A Natural Language Processor with Neural Networks

Wilfred M. Martínez
José A. Borges, Nestor J. Rodríguez, Shawn Hunt
Human-Computer Interaction Laboratory
Electrical and Computer Engineering Department
University of Puerto Rico, Mayagüez Campus
Mayagüez, Puerto Rico 00681-5000
martinez@rmece01.upr.clu.edu

ABSTRACT

A **N**atural **L**anguage **P**rocessor (**NLP**) for a multimodal, multilingual computer interface is presented. This NLP or **NA**tural **L**anguage **PRO**cessor with **N**eural **N**etworks (**NALPRONN**) demonstrates that artificial neural networks can be used in such systems. NALPRONN implements a generalized NLP system that emphasizes the modularity, adaptability, and flexibility properties that are used in a number of computer applications. It processes information for different application language subsets and redirects information to a specific active application. NALPRONN has processing modules which are backpropagation networks, and memory modules which are self-organized feature map networks. Additional networks can be added for further processing tasks. The processing modules are the input subsystem and the output subsystem, and the memory modules are the lexicon subsystem and the monitor subsystem.

1. INTRODUCTION

A great deal of research is currently being done to develop natural forms of interaction with machines. This requires knowledge of how human beings communicate in order to apply it to man-machine communication. For this purpose, studies and results from artificial intelligence, linguistics, philosophy, and psychology have been grouped into what is called cognitive science [1]. From its postulates, computer programs can be created to model the human communication process in an attempt to formalize what is required to achieve a *natural* mean of communication. These programs may be called NLP's.

NLP's are generally studied with computer programs that are applied to a variety of functions [2]. The applications for computer-based natural language understanding systems are: discourse, information access, information acquisition or transformation, interaction with intelligent programs, interaction with machines, and language generation [1]. A NLP suitable for a variety of applications is desirable since it would appeal to a wide range of the user population. This makes a NLP system attractive.

Additional features that make a NLP system attractive are: the modularity of a system for integration as a specialized part of a bigger system, the adaptability to different or new applications, and the flexibility to be used in distinct environments with different vocabularies. All these features, if taken into account, contribute to the development of efficient and effective NLP's that will diversify computers' usage and enhance interactive computing.

It is possible to construct a NLP system using artificial neural network technology. Neural network based systems for natural language communication have been very recently taken as the technology of choice for studies concerning the area of cognitive science. One reason for this is that they are subsymbolic systems where the processing tasks as well as the data are distributed over the whole. This subsymbolic approach to natural language processing possesses such properties as learning from examples, context sensitivity, generalization, robustness of behavior, and intuitive reasoning [3].

2. THE NALPRONN SYSTEM

Distributed processing and distributed data are central to the NALPRONN system. The system performs its processing task by means of submodules or subtasks. Each submodule independently but simultaneously learns how to process data items and extract processing knowledge automatically from examples. Data items are distributively represented. In such data, the underlying properties of the words represented are distributed over the whole data type used and not given as separate items [4]. These representations reflect the meaning of the words from which they were derived. They generalize knowledge over the whole data, that is, similarities between classes are made specific.

The NALPRONN system is based on the architectural theories proposed by R. Miikkulainen in [3]. Here Miikkulainen shows the possibility of creating large-scale NLP's using artificial neural networks. Instead of being the script processor developed by Miikkulainen, the NALPRONN system attempts the implementation of a generalized NLP. This NLP processes vector data items in a parallel distributed fashion.

NALPRONN is designed for integration in a system with a front-end consisting of a speech recognizer and a back-end of different applications to be managed. The adaptability to new applications is of key importance since this will increase its usability. There are, of course, constraints. A system connected to NALPRONN must know the NALPRONN protocol of communication for input and output, it must know where and when the information is available, and each memory module for each application in

0-7803-2559-1/95 $4.00 © 1995 IEEE

NALPRONN must contain the necessary communication knowledge.

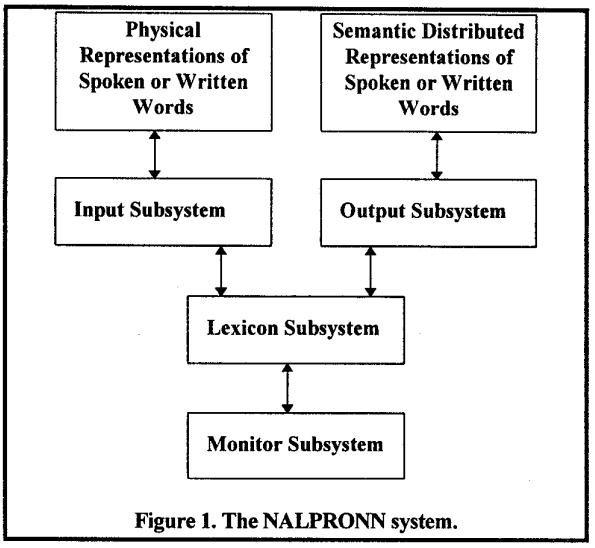

Figure 1. The NALPRONN system.

Memory modules learn to represent the meaning of words, phrases or sentences for each application. The representations are created based on examples of how the words are used in each application language subset. All subsystems in NALPRONN learn how to use the representations under similar conditions.

The NALPRONN system consists of a number of specific task modules made out of networks of artificial neurons. Theses modules are: the input and output representations, the input subsystem, the output subsystem, the lexicon subsystem, and the monitor subsystem (see figure 1). The system accepts coded representations of already processed spoken and/or written application language subsets, rearranges the data for correct management, translates the data for internal processing, and stores and monitors the data. The system decides which application universe is active and the correct representation for the selected application. The tasks are performed by different modules which are trained independently with data that is used by all of them.

Representations

The NALPRONN system manages distributed representations of information data. Their creation takes advantage of the knowledge contained in a specific data type, as they are developed from the content, usage, and/or meaning of application words. The meaning of words is extracted from the different contexts in which they are used.

The lexical symbols and the semantic concepts are implemented with one-dimentional vectors (see figure 2). These vectors contain normalized numerical values obtained from the physical or semantic word codification process.

The physical words are changed into filtered physical words and from there into lexical representations. All data is changed into a numerical code. Before the filtering process begins, the words are changed into ASCII format. The data is then normalized and fed into the networks of the system.

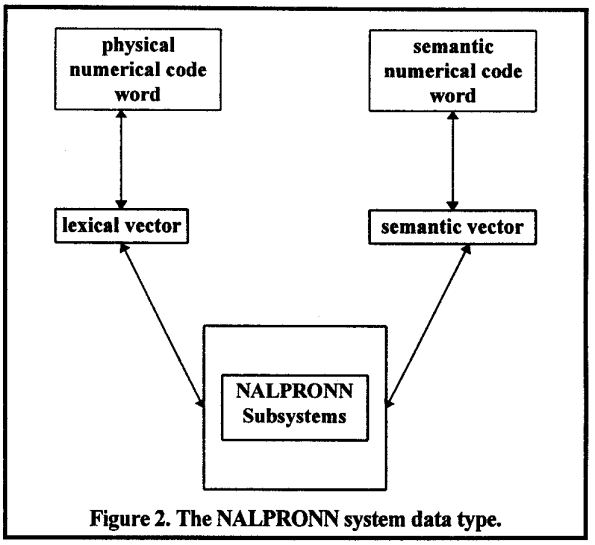

Figure 2. The NALPRONN system data type.

The semantic vectors contain the semantic numerical code words that represent concepts taken from the usage of physical words. These representations are the pieces of data that contain the knowledge of the system. A network of artificial neurons creates the semantic representations from the usage of the words using the backpropagation learning algorithm. The distributed semantic representation of the concept words provides the internal working process data type.

The semantic representations are obtained using a three-layered backpropagation network with an external lexicon from which training patterns are formed during training. The training environment in which training patterns change continuously is used to train the network in a supervised training process.

The Lexicon Subsystem

The lexicon in the NALPRONN system contains the words as symbols and conceptual representations of an application model. For the system to be adaptable and flexible, a lexicon subsystem is developed for each application. If a new application is needed, new memory modules are added to the subsystem.

The **LEX**icon (**LEX**) subsystem is an expert on an application universe. It contains all the knowledge for the efficient mapping between a physical symbol and a conceptual symbol. Its function is to perform a transformation between the physical representation of the words and the conceptual or semantic representation of them.

In the LEX subsystem, memory is represented as feature maps that are associatively connected to provide a means for a transformation of data. Distributed representations of the physical

and concept entities are mapped through such associations so that the transformation between lexical and semantic representations can take place.

Human semantic long-term memory is simulated in the LEX subsystem by associative structural connections of the concepts and skills that it has acquired. Memories are represented as feature maps. These *competitive networks* implement the physical symbol memory and the semantic memory as two-dimentional layers of neurons. A learning process creates associative connections between a physical symbol map and a semantic map (see figure 3).

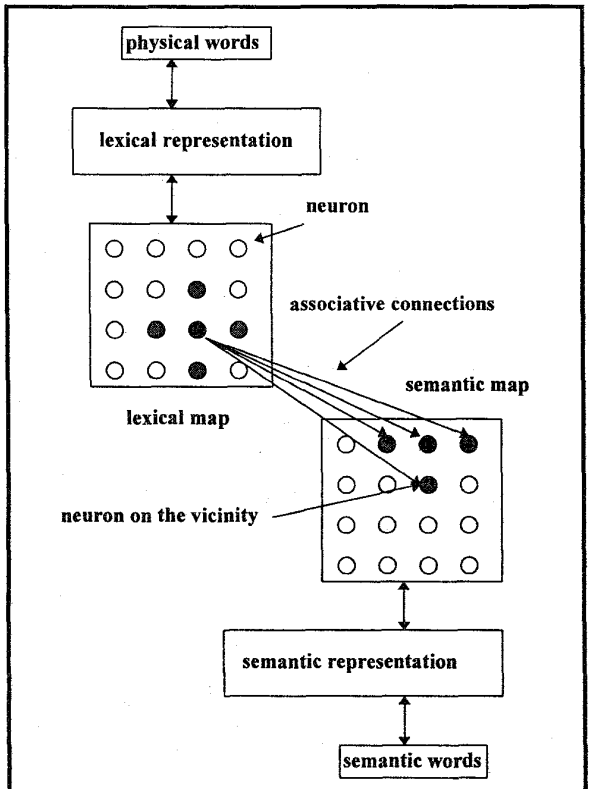

Figure 3. LEX subsystem data transformation.

The transformation process or mapping is many-to-many, that is, the associations implement the connections between words and concepts including words with multiple meanings and meanings with different words [5]. The lexical map is associatively connected to the semantic map, and the semantic map is associatively connected to the lexical map. The associative connections exist from each unit in one map to each unit in the other and vice versa. The measure of connectivity comes from the weight for each connection which is stored as a vector for each unit.

First, a self-organizing process is used to form the topological layout of the input data into the feature maps. The self-organized artificial neural networks learn to detect regularities and correlations in each input, adapting future responses to that input. Finally, the associative connections between the maps are formed with an unsupervised learning process.

The Input Subsystem

The processing or pre-processing of data depends on whether the system is linguistic or connectionist. In a linguistic approach, a parser decomposes data depending on strict syntactic rules or rules of grammar. In the connectionist approach used here, lexical disambiguation is used together with the distribution of the properties of the input to form a representation of the data. For the connectionist approach an input filter may be used.

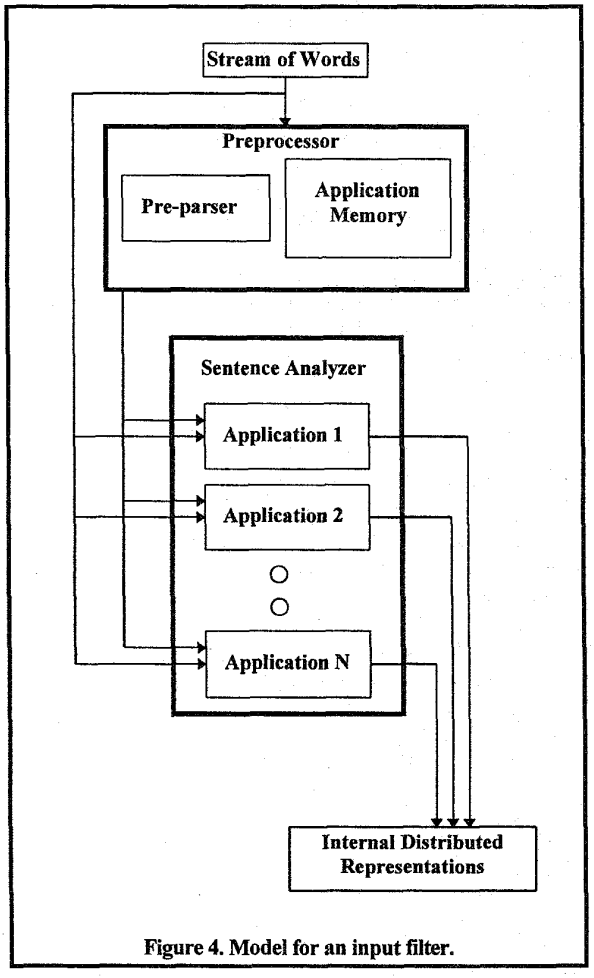

Figure 4. Model for an input filter.

Figure 4 shows the input filter which is composed of a preprocessor and a sentence analyzer. The preprocessor contains two elements: a pre-parser and an application memory. The preprocessor's function is to redirect the streams of input concepts to their correct applications. If the user issues commands in the form of a single word, the pre-parser can be eliminated and the application memory alone can be used for redirection purposes.

The sentence analyzer groups the sentence analyzers for each application. Their function is to transform the issued commands into distributed representations to be used by the output subsystem of each application.

From the previous model, a connectionist class data manager was developed and implemented as an application memory with feature maps, and as a sentence analyzer with recurrent backpropagation networks. This data manager acts as an application activator and as a data input filter.

This **IN**put **FIL**ter (**INFIL**) performs the necessary transformations for correct data management and internal/external representation. A sequence or stream of word's representations is transformed into a stream of concept's representations for a specific active application.

The tasks of application filtering and data management are performed by a preprocessor and a sentence analyzer respectively. The preprocessor decides which application is active at working time, and the sentence analyzer produces the distributed representations used by the active application.

The INFIL preprocessor contains an application memory implemented as a single feature map. The sentence analyzer is implemented as a group of independent recurrent backpropagation artificial neural networks.

The Output Subsystem

In NLP design, once an input sentence is analyzed and separated into its constituents, it must be rearranged into a format that the receiver of the information understands. From the processing of language, knowledge is obtained, that is, a meaning is attached to each constituent of the input sentence. It is this meaning that is issued as critical information to an application.

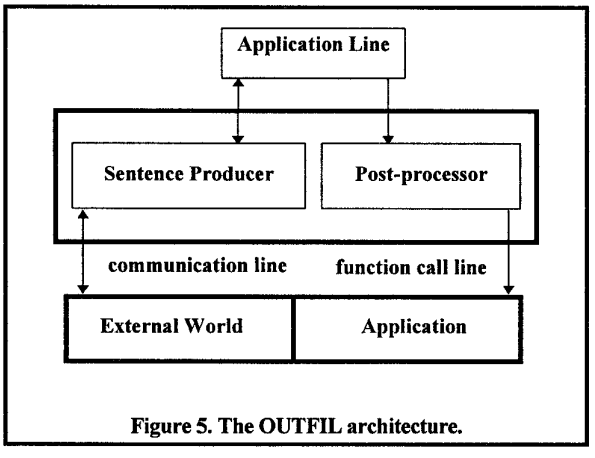

Figure 5. The OUTFIL architecture.

The **OUT**put **FIL**ter (**OUTFIL**) performs transformations necessary for the correct data management of the internal representations. It allocates pieces of information into streams of communication so that a selected application can use them. The selected application must know the communication protocol.

The output subsystem is composed of a number of distinct OUTFIL's. There is one OUTFIL for each application. The OUTFIL has two elements: a sentence producer and a post-processor (see figure 5).

The sentence producer is used for outside world communication. It serves as a monitor of data or as a reply communication channel. It is implemented as a recurrent backpropagation network. The post-processor prepares the pieces of information from the internal representation into useful data for an application. It obtains the function to be performed by the active application, and sends the corresponding attributes for that function.

3. THE NALPRONN SYSTEM AT WORK

In NALPRONN, a front-end and a back-end were constructed so that the modules perform and well in an orderly fashion. The front-end maintains the system waiting for application activation and for commands for the activated application. After an application is activated a communication channel is opened for information interchange to occur. The back-end maintains an open channel of communication with the activated application. The internal modules perform their tasks accordingly with the specific application language subset words.

Application words used for system performance and workability analysis came from a voice activated telephone application environment. One of the key aspects of the integration of voice and computing is telephony control [6]. The test implementation developed here integrates speech processing with the NLP for a telephone application universe which means that knowledge of the behavior of the integration of voice and telephony control can be obtained in this manner. The data for this task was obtained from observed experiences of interactions with telephones.

Performance analysis using the telephone application data indicates the success in the usage of the neural network technology for the construction of NLP's. Results indicate that long-term memories can be simulated by means of self-organizing feature maps and processing modules by backpropagation networks. The memories together with the processing modules, constitute the working elements of the NALPRONN system.

Upon arrival of a stream of words, the working elements perform their tasks accordingly. The input filter accepts and distributes the entering pieces of information, the lexicon performs the physical word-concept mapping of the constituents, and the output filter issues the redirected pieces of information into the specific application.

The performance analysis was achieved by injecting noise to the representations the subsystems worked with. Performance is most easily and directly measured in terms of the error rate under distinct noise conditions. Analysis of different noise levels was performed at the representation level. Each component of a

representation is affected theoretically by noise. To accomplish this, the following formula, adapted from [7], is used:

noisy component element = (1 - noise level) * component element + (noise level * random value).

The injected noise described on the previous formula represents a possible communication error. A person may say or write a word, phrase or sentence with missing characters. This means, for example that an error can occur by substituting a word for another (eg. the word all instead of the word call).

For each application lexicon subsystem, the lexical and semantic representations become stored in the weights of the units (at training time) of their respective maps in a self-organizing process. Input vectors are mapped onto locations on a specific map, that is, for each input word there is an image unit on a specific map. The maps are organized according to word or concept properties. The length of the words is the principal property of organization for the lexical map, whereas the property of organization for the semantic map is the semantic category. When analyzed independently, the maps behave very robustly in noise. The output produced without noise, is over 90% correct within 1% average error or less in the produced vector. When the independent maps are affected by noise, the statistics obtained are over 90% correct but with an increase in the average error.

The data used for training a LEX subsystem contained meanings that are expressed with different words (eg. *call freddy* means *call freddy*, but *telephone freddy* may mean *call freddy* also), but no word has more than one meaning. In any case, a word representation is transformed into a semantic representation and vice versa by means of the associations. The system was tested to observe how much noise changes the words or concepts. The lexical to semantic transformations are more succeptible to noise. At around 10% of noise level the system begins to err. This is because the encoding properties of the lexical representation vectors makes them less distributed. At around 30% of noise, the semantic to lexical transformations begin to fail. This is due to the more robust encoding scheme and distribution of the representations.

The input filter sentence analyzer and the output filter sentence producer were trained simultaneously, although independently with the same training data. The sentence analyzer learns to distribute the sentence into its constituents, forming a vector-like set of slots that conform to the distributed representation of a sentence. The sentence producer takes the constituents from the distributed representation and creates the corresponding sentence.

The input filter application memory redirects each input application to a distinct place on the application by mapping each input with a specific image unit that represents a unique application. Since a single map is used, errors are less likely to occur. The map is organized mostly according to the physical properties of the application words. The data for the implementation is a list of application titles (eg. phone, wordprocessor, spreadsheets, etc.). Analysis of different noise levels revealed that the percentage of correct words was over 90%, although the average error for each application word increases with an increase of noise level. Nevertheless, this increase in average error is not sufficient as to change the word attached to the vector representation.

4. FUTURE WORK

The NALPRONN system is in the early stages of integrating the different subsystems that constitute the NLP. Further work remains to be done in relation to its performance as a whole system. Nevertheless, present results indicate the feasibility of the construction of an NLP with neural networks. Results of its properties as a human simulator will be made clear as the analysis continues. Also, further studies remain to be done in relation to the adaptability and flexibility properties, and the feasibility of a monitor subsystem to store and monitor the data.

In relation to the lexicon, work remains on the subjects of bilinguism and generalization. Extending the human languages that the NALPRONN system understand is a matter of creating long-term memories (feature maps) that contain new language word-concept pairs. A generalized system can be achieved by developing long-term memories for new applications.

The development and analysis of a pre-parser for the INFIL subsystem is an interesting task. It entails the construction of a small scaled NALPRONN system. At the other end, the post-processor for the OUTFIL subsystem gives rise to the implementation of communication schemes for process information interchange. Channels of communication are created so that the OUTFIL post-processor effectively passes information to the application.

5. REFERENCES

[1] Gevarter, William B., "Artificial Intelligence, Expert Systems, Computer Vision, and Natural Language Processing",Noyes Publications, New Jersey, 1984.

[2] O'Shea, Tim and Eisenstadt, Marc (Editors), "Artificial Intelligence: Tools, Techniques, and Applications", Harper and Row, Publishers, New York, 1984.

[3] Miikkulainen, R.,"DISCERN: A Distributed Neural Network Model of Script Processing and Memory", Proceedings of the Third Twente Workshop of Language Technology, Twente, The Netherlands, 1992.

[4] Martínez, Wilfred M., et. al., "A Natural Language Processor for a Speech Computer Interface", Computing Research Conference- CRC'95 (Proceedings), Mayagüez, P.R., 1995.

[5] Miikkulainen, R., "A Distributed Feature Map Model of the Lexicon", Proceedings of the 12th Annual Cognitive Science Society Conference, Lawrence Earlbaum Associates, Hillsdale, N. J., 1990.

[6] Kamel, Ragui, "Voice in Computer" (Guest Editor's Introduction), Computer, Vol. 23, No. 8, IEEE Computer Society, Los Alamitos Ca., Aug. 1990.

[7] Miikkulainen, R., "DISCERN: A Distributed Artificial Neural Network Model of Script Proccessing and Memory", PhD. Thesis, Computer Science Department, University of California, Los Angeles, 1991.