language $\mathcal{L}'$ mentioned above. If we look at Figure 9 which works with a $\mathcal{L}$ which is composed of all the finite union of rectangles in $\mathcal{R}^2$, given any generalisation G_i in $\mathcal{L}'$ of $\{A, B\}$, we can always find another generalisation G_j included in G_i ($G_j \neq G_i$). For this purpose, it is enough to draw a connected chain of squares included in G_i which links both A and B. Therefore, if we define the mg in terms of the inclusion between sets, then the mg does not exist for this special pattern language $\mathcal{L}'$.

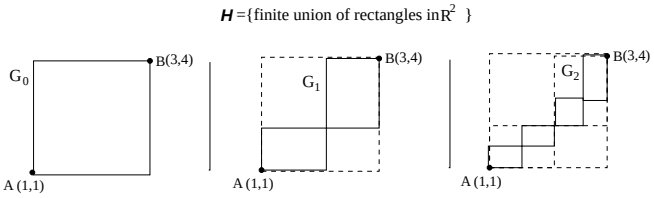

Figure 9: For every generalisation G_i of $\{A, B\}$, we can always find another generalisation G_j of $\{A, B\}$ in $\mathcal{L}'$ such that $G_j \subset G_i$. In this picture, we have that $G_2 \subset G_1 \subset G_0$.

In general, this problem will be presented when we work with continuous metric spaces and we use a pattern language built from a set of basic constructors. But it turns out that this kind of pattern language is quite common in symbolic learning, i.e. regular languages. Therefore, we need to change the inclusion between sets as a generality criterion. A possibility is as follows.

Note that the "complexity" of the pattern G_i grows as more squares are employed. But this is only reasonable if a sufficient number of cases justifies a complex pattern, just as MDL/MML principle states []. From a practical point of view, a pattern made of thousand of small squares becomes hardly comprehensible, even if it fits the evidence E really well. Hence, it could be more

interesting to reach a trade-off between minimality and comprehensibility. For this purpose, we will introduce a special function, called the cost function and denoted by $k(E, h)$, which takes both the complexity of the pattern h and how good the pattern fits E into account. Returning to Figure 9, the cost function $k(E, h)$ should be able to discriminate those unnecessary complicated patterns, such as G_2, from those being more suitable such as G_1 but at the same time should discard simple rectangles which are not well adjusted, as G_2 in Figure 7. Hence, the mg will be defined in terms of a cost function instead of the inclusion operator between sets. As the mg depends on both a cost function k and a pattern language $\mathcal{L}$, it will be denoted by $mg_{k,\mathcal{L}}$.

Now, we are in conditions of formally introducing the concept of mg. First, we will define what a cost function is. Secondly, we will use this cost function to say, given two generalisations, which one is less general. Finally, the mg will be presented.

Definition 13 (cost function) *Let (X, d) and $\mathcal{L}$ be a connected metric space and a pattern language, respectively. Given a finite set of elements $E \subset X$, we will say that $k(E, h) : 2^X \times \mathcal{L} \to \mathcal{Y}$ is a cost function, if $\mathcal{Y}$ is an ordered set and $k(E, h)$ returns the top element of $\mathcal{Y}$ when h does not cover E or h is not a connected set.*

As we will see later, the image set $\mathcal{Y}$ will usually be the real numbers or a subset of it which will be endowed the usual order. In addition, most of the $k(E, h)$ functions that we are going to use, will be expressed as the sum of the auxiliary functions $c(h)$ (it measures how complicated the pattern is) and $c(E|h)$ (it measures how the pattern fits the data E). Following the MML/MDL formulation, we state no restrictions on $c(h)$ but we are interested on $c(E|h)$ which are defined in terms of distances. This is so the distances are the basis of the underlying metric space and "fitness" must be defined in terms of distances (more precisely distances to the border of the set). And this is so as well because for a binary generalisation operator, if $c(E|h)$ could be defined by expressing the examples, we would have that $c(h)$ would be usually much greater than $c(E|h)$. A coherent cost function $k(E, h)$ should satisfy Properties 1 and, $2a$ or $2b$ which we present below:

- **(1) The existence of a minimum:** given a $k(E, h)$ function, for every finite set of elements E, there always exists a pattern $h' \in \mathcal{L}$ such that $k(E, h') \leq k(E, h)$ for every $h \in \mathcal{L}$.
- **($2a$) Covering the sample:** given two hypothesis h_1 and h_2 belonging to $\mathcal{L}$, if $h_1 \subset h_2$ and $c(h_1) = c(h_2)$ then, $c(E|h_1) \leq c(E|h_2)$.
- **($2b$) Covering the sample using the minimum hypothesis:** given h' and h two hypothesis belonging to $\mathcal{L}$ and E a finite set of elements, such that $h' \subset h$ and $k(E, h')$ performs the minimum. If $c(h') = c(h)$, then $c(E|h') \leq c(E|h)$.

Generally speaking, Property 1 concerns the existence of a pattern which performs an optimal trade-off between complexity and overfitting. The rest of them focus on overfitting only. Property $2a$ means, if we have two equally "complicated" hypothesis, which one should be chosen? Intuitively, the one fitting the sample better. However, this might be difficult to achieve sometimes. Then, we could relax this condition by forcing that only the optimal hypothesis satisfies it. This yields $2b$. In other words, if h' is an optimal hypothesis covering E, then there should not exist any equally complicated hypothesis included in h'.

Before going on, let us introduce some interesting definitions of $k(E, h)$ and show how they work. Concretely, we are focusing on those cost functions expressed in terms of $c(h)$ and $c(h|E)$. As $c(h)$

measures how complex a pattern is, this function will strongly depend on the sort of data and the pattern space $\mathcal{L}$ we are handling. For instance, consider a closed interval. If the generalisation of two real numbers is a closed interval containing them, then the complexity of the interval could be its length. On the contrary consider a graph, if the elements to be generalised are two graphs, and the generalisation is based on the idea of common subgraph, then the number of cycles in the shared subgraph could be a measure of its complexity (see Table 2). On the other hand, $c(E|h) = \sum_{\forall\, e \in E} c(e|h)$.

Sort of data	$\mathcal{L}$	$c(h)$	Example
Numerical	Closed intervals	Length of the interval	$c([a,b]) = \|a-b\|$
Finite lists over an alphabet of symbols Σ	Lists built from an alphabet Σ and a special alphabet V of variables.	Number of symbols	$c(X_0abX_1b) = 5)$
First order predicates	Herbrand base with variables	Number of different variables	$c(p(X,X,Y,a)) = 2$
"	"	Number of symbols	$c(p(X,X,Y,a)) = 5$
Any	Any	Constant function	$\forall h \in \mathcal{L},\ c(h) = k$

Table 2: Some definitions of the function $c(h)$ for several sorts of data.

Now, let us see some definitions of $c(E|h)$. All of them must be based on the underlying ditance. In fact, all the definitions we present here are based on the well-known concept of border of a set[6]. Intuitively, if a patttern h_1 fits better E than a pattern h_2, then the border of h_1 (∂h_1) will somehow be nearer to E than the border of h_2 (∂h_2) (see Figure below).

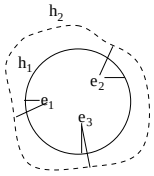

Figure 10: The pattern h_1 fits better E than h_2 and consequently, ∂h_1 is "nearer" to E than ∂h_2.

As the border of a set exists in every metric space, the function $c(E|h)$ will be much more independent of the sort of data we are handling than $c(h)$ is. Therefore, it motivates that several definitions of $c(E|h)$ can be employed for different sorts of data, as we show in Table 3.

In general, the functions $c(h)$ and $c(E|h)$ can be combined obtaining a more flexible way of defining $k(E,h)$ (see Example 7).

Example 7 *Let (R^2, d) be the plane of real numbers with the Euclidean distance, and the pattern language $\mathcal{L}$ is the finite union of rectangles in R^2. We define $c(h)$ as the minimum number of*

[6]Let (X,d) be a metric space. We will say that an element e belonging to set $A \subseteq X$ is a border point, if for every $\epsilon > 0$, $B(e,\epsilon)$ is not totally included in A. According to the standard notation, the border of a set A will be denoted by ∂A.

Sort of data	$\mathcal{L}$	$c(e\|h)$	Example
Any	Any	$\sum_{\forall e \in E} sup_{r \in \mathcal{R}} B(e,r) \not\subset h$	Left Figure 11
Any	Any	$\sum_{\forall e \in E} min_{e' \in \partial h} d(e,e')$	Middle Figure 11
Any	h represents an acotate set	$\sum_{\forall e \in E} min_{e' \in \partial h} d(e,e')$ $+d(e, \text{furthest point} \in \partial h)$	Right Figure 11

Table 3: Some definitions of the function $c(E|h)$.

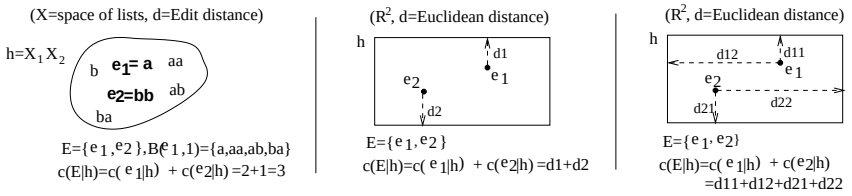

Figure 11: Some pictures illustrating each definition of $c(E|h)$ collected in Table 3. **(Left picture)** The space of finite lists with the Edit distance without substitution. The closed ball $B(e_1, 1)$ is included in the pattern h. **(Middle picture)** The pattern h is a rectangle covering e_1 and e_2. The terms d_i represents the distance from e_i to its nearest point in ∂h. **(Right figure)** This time, both the distance from e_i to its nearest and furthest point in ∂h (d_{i1} and d_{i2}) are taken into account.

rectangles required to specify h and $c(h|E)$ is the second function collected in Table 3. If $E = \{e_1 = (0,0), e_2 = (1,1), e_3 = (2,1)\}$ and $h = ABCD \cup DEFG$ (see Figure 12), then $c(h) = 2$, $c(e_1|h) = 0$, $c(e_2|h) = 3/2\sqrt{2}$, $c(e_3|h) = 3/2$ and finally, $k(E,h) = 2 + 3/2(1+\sqrt{2})$.

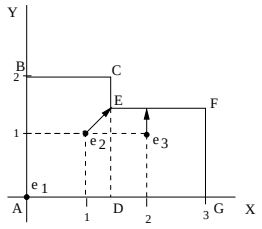

Figure 12: A possible cost function for R^2, considering the Euclidean distance and the finite union of rectangles in R^2 as the pattern language.

The definition of the *mg* will be based on the concept of cost function. Recall that before introducing the *mg*, we need to establish, given two generalisations, which one is the less general.

Definition 14 (Generality relation) *Let (X,d) and $\mathcal{L}$ be a connected metric space and a pattern language, respectively. Given a finite set of elements $E \subset X$, a cost function $k(E,h)$ and two generalisations expressed by the patterns h_1 and h_2, we will say that h_1 is less general than h_2 wrt. E, if $k(E,h_1) < k(E,h_2)$.*

Example 8 *Let $k(E,h) = c(h) + c(E|h)$ be a cost function where $c(h)$ is the constant function and $c(E|h)$ is the second function collected in Table 3. Imagine that the metric space is $\mathcal{R}^2$ with the Euclidean distance and $\mathcal{L}$ is the set of all rectangles in $\mathcal{R}^2$. Then, according to the definition above, the generalisation G_1 depicted on the right of Figure 7 is less general than the generalisation G_2, since $k(G_1, \{A, B\}) < k(G_2, \{A, B\})$.*

Definition 15 (Minimal generalisation) *Let (X, d) and $\mathcal{L}$ be a connected metric space and a pattern language, respectively. Given a finite set of elements $E \subset X$ and a cost function $k(E, h)$, we will say that the generalisation expressed by the pattern h is a minimal generalisation for $k(E, h)$ in $\mathcal{L}$ (denoted it by $mg(E)_{k,\mathcal{L}}$), if $k(E, h) \leq k(E, h')$, for every $h' \in \mathcal{L}$.*

Example 9 *Following with Example 8 and the definition above, it is quite immediate that the square G_1 is one of the minimal generalisations of $E = \{A, B\}$ for the cost function k in the pattern language $\mathcal{L}$ used in the example.*

Definition 16 (Minimal generalisation operator) *Let (X, d) be a connected metric space and let Δ be a binary generalisation operator defined in X using the pattern language $\mathcal{L}$. Given a finite set of elements $E \subset X$ and a cost function $k(E, h)$, we will say that Δ is a minimal generalisation operator for k in $\mathcal{L}$, if*

$$\Delta(E) = mg(E)_{k,\mathcal{L}}, \textit{for every finite set } E \in X.$$

Example 10 *Considering Example 9 and restricting E to a binary set (e.g. $E = \{A, B\}$), Δ is a minimal binary generalisation operator for k in $\mathcal{L}$, if for every $E = \{A, B\}$, Δ returns a rectangle G such that A and B are placed at the border of G.*

To summarise and according to our proposal, extending the concept of *lgg* for several sorts of data requires generalisations to be associated to the same pattern language $\mathcal{L}$, to talk about "minimal" instead of "least" generalisation (mg) and to use a cost function to quantify how general a generalisation is.