World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:3, No:7, 2009

Connectionist Approach to Generic Text Summarization

Rajesh S.Prasad, U. V. Kulkarni, and Jayashree.R.Prasad

Abstract**—**As the enormous amount of on-line text grows on the World-Wide Web, the development of methods for automatically summarizing this text becomes more important. The primary goal of this research is to create an efficient tool that is able to summarize large documents automatically. We propose an Evolving connectionist System that is adaptive, incremental learning and knowledge representation system that evolves its structure and functionality. In this paper, we propose a novel approach for Part of Speech disambiguation using a recurrent neural network, a paradigm capable of dealing with sequential data. We observed that connectionist approach to text summarization has a natural way of learning grammatical structures through experience. Experimental results show that our approach achieves acceptable performance.

Keywords**—**Artificial Neural Networks (ANN); Computational Intelligence (CI); Connectionist Text Summarizer ECTS (ECTS); Evolving Connectionist systems; Evolving systems; Fuzzy systems (FS); Part of Speech (POS) disambiguation.

I. Introduction

OVER the past half a century, the problem of text summarization has been addressed from many different perspective, in various domains and using various paradigms. This paper intends to investigate Connectionist architecture for the Text Summarization system, taking into account of existing new developments in adaptive evolving systems. Evolving processes, through both individual development and evolution, inexorably led the human race to our supreme intelligence and our superior position in the animal kingdom [2] [3].

In this paper, we consider the system of an Automatic Text Summarization as Evolving system which learns incrementally through experience in the environment. This paper highlights the practical experiences, Connectionist learning environment and new ideas to promote further validations.

II. Practical Experiences

Despite the successfully developed and used methods of Computational Intelligence (CI), such as Artificial Neural networks (ANN), fuzzy systems (FS), evolutionary computation, hybrid systems, and other methods and techniques, there are a number of problems while applying these techniques to Text Summarization problem:

A. Difficulty in preselecting the system's architecture.
B. Catastrophic forgetting.
C. Excessive training time required.
D. Lack of knowledge representation facilities.

To overcome the above problems, improved and new connectionist and hybrid methods and techniques are required both in terms of learning algorithms and systems learning [2].

III. Connectionist Learning Environment

Machine learning includes methods for feature selection, model creation, model validation, and knowledge extraction.

A. Feature Selection and Evaluation

The authors propose the feature selection method based on combination of Knowledge-poor and knowledge-rich approaches [24]. The knowledge poor approach tries to rank the informativeness of a sentence by using some weighted features, such as the frequency of words, title words, cue words/phrases, the location of sentences, and the syntactic structure of sentences. The sentences with the highest scores are then regarded as the most significant and extracted.

This principle [5], entails researches using various optimization techniques; for example, Bayes rule, GAs, and SVM. Unlike this approach, modern approaches use corpus-based or discourse-analysis-based machine learning techniques to estimate feature weights [5].

Text structures or patterns frequently found in a text and a corpus are used to achieve a higher compression of text. In contrast, the knowledge rich approach tries to analyze a text using knowledge, such as the grammar or lexical databases of the target language [10].

Apparently, the knowledge- rich approach relies on a priori built-in knowledge. It is usually domain specific and more complex due to the difficulties in building an effective machine usable knowledge base. Hahn [8] points out that without background knowledge, it is difficult to obtain a high compression rate in text summarization. Naturally, many knowledge-poor text summarization researches, more suitably categorized under the hybrid approach, inevitably incorporate more or less background knowledge [24].

Among the traditional filtering methods, correlation, t-test, and signal-to-noise ratio (SNR) used for feature evaluation; we aim to use *Correlation Coefficients* that represent relationship between the features (variables) including a class feature (variable).For every feature x_i $(i=1,2,..,d_1)$ its correlation coefficients $Corr(x_i,y_i)$ with all other features, including output features y_i (j=1,2,…,d_2), are calculated. The following formula is used to calculate the Pearson correlation between two features x and y based on values for each of them:

$$Corr = \sum_{i=1} ((x_i - M_x)(y_i - M_y)) / [(n-1)\, Stdx\, Stdy]$$

Where M_x and M_y are the mean values of the two features *x* and *y*, and *Stdx* and *Stdy* are their respective standard deviations.

B. Feature Classification.

Here we talk mainly about learning in connectionist Text summarizer even though the principles of these methods and the classification scheme presented below are valid for other machine learning methods as well.

The learning algorithm is influenced by a concept introduced by Donald O.Hebb[9]. He proposed a model for unsupervised learning in which synaptic strength (weight) is increased if the source and the destination neurons become simultaneously activated. It is expressed as:

$$W_{ij}(t+1) = W_{ij}(t) + c \cdot o_i \; o_j$$

Where $W_{ij}(t)$ is the weight of the connection between the *i*th and *j*th neurons at the moment *t*, and are the output signals of neurons *i* and *j* at the same moment *t*. The weight is the adjusted at the next time moment (t +1).

In general terms, a connectionist system { S, W, P, E, F, L} that is defined by its structure *S*, its parameter set *P*, its connection weights W, its function *F*, its goal function *J*, and a learning procedure *L*, learns if the system optimizes its structure and its function *F* when observing events *z1, z2, z3,…* from a problem space *Z*. Through a learning process, the system improves its reaction to the observed events and captures useful information that may be later represented as knowledge. The goal of a learning system is defined as finding the minimum of an objective function *J*(*S*) named 'the expected risk function'. The function *J*(*S*) can be represented by a loss function *Q*(*Z*,*S*) and an unknown probability distribution Problem (*Z*).

A classification scheme is presented below. The scheme is general one, as it is not valid only for connectionist learning models but also for other learning paradigms, for example, evolutionary learning, case-based learning, analogy-based learning and reasoning. On the other hand, the scheme is not comprehensive, as it does not present all existing connectionist learning models.

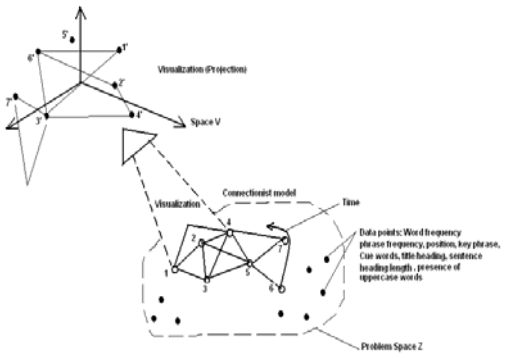

Fig. 1 (a) A computational model is built in the original data space; i.e. original problem variables are used and a network of connections is built to model their interaction; a special visualization procedure may be used to visualize a model in a different space

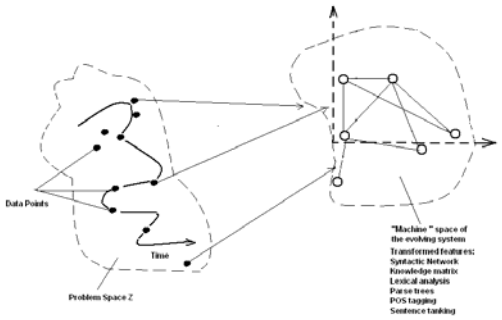

Fig. 1(b) A computational model is built in a new ('machine') space, where the original variables are transformed into a new set of variables

A connectionist system that learns from observations *z1, z2, z3,…*from a problem space *Z* can be designed to perform learning in different ways as shown in Fig. 1.

IV. System Architecture

System architecture of the proposed Evolutionary Text Summarizer (ECTS) is depicted in the following Fig. 2. This system essentially consists of a recurrent neural network that performs the task of Part of Speech (POS) disambiguation. Recurrent neural networks are fundamentally different from feedforward architectures in the sense that they not only operate on an input space but also on an internal *state space*, a trace of what already has been processed by the network [7]. Recurrent connectionist systems have feedback connections from a hidden or output layer of neurons back to inputs or to the hidden layer nodes as depicted in Fig. 2.

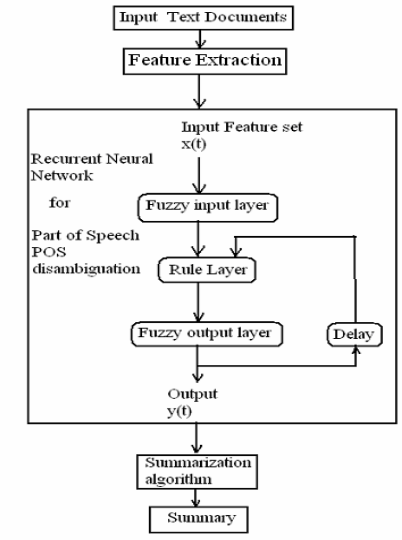

Fig. 2 System architecture of Evolutionary Connectionist Text Summarizer using Recurrent Neural Network

A. Feature Extraction

Each document is converted into a list of sentences. Each sentence is represented as a vector *[f1,f2,...,f7*], composed of 7 features.

TABLE I
FEATURE EXTRACTION FOR SENTENCES

F1	Paragraph follows title
F2	Paragraph location in document
F3	Sentence location in paragraph
F4	First sentence in paragraph
F5	Sentence length
F6	Number of thematic words in the sentence
F7	Number of title words in the sentence

The selection of features plays an important role in determining the type of sentences that will be selected as part of the summary and, therefore, would influence the performance of the neural network data. The classification function categorizes each sentence.

Each sentence is given a score. To evaluate the system, a corpus of technical documents with manual abstracts is used in the following way: for each sentence in the manual abstract, the authors manually analyzed its match with the actual document sentences and created a mapping (e.g. exact match with a sentence, matching a join of two sentences, not matchable, etc.). The auto-extracts are then evaluated against this mapping. Feature analysis revealed that a system using only the position and the cue features, along with the sentence length sentence feature, performed best [12].

B. Training

The process of training the Neural Network involves conversion of the textual representation of the sentence into a meaningful pattern. The important issues considered during this process are:

I. Choice of sentences
II. Adding morphological information
III. Manual POS disambiguation
IV. Coding sentences

Every word in a sentence is associated with a bit-vector the size of which is equal to the number of different grammatical categories (for parts of speech) in the specific language. The idea is that for a given word a 1 in a vector field corresponds to a category, admissible for that word, while a 0 corresponds to an inadmissible part of speech. We use the following correspondence between bit-vectors and grammatical categories.

TABLE II
BIT VECTOR VALUES AND CORRESPONDING GRAMMATICAL CATEGORIES

Bit vector		Category
0	0,0,0,0,0,0,0,0,0,0	blank
1	0,0,0,0,0,0,0,0,0,1	conjunction
2	0,0,0,0,0,0,0,0,1,0	preposition
4	0,0,0,0,0,0,0,1,0,0	noun
8	0,0,0,0,0,0,1,0,0,0	numeral
16	0,0,0,0,0,1,0,0,0,0	verb
32	0,0,0,0,1,0,0,0,0,0	particle
64	0,0,0,1,0,0,0,0,0,0	adverb
128	0,0,1,0,0,0,0,0,0,0	pronoun
256	0,1,0,0,0,0,0,0,0,0	adjective
512	1,0,0,0,0,0,0,0,0,0	interjection

C. POS Disambiguation

The POS disambiguation problem has been successfully solved with the rule-based and stochastic paradigms being the main tools to accomplish this. We have chosen to use a neural network, since it has become a standard tool for research in language understanding and connectionism in the recent years. The advantage of neural networks over other paradigms is that they can still be useful when the rules are not known either because the topic is too complex or because no human expert is available. If training data is available, the system may be able to learn enough about its environment just as well as (or in some cases even better than) an expert system. This approach also has the benefit of easy modification by training with an updated training set, thus eliminating programming changes and rule reconnection. The data-driven aspect of neural networks allows adjustment of changing environments and events. Another advantage of neural nets is the speed of operation after the network is trained. Natural language and the mechanisms people use for understanding it are rather complex. In real world, when reading a text one can go back to the beginning of a given sentence (or even to the words in the previous sentences) or look at the words to follow in it and based on those data deduce the correct information (including part of speech) for a given word in that sentence. It turns out that analyzing within the boundaries of a sentence is sufficient for solving the POS problem. It takes experience from a human reader to find the right grammatical category of a word [4], [5], [7]. All the more this is true for a machine analyzer. To determine the part of speech for any given word in a sentence, the succeeding and/or the preceding words and in particular their categories can help. These characteristics of a natural language suggest that in order to successfully model the problem of POS disambiguation, a paradigm capable of dealing with sequential data (i.e. data, which could be interrelated in time) should be followed [5],[15]. That is why the Simple Recurrent Network (SRN) model has been considered here as the best candidate for our purposes. Refer Fig. 3.

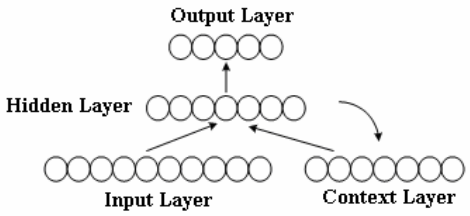

Fig. 3 Simple Recurrent Network

The connections between layers are as follows: each neuron from the input or context layer is connected to each neuron from the hidden layer; each hidden neuron is connected to each output neuron; with the recurrent connection between the hidden and context layer it is different - each neuron from one layer is connected to exactly one neuron from the other and all neurons are participating in a connection.

Simple Recurrent Networks have been developed as an extension to the popular feed-forward supervised neural

network model, the Multi-layered Perception and it does the same basic computations [21].

The activation of neurons from the context layer is a (modified) copy of the activation of hidden neurons. On the next iteration, when the functional signal propagates forward through the network, the context neurons take part in computing the activation of hidden layer. Internal activation for hidden neuron *j* is computed as:

$$v_j(n) = \sum_i w_{ji}(n) x_i(n) \quad (1)$$

where *i* varies over all neurons from both the input and context layers; $w_{ji}(n)$ denotes the weight connecting the output of neuron *i* to the input of neuron *j* on iteration *n*; and $x_i(n)$ denotes the i^{th} neuron from the input or context layer on the same iteration, respectively.

This copying process can be implemented in two different ways:

- Direct copying , where the activation of a hidden neuron is copied (without any changes) into the corresponding context neuron:

$$c_i(n) = h_i(n) \quad (2)$$

where $c_i(n)$ denotes the i^{th} context neuron at moment *n*; and $h_i(n)$ represents the i^{th} hidden neuron at moment *n*.

- Using both the current activation of hidden neurons and the context activation from the moment before:

$$c_i(n) = (1-\beta) c_i(n-1) + \beta h_i(n) \quad (3)$$

where β is a constant from the interval [0.2, 1]; c_i denotes the i^{th} context neuron at moment *n* (or *n*–1, respectively); and $h_i(n)$ represents the i^{th} hidden neuron at moment *n*.

Clearly, when previous context is used, the network will store histories of past activations in rather broad boundaries. The parameter β provides flexibility, since its value alters the weight (importance) of the participating hidden and context layer neurons in the computation of new context neuron activations. On the downside, the optimal value for this constant can not be determined in advance, because it is problem specific. Many experiments and observations are necessary in order to narrow down on its correct value.

As mentioned above, the internal activation for neurons is computed based on equation (1). The next step is to compute the functional signal. (It represents the contribution of each neuron in the activation of neurons from the layer after it.) The functional signal in the output of a neuron, which is the result of applying the activation function on the internal activation, is represented as:

$$\varphi(v_j(n)) = \frac{1}{1+e^{-v_j(n)}} \quad (4)$$

In implementation, the logistic function has been chosen as the most commonly used activation function in implementations of neural nets and in particular, in the field of natural language processing with Neural Networks, where the results have been promising.

To copy from the hidden to the context layer, the formula (3) is used. As was mentioned above, it provides for "remembering" the history of long periods and in addition to that the transition from one state of the context to another is much smoother.

To adjust the weights in the network we use the so called *delta rule*:

$$\Delta w_{ji}(n) = \eta \delta_j(n) y_i(n) \quad (5)$$

where η is the learning rate, $\delta_j(n)$ denotes the local gradient for the j^{th} neuron at moment *n*, $y_i(n)$ represents the functional signal in the output of the i^{th} neuron at moment *n*. The local gradient is defined as follows:

- If the j^{th} neuron is from the output layer, then:

$$\delta_j(n) = e_j(n) \varphi_j'(v_j(n)) \quad (6)$$

where $e_j(n)$ refers to the error signal at the output of neuron *j* for iteration *n*, and is defined

$$e_j(n) = d_j(n) - y_j(n) \quad (7)$$

where the symbol d_j denotes the desired response of the neuron *j*, and the symbol $y_j(n)$ represents the actual response.

- If the j^{th} neuron is from the hidden layer, then $\delta_j(n)$ equals the product of the associated derivative $\varphi_j'(v_j(n))$ and the weight sum of the δ's, computed over all neurons in the next layer that are connected to neuron *j*:

$$\delta_j(n) = \varphi_j'(v_j(n)) \sum_k \delta_k(n) w_{kj}(n) \quad (8)$$

For the learning process, the Backpropagation algorithm is used.

As mentioned above, the one property of a neural network, which is of primary significance in order to reach its potential, is the ability of the net to *learn* from its environment and to improve its performance through learning. A neural network learns from its environment through an iterative process of adjustments applied to its weights [9]. Ideally, the network would improve its knowledge of the environment on each iteration of the learning process. How to decide if "suitable" values for weights have been found depends on the stopping criteria. In the current implementation the stopping criteria was to reach a fixed number of iterations on the training set. After each learning iteration the network is tested for its generalization performance. The learning process continues until that performance is considered adequate in terms of the established criteria.

D. Text Summarization based on Sentence Selection

Once the network has been trained, pruned, and generalized, it can be used as a tool to filter sentences in any paragraph and determine whether each sentence should be included in the summary or not. This phase is accomplished by providing control parameters for the radius and frequency of hidden layer activation clusters to select highly ranked sentences [1] [10]. The sentence ranking is directly proportional to cluster frequency and inversely proportional to cluster radius. Only sentences that satisfy the required cluster boundry and

frequency are selected as high-ranking summary sentences.

V. Results

We used 50 different articles from internet as a test set for ECTS. The accuracy of ECTS ranged from 95% to 100% with an average accuracy of 94% when compared to the summaries of the human reader. That is, the network was able to select all sentences that were labeled as summary sentence in most of the articles. However, there was a deviation of ECTS for one or two sentences in seven text documents.

Authors further wish to point out a few important issues in POS disambiguation. These are as listed below:

A. Initialization of the Weights in the Network

For the initial weights random values from the interval [0, 1] were picked, and in a second test the interval was broadened to [−1, 1]. For the symmetric interval, the training time for the neural network (learning to recognize the correct POS) was shorter and the number of iterations was smaller by a factor of 6.

B. Size of the Training Set

Using training sets with an increased number of sentences yielded still better results – the number of epochs until the correct network behavior was reached, decreased. On the downside, the time each one epoch took in the learning process increased.

C. Size of the Hidden Layer

Increasing the number of hidden neurons caused an increase in the number of context neurons and hence the number of weights went up. Therefore a network with a large size of the hidden layer (and context layer, respectively) would require more memory and time for the training process.

D. Values of the Learning Rate and Copying Constant between the Hidden and the Context Layer

The speed of the learning process depends on both the learning rate and the parameter, controlling the copying from hidden to context layer. The smaller the learning rate, the smaller the changes to the weights in the network from one iteration to the next [22]. If on the other hand the learning rate grows too much, the network may become unstable (i.e. oscillatory). The copying parameter controls how deep in the past the network can "look" while building its current context.

E. Accuracy

This parameter refers to how many words in the context before and after a given word would change the behavior of the trained network. (These tests proved to be the most interesting ones too.) It was established that the context provided by previous words is of great importance to solving the POS disambiguation problem: sometimes, when previous context was missing, the network would not be able to determine the correct POS. When the network received a complete sentence as input, it recognized POS correctly, but when some words were later removed from that sentence, the accuracy went down.

VI. Conclusion

It is concluded that the achieved results are a promising start toward further studies. We attempt to show in this paper that connectionist approach to Text Summarization has a natural way of learning grammatical structures through experience [15], [16].We also show that Connectionist models are powerful tools for machine learning and are the best choice to model evolutionary systems.

References

[1] Amari, S., "A Theory of Adaptive Pattern Classifiers", IEEE Trans., Electron. Comput. 16:1143-1163.
[2] Amari, S. and Kasabov, N., "Brain-like Computing and Intelligent Information System, Springer Verlag, Singapore.
[3] Arbib M, "The Handbook of Brain Theory and Neural Networks", MIT Press, Cambridge, MA, (1995, 2002).
[4] Barak, A. Pearlmutter, "Gradient calculation for dynamic recurrent neural networks: a survey", IEEE Transactions on Neural Networks, 6(5), pp. 1212-1228, 1995.
[5] Elman, Jeffrey L., "Distributed representations, Simple recurrent networks, and grammatical Structure", Machine Learning, 7, pp. 195-225.
[6] Freeman, W., "Neurodynamics", Springer, London, 2000.
[7] Garen Arevian, "Recurrent Neural Networks for Robust Real-World Text Classification", IEEE/WIC/ACM International Conference on Web Intelligence, 2007.
[8] Han, K.-H. And Kim, J.-H. "Quantum-inspired evolutionary algorithm for a class of Combinational optimization", IEEE Transactions. Evol. Comput., Vol. 6, No. 6, pp 580-593.
[9] Hebb, D., "The Organization of Behavior", Willey, New York.Haykin, S., "Neural Networks: A Comprehensive Foundation", 2nd edition, Prentice Hall, 1998.
[10] Hongyan Jing, "Sentence Reduction for Automatic Text Summarization", Department of Computer Science Columbia University New York, NY 10027, USA
[11] J. L. Elman, "Distributed representations, simple recurrent networks, and grammatical structure," Machine Learning, vol. 7, pp. 195–225, 1991.
[12] Khosrow Kaikhah, "Text Summarization Using Neural Networks", Faculty Publications-Comp. Science Texas State University,2004.
[13] Lei Yu1, Jia Ma1, Fuji Ren1, 2, Shingo Kuroiwa1, "Automatic Text Summarization Based on Lexical Chains and Structural Features", ACIS, Journal of Artificial Intelligence Research (1997), pp 574-578.
[14] Medsker, Larry R., "Hybrid Neural Network and Expert Systems", Kluwer Academic Publishers.
[15] Mayberry,M. R. and Miikkulainen R., "Lexical Disambiguation on Distributed Representations of Context Frequency", In Proceedings of the 16th Annual Conference of the Cognitive Science Society (COGSCI-94, Atlanta, GA), pp. 601-606.
[16] Nikola Kasabov, "Evolving Connectionist Systems", Springer, second edition, 2007.
[17] Rosch, E. and Lloyd, "Cognition and Categorization", Lawrence Erlbaum, Hillssale, NJ, 1978.
[18] Skapura, David M., "Building Neural Networks", ACM Press, New York, pp.154-161.
[19] Smith, E.E. and Medin, D.L. "Categories and Concepts", Harward University Press, Cambridge, MA, 1981.
[20] Stoianov, I. P., Nerbonne, J. and Bouma, H., "Modelling the Phonotactic Structure of Natural Language Words with Simple Recurrent Networks", Computational Linguistics in Netherlands 1997.
[21] Tebelskis, J., "Speech Recognition using Neural Networks", CMU-CS-95-142, Carnegie Mellon University, Pittsburgh, PA, May 1995.
[22] Wermter, S., Weber V., "SCREEN: Learning a Flat Syntactic and Semantic Spoken Language Analysis Using Artificial Neural Network", pp. 35-85.
[23] William M. Ramsey, Stephen P. Stich, David E. Rumelhart, "Philosophy and connectionist theory", Lawrence Erlbaum Associates, 1991,ISBN 0805805923, 9780805805925
[24] Yau-Hwang Kuo and Hsun-Hui Huang, "Automatic Extraction of Key Sentences via Word Sense Identification for Chinese Text Summarization", Journal of Advanced Computational Intelligence Vol.11 No.4, 2007 and Intelligent Informatics.
[25] Taylor, J.G., "The Race for Consciousness", MIT Press, Cambridge, MA., 1999.