

Dynamic Conflict Resolution in a Connectionist Rule-Based System*

Clayton McMillan
Dept. of Computer Science
University of Hamburg
D-2000 Hamburg 50
Germany

Michael C. Mozer and Paul Smolensky
Dept. of Computer Science and
Institute of Cognitive Science
University of Colorado
Boulder, CO 80309 USA

Abstract

In performing higher level cognitive tasks in a connectionist network, we have argued that it is useful to incorporate symbol processing formalisms and mechanisms directly into the network architecture. We have previously described a neural network, RuleNet [McMillan *et al.*, 1992] that maps strings of input symbols to strings of output symbols. RuleNet discovers a functional categorization of the input symbols, as well as a compact set of symbolic rules that operate on these categories to generate the output strings. The integration of subsymbolic categorization and symbolic rule following in a single system yields a variety of important consequences. This paper explores one consequence in detail, namely a novel way in which RuleNet performs conflict resolution among rules: When a conflict arises, RuleNet exploits the varying category membership strengths of input symbols to select a rule. This dynamic, content-based conflict resolution technique contrasts with traditional, broad heuristic techniques such as selection of the most specific or least recently used rule. RuleNet also has the ability to learn a priority ranking of rules, and this ranking can interact with the category-strength selection technique in a uniform quantitative framework. This uniform framework—the combination of evidence in a connectionist system—is a powerful and elegant approach to conflict resolution.

1 Introduction

Standard connectionist techniques have been applied successfully in performing a number of low level cognitive tasks. There has been an increasing awareness in recent literature, however, that to perform higher level tasks much

more structure is required than in a simple three-layer back-propagation net. One general approach is to incorporate a tested technique from symbolic AI into a neural network, extending it so as to get the best from both the worlds of symbolic and subsymbolic processing. A promising area of research in this direction is the development of connectionist networks as rule-based systems. Several such systems have been explored, e.g., Sun [1991], Towell and Shavlik [1992], and a network called RuleNet, which imposes constraints on weight changes and exploits a variation on back-propagation learning to induce symbolic rules in a string mapping domain [McMillan, *et al.*, 1992].

The RuleNet architecture is of interest because it can be divided into two components: A network that induces symbolic rules and a network that performs subsymbolic categorization. The categorization is a function of multiple input features and their potentially higher order relationships, rather than a predicate-based grouping of arbitrary symbols. The rules induced are written in and are directly a function of the categories. At the same time, the categories are a function of the way symbols are used in the rules. In this sense, symbolic and subsymbolic processing are directly integrated, rather than indirectly through modules, as is typical in hybrid systems. This paper explores in detail some previously unreported results in RuleNet, in particular how the network implicitly performs conflict resolution in a novel way, exploiting the strengths of membership of input symbols in the categories manipulated by rules. This mechanism is of interest for two reasons. First, it is not based upon the general but arbitrary schemes typically employed in rule-based systems. Some examples of more traditional schemes are: Firing the rule: 1) with the most specific conditions, or 2) with the longest list of conditions, or 3) that matches the highest priority data or 4) that was least recently used. Instead RuleNet's conflict resolution is based upon input context and is adaptive to different problem domains. Second, it is performed at a subsymbolic level capturing subtle variations in inputs, and is thus potentially more computationally powerful than traditional conflict resolution schemes. At the same time, a mechanism for implementing the traditional priority ranking of rules is available to the network, as well as the possibility of combining the category level and rule level strategies, as required by the demands of the training data.

Section 2 reviews the domain in which RuleNet operates and the RuleNet architecture. Section 3 describes

* This research was supported by a Zuse-Gastprofessor Award from the German Academic Exchange Service to CM, by NSF Presidential Young Investigator award IRI-9058450, grant 90-21 from the James S. McDonnell Foundation, and DEC external research grant 1250 to MC, and by NSF grants IRI-8609599 and ECE-8617947, and a grant from the Sloan Foundations's computational neuroscience program to PS.

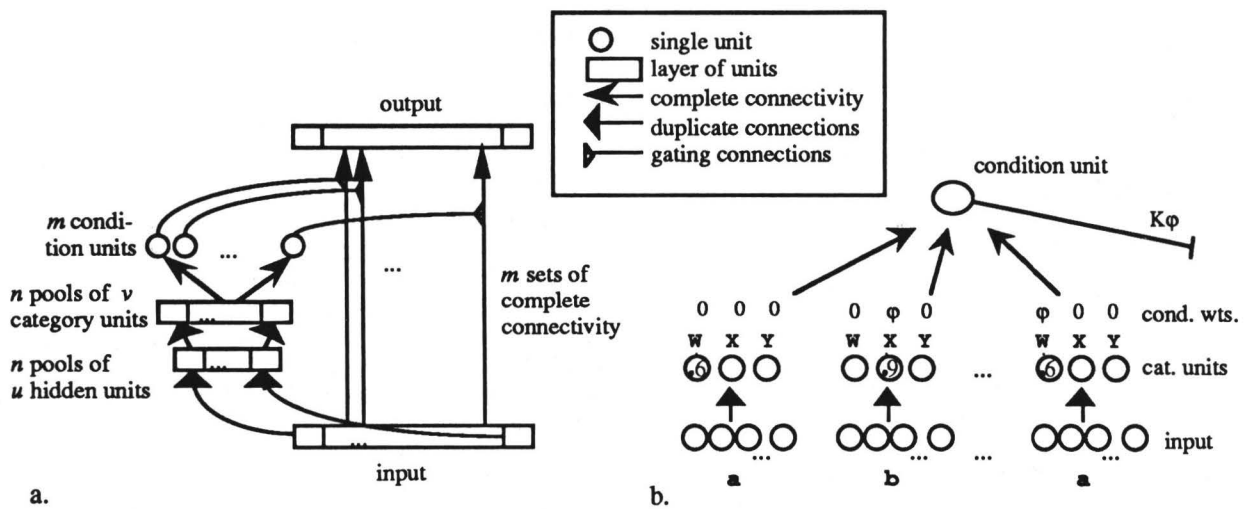


Figure 1 a. The RuleNet architecture. b. Example of how the net input to a single condition unit is calculated between the input and condition layers, given the input string **ab...a**: $net = (X \text{ in slot } 2)\phi + (W \text{ in slot } n)\phi + K\phi = .9\phi + .6\phi + K\phi$.

the task of case-role assignment and in particular how RuleNet employs implicit conflict resolution to efficiently learn the input/output mapping and generalize to novel patterns. Section 4 concludes with remarks on the potential of this approach.

2 Domain and RuleNet Architecture

Domain. RuleNet operates in domains that map input strings to output strings. A string consists of n slots, each containing a symbol. For example, the string **abcd** contains the symbol **c** in slot 3. The mapping may be completely characterized by rules of the general form "if certain symbols are present in the input then perform a certain mapping from the input slots to the output slots." The conditions do not operate directly on the input symbols, but rather on categories defined over the input symbols. For example, the words **boy** and **girl** are instances of the higher level category **HUMAN**. In this paper, instances are denoted with lowercase bold font and categories with uppercase font. It should be apparent from context whether a letter string refers to a single instance, such as **boy**, or a string of instances, such as **abcd**.

Three types of conditions are allowed: 1) a *simple* condition, which states that an instance of some category must be present in a particular slot of the input string, 2) a *conjunction* of two simple conditions, and 3) a *disjunction* of two simple conditions. A typical condition might be that an instance of the category **W** must be present in slot 1 of the input string and an instance of category **Y** must be present in slot 3. These binary conditions can easily be extended to conditions of arbitrary arity, including negation [McMillan *et al.*, in preparation]. The action performed by a rule produces an output string in which the content of each slot is either a fixed symbol in the output alphabet or a copy of a particular input slot, with the additional constraint that each input slot maps to at most one output slot. A typical action might be to switch the symbols in slots 1 and 2 of the input, replace slot 3 with the symbol **a**, and copy slot 4 of the input to the output string unchanged, e.g., **abcd** → **baad**. McMillan *et al.* call rules of this general form *second-order categorical permutation rules* (SCP rules).

An example of an SCP rule for strings of length four is:

if $input_1 = \text{instance of } W$ and $input_3 = \text{instance of } Y$ then
 $output_1 = input_4, output_2 = h, output_3 = input_2, output_4 = input_1$
 where $input_\alpha$ and $output_\beta$ denote input slot α and output slot β , respectively.

RuleNet's task is to induce a compact set of rules that accurately characterizes a set of training examples in this domain. To test the network's performance, McMillan *et al.* generate training examples using a predefined rule base. Input strings that meet the conditions of more than one rule in the set are excluded. For example, the rule above may be used to generate the exemplars:

hedk → **khch**, **clck** → **khlc**, **gbdj** → **jhbg**, **gdbk** → **khdy**

where category **W** consists of **a, b, c, d, i**, and category **Y** consists of **f, g, h**. RuleNet's task then, is twofold: It must discover the categories solely based upon the usage of their instances, and it must induce rules based upon those categories.

Architecture. RuleNet can implement SCP rules of the type detailed above. As shown in Figure 1, RuleNet has five layers of units. The input layer is divided into n pools of units, one for each slot, and activates the category layer, which is also divided into n pools. Input pool α maps to category pool α . Units in category pool α represent possible categorizations of the symbol in input slot α . One or more category units will respond to each input symbol. Category units have activations in the range $[0,1]$. The activation rule for category units is:

$$z_i = \sum_{j=1}^l w_{ij} x_j + b_i$$

where z_i is the activation of category unit i , l is the number of input units, x_j is input unit j , w_{ij} is the category weight from x_j to category unit i , and b_i is the bias. It is also possible to use a non-linear hidden layer between the input and category layers, in which case the instance representation can be distributed.

There are m units in the condition layer, one per possible rule represented in the network. The net input to condition unit i is:

$$net_i = \sum_{j=1}^p w_{ij} z_j + b_i$$

where p is the number of category units, w_{ij} is the weight from category unit j to condition unit i , and b_i is the bias. The activation rule for condition unit i is:

$$c_i = f(net_i) / \sum_{j=1}^m f(net_j)$$

where f is the sigmoid activation function. Thus, the activations across condition units are normalized and can be interpreted as the probability that a given rule applies to the input string. Under this interpretation the condition layer functions in a winner-take-all fashion where the rule of the most active condition unit fires and the rest do not. Firing means that the corresponding action will transform the input to the output. There is one set of weights that completely connects the input layer to the output layer for each of the m rules, representing the action part of rule i .

The strategy for interpreting the weights as rules is two-fold. First, weight changes are constrained in the network to be close to weights that can be interpreted as SCP rules, and second, these learned weights are periodically projected to the nearest set of weights that corresponds exactly to SCP rules. This presupposes that such a set of weights exists. In fact, for every SCP rule a corresponding set of parameterized weights—called a *weight template*—can be specified that exactly implements that rule in RuleNet.

For the purposes of conflict resolution we need only consider category and condition templates:

Category templates. The category templates specify that the mapping from each input slot α to category pool α , for $1 \leq \alpha \leq n$ is the same for each α . This imposes category invariance across the input string. As shown in the example in Figure 1.b, instance **a** activates category unit **W** to the same degree in slots 1 and n . In the present work category units are linear, so to achieve the constraint that categories are limited to $[0,1]$, category weights are limited to the range $[0,1]$.

Condition templates. There are three types of condition templates. In the template for a simple condition exactly one weight, weight j , and the bias are non-zero. The equation for the net input to condition unit i above can therefore be expressed as,

$$net_i = z_j \phi_i + K \phi_i$$

where ϕ_i is an arbitrary parameter greater than zero, with one ϕ_i per condition unit, and K is a constant. In the template for a disjunctive condition all weights are zero except for one weight in each of two slots, weights j and k , and the bias. The net input is therefore:

$$net_i = z_j \phi_i + z_k \phi_i + K \phi_i$$

Since condition units are normalized, a value of $K = -1$ is appropriate for both simple and disjunctive conditions. Finally, the template for a conjunctive condition has a similar form to the disjunctive template, except that the bias constant, L , is -2 . The net input can thus be expressed as:

$$net_i = z_j \phi_i + z_k \phi_i + L \phi_i$$

An example of a condition vector which detects the disjunction that $input_2$ is an instance of **X** or $input_n$ is an instance of **W** is shown in Figure 1.b.

Projection. Constraints on the weights changes force RuleNet to learn a set of weights that is close to the weight

templates, but there is no guarantee that they can directly be interpreted as SCP rules. Instead, a least squares projection procedure is used to map the learned condition weights to the closest condition weight template and the learned action weights to the closest action weight template, while simultaneously assigning values to the ϕ_i parameters in these templates. For a detailed description of projection see McMillan [1992].

Learning. RuleNet is allowed to learn a training set for some number of epochs, using the back propagation learning algorithm, and then projection is performed on the learned weights. The result is a new set of weights which corresponds exactly to a set of SCP rules. These new weights are then injected back into the network and training continues until all patterns in the training set have been mastered. This procedure is called *iterative projection*.

Results. McMillan *et al.* [1992] show that RuleNet can induce a compact set of rules that completely describes the training data. A number of training sets in the abstract string mapping domain described at the beginning of this section were generated from rules over a four category alphabet with between three and five instances per category. In virtually every simulation RuleNet induced not only a set of rules that covered the training data, but in fact induced the original set of rules used to generate the data. The training sets in these simulations represented 3-5% of the total patterns that could be generated with the rule sets. Because RuleNet induced the original rules, it was able to generalize 300-2000% better than a three layer feed forward backprop net and the *local expert* network described in Jacobs *et al.* [1991], upon which RuleNet is based. These simulations demonstrate clearly that much is to be gained by incorporating available knowledge and structure of higher level domains into the structure (in the form of soft and hard constraints) and learning mechanism (in the form of iterative projection) of a neural network such as RuleNet.

3 Conflict Resolution

As indicated in Section 2, input strings that fire more than one rule are explicitly excluded from the training and test sets, because no conflict resolution mechanism is built into the RuleNet architecture. This restriction is fine for domains where at least one set of mutually exclusive rules that covers the training exemplars is guaranteed to exist. In this case it is possible to simply exclude input exemplars that cause conflicts, avoiding the problem altogether. This is exactly what was done in the above simulations. However, what if one wishes to apply RuleNet to a problem in a domain where the rules aren't known a priori, where patterns that fire multiple rules are likely to exist? It turns out that the conflict resolution problem is solved for free in the connectionist hardware of RuleNet. We discuss this further in the context of the case-role assignment task.

Case-role assignment is the task of assigning syntactic constituents to their underlying semantic or thematic roles. For example, in the sentence, "The boy broke the window with the rock", boy is the subject at the syntactic level and the *agent* (actor) at the semantic level, window is the object and *patient* (thing being acted upon) at the syntactic and semantic levels, respectively, and rock is the noun in the "with" prepositional phrase (hereafter w-NP) at the syntactic level and the *instrument* (used by the agent) at the semantic

Category	Instance and Strength of Membership
ED/AN (edible/animate)	bat (.999), boy, cheese, chicken, girl (.882), man, woman, potato (.909), pasta (.890), carrot (.718), dog, wolf, sheep, ate.
BRKR/ATE (breaker/"ate")	ball (.999), bat (.659), paperwt, hatchet, hammer, rock, spoon, ate (.993)
SEMOBJ	ball, paperwt (.748), curtain, desk, doll, fork (.850), hatchet (.688), hammer, plate, rock, potato (.983), spoon, vase, window, ate
GARN/POSS (garnish/ possession)	ball (.987), bat (.975), cheese, chicken, curtain (.781), doll, woman (.565), potato (.802), pasta, carrot, window (.998), dog (.877), wolf (.637), sheep, lion (.775), hit (.925), broke (.530), "___" (.555)
UTENSIL	ball (.614), paperwt, fork, hatchet, hammer, plate, rock, spoon, vase, moved
NULL	ate, "___" (.841)
MOVED	moved (.752), boy (.712), cheese (.739), curtain (.831), doll (.826), hammer (.856), pasta (.972), carrot (.992), window (.987), dog, wolf, sheep, lion

Table 1: The seven functional categories induced by RuleNet in one simulation run. Instances without a membership strength in parentheses have the maximum strength of 1.0. Instances with strengths less than 0.5 are not shown.

level. The problem is, given the syntactic representation as the input, how can a system map the syntactic constituents to a semantic representation as the output?

McClelland and Kawamoto [1986] view this as a mapping problem. In their implementation a scaled down version of the problem is viewed as the mapping of strings of length four to strings of length four, where each slot in the input and output stands for a syntactic or semantic role, respectively. In the input, words 1-4 represent the subject, verb, object, and w-NP, respectively. In the output, words 1-4 represent the agent, patient, instrument, and a possible second interpretation of the w-NP, respectively. The second interpretation of the w-NP is as the *modifier* of the patient, such as in the sentence, "The boy broke the vase with the rose." Some input/output exemplars in terms of the RuleNet domain are:

subj	verb	object	w-NP		agent	patient	instr.	modifier
boy	broke	vase	rock	→	boy	vase	rock	___
boy	broke	vase	rose	→	boy	vase	___	rose
vase	broke	___	___	→	___	vase	___	___

where "___" indicates a null slot value.

McClelland and Kawamoto devise a distributed representation of words and concatenate them together into input and output *strings* of words. They then apply a special purpose architecture to 152 such exemplars, which represent sentences created using 19 generators over 12 categories of words. The 12 categories cover 34 nouns and verbs. Their network is able to learn the mapping and show a number of

interesting properties, both in the way it handles ambiguities in the training set, as well as in the way it generalizes to novel sentences.

RuleNet was trained on exactly the same 152 sentences (using a local representation of instances) for 5000 epochs, projecting every 500 epochs. The network was allowed to make use of up to 10 rules and eight categories. Five sets of simulations were run. On the average, RuleNet learned 6.25 rules covering 92% of the training set and 84% of the 1355 novel sentences that can be generated from McClelland and Kawamoto's sentence generators. Note that they tested only 38 novel sentences, 2 for each generator, so a direct comparison of generalization is difficult. Nevertheless, RuleNet's performance suggests that even in non-clean rule-governed domains, the explicit exploitation of rules in a neural network is useful.

The purpose of this section is to show how RuleNet uses conflict resolution to achieve this high level of performance. To see this, it is instructive to examine the results of one simulation. Table 1 shows seven categories of words discovered by RuleNet and Table 2 shows the six rules extracted from the weights in the network. These rules are not simply the authors' interpretation of what was learned, but were extracted directly from the weights in the network.

The category labels in Table 1 are the authors' interpretation of the underlying commonalities of the words in the corresponding categories, based upon the way they are used in the rules. RuleNet, of course has no notion of

Rule	Description and Sample Sentence
1	if obj = ED/AN and w-NP = GARN/POSS then agent = subj, patient = obj, modifier = w-NP <i>The boy ate the pasta with the cheese.</i>
2	if obj = SEMOBJ and w-NP = UTENSIL then agent = subj, patient = obj, instrument = w-NP <i>The boy ate the pasta with the fork.</i>
3	if subj = BRKR/ATE and verb = GARN/POSS then instrument = subj, patient = obj <i>The paperweight broke the window.</i>
4	if subj = SEMOBJ and obj = NULL then patient = subj <i>The window broke. The ball moved.</i>
5	if verb = BRKR/ATE and obj = NULL then agent = subj, patient = food <i>The lion ate.</i>
6	if verb = MOVED and obj = NULL then agent = subj, patient = self <i>The man moved.</i>

Table 2: The six rules extracted from RuleNet in the simulation run of Table 1.

Sentence	Conflict- ing rules	Strength of Category Membership		Cond wt	Bias	Net Input
1. The potato moved.	4	<u>potato</u> in <u>SEMOBJ</u> _{subj} (.983)	"—" in <u>NULL</u> _{obj} + .841)	× 8.57	- 17.14	= -1.49
	6	<u>moved</u> in <u>MOVED</u> _{verb} (.624)	"—" in <u>NULL</u> _{obj} + .841)	× 4.90	- 9.81	= -2.62
2. The ball hit the boy with the dog.	3	<u>ball</u> in <u>BRKR/ATE</u> _{subj} (.999)	<u>hit</u> in <u>GARN/POSS</u> _{verb} + .925)	× 7.61	- 15.21	= -.568
	1	<u>boy</u> in <u>ED/AN</u> _{obj} (1.0)	<u>dog</u> in <u>GARN/POSS</u> _{w-NP} + .877)	× 6.85	- 13.69	= -.833
3. *The fork moved.	6	<u>moved</u> in <u>MOVED</u> _{verb} (.624)	"—" in <u>NULL</u> _{obj} + .841)	× 4.90	- 9.81	= -2.63
	4	<u>fork</u> in <u>SEMOBJ</u> _{subj} (.850)	"—" in <u>NULL</u> _{obj} + .841)	× 8.57	- 17.14	= -2.65
4. *The boy hit the lion with the ball.	1	<u>lion</u> in <u>ED/AN</u> _{obj} (.0)	<u>ball</u> in <u>GARN/POSS</u> _{w-NP} + .978)	× 6.85	- 13.69	= -6.99
	2	<u>lion</u> in <u>SEMOBJ</u> _{obj} (.0)	<u>ball</u> in <u>UTENSIL</u> _{w-NP} + .614)	× 5.96	- 11.91	= -8.25

Table 3: Examples from the simulation run of Table 1 showing sentences which meet the conditions of two rules. Calculation of the net input to the condition unit of each rule (see Section 2) demonstrates how conflicts are resolved through category strengths. (* indicates a sentence that fires an incorrect rule and the subscripts subj, obj, and w-NP indicate the slot in the input corresponding to the category. Firing rules are underlined).

something being edible or animate, it simply has discovered that the words in this category are used in a similar fashion. The label **SEMOBJ** indicates a *semantic object*, something that is typically acted upon. The rest are self explanatory.

Note that RuleNet has reduced the 12 word categories used in McClelland and Kawamoto's sentence generators to a compact set of seven. This is possible because there is considerable redundancy in their generator, as is the case in natural language in general. RuleNet, however, only needs to discover categories sufficiently diverse to distinguish one condition from another. In addition, most words appear in more than one category and many appear to differing degrees of membership. The strengths reflect the usage of words in various categories in the training data. Thus a **bat** is more animate (**ED/AN**.999) than a breaker (**BRKR/ATE**.659) or a possession (**GARN/POSS**.975). Several words fit into counterintuitive categories, such as **ate** in **BRKR/ATE**. This suggests a much smaller categorization can be devised to account for the usage of these words than intuition might suppose. Note, for example, that **ate** is the only verb in the category **BRKR/ATE**. Therefore, in sentences where there is a word of this category in the verb slot, that word is guaranteed to be **ate**. This is what allows RuleNet to induce rules such as rule 5 in Table 2. Finally, all of the words in the category **MOVED** except **moved** go unused because this category only appears once, in rule 6 in the context of the verb slot. The remaining words in **MOVED** and **ate** in **NULL** are artifacts of learning and play no role in the performance of the net.

To see how differing levels of category membership facilitate conflict resolution, examine the six rules in Table 2. RuleNet was not able to devise a set of rules that are mutually exclusive over the training set. Consider rule 6. The condition on the input is that the verb must be in **MOVED** and that the object slot must be empty. The action indicates that the subject should be copied to the agent slot and that a fixed symbol **self** should be written into the patient slot. Three types of sentences produced by McClelland and Kawamoto's sentence generators meet the condition of this rule:

- (1) The {boy, girl, man, woman} moved,
- (2) The {bat, chicken, dog, wolf, sheep, lion} moved,

- (3) The {ball, doll, paperwt...} moved.

What we observe in the simulations, however, is that rule 6 is fired only by (1) and (2) and rule 4 is fired by (3). What's going on here? Rule 6 also matches for (3), as well as for (1) and (2). Why does rule 4 fire for (3) instead of rule 6?

First, note that rule 4 also matches the condition of sentences of type (3). Thus, at a purely symbolic level these two rules should stand an *equal* chance of firing, although rule 4 always fires. What has happened is that RuleNet has actually imposed its own kind of conflict resolution by weighting the membership of instances in categories, as seen in Table 3. Sentence 1, "The potato moved," is of type (3) and should fire both rules 4 and 6. As shown in Table 3, however, when applied to rule 4 "The potato moved" results in a net input to condition unit 4 of -1.49 and when applied to rule 6 it results in a net input of -2.62 to condition unit 6. Since condition unit activations are sigmoidal and normalized, the resulting outputs of condition units 4 and 6 are .73 and .27 respectively, which is sufficient to correctly perform the winner-take-all decision required for firing a single rule.

Interestingly, because all six rules are binary conjunctions and because the conjunctive bias constant L is -2, the magnitude of the condition weights is irrelevant in this example in resolving conflicts. Consider any two conjunctive rules i and j and assume all category strengths are 1.0. Then the net input to condition unit i is $net_i = (1.0\phi_i + 1.0\phi_i) - 2\phi_i = 0.0$ and the net input to condition unit j is $net_j = (1.0\phi_j + 1.0\phi_j) - 2\phi_j = 0.0$. Regardless of what value is assigned to ϕ_i and ϕ_j the net input to these two units and their corresponding activations will be equal. In this example and this set of rules, then, graded categorization is essential to conflict resolution.

The context dependent interaction seen in sentence 1 of Table 2 can be used to resolve conflicts based upon very subtle differences in meaning. Sentence 2 in Table 3, "The ball hit the boy with the dog" for example, meets the conditions of both rules 1 and 3. Rule 1 corresponds to an interpretation in which **ball** is animate and **dog** is a possession of **boy**. The interpretation in rule 3 is of **ball** as an instrument of an unspecified agent. As shown in Table

3, the net input to condition units 1 and 3 is $-.833$ and $-.568$ respectively, resulting in a normalized activation of $.456$ for rule 1 and $.544$ for rule 3. Rule 3 fires rather than rule 1. The small difference in activations is due to the fact that this sentence is not in the training set and to the relative acceptability of either interpretation of **ball** in the context of the data. Presumably the network prefers rule 3 because it has only seen **ball** playing a passive role in other examples. Note that a third possible interpretation of **ball** as agent and **dog** as instrument (rule 2) is not even considered because **boy** is not in **SEMOBJ** and **dog** is not in **UTENSIL**.

It is precisely this same disparity in the strengths of category membership and condition weights that leads to the errors in RuleNet's performance on the training set. In this run RuleNet incorrectly mapped two of the 152 training examples, sentence 3 in Table 3, "The fork moved" and sentence 4, "The boy hit the lion with the ball." Sentence 3 is similar to sentence 1, except rule 6 fires instead of rule 4, resulting in **fork** being assigned to agent rather than patient. Sentence 4 fires rule 1 instead of rule 2: **ball** is interpreted as the possession of **lion** rather than the instrument used by **boy**. While RuleNet's output in these two examples did not match the targets in the training set, the interpretations are not preposterous, e.g., if the fork were in a Walt Disney film or the boy were Superboy. Similar effects appear in the 219 errors RuleNet makes on the test set of 1355 sentences.

Thus, RuleNet performs conflict resolution at the level of categories, which enables the network to exploit the strengths of individual category memberships as well as surrounding context. Just as humans may view a ball as having a graded and context dependent quality of animateness, so can RuleNet make use of this fact. In addition, RuleNet induces this information from the data itself. No explicit categorization or rating of category membership is required of the modeler.

As suggested earlier, RuleNet also has available to it a priority ranking of rules in the form of the parameter ϕ in the condition template (Section 2). Consider an input which matches each of two rules, i and j , with a simple condition. If the category strength for the condition in each rule is 1, then which rule fires will be determined exclusively by the magnitude of ϕ . For example, assume in rule i that $\phi_i = 1.5$ and in rule j that $\phi_j = 2$. The net input to $i = (1.5 \times 1) + K$ and the net input to $j = (1 \times 2) + 2K$, so j will always fire. In addition, there is no reason why RuleNet might not combine conflict resolution at the category level and at the level of rules.

4 Conclusion

While purely symbolic rule-based systems rely on broad and somewhat generic conflict resolution heuristics, RuleNet suggests a dynamic, flexible technique for rule arbitration. This technique was not designed into the network architecture; rather, it was a consequence of the numeric weights and activation rules of the connectionist network. Indeed the fact that RuleNet could resolve rule conflicts at all surprised us. It took some study of the inner workings of the network for us to understand the clever strategy that RuleNet had discovered. The strategy elegantly combines category membership strength with a fixed rule strength.

The strategy makes use of the goodness-of-match between input instances and the categories that form the rule conditions. Graded matching has often been touted as a potential benefit of connectionist approaches to rule-based reasoning (e.g., Touretzky and Hinton [1988]), but this is the first convincing demonstration of which we're aware that connectionist rule-based systems can make use of graded matching to obtain a powerful and elegant rule selection mechanism.

References

- [Jacobs, et al, 1991] R. Jacobs, M. Jordan, S. Nowlan, G. Hinton. Adaptive mixtures of local experts. *Neural Computation*, 3:79-87.
- [McClelland and Kawamoto, 1986] J. McClelland and A. Kawamoto. Mechanisms of sentence processing: assigning roles to constituents. In J.L. McClelland, D.E. Rumelhart, and the PDP Research Group, *Parallel Distributed Processing: Explorations in the microstructure of cognition*, Vol. 2. Cambridge, MA: MIT Press/Bradford Books.
- [McMillan et al., in preparation] C. McMillan, J. Alexander, M.C. Mozer, and P. Smolensky. *Behavior in rule-governed domains: a connectionist account*.
- [McMillan, 1992] C. McMillan. *Rule induction in a neural network through integrated symbolic and subsymbolic processing*. Ph.D. Thesis, University of Colorado, Boulder, Colorado, January.
- [McMillan, et al, 1992] C. McMillan, M.C. Mozer, and P. Smolensky. Rule induction through integrated symbolic and subsymbolic processing, in J.E. Moody, S.J. Hanson, & R.P. Lippmann (eds.) *Advances in neural information processing systems 4*, San Mateo, CA: Morgan Kaufmann.
- [Sun, 1991] R. Sun. Connectionist models of rule-based reasoning. In *Proceedings of the 13th Conference of the Cognitive Science Society*, pp. 437-442. Hillsdale, NJ: Lawrence Erlbaum and Associates.
- [Touretzky and Hinton, 1988] D. Touretzky and G. Hinton. A distributed connectionist production system. *Cognitive Science*, 12(3), pp. 423-466.
- [Towell and Shavlik, 1992] G. Towell and J. Shavlik. Interpretation of artificial neural networks: mapping knowledge-based neural nets into rules. in J.E. Moody, S.J. Hanson, & R.P. Lippmann (eds.) *Advances in neural information processing systems 4*, San Mateo, CA: Morgan Kaufmann.