Neural Fuzzy Preference Integration using Neural Preference Moore Machines

Stefan Wermter

University of Sunderland

Centre of Informatics, SCET

St. Peter's Way

Sunderland SR6 0DD, United Kingdom

Email: stefan.wermter@sunderland.ac.uk

Phone: +44 191 515 3279

Fax: +44 191 515 2781

http://www.his.sunderland.ac.uk/~cs0stw/

Keywords: Neural preference Moore machines, Neuro/fuzzy integration, Neural preferences, Moore machines, Hybrid systems

Abstract

This paper describes preference classes and preference Moore machines as a basis for integrating different hybrid neural representations. Preference classes are shown to provide a basic link between neural preferences and fuzzy representations at the *preference class level*. Preference Moore machines provide a link between recurrent neural networks and symbolic transducers at the *preference Moore machine level*. We demonstrate how the concepts of preference classes and preference Moore machines can be used to interpret neural network representations and to integrate knowledge from hybrid neural representations. One main contribution of this paper is the introduction and analysis of neural preference Moore machines and their link to a fuzzy interpretation. Furthermore, we illustrate the interpretation and combination of various neural preference Moore machines with additional real world examples.

1 Introduction

Previously, there has been a lot of work on hybrid neural integration and combination [Reilly and Sharkey, 1992, Miikkulainen, 1993, Yager, 1994, Wermter, 1995, Medsker, 1995, Wermter et al., 1996, Dorffner, 1997]. Currently, it is an open question whether neural or symbolic approaches alone will be sufficient to provide a general framework for intelligent performance, e.g. for processing natural language [Dyer, 1991, Honavar and Uhr, 1994, Sun and Bookman, 1995, Wermter and Sun, 2000]. Therefore, hybrid neural approaches - which allow a vertical integration of principles from neuroscience and neural networks with symbolic interpretations - are currently examined and this paper is a contribution to this hybrid neural integration.

Hybrid neural/symbolic representations have been found advantageous in some contexts since different mutually complementary properties can be combined [Medsker, 1995, Wermter et al., 1996, Dorffner, 1997]. Symbolic representations have advantages with respect to easy interpretation, explicit control, fast initial coding, dynamic variable binding and knowledge abstraction. On the other hand, neural representations show advantages for gradual plausibility, learning, robust fault-tolerant processing, and generalization for new similar input. Since these advantages are mutually complementary, a hybrid neural architecture can be useful if different processing strategies have to be supported [Wermter et al., 1996].

Hybrid neural/symbolic methods have been shown to be able to reach a level where they can actually be further developed in real-world scenarios. A combination of symbolic and neural representations is possible in various *hybrid processing architectures*, which contain both symbolic and neural modules appropriate to a specific task [Wermter, 1997].

A *loosely coupled hybrid architecture* has separate symbolic and neural modules. The control flow is sequential in the sense that processing has to be finished in one module before the next module can begin. Only one module is active at any time and the communication between modules is unidirectional. An example architecture where the division of symbolic and neural work is loosely coupled has been described in a model for structural parsing within the SCAN framework [Wermter, 1995] combining a chart parser and feedforward networks. There are several other loosely coupled hybrid processing architectures for semantic analysis of database queries [Cheng et al., 1994] or dialog processing [Jurafsky et al., 1994].

A *tightly coupled hybrid architecture* contains separate symbolic and neural modules, and control and communication are via common internal data structures in each module. The main difference between loosely and tightly coupled hybrid architectures is common data structures which allow a bidirectional exchange of knowledge between two or more modules. For instance, systems for neural deterministic parsing [Kwasny and Faisal, 1992] and inferencing [Hendler, 1991] have been built where the control changes between symbolic marker passing and neural similarity determination.

In an *integrated hybrid architecture* there is no discernible external difference between symbolic and neural modules, since the modules have the same interface and they are embedded in the same architecture. The control flow may be parallel and the communication between symbolic and neural modules is via messages. Communication may be bidirectional between many modules, although not all possible communication channels have to be used. One example of an integrated hybrid architecture was developed for exploring integrated hybrid processing for spontaneous spoken language analysis [Wermter and Löchel, 1996, Wermter and Weber, 1997, Wermter and Meurer, 1997].

Besides all this work on hybrid neural/symbolic integration for certain applications there is very little work towards more general rigorous models of neural interpretation and symbolic integration. Much work has been guided by the particular application at hand. While it is very useful to integrate extensive task and domain knowledge into a system, there is a lack of more general interaction across different models and applications.

We aim at addressing this lack of principles for neural symbolic integration. In earlier work we have suggested preference Moore machines as one possible way of hybrid neural integration [Wermter, 1999]. In this article we build on this work and provide the detailed theoretical background for preference Moore machines, preference classes and their relationship to neural and symbolic representations. The paper is structured as follows. First we introduce the general concept of multidimensional neural preferences. The work and communication of neural networks relies on such multidimensional preferences. For ranking different preferences we introduce a new reference order and demonstrate that this order compares favorably with previously used orders like the dominance order.

Then we relate multidimensional neural preferences to multidimensional fuzzy set representations and show that the corner preference order on preference classes is a partial order. This allows us to rank different neural preferences and provides a basic link between neural preferences and symbolic fuzzy representations at the *preference class level*.

Then we introduce preference Moore machines and relate traditional symbolic transducers and recurrent networks to preference Moore machines. Preference Moore machines can be interpreted as a basic building block for neural network architectures as well as for symbolic architectures. Preference Moore machines provide a link between various recurrent networks and symbolic transducers at the *preference Moore machine level*.

Finally, we introduce operations like intersection and union on preference classes and prove that these operations on preference classes are commutative, associative and monotonic. Furthermore we give a general example of the integration based on preference classes. These operations provide a link between several neural or symbolic modules at the *system architecture level*. Finally we illustrate a various preference Moore machines with concrete examples, including from the Reuters news classification.

2 Neural Preferences and their Preference Value

Artificial neural networks receive analogous input from a number of network units (input layer) and they produce output to a number of network units (output layer). This motivates us to describe their external interface by a general multidimensional preference for input and/or output.

2.1 Preference, Preference Mapping and Order of a Preference

In this section we will describe the basic concepts of a preference and the order of a preference based on the concept of a layer of units in a network.

Definition 1 (Preference) *A preference is a continuous representation which is represented by an m-dimensional vector $p \in [0,1]^m$.*

Definition 2 (Preference Mapping) *A preference mapping is a mapping between preferences:* $[0,1]^n \rightarrow [0,1]^m$, *with* n, m *positive integer.*

Such a preference mapping could be a transformation of the input or a prediction of the next input based on the current input.

After we have defined a preference the question arises how we can rank preferences according to their strength. Thus, we need to specify an order for m-dimensional preferences in $[0,1]^m$. Within this m-dimensional space we will consider a preference as being large if the values of the individual vector elements are close to 1 or 0. In contrast, we will consider a preference as being small if the values of the individual vector elements are close to 0.5. This is our *goal criterion* for determining a partial order on preferences. In the subsequent sections we will consider different alternatives for eventually determining an appropriate preference order for m-dimensional preferences which relies on our goal criterion for ranking preferences.

2.2 Dominance Order of Multidimensional Preferences

We want to determine an appropriate preference order in $[0,1]^m$ and have to find a useful partial order[1] in $[0,1]^m$. A simple generalization of the order $\geq$ from $[0,1]$ to $[0,1]^m$ is possible with the dominance order. This m-dimensional order has already been widely used, for instance for certain applications in fuzzy reasoning [Kosko, 1992]. Therefore, we will consider this dominance order as a first alternative and will explain why the common dominance order is not yet sufficient for our preferences.

Definition 3 (Dominance Order) *Let* $a = (a_1, \cdots, a_m)$, $b = (b_1, \cdots, b_m)$ *be two* m*-dimensional preferences from* $[0,1]^m$. *Then the dominance order* $\geq_{dom}$ *is defined as* $(a_1, \cdots, a_m) \geq_{dom} (b_1, \cdots, b_m)$, *if* $a_i \geq b_i$ *for all* $i \in \{1, \cdots, m\}$. *Here* $\geq$ *is the usual order on real numbers.*

That is, a preference a from $[0,1]^m$ is greater or equal to a preference b from $[0,1]^m$, if each element a_i of a is greater than or equal to each corresponding element b_i from b. For instance it holds: $(0.9\ 0.8) \geq_{dom} (0.5\ 0.6)$ but also $(0.5\ 0.5) \geq_{dom} (0.1\ 0.1)$ However, as we specified

[1]A relation is a *partial order*, if it is reflexive, antisymmetric and transitive.

above in section 2.1, our goal criterion states, that values close to 1 or 0 from the interval $[0, 1]$ represent a large preference, values close to the center 0.5 a small preference. That is, a preference close to 1 or 0 is more reliable than a preference close to 0.5. As shown in the previous example, this is not the case for the dominance order since $(0.5\ 0.5) \geq_{dom} (0.1\ 0.1)$. Therefore, we cannot use this generalization of the usual order to the dominance order for our partial ordering of preferences.

Furthermore, for the dominance order it would also hold: $(0.9\ 0.1)$ and $(0.6\ 0.4)$ are not comparable because $0.9 > 0.6$ and $0.1 < 0.4$. However, in this case $(0.9\ 0.1)$ is clearly the larger preference for $(1\ 0)$. For a gradual, plausible and robust processing we want to support the better preferences. Therefore it is necessary that such preferences are comparable. This is another reason why we do not want to use this well known dominance order as a simple generalization of the usual order. Therefore we consider a different order which comes closer to our goal criterion.

2.3 Maximum Value Order

A different approach for determining an order in $[0, 1]^m$ is to determine the larger preference using the largest sum of the differences of all preference elements from the maximum value.

Definition 4 (Maximum Value Order) *Let $a = (a_1, \cdots, a_m)$, $b = (b_1, \cdots, b_m)$ be two m-dimensional preferences from $[0, 1]^m$. Then the maximum value order $\geq_{mv}$ is defined as $(a_1, \cdots, a_m) \geq_{mv} (b_1, \cdots, b_m)$, if $a_i \geq a_j$ for all $j \neq i$ and $b_k \geq b_l$ for all $l \neq k$ and $\sum_{j,j\neq i}(a_i - a_j) \geq \sum_{l,l\neq k}(b_k - b_l)$. Here $\geq$ is the usual order on real numbers.*

We consider our example from the dominance order. In contrast to the dominance order $(0.9\ 0.1)$ and $(0.6\ 0.4)$ are now comparable for the maximum value order and it holds: $(0.9\ 0.1) \geq_{mv} (0.6\ 0.4)$, because $0.9 - 0.1 > 0.6 - 0.4$. However, it also holds $(0.9\ 0.8) =_{mv} (0.5\ 0.6)$, because the distance from the maximum value is equal in both cases $(0.9 - 0.8 = 0.6 - 0.5 = 0.1)$, although $(0.9\ 0.8)$ is clearly the larger preference for $(1\ 1)$. Therefore, $\geq_{mv}$ is closer to our desired order but not yet quite appropriate for supporting multidimensional preferences since such clear differences cannot yet be considered. Therefore this order does not yet fulfill our goal criterion.

2.4 Reference Order

The dominance order and maximum value order do not yet fulfill our goal criterion that values close to 1 or 0 in the interval $[0,1]$ should support a larger preference, and values close to 0.5 a smaller preference. A larger preference in m-dimensional space can be defined by using a larger distance from a reference. Therefore, we define a preference a as being larger than another preference b, if a has a smaller distance to a reference.

Definition 5 (Reference Order) *Let $a = (a_1, \cdots, a_m)$, $b = (b_1, \cdots, b_m)$ be two m-dimensional preferences from $[0,1]^m$ and a reference $r = (r_1, \cdots, r_m)$ from $[0,1]^m$. Then the reference order $\geq_r$ is defined as: $(a_1, \cdots, a_m) \geq_r (b_1, \cdots, b_m)$, if $|r - a| \leq |r - b|$. Here $\leq$ is the usual order on real numbers and $|r - a| = \sqrt{(r_1 - a_1)^2 + \cdots + (r_m - a_m)^2}$ is the Euclidean distance of the preference a to the reference r.*

It is possible to determine multiple references, for instance the corner references from $\{0,1\}^n$. These corner references are particularly interesting since they allow a direct symbolic interpretation of the discrete vector values.

Definition 6 (Corner Reference Order) *If r is a corner reference $r = (r_1, \cdots, r_m) \in \{0,1\}^m$, then the reference order $\geq_r$ is based on the distance of two preferences from this corner reference. We call this special form of the reference order the corner reference order.*

That is, referring to a corner reference r, a preference a is greater than or equal to a preference b, if the distance of a to r is smaller than or equal to the distance of b to r. The corner reference can be interpreted as a strict sharp preference. Furthermore, for specific tasks one can chose certain subsets of corner preferences from the 2^m possible corner references. For instance, for disambiguations one may be interested in the corner references which have one vector element equal to 1 and all other elements equal to 0. Below we will specify that $ref(a)$ is the next corner reference with minimal distance to a currently considered preference a. We define in detail:

Definition 7 (Next Corner Reference) *The next corner reference $ref(a) \in \{0,1\}^m$, which is closest to $a \in [0,1]^m$ is determined for $i \in \{1, \cdots, m\}$ as:*

$$ref(a)_i = 0, if\ a_i < 0.5$$

$$ref(a)_i = 1, if\ a_i \geq 0.5$$

We consider our example which we used for the dominance order and the maximum value order: Now (0.9 0.1) and (0.6 0.4) are comparable and (0.9 0.1) $\geq_r$ (0.6 0.4), because (0.9 0.1) is closer to the next corner reference (1 0) than (0.6 0.4) to (1 0). Furthermore it holds: (0.9 0.8) $\geq_r$ (0.5 0.6), because the distance of (0.9 0.8) to (1 1) is smaller than the distance of (0.5 0.6) to a corner reference. Therefore the corner reference order fulfills the desired properties. The closer a preference is to a corner reference the greater the preference.

2.5 Preference Value of a Preference

Now we want to assign a preference value from the interval $[0, 1]$ to each preference a related to its next corner reference r in the m-dimensional space:

Definition 8 (Preference Value of a Preference (Strength of a Preference)) *Let $ref(a)$ be the next corner reference for a preference a in m-dimensional space. Let $distance(a, ref(a))$ be the Euclidean distance between a and $ref(a)$. Then we define the preference value of a preference a with respect to $ref(a)$ as:*

$$pref_{ref(a)}(a) = 1 - \frac{distance(a, ref(a))}{\frac{\sqrt{m}}{2}}$$

$\sqrt{m}/2$ is the maximum distance in m-dimensional space to the next corner reference, that is the distance from the center to the corner references. Therefore the values of $pref_{ref(a)}(a)$ are between 0 and 1. If a is close to its next corner reference $ref(a)$, then $pref_{ref(a)}(a)$ is close to 1. If a is close to the center reference $(0.5, \cdots, 0.5)$, then $pref_{ref(a)}(a)$ is close to 0. Figure 1 shows the preference values for the two-dimensional space. For each two-dimensional preference $(x\ y)$ the corresponding preference value z is shown. In general, the value $pref_{ref(a)}(a)$ has been given as the preference value of a preference a referring to a reference $ref(a)$. For instance, a preference can represent the representation of a categorization. Then the preference value would specify how strong a certain category assignment would be.

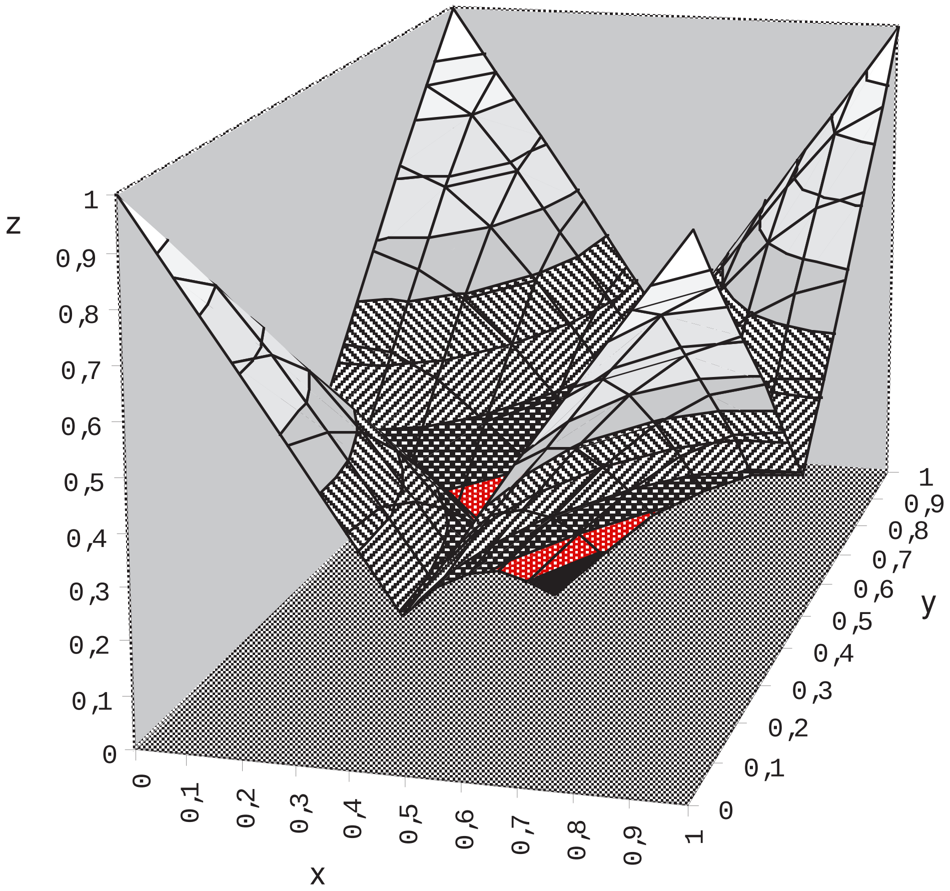

Figure 1: Preference values z of two-dimensional preferences (x y)

3 Neural Preferences and Fuzzy Set Representations

In the previous section we have defined neural preference and preference value in a general manner and we have explored several alternatives for a partial ordering on these preferences. The preference value of a preference describes the strength of this preference. In this section we want to explore the relationship between neural preferences and multidimensional fuzzy set representations. Since fuzzy set theory [Zadeh, 1965] is a well-established symbolic theory the relationship between neural preferences and fuzzy representations can help to develop principles of interaction between neural representations and symbolic representations. Before we focus on these relationships in more detail we briefly summarize some terminology from fuzzy set theory.

3.1 Terminology in Fuzzy Set Theory

In this section we summarize some important terms of fuzzy set theory [Zadeh, 1965, Klir and Folger, 1988]. These will serve as a basis from which we start our subsequent development of new concepts and relationships. To clarify the terms we summarize underlying well-known concepts.

Definition 9 (Fuzzy Set) *Let X be a universal set. Then the membership function m_A has the form $m_A : X \rightarrow [0,1]$ and defines a fuzzy set A. $[0,1]$ is the inclusive interval of real numbers from 0 to 1.*

Example: $X = \{0, \cdots, 100\}$ and m_{old} is a function $X \rightarrow [0,1]$, for which each element from X is assigned a value from $[0,1]$. For instance, $m_{old}(3) = 0.0$, $m_{old}(50) = 0.1$ and $m_{old}(80) = 0.95$. A person with the age 80 belongs with a membership value of 0.95 to the old persons, a person with the age 3 only with a membership value 0.0. Figure 2 shows a graphical representation of three different membership functions, which are modeled by simple triangle functions.

Definition 10 (Union of Fuzzy Sets) *The union of two fuzzy sets A and B is a fuzzy set $A \cup B$. A possible membership function is $m_{A\cup B}(x) = max[m_A(x), m_B(x)]$. Another possible membership function is $m_{A\cup B}(x) = m_A(x) + m_B(x) - m_A(x) * m_B(x)$.*

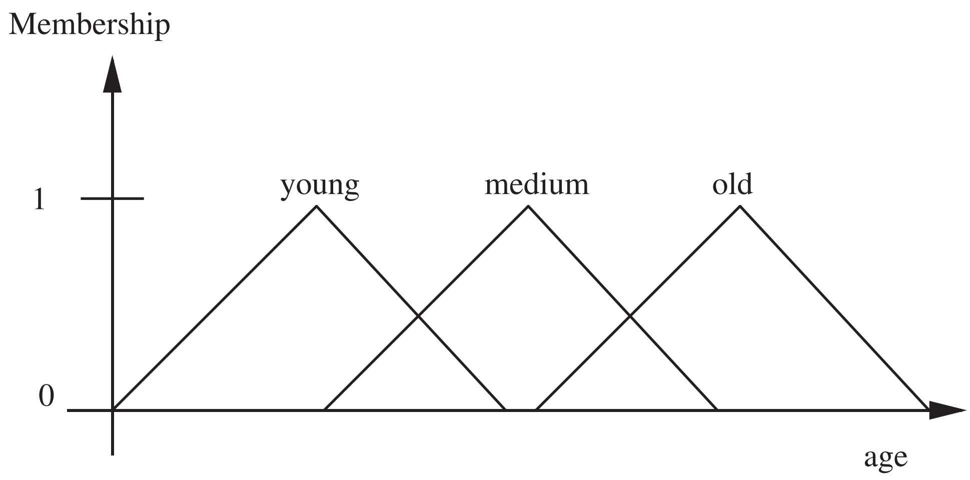

Figure 2: Fuzzy sets in one-dimensional space

In general there are basic axioms which must be fulfilled by operations for fuzzy union and which are summarized below:

Axiom 1 (Union of Fuzzy Sets) *The union of two fuzzy sets A and B is defined by a function $U : [0,1] \times [0,1] \rightarrow [0,1]$, which assigns each pair of membership degrees $m_A(x)$ and $m_B(x)$ a membership degree $m_{A \cup B}(x) = U(m_A(x), m_B(x))$, which belongs to an element of the union of A and B. U must obey the following axioms:*

U1: $U(0,0) = 0, U(0,1) = U(1,0) = U(1,1) = 1$, that is, U is like the union of sharp sets.

U2: $U(a,b) = U(b,a)$, that is, U is commutative.

U3: If $a \leq b$ and $c \leq d$, then $U(a,c) \leq U(b,d)$, that is, U is monotonic.

U4: $U(U(a,b)c) = U(a, U(b,c))$, that is, U is associative.

Here a, b, c are membership degrees $m_A(x) = a$, $m_B(x) = b$ and $m_C(x) = c$ for $x \in X$.

Definition 11 (Intersection of Fuzzy Sets) *The intersection of two fuzzy sets A and B is a fuzzy set $A \cap B$. A possible membership function is $m_{A \cap B}(x) = min[m_A(x), m_B(x)]$. Another possible membership function is $m_{A \cap B}(x) = m_A(x) * m_B(x)$.*

Again, there are basic axioms which must be fulfilled by operations for fuzzy intersection and which are summarized below:

Axiom 2 (Intersection of Fuzzy Sets) *The intersection of two fuzzy sets A and B is defined by a function $I : [0,1] \times [0,1] \rightarrow [0,1]$, which assigns each pair of membership degrees $m_A(x)$ and $m_B(x)$ a membership degree $m_{A \cap B}(x) = I(m_A(x), m_B(x))$, which belongs to an element of the intersection of A and B. I must obey the following axioms:*
I1: $I(1,1) = 1, I(0,1) = I(1,0) = I(0,0) = 0$, that is, I is like the sharp intersection.
I2: $I(a,b) = I(b,a)$, that is, I is commutative.
I3: If $a \leq b$ and $c \leq d$, then $I(a,c) \leq I(b,d)$, that is I is monotonic.
I4: $I(I(a,b)c) = I(a, I(b,c))$, that is I is associative.

Here a, b, c are membership degrees $m_A(x) = a$, $m_B(x) = b$ and $m_C(x) = c$ for $x \in X$.

3.2 Neural Preferences, Preference Classes and Multi-dimensional Fuzzy Sets

For focusing on the relationships between neural and fuzzy representations we consider a feedforward neural network and an elementary membership function of a fuzzy set. Let the input vector for such a network be a vector from $[0,1]^n$, which is mapped by the network to a single element from $[0,1]$. For fuzzy sets, a tuple from $[0,1]^n$ can be the input for a membership function which maps to the interval $[0,1]$. Then the activation value of a neural output element can be interpreted as the fuzzy value of a membership function. That is, at first we can identify a relationship between fuzzy representations and neural representations which refers to the interpretation of the single output element.

In the more general case, however, it must be possible to interpret networks with multiple output elements. Therefore we need to generalize fuzzy representations in such a manner that an n-dimensional input tuple can be assigned an m-dimensional value (see figure 3). This fundamental basis of fuzzy representations and neural representations is supported by the fact that both fuzzy sets and neural network vectors can be interpreted as points in m-dimensional space $[0,1]^m$.

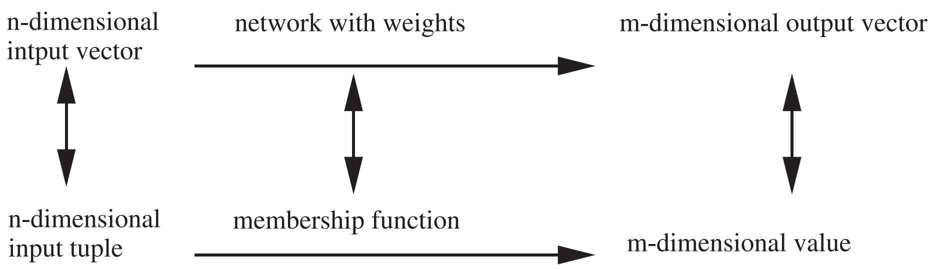

Figure 3: Relationship between fuzzy sets and neural networks as a preference mapping into a multi-dimensional set

How can we associate the preference value of a preference from section 2.5 with fuzzy sets and neural networks? An m-dimensional preference can be seen as an m-dimensional vector of a neural network as well as an m-dimensional fuzzy set. For a *neural interpretation* the preference value is a measure of how far away a neural preference is from a discrete symbolic corner vector, which represents the corner reference. For a *fuzzy interpretation* the preference value is measure for how far away a fuzzy set is from the corresponding symbolic sharp set which represents the corner reference.

The preference value of a preference can be viewed as a generalization of a membership function of a simple one-dimensional set to an m-dimensional set of preferences. Simple membership functions for one-dimensional sets are for instance triangle functions, which are defined in an overlapping manner on one-dimensional sets (e.g. age, height, etc. See also figure 2). If we compare figures 1 and figure 2, in each figure we can see a number of overlapping fuzzy sets which determine the membership value of an element of the domain set. If the preference value is high, then the preference is close to the corner reference and therefore the fuzziness is small. On the other hand, if the preference value is low then the preference is far away from the corner reference and the fuzziness is large. Because the preference can be an output vector of a neural network, the preference value (and the membership value) of an m-dimensional vector for a corresponding symbolic sharp set can be determined by the learning in a neural network.

For each preference in m-dimensional space we can specify a preference value in $[0, 1]$. Because of the definition of the corner reference order and the definition of the preference value, only preferences with the same corner reference can be compared. This property is useful, since preferences (0.9 0.3) and (0.3 0.9) for different references (1 0) and (0 1) would provide the same preference value $pref_{(1\ 0)}(0.9\ 0.3)$ and $pref_{(0\ 1)}(0.3\ 0.9)$ but it cannot be decided whether (0.9 0.3) or (0.3 0.9) are greater, since these preferences belong to different corner references. It is only possible to compare preferences which have the same corner reference. Those preferences which have the same distance to the same corner reference are judged as equal, for instance (0.9 0.8) and (0.8 0.9), because $pref_{(1\ 1)}(0.9\ 0.8) = pref_{(1\ 1)}(0.8\ 0.9)$. It is not possible to determine which of these preferences is greater and closer to the corner reference (1 1).

Our previous definition of the corner reference order is not yet a partial order. However, a partial order is a minimum requirement for the definition of all fuzzy sets with multi-dimensional goal domains [Klir and Folger, 1988]. The corner reference order is already transitive and reflexive, but it is not antisymmetric. For antisymmetry it must hold: if $x \geq_r y$ and $y \geq_r x$ then $x =_r y$. However $(0.8\ 0.9) \geq_r (0.9\ 0.8)$ and $(0.9\ 0.8) \geq_r (0.8\ 0.9)$, but both preferences are different. Therefore, we want to cluster those preferences which belong to the same next corner reference into one class. We want to define the corner reference order based on these classes.

Definition 12 (Class of Preferences) *Let $a = (a_1, \cdots, a_m)$ be a preference and $ref(a) = (r_1, \cdots, r_m) \in \{0, 1\}^m$ is next corner reference. Then the class of preferences of a is called $class(a)$ and contains all those preferences for next corner reference $ref(a)$, which have the same distance from $ref(a)$ as a.*

Definition 13 (Next Reference Order on Preference Classes) *Let $a = (a_1, \cdots, a_m)$, $b = (b_1, \cdots, b_m)$ be two preferences and their common next corner reference $r = (r_1, \cdots, r_m)$. Then the corner reference order on classes of preferences $\geq_{rc}$ is defined as follows: $class(a) \geq_{rc} class(b)$, if $|r - a| \leq |r - b|$. Here $\leq$ is the usual order for real numbers and*

$|r-a| = \sqrt{(r_1-a_1)^2 + \cdots + (r_m-a_m)^2}$ *is the distance of the preference a from reference r.* *We say that preference class* $class(a)$ *is greater than or equal to the preference class* $class(b)$.

Definition 14 (Preference Value of a Preference Class) *The preference value of a preference class* $class(a)$ *is the preference value of an arbitrary preference which belongs to this class.*

Theorem 1 *The corner reference order for preference classes is a partial ordering.*

Proof:

Let $a = (a_1, \cdots, a_m)$, $b = (b_1, \cdots, b_m)$ be two preferences with their corresponding preference classes $class(a)$ and $class(b)$. Let $r = (r_1, \cdots, r_m) \in \{0,1\}^m$ be their common next corner reference. We have to show reflexivity, antisymmetry and transitivity for the preference classes.

Reflexivity:

$class(a) \geq_{rc} class(a)$, since $|r-a| \leq |r-a|$.

Antisymmetry:

Let $class(a) \geq_{rc} class(b)$ and $class(b) \geq_{rc} class(a)$

Then $|r-a| \leq |r-b|$ and $|r-b| \leq |r-a|$

Then the distance of a and b is equal, that is they are in the same preference class, that is $class(a) =_{rc} class(b)$

Transitivity:

Let $class(a) \geq_{rc} class(b)$ and $class(b) \geq_{rc} class(c)$

Then $|r-a| \leq |r-b|$ and $|r-b| \leq |r-c|$ and it follows $|r-a| \leq |r-c|$, that is $class(a) \geq_{rc} class(c)$.

$\square$

The corner reference order for classes of preferences is a partial order which meets the particular requirements for a neural interpretation of preferences (multidimensional and uncertain close to 0.5) but also the general requirements for a fuzzy interpretation of preferences (at least partial order in goal domain) and also the general requirements of neural and symbolic integration (symbolic corner reference as a reference for classes of neural preferences). The preference value of a class of output preferences of a neural network can be understood as the membership degree of these output preferences for an m-dimensional fuzzy set which represents a reference (for instance a corner reference) in m-dimensional space. Figure 4 shows examples of four preference classes which have the same distance to their corresponding corner reference.

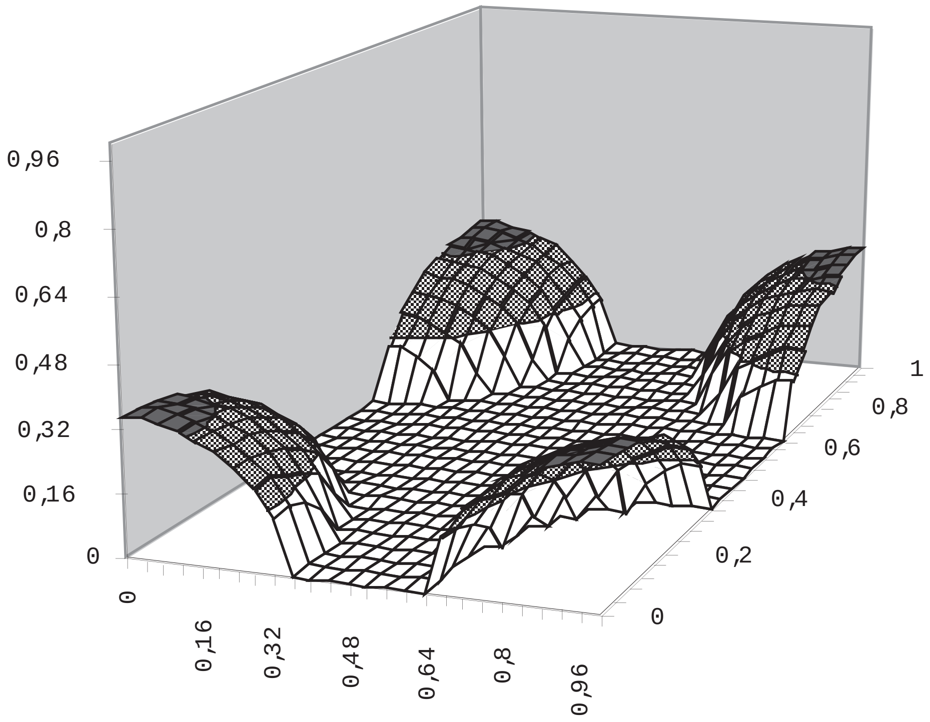

Figure 4: Classes of preferences in three-dimensional space

Another reason for the use of classes of preferences is based on symbolic processing. If a preference value for $(Feature1, Feature2)$ has to be specified, a single value, e.g. 0.8, can be given. This preference value corresponds to all those preferences which have the same corresponding distance from the specified corner reference. Therefore a class of preferences supports also the integration of symbolic and neural representations. A class of preferences represents a high-dimensional hypersphere of an unlimited number of preferences with the same distance from the specified corner reference.

4 Preference Moore Machines for Preference Mappings

In this section we consider preference Moore machines as one possibility to relate principles of symbolic computational representations and neural representations by means of preferences. We have chosen this type of machines since they are simple and widely applicable.

4.1 Synchronous Sequential Machines

One simple well-known and efficient method of representing previous context is the use of finite state automata and transducers [Hopcroft and Ullman, 1979]. Basically automata and transducers are in a certain context state and analyze a certain word (symbol). Then they transfer into a new state and potentially generate a new word (symbol). Using changing states it is possible to encode the sequential context.

Definition 15 (Synchronous Sequential Machine, Transducer) *A synchronous sequential machine M is a tuple $M = (I, O, S, sf, of)$, with*

1. *I, O finite, nonempty sets of input and output.*
2. *S nonempty set of states.*
3. *The function $sf : I \times S \to S$ is a state transition function.*
4. *The function of is the output function. If the output depends on the state and the input, the machine is a so-called Mealy machine with the output function $of : I \times S \to O$. If the output only depends on the state the machine is a so-called Moore machine with the output function $of : S \to O$. These synchronous sequential machines are sometimes called transducers.*

A sequential machine assigns an output and a new state to an input and an old state. This can be done for a whole sequence of inputs and states in discrete time. The set S is not necessarily finite [Booth, 1967] although this is assumed for finite machines. While automata or acceptors of languages decide whether a certain input belongs to the corresponding grammar, sequential machines are transducers which change their internal states dynamically for the inputs and previous states but they also provide an output for each input.

Mealy and Moore machines are slightly different. Moore machines determine the state first and afterwards this state is used to provide the output. In contrast, the output for a Mealy machine depends also directly on the input. However, it can be shown that for each Moore machine there is an equivalent Mealy machine and vice versa [Booth, 1967, Hopcroft and Ullman, 1979].

In our case we concentrate on Moore machines since the output of neural networks is usually based on the internal state. This holds for instance for feedforward networks or simple recurrent networks. While sometimes [Sun, 1995] a sequential machine has been used for modeling a single element of a neural network, we want to use a sequential machine as a description for a whole network. This is also motivated by the fact that real neuron systems can be seen as physical entities which perform state transitions [Churchland and Sejnowski, 1992].

It is possible to define state transition tables which assign each combination of input and current state a new output and a new state. That way a symbolic synchronous sequential machine is specified. If clear regularities are known and the number is limited, such tables can be found manually. However, the number of input and state combinations quickly gets so large that automatic procedures are necessary.

4.2 Synchronous Neural Preference Moore Machine

Traditional usual state transition tables are discrete and symbolic. Therefore they do not support gradual representations. For instance, input or state can be ambiguous and have different gradual preferences for different interpretations. For instance "meeting" could have a larger preference for the syntactic interpretation as a noun and a smaller preference for a verb form. Therefore we want to use preferences for input, output and states of such machines. These preferences should be able to take values from $[0, 1]^m$ so that multiple preferences can be represented and integrated.

We use an n-dimensional preference for the input and an m-dimensional preference for the output. Then we get a new synchronous machine which we will call a *preference Moore machine*. We want to describe such a synchronous sequential preference Moore machine which transforms sequential input preferences to sequential output preferences. We will see that simple recurrent networks or feedforward networks can be interpreted as neural preference Moore machines. Furthermore, we will show how symbolic and neural knowledge can be integrated quite naturally using preference Moore machines.

Definition 16 (Preference Moore Machine) *A preference Moore machine PM is a synchronous sequential machine, which is characterized by a 4-tuple $PM = (I, O, S, pf)$, with I, O and S non-empty sets of inputs, outputs and states. $pf : I \times S \to O \times S$ is the sequential preference mapping and contains the state transition function sf and the output function of. Here I, O and S are n-, m- and l-dimensional preferences with values from $[0,1]^n$, $[0,1]^m$ and $[0,1]^l$, respectively.*

The preference Moore machine realizes a sequential preference mapping, which uses the current state preference S and the input preference I to assign an output preference O and a new state preference.

Simple recurrent networks (SRN) [Elman, 1991] or plausibility networks [Wermter, 1995] have the potential to learn a sequential preference mapping $pf : I \times S \to O \times S$ based on input and output examples while traditional Moore machines or Fuzzy-Sequential-Functions [Santos, 1973] use manual encodings.

A simple recurrent neural network constitutes a neural preference Moore machine which generates a sequence of output preferences for a sequence of input preferences. Here internal state preferences are used as local memory. A feedforward network represents a neural preference Moore machine with a degenerated memory. A plausibility network [Wermter, 1995] constitutes a more general form of neural preference Moore machine.

On the one hand, we can associate a neural preference Moore machine in a preference space with its symbolic interpretation and on the other hand we can represent a symbolic transducer in a neural manner. Using the symbolic m-dimensional preferences and the corner reference order it is possible to interpret neural preferences symbolically as well as to integrate symbolic preferences with neural preferences. Each preference of a neural trajectory is a representative of its preference class and it is possible to assign a symbolic description as a corner reference together with a preference value. That way neural preferences can be interpreted symbolically. On the other hand, symbolic knowledge can be integrated with neural knowledge by associating a preference value with a symbolic corner reference. This preference value of the symbolic reference determines which neural preference class is associated with the symbolic reference.

5 Combination of Preferences and Preference Machines

Symbolic regular relations can be understood as top-down specification for symbolic Moore machines. On the other hand, a training set can be viewed as a bottom-up specification for neural preference Moore machines. In this section we want to look at the integration of preferences.

5.1 Preference Classes as the Basis for Combination

We want to examine a possible integration of symbolic and neural Moore machines. We will consider a single neural or symbolic Moore machine as a unit, whose input and output should be integrated. We need a common basis for an integration of a symbolic output of a symbolic Moore machine and a vector output of a neural Moore machine. Therefore, the question arises where a symbol and a vector representation can be combined so that an integration is possible. We suggest that a preference class could be a suitable connection between different symbolic and/or neural Moore machines (see also section 3.2). On the one hand, a special m-dimensional output vector of a neural network is part of an m-dimensional preference class. On the other hand, the vectors which have the same preference value for a given reference build an m-dimensional preference class.

If a preference value from $[0, 1]$ is specified for a symbolic sharp representation from $\{0, 1\}^m$ then a preference class is associated with a sharp symbolic representation. For instance the symbolic representation $(noun, not\ verb)$ can be associated with a preference value 0.8. This associates a preference class which favors $(noun, not\ verb)$ but only with a preference of 0.8

instead of 1. By means of these preference classes a symbolic discrete representation gets more gradual and it is associated with a set of preferences which can be interpreted at a neural level as well.

However, if one associates a neural vector representation from $[0,1]^m$ with its preference class, then each vector has a preference class. Therefore an abstraction from an individual vector is possible and this vector is summarized together with other vectors with the same preference value to a preference class. This preference class can be interpreted symbolically by specifying the symbolic representation and its preference value. Therefore, a preference class is a connection for an integration of symbolic and neural knowledge. The basic underlying integration possibilities are the operations of intersection and union of preference classes. That way it is possible to integrate different knowledge sources. Below we will focus on these operations on preference classes.

5.2 Preference Class Operations

We assume $[0,1]^n \rightarrow [0,1]^m$ be a mapping which associates input preferences with output preferences. For preference classes from $[0,1]^m$ we can define the operations for intersection and union. Let $a = (a_1, \cdots, a_m)$, $b = (b_1, \cdots, b_m)$ be two preferences from $[0,1]^m$ with their corresponding preference classes $class_{ref(a)}(a)$ and $class_{ref(b)}(b)$. Then let $pref_{ref(a)}(a)$ be preference value of a preference a for a reference $ref(a)$; similarly this holds for $pref_{ref(b)}(b)$. If it is clear that the reference of a preference is the next corner reference $ref(a) \in \{0,1\}^m$, we write for short $pref(a)$ rather than $pref_{ref(a)}(a)$ and $class(a)$ rather than $class_{ref(a)}(a)$. We will consider a preference class in a slightly modified compact notation as a pair of reference and preference value: $preference\ class = (reference, preference\ value)$. For instance, $((0\ 1), 0.3)$ is a preference class in the two-dimensional space which contains all preferences which have the preference value 0.3 for the reference $(0\ 1)$.

5.2.1 Union of Preference Classes

Definition 17 (Union of two m-dimensional Preference Classes) *The union of two preference classes $(ref(a), pref(a))$ and $(ref(b), pref(b))$ is a preference class which has* *the reference* $(max(ref(a)_1, ref(b)_1), \cdots max(ref(a)_m, ref(b)_m))$ *and the preference value* $max(pref_{ref(a)}(a), pref_{ref(b)}(b))$*. We call this union of preference classes PU:*

$$PU : PU(((ref(a)_1, \cdots ref(a)_m), pref(a)), ((ref(b)_1, \cdots ref(b)_m), pref(b)))$$

$$= ((max\{ref(a)_1, ref(b)_1\} \cdots max\{ref(a)_m, ref(b)_m\}), max\{pref(a), pref(b)\})$$

If the (symbolically interpretable) reference of two preference classes is equal then the reference will be kept and the union provides the preference class with the larger preference value. If the reference of two preference classes is different then the union is extended to the references. Then the union provides the united preference class with the larger preference value. Later we will see an illustrative example in section 6.4.

5.2.2 Intersection of Preference Classes

Definition 18 (Intersection of two m-dimensional Preference Classes)
The intersection of two preference classes $(ref(a), pref(a))$ *and* $(ref(b), pref(b))$ *is a preference class which has the reference* $(min(ref(a)_1, ref(b)_1), \cdots min(ref(a)_m, ref(b)_m))$ *and the preference value* $min(pref_{ref(a)}(a), pref_{ref(b)}(b))$*. We call this intersection of preference classes PI:*

$$PI : PI(((ref(a)_1, \cdots ref(a)_m), pref(a)), ((ref(b)_1, \cdots ref(b)_m), pref(b)))$$

$$= ((min\{ref(a)_1, ref(b)_1\} \cdots min\{ref(a)_m, ref(b)_m\}), min\{pref(a), pref(b)\})$$

If the (symbolically interpretable) reference of two preference classes is equal then the reference will be kept and the intersection provides the preference class with the smaller preference value. If the reference of two preference classes is different then the intersection is extended to the references. Then the intersection provides the intersected preference class with the smaller preference value. Later will will see an illustrative example in section 6.4.

5.3 M-dimensional Preference Classes and Fuzzy Sets

In section 3.1 we have specified a number of axioms for the union and the intersection of fuzzy sets. These axioms had to be fulfilled so that a definition of the union or the intersection for fuzzy sets was useful [Klir and Folger, 1988, Zimmermann, 1991]. In a similar manner we will now examine whether our operations for preference classes fulfill these axioms which are a basic precondition for a relationship of preference classes to fuzzy sets. The axioms are: 1) generalization of sharp sets, 2) commutativity, 3) monotonicity and 4) associativity.

Theorem 2 *Let $\geq_{rc}$ be the partial ordering for the m-dimensional preference space $[0,1]^m$. Then the union PU on preference classes fulfills the axioms generalization of sharp sets (PU1), commutativity (PU2), monotonicity (PU3), and associativity (PU4).*

Proof:

PU1:

$$\begin{aligned}
PU((ref(a),0),(ref(b),0)) &= ((max\{ref(a)_1,ref(b)_1\}\cdots max\{ref(a)_m,ref(b)_m\}),0)\\
PU((ref(a),1),(ref(b),0)) &= ((max\{ref(a)_1,ref(b)_1\}\cdots max\{ref(a)_m,ref(b)_m\}),1)\\
PU((ref(a),0),(ref(b),1)) &= ((max\{ref(a)_1,ref(b)_1\}\cdots max\{ref(a)_m,ref(b)_m\}),1)\\
PU((ref(a),1),(ref(b),1)) &= ((max\{ref(a)_1,ref(b)_1\}\cdots max\{ref(a)_m,ref(b)_m\}),1)
\end{aligned}$$

that is PU behaves like the union of sharp m-dimensional sets. Here $(ref(a),0)) = (ref(b),0))$ is the center of the m-dimensional preference space, $(ref(a),1))$ or $(ref(b),1))$ is a corner reference of the m-dimensional preference space.

PU2:

$$\begin{aligned}
& PU((ref(a),pref(a)),(ref(b),pref(b)))\\
= \; & ((max\{ref(a)_1,ref(b)_1\}\cdots max\{ref(a)_m,ref(b)_m\}),max\{pref(a),pref(b)\})\\
= \; & ((max\{ref(b)_1,ref(a)_1\}\cdots max\{ref(b)_m,ref(a)_m\}),max\{pref(b),pref(a)\})\\
= \; & PU((ref(b),pref(b)),(ref(a),pref(a)))
\end{aligned}$$

that is PU is commutative.

PU3:

Let $(ref(a), pref(a)) \leq_{rc} (ref(b), pref(b))$ and $(ref(c), pref(c)) \leq_{rc} (ref(d), pref(d))$. That is, a preference a is comparable with a preference b (for corner references: $ref(a) = ref(b)$) and the preference value of a is smaller than the preference value of b. For c and d the corresponding holds.

$$\begin{aligned}
& PU((ref(a), pref(a)), (ref(c), pref(c))) \\
= \;& ((max\{ref(a)_1, ref(c)_1\} \cdots max\{ref(a)_m, ref(c)_m\}), max\{pref(a), pref(c)\}) \\
\leq_{rc} \;& ((max\{ref(b)_1, ref(d)_1\} \cdots max\{ref(b)_m, ref(d)_m\}), max\{pref(b), pref(d)\}) \\
= \;& PU((ref(b), pref(b)), (ref(d), pref(d)))
\end{aligned}$$

that is PU is monotonic.

PU4:

$$\begin{aligned}
& PU(PU((ref(a), pref(a)), (ref(b), pref(b))), (ref(c), pref(c))) \\
= \;& ((max\{max\{ref(a)_1, ref(b)_1\}, ref(c)_1\} \cdots max\{max\{ref(a)_m, ref(b)_m\}, ref(c)_m\}), \\
& max\{max\{pref(a), pref(b)\}, pref(c)\}) \\
= \;& ((max\{ref(a)_1, max\{ref(b)_1, ref(c)_1\}\} \cdots max\{ref(a)_m, max\{ref(b)_m, ref(c)_m\}\}), \\
& max\{pref(a), max\{pref(b), pref(c)\}\}) \\
= \;& PU((ref(a), pref(a)), PU((ref(b), pref(b)), (ref(c), pref(c))))
\end{aligned}$$

that is PU is associative.

□

Theorem 3 *Let $\geq_{rc}$ be the partial ordering for the m-dimensional preference space $[0,1]^m$. Then the intersection PI on preference classes fulfills the axioms generalization of sharp sets (PI1), commutativity (PI2), monotonicity (PI3), and associativity (PI4).*

Proof:

PI1:

$$PI((ref(a),0),(ref(b),0)) = ((min\{ref(a)_1, ref(b)_1\} \cdots min\{ref(a)_m, ref(b)_m\}), 0)$$
$$PI((ref(a),1),(ref(b),0)) = ((min\{ref(a)_1, ref(b)_1\} \cdots min\{ref(a)_m, ref(b)_m\}), 0)$$
$$PI((ref(a),0),(ref(b),1)) = ((min\{ref(a)_1, ref(b)_1\} \cdots min\{ref(a)_m, ref(b)_m\}), 0)$$
$$PI((ref(a),1),(ref(b),1)) = ((min\{ref(a)_1, ref(b)_1\} \cdots min\{ref(a)_m, ref(b)_m\}), 1)$$

that is PI behaves like the intersection of sharp m-dimensional sets. Here $(ref(a),0)) = (ref(b),0))$ is the center of the m-dimensional preference space, $(ref(a),1))$ or $(ref(b),1))$ is a corner reference of the m-dimensional preference space.

PI2:

$$\begin{aligned} & PI((ref(a),pref(a)),(ref(b),pref(b))) \\ = \;& ((min\{ref(a)_1, ref(b)_1\} \cdots min\{ref(a)_m, ref(b)_m\}), min\{pref(a), pref(b)\}) \\ = \;& ((min\{ref(b)_1, ref(a)_1\} \cdots min\{ref(b)_m, ref(a)_m\}), min\{pref(b), pref(a)\}) \\ = \;& PI((ref(b),pref(b)),(ref(a),pref(a))) \end{aligned}$$

that is PI is commutative.

PI3:

Let $(ref(a), pref(a)) \leq_{rc} (ref(b), pref(b))$ and $(ref(c), pref(c)) \leq_{rc} (ref(d), pref(d))$. That is, a preference a is comparable with a preference b (for corner references: $ref(a) = ref(b)$) and the preference value of a is smaller than the preference value of b. For c and d the corresponding holds.

$$PI((ref(a),pref(a)),(ref(c),pref(c)))$$

$$
\begin{aligned}
&= ((min\{ref(a)_1, ref(c)_1\} \cdots min\{ref(a)_m, ref(c)_m\}), min\{pref(a), pref(c)\}) \\
&\leq_{rc} ((min\{ref(b)_1, ref(d)_1\} \cdots min\{ref(b)_m, ref(d)_m\}), min\{pref(b), pref(d)\}) \\
&= PI((ref(b), pref(b)), (ref(d), pref(d)))
\end{aligned}
$$

that is PI is monotonic.

PI4:

$$
\begin{aligned}
& PI(PI((ref(a), pref(a)), (ref(b), pref(b))), (ref(c), pref(c))) \\
= \; & ((min\{min\{ref(a)_1, ref(b)_1\}, ref(c)_1\} \cdots min\{min\{ref(a)_m, ref(b)_m\}, ref(c)_m\}), \\
& min\{min\{pref(a), pref(b)\}, pref(c)\}) \\
= \; & ((min\{ref(a)_1, min\{ref(b)_1, ref(c)_1\}\} \cdots min\{ref(a)_m, min\{ref(b)_m, ref(c)_m\}\}), \\
& min\{pref(a), min\{pref(b), pref(c)\}\}) \\
= \; & PI((ref(a), pref(a)), PI((ref(b), pref(b)), (ref(c), pref(c))))
\end{aligned}
$$

that is PI is associative.

□

We have shown that the union and intersection on preference classes and fuzzy sets fulfill equivalent axioms. Furthermore fuzzy sets and preference classes represent uncertainty by a fuzzy value or a preference value. Therefore there is a tight relationship between fuzzy sets and preference classes if we interpret them as points in m-dimensional space.

6 Real-world Case Studies for Preference Moore Machines

The main point of this paper has been on the theoretical aspects of preference Moore machines. However, in related work we have described more experimental and applied aspects of preference Moore machinery for syntactic and semantic spoken language analysis [Wermter and Weber, 1997], for Moore machine extraction from recurrent networks

[Wermter, 2000] and classification [Wermter et al., 1999b]. In this section we want to illustrate some additinal properties of preference Moore machines based on a syntactic tagging task and a semantic classification task.

6.1 Syntactic Tagging: From Symbolic Regular Relations to Moore Machines

So far we have described preference Moore machines as a principle for integration. However, we also need to focus on possibilities how higher level symbolic knowledge can be transferred into such sequential machines. Therefore we will briefly outline one opportunity, namely the transfer of regular relations into preference Moore machines.

We will focus on the replacement for 2-relations since these can be interpreted as assigning an output to an input. Such symbolic 2-level representations have been used primarily for phonological and morphological lexicon processing [Koskenniemi, 1983, Kaplan and Kay, 1994]. For instance a morphological 2-level rule system has been built for Finnish which uses lexical strings and surface strings [Koskenniemi, 1983]. The task of the rules is to express the differences between lexical representation and morphological or phonological surface representation of a word.

Different replacement operations can be defined and there has been some progress for the basic formalism of replacement in the calculus of finite machines [Karttunen, 1995, Kempe and Karttunen, 1996], for instance for a conditional parallel replacement. The *conditional parallel replacement* is a relation which maps a set of n expressions U_i in an upper language U in a set of corresponding n expressions L_i in a lower language L iff they occur in a certain left and right context (l_i, r_i).

$$U_1 \rightarrow L_1 || l_1 \_ r_1, \cdots, U_n \rightarrow L_n || l_n \_ r_n$$

For this parallel replacement it is possible that for one input several different outputs can be generated. However, considering the use of recurrent networks we are interested in a single output. Furthermore, we want to consider only the left context corresponding to a left-to-right processing. Therefore we rather need a directed replacement operator which allows the largest possible replacements while moving from left to right. Such symbolic replacement operators for regular relations are currently being investigated [Karttunen, 1996]. While it is still possible to keep the main restrictions of finite symbolic machines, the specification of regular relations is much shorter and more efficient. The longest left-to-right replacement is abbreviated as follows:

$$T@ \rightarrow L...R$$

Moving from left to right, the longest T is identified and new markers L and R are introduced. T, L and R are a regular language (or as a special case a string).

We focus on an example task of assigning phrasal categories to basic syntactic categories. If we use the notation of the longest left-to-right replacement then we can specify the abstract phrasal category of a noun group ng in a simplified version as the following regular relation. That is, a nominal group ng consists of an optional determiner (d) followed by an optional adverb (a), an arbitrarily long sequence of adjectives $j*$ and an arbitrarily long non-empty sequence of nouns $n+$ or a noun group consists of a pronoun u. In a similar manner we show this principle for several other phrasal categories, in particular for verbal groups vg and prepositional groups pg.

$$[[(d)(a)j*n+]|u\ @\rightarrow\ \{ng...\},$$

$$v+\ @\rightarrow\ \{vg...\},$$

$$r(d)(a)j*n+\ @\rightarrow\ \{pg...\}]$$

A verbal group vg can be a non-empty sequence of verbs $v+$, a prepositional group pg can consist of a preposition r, an optional determiner (d) followed by an optional adverb (a), an arbitrarily long sequence of adjectives $j*$ and an arbitrarily long non-empty sequence of nouns $n+$. If such sequences of basic syntactic categories occur then the corresponding parallel replacements can be performed. Of course such a regular relation is not a complete description of all possible syntactic sequences, but many often occurring constructions can already be dealt with.

Sequences of regular expressions and relations can be realized by symbolic Moore machines. That way important subsets of natural language can be analyzed. In this section we want to focus on a specific example for the replacement of 2-relations. Here the left-to-right-replacement operator [Karttunen, 1996] was used to specify a transducer for simple phrases.

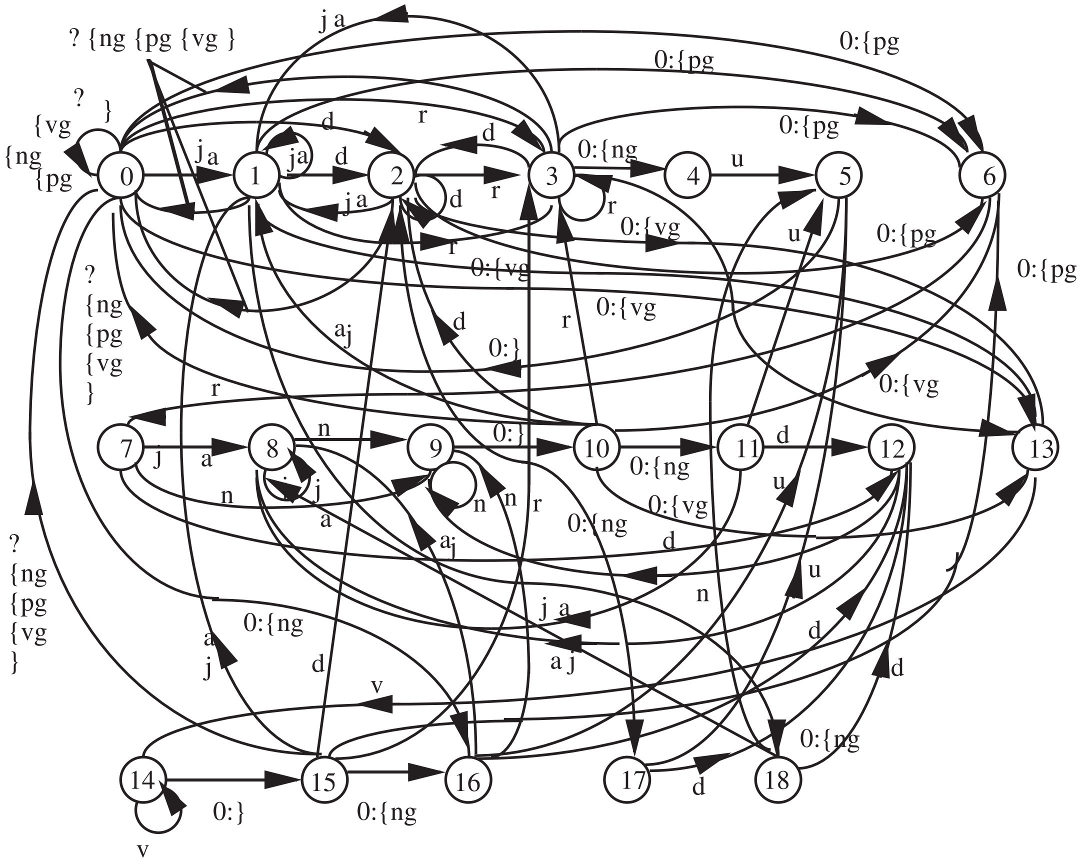

Figure 5: A symbolic Moore Machine for simple noun groups, verbal groups and prepositional groups as generated for the three relations specified above.

The automatically generated 2-band transducer which corresponds to these relations is described in figure 5. The transducer has 19 states and 102 edges. The symbols of the alphabet are: a, d, j, n, r, u, v, $\{ng$, $\{pg$, $\{vg$, $\}$, ?, 0. The symbols a, d, j, n, r, u, v are basic syntactic categories (adverb, determiner, adjective, noun, preposition, pronoun, verb. The symbol ? stands for an arbitrary symbol which is not within the alphabet. The symbol 0 stands for the empty string. States are named with "s" and an additional number, final states are named with “fs”. Each edge has a marker which represents the input and output for the edge transition. For instance, starting from state 0 and for an empty input, the symbol $\{ng$ is generated and the state is transferred to $s16$. This transition is shown as $fs0 :< 0 : \{ng >\to s16$. If input and output of the edge are identical then we only show the symbol once, as for instance in $fs0 : a \to fs1$.

Table 1 shows an example for the replacement in symbolic 2-relations. For each example sentence the input is shown as the basic syntactic category as well as the output as the phrasal syntactic category. The sentence “The fourteenth is a Wednesday” is recognized as a sequence of a noun group, a verbal group and a noun group. In the sentence “I thought in the next week in any case in April” we have a sequence noun group, verbal group, prepositional group, prepositional group, prepositional group from left to right. The restrictions on groups allows that several prepositional groups can be within an overall prepositional phrase. The particular Moore machine which performed this assignment had 19 states and 102 transitions.

<table>
<tr><td>Sentence:
Input:</td><td>The
d</td><td>fourteenth
n</td><td>is
v</td><td>a
d</td><td>Wednesday
n</td></tr>
<tr><td>Replacement:</td><td colspan="2">ng d n</td><td>vg v</td><td colspan="2">ng d n</td></tr>
</table>

<table>
<tr><td>Sentence:
Input:</td><td>I
u</td><td>thought
v</td><td>in
r</td><td>the
d</td><td>next
j</td><td>week
n</td><td>in
r</td><td>any
d</td><td>case
n</td><td>in
r</td><td>April
n</td></tr>
<tr><td>Replacement:</td><td>ng u</td><td>vg v</td><td colspan="4">pg r d j n</td><td colspan="3">pg r d n</td><td colspan="2">pg r n</td></tr>
</table>

Table 1: Examples for replacement in symbolic 2 relations

6.2 Semantic Classification: Learning a Neural Preference Moore Machine

In this section we demonstrate how preference Moore machines can be used in a real world text routing scenario. The Reuters newswire collection [Lewis, 1997] contains real-world documents which appeared on the Reuters newswire. All news titles in the Reuters corpus belong to one or more of eight main categories: Money/Foreign Exchange, Shipping, Interest Rates, Economic Indicators, Currency, Corporate, Commodity, Energy.

We want to use this categorization task to give an example of a neural preference Moore machine and to illustrate preference values and the union of the preferences. In our experiments we use 10 733 titles whose documents have a title and at least one topic. The total number of words is 82 339 and the number of different words in the titles is 11 104. For our training set, we use 1 040 news titles, the first 130 of each of the 8 categories. All the other 9 693 news titles are used for testing the generalization to new and unseen examples.

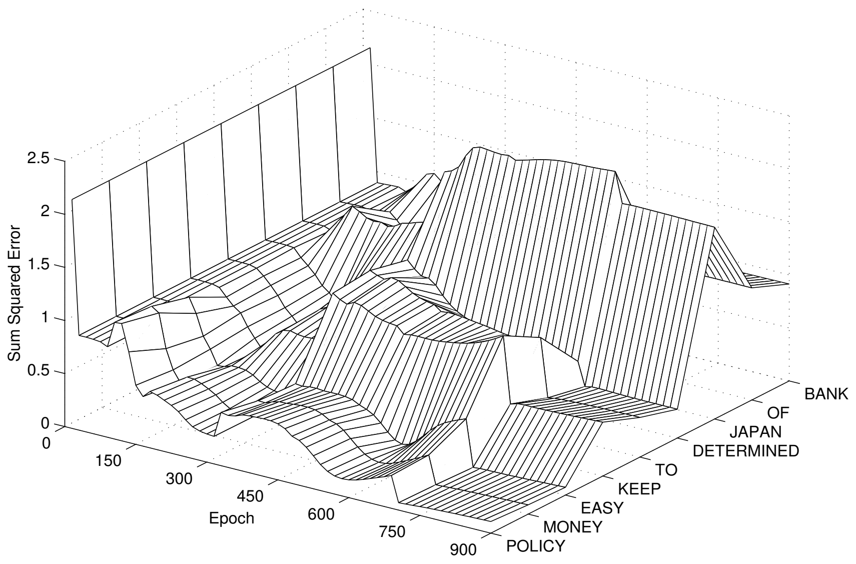

Figure 6: The error surface of the title "Bank of Japan Determined To Keep Easy Money Policy"

In our experiments, we use a recurrent plausibility network [Wermter, 1995] as a Preference Moore machine with two hidden and two context layers (for more details of the particular architecture see [Wermter et al., 1999b]). Input to the network is the word representation, one word at a time. Output is the desired semantic routing category. Training is performed until the sum squared error does not decrease anymore, typically after 900 epochs of training. An example and its training behavior is shown in Figure 6:"Bank of Japan Determined To Keep Easy Money Policy". This example belongs to the "interest" category. The beginning words “Bank of Japan” could be classified under different categories such as “money/foreign exchange” and “currency”. However, the context from words such as “easy money policy” eventually allows the network to learn the correct classification. In contrast to the encoded symbolic Moore machine from the previous subsection, we have here an example of a learning neural preference Moore machine which is trained to perform a classification task.

6.3 Semantic Classification: Combining Several Preference Moore Machines

So far we have illustrated the learning process of a neural preference Moore machine with an example. Each output vector of the network is a preference which belongs to a preference class and has a certain preference value. Using the previous operations intersection or union one can combine two or more sequences of n-dimensional output preferences from multiple networks.

Table 2 shows the results from the preference class union operations applied to the output neural preferences of two neural preference Moore machines in our text routing domain. The two networks have the same architectures and training sessions but started with different initial weights. They have similar overall performance on the Reuters news corpus but the machines differ somewhat which titles they classify correctly.

Category	Test set		
	recall	precision	pref. value
NPM_1	92.57	91.87	0.90
NPM_2	92.67	91.32	0.90
$NPM_1 \cup NPM_2$	95.71	91.96	0.93

Table 2: Neural preference class operations on Neural Preference Moore machines (NPM)

The union of two neural preference Moore machines (NPM) increases the recall and slightly the precision of the final classification. Since the initial recall classification performance is already high, this improvement can be seen as quite significant. A combination of two preference Moore machines seems quite appropriate since when we built the union of three preference

Moore machines the performance is not improved anymore. In general, this experiment demonstrates how for a real world task the introduction of the union of preference classes can improve the performance. For a single preference Moore machine we have 92.67% recall for the best network. With the hybrid approach and combining two neural preference Moore machines we reach a recall of 95.71% with a slightly improved precision rate.

We have just demonstrated particular neural preference Moore machine in a certain domain, preferences and combinations of several such preference Moore machines. Symbolic Moore machines are integrated in exactly the same manner, since based on the preference space they are only a special abstraction of a specific neural preference Moore machine. We illustrate the general case in the next section.

6.4 General Study of Integration of Preference Classes

We will illustrate the general use of the union and intersection for preference classes. We consider the 2-dimensional space. In the following illustration we call the preference values with "S" for small and "L" for large. Then $((00), S)$ is the preference class which contains those preferences which have a preference value S with respect to the reference (00). Table 3 shows the operation of the preference classes with equal reference for the two-dimensional space.

If the preference classes have the same corner reference they are directly comparable with their preference values. The preference class $PU((ref(a), pref(a)), (ref(b), pref(b)))$ is the preference class with the largest preference, that is $PU(((00), S), ((00), L)) = ((00), L)$. On the other hand, $PI((ref(a), pref(a)), (ref(b), pref(b)))$ is the preference class with the smallest preference, that is $PI(((00), S), ((00), L)) = ((00), S)$.

If the preferences classes have a different corner reference the preference classes cannot be judged only by their preference value. In this case the preference classes $PU((ref(a), pref(a)), (ref(b), pref(b)))$ and $PI((ref(a), pref(a)), (ref(b), pref(b)))$ are a generalization of U and I for the two-dimensional space. Therefore, it holds for instance $PU(((00), S), ((01), S)) = ((01), S)$ but $PI(((00), S), ((01), S)) = ((00), S)$. This is based on the following motivation: For instance, if there is a preference for $(no\ noun, no\ verb)$

$class(a)$	$class(b)$	$PI(class(a), class(b))$	$PU(class(a), class(b))$
((0 0),S)	((0 0),S)	((0 0),S)	((0 0),S)
((0 0),S)	((0 0),L)	((0 0),S)	((0 0),L)
((0 0),L)	((0 0),S)	((0 0),S)	((0 0),L)
((0 0),L)	((0 0),L)	((0 0),L)	((0 0),L)
((0 1),S)	((0 1),S)	((0 1),S)	((0 1),S)
((0 1),S)	((0 1),L)	((0 1),S)	((0 1),L)
((0 1),L)	((0 1),S)	((0 1),S)	((0 1),L)
((0 1),L)	((0 1),L)	((0 1),L)	((0 1),L)
((1 0),S)	((1 0),S)	((1 0),S)	((1 0),S)
((1 0),S)	((1 0),L)	((1 0),S)	((1 0),L)
((1 0),L)	((1 0),S)	((1 0),S)	((1 0),L)
((1 0),L)	((1 0),L)	((1 0),L)	((1 0),L)
((1 1),S)	((1 1),S)	((1 1),S)	((1 1),S)
((1 1),S)	((1 1),L)	((1 1),S)	((1 1),L)
((1 1),L)	((1 1),S)	((1 1),S)	((1 1),L)
((1 1),L)	((1 1),L)	((1 1),L)	((1 1),L)

Table 3: Intersection and union of preference classes for equal references. S and L can take arbitrary real values from $[0, 1]$, for which holds: $S \leq L$.

$class(a)$	$class(b)$	$PI(class(a), class(b))$	$PU(class(a), class(b))$
((0 0),S)	((0 1),S)	((0 0),S)	((0 1),S)
((0 0),S)	((0 1),L)	((0 0),S)	((0 1),L)
((0 0),L)	((0 1),S)	((0 0),S)	((0 1),L)
((0 0),L)	((0 1),L)	((0 0),L)	((0 1),L)
((0 0),S)	((1 0),S)	((0 0),S)	((1 0),S)
((0 0),S)	((1 0),L)	((0 0),S)	((1 0),L)
((0 0),L)	((1 0),S)	((0 0),S)	((1 0),L)
((0 0),L)	((1 0),L)	((0 0),L)	((1 0),L)
((0 0),S)	((1 1),S)	((0 0),S)	((1 1),S)
((0 0),S)	((1 1),L)	((0 0),S)	((1 1),L)
((0 0),L)	((1 1),S)	((0 0),S)	((1 1),L)
((0 0),L)	((1 1),L)	((0 0),L)	((1 1),L)
((0 1),S)	((1 0),S)	((0 0),S)	((1 1),S)
((0 1),S)	((1 0),L)	((0 0),S)	((1 1),L)
((0 1),L)	((1 0),S)	((0 0),S)	((1 1),L)
((0 1),L)	((1 0),L)	((0 0),L)	((1 1),L)
((0 1),S)	((1 1),S)	((0 1),S)	((1 1),S)
((0 1),S)	((1 1),L)	((0 1),S)	((1 1),L)
((0 1),L)	((1 1),S)	((0 1),S)	((1 1),L)
((0 1),L)	((1 1),L)	((0 1),L)	((1 1),L)
((1 0),S)	((1 1),S)	((1 0),S)	((1 1),S)
((1 0),S)	((1 1),L)	((1 0),S)	((1 1),L)
((1 0),L)	((1 1),S)	((1 0),S)	((1 1),L)
((1 0),L)	((1 1),L)	((1 0),L)	((1 1),L)

Table 4: Intersection and union of preference classes for different references. S and L can take arbitrary values from $[0, 1]$ for which holds: $S \leq L$.

and at the same time a preference for $(no\ noun, verb)$, then PU provides the optimistic integration, namely $(no\ noun, verb)$ and PI provides the pessimistic integration, namely $(no\ noun, no\ verb)$. The preference value of the intersection of preference classes is the minimum of the preference values of the arguments, and the preference value of the union of preference classes is the maximum of the preference values of the arguments.

7 Discussion

Recently the question whether recurrent networks can emulate each symbolic Moore machine and each finite automaton has been examined [Kremer, 1995, Kremer, 1996]. On the other hand it has been shown [Goudreau and Giles, 1995, Goudreau et al., 1994] that a recurrent network with only one input layer, one context layer and one output layer (so-called Single-Layer-First-Order-Network) is not sufficient for realizing arbitrary finite automata. Therefore, the link with our preferences between recurrent neural networks and symbolic transducers is particularly important.

Preference Moore machines are still a relatively simple form of computational machine in terms of the Chomsky hierarchy. In the future, other forms of symbolic/neural integration may develop for other types of machines. Here we focused on Moore machines because they are relatively simple and efficient. In the future more complex machines, like different pushdown automata with explicit unlimited memory, may be further candidates for additional principles of neural symbolic integration. So far, it could only be shown that simple recurrent networks can emulate certain restricted properties of a pushdown automaton, in particular the recursive representation of structures up to a limited depth [Elman, 1991, Wiles and Elman, 1996]. Furthermore, more different preference machines may be developed which are more realistic with respect to real biological neural networks.

In contrast to traditional symbolic regular representations, neural preference Moore machines can represent gradual and learned representations. Furthermore, the number of input, state and output preferences is not necessarily finite. Therefore neural preference Moore machines are more powerful than finite transducers. Our recurrent neural networks can be seen as learning $n \times m$ Fuzzy-transducers which augment a simple finite symbolic transducer with respect to learning within a gradual preference space. From this perspective symbolic knowledge is a special abstraction from a neural preference space.

Neural networks as well as fuzzy rules on fuzzy sets are model-free dynamic systems [Lin and Lee, 1994]. That is, both neural and fuzzy representations should be used especially for those tasks where appropriate mathematical system descriptions are not available or where the system descriptions are non-linear [Wang and Mendel, 1992]. Furthermore, it has been proved that simple feedforward networks as well as fuzzy rules can realize universal function approximators [Wang and Mendel, 1992, Hornik et al., 1989, Omlin et al., 1995].

However, neural representations and fuzzy representations also have several differences. Fuzzy representations like fuzzy sets and fuzzy rules have some advantages for inferences and direct interpretation. Furthermore, fuzzy rules can be determined based on examples or based on manual encoding, which is particularly useful for small data sets [Wang and Mendel, 1992]. Neural representations have some advantages for automatic learning and context integration. Usually learning is not a major strength of fuzzy representations [Takagi, 1994]. However, neural networks support learning extensively. The determination of the membership function, which is often done ad hoc in fuzzy representations, is a main problem for the development of fuzzy systems [Kruse et al., 1993, Jang and Sun, 1995][2]. However learning a membership function can be supported by learning neural networks [Jang and Sun, 1995]. By providing a link between fuzzy and neural preferences and machines it is possible that both representations can be used in a complementary manner.

Although simple computational means like symbolic Moore machines or regular languages are not sufficient to describe all possible constructions of natural language completely (see e.g. [Winograd, 1983]), they still represent a central minimal requirement for the representation of natural language. Therefore, they are situated at the lower level in the Chomsky hierarchy of languages [Hopcroft and Ullman, 1979]. However, it is possible to design efficient realiza-

[2] "The specification of membership functions is quite subjective, which means the membership functions specified for the same concept (say, "cold") by different persons may vary considerably." [Jang and Sun, 1995] S. 2. "The characteristic function which defines a fuzzy set *characterizes* the relationship between real world entities and specific concepts" ... "however, it does not model or explain how this fuzzy relationship comes about." [Freksa, 1994]

tions of symbolic finite automata for different areas [Kaplan, 1995, Karttunen, 1996], e.g. for morphology or lexicon access.

Recently we have worked extensively on many forms of experimental neural/symbolic integration based on speech/language analysis. In a larger speech/language system SCREEN we have also used more than 15 different modules based on simple recurrent networks and symbolic automata [Wermter and Löchel, 1996, Wermter and Weber, 1997, Wermter and Meurer, 1997]. This system demonstrated that it was possible to analyze the acoustics, syntax, semantics and dialog level of noisy spoken input using simple recurrent networks and symbolic automata. However, for making progress beyond a single research problem more general principles of interaction have to be developed. We suggest that our new concepts for preference Moore machine integration based on preference classes contribute a new approach towards general principles of neural symbolic integration.

Preference Moore machines and their integration can be used for a large number of potential problems, especially where sequential noisy preference mappings may be necessary. We have started to explore various forms of neural preference Moore machines for text routing [Wermter et al., 1999b, Wermter et al., 1999a]. Possible new tasks could include preference Moore machines for flexible control of robot actions, handwriting recognition, information extraction, or spoken language analysis. All these example areas benefit from previous state context for sequential processing, robust preference representations for potentially noisy input and learning representations to acquire unknown regularities. Neural preference Moore machines, like plausibility networks, can support such properties but can also benefit from the integration with known symbolic heuristics in these fields. Furthermore, in future work it would also be interesting to explore the relationship between preference Moore machines and real neural networks in order to integrate more realistic neuroscience computing principles into neural network architectures.

Acknowledgments

I would like to thank Malcolm Farrow for his comments on the paper, I would like to thank Christo Panchev for computing the union of preferences on the Reuters data.

References

[Booth, 1967] Booth, T. L. (1967). *Sequential Machines and Automata Theory.* John Wiley, New York.

[Cheng et al., 1994] Cheng, Y., Fortier, P., and Normandin, Y. (1994). A system integrating connectionist and symbolic approaches for spoken language understanding. In *Proceedings of the International Conference on Spoken Language Processing*, pages 1511–1514, Yokohama.

[Churchland and Sejnowski, 1992] Churchland, P. S. and Sejnowski, T. J. (1992). *The Computational Brain.* MIT Press, Cambridge, MA.

[Dorffner, 1997] Dorffner, G. (1997). *Neural Networks and a New AI.* Chapman and Hall, London, UK.

[Dyer, 1991] Dyer, M. G. (1991). Symbolic neuroengineering for natural language processing: a multilevel research approach. In Barnden, J. A. and Pollack, J. B., editors, *Advances in Connectionist and Neural Computation Theory, Vol.1: High Level Connectionist Models*, pages 32–86. Ablex Publishing Corporation, Norwood, NJ.

[Elman, 1991] Elman, J. L. (1991). Distributed representations, simple recurrent networks, and grammatical structure. *Machine Learning*, 7:195–226.

[Freksa, 1994] Freksa, C. (1994). Fuzzy systems in AI. In Kruse, R., Gebhardt, J., and Palm, R., editors, *Fuzzy Systems in Computer Science.* Viehweg, Braunschweig.

[Goudreau and Giles, 1995] Goudreau, M. W. and Giles, C. L. (1995). On recurrent neural networks and representing finite-state recognizers. In *Proceedings of the Third International Conference on Neural Networks*, pages 51–55.

[Goudreau et al., 1994] Goudreau, M. W., Giles, C. L., Chakradhar, S. T., and Chen, D. (1994). First-order vs. second-order single layer recurrent neural networks. *IEEE Transactions on Neural Networks*, 5(3):511–513.

[Hendler, 1991] Hendler, J. (1991). Developing hybrid symbolic/connectionist models. In Barnden, J. A. and Pollack, J. B., editors, *Advances in Connectionist and Neural Computation Theory, Vol.1: High Level Connectionist Models*, pages 165–179. Ablex Publishing Corporation, Norwood, NJ.

[Honavar and Uhr, 1994] Honavar, V. and Uhr, L. (1994). *Artificial Intelligence and Neural Networks: Steps Toward Principled Integration.* Academic Press, Cambridge, MA.

[Hopcroft and Ullman, 1979] Hopcroft, J. and Ullman, J. (1979). *Introduction to Automata Theory, Languages, and Computation.* Addison Wesley, Reading, MA.

[Hornik et al., 1989] Hornik, K., Stinchcombe, W., and White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, 2:359–366.

[Jang and Sun, 1995] Jang, J. R. and Sun, C. (1995). Neuro-fuzzy modeling and control. In *Proceedings of the IEEE.*

[Jurafsky et al., 1994] Jurafsky, D., Wooters, C., Tajchman, G., Segal, J., Stolcke, A., Fosler, E., and Morgan, N. (1994). The Berkeley Restaurant Project. In *Proceedings of the International Conference on Speech and Language Processing*, pages 2139–2142, Yokohama.

[Kaplan, 1995] Kaplan, R. (1995). Finite state technology. In Cole, R. A., Mariani, J., Uszkoreit, H., Zaenen, A., Zue, V., Varile, G., and Zampolli, A., editors, *Survey of the State of the Art in Human Language Technology*, pages 419–422. NSF, EU.

[Kaplan and Kay, 1994] Kaplan, R. M. and Kay, M. (1994). Regular models of phonological rule systems. *Computational Linguistics*, 20(3):331–378.

[Karttunen, 1995] Karttunen, L. (1995). The replace-operator. In *Proceedings of the Meeting of the Association for Computational Linguistics*, Cambridge.

[Karttunen, 1996] Karttunen, L. (1996). Directed replacement. In *Proceedings of the Meeting of the Association for Computational Linguistics*, Santa Cruz.

[Kempe and Karttunen, 1996] Kempe, A. and Karttunen, L. (1996). Parallel replacement in finite state calculus. In *Proceedings of the International Conference on Computational Linguistics*, pages 622–627, Copenhagen.

[Klir and Folger, 1988] Klir, G. J. and Folger, T. A. (1988). *Fuzzy Sets, Uncertainty and Information*. Prentice Hall.

[Koskenniemi, 1983] Koskenniemi, K. (1983). Two level morphology: a general computational model for word-form recognition and production. Technical Report PhD thesis, Dept. of General Linguistics, University of Helsinki.

[Kosko, 1992] Kosko, B. (1992). *Neural Networks and Fuzzy Systems*. Prentice-Hall, Englewood Cliffs, NJ.

[Kremer, 1995] Kremer, S. C. (1995). On the computational power of Elman-style recurrent networks. *IEEE Transactions on Neural Networks*, 6(4):1000–1004.

[Kremer, 1996] Kremer, S. C. (1996). A theory of grammatical induction in the connectionist paradigm. Technical Report PhD dissertation, Dept. of Computing Science, University of Alberta, Edmonton.

[Kruse et al., 1993] Kruse, R., Gebhardt, J., and Klawonn, F. (1993). *Fuzzy Systeme*. Teubner, Stuttgart.

[Kwasny and Faisal, 1992] Kwasny, S. C. and Faisal, K. A. (1992). Connectionism and determinism in a syntactic parser. In Sharkey, N., editor, *Connectionist natural language processing*, pages 119–162. Lawrence Erlbaum.

[Lewis, 1997] Lewis, D. D. (1997). Reuters-21578 text categorization test collection. http://www.research.att.com/~lewis.

[Lin and Lee, 1994] Lin, C. T. and Lee, C. S. G. (1994). Supervised and unsupervised learning with fuzzy similarity for neutral network-based fuzzy logic control systems. In Yager, R. R. and Zadeh, L. A., editors, *Fuzzy sets, neural networks and soft computing*, pages 85–125. Van Nostrand, New York.

[Medsker, 1995] Medsker, L. R. (1995). *Hybrid Intelligent Systems.* Kluwer Academic Publishers, Boston.

[Miikkulainen, 1993] Miikkulainen, R. (1993). *Subsymbolic Natural Language Processing.* MIT Press, Cambridge, MA.

[Omlin et al., 1995] Omlin, C. W., Thornber, K. K., and Giles, C. L. (1995). Fuzzy finite-state automata can be deterministically encoded into recurrent neural networks. Technical Report CS-TR-3599, University of Maryland, College Park.

[Reilly and Sharkey, 1992] Reilly, R. G. and Sharkey, N. E. (1992). *Connectionist Approaches to Natural Language Processing.* Lawrence Erlbaum Associates, Hillsdale, NJ.

[Santos, 1973] Santos, E. S. (1973). Fuzzy sequential functions. *Journal of Cybernetics*, 3(3):15–31.

[Sun, 1995] Sun, R. (1995). Schemas, logics and neural assemblies. *Applied Intelligence*, 5:83–102.

[Sun and Bookman, 1995] Sun, R. and Bookman, L. (1995). *Computational Architectures Integrating Neural and Symbolic Processes.* Kluwer Academic Publishers, Boston, MA.

[Takagi, 1994] Takagi, T. (1994). Context sensitive knowledge processing based on conceptual fuzzy sets. In Yager, R. R. and Zadeh, L. A., editors, *Fuzzy sets, neural networks and soft computing*, pages 331–344. Van Nostrand, New York.

[Wang and Mendel, 1992] Wang, L.-X. and Mendel, J. M. (1992). Generating fuzzy rules by learning from examples. *IEEE Transactions on Systems, Man, and Cybernetics*, 22(6):1414–1427.

[Wermter, 1995] Wermter, S. (1995). *Hybrid Connectionist Natural Language Processing.* Chapman and Hall, Thomson International, London, UK.

[Wermter, 1997] Wermter, S. (1997). Hybrid approaches to neural network-based language processing. Technical Report TR-97-030, International Computer Science Institute, Berkeley, CA.

[Wermter, 1999] Wermter, S. (1999). Preference Moore machines for neural fuzzy integration. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 840–845, Stockholm.

[Wermter, 2000] Wermter, S. (2000). Knowledge extraction from transducer neural networks. *Journal of Applied Intelligence*, 12:27–42.

[Wermter et al., 1999a] Wermter, S., Arevian, G., and Panchev, C. (1999a). Recurrent neural network learning for text routing. In *Proceedings of the International Conference on Artificial Neural Networks*, pages 898–903, Edinburgh, UK.

[Wermter and Löchel, 1996] Wermter, S. and Löchel, M. (1996). Learning dialog act processing. In *Proceedings of the International Conference on Computational Linguistics*, pages 740–745, Copenhagen, Denmark.

[Wermter and Meurer, 1997] Wermter, S. and Meurer, M. (1997). Building lexical representations dynamically using artificial neural networks. In *Proceedings of the International Conference of the Cognitive Science Society*, pages 802–807, Stanford.

[Wermter et al., 1999b] Wermter, S., Panchev, C., and Arevian, G. (1999b). Hybrid neural plausibility networks for news agents. In *Proceedings of the National Conference on Artificial Intelligence*, pages 93–98, Orlando, USA.

[Wermter et al., 1996] Wermter, S., Riloff, E., and Scheler, G. (1996). *Connectionist, Statistical and Symbolic Approaches to Learning for Natural Language Processing.* Springer, Berlin.

[Wermter and Sun, 2000] Wermter, S. and Sun, R. (2000). *Hybrid Neural Symbolic Systems.* Springer, Heidelberg.

[Wermter and Weber, 1997] Wermter, S. and Weber, V. (1997). SCREEN: Learning a flat syntactic and semantic spoken language analysis using artificial neural networks. *Journal of Artificial Intelligence Research*, 6(1):35–85.

[Wiles and Elman, 1996] Wiles, J. and Elman, J. (1996). Learning to count without a counter: A case study of dynamics and activation landscapes in recurrent networks. In *Proceedings of the AAAI Workshop on Computational Cognitive Modeling: Source of the Power*, Portland, Oregon.

[Winograd, 1983] Winograd, T. (1983). *Language as a Cognitive Process.* Addison-Wesley, Reading, MA.

[Yager, 1994] Yager, R. R. (1994). Modeling and formulating fuzzy knowledge bases using neural networks. *Neural Networks*, 7(8):1273–1283.

[Zadeh, 1965] Zadeh, L. (1965). Fuzzy sets. *Information and Control*, 8:338–353.

[Zimmermann, 1991] Zimmermann, H. (1991). *Fuzzy Set Theory and its Applications.* Kluwer, Boston.