Similarity-based Heterogeneous Neuron Models

Lluís A. Belanche Muñoz[1]

Abstract. This paper introduces a general class of neuron models, accepting heterogeneous inputs in the form of mixtures of continuous (crisp or fuzzy) numbers, linguistic information, and discrete (either ordinal or nominal) quantities, with provision also for missing information. Their internal stimulation is based on an explicit *similarity relation* between the input and weight tuples (which are also heterogeneous). The framework is comprehensive and several models can be derived as instances –in particular, two of the commonly used models are shown to compute a specific similarity function provided all inputs are real-valued and complete. An example family of models defined by composition of a Gower-based similarity with a sigmoid function is shown to lead to network designs (*Heterogeneous Neural Networks*) capable of learning from non-trivial data sets with a remarkable effectiveness, comparable to that of classical models.

1 INTRODUCTION

Artificial Neural Networks (ANN) [1] have been successfully applied to a variety of fields –specially to Pattern Recognition– and constitute a class of models amenable to learn non-trivial tasks from representative samples. When exposed to a supervised training process, they build an internal representation of the underlying target function by combining certain parameterized base functions (PBF), either in local models as in Radial Basis Function ANN (RBF) or making up a global model as in the MultiLayer Perceptron (MLP). In both cases, the network relies in the representation capacity of the PBF (that is, of the neuron model) as the cornerstone for a good approximation.

A marked shortcoming of the neuron models existent in the literature is the difficulty of adding prior knowledge (either of the problem to be solved or of the data themselves) to the model; a neuron is used as a mapping from $\mathbb{R}^n$ to $\mathbb{R}$. For such a blind processing element, part of the task is to find a structure in the data, to transform it from a *raw form* (possibly in a space of high dimension, only here and there covered by example cases) to a new space (the *hidden space*, or space spanned by the hidden units) in such a way that the problem is simpler when projected to it. The same processing is repeated for all the subsequent hidden layers. This is true at least for the most widespread PBF: that used in the MLP –basically a scalar product between the input and weight vectors plus an offset, followed by a squashing function– and that used in RBF –a distance metric followed by a localized response function. The task of the hidden layer(s) is to find a new, more convenient representation for the problem *given* the data representation chosen, a crucial factor for a successful learning process that can have a great impact on generalization ability [2].

The appealingly simple and general "physical similarity" computation performed by a neuron working on scalar product or Euclidean distance has already been put in doubt as being so general or conceptually right in all situations [3]. The term physical similarity is coined to reflect the fact that this measure interprets the collection of input variables, whatever they are meant to represent, as the coordinates of a point in n-dimensional Euclidean space. Patterns $\vec{x}$ that are similar to a neuron with weight vector $\vec{w}^{\,i}$ (because the inner product $\vec{x}\cdot\vec{w}^{\,i}$ is high or $||\vec{x}-\vec{w}^{\,i}||$ is low[2]) need not be similar in the eyes of the user. The learning task consists precisely on this: to show the network what is the actual similarity relation. If this knowledge (assumed relevant) is supplied to a neural network or included in its design, it will not have to be discovered. In theory, ANN design follows the principle:

Principle 1.1 *Similar patterns should yield similar outputs.*

However, what "similar patterns" means is problem-dependent, and only in counted occasions will coincide with the fixed interpretation of similarity that the network is going to perform. There is a more basic principle [1] to be satisfied:

Principle 1.2 *Similar patterns should have similar representations.*

The fulfillment of this principle easies the fulfillment of the former, although it does not directly imply it, because neuron models will make use of their own fixed physical measure on the representations. The transformation to another representation form in which patterns requiring similar *problem* responses are indeed similar to one another for the network is performed, in theory, by the hidden layer(s) in multilayer networks (Fig. 1). Quoting [3]:

> "... if we use a sequence of such transformations [the hidden layers], each involving certain non-linearities ... we can entirely rearrange the similarity relations among the original input vectors."

This view is conceptually right and task-independent. In practice, it may be that for complex transformations several layers are needed, or very many neurons per layer if the maximum number of hidden layers is restricted to one. This gives support to a more precise formulation of neuron models as similarity computing devices, so that if more information is implanted into its design –at least for the input layer– capturing better the required similarity in the input space, the task of computing the required transformations may be simpler to learn.

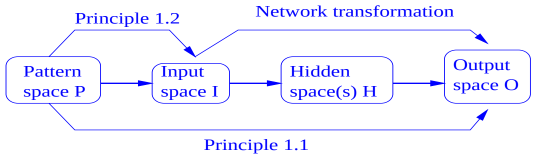

Figure 1. A Neural Network as a similarity transformer: The similarities in each space should be such that: $i)$ $s_I(i,i')$ captures the original likeness s_P in pattern space, and $ii)$ $s_I(i,i') \Rightarrow s_H(h,h') \Rightarrow s_O(o,o')$.

[1] Secció d'Intel·ligència Artificial. Dept. de Llenguatges i Sistemes Informàtics. Universitat Politècnica de Catalunya. c/Jordi Girona Salgado 1-3, 08034. Barcelona, Spain. e-mail: belanche@lsi.upc.es

[2] These quantities are functionally related for normalized vectors.

In principle, a feed-forward ANN can represent any mapping to a desired degree of accuracy in a high-dimensional space. In practice, a pre-processing scheme is often applied to the data samples to ease the task. Attention is specially put on continuous variables, usually (linearly) scaled to the unit interval or to zero-mean, unit standard deviation. This has beneficial effects by avoiding excessively high (absolute) weight values, and constitutes a rough means to equal the initial impact of variables.

In many important domains from the real world, objects are described by a mixture of continuous and discrete variables, usually containing missing information and characterized by an underlying vagueness, uncertainty or imprecision. For example, in the well-known UCI repository [4] over half of the problems contain *explicitly declared* nominal attributes, let alone other discrete types or fuzzy information, usually unreported. This heterogeneous information has to be encoded (or better, cast) in the form of real-valued quantities, although in most cases there is enough domain knowledge to design a more specific measure. Without the aim of being exhaustive, the commonly used methods [5, 2] are the following:

Ordinal variables coded as real-valued or using a *thermometer*.
Nominal variables coded using a 1-out-of-c or c-1 representation.
Missing information is difficult to handle, specially when the lost parts are of significant size. It can be either removed (the entire case) or "filled in" with the mean, median, nearest neighbour or by adding another input equal to one only if the value is absent. Statistical approaches need to make assumptions about or model the input distribution itself [2], or are computationally intensive[6].
Vagueness and uncertainty are considerations usually put aside.

These encodings being intuitive, it is not clear what is the precise effect on network performance, because of the change in input distribution, the increase (sometimes acute) in dimension and other subtler mathematical effects derived from imposing an artificial order or extending one to an infinite continuum. In the ANN paradigm, the inherent difficulty of the learning task can be exacerbated by having the network to discover the new correlations that stem from an structured increase in the number of inputs. In all these situations, the data (and with them, the problem itself) are being adapted to the neuron model (or, worse, to the learning algorithm), and not otherwise, as would seem more plausible. A reasonable solution –and the one followed in this work– is to have, on the one hand, a normalized neuron computation, *instead* of normalizing the inputs and, on the other, the neuron computation is explicitly defined in the form of a heterogeneous similarity measure.

2 SIMILARITY MEASURES

Let us represent patterns belonging to a space X (of which nothing is assumed, apart that $X \neq \emptyset$) as a vector $\vec{x}_i$ of n components, where each component x_{ij} represents the value of a particular feature (descriptive variable) a_j for object i, from a predefined set of features $A = \{a_1, a_2, \ldots, a_n\}$. A *similarity measure* is a unique number expressing how "like" two given objects are, given only the features in A [7]. Let us denote by s_{ij} the similarity between $\vec{x}_i$ and $\vec{x}_j$, that is, $s : X \times X \to \mathbb{R}^+ \cup \{0\}$ and $s_{ij} = s(\vec{x}_i, \vec{x}_j)$.

Definition 2.1 *A similarity measure fulfills the following properties:*

1. Non-negativity. $s_{ij} \geq 0 \;\; \forall \vec{x}_i, \vec{x}_j \in X$
2. Symmetry. $s_{ij} = s_{ji} \;\; \forall \vec{x}_i, \vec{x}_j \in X$
3. Boundedness. *There is a maximum similarity: that of an object with itself.* $\exists s_{max} \in \mathbb{R}^+ : \; s_{ij} \leq s_{max} \;\; \forall \vec{x}_i, \vec{x}_j \in X$
4. Minimality *(Reflexivity in the strong sense).*
$$s_{ij} = s_{max} \Leftrightarrow \vec{x}_i = \vec{x}_j \;\; \forall \vec{x}_i, \vec{x}_j \in X$$
5. Semantics. *The semantics of $s_{ij} > s_{ik}$ is that object i is more similar to object j than is to object k.*

A possible (arbitrary) value is to have $s_{max} = 1$. If, in property (4.), we replace the $\Leftrightarrow$ by $\Leftarrow$, so that two different objects can be assigned s_{max} –as if they were the same object– then we obtain a *pseudo-similarity* measure. The way different partial similarities should be combined to give an overall score is an elusive topic. An aggregation operator fulfills also a *semantic* role, as a a means to express functional or psychological aspects of similarity. Note that its definition is made easier by the fact that the partial values are, by definition, normalized. For $z \in \mathbb{R}, n \in \mathbb{N}^+$ and an operator T, let:

$$z^n[T] = \begin{cases} z & \text{if } n = 1 \\ T(\underbrace{z, z, \ldots, z}_{n \; times}) & \text{if } n > 1 \end{cases} \tag{1}$$

Definition 2.2 *Consider a collection of n quantities, grouped as a vector $\vec{s} = \{s_1, s_2, \ldots, s_n\}$, $s_i \in [0, s_{max}] \subset \mathbb{R}$. A similarity aggregation operator is a function $\Theta_s : [0, s_{max}]^n \to [0, s_{max}]$, fulfilling:*

i) *Minimality.* $\Theta_s(\vec{s}) = 0 \Rightarrow \exists i : s_i = 0 \wedge (\forall i : s_i = 0) \Rightarrow \Theta_s(\vec{s}) = 0$.
ii) *Symmetry.* $\Theta_s(\vec{s}) = \Theta_s(\sigma(\vec{s}))$ *for any* $\sigma(\vec{s})$ *permutation of* $\vec{s}$.
iii) *Monotonicity.* $\Theta_s(\vec{s}) > \Theta_s(\vec{s}\,')$ *whenever there exists a* s_i, $1 \leq i \leq n$ *such that* $s_i > s_i' \wedge \forall j : 1 \leq j \leq n \wedge j \neq i : s_j = s_j'$.
iv) *Idempotency. For an (arbitrary)* s_i, $s_i^n[\Theta_s] = s_i, \forall n \geq 1$. *Note that this specifically includes the boundary values* $s_{max}^n[\Theta_s] = s_{max}$ *and* $0^n[\Theta_s] = 0$. *Idempotency implies* $s_i^{n+1}[\Theta_s] = s_i^n[\Theta_s], \forall n \geq 1$. *A less restrictive condition is the* relaxed *idempotency: (a) Idempotency for boundary values (b) The terms* $\{s_i^n[\Theta_s]\}_{n \geq 1}$ *forming a monotonic succession.*
v) *Cancellation law.* $\Theta_s(\{s_1, s_2\}) = \Theta_s(\{s_1, s_3\})$, $s_1 > 0 \Rightarrow s_2 = s_3$.
vi) *Continuity.* Θ_s *should be continuous in all its arguments so the reactions to small changes in* s_i *are not "jumpy". It also ensures that all possible values of* $\Theta_s(\vec{s})$ *can in principle be generated, that is,* $\forall s : s \in [0, s_{max}] : (\exists \vec{s}^* : \; s = \Theta_s(\vec{s}^*))$.
vii) *Compensativeness.* Θ_s *should be a Cauchy mean, i.e.,*
$$\min_i s_i \leq \Theta_s(\vec{s}) \leq \max_i s_i$$
A Cauchy mean is such that a good (bad) score s_i *can be compensated by a bad (good) score on another* s_j.
viii) *If some components are missing, the value of* $\Theta_s(\vec{s})$ *is unaffected.*

The last two conditions are a way of introducing a specific semantics in the aggregation, and others should be possible. The adopted semantics in the above definition expresses that:

1. Even small contributions can only *add* something in favour for the overall measure;
2. The eventually missing pieces are regarded as *ignorance* and do not contribute in favour nor against the overall measure.

Proposition 2.1 *A measure s obtained from any such aggregation operator $s = \Theta_s(\vec{s})$, provided $\vec{s} = \{s_1, \ldots, s_n\}$ are also similarities s_i in X_i, fulfills the properties of Definition (2.1), making s a similarity measure in $X = X_1 \times X_2 \times \ldots \times X_n$.*

The following is a valid family of similarity aggregation functions:

$$\Theta(\vec{s};\vec{v}) = f^{-1}\left(\sum_{i=1}^{n} v_i f(s_i)\right) \tag{2}$$

where f is a strictly increasing function $f : [0, s_{max}] \to [0, s_{max}]$ such that $f(0) = 0$ and $f(s_{max}) = s_{max}$, and $\sum_{i=1}^{n} v_i = 1$, with $v_i > 0$. Consider $f(z) = z^m, m \in \mathbb{N}^+$. For $m = 1$ we get a weighted arithmetic mean, reducing to pure arithmetic average setting $v_i = \frac{1}{n}$.

Note that nothing is said about the linearity of the resulting measure. It is useful to introduce a class of functions that –regardless of its linear or non-linear character– maintains the similarity properties.

Definition 2.3 *A similarity keeping function (denoted $\check{s}$) is a strictly increasing continuous function $\check{s} : [0, s_{max}] \to [0, \check{s}_{max}]$, $\check{s}(0) = 0, \check{s}(s_{max}) = \check{s}_{max} > 0$ and $\exists \epsilon > 0, \forall x \leq \epsilon\ \check{s}(x) \leq x$.*

These functions keep (strict) monotonicity when composed to another (strictly) monotonic function and, since every strictly monotonic function is injective, applied to a measure obtained by aggregation, the original properties –of particular interest, (iii, vi)– are kept.

Proposition 2.2 *Let s be a similarity function in X defined on $[0, s_{max}]$ and $\check{s}$ a similarity keeping function (either linear or non-linear). Then $\check{s} \circ s$ is a similarity function in X.*

The final piece for constructing general similarity measures is a transformation operator to obtain a similarity out of a metric distance. If the original distance is normalized, the definitions can be refined.

Definition 2.4 *A similarity transforming function (denoted $\hat{s}$) is a strictly decreasing monotonic and continuous function $\hat{s} : [0, +\infty) \to [0, s_{max}]$ such that $\hat{s}(0) = s_{max}$ and $\lim_{z \to +\infty} \hat{s}(z) = 0$.*

Proposition 2.3 *Let d be a distance function defined on X and $\hat{s}$ a similarity transforming function (either linear or non-linear). Then $s(x, y) = \hat{s}(d(x, y))$ is a similarity function in X, for any $x, y \in X$.*

Proposition 2.4 *Let $\check{s}, \hat{s}$ be two similarity keeping and transforming functions. Then $\check{s} \circ \hat{s}$ is a similarity transforming function in $[0, \check{s}_{max}]$.*

Proposition 2.5 *Let $d : X \to [0, d_{max}]$ be a normalized distance on X and $\hat{s}$ a similarity transforming function. For an $\hat{s}$ such that $\hat{s}(d_{max}) = 0$, the function $s(x, y) = \hat{s}(d(x, y))$ is a similarity in X.*

2.1 Heterogeneous similarity measures

All the similarity measures are defined such that $s_{max} = 1$, so to allow a simpler and more general design of aggregation functions.

2.1.1 Ordinal variables

It is assumed that a *linear order* exists such that the values of the variable form a total linearly ordered space. Let $(\mathcal{O}, \preceq)$ be an ordered *ordinal* space and $x, y \in \mathcal{O}$, with the notation $(|\cdot|)$ for set cardinality. An equality relation $=$ is assumed so that a strict inequality:

$$x' \prec x \equiv x' \preceq x \wedge \neg(x' = x)$$

is defined in the usual way. Define now $\eta : \mathcal{O} \to [1, m] \subset \mathbb{N}^+$ as

$$\eta(x) = |x' \in \mathcal{O} : x' \prec x| + 1, \qquad x \in \mathcal{O}$$

Since the order $\preceq$ is linear, this is a bijection. Since every finite set is well-ordered, $\mathcal{O}$ has a first (x_f) and a last element (x_l), i.e., elements such that $\nexists x' \in \mathcal{O} : x' \prec x_f$ and $\nexists x' \in \mathcal{O} : x_l \prec x'$, respectively. Therefore, $\eta(x_f) = 1$, $\eta(x_l) = |\mathcal{O}| = m$ and:

$$\eta(x') = \eta(x) + 1 \Leftrightarrow x' = succ(x)$$

where the $\Leftarrow$ holds by definition of η and $\Rightarrow$ by the linear character of $\preceq$. Since every subset of a metric space is also metric, we can devise a distance in $\mathcal{O}$ resorting to the standard metric in $\mathbb{R}$, for $\eta(\mathcal{O}) \subset \mathbb{R}$, and making use of the fact that η is a bijection. Hence, for $x, y \in \mathcal{O}$, a normalized distance can be obtained as:

$$d(x, y) = \frac{|\eta(x) - \eta(y)|}{|\mathcal{O}| - 1} \in [0, 1] \tag{3}$$

and, by selecting $\hat{s}(z) = 1 - z$, $s(x, y) = \hat{s}(d(x, y))$ is a similarity measure in $\mathcal{O}$ (though others are possible). Note that, in a working implementation, the set $\mathcal{O}$ can be safely replaced by $[1, m] \subset \mathbb{N}^+$ as long as its elements are not arithmetically operated beyond (3).

2.1.2 Nominal variables

The most basic similarity measure for these variables is the overlap. Let $\mathcal{N}$ be a *nominal* space and $x, y \in \mathcal{N}$.

$$s(x, y) = \begin{cases} 1 & \text{if } x = y \\ 0 & \text{if } x \neq y \end{cases} \tag{4}$$

Proposition 2.6 *The function defined in (4) is a similarity in $\mathcal{N}$.*

2.1.3 Continuous variables

Let $x, y \in \Gamma = [r^-, r^+] \subset \mathbb{R}, r^+ > r^-$. The standard metric in $\mathbb{R}$ is a metric in Γ. A normalized distance can be obtained by setting:

$$d(x, y) = \frac{|x - y|}{\sup_{x,y \in \Gamma} |x - y|} \in [0, 1] \tag{5}$$

which is the standard distance weighted by the maximum deviation. Any $s(x, y) = \hat{s}(d(x, y))$ is a similarity measure in Γ. In particular, the family $\hat{s}_0(z) = (1 - z^d)^\alpha, 0 < d \leq 1, \alpha \geq 1$ can be used.

2.1.4 Fuzzy variables

The word *fuzzy* is viewed in this work taking the *epistemic* interpretation of a fuzzy set, i.e., as describing the vague observation of a theoretically crisp object. We begin by defining a basic concept:

Definition 2.5 *A fuzzy number in X (the reference set) is a convex and normalized fuzzy set F, with piecewise continuous μ_F, such that $\exists! x \in X : \mu_F(x) = 1$. Symmetry of μ_F is not required.*

Let $\mathbb{F}_n(X)$ be the (crisp) set of all the fuzzy numbers in X, where X is assumed a continuum. Since it is unlikely that collected data come already in a fuzzyfied form, a *fuzzyfication* process has to be introduced. In Fuzzy Control, crisp values x_0, in absence of other information, are transformed onto a fuzzy number of the form $\mu_{x_0}(x) = \delta(x = x_0)$. In this work, however, a numerical variable will be considered a fuzzy number (and treated as such) *only* if additional information is available; otherwise it is considered as continuous. One of the forms this information can take is about the reliability of measured values, which can be translated into the fuzzy number's spread. For every $x \in \mathbb{R}$, let $\tilde{x}$ be a fuzzy number of a given form centered at x with a spread proportional to x. The transformation:

$$\tilde{\cdot} : \ x \in \mathbb{R} \xrightarrow{\sim} \tilde{x} \in \mathbb{F}_n(\mathbb{R}) \tag{6}$$

associates a fuzzy number $\tilde{x}$ to x as $\mu_{\tilde{x}} = \mu_{[x,\sigma(x)]}$ –where x is the center and $\sigma(x)$ the fuzzyness– is the membership function of a fuzzy number expressing a *degree of uncertainty*, an interpretation proposed by Zadeh when introducing possibility theory [8]. The membership $\mu_{\tilde{x}}(u)$ represents the degree of possibility that x has value u. Domain knowledge can be added to refine the transformation; e.g., if x comes from a measuring device known to be biased towards underestimating the actual value at most a 2%, and overestimating it at most a 1%, a (non-symmetric) fuzzy number as depicted in Fig. 2 can be derived.

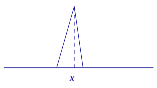

Figure 2. Example form for $\mu_{\tilde{x}}$ (not to scale).

Given two fuzzy numbers, the question is: how similar are they? The *possibility* measure expresses co-occurrence or simultaneity of two vague propositions, with a value of one standing for absolute certainty. For two fuzzy sets $\tilde{A}, \tilde{B}$ possibility is defined as:

$$\Pi_{\tilde{A}}(\tilde{B}) = \sup_{u \in X} \left(\mu_{\tilde{A} \cap \tilde{B}}(u)\right)$$

where $\mu_{\tilde{A} \cap \tilde{B}}(u) = \min\left(\mu_{\tilde{A}}(u), \mu_{\tilde{B}}(u)\right)$. Note that this measure is symmetric, in the sense $\Pi_{\tilde{A}}(\tilde{B}) = \Pi_{\tilde{B}}(\tilde{A})$.

Proposition 2.7 *Given $X, Y \in \mathbb{F}_n(\Gamma), \Gamma \subset \mathbb{R}$, and making use of the transformation in (6), the function $s : \mathbb{F}_n(\Gamma), \mathbb{F}_n(\Gamma) \to [0,1]$ defined as:*

$$s(x,y) = \Pi^*(\tilde{x}, \tilde{y}) \tag{7}$$

where $\Pi^*(\tilde{x}, \tilde{y}) = \Pi_{\tilde{x}}(\tilde{y})$, is a similarity measure in $\mathbb{F}_n(\Gamma)$.

Proof: Since $s \in [0,1]$, non-negativity and boundedness conditions (with $s_{max} = 1$) are easily met. There is also symmetry and a clear semantics. Minimality is met because, given $\alpha(x)$ is the same function for all $x \in \mathbb{R}$, it holds that $x = y \Leftrightarrow \tilde{x} = \tilde{y} \Leftrightarrow \Pi^*(\tilde{x}, \tilde{y}) = 1$.

2.1.5 Linguistic variables

The integration of numeric and qualitative information —the latter in the form of fuzzy sets— has been pointed out to have several advantages in the framework of pattern recognition. The addition of linguistic features to express vagueness both in the input patterns *and* in the inner workings of the system itself can lead to new architectures with enhanced expressiveness and flexibility. This approach considers variables by means of linguistic terms (words in the most basic situation). Since words are less precise than numbers, this approach enables the use of vague information, in cases where a precise quantity is unknown or there is a need or convenience to abstract it out. Among the possible forms such a fuzzy set can take, the most popular are the trapezoidal and bell-shaped. Let $supp(F)$ denote the *support* of a fuzzy set F and let $[\mathbb{R}]$ denote the set of all finite and closed real intervals: $[\mathbb{R}] = \{[a,b] \subset \mathbb{R} \; / \; a, b \in \mathbb{R}, a < b\}$.

Definition 2.6 *A fuzzy quantity in $\mathbb{R}$ is a convex normalized fuzzy set F with continuous μ_F such that $supp(F) \in [\mathbb{R}]$.*

Let $\mathbb{F}_q(X)$ be the (crisp) set of all the fuzzy quantities in X (note that a fuzzy number is a unimodal fuzzy quantity). Given $\tilde{A}, \tilde{B} \in \mathbb{F}_q(\Gamma), \Gamma \in \mathbb{R}$, with support sets $\Gamma_{\tilde{A}}, \Gamma_{\tilde{B}} \in \Gamma$, define their ratio as:

$$r(\tilde{A}, \tilde{B}) = \frac{\int_{\Gamma_{\tilde{A}} \cup \Gamma_{\tilde{B}}} \mu_{\tilde{A} \cap \tilde{B}}(u)du}{\int_{\Gamma_{\tilde{A}} \cup \Gamma_{\tilde{B}}} \mu_{\tilde{A} \cup \tilde{B}}(u)du} \tag{8}$$

This is the continuous version of a measure between general fuzzy sets [9]. For fuzzy quantities, it allows for a reasonably accurate assessment of similarity. It can be regarded as a generalization of a measure for crisp sets $A, B \in [\mathbb{R}]$, in which a value either belongs or not to the interval. The step to fuzzy sets makes the integral operator necessary. Note that, although (7) could in principle be applied, its behaviour in some cases (whenever $\#\{x \in X / \mu_{\tilde{A} \cap \tilde{B}}(x) = 1\} > 1$) makes it unfit as a similarity index (it does not fulfill minimality).

Proposition 2.8 *Given $\tilde{A}, \tilde{B} \in \mathbb{F}_q(\Gamma), \Gamma \in [\mathbb{R}]$, the function $s : \mathbb{F}_q(\Gamma), \mathbb{F}_q(\Gamma) \to [0,1]$ defined as in (8), for $(\cap = min, \cup = max)$, is a similarity in $\mathbb{F}_q(\Gamma)$.*

Proof: Symmetry, non-negativity and boundedness conditions (with $s_{max} = 1$) hold. There is also a clear semantics. Minimality is met because, as in crisp sets, the property $\tilde{A} \cap \tilde{B} = \tilde{A} \cup \tilde{B} \Leftrightarrow \tilde{A} = \tilde{B}$ holds.

It is a design decision –involving the existence of a continuous substrate and enough domain knowledge– which variables are amenable to be treated as linguistic and which as ordinal. A complete coverage and a correct degree of overlapping should be ensured.

2.2 A Framework for General Neuron Models

An heterogeneous space, denoted $\hat{\mathcal{H}}^n$, is defined as the Cartesian product of a number n of *source* sets, as follows. Let $X^{(\alpha)} = X_1 \times \cdots \times X_\alpha$ denote the Cartesian product with $X^{(0)} = \emptyset$. Consider now a collection of extended sets $\hat{\mathcal{R}}_1, \ldots, \hat{\mathcal{R}}_{n_r}$, where $\hat{\mathcal{R}}_i = \mathbb{R}_i \cup \{\mathcal{X}\}, \mathbb{R}_i \in [\mathbb{R}]$, and $1 \leq i \leq n_r$. Consider also collections of extended sets $\hat{\mathcal{O}}_1, \ldots, \hat{\mathcal{O}}_{n_o}$, with $\hat{\mathcal{O}}_i = \mathcal{O}_i \cup \{\mathcal{X}\}, 1 \leq i \leq n_o$, where each $\mathcal{O}_i$ is a finite and linearly ordered set, and of extended sets $\hat{\mathcal{N}}_1, \ldots, \hat{\mathcal{N}}_{n_n}$, with $\hat{\mathcal{N}}_i = \mathcal{N}_i \cup \{\mathcal{X}\}, 1 \leq i \leq n_n$, where each $\mathcal{N}_i$ is a finite and unordered set. Consider now the collection of n_f extended families of fuzzy sets of the form $\hat{\mathcal{F}}_1, \ldots, \hat{\mathcal{F}}_{n_f}$, where $\hat{\mathcal{F}}_i = \mathcal{F}_i \cup \{\mathcal{X}\}, \mathcal{F}_i \subset \mathbb{F}_q(\Gamma_i), \Gamma_i \in [\mathbb{R}]$, and $1 \leq i \leq n_f$. Note that $\mathbb{F}_n(\Gamma_i) \subset \mathbb{F}_q(\Gamma_i)$. In all cases, the extension is given by the special symbol $\mathcal{X}$, which denotes the *unknown* element (missing information) for which only equality is defined and behaving as an *incomparable* element w.r.t. any ordering relation. In these conditions, set:

$$\hat{\mathcal{H}}^n \equiv \hat{\mathcal{R}}^{(n_r)} \times \hat{\mathcal{O}}^{(n_o)} \times \hat{\mathcal{N}}^{(n_n)} \times \hat{\mathcal{F}}^{(n_f)} \tag{9}$$

with $n = n_r + n_f + n_o + n_n > 0$. According to (9), the elements of $\hat{\mathcal{H}}^n$ are general tuples of n components: real numbers, fuzzy numbers or quantities, ordinals, nominals and missing data. We call such a structure an *heterogeneous space*.

Definition 2.7 *Given $\vec{x} \in \hat{\mathcal{H}}^n$, an heterogeneous neuron or s-neuron with weight vector $\vec{w}^{\,i}$ is a function of the form:*

$$F_i(\vec{x}) = \{s(\vec{x}, \vec{w}^{\,i}), \vec{w}^{\,i} \in \hat{\mathcal{H}}^n\} \tag{10}$$

with s a similarity or pseudo-similarity measure in $\hat{\mathcal{H}}^n$.

An ANN making use of s-neurons is an *Heterogeneous Neural Network* or HNN. For complete data, by setting $\hat{\mathcal{H}}^n = \mathbb{R}^n \, (n = n_r)$, the standard neuron models –included in the above definition– are proven [10] to compute a similarity measure under mild conditions[3].

A basic but very useful heterogeneous measure can be devised using a Gower-like similarity index [11], an additive (2) similarity aggregation operator of (3), (4), (5) for $d = \frac{1}{2}, \alpha = 2$, (7) and (8) as:

$$s_G(\vec{x}_i, \vec{x}_j) = \frac{\sum_{k=1}^{n} s(x_{ik}, x_{jk})\, \delta_{ijk}}{\sum_{k=1}^{n} \delta_{ijk}} \tag{11}$$

where $\delta_{ijk} = \delta(x_{ik} \neq \mathcal{X} \wedge x_{jk} \neq \mathcal{X})$ is a binary function expressing whether the objects are *comparable* or not according to variable k. This treatment respects the adopted semantics by handling what is not known in a way it does not affect the known values.

[3] Proofs for the rest of Propositions can also be found therein.

Due to the existence of domains other than the real continuum, the presence of missing data and the eventual use of a non-differentiable s function, the general training procedure for the HNN is based on a *Breeder Genetic Algorithm* (BGA) [12, 13], a method in mid position between Genetic Algorithms (GA) and Evolution Strategies. While in GA selection is stochastic and meant to mimic Darwinian evolution, BGA selection is driven by a deterministic *breeding* mechanism, an artificial method in which only the best individuals —usually a fixed percentage τ of total population size μ— are selected to be recombined and mutated, as the basis to form a new generation. In addition, the BGA as used here does not need any coding scheme.

3 AN EXPERIMENTAL COMPARISON

A number of experiments are carried out to illustrate the validity of the approach. Five data sets are used, (*Pima Diabetes, Horse Colic, Credit Card* and *Solar Flares* taken from [5], *Sinus Exp* from [13]) chosen as representatives because of their richness in data types. The last problem has continuous variables only, *Solar Flares* has not any, and the other three display a mixture of variables, and varying percentages of missing information. They are all displayed in Table 1.

Table 1. Some basic characteristics of the data sets. #P: number of cases, %Def: default accuracy. The last column shows data heterogeneity.

Name	Type	#P	%Def	missing	In→Out	Data
Pima	C	768	65.1%	10.6%	$8 \to 2$	6R, 0N, 2I
Credit	C	690	55.5%	0.65%	$15 \to 2$	6R, 9N, 0I
Colic	C	364	61.5%	26.1%	$20 \to 3$	7R, 4N, 9I
Flares	R	1066	-	0.0%	$9 \to 3$	0R, 5N, 4I
Sinus	R	500	-	0.0%	$2 \to 1$	2R, 0N, 0I

C classification R regression R real N nominal I ordinal

For every data set, the available documentation is analysed to classify the variables as real, nominal, ordinal or integer, fuzzy number or linguistic. Originally missing information is also identified. For instance, there are two ordinal variables in *Pima Diabetes* (number of times pregnant and age in years) while many of the attributes have zero values that are physically impossible (e.g. diastolic blood pressure or body mass). Except for *Sinus Exp*, a logistic activation is applied to (11). The corresponding data set for the real-valued neurons is constructed using the mentioned techniques. Specifically, ordinals mapped to an equidistant linear scale, 1-out-of-c coding for nominal and an additional input for those variables with missing values.

All of the tested models use exactly the same experimental environment, in which everything is kept constant except the neuron model itself. The network architecture is fixed to one single layer of 8 neurons plus as many output neurons as required by the task, sigmoidal for classification problems and linear otherwise. The input variables are normalized to $[0, 1]$. This is not needed by the heterogeneous neurons because they compute a normalized measure, but is beneficial for the standard models. The output is not normalized. The weights (including biases and standard deviations) for the classical models are let to vary in $[-10, 10]$, a sufficiently wide range given the normalization chosen; the same interval is used for the hidden-to-output weights in all the networks. The BGA is set to the following parameters: $\mu = 100, \tau = 24$, EIR recombination with $\delta = 0.45$, and continuous mutation with $\rho = 0.5, k = 8$. This setting has been proven useful in the context of ANN optimization [13]. For each data set $n = 30$ independent runs are performed using the hold-out method, as follows. The original data set is split in three parts, for training (TR), validation (VA) and test (TE), 50-25-25 except for *Sinus* (20-20-60). The BGA task is to minimize MSE (mean square error) on the TR part, until 40,000 error evaluations are used –i.e. end of resources– in each run. We present performance results in Table 2.

Table 2. Mean performance results for the tested neuron models. Mean (best) accuracy is shown where appropriate.

	MLP		RBF		HNN	
Name	TR	TE	TR	TE	TR	TE
Pima	0.1374	**0.1527**	0.1825	0.1712	0.1077	0.1669
	81.0 (83.3)	78.3	73.7 (75.3)	77.4	86.2 (88.5)	76.8
Credit	0.0455	0.1751	0.1579	0.1867	0.0697	**0.1318**
	95.4 (97.1)	79.6	79.0 (80.6)	73.0	91.9 (93.6)	81.7
Colic	0.0597	0.1927	0.1435	0.1523	0.0584	0.1565
	86.3 (93.4)	66.3	67.5 (72.5)	67.7	90.3 (94.0)	68.9
Flares	0.2060	0.3925	0.2688	**0.2439**	0.2650	0.3017
Sinus	0.3189	0.3677	1.4526	1.8594	0.1814	**0.2451**

Approximation ability is computed as the final MSE on the TR part and generalization measured as the MSE on the TE part, using the net with lowest MSE on the VA part. This is the simplest approach to the comparison of different networks and it does not involve excessive computational overhead. It has been followed also to evaluate both approximation and generalization ability. Generalization errors in boldface are significant under a Mann-Whitney test [14] against the best of the other two models, at the 95% confidence level.

4 CONCLUSIONS

We have presented a general class of neuron models suited for heterogeneous pattern recognition. The rationale behind the approach consists in respecting the nature of the data and endowing the neuron models with an explicit similarity measure. The resulting networks are shown to learn from complex data sets in a satisfactory way. A preliminary studied model had been applied to a real problem in Environmental Sciences [15] with encouraging results.

ACKNOWLEDGEMENTS

This work is supported by the Spanish CICYT grant TAP99-0747.

REFERENCES

[1] S. Haykin. *Neural Networks: A Comprehensive Foundation.* MacMillan, 1994.

[2] C. Bishop. *Neural Networks for Pattern Recognition*. Oxford, 1995.

[3] D.E. Rumelhart, R. Durbin, R. Golden, Y. Chauvin. Backpropagation: The Basic Theory. In *Mathematical Perspectives of Neural Networks*. Smolensky, Mozer, Rumelhart (eds.), Lawrence Erlbaum, 1993.

[4] P.M. Murphy, D. Aha. UCI Repository of machine learning databases. UCI Dept. of Information and Computer Science, 1991.

[5] L. Prechelt. Proben1-A set of Neural Network Benchmark Problems and Benchmarking Rules. Fac. für Informatik. U. Karlsruhe T.R. 21/94.

[6] V. Tresp, S. Ahmad, R. Neuneier. Training Neural Networks with Deficient Data. In Cowan, Tesauro and Alespector (eds.). Advances in Neural Information Processing Systems 6, Morgan Kaufmann, 1994.

[7] J.L. Chandon, S. Pinson. *Analyse Typologique. Théorie et Applications*. Masson, 1981.

[8] L. Zadeh. Fuzzy Sets as a basis for a theory of possibility. *Fuzzy Sets and Systems*, 1: 3-28, 1978.

[9] C. Pappis, N. Karacapilidis. A comparative assessment of measures of similarity of fuzzy values. *Fuzzy sets and systems*, Vol. 56: 171-4, 1993.

[10] Ll. Belanche. A Theory for Heterogeneous Neuron Models based on Similarity. Tech. Rep. LSI-00-06-R. U. Politècnica de Catalunya, 2000.

[11] J.C. Gower. A General Coefficient of Similarity and some of its Properties. *Biometrics*, 27: 857-871, 1971.

[12] H. Mühlenbein, D. Schlierkamp-Voosen. Predictive Models for the Breeder Genetic Algorithm. *Evolutionary Computation*, 1(1), 1993.

[13] I. De Falco, A. Della Cioppa, P. Natale, E. Tarantino. Artificial Neural Networks Optimization by means of Evolutionary Algorithms. In *Soft Computing in Eng. Design and Manufacturing*. Springer, 1997.

[14] R. Steel, J. Torrie. *Principles and Procedures of Statistics. A Biomedical Approach.* McGraw-Hill, 1980.

[15] J.J. Valdés, Ll. Belanche, R. Alquézar. Fuzzy Heterogeneous Neurons for Imprecise Classification Problems. *Int. J. of Intel. Sys.*,15(3), 2000.