Connectionism and Explanation

Eugene Charniak
Department of Computer Science
Brown University, Providence RI

I gather that the panel on connectionism was picked to have a variety of viewpoints represented, some very pro, some very against, and me – I take an extreme waffle position. I really like connectionism, and I wish it would work for me. But so far it can't and theoretical breakthroughs would been needed to change things. I will try to explain here why I see things this way.

Before I start, however, it would be a good idea to clear up one possible source of confusion. My recent work has been on the problem of explanation in stories: how, given a story one can assign a motivation to a character based upon his or her actions. I also believe that the processes which are needed to do this will shed a lot of light on traditional parsing issues. The source of confusion comes from the fact that my recent models of explanation have involved a process of *marker passing*, or *spreading activation*. Basically I am using a breadth first search in an associative network to find connections between concepts, in the hope that such connections will suggest explanations. My typical example is

Jack wanted to kill himself. He got a rope.

Here the connection between kill and rope would be the clue.

Many people upon seeing this work hook me up with the connectionist school. This is not correct. I do not consider myself a connectionist, and real connectionists do not consider me one either. At best by marker passer might be seen as indicating "localist" (as opposed to "distributed") connectionist leanings, since some of what Jerry Feldman and his students do have some of the same flavor. But marker passing is only a small part of my system, and after it is finished I feel free to use deduction, unification, search, and, heaven forbid, *cons*.

Nevertheless I am sympathetic to connectionism, and to give some idea why, let me discuss a minor knowledge representation problem which I recently encountered. I have already noted that I am interested in the problem of explanation in language comprehension. One obvious idea is to use the objects in a story as a source of possible explanations. So, upon seeing a sentence like "Jack got some milk" we might suggest explanations like "He will eat cereal" "He will drink the milk" etc. Presumably we know that milk is put over cereal, and that milk is a beverage, and beverages are typically used for drinking. Thus it seems reasonable to index activities by the objects that get used in them (there could be other ways to index as well) and then, given an action like Jack's getting milk, look at milk, and the things above milk in the *isa hierarchy* for the actions which are indexed there. Naturally one must then decide between the possibilities, or put off the decision in hopes of further information, but how this is might be done need not concern us here.

Now consider the following facts which one might wish to express.

1) Shopping takes place at stores
2) Supermarket-shopping is one kind of shopping
3) Supermarket-shopping takes place at supermarkets. (it has other distinctive characteristics as well, but we will ignore these)
4) Supermarkets are one kind of store.

Facts in this form lead to what I have taken to calling a square formation, because when written down as an associative network they form a square, as shown in Figure 1. I suspect this is quite common in these kinds of representations, because often one wants to store the

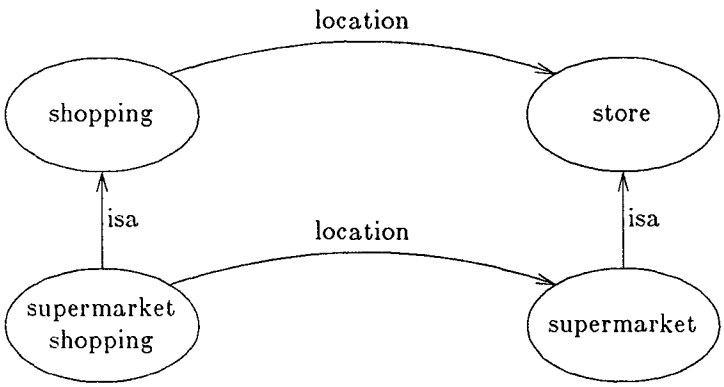

Figure 1.

information at many levels of generality. For one thing, if we were only told that Jack went to the store, one might still want to infer that he will shop, and secondly, many facts about specific kinds of shopping can be expressed at the higher level, and thus save space by not including them at all the lower nodes in the hierarchy: for example, the fact that shoppings typically start by going to the store.

So far so good, but now consider what happens when one uses the object hierarchy for finding explanations for "Jack went to the supermarket". It will suggest supermarket shopping, which is fine, but it will also suggest plain old shopping, since supermarkets are stores, and we use the suggestions from all (or at least many) levels of the isa hierarchy (remember that for milk we wanted suggestions from both "milk" and "beverage"). The problem is that we have redundancy. It appears superficially that we have two independent suggestions as to Jack's motivation for going to the store, one being supermarket shopping, and one be shopping, but really they are the same. Somehow this has to be weeded out.

I do not mention this as an example of a really tough problem. It is pretty easy to think of ways to get rid of the unwanted motivations, or, as my current system does, consider both and rank the more specific as better. Rather this is the kind of minor annoyance which we have to put up with all the time. I can solve it, but it sure would be nice to have a representation in which such things never came up in the first place.

It may be wishful thinking, but it seems that this problem would not come up if I were to use a connectionist model of knowledge representation. For those of you not familiar with connectionist networks, let me just give an example of some work by Rumelhart which has some relevance to my problem. Rumelhart was trying to show how one could model schemas (which are pretty much the same as frames and scripts) using connectionism. What he did was to create nodes corresponding to the various objects one finds in a house: bathtubs, couches, clocks, etc. He would then present typical rooms to the system and modify the strength of connection between nodes so they became proportional to how often the objects were found in the same room. For example, sink and bathtub would be highly connected, as would sink and refrigerator, but refrigerator and bathtub would be negatively correlated. He would then turn on some nodes, like clock and sink, and look to see what else got turned on. As you might expect, things common to kitchens would light up in this case. If one put in just chair the system would not be able to decide on the location but some things, like wall and ceiling would light up anyway, since all of the places which use chairs would have them. Perhaps bathtub would be turned off

as well, since chairs typically are not found in bathrooms.

The point is that connectionist networks are very good at filling out patterns from incomplete information, and doing this on the basis of lots of special cases. It should not take too much to imagine how this might work for my problem. One would give the system nodes for supermarkets, baskets, shopping, food, pushing baskets, bakery-shops, taking a number to be waited on, etc. One would also provide many examples of common patterns to assign the weights between nodes (or possibly but in weights by hand). Then if you put in supermarket you would get supermarket-shopping, along with baskets, etc., whereas if you put in store, you would not get the details, but only those things which all shoppings could agree upon, like a buying event, going to the store, etc. Thus we get the effect of an isa hierarchy without the precise mechanism, and the square problem goes away.

To keep my mind flexible I occasionally think about how my work would look in a connectionist model – the above observation was the result of one of such times. I find it interesting that the problems I run into never seem to have counterparts in the connectionist view of the world. It really is a different way of thinking about things.

So why don't I adopt the view? The answer is clear if we take the earlier thought experiment, about a connectionist model of shopping events, and really ask what it would look like in detail. One can, as I mentioned, create nodes for the various objects used in various stores, and for the various actions done in them. But suppose I get in a sentence like "Jack went to the supermarket" and I want to infer that he will be shopping. I can have a "supermarket-shopping" node, but to infer that *Jack* will be shopping by this method I would need a "Jack-supermarket-shopping" node. Given that I never heard of Jack prior to this story, this is obviously problematic. How would such a node be created? What would it be connected to? Furthermore, this problem is compounded by a second, which we might call the "infer everything" problem. The square problem cannot come up in a connectionist model because there is only one body of nodes, and it is their joint action which makes up the explanation, by, in effect, creating a "picture" albeit a very abstract one, of the situation which is envisioned as the explanation. If this "picture" is all one has, then it has to have everything filled out since, it is not clear how, without learning more, it could be modified. Thus, if this picture is to have any explanatory power, it would have to include nodes for things like "Jack get basket" "Jack pick up food" "Jack put food in basket" "Jack take food to checkout counter" etc. So it is not just that we would need a new node for "Jack supermarket shopping", we would need nodes for everything else as well. In normal AI knowledge representations this is not necessary. It is perfectly possible to *construct* new data structures (based upon deductions from the general plan for shopping) which represent the details of Jack's activity. I cannot see how this could be done in a connectionist scheme.

More generally, the problem of representing new propositions in connectionist networks is a real mess. The connectionists know about this however, and there has been some work on the topic. The basic idea is that one uses the state of the entire network to represent a proposition rather than concentrating it at a node. So, one might have one set of nodes which represent the first argument of a proposition, one for the second, and one for the predicate, and each set of nodes could indicate different individuals, depending on the pattern. The networks could be trained so that, say, if "father-of" was the proposition, it would tend to like "jack" as argument one, and "ann" as argument two, assuming one wanted to store the fact that Jack is the father of Ann. Using the connectionist ability to complete patterns, one can also see how such a network might fill in "jack" if both "father-of" and "ann" where put into the appropriate places. David McClelland does something like this in his work on case assignment using connectionist networks.

There are other ways to repreresent propositions as well (perhaps best being the work of David Touretzky), but they all loose what is so nice about the unsophisticated version of the networks. Before, the network as a whole represented the situation as a whole, and filling in

gave us things like motivation etc. Now the network just represents a single proposition. Filling in gets us the rest of the proposition, which is nice if one wants to find propositions on the basis of partial content, but I all ready know how to do that. AI has a whole bag of tricks for solving this problem, and by and large they do it much better than connectionist versions. What I want is something to fill in the context, but this the proposition representations schemes have not been able to do.

This is not to say that the problem is unsolvable. In my imagination, just beyond an obscuring haze, is an idea that one might combine the "entire scene" type networks and the propositional ones. One would have a network in which lots of connected propositions are all represented at once, with special sets of nodes for the objects which bind their variables. Jeff Hinton believes in something like this, and while I am around his infectious enthusiasm I can almost believe it too.

But I have not been able to penetrate the haze, and thus I think it is still fair to say that connectionist networks cannot represent propositions at all. To the degree they can it is like a horse walking on two legs – it does not do it very well, and you loose all that is distinctive about the creature. I use a horse because I want to gallop. If I have to walk I would rather do it in Lisp.