From Word Stream to Gestalt: A Direct Semantic Parse for Complex Sentences*

Bobby D. Bryant
Risto Miikkulainen
Department of Computer Sciences
The University of Texas at Austin
Austin, Texas 78712
(bdbryant,risto@cs.utexas.edu)

TR-AI98-274

26th June 2001

Abstract

The integrated processing-decoding network model of St. John and McClelland (1990) was revised to allow extracting the predicate content of complex sentences directly from an incoming stream of word tokens. The input stream was presented to the network without any syntactic markup such as bracketization, and the extraction was done without any explicit emulation of stacking, segmentation, or other such operations that are ordinarily associated with parsing a sentence. The lack of such explicit syntactic operations allowed a simulated neural network of minor complexity to be trained to the task under a simple regimen.

1 Introduction

Various challenges exist for any cognitive model of language processing, and scalable solutions suitable for use in simulated neural networks have been slow in arriving. Even such a basic matter as representing linguistic objects has been all but a show-stopper, and although neural networks offer a number of desirable processing capabilities, they face special challenges when used to model the cognitive operation of processing complex linguistic objects.

Representation of linguistic objects in the static structure of a simulated neural network is difficult because of the variety in the length and complexity of the objects. Variable length presents itself whether processing sounds into morphemes, morphemes into words, words into sentences, or sentences into discourses. Variable complexity presents itself in the latter two categories (at least), and is a factor regardless of whether the objects under consideration are viewed in terms of form or content.

For a network that processes sentences, variable length is an issue primarily relating to the inputs and variable complexity is an issue primarily relating to the outputs. Some sort of recurrency seems to be the most common way of addressing variable input length, with the activations in a recurrent layer converging toward a static representation of a whole as its elements are fed into the network piecewise (Elman 1990). There seem to be two major approaches to addressing variable complexity in the output: one extracts well-integrated chunks of the object as they become available, so that the whole is never explicitly represented in static form (Miikkulainen 1996); the other uses some form of coercion to force the formation of a static, compressed representation of the whole as a *whole*. Within this latter category there have been models producing compressed representations both of syntactic (Pollack 1988) and of semantic (St. John and McClelland 1990) structures. The experiments reported here in section 4 were an investigation into the utility and robustness of static semantic representations created by such a compression scheme for modestly complex sentences.

*This research was supported in part by the Texas Higher Education Coordinating Board under grant ARP-444.

Even with representations in hand, processing objects of variable complexity can challenge the design of a network model and raise questions of cognitive plausibility for the solution. Among the models mentioned above, for example, the RAAM (Pollack 1988) required explicit specification of constituent bracketization in the training set, and SPEC (Miikkulainen 1996) required explicitly training modular components to perform hidden tasks. The experiment reported here in section 5 attempted to bypass the former problem by using *semantic* targets, which are potentially derivable from the environment by a learner, and to bypass the latter problem by making do without overt modules and explicit hidden tasks altogether. To state these goals more clearly: an attempt was made to process sentences by direct transduction from representations of their surface form to representations of their semantic content in order to test the hypotheses that *syntactic structures are information-bearing structures* and that *information can be extracted from such structures without overtly performing a traditional syntactic parse*.

2 The Corpus

2.1 Sentences and Semantic Representations

In order to control the length and complexity of the sentence data, a hand-generated corpus was used for all the experiments reported here. The corpus consisted of records pairing the plaintext of a sentence with a set of *shallow semantic representations* for the content of its various clauses. The plaintext was specified in the corpus as a simple list of words without capitalization or punctuation, as shown in sentence (1).

(1) the pirate that chased the monkey saw the parrot

The semantic representations for the individual clauses were specified as lists of fillers for the relevant *theta roles* (Chafe 1970; Cook 1989), given in the order AGENT ACT PATIENT BENEFICIARY to make them easy to read. The shallow semantic representations (2) and (3) illustrate the fillers appropriate for the two clauses of sentence (1).

(2) PIRATE SEE PARROT —

(3) PIRATE CHASE MONKEY —

The dashes in representations (2) and (3) indicated that the verbs in those clauses did not make reference to anything in the BENEFICIARY theta role. In this corpus the BENEFICIARY slot was applicable only to clauses with ditransitive verbs, such as the relative clause in sentence (4). The semantic representation for that sentence is given in (5) and (6).

(4) the pirate that gave the parrot to the captain saw the monkey

(5) PIRATE SEE MONKEY —

(6) PIRATE GIVE PARROT CAPTAIN

Clauses with intransitive verbs, such as the simple sentence (7), lacked a PATIENT as well as a BENEFICIARY, and so had the dash as the filler for both these roles, as shown in semantic representation (8).

(7) the captain ran

(8) CAPTAIN RAN — —

As illustrated by sentence (4) and the associated semantic representation in (5) and (6), the argument structures of the various clauses in a sentence are independent of one another. For the connectionist implementation it was necessary to provide a constant number of slots for each clause sufficient for any clause in the corpus. In the implementation, a dash was treated as just another semantic symbol, with its own unique numeric representation.

The fillers in the semantic representations were specified in all-caps and without inflections. In future experiments we plan to parse out overt and implicit inflections in the plaintext as fillers for additional *markup slots* to be added to the semantic representations. All-caps were used to indicate that the model distinguishes between the *lexical words* in the surface form of a sentence and the *semantic symbols* evoked by those words. As described in section 3 below, a *word* and the associated *symbol* had independent representations in the connectionist implementation, and learning the arbitrary association between words and symbols was one of the challenges for the network. The use of separate representations of words and symbols is an elaboration of the notion of *identity constraints* in St. John (1992), and was included in the corpus for the purpose of cognitive modelling of the human ability to learn such associations.

Such a set of shallow semantic representations gives no indication of the relation between the various clauses in a sentence; a sentence such as (1) was simply taken to make two independent but simultaneously valid assertions about *the pirate*, namely those specified in the *shallow* semantic representations (2) and (3). Specifying more complex semantic relations between clauses is left for future work.[1]

The definite article *the* played no semantic role in the current corpus; it was not contrasted with the indefinite article nor with any of the demonstratives, but rather was used as a notionally unmarked form to allow construction of complete sentences for the parsing experiment described in section 5.

2.2 Content of the Corpus

The lexicon allowed expressing various simple statements about curious events in a tropical paradise (table 1). Sentences were formed from the words in the lexicon according to a set of rules about which nouns could serve as

Table 1: **Lexicon for the toy corpus.** The function words *that*, *the*, and *to* were used to build complete sentences from the words in the table.

Common Nouns	Intransitive Verbs	Transitive Verbs	Ditransitive Verbs
captain	fled	caught	gave
monkey	flew	chased	showed
parrot	ran	heard	sold
pirate		saw	

the subject/AGENT for the various verbs. (As the corpus contained neither middle nor passive verbs, the grammatical subject always identified the semantic AGENT.)

Modestly complex sentences were formed by allowing any or all of the nouns in a matrix clause to be modified by relative clauses. Such clauses might be of either the *subject extracted* or *object extracted* type, as illustrated in sentence fragments (9) and (10) respectively.

(9) ...that saw the monkey...

(10) ...that the monkey saw...

Additional syntactic variety was provided by allowing variations in word order that allow the indirect objects of ditransitive verbs to appear either with or without the preposition *to*, as illustrated in sentence fragments (11) and (12).

(11) ...gave the parrot to the captain...

(12) ...gave the captain the parrot...

The experiments described below were based on a corpus of 310 sentences created by this grammar. The corpus included intransitive, monotransitive, and ditransitive verbs, and allowed multiple relative clauses per sentence, averaging c. 1.9 each, for a total of 598 clauses, but without nesting clauses more than one deep.[2]

[1] In particular, sentences with relative clauses such as the one in sentence (1) might be expected to appear in discourse contexts where the matrix clause conveys new information and the relative clause conveys old information. Semantic material shared between such clauses is co-indexed in the discourse, allowing an *anaphoric unification process* to integrate the new material into a growing representation of the discourse. Taking sentence (1) as an example, if a discourse has previously established that, say, (PIRATE_{17} CHASE MONKEY —) and the new sentence asserts that (PIRATE_i SEE PARROT —) $\wedge$ (PIRATE_i CHASE MONKEY —), it is a straightforward guess that $i = 17$ and thus that (PIRATE_{17} SEE PARROT —). A number of other semantic clause-relations can be posited, such as *contingency* or *cause-and-effect*; for the present it should merely be noted that specification of the semantic relation between clauses will be different from and perhaps more complex than the purely syntactic notions of *matrix clause* and *subordinate clause*.

[2] Embeddings were limited to a single layer because of the combinatorial explosion of possibilities offered by even such a limited lexicon and grammatical structure. Nor was it possible, in a hand-generated corpus, to exhaust all sentence possibilities even with this limited depth of embeddings. Except where noted otherwise in the text, coverage was very nearly exhaustive for simple sentences and for complex sentences using

3 The Architecture

The architecture used for the experiments was a modification of that used by St. John and McClelland (1990) for forming *gestalt* semantic representations for single-clause sentences. In that architecture two related but functionally distinct networks were joined together, with a shared hidden layer serving simultaneously as the output of one and the input of the other (figure 1).

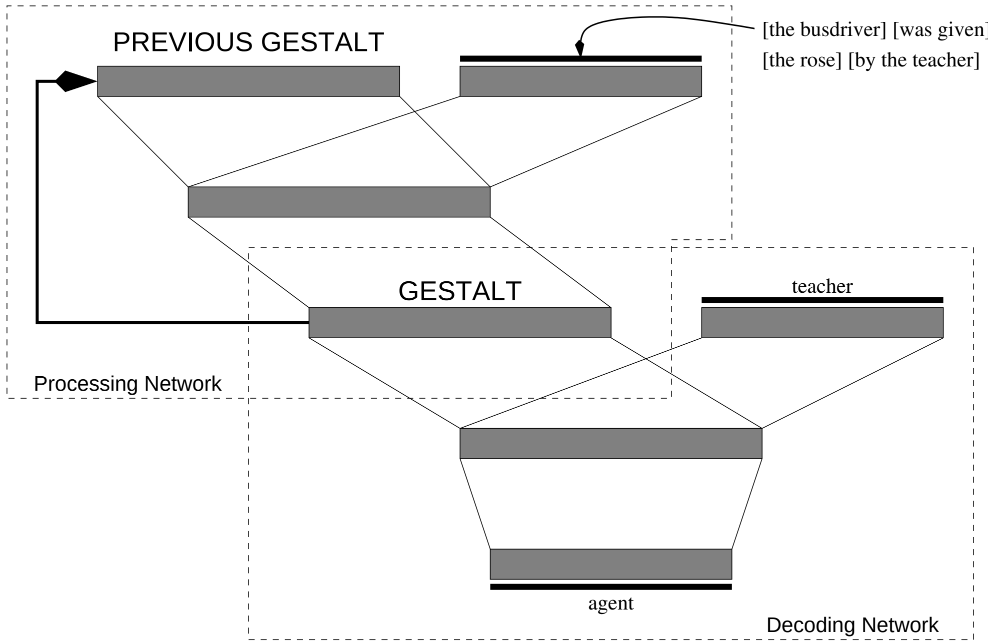

Figure 1: **Architecture of the St. John and McClelland's (1990) conjoined processing/decoding networks.** A single PROBE layer is used to query the gestalt. A sentence is fed into the processing network one grammatical constituent at a time to form the gestalt. Once the gestalt has been formed, a probe produces a response at the output. The probe can query the filler for a given semantic role, or vice versa.

In St. John and McClelland's architecture the upstream *Processing Network* had the task of learning to extract a distributed, case-based semantic representation, which the authors called the *sentence gestalt*, from an incoming stream of sentence constituents. The downstream *Decoding Network* had the task of learning to extract the bindings between specific semantic roles and their fillers from this gestalt representation. By conjoining the two sub-networks, St. John and McClelland were able to sidestep the usual requirement of providing static targets for the *Processing Network*: rather than having their values specified in the training set, the sentence representations in the gestalt layer were formed by coercing them to satisfy a *performance requirement* imposed by the decoding network. For example, as shown in figure 1, the constituents of the sentence *the busdriver was given the rose by teacher* were fed into the processing network sequentially to form a gestalt representation of the sentence, after which a probe with a constituent such as *the teacher* must return the role played by that constituent in the sentence (or vice versa). Training the network with the sentences and probes forced the network to form suitable gestalts for the sentences in their corpus.

A shortcoming of the architecture lay in its inability to cope with sentences consisting of more than a single clause. For example, with a sentence such as example (13), probing with the role *agent* should properly respond with the filler *monkey* for the matrix clause, but with *pirate* for the relative clause.[3]

(13) the monkey that the pirate chased saw the parrot

only the transitive verbs. Due to the combinatorial explosion, the space of possible sentences using the intransitive and ditransitive verbs is less densely sampled than that of the sentences using only the transitive verbs. Deeper embeddings and better-managed samplings of possible-sentence space are on the agenda for future work, which will use a machine-generated corpus. Note, however, that even in the long term we do not aim at parsing embeddings to an unbounded depth — rather, the capabilities of the model should be similar to the observed capabilities of humans.

[3] Since St. John and McClelland did not make an overt distinction between *lexical word* and *semantic symbol* in their model, I eschew using the all-caps convention described in section 2.1 when referring to the fillers for his PROBE and RESPONSE layers.

Similarly, probing with the filler *monkey* should properly respond with the role *agent* for the matrix clause, but with *patient* for the relative clause. As the architecture made no provision for distinguishing between clauses in such circumstances, the model was not capable of dealing with multi-clause sentences.

Generally following a suggestion in St. John (1992), the architecture was modified in an attempt to address that limitation. In the revised architecture, the *Decoding Network* was provided with multiple probe inputs, one for each slot in the shallow semantic representations specified by the corpus (figure 2). However, only a single response output

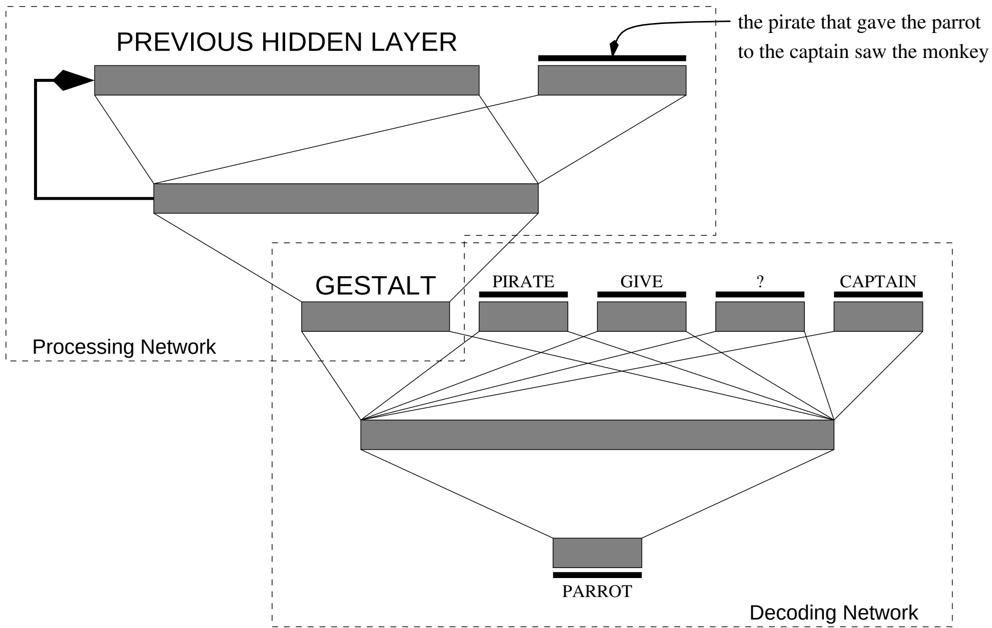

Figure 2: **Architecture of the enhanced network.** The enhanced network provides multiple probe layers, allowing disambiguation of queries relating to specific clauses in a multi-clause sentence. A sentence is fed into the processing network one word at at time to form the gestalt; once the gestalt has been formed, a query is fed into the four probe inputs and the response to that query appears on the network's output. The figure shows one possible query for sentence (4).

was provided: the *Decoding Network*'s task was to learn to "fill in the blank" for a single slot in a shallow semantic representation when the probes were loaded with numeric representations of the fillers appropriate for a specific clause, one of which was replaced by a query marker indicating the slot to be filled. I call such a probe set a *query*. For example, as shown in figure 2, the words of sentence (4) were fed into the *Processing Network* sequentially to form a gestalt representation of the sentence, after which probes based on the semantic representations of the sentence's two clauses, (5) and (6), must produce the appropriate response.

The query marker that replaces an individual role filler in a query is indicated in the figure and the examples below by a question mark in the position of the slot to be filled. For purposes of implementation it was treated as yet another semantic symbol, since it had to fit into the probe layers, which were designed to accept the representations of the ordinary semantic symbols. However, the query marker was never required to appear in the response output, and thus was not considered when determining which symbol an actual output most nearly resembled. The numeric representation for the query marker was given a value of all-zeros so that it would not contribute anything to the activation levels in the hidden layer of the *Decoding Network*.

Unlike the original model, the revised architecture was only expected to provide role fillers in response to such queries; it never provided slot names. As an example of a query, recall that the two clauses of sentence (13) should be associated with the semantic fillers (MONKEY SEE PARROT —) and (PIRATE CHASE MONKEY —). Thus when probed with either (? SEE PARROT —) or (PIRATE CHASE ? —), the response should be MONKEY.

Notice that with four semantic roles per clause, each clause allows for four distinct query-response pairs, each of which must potentially appear in a training set for full coverage of the sentence. Thus the corpus provided a pool of 2392 training examples for the 598 clauses in the 310 sentences.

Units in the input, probe, and response layers were designed to hold values in the range [0.0, 1.0], and units in hidden layers held values in the range [-1.0, 1.0] for ease of learning (Haykin 1994). The logistic function was used to squash activations at the response output, and the hyperbolic tangent function was used to squash activations in the hidden layers. Recurrency was managed by means of a simple copy-back, as in a simple recurrent network (Elman 1990); all other connections were "full" connections, designed to be trained from initially random weights by backpropagation.

The input layer of the *Processing Network* accepted a sequence of word representations encoding the plaintext of a sentence.[4] Simple orthogonal representations were used for the words, with one unit in each word set to 1.0 and the rest to 0.0. Similarly, the probe and response layers of the *Decoding Network* accepted and yielded simple orthogonally coded representations for the semantic symbols, so that the output of the trained network could be interpreted as confidence values (Renals and Rohwer 1990).

The sizes of the input, probe, and response layers were thus dictated by the size of the lexicon. As the vocabulary never exceeded 17 lexical words and 15 semantic symbols, the largest input layer used in the experiments was of 17 units and the largest probe and response layers were of 15 units.[5]

After exploratory experiments, the size of the gestalt layer was fixed at 15 units and the hidden layer in the *Decoding Network* was fixed to 150 units. The recurrent layer in the *Processing Network* was also set to 150 units for the grammar-processing experiments described in section 5, and it worked well enough in the task that no tweaks to its size have yet been tested.

When a network with this architecture has been trained successfully it is possible to obtain well-formed *gestalt* semantic representations of individual sentences by cycling the words of the plaintext through the *Processing Network* and extracting the activations from the gestalt layer after the end of the sentence. To facilitate the generation of sentence gestalts, and to allow the experiments to focus on the capabilities of the *Processing* and *Decoding* networks in isolation, the logical distinction between the two sub-networks was reified by providing routines allowing activation values to be loaded into, extracted from the gestalt layer as needed, and allowing the two sub-networks to be trained and operated either independently or in conjunction.

4 Experiment I : Distributed Semantic Representations

The first set of experiments concentrated on the properties of representations that could be induced in the gestalt layer by training the *Decoding Network* in isolation. These experiments were aimed at finding out whether the desired gestalt representations could be formed for multi-clause sentences and whether the *Decoding Network* could be trained to extract information from them selectively, decoupling this process from the task of parsing the word sequences.

Gestalt representations for sentences in the corpus were created and refined as follows. The plaintext of each sentence was fed once through the untrained *Processing Network*, without backpropagation, forming what amounted to a wild guess at a representation for the sentence in the gestalt layer. Each such representation was saved (and the recurrent layer cleared) before proceeding to the next sentence. Thereafter the *Processing Network* was ignored and work continued using the *Decoding Network* alone. Using the saved guesses as initial values for the gestalt sentence representations, the decoding network was trained in isolation and the FGREP mechanism was used to coerce them toward more felicitous values (Miikkulainen and Dyer 1989, 1991). Since the inputs (the gestalt representations) were modified during this training process there was no obvious way to test for generalization to unseen inputs, so the entire corpus was used for both training and testing.

For this procedure, η was a constant 0.1, α was a constant 0.9, and the FGREP learning rate was $0.1 \times \eta$. The network was trained for 500 epochs and the experiment was repeated with four different random seeds. The seed controlled both generation of the initial random weights for the network and sortition in the order of presentation of the training examples. The order of presentation was scrambled *by query* rather than *by sentence*, so that numerous queries pertaining to a single sentence would not appear successively within an epoch. When tested on the gestalt representations resulting from the FGREPping, for one seed the trained network gave an incorrect response for a

[4] The inputs to St. John and McClelland's original network were encodings for phrase-sized chunks of sentences rather than encodings for individual words, considerably reducing the required number of passes through the recurrent layer when processing a sentence. Since the enhanced architecture used individual words for its inputs, it required as many as 13 passes through the recurrent layer for the longer sentences.

[5] *15 units:* 14 for the symbols associated with the words in table 1, and 1 more for the null symbol indicated by the dash. Those words serving a strictly grammatical function, *that, the,* and *to*, did not have associated semantic symbols as they were not directly associated with specific fillers in the case-role structures. The query marker, being represented by all-zeros, did not participate in the orthogonal representation of the symbols, and so did not require an additional unit for its representation.

single one of the 2392 queries; for the other three seeds there were no errors in the trained networks.[6] Reasonably high confidence values were obtained regularly as well, namely output values ≥ 0.8 for the correct answer and ≤ 0.2 for all other possible answers. Few exceptions were found; indeed, the most common results were bounded by the stricter values 0.85 and 0.15.

The utility of the representations created in these experiments was tested by examining the similarity of independently created representations for similar content. If the representations generated by this procedure are to serve as gestalt semantic representations for sentences it is essential that sentences bearing the same content be associated with very similar representations, without regard to word-order variations in the the plaintext. The similarity properties of the representations generated by the experiment were tested through cluster analysis, yielding a similarity tree for the 310 sentences in the corpus. Though the resulting tree of similarity relationships does not capture all the possible dimensions of similarity between the sentences in the training data, it sometimes reveals striking successes. For instance, the closest pair of representations in the cluster tree corresponded to sentences (14) and (15),

(14) the pirate saw the monkey that caught the parrot

(15) the monkey that the pirate saw caught the parrot

Since these representations were associated with different sentences, they were created, stored, and refined independently by the training algorithm. However, their clause-level case-role fillers are identical, namely those in the set of shallow semantic representations shown in (16) and (17). Under the simplifying assumptions of the model, the close similarity of the representations for the two sentences is precisely the correct behavior.

(16) PIRATE SEE MONKEY —

(17) MONKEY CATCH PARROT —

Other minimal clusters showed additional sentences of varying degrees of similarity paired off in the same fashion. The results, though qualitative, seem to show that similar sentences result in similar representations, suggesting that the gestalts can indeed be useful as targets for sentence processing. The experiment in section 5 proved to be a more demanding test of the utility of the representations.

5 Experiment II : The Virtual Parse

A different sort of experiment tested the ability of the *Processing Network* to learn to transduce the plaintext of a sentence into gestalt representations such as those created by the experiments described in section 4, and to generalize that ability to previously unseen examples. This experiment was conducted as follows.

A subset of the sentences in the corpus was randomly selected to serve as a training set, and a disjoint subset of similar size was randomly selected to serve as a validation set. The remaining sentences were reserved as a test set. In a first phase of training, those queries pertaining to the sentences of the training set (only) were used to train the *Decoding Network* by a modification of the procedure described in section 4, as described below, and after training was complete the resulting gestalt representations for those sentences were saved for use as targets for training the *Processing Network* independently in a second phase.

The training in the first phase was done according to the procedure described in section 4, except that the *Decoding Network* was tested on the training examples every 10^{th} training epoch, and training was stopped when all queries were answered correctly or when an arbitrary ceiling of 5000 epochs had been reached, whichever occurred first. (As before, since the FGREP mechanism modified the input representations as training progressed, it did not make sense to use the unmodified representations of the validation set to control training for this phase of the experiment.) A snapshot of the weights and FGREPped gestalt representations was saved after any test that showed an improvement in the number

[6]When counting the *number of correct responses* here and below, the network's *response* was identified as the semantic symbol having the representation closest in Euclidean distance to the actual activations in the RESPONSE layer.

Of interest — and possibly of substantial importance — is the fact that applying an identical training regimen to a network where the initial values of the gestalt representations were generated randomly (rather than by passing the plaintext through the untrained *Processing Network*) resulted in an error rate well over two orders of magnitude higher than that described above. For example, training for 500 epochs resulted in an average of 87.7 errors across three different random seeds (*vs.* an average of 0.25 for the method described in the text), and training for 1000 epochs still resulted in an average of 113.0 under the same circumstances. It is tempting to conclude that the surface forms of the sentences encode the semantic information in robust enough a form that some of the information structure is preserved even after "filtering" by the untrained *Processing Network*.

of queries answered correctly. In cases where the ceiling was reached, the snapshot was used to restore the weights and gestalt representations to the state providing the best performance; otherwise the final weights and representations were retained. The same learning parameters were used as before.

For each training epoch in the second phase of training, each sentence in the training set had its newly formed gestalt representation fixed as a target in the gestalt layer while the representations of the words in the plaintext were propagated sequentially through the recurrent *Processing Network,* without backpropagation until the end of the sentence had been reached. At the end of the sentence the *Processing Network* (only) was trained *once* by backpropagation and the recurrent layer was cleared before proceeding to the next sentence. This process was interrupted at regular intervals for testing *throughput* on the sentences in the *validation set* — that is, for each sentence in the validation set the representations of the words in the plaintext were propagated sequentially through the recurrent *Processing Network* without training it, the resulting activations in the gestalt layer were held fixed while all queries pertaining to the current sentence were tried sequentially in the *Decoding Network* without training, and the number of incorrect responses was accumulated. Training continued until such a test on the validation set resulted in more errors than had the previous test, at which point it was terminated to avoid overtraining. A final test was then run on the previously unseen *test set*, and performance statistics were collected.

This procedure for training the *Processing Network* was done using a constant η of 0.001 and an α of 0.1. Validation tests were made every 5000^{th} epoch, with an absolute limit of 50,000 training epochs. The entire experiment was repeated with training and validation sets each consisting of 1, 2, 5, 10, 15, 20, and 25 percent of the sentences in the corpus, with the remaining sentences reserved for testing, and each such variant was repeated five times with different random seeds. The performances on throughput with the test set are given in figure 3.[7]

6 Discussion

The results of the experiments were generally encouraging. Even so, a few caveats are in order with respect to scalability. For instance, continued use of orthogonal representations for the lexical words and semantic symbols cannot be sustained on the scale of a realistic lexicon, and even for this toy corpus the size of a hidden layer was allowed to grow relative to the size and number of the probe inputs holding the symbol representations (as described in section 3 above). Allowing such growth in the layers on both sides of a full connection generates more than linear growth in a network's training time, and cannot be sustained to arbitrary sizes.

Increments to the complexity of the training data had a similar effect. For example, when the first few sentences with ditransitive verbs were first added to the corpus, a fourth filler had to be added to *all* existing clause structures so that all training examples would match the architecture with the fourth probe layer added, even though it was the null filler that was required for all these existing clauses. In addition to the increment of training time resulting from the addition of new connections to the network, a new query had to be generated for each clause so that the new fillers for the fourth slot could be probed for. This caused the number of training examples to grow by $\frac{1}{3}$, even before reckoning in those for the new sentences. Thus it may be that it was the growth in the network architecture and in the number of training examples that allowed the model to maintain its performance when the new complexity demand was placed on it. Some of this growth can be managed in future experiments by limiting training to a randomly selected subset of a large corpus implicit in a toy grammar, so that the range of lexical and structural possibilities is well represented without the huge corpus needed for exhaustive coverage.

For the parsing experiment reported in section 5, however, overall training time increased at a rate less than linear with the increase in size of the training set. It may be possible to attribute this to the regularities in the structure of language, if unlike examples reinforce each other rather than counteracting each other during training. Whatever the explanation, the trend is contrary to ordinary expectations while training a network, and may bode well for longer-term scalability.

Developing and working with the corpus also revealed problems that have not yet been addressed to any significant extent by the model. One is the matter of identifying the content of a representation without prior knowledge of what

[7] To be more specific, the percentages listed were actually the *probabilities* that a given sentence would be selected for the training set (or the validation set). The actual sizes of these sets thus varied somewhat around the nominal size given by the percentages. Since a random seed governed this selection process, and varying the seed was used to ensure that different sets of sentences were selected for each of the five runs with the same probability parameter, there was no assurance that the training sets (nor the validation sets) were disjoint between the runs with different seeds. Care was taken, however, to ensure that the training, validation, and test sets were disjoint *within a given run*.

N.B. — The averages reported in the table were not weighted according to the *actual* number of sentences in the test sets; they are merely a simple average of the error rates of the five instances of each *notional* set size.

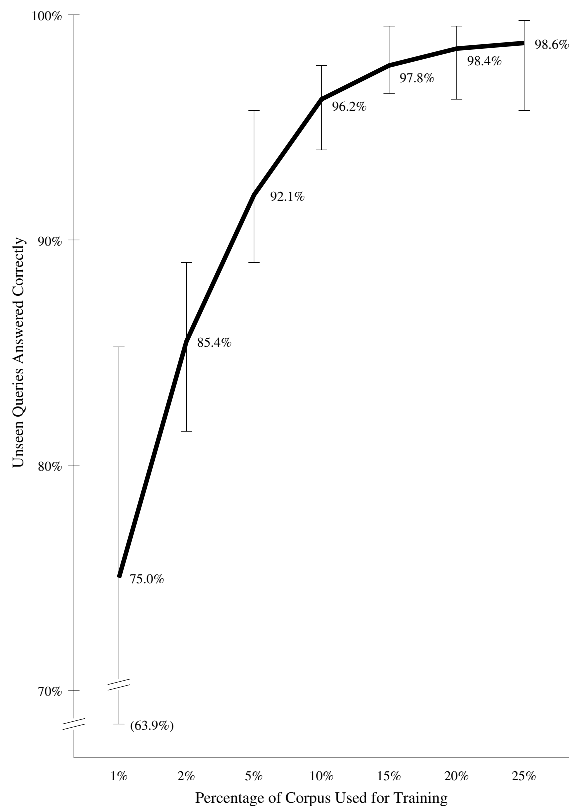

Figure 3: **Degree of generalization on throughput testing.** The results shown are the average success rates for five independently randomized runs on each of the indicated training set sizes. Training on a relatively small subset of the corpus allowed the system to learn to perform the semantic parse with high reliability.

it is. It may be possible to modify the model so that information can be extracted by means of less complete or less specific queries, such as by working from the general to the particular, so that a representation can be identified in a relatively few steps.

The quality of generalization on the throughput experiment was a pleasant surprise; there was no expectation that such good performance would be obtained by such a simple processor and training regimen. No scalability experiments were done for the grammar processor, and it remains to be seen how well longer sentences and deeper embeddings may be learned, but since the model does not aspire to managing unbounded embeddings there seems to be reasonable hope that it can be trained to succeed on sentences of a complexity similar to those regularly managed by humans.

It may in fact be possible to model some aspects of the traditional performance/competence distinction directly by means of this or a similar architecture. Performance limitations on a network are a familiar phenomenon. In the current case it may be expected that excessively long sentences would cause a compounding of errors due to repeated passes through the recurrent layer until the activation patterns are scrambled beyond interpretation, or perhaps a sufficiently complex sentence would saturate the ability of the gestalt layer to maintain superimposed clause-level patterns. Detecting competence in the network will be a more subtle challenge. It is the very nature of simulated neural networks to learn associations among the elements in their training sets, and it may reasonably be expected that this network "noticed" that whenever, say, RUN filled the ACT slot, the null symbol must necessarily fill both the PATIENT and BENEFICIARY slots. Other fillers for ACT will have their own sets of options and restrictions. If the network did learn these as expected, it has in some sense learned the *argument structure* of the verbs associated with the various possible fillers of the ACT slot. A carefully planned experiment may be able to extract this information directly from the weights in the trained network, though admittedly our knowledge-extraction capabilities are still quite limited. How the network might have acquired competence with respect to other patterns salient in the corpus, such as the facts that any noun can be modified by a relative clause and that the objects of a ditransitive verb may have their order reversed, depending on the presence or absence of the preposition *to*, is less easily visualized; but this is a promising area for further research, since observation of the trained network indicates that it did make use of these regularities when generalizing its performance to previously unseen sentences.

These observations bear on what is perhaps the most interesting property of the network, namely the overt display of discrete *switching behavior* in a static architecture operated by means of weighted connections. For example, successfully probing a given gestalt representation with different queries requires the network to perform the logical equivalent of extracting the information from a "different part" of the representation. Given the representation of sentence (18), which contains the information schematized in (19) and (20), a successful decoding network must be able to *switch* between the responses MONKEY, SEE, and PARROT when probed with (? SEE PARROT —), (MONKEY ? PARROT —), and (MONKEY SEE ? —), respectively. It must also, of course, be able to switch between the various elements of the sentence's second clause structure as well, and even be able to determine which clause to extract the answer from.

(18) the monkey that saw the parrot chased the captain

(19) MONKEY CHASE CAPTAIN —

(20) MONKEY SEE PARROT —

Such patterns may, of course, be learned by rote if present in the training set; but the ability to generalize the behavior to previously unseen examples in the throughput tests seems to indicate that the *Decoding Network* is enacting a state switch based not only on the position of the query marker, but also on the presence or absence of some signal in the gestalt layer.

More interesting yet is the behavior of the network under variations in word order. For otherwise identical sentences containing the variants shown in (9) and (10) or in (11) and (12), the *Processing Network* must propagate the information forward — regardless of which clause it occurs in — to the gestalt layer and deposit it there in a form that the *Decoding Network* can interpret as a signal requiring swapping of the fillers for the roles AGENT and PATIENT or PATIENT and BENEFICIARY when probing for those fillers on the variant sentences.

7 Conclusion

The representation experiments seem to indicate that this or some related model will make it possible to coerce a suitable static representation for the varying complexity of linguistic objects such as sentences. The model promises to scale well in terms of architectural elements and number of training epochs, though substantial pruning of the number of examples in large corpora reflecting complex grammars will still be needed in order to retain reasonable training times. The experiments raise questions about the robustness of the representations under ambiguity, and a substantial challenge remains with respect to the task of extracting information from a representation without *a priori* information regarding its content. However, under the controlled environment of the model the representations seem to be convenient and reliable enough to serve as targets for further experiments on sentence-processing tasks while the known problems are investigated further.

The sentence processing experiments were successful beyond expectation while using an extraordinarily simple architecture and training regimen. They offer hope that further experiments will bear out the hypotheses upon which the model was built, and offer the immediate challenges of learning to process a corpus of greater size and complexity while shaving points off the error rates. Meanwhile a better-instrumented version of the model and a better-controlled corpus will offer the opportunity to compare the model's behavior to observable human behavior and begin a process of empirical correction to the anatomy of the model.

References

Chafe, W. L. (1970). *Meaning and the Structure of Language*. Chicago: The University of Chicago Press.

Cook, W. A. (1989). *Case Grammar Theory*. Washington, DC: Georgetown University Press.

Elman, J. L. (1990). Finding structure in time. *Cognitive Science*, 14:179–211.

Haykin, S. (1994). *Neural Networks: A Comprehensive Foundation*. New York: Macmillan.

Miikkulainen, R. (1996). Subsymbolic case-role analysis of sentences with embedded clauses. *Cognitive Science*, 20:47–73.

Miikkulainen, R., and Dyer, M. G. (1989). Encoding input/output representations in connectionist cognitive systems. In Touretzky, D. S., Hinton, G. E., and Sejnowski, T. J., editors, *Proceedings of the 1988 Connectionist Models Summer School*, 347–356. San Francisco, CA: Morgan Kaufmann.

Miikkulainen, R., and Dyer, M. G. (1991). Natural language processing with modular neural networks and distributed lexicon. *Cognitive Science*, 15:343–399.

Pollack, J. B. (1988). Recursive auto-associative memory: Devising compositional distributed representations. In *Proceedings of the 10th Annual Conference of the Cognitive Science Society*, 33–39. Hillsdale, NJ: Erlbaum.

Renals, S., and Rohwer, R. (1990). A study of network dynamics. *Journal of Statistical Physics*, 58:825–848.

St. John, M. F. (1992). The story gestalt: A model of knowledge-intensive processes in text comprehension. *Cognitive Science*, 16:271–306.

St. John, M. F., and McClelland, J. L. (1990). Learning and applying contextual constraints in sentence comprehension. *Artificial Intelligence*, 46:217–258.