ED 308 709 FL 018 072

AUTHOR	Samad, Tariq
TITLE	A Connectionist Network That Learns To Process Some (Very) Simple Sentences.
PUB DATE	Oct 86
NOTE	14p.; In: Proceedings of the Annual Eastern States Conference on Linguistics (3rd, Pittsburgh, PA, October 10-11, 1986). The complete Proceedings are available from the Dept. of Linguistics, Ohio State University, Columbus.
PUB TYPE	Reports - Descriptive (141) -- Speeches/Conference Papers (150)
EDRS PRICE	MF01 Plus Postage. PC Not Available from EDRS.
DESCRIPTORS	*Algorithms; Case (Grammar); Classification; Computer Storage Devices; *Information Networks; Input Output; *Language Processing; *Sentence Structure
IDENTIFIERS	*Connectionism

ABSTRACT

The application of the "back-propagation" learning algorithm to the task of determining the right set of features corresponding to the words in an input sentence is described. Features that are specific to particular nouns and verbs, that indicate whether a nominal constituent is singular or plural, definite or indefinite, and that furnish case-frame information, are discussed. On examination, it appears that the network has learned concepts appropriate to the domain of natural language processing. The learning also generalizes well to novel sentences. Three related experiments are described. The shortcomings of the network are discussed, and ideas are suggested for an alternative model that should overcome some of these shortcomings. (Author/DJD)

A Connectionist Network That Learns To Process Some (Very) Simple Sentences

by

Tariq Samad

from

Proceedings of the Third

EASTERN STATES CONFERENCE ON LINGUISTICS

The University of Pittsburgh

Pittsburgh, Pennsylvania

October 10-11, 1986

Fred Marshall
Editor
The University of Pittsburgh

Ann Miller
Associate Editor
The Ohio State University

Zheng-sheng Zhang
Associate Editor
The Ohio State University

A CONNECTIONIST NETWORK THAT LEARNS TO PROCESS SOME (VERY) SIMPLE SENTENCES

Tariq Samad
Corporate Systems Development Division, Honeywell Inc.,
1000 Boone Avenue North,
Golden Valley, MN 55427

ABSTRACT

Connectionist networks are becoming popular models for research in natural language processing. This paper describes an application of the "back-propagation" learning algorithm to the task of determining the right set of features corresponding to the words in an input sentence. Features that are specific to particular nouns and verbs, that indicate whether a nominal constituent is singular or plural, and definite or indefinite, and that furnish case-frame information, are included. On examination of the network after learning, it appears that the network has learnt concepts appropriate for the domain. The learning also generalizes well to novel sentences. Three related experiments are described. The shortcomings of the network are discussed, and some ideas are suggested for an alternative model that should overcome some of these shortcomings.

I. Introduction

Much interest has been generated by recent work in connectionist networks--networks that consist of densely interconnected but functionally simple processing units. There is optimism that connectionism can provide a promising alternative to the rule-based, symbolic-processing framework that has been dominant in artificial intelligence (and computational linguistics). Such aspects of human cognitive function as the sequence of learning stages, the acquisition of concepts, tolerance of error and noise in sensory input, and graceful degradation in the face of minor damage, have been problems that have eluded clean solutions in the traditional framework. Work in connectionism, on the other hand, has demonstrated that these problems can be solved almost as a side-effect of the problem-solving architecture.

Connectionism seems particularly appropriate for a domain such as natural language processing, which defies attempts at cleanly structured approaches, and in which "errors" in the input (such as ungrammaticality and, in writing, misspellings) are frequently encountered.

This paper describes some initial results of an ongoing investigation of a particular connectionist architecture as applied to the task of learning features corresponding to the words in a simple sentence. These features include case-role information. A connectionist network has been developed as the vehicle for this investigation. Unlike some other connectionist approaches to natural language processing, the input to this network is not pre-processed--i.e., it is a textual sentence, not the output of a parser.

The next section briefly reviews some previous work on connectionist approaches to natural language processing. Section III outlines the "back-propagation" architecture on which much recent work, including this one, is based, and it describes the details of the network that I have been using. Section IV describes three experiments that have been conducted with the network. Section V discusses the features and limitations of the current model, and Section VI sketches out an alternative model that should work better.

II. Previous Work

Parsing, case-role assignment, and word-sense disambiguation have all been investigated in the context of connectionism. Waltz and Pollack (1985) describe a connectionist network that disambiguates words based on semantic associations with other words in the sentence. These semantic associations are modelled by the complementary processes of "spreading activation" and "lateral inhibition". The disambiguation process is not "rule-based"; instead there are links (excitatory and inhibitory) between concept nodes (concepts include words such as "shot" and "bucks", "readings" such as "shot", "fire", "resist", and syntactic categories such as "N" and "VP") and the concept nodes have real-valued activation levels associated with them. Through an iterative relaxation procedure, the network settles into a stable state. Waltz and Pollack show how their network can account for the context-sensitive readings of ambiguous words, and how the non-monotonic change in activation level can explain the phenomenon of garden-path sentences. In the context of this work, there are two limitations of their model: the issue of how the links are formed is not discussed (the appropriate links have to be hard-coded for each sentence), and the parsing mechanism is still a traditional chart parser.

Selman (1985) has developed a connectionist network that is based on a context-free (and non-recursive) grammar. Given an input "sentence" such as "noun verb preposition determiner noun," the network is put through a simulated annealing stage. At the point of equilibrium, the active units in the network (those representing syntactic categories--there are also "binder" units in the network that are used to inhibit competing parse-fragments) trace out the parsed structure. Selman's network is limited to finite-length sentences, and the question of how the structure of the network can be learnt is not addressed (a compiler transforms a rule specification of the grammar into a network).

Just recently, McLelland and Kawamoto ('986) describe a system that takes as input a parsed sentence (except that prepositional phrases are not attached) and that produces the case-role assignments for the nominal constituents. Words in the input are represented by sets of "micro-features". Theirs is a learning system; the system is trained on a set of examples, and then tested on novel sentences (and on novel sets of microfeatures). McLelland and Kawamoto discuss as a possible extension of their system a multi-layer network that learns the microfeatures themselves. The network described in this paper does just that. (My research was carried out in ignorance of [McLelland and Kawamoto, 1986], which I obtained as this paper was being written.)

III. The Back-Propagation Architecture

A fair amount of recent work in connectionism, including (McLelland and Kawamoto, 1986) and this paper, is based on the "back-propagation" architecture (Rumelhart, Hinton, and Williams, 1985), which is an elaboration of perceptron learning (Minsky and Papert, 1969) to multi-layer non-linear networks.

Back-propagation networks partition units into layers: an input layer, zero or more "hidden" layers, and the output layer. Usually, all connections are from units at a lower layer (the input layer being the lowest), to units at a higher layer. Usually, then, back-propagation networks that have been built have been "feed-forward" networks, with no cycles or feedback. The connections between units have real-valued weights on them, and each unit has a threshold associated with it. The output of a unit i in a hidden or output layer is a non-linear function of the weighted sum of its total input.

The attraction of the back-propagation architecture is the presence of a learning procedure that can be used to discover an appropriate set of connection-weights and threshold-strengths. The learning procedure requires the generation of a set of input-output pairs which serves as the training set. For each element of the training set, the learning procedure essentially consists of the following (for a three-layer network):

```
1.  Present an input to the network.
2.  Have the network compute its output.
3.  Compare the actual output to the expected output.
    If they differ (within some tolerance), then:
           Adjust the connection-weights between the hidden and
                output units, and adjust the output thresholds.
           Adjust the connection-weights between the input and
                hidden units, and adjust the hidden thresholds.
```

The amount by which each weight is adjusted is given by the learning algorithm. Details can be found in (Rumelhart, Hinton, and Williams, 1985). After iterating through the training set some number of times, the network can often discover a set of weights and thresholds that result in error-free performance on the training set. Novel sentences can then be given to the network and its response tested.

The network I have been using is a three-layer one, with one layer of hidden units. In the first two experiments described later, the network is fully connected; that is, each unit in a layer is connected to each unit at the next higher layer.

The input layer can be thought of as a seven-word "window" on the sentence: the sentence is "rippled" through the input layer. At any point, the center word in the window is being processed; the three preceding and three succeeding words (all or some of which can be "nil" for words at or near the beginning or end of the sentence) determine the "context" that is available to the network (since there are no cycles or feedback, there is no mechanism for "memory" in the network). This "rippling" scheme, originally used by Sejnowski in his NETtalk system (Sejnowski and Rosenberg, 1986), overcomes the restriction of finite length inputs. The figure at the end of this paper shows the structure of the network.

In the first two experiments described in this paper, each word in the input window is represented by eight units (thus in these cases there are fifty-six input units). Each word in the lexicon has an eight bit binary encoding. The input units for each window-position are assigned the encoding for the word in that position (thus the value of an input unit is always 1 or 0). The encodings were derived without any bias, except that all and only plural words had their eighth bit set to "1".

For each word in a sentence, the network computes the values of the hidden units first, and then the values of the output units. Again in two of the three experiments described, each of the output units represents a "feature". The set of features activated for a particular word in a sentence is the output for that word. For example, the word "John" in the sentence "John gave Mary a book" should activate the following features: JOHN, DEFINITE, SINGULAR, $AGENT (where $AGENT refers to the case-role of "John"). There is one output unit labelled "*WAIT*" which is the desired output for all determiners, prepositions, and auxiliaries.

The output of the network, then, is an association of features with words in the sentence; the output is not a parse-tree. It is not obvious how a traditional parse of a sentence could be realized by such a network. However, the case-frame information provided by the network may be sufficient for most subsequent processing needs.

The maximum and minimum possible values for units were 1 and 0 respectively. During the learning phase, an output unit was considered ON when its value was above 0.8, and OFF when its value was below 0.2. During the testing phase, these activation thresholds were relaxed; a unit was considered ON if its value was above 0.6, and OFF otherwise. The network is implemented in Zetalisp on Symbolics' 3600/3640 computers.

IV. Three Experiments

IV.1 Experiment 1

There were 25 hidden units and 21 output units in this experiment. Each output unit represented one of the following features: JOHN, JACK, JOAN, JILL, MAN, WOMAN, CHILD, GIRL, BOOK, NOTE, PAPER, STORY, DEFINITE, INDEFINITE, SINGULAR, GIVE, $AGENT, $OBJECT, $RECIPIENT, $PRED, *WAIT*.

The training set consisted of 47 sentences conforming to the following grammar:

S --> <animate-NP> gave <animate-NP> <inanimate-NP>

<animate-NP> --> <proper-noun> | <det> <animate-CN>

<inanimate-NP> --> <det> <inanimate-CN>

<proper-noun> --> John | Jack | Joan | Jill

<det> --> a | the

<animate-CN> --> man | woman | child | girl

<inanimate-CN> --> book | note | paper | story

The sentences were not randomly generated from the grammar; there were some deliberate omissions. 47 sentences constitute less than five percent of the total number (1,152) of distinct sentences that can be generated with the above grammar/lexicon.

The network took 108 iterations through the 47 sentences to learn perfectly. After the network had learnt, the hidden units were examined for "patterns". A couple of observations can be made:

i) It was fairly obvious how the network was identifying features such as DEFINITE and $AGENT. There were particular hidden units that were in one state when there was a definite noun in the center of the window, and in the complementary state when there was an indefinite noun. Similarly for other features such as $AGENT, $OBJECT, and $RECIPIENT. Most of these "hidden" feature detectors were independent of the central word itself. For instance unit 6 was ON for all indefinite nouns, and OFF for all but one definite nouns (including proper nouns). For some nouns, definiteness was redundantly encoded by other units as well.

ii) The network learnt the concepts of "proper-noun", "animate-common-noun", and "inanimate-common-noun". There were patterns of hidden unit activity that were unique for each of these. (Note that the network was not "asked" to learn these concepts; no output units were labelled with these grammatical abstractions.) There were overlaps between the concept patterns.

Proper nouns and animate common nouns had 10 hidden units with common activity, inanimate common nouns and animate common nouns 8, and inanimate common nouns and proper nouns 7. (Seven hidden units were in fact always in one state--an indication that 18 or fewer hidden units are sufficient for this task.) Thus along with discovering these abstractions, the network did in some sense discover that there was less in common between inanimate common nouns and proper nouns than either of the two other pairings. (Animate common nouns and proper nouns had the same case-role constraints, and inanimate common nouns and animate common nouns both required preceding determiners, whereas proper nouns and inanimate common nouns differed in both these respects.) Since there were units that were in one state of activity for inanimate and animate common nouns, and in the complementary state for proper nouns, the network had also learnt to distinguish between common nouns and proper nouns.

The network was tested on 12 "difficult" sentences, composed for the most part to complement the training set. The errors exhibited on these sentences were due to the insufficiency of training set coverage, as the following examples illustrate:

i) All "Joan gave <proper-noun> <inanimate-NP>" sentences (there were three of them) in the training set had "Jack" as the "<proper-noun>". Two of the test sentences were of the same form and had "Jill" and "John" as the "<proper-noun>". In both these cases, the correct unit for the recipient was activated, but at a below activation threshold level. In both, the unit JACK was at a higher activation level than the correct unit.

ii) There was only one sentence of the form "The child gave <det> <animate-CN> <inanimate-NP>" in the training set, with "man" as the animate common noun and "the paper" as the inanimate noun phrase. On the test sentence "The child gave the woman the note," the WOMAN unit was not activated to a threshold level for the word "woman". NOTE was correctly identified.

iii) There were no training sentences of the form "Jill gave <det> <animate-CN> <inanimate-NP>," and the network could not recognize "woman" in the test sentence, "Jill gave the woman a paper."

Most errors could be related to gaps in the training set. There were eight errors (incorrect output unit values) in the twelve sentences. Learning for INDEFINITE/DEFINITE, for SINGULAR, for the verb-unit GIVE, and for the $PRED and case-role units, for all of which of course the network had many more examples than it did for specific nouns, seemed perfect.

IV.2. Experiment 2

Output units PLURAL and READ were added. The number of hidden units was still 25. The grammar was significantly extended to include the dative construction for the verb "give", passives (both variations in the case of "give"), and yes-no questions (for both active and passive voices). The grammar is given below:

```
S --> <animate-NP>      gave    { <animate-NP>        <inanimate-NP> }
                                { <inanimate-NP> to <animate-NP>    }

S --> Did  <animate-NP>   give  { <animate-NP>    <inanimate-NP> }
                                { <inanimate-NP> to <animate-NP> }

S --> { Ø   }    <animate-NP>      read     <inanimate-NP>
      { Did }

S --> { <inanimate-sing-NP> was } given to <animate-NP> by <animate-NP>
      { Was <inanimate-sing-NP> }
      { <inanimate-pl-NP> were  }
      { Were <inanimate-pl-NP>  }

S --> { <animate-sing-NP> was  }  given <inanimate-NP> by <animate-NP>
      { Was <animate-sing-NP>  }
      { <animate-pl-NP> were   }
      { Were <animate-pl-NP>   }

S --> { <inanimate-sing-NP> was }   read      by      <animate-NP>
      { Was <inanimate-sing-NP> }
      { <inanimate-pl-NP> were  }
      { Were <inanimate-pl-NP>  }
```

The lexicon was a superset of the lexicon for Experiment 1: "read" was added as a transitive verb, the plurals for all common nouns were added, "some" was added as the indefinite determiner for plurals, and the preposition "to" and the auxiliaries were added. Since the grammar in this case was more complex, 500 sentences were (randomly) generated from it and used as the training set. (Duplicates were not discarded during the generation; there were 483 unique sentences.) 500 is less than one percent of the total number of unique sentences that can be generated with the above grammar and the lexicon.

Learning took 29 iterations through the 500 sentences. (About 15,000 sentences were processed, as contrasted to about 5,000 for Experiment 1.)

Most of the patterns noticed in Experiment 1 were apparent here too. There were additional patterns that were due to the more complex grammar. For example, a particular unit was ON for passivized agents, and OFF otherwise. The particular encoding for plurality (bit 8 of the word representation set to 1), was learnt; when the representation for a proper noun was changed by making bit 8 equal to one, the PLURAL unit was activated (although in a few cases some other output units were affected as well).

The network was tested on 1000 randomly generated sentences. Learning was remarkably good. There were only 12 unit-errors for the 1000 sentences, and only 8 word-errors (some words had more than one wrong feature). I also took this opportunity to judge the choice of 0.6 as the activation threshold for testing purposes. A choice of 0.5 or 0.4 would have reduced the errors almost by half, as the table below shows:

Threshhold	Unit Errors	Word Errors
0.8	221	184
0.7	25	18
0.6	12	8
0.5	6	5
0.4	6	5
0.3	24	17
0.2	317	254

IV.3 Experiment 3

One problem with the network structure of the previous experiments is that although the activated output units are associated with the central word in the input window, yet there is no reason for the network to prefer associations with the central word over associations with non-central words. One mechanism for effecting such a preference would be to have relatively more connections with the central word and relatively fewer ones with the others. Thus as a third experiment, about 35% of the connections between the non-central words in the window and the hidden units were permanently set to zero (i.e., they were not adjusted during the learning process). The connections to be zeroed were chosen at random. (The experiment was also repeated with 50% and 43% zeroed peripheral connections; in both cases the network ended up stuck in a local minimum and could not learn the training set perfectly.)

For shorter execution time (each iteration in Experiment 2 took over one and a half hours), three other changes were made to the model. The number of hidden units was reduced to 15, words were expressed in 5 bits instead of 8, and the output units were encoded into 8 bits: N1-4, D1, C1-3. The "N" units encoded the 12 nominals, the "D" unit encoded the definite/indefinite distinction, and the "C" units encoded the case roles $PRED, $AGNT, $OBJ, and $OBJ2, as well as *WAIT*. There were no output units for the verbs ("give" and "read" again).

The learning corpus consisted of 100 active, declarative sentences. Both the dative (e.g., "John gave a book to Jill") and the indirect-object constructions (e.g., "John gave Jill a book") were allowed. No plural words were allowed. A set of 1000 sentences was similarly generated and used as the testing set.

The experiment was performed with both a fully connected network, and with the modified network. With a fully connected network, learning took 122 iterations. There were 352 unit-errors in the 1000 sentences. If we assume an average sentence length of six words, then 1000 sentences correspond to 1000 * 6 * 8 = 48,000 units, a (unit) error rate of about 0.7%. The more conservative word-error rate comes out to be about 4.5% (there were 269 word errors).

With the modified network, learning took 171 iterations. Unit errors dropped to 234 (0.5%), and word-errors to 193 (3.2%). Thus, when the network was forced to pay relatively more attention to the central word, performance was significantly better.

It was expected that for the fully connected network, there would be fewer errors with those output units that depended on non-central words for their correct determination. This was borne out by the fact that mistaken attributes of definiteness/indefiniteness were more prevalent in the modified network. However, the proportion of errors of case-role identification were relatively greater for the fully connected network, somewhat surprising since case-role identification does not depend only on the central word. (Note though that determination of the $OBJECT case-role unit did not in fact require any context, since the words that were possible objects could not fill any other case-role.)

The experiment was also repeated for non-encoded output units (i.e., with 19 output units). There were no disabled connections in this case. The network took 136 iterations to learn, and made the fewest of any errors on the 1000 sentence test set--311 unit-errors, 206 word-errors.

V. Features and Limitations

Let me first summarize some of the points made above.

i) It is interesting that learning based on relatively few sentences can generalize so well to other sentences. In Experiment 2 we saw that with a training set of less than one percent of the possible input sentences, the network made remarkably few errors (the unit-error rate comes out to less than 0.01%) on others.

ii) The network learnt by "discovering" concepts that were appropriate for the domain. Commonalities in the training set resulted in the hidden units abstracting these concepts. It is again remarkable that, with no other access to a pre-determined grammar than a fraction of the possible sentences that that grammar can generate (in particular, with no "innate" notion of the structure of the grammar), the network can discover the non-terminals that were used in the grammar. Although it is much too early to make any such claims today, research in connectionism may someday present an alternative to the traditional linguistic assumption that learning language requires a tightly constrained mental structure.

iii) Performance after learning is better if the activation thresholds for the output units are relaxed to a value around 0.5.

iv) Decoded output units are better than encoded ones. The difference between the encoded network and the decoded network in the number of iterations required to learn is not really significant, and the other qualification that (about 20%) more weights have to be modified per iteration is only relevant for serial simulations.

On the other hand, there are shortcomings of the network as well:

i) Sentence fragments are inconsistently mapped: When given a full noun-phrase as input (either "<proper-noun>" or "<det> <common-noun>"), the appropriate (non-case-role) output units are activated; with just a common noun as input, they are not. An input consisting solely of "man", for example, will not activate the MAN output unit. There is a plausible explanation for this anomaly: In the learning phase, the network never sees common nouns without a preceding determiner. The output units for a noun (even those other than DEFINITE/INDEFINITE) are determined by some joint function of the determiner and common noun. The function that is learnt could be a more complex one than a more compositional analysis, and one can say that the network has settled to some non-global minimum. There might be two possible approaches to force the network to find the global minimum: decrease the number of hidden units to a point where expressing the more complex function is impossible, or increase the cardinality of the training set. I intend to investigate both these approaches in the near future. Incidentally, the network has no problem with number disagreement: the number marking on the noun is picked up.

ii) One layer of hidden units may not be enough for the hidden units to learn to represent all the concepts implicit in the training sentences. It is well known that not all possible logical relationships among one layer of units can be expressed in the next higher layer (Minsky and Papert, 1969). This was the original rationale for a hidden layer, and if the hidden layer is to be "meaningful" and "complete", another layer of units must be inserted between the hidden layer and the input layer.

iii) The input to the network is constrained to be a simple sentence. Handling multiple clauses is a problem since the output units encode case-role information but do not indicate the verb to which the case-role attaches. The more general problem is that language has a nested, recursive structure. The model described in this paper is, conversely, "flat".

iv) Related to the above point is the limited context available to the network. The only information that the network can rely on when determining the output features for a particular word in a sentence are the three immediately preceding words and the three immediately succeeding words. This is clearly insufficient. In order to overcome this limitation in a meaningful fashion, some notion of "memory" has to be incorporated. Without memory, the processing of previously seen words cannot influence the processing of the current word. Incorporating a memory mechanism requires adding feedback connections in the network (that is, connections from higher layers to lower layers, or intra-layer connections). Such connections have been used in other connectionist architectures such as the Boltzmann machine (Hinton, Sejnowski, and Ackley, 1984), as well as in back-propagation (Rumelhart, Hinton, and

Williams, 1985). However, the learning algorithms that have been applied in these cases have been unfeasibly slow. There is some evidence that a constrained context mechanism is present in the human language facility, but the constraint has to be quantified simply in terms of lexical items (Marcus, 1980).

VI. Towards a Better Model

Finally, it is important to note that the network described in this paper, particularly in the first two experiments, is "general-purpose". It has not been customized for natural language processing. Essentially the same network has been used earlier by Sejnowski and Rosenberg (1986) for the task of learning the phonemic features of lexical and phonetic input.

In Experiment 3, the network was modified to make it intuitively more appropriate for the task. This was a minor modification, and did not address the more important issues raised above. Nevertheless, it was encouraging to note that the intuitively plausible modification resulted in significantly better performance. (The modified network also took longer to learn, but that statistic may be misleading--the value of the learning rate constant was chosen after some experimentation solely with fully connected networks.)

Experiment 3 suggests a further experiment. A network could be constructed that is explicitly modelled after some structural features of language (or English in particular). For example, some aspects of the structural compositionality of English could be captured by having units for nouns serving as input to units for case-roles; similarly, units for determiners need not be connected to words after the central word. Thus instead of learning totally from scratch, there would be a bias in the network appropriate for the task. The marriage of a somewhat general-purpose learning algorithm and a somewhat special-purpose network structure should lead to significantly better performance after significantly fewer trials.

The memory mechanism for such a model need also not be entirely general-purpose. Sentences with long-distance dependencies are often difficult for people to understand, and it would be a feature, not a flaw, of a model for which the same was true. Some of the work on "deterministic parsing" would be directly relevant here (Marcus,1980).

An "incremental" form of learning could also be investigated. The network could first be given nouns to learn, then noun (and prepositional) phrases, and finally full sentences. Some of the connections used for earlier stages of learning could be "frozen", so that learning a later stage does not cause the "forgetting" of what was learnt before. Tying the stages to successively more complex input should help the network make the right structural distinctions.

There is, of course, much work to be done and many details to fill out, but these seem promising ideas for future work.

REFERENCES

Hinton, G.E., T.J. Sejnowski, and D.H. Ackley (1984). Boltzmann machines: constraint satisfaction networks that learn. Technical Report CMU-CS-84-119, Computer Science Department, Carnegie-Mellon University.

Marcus, M. (1980). A Theory of Syntactic Recognition for Natural Language Processing. MIT Press.

McClelland, J.L., and A.H. Kawamoto (1986). Mechanisms of sentence processing: Assigning roles to constituents of sentences, in Parallel Distributed Processing: Explorations in the Microstructures of Cognition, McLelland, J.L., D.E. Rumelhart, and the PDP Research Group (eds.), MIT Press.

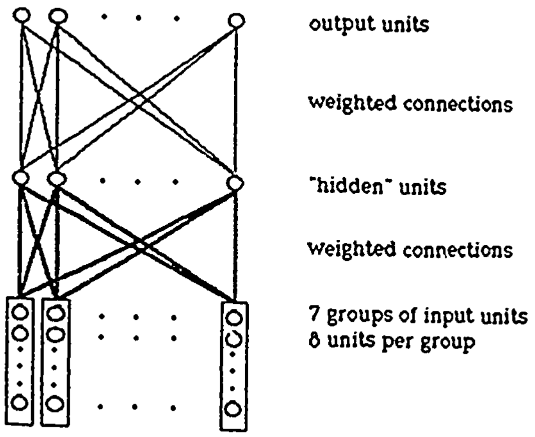
Minsky, M, and S. Papert (1969). Perceptrons. MIT Press.

Rumelhart, D.E., G.E. Hinton, and R.J. Williams (1985). Learning internal representations by error propagation. ICS Report 8506, Institute for Cognitive Science, UCSD, La Jolla, Ca.

Sejnowski, T.J., and C.R. Rosenberg (1986). NETtalk: a parallel network that learns to read aloud. Technical Report JHU/EECS-86/01, Electrical Engineering and Computer Science, Johns Hopkins University.

Selman, B. (1985). Rule-based processing in a connectionist system for natural language understanding. Technical Report CSRI-168, Computer Systems Research Institute, University of Toronto.

Waltz, D.L., and J.B. Pollack (1985). Massively parallel parsing, Cognitive Science, vol. 9, pp. 57-74.



The Structure of the Network
(Threshold weights not shown)