Dialogue Processing with Neural Networks

Ye-Yi Wang and Alex Waibel
24 June 1992
CMU-CS-92-165

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

This paper describes a novel connectionist system for dialogue processing. Based on a script-like formalism, the system consists of several modular neural networks which can track the semantic flow of a dialogue. The system can be extended to understand and translate dialogues in a fixed domain.

This research was partially supported by ATR Interpreting Telephony Research Laboratories, NEC and Siemens.

The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the ATR, NEC or Siemens.

1. Introduction

Connectionist models appear to be an attractive framework for natural language processing and high-level cognitive processing. Their ability to manipulate symbols through a time sequence makes it possible to process a language; their robustness on noisy inputs enables them to be reliable systems on ill-formed language inputs like those output from a speech recognition system; their ability to integrate multiply constrained sources of information provides an effective formalism for natural language processing involving the integration of a variety of sources of knowledge about languages, such as syntax, semantics and pragmatics.

Much recent work has applied connectionist models to natural language processing, for instance, language acquisition modeling [8], connectionist parser construction, [6, 7, 13] and semantic structure analysis [4, 5]. A modular connectionist architecture for paraphrasing script-based stories has been described in [10], which has combined different neural network modules for various tasks of parsing, semantic analysis and text generation together. It has several stages of different sub-tasks, and it uses explicit predefined representations for syntactic and semantic structures of sentences. The system can only process languages with such structural restrictions.

To process a loosely structured language like spoken language, we propose a more flexible model for dialogue processing. In our model there are no predefined syntactic or semantic structures. The model develops by itself a suitable representation of structure. This enables the model to cope with those languages having loose structure effectively. There are no separate syntactic and semantic processes either, which allows us to take full advantage of neural networks' ability to integrate multiply constrained sources of knowledge.

The basic model for dialogue understanding has two parts. One of them can track the semantic flow of a dialogue in a specific domain, for example, the conference registration conversations in our experiments. It can tell what the current topic of a dialogue is. It has a two-level architecture. The other part of the model is the Information Extractor, which can retrieve specific information from a dialogue. With some modifications to the basic structure of the model, the system can perform other natural language processing tasks, such as dialogue translation.

2. The Conference Registration Task

Currently our experiment of dialogue processing is limited to the domain of Conference Registration. Below is a sample conference registration conversation:

```
OFFICE: Conference office.

CALLER: I would like to take part in the conference.
        What should I do?
        Is an attendance fee required?

OFFICE: Yes. 200 dollars per person is required as a registration fee.

CALLER: I see. Thank you.

OFFICE: Do you already have a registration form?

CALLER: Not yet.
```

```
        Please send me a form.

OFFICE: Then, could you give me your name and address?

CALLER: The address is 12 Grant Street, Pittsburgh,
        Pennsylvania, 15133.
        The name is David Thompson.

OFFICE: Would you spell your last name for me?

CALLER: Sure, it’s T-H-O-M-P-S-O-N.

OFFICE: I see. Thanks.
        Goodbye.
```

Our goal in this work is to build a model which can process conference registration dialogues like the above one. The model should be learnable and robust on ill-formed inputs. In detail, the tasks of the model are:

1. **Semantic Flow Tracking**: For each input sentence in a context, the system should be able to determine its corresponding semantic category (also known as script slot, or topic). For example, in our Conference Registration task, the system should categorize the utterance "I want to register for the conference" as part of *Asking for Registration Form*, and "Is the registration fee required?" as part of *Asking about Registration Fee*. The difficulty for this task is that there are ambiguous sentences which may occur in several different categories. For instance, "I see" can fall into either *Asking for Registration Form* or *Asking about Registration Fee*. This makes the problem more than one of simple sentence classification. The system must use contextual information to determine which is the right semantic category.
2. **Specific Information Extraction**: Besides the semantic flow information, there is much more information which a system should be able to acquire from a dialogue in order to understand the dialogue adequately. For example, a system should be able to retrieve some specific information, such as the name of the caller and the amount of the application fee from a dialogue, and to ‘remember’ the knowledge for further use.
3. **Dialogue Translation**: This task is based on the previous two tasks. It integrates the knowledge about input dialogues obtained from the two tasks, (i.e., the category of each sentence in a dialogue from the semantic flow tracking and the various information contained in each sentence from the specific information extraction) to translate the sentence into a target language.

3. The Model

3.1. The Representation of Words and Sentences

Dialogue and language understanding are high-level cognitive processes. It is believed that this involves a hierarchy of understanding, corresponding to the hierarchy of the semantic components of a dialogue: at the top is the general topic of the whole dialogue, followed by the sub-topics of dialogue sections, followed by the meanings of sentences, and finally the word meanings. It is unlikely that a dialogue could be understood without an understanding of the meanings of its semantic components at each level of the hierarchy. When several components at a lower level of the semantic hierarchy form an entity in an upper level, the upper level entity contains not only the meaning of the components, but also the structural information, i.e., how these lower level components are integrated. If an understanding system is based on the word meaning only, it will miss the structural information of the semantic components at intermediate levels of the hierarchy. Therefore, a high-level language processing task like dialogue flow tracking should be based on the higher level information of linguistic structures and meanings, rather than the meanings of words. In our system, the dialogue flow tracking is based on the structures and meanings of sentences in a dialogue: the sentence representations.

On the other hand, meanings of words are also important for dialogue processing. According to Frege's Principle [1], the meaning of a sentence depends on the meanings of the phrases within it, which in turn depend on the meaning of the words within them. Without the information about each word in a sentence, it is impossible to obtain higher level linguistic information like the syntactic structure and the semantic meaning of a sentence, which is crucial for language understanding tasks like dialogue flow tracking.

Since both the information about words and the information about sentences are crucial for dialogue processing, they must be included in our model. Our system therefore involves the representations of meanings or structures for both words and sentences, which are represented as distributed patterns: words are represented as 26-bit syntactic and semantic feature patterns, while sentences as patterns with 30 bits.

Coding word patterns is relatively easier than coding sentence patterns, for we know what information is needed to distinguish one word from another. For example, a noun is different from a verb, and a proper noun differs from a common noun. On the other hand, it is difficult to know which feature is important to distinguish a sentence from another. The distinguishing semantic features are often not obvious.

To address this disparity in the coding problem, we use different methods to code the distributed patterns for words and for sentences. The word patterns are hand-coded (for about 430 words in the lexicon). Each bit in a pattern represents a particular syntactic or semantic feature. For example, a particular bit in a pattern is used to encode whether the word is a noun, and another bit might be used to indicate whether it is a person's name.

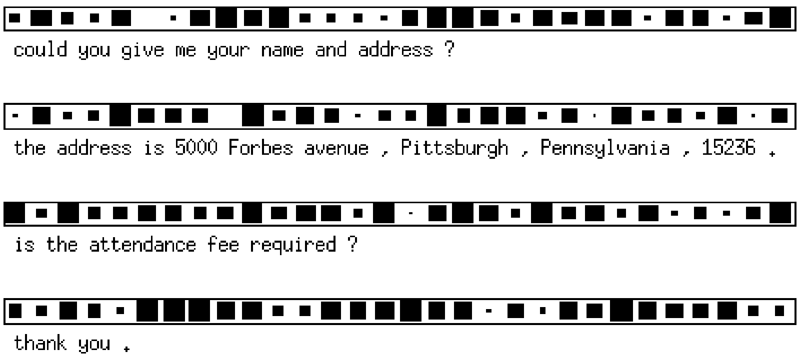

Figure 3-1: Initial Distributed Patterns of 4 Sentences

The sentence patterns, on the other hand, are developed by learning. Initially a 30-bit pattern is generated randomly for each sentence in the training set [Figure 3-1]. The patterns are modified automatically in the neural networks' training process in a way such that the learned patterns will help the networks to achieve better performance [9]. We do not yet know which specific features are included in the patterns. However, we believe that the patterns encode syntactic, semantic and probably pragmatic features to help the system to understand dialogues adequately.

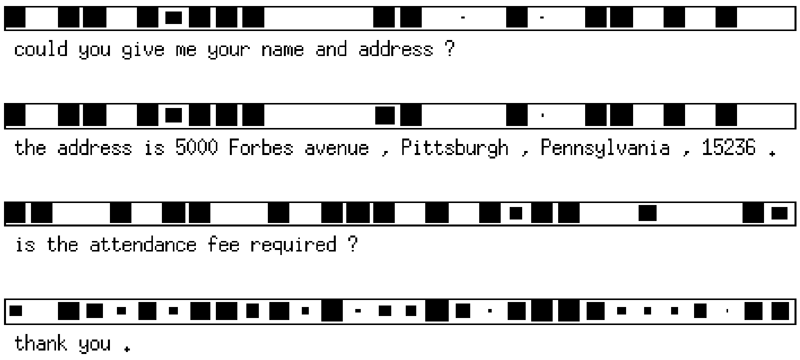

Figure 3-2: Final Distributed Patterns of 4 Sentences

For task 1 listed above, the representation learned has the property that sentences which occur typically in the same semantic categories have similar feature patterns, whereas sentences occurring typically in different semantic categories have very different feature patterns [Figure 3-2]. Since sentences with similar surface appearance often occur in same semantic categories, they usually get similar feature patterns. This property will help the system to achieve good generalization performance.

3.2. The Architecture for Dialogue Flow Tracking

The model consists of two components for dialogue flow tracking, namely the "Sentence Representation Generator" (SRG) and the "Script Analyst" (SA) [Figure 3-3]. Each component is a recurrent neural network, as proposed by J. Elman. [2, 3]

1. ***Sentence Representation Generator SRG:***

   As discussed above, high-level linguistic information is important in dialogue processing. This raises a problem: how can we obtain the information? It is impractical to assign a representation of structure and meaning to each sentence, for there can be infinite number of sentences.

   According to the principle of compositionality [1], the higher level meaning can be obtained from combining lower level linguistic information. Applying the principle here, we can expect that it is possible to obtain the representation of a sentence from information about the words in the sentence.

   The SRG is such a component producing the higher level information (the syntactic structure and semantic meaning of a sentence) from the lower level information (the syntactic and semantic features of the words contained in the sentence). Its input is a temporal sequence of feature patterns of the words in a sentence, and its output is the

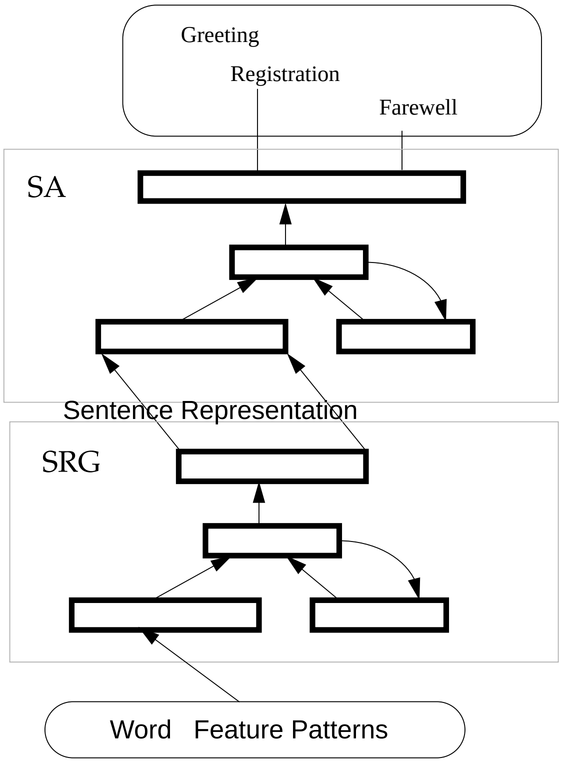

Figure 3-3: Architecture for Dialogue Semantic Flow Tracking

distributed pattern of the sentence.

The SRG works in the following way: at each time *t*, the pattern of the *t*-th word in a sentence is fed forward to the network as input, along with the activations of the hidden units at time *t-1* stored in the network's context units (these context units are set to zero at time 0)[1] After all the words in the sentence are fed to the network, the final output of the SRG is the distributed pattern of that sentence which encapsulates its syntactic and semantic features.

2. ***Script Analyst SA:***

The SA fulfills the task of tracking the semantic flow of dialogues. The inputs to the SA are the sentence representations obtained from the SRG, and the outputs from the SA are the local representations of the roles of the sentences (or semantic categories) within the dialogue (Greeting, Asking for Registration Form, etc., as in our domain of Conference Registration);

The SA works in a similar way to the SRG. At time *t*, it accepts the representation of the *t*-th sentence (obtained from the SRG) as input, along with the contextual information copied

[1] It is argued that such context units encode properly the contextual information about the potions of the sentences the network has ever seen. [11]

from the hidden layer. The input patterns are fed forward to the output layer. The network's output specifies the semantic category of the input sentence in the context with the corresponding output unit being active. However, there are some differences between the SRG and the SA: in the SA the output is a local representation of the category information about each input sentence; in the output layer, as only one unit is active, the *winner-take-all* strategy will inhibit all the other output units. As the topic of the dialogue changes, the ''state'' of the dialogue will change, and the corresponding output units will flip.

The two-level architecture (the SRG and the SA) makes the sentence representation transparent to users, while allowing dialogue flow tracking to be based on high-level semantic information, i.e., sentence representations rather than word feature patterns.

3.3. The Architecture for Information Extraction

As we mentioned above, in order to understand a dialogue, a system should be able to acquire much more knowledge than the semantic category of a sentence. It should be able to locate specific information like the caller's name in a dialogue. Another network, the Information Extractor, is introduced into the model for this purpose.

The Information Extractor (IE) is also a recurrent neural network [Figure 3-4]. It takes the feature patterns of the words in a sentence sequentially as inputs, and activates a specific unit in the output layer if the input word is the corresponding specific information. For example, if the input word is *John*, then the *FirstName* output unit will be activated in the output layer. The output here is a local representation similar to that of the SA, except possibly having no active output unit in the IE.

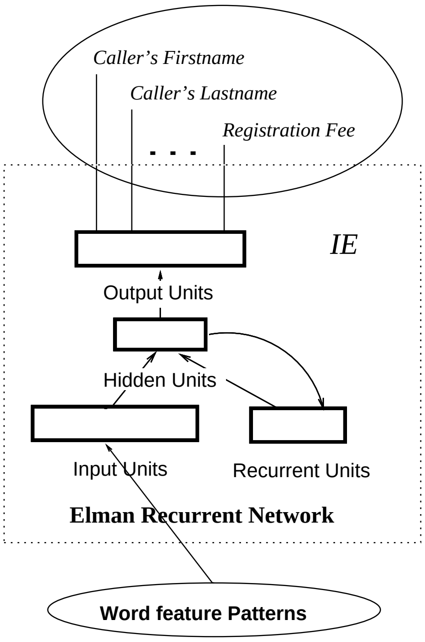

Figure 3-4: Architecture of the Information Extractor

4. The Training of the Networks

The system is trained in two stages: the training for the SA and then the training for the SRG thereafter.

The SA is trained first. Initially each sentence in the training set is represented by a random pattern. An extended back-propagation procedure [9] is then used to modify the weights of the SA *as well as* its input sentence representation. In a sample dialogue, the representation of a sentence is fed to the network as input, and all the context units are set to zero initially. A forward pass is performed and the mean square error at the output is calculated. The error is back-propagated, and the weights are modified with gradient descent. The error is, however, backpropagated one layer further than in the standard backpropagation algorithm -- to the virtual input layer [9] -- to modify the weights of the connections between the virtual input layer and the SA's input layer. In this way, the representation of the SA's input pattern is modified during the backward pass. This process is repeated for all sentences in the dialogue with the context units always being set to the activations of the hidden units in the previous step. The network is trained with all the sample dialogues for several epochs until the final output error of each dialogue is less than a certain threshold. The final representations of the sentences in the sample dialogues can then be used as the targets for the training of the SRG.

The training for the SRG could be performed in a similar way. At present, however, the input feature patterns of words are already hand-coded, so it is not necessary to train the word representations. We use standard backpropagation to train this network, with the sequence of word feature patterns presented at the input one at a time, and with the pattern of that sentence (developed from the training of the SA) as the target output.

Several measures have been taken during the training to improve the performance of the SRG.

4.1. Sample Balance

One problem which affects the performance of the SRG is that the sentences in sample file are not well balanced: the sentence length varies from one word to 21 words. In some cases, more than ten sentences may have the same or a similar representation, while some other sentences are represented by unique patterns of their own. This makes it hard for the network to generate the accurate sentence representation for all the sentences: the network is trained over more backpropagation epochs in generating the representation of those long sentences, whereas the network is inadequately trained to generate an accurate representation of the short sentences. The network is trained to generate the representation shared by many sentences much more easily than to generate an unique sentence pattern. This causes a performance imbalance for the SRG.

To solve this problem, we have normalized the learning rate for different sentences: for long sentences with representations shared by many other sentences, we use small learning rates, whereas for short sentences with representations shared by few sentences, we use large learning rates. The following formula is used to calculate the learning rate for sentence s_i with representation r_j:

$$\varepsilon_i = \frac{\varepsilon}{l_i \sqrt{n_j}}$$

Here ε is the overall learning rate, ε_i is the learning rate for sentence s_i, l_i is the the number of words in sentence s_i, and n_j is the number of sentences with representations r_j.

4.2. Noise and Recurrent Weight Decay

Because of the the limited amount of training data, the SRG may overfit to the initial words in a sentence. For example, in our conference registration task, when the SRG encounters the input words *I, would* and *like*, it will immediately posit that the sentence is *I would like to register for the conference*, and generate the representation of that sentence, since there is no other sentences in the sample file beginning with *I would like*. This causes the system to be excessively sensitive to the noise in the beginning of a sentence, and makes it hard to train the SRG incrementally for new sentences with the same initial words.

We have taken the following measures to overcome this problem:

- Introducing noise words in the sample sentences: Noise words are randomly chosen from the lexicon, and substituted at random for the actual words of the sentence. No noise words are added to short sentences with fewer than 4 words. The number of noise words is randomly chosen, with the maximum determined by the length of the sentence.
- Decaying weights from context units (recurrent units) to hidden units: In doing so, initial words in a sentence no longer dominate in their effect on the final representation of that sentence. Instead, the SRG generates the representation according to the relevance of the words in a sentence.

5. Results

The system was evaluated in a Conference Registration task. The dialogues of this task covered 5 semantic categories, namely ***Greeting***, ***Asking for Registration Form***, ***Requesting caller's information***, ***Information about Registration Fee*** and ***Bidding farewell***. There were about 60 different sentences in the training set for these dialogues. The lexicon consisted of 435 words.

We used recurrent neural networks for the SA and the SRG, as mentioned above. The SA network had 30 input units, that was the number of features of sentence patterns, 10 recurrent (context) units, 10 hidden units and 5 output units corresponding to the 5 semantic categories. The SRG network consisted of 26 input units (the number of features for word patterns), 16 hidden units, 16 recurrent units, and 30 output units.

The SA and the SRG were trained separately and then combined to track the semantic flow of dialogues. It took about 250 training epochs for the SA, and about 1200 epochs for the SRG. It is also possible, although it remains to be done, to train the two networks together as a big network for several epochs after each of them has been trained separately. This final tuning might be helpful in improving the performance of the system [12].

The system achieved 100% accuracy on the training set of dialogues and 95% accuracy on the testing set of 15 dialogues containing 246 sentences.

5.1. Ambiguous Sentences

It might seem that the system is simply a sentence classification network. For each input sentence, the system classifies it into a semantic category. However, the system achieves more than this, for there are many sentences that are ambiguous and which may occur in many different categories. For example, sentences like "*I understand*", "*Yes, that's right*" and "*Thank you very much*" can occur in almost every category. Our system can use the contextual information to eliminate the ambiguity. For example, if the sentences "*I understand*" or "*Thank you*" follows the sentence "*The application fee is 200 dollars per person*", the networks activate the *Information about application fee* dialogue output units. Instead, if the same sentence is preceded by "*My address is ...*", the networks activate the *Requesting caller's information* unit.

5.2. Nesting of Dialogue Topics

We have seen that contextual information can help to determine the current topic of a dialogue. However, context alone is not enough to predict the semantic category of the current input sentence. In our training and testing data, the dialogue structure is complicated. For example, the speaker may talk about the registration form first, and then shift the topic to the application fee, and then return to the application form topic. At any point in the dialogue, there may be several different possible successive topics. Our system can combine the meaning of the current sentence in the dialogue together with the context to choose the correct semantic category.

5.3. Generalization and Noise Tolerance

As described above, we use syntactic and semantic feature patterns to represent words, so similar words like "John" and "Bill" have the same feature pattern. This allows the network to gain good generalization performance, so that even if the system did not encounter the sentence "*My name is John Smith*", it can generate the same sentence representation as has the sample sentence "*My name is Bill Thompson*".

The system can deal with unfamiliar sentences properly. Because the SRG generates the representation of a sentence in view of the whole sentence pattern rather than a particular word, (recall that we add noise to the training set and decay the recurrent weights during training to prevent the network from generating the sentence representation only depending on several words in the sentence) it is not surprising that it can generate very similar (almost the same) patterns for sentences like "*I would like to register for the conference*" and "*I want to register for the conference*". Therefore even though the second sentence does not exist in the training samples, the system can categorize it correctly.

For the same reason, the system can tolerate noise in input sentences. If the input is "*I would like to register for the registration*" instead of "*I would like to register for the conference*", the SRG can still generate the representation of the first noisy sentence which is sufficiently close to the representation of the second sentence. With this representation, the SA can analyze the dialogue correctly.

5.4. Scalability

We tried to extend the system to deal with a bigger task: the Conference Registration task consisting of dialogues spanning 17 semantic categories. The dialogues included more than 200 different sentences. However, the result was not satisfiable: the error rate was more than 35%.

We studied the cause of this problem. The unscalability could be attributed to two facts about the system and the experiment:

1. Lack of Training Data. Dialogue flow tracking is a high level language processing task. Generally speaking, the higher the processing level is, the more the training data

(dialogues) is required. For example, for speech recognition, each word is an example, and there are hundreds of words in a conversation. For parsing, each sentence is an example, and there are around fifteen sentences as samples in each of the conversations in our task. For semantic flow tracking, each dialogue is just one example. In our experiments, we had only 12 dialogues available for training. Althoght we perturbed the dialogues to "generate" some new data, it was still far away from the requirement for network training.

2. Over-Simplified SRG Structure. Careful investigation revealed that most of the errors were caused by the SRG. It failed to generated accurate sentence representations. This was partially due to the over-simplified SRG structure. The SRG was simply a recurrent network which sequentially accepted words in a sentence as inputs. In doing so, it treated a sentence as a sequence of words without internal structures. Therefore it did not make the most of linguistic information in a sentence. A more complicated structure of SRG, for example, a structure with a component for syntactic structure analysis, might be helpful to improve the performance.

6. An MT Experiment

We have extended the model described above to a simple dialogue translation system. The translation system is ‘‘simple’’ because there is no actual target sentence generation module in the system. What the system does is to determine the meaning of input sentences, choose a corresponding target language sentence template, and merge the template with the information obtained from the Information Extractor to form the sentence in the target sentence.

The architecture of the Dialogue Translation System is illustrated by [Figure 6-1]

Since Dialogue Translation is a high-level language processing task, it should be based on a high-level of linguistic information. For this reason, the sentence representations obtained from the SRG serve as the input to the translation network.

On the assumption that the semantic flow of a dialogue (in other words, the context of a sentence) is helpful for the translation of that sentence, we embed part of the SA network into the translation network: the connections from the input and context units to the hidden units in the SA are copied to the translation network (the dashed connections in Figure 6-1) and kept frozen during the training of the translation network.

However, dialogue tracking is not sufficient for translation. The sentence representation for tracking semantic flow of dialogues provides insufficient information for dialogue translation. Because the sentence representation for dialogue flow tracking is developed automatically during the training pass of the SA, all the information it contains is only sufficient for script analysis. Two sentences which occur typically in the same dialogue category usually have very similar representations. If the translator is based on such representations, it is not surprising that it will confuse these two sentences.

To overcome this problem, we introduce another sentence representation as a complement of the previous one, hoping that it can be used to distinguish the different sentences in the same semantic category of a dialogue. This extra representation uses 20-bit distributed patterns to represent sentences. It in turn can be obtained from an SRG, and is fed to the translation network as input (The shadowed part in Figure 6-1).

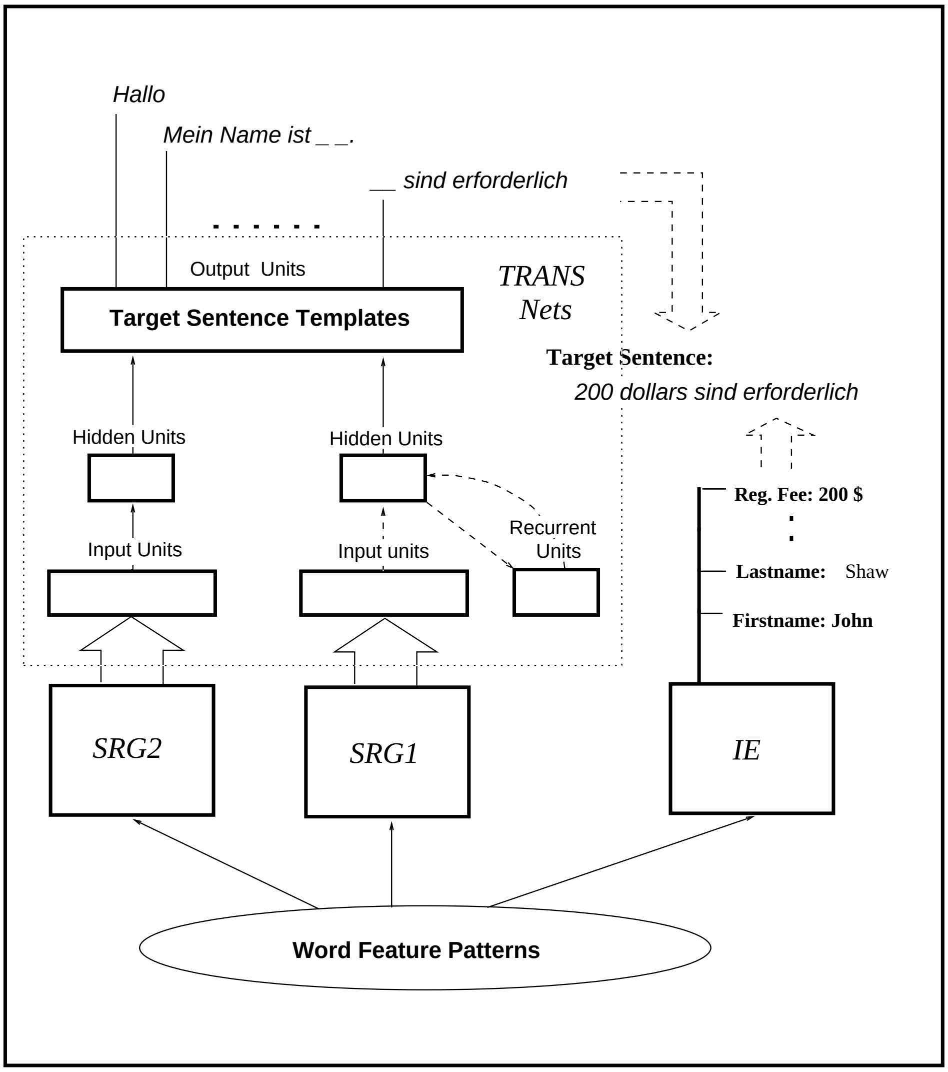

Figure 6-1: Architecture of the Translation Network

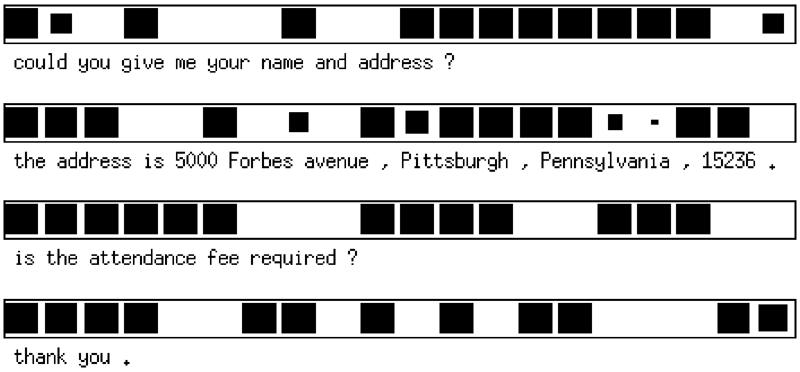

Figure 6-2: Final Distributed Patterns of 4 Sentences: Representation 2

The development of sentence representation 2 is similar to that of sentence representation 1: initially it is randomly generated for the sentences input to the translation network. During the training of the translation network, it is modified automatically with extended back propagation in a way such that the modified representation will improve the network's ability to distinguished different sentences. After the training of the translation network, the final representation [Figure 6-2] is used as the target for the training of the SRG2.

With the same test dialogues as those for dialogue state tracking, the networks have been able to translate the sentences with 98% accuracy in semantic sense. This accuracy is higher than that for dialogue tracking. This is due to the fact that the additional sentence representation helps to remove the dialogue tracking errors.

7. Discussion

The model used here departs from conventional models for language processing in that it has no separate syntactic and semantic passes. We note that syntax and semantics are integrated constraints in the cognitive task of language interpretation, and that there is some correlation between the two. Yet it is hard for symbolic AI to implement this correlation since we do not know in detail how this correlation works. One advantage of connectionist systems is that they can implement this kind of correlation implicitly by multiple constraint satisfaction. They can learn to use this correlation in language processing without specifying in detail the what the correlation is.

Another advantage of our connectionist language processing system is that it avoids explicit representation for syntactic and semantic structures. One problem that symbolic AI cannot avoid is semantic representations for sentences. Many different representations with varying advantages and drawbacks have been used for different systems, but none of them is perfect. The problem is that we still do not know what semantics is, since it is still a controversial point among philosophers and linguists. Our system uses a neat distributed representation for meaning of sentences, which is learned by the system itself. In doing so, we avoid the explicit representation of sentences. We can say we have the best representation for sentences if we agree that the best representation is what can allow the system to achieve the best performance in a certain task, since the learning algorithm guarantees that the representation it developed is the most suitable one for a specific task.

There might seem to be a problem with such representations of sentences. In different tasks, the feature patterns of the same sentence are different. For example, we need an extra sentence representation in translation networks. However symbolic AI has the same problem: for some tasks, logic representation is adequate, but for other tasks, semantic network representation may be more suitable. The representations in our system at least use the same framework (Distributed Patterns). Moreover, it might be reasonable for a system to have different semantic representations for different tasks, since it is hard to believe that humans have a unique complicated semantic representation suitable for various problems in mind and works in such an inefficient that uses the complicated representation for a very simple task. We have yet to find the support for this point in linguistic and psychological theories.

8. Conclusion and Future Work

The connectionist model for dialogue processing described here can fulfill the tasks of semantic flow tracking and simple dialogue translation efficiently in the domain of conference registration. In the model, words and sentences are represented as distributed feature patterns. Modular network architecture is used, with each module performing a sub-task of language understanding.

In contrast with other models for language processing, this model has no separate syntactic and semantic passes. Instead, it treats syntax and semantics as integrated constraints in cognitive language processing. In doing so, it avoids an explicit representation of syntactic and semantic structures. This is one of the major advantages of this model, for spoken language is loosely structured, and it is difficult for a multistaged (syntactic/semantic) model to deal with such variability. This advantage might be particularly useful in dealing with the variability and noise typical in ill-formed automatically recognized spoken language.

Our model is also learnable. The system can not only learn to perform a specific task like semantic flow tracking, but can also learn the sentence representations used for the task.

It would be desirable to develop a new robust translation network architecture. The networks should be able to deal with more flexible dialogue structures and have the mechanism to learn adaptively to translate new sentences in dialogues. The output of the translation network -- the local representation of sentence templates -- should be replaced by a distributed representation of dialogue and sentence features, and a text generation network should be involved in generating the target sentences instead of simply choosing target sentence templates.

References

1. David R. Dowty and Robert E. Wall. *Introduction to Montague Semantics.* D. Reidel Publishing Co., 1981.

2. J. L. Elman. Finding Structure in Time. Tech. Rept. 8801, Center for Research in Language, University of California, San Diego, 1988.

3. J. L. Elman. Representation and Structure in Connectionist Models. Tech. Rept. 8903, Center for Research in Language, University of California, San Diego, 1989.

4. A. L. Gorin and S. E. Levinson. Adaptive Acquisition of Language. Speech Research Department, AT&T Bell Laboratories, Murray Hill, 1989.

5. A. L. Gorin and S.E. Levinson and A. N. Gertner. Adaptive Acquisition of Spoken Language. Proceedings of the 1991 International Conference on Acoustics, Speech, and Signal Processing, 1991.

6. A. N. Jain and A. H. Waibel. A Connectionist Parser Aimed at Spoken Language. Proceedings of the International Workshop on Parsing Technologies, Carnegie Mellon University, Pittsburgh, PA, August, 1989.

7. J. L. McClelland and A. H. Kawamoto. Mechanisms of Sentence Processing: Assigning Roles to Constituents. In *Parallel Distributed Processing*, J. L. McClelland and D. E. Rumelhart, Ed., The MIT Press, 1986.

8. J. L. McClelland and A. H. Kawamoto. Mechanisms of Sentence Processing: Assigning Roles to Constituents. In *Parallel Distributed Processing*, J. L. McClelland and D. E. Rumelhart, Ed., The MIT Press, 1986.

9. R. Miikkulainen and M. G. Dyer. Encoding Input/Output Representations in Connectionist Cognitive Systems. Proceedings of the 1988 Connectionist Models Summer School, 1989.

10. R. Miikkulainen and M. G. Dyer. A Modular neural network architecture for sequential paraphrasing of script-based stories. Proceedings of the International Joint Conference on Neural Networks, 1989.

11. David Servan-Schreiber, Axel Cleeremans and James L. McClelland. Encoding Sequential Structure in Simple Recurrent Networks. Tech. Rept. CMU-CS-88-183, School of Computer Science, Carnegie Mellon University, Pittsburgh, 1988.

12. A. H. Waibel, H. Sawai, K. Shikano. Modularity and Scaling in Large Phonemic Neural Networks. Tech. Rept. TR-I-0034, ATR Interpreting Telephony Research Laboratories, 1988.

13. D. Waltz and J. Pollack. "Massively Parallel Parsing: A Strongly Interactive Model of Natural Language Interpretation". *Cognitive Science 9* (1985).

Table of Contents

List of Figures