

An Auto-Associative Neural Network for Information Retrieval

Guy Desjardins, Robert Proulx, *Member, IEEE*, and Robert Godin, *Member, IEEE*

Abstract – Neural network is an important paradigm that has received little attention from the community of researchers in information retrieval, especially the auto-associative neural networks. These networks are capable of discovering patterns of terms among documents. We propose an auto-associative neural network to model the classification and to perform the matching task. The unique layer network is trained with the documents of the collection and then used to recall the most relevant documents to specific queries.

Our model has been tested on a TREC sub-collection. The results are compared against the vector space model [1]. The experiment shows higher level of global precision and recall. The recall-precision curves show an important improvement on the precisions for the low levels of recall, which indicates a faster retrieval of the first relevant documents. This strength of the auto-associative neural network makes it an attractive model in information retrieval for general collections.

I. INTRODUCTION

Information retrieval is concerned with the classification processes and the selective recovery of information. In the literature, few researchers have tackled the problems of the information retrieval domain with an artificial neural network approach. Among those who did, the majority focused on the classification problem [2]. Many researchers have contributed to enhance the existing information retrieval solutions by automating processes that enrich the representation. We have seen automatic thesaurus construction [3, 4, 5], desambiguation processes [6], query reformulation processes [7, 8] and other helping processes [9]. Nowadays, these added processes are turning towards the construction of much ontology and the semantic Web. The present research is concerned only with improvements at the core of the retrieval process.

The information retrieval process can be divided into three main tasks: indexing the collection, translating the representation of the documents and the queries and performing the match between the two. (Figure 1) In the first task, the documents are read by a parser, which identifies the individual tokens of the text (the words), eliminates the tool words from a stoplist, extracts the root of the words (stemming) and calculates some statistics on the corpus.

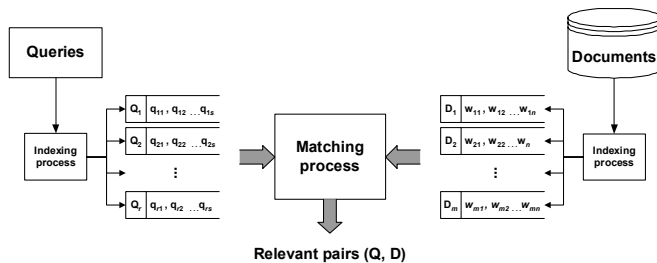


Figure 1 - Information retrieval process

The basic elements used by a retrieval system are the remaining root-words called the *terms* of the corpus. Before performing the second task, one has to decide on the representation to be used in the retrieval task. The most popular representation today is the vector representation where each document is represented by a vector of the terms included in the document. The elements of the vectors can be of different metrics: binary, frequencies, weights, entropies, etc. (Figure 2) According to the metric selected, the system translates the documents of the collection into numerical vectors, one vector for each document. If we consider the whole corpus of terms, then each document is represented by a vector in a space of dimensions equal to the number of distinct terms.

The matching task consists of calculating a similarity degree between the documents of the collection and a submitted query. If the query uses the same vector space representation, then it can be translated into a vector of terms and we can calculate the distance between the two vectors in the multi-dimensional space. One of the most widely used similarity function is the calculation of the cosine of the angle between a query vector and the document vectors.

$$sim(q, d_j) = \frac{\sum_{i=1}^n w_{ij} \times w_{iq}}{\sqrt{\sum_{i=1}^n w_{ij}^2} \times \sqrt{\sum_{i=1}^n w_{iq}^2}} \quad (1)$$

where w_{ij} is the weight of term i in the document d_j ,
 w_{iq} is the weight of term i in the query q .

With this calculation, the documents can be sorted in decreasing order of their similarity to the query. Then the retrieval system returns either the first m documents or all documents where the similarity is higher than a predetermined threshold.

Most of the recent retrieval models build upon the vector space model initiated by Salton back in 1971 [1] and attempt to achieve a more accurate classification by incorporating a form of co-occurrences into the vector representation. The main objective of these models is to select useful co-occurrences of terms in the collection of documents and use this extra information to either augment the query information or enhance the classification of the documents.

Since the beginning of the 90's, the artificial neural network paradigm has received more attention from researchers finding new ways to apply and implement it. The information retrieval domain has seen a few but interesting applications of neural networks. The researchers, who have modeled a neural network for information retrieval, have

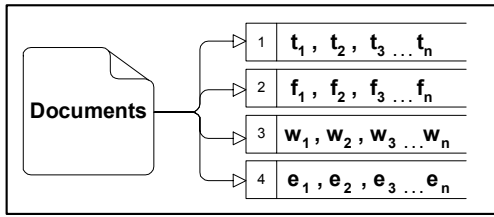


Figure 2 - Vector Space Model representation

mainly adopted a multi-layer perceptron with a delta rule and the error backpropagation learning process [8, 10, 11, 12], a self-organizing map with a Kohonen rule and a competition-based learning process [2, 13, 14, 15, 16, 17, 18] or a bi-directional associative memory [5].

In this paper, we propose an auto-associative neural network to model both the classification and the retrieval problem, using a simple Hebbian-anti-Hebbian association rule and a regular hyperbolic tangent activation function. In our design, the correlated terms are selected by means of the convergence of the auto-associative process and added to the internal representation of the documents. Then the queries and the documents are translated into the new augmented representation in order to achieve the matching task of the retrieval process.

The remaining sections are as follows. The variety of neural networks approaches applied to the information retrieval problems is reviewed in Section II. Section III describes the auto-associative neural network model and its implementation. Section IV reports on the first results obtained as compared with the vector space model (VSM). The last section concludes with a discussion on the drawbacks and proposes some direction for future work.

II. ARTIFICIAL NEURAL NETWORKS IN INFORMATION RETRIEVAL

Artificial neural networks have been used with success in many domains where the problems call for learning and classification capabilities in a non-linear space. Finding the discriminating terms and sets of correlated terms in a collection of documents is a complex non-linear problem to solve. A few researchers have attempted to use neural networks for information retrieval.

The BAM neural network (*Bi-directional Associative Memory*) has been used to model the bi-directional relation between the terms and the documents [5], i.e. each document can be represented by the set of terms it contains and each term can be represented by the set of documents it indexes. After manually fixing the weights of the connections or the activation level of the neurons, the network is used to perform the query-document matching task. These experiments reproduced the $tf \times idf$ calculation (see section IV) and the cosine similarity measure commonly used in information retrieval. No learning rule is implemented within these neural networks. The design is fixed for a specific collection and cannot be generalized over other collections.

Other researchers have designed MLP neural networks (*Multi-Layer Perceptron*) trained with query-document pairs and supervised by a relevant/non-relevant output goal [10, 12]. The weights of the hidden layer connections are updated using an error backpropagation learning rule. In [11], Farkas used a MLP with a recurrent layer, which re-inputs the hidden layer's internal state into the next training sequence, along with the new input vector. This recurrent MLP design was able to memorize sequential patterns within the flow of documents during training. All MLP neural networks are supervised neural networks. During the training phase, the error backpropagation learning process needs to be taught the desired output in order to calculate the error. It then propagates the error backward into the network to adjust the synaptic weights. This is how it learns to produce the desired output. These networks are efficient at learning to reproduce the decision process of an expert. They can help to classify the documents of a collection into categories. The drawback is that the categories must be manually established beforehand. Since an expert would assign a set of categories in accordance with the specificity of the collection, the MLP network will specialize itself onto that specific collection and, therefore, cannot generalize over other collections. MLP neural networks cannot discover the right classes by themselves.

At the present time, the self-organizing map paradigm is getting the attention of the researchers of the information retrieval community. Self-organizing maps are unsupervised neural networks primarily used to classify the documents relative to their similarities and to obtain a visual map of the collection [2, 13, 14, 17, 18]. At the input layer, each node represents one term of the corpus. The output layer is an arbitrary grid of nodes that represents the map of the classes. The network is trained with the term vectors representing the documents of the collection. The resulting map is intended to guide the users through an interactive query reformulation process. The map shows similar documents within each output node and the similarity between the documents of different classes decreases as the distance between the nodes increases on the map. This spatial proximity property is obtained with the Kohonen learning rule, which adjusts the weights of the connections between the input and the output layers. During the training phase, each input vector activates the output nodes to different magnitudes. Output nodes compete with one another in order to classify the input vector. The winning node will have its afferent connections updated. The neighbouring nodes connections are also updated in a decreasing proportion relative to their distance to the winning node on the output map.

One drawback of this design is that the network classifies documents instead of terms. If some terms are to be added to the query, one has to manually translate the map into classes of terms prior to undergoing a query reformulation process. In [15], we proposed a model of a self-organizing map to overcome this problem. The SOM network is fed by the terms of the corpus where each term is represented by a vector of documents. The final map represents a classification of the terms instead of the documents. The

obtained classes of terms can then be directly interpreted as concepts embedded into documents and each document can be automatically translated into a vector of concepts. The drawback of this approach is that each term belonged to only one concept whereas in real life, concepts may overlap each other.

Auto-associative neural networks can overcome this rather rigid classification by capturing and superposing the co-occurrences information into their internal state. Auto-associative neural networks are unpopular because the fast convergence of the solutions to a few localized sub-spaces is difficult to control and because the number of neurons on the unique layer limits the memory storage capacity. However, auto-associative neural networks exhibit strong memory recall with a good tolerance to noise [19, 20, 21]. They nowadays probably represent the most psychologically plausible memory paradigm among the neural network models. They have proven to be very efficient in learning and recalling visual pictures and signatures [22].

Chen has designed an auto-associative unsupervised neural network that captures such term co-occurrences [7]. His goal was to feed a query reformulation process. In the design, the nodes of the network represent the terms of the corpus. The network is trained with the term vectors representing the documents of the collection. The associative connections between terms are updated with a standard Hebbian learning rule. At recall time, the terms of the queries are presented to the input of the network, which responds with patterns of correlated terms. The missing co-occurrences are added to the query and the enhanced query is cyclically presented to the network until no new correlated term is added. The objective of this network was to enrich the representation of the query before processing the query-document matching task. This design does not take full advantage of the co-occurrences patterns stored into the synaptic matrix of the network. The synaptic matrix does not contain only the presence or absence of co-occurrences but also the strength of the correlated terms. When used in a query reformulation process, the correlated terms are added with equal strengths.

Next section describes how we built upon this work to produce our model and how we maximized the benefits of the synaptic matrix.

III. THE AUTO-ASSOCIATIVE NEURAL NETWORK MODEL

Auto-associative neural networks are fully connected recurrent networks where all nodes are inter-connected except to themselves. (Figure 3) The network is fed with a training vector $X_i(0)$ where each element corresponds to one node of the unique layer. The activation is spread through the connections and generates a new output vector $X_i(1)$. Then the synaptic weights W are adjusted using a Hebbian rule. The new output vector is reprocessed through the network and generates a second new output vector $X_i(2)$. This cycle continues until two consecutive output vectors exhibit the same values. At that time, the output vector $X_i(t)$ reaches a stable state and the system proceeds with the next training vector. After all vectors are processed at least once,

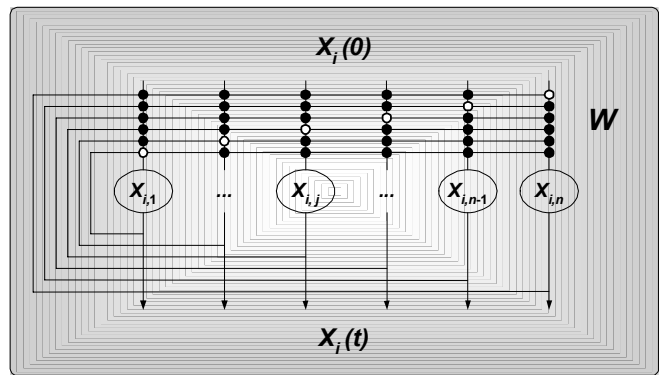


Figure 3 - Auto-associative neural network architecture

(each training vector can be processed many times, depending on the nature of the problem) the training of the network is completed and the learned patterns are recorded within the synaptic matrix W . From there on, any new vector can be processed through the network, which will respond with the nearest stable state vector.

In information retrieval, the problem is to classify the documents of a collection and then recall the most relevant ones to a query. From the indexing process, we obtained the vector representation of the documents. The auto-associative neural network is trained with the document vectors. The network memorizes the documents by discovering and learning their common patterns. At recall time, a query vector is submitted and the network responds with the nearest document vector's stable state.

This 'query' stable state is used in a similarity calculation with each 'document' stable state to order the document's relevancy to the query.

A standard hyperbolic tangent was used as the spreading activation function.

$$\mu_i(t+1) = \frac{1}{2} \left[1 + \tanh \left(\left(\left(\sum_{j=1, j \neq i}^N w_{ij} \mu_j(t) \right) - \theta \right) / \lambda \right) \right] \quad (2)$$

where $\mu_i(t)$ is the activation level of node i at iteration t ,
 w_{ij} is the connection weight from node i to node j ,
 λ is the slope of the function,
 θ is the activation threshold.

When processing a document vector, the output state stabilizes when $|\mu_i(t+1) - \mu_i(t)| < \epsilon, \forall i$. λ and θ are two free parameters to be determined empirically. During the training phase, the weights of the synaptic matrix W are updated according to the following Hebbian learning rule.

$$w_{ij}(t+1) = w_{ij}(t) + \alpha [x_i(t) x_j(t)] - \beta [x_i(t) x_j(t)] \quad (3)$$

where $w_{ij}(t)$ is the connection weight from node i to node j at iteration t ,
 $x_i(t)$ is the output of node i at iteration t ,
 α is the learning rate parameter,
 β is a forgetting rate parameter.

IV. EXPERIMENT SET UP AND RESULTS

The first part of this rule is a regular Hebbian learning rule where the parameter α controls the learning rate. The rule states that the weights connecting the nodes i and j will get updated by a portion ($0 < \alpha \leq 1$) of the 'correlation' between the two nodes ($x_i \cdot x_j$). In other words, the more frequently the two nodes are simultaneously activated, the stronger their connection will be. This mutual reinforcement gives the network the auto-associative memory property. One of the frequent problems with auto-associative neural networks is that the internal patterns converge too fast toward few attractor vectors. This situation leads the network to correctly cover only a small portion of the multi-dimensional space. This problem appears even more frequently in high dimensional spaces with sparse data, which is precisely the case in information retrieval. Anti-Hebbian learning rules have been developed to overcome this problem [23]. Bégin and Proulx have demonstrated that an anti-Hebbian factor put equal to half of the learning rate ($\beta = \alpha/2$) and applied once each other iteration helps controlling the overall convergence and produces a better coverage of the space [22]. We have added this anti-Hebbian factor to the learning rule. (See the β factor in equation 3) The overall rule is a two stage Hebbian-anti-Hebbian learning rule.

Once the training is done, the vectors are processed for the last time through the network, without applying the learning rule, in order to record their final stable output. The queries are then processed in the same manner. At the end, the system gathers all queries and documents output vectors. These vectors are the new representation, which includes the original single term representation augmented with the correlation information.

One more modification is made to this design. The actual model was not running in a reasonable timeframe because of the huge number of calculations involved with the connection matrix. As it is described in the next section, our test data includes a corpus of 26 000 terms, which defines the neural network with the same number of neurons. Such a network would perform near $2,7 \times 10^9$ calculations of the activation function for each input vector. In order to alleviate these calculations, we deactivated the calculation for all inactive neurons within each document vector. The modification brought the number of calculations down to an average of 400 per document vector. This dynamic selection of connexions is applied during both the training and the recall phases. We believe the modification will not impact the output results significantly because the local inactive neurons in a specific document do not interact with each other; hence they should not influence the global patterns being recorded into the synaptic matrix. Eventually, this assumption would need to be tested thoroughly. It seems reasonable for a first exploration of the auto-associative neural network in the domain of information retrieval.

Next we describe the data source, the evaluation method and the results of the experiment.

The test collection of documents was selected from the TREC FT943 collection, which includes some articles from the Financial Times Limited for the third quarter of 1994. This magazine covers a variety of financial news from England and international events. From the collection, we have selected 2 000 documents and 7 queries. The selection was performed strictly based on the statistical distribution of the relevant documents over the queries. This selection is justified by the need to reduce the number of documents to a manageable size while retaining a minimum density of the relevant documents to the queries. The selection criterion was to find a sub-collection of 2 000 documents for which a maximum of 10 queries would totalize a minimum of 20 relevant documents. This process selected the TREC topic numbers 261, 269, 331, 352, 404, 435 and 450. The indexing process retained 25 838 terms as the final corpus.

We used a weight representation for the documents and the queries, where the weights are defined according to the $tf \times idf$ scheme [1].

$$w_{ij} = tf_{ij} \times idf_i = tf_{ij} \times \log\left(\frac{N}{n_i}\right) \quad (4)$$

where w_{ij} is the weight of term i for the document j ,
 tf_{ij} is the frequency of term i in document j ,
 idf_i is the inverse document frequency of term i ,
 n_i is the number of documents that include term i ,
 N is the total number of documents.

The idf factor captures the discriminating power of the term. If a term appears in too many documents, then it is not useful for discriminating those documents. This is reflected in the logarithm function. As the number of documents including term i (n_i) approaches the total number of documents in the collection (N), the idf factor approaches zero. The final weight of such term will approach zero, no matter how frequent it is in each document. On the contrary, if only one document includes a term, its idf factor will retain a maximum discriminating power. The $tf \times idf$ weighting scheme states that a term is as important to represent a document, as it is frequent within the document and rare among all documents of the collection.

We used four metrics to assess the performance of the retrieval. All of them are based on the measurement of the basic recall and precision metrics.

$$recall = |\text{Retrieved} \cap \text{Relevant}| / |\text{Relevant}| \quad (5)$$

$$precision = |\text{Retrieved} \cap \text{Relevant}| / |\text{Retrieved}| \quad (6)$$

where $|\text{Relevant}|$ is the total number of relevant documents for a specific query,

$|\text{Retrieved}|$ is the total number of documents retrieved by the system.

The recall ratio measures the completeness of the retrieval whereas the precision ratio measures the effectiveness of the

retrieval. An ideal system would retrieve all relevant documents and only the relevant documents. Since these two measures are conversely related, it is accustomed to show them both on a precision-recall graph. In order to obtain these curves, the precision of the individual query retrievals are interpolated at eleven standard levels of recall (0, 10%, 20%, ..., 90% and 100%) and then averaged over all queries.

The average precision per recall level is the first metric of the four we have used. The three remaining metrics are:

$$M\text{-precision} = (1/|\text{Relevant}|) \sum P_j, \forall j \rightarrow D_j \quad (7)$$

where P_j is the precision measure at each time the system retrieves a relevant document D_j .

$$R\text{-precision} = P(r = |\text{Relevant}|) \quad (8)$$

where P is the precision measure at a recall level equals to the number of relevant documents for the specific query.

$$F(j) = \frac{2}{\frac{1}{r(j)} + \frac{1}{p(j)}} \quad (9)$$

where $p(j)$ is the precision measure at the retrieved document j ,
 $r(j)$ is the recall measure at the retrieved document j .

Equation (7) is the *average precision at seen relevant documents* and depicts the capacity of a system to retrieve the relevant documents quickly. Equation (8) depicts the precision at the 'ideal recall level', i.e. when the number of documents retrieved equals the number of relevant documents. An ideal system retrieving all relevant documents first would score 1 on *R-precision*. Equation (9) is a composite metric called the *harmonic mean*, which combines the recall and the precision into a single metric. This measure can be used to find the best possible compromise between recall and precision, which would be the level j where the function returns the maximum value. The maximum harmonic mean can also be used as a single point of comparison between two models. In reporting the retrieval results, we have computed the maximum harmonic mean. These three metrics have been computed over the individual query results and then averaged over all queries. In contrast to the average precision per standard level of recall (first metric), the last three metrics are averaged over the queries weighted by the number of relevant documents for each query.

Figure 4 shows the precision-recall curves for the auto-associative neural network model and the vector space model. The VSM curve shows a regularly decreasing precision over all levels of recall, going from 8,53% down to 3,68%. In contrast, the A-A NN model shows much higher precision at low levels of recall (< 30%), but lower precisions at higher levels of recall (> 40%). In general, a slowly decreasing curve offers a better retrieval performance on collections with specialized vocabulary, where the focus is on the high levels of recall ($\geq 80\%$). Conversely, on

general collections, a rapidly decreasing curve put the focus on the low levels of recall ($\leq 30\%$). It is important to improve the precision at low levels of recall because non-specialist users will only pay attention to the first documents returned by a search engine. In this regard, it seems that the neural network model would perform better on general collections.

Table I shows the values obtained with the other three metrics. All these global average precision measurements agree that the A-A NN model has superior retrieval results. From the *M-precision*, it becomes evident that the A-A NN model retrieved the relevant documents faster, on average, than the VSM model. The *R-precision* also supports this finding. The zero value obtained for the VSM model indicates that it was unable to retrieve any relevant document within the first k returned documents, where k equals the total number of relevant documents for each query. For the A-A NN model, 20% of the relevant documents were retrieved within the first k returned documents. The *maximum harmonic mean* also shows a higher performance for the A-A NN model, although the difference is less noticeable because the metric is less sensitive to fast retrieval.

Table II gives an insight of the fast retrieval. It shows the rank of retrieval for the relevant documents to each individual query. Only queries with different ranking between the two models are shown and, for these queries, only the different ranking documents are shown. For example, the first two relevant documents to query number 352 were retrieved first and fourth by the A-A NN model, whereas they were retrieved as the 14th and 18th documents by the VSM model. The best rank for each relevant document is indicated in bold. The figures indicate that the A-A NN model retrieved some of the leading relevant documents faster than the VSM model, but slower afterwards. The overall effect favours the A-A NN model, as indicated by the weighted average precisions.

This behaviour of the auto-associative neural network can be explained by the fast convergence toward the attractor vectors. The strong pattern recognition for only a few documents may be a result of the fast convergence. However, there is a weaker association to the other documents.

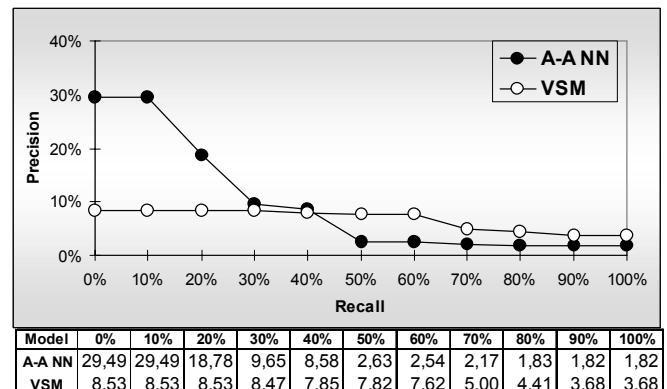


Figure 4 - Precision-recall curves

TABLE I
WEIGHTED AVERAGE PRECISIONS

Model	M-Precision	R-Precision	Maximum Harmonic
A-A NN	16,29	20,00	0,2240
VSM	7,52	0,00	0,1855

TABLE II
RETRIEVAL RANK OF THE RELEVANT DOCUMENTS

Query #	A-A NN
261	22
331	1 - 8 - 27 - 129 - 138 - 221 - 230
352	1 - 4 - 36 - 83 - 87
435	61
	VSM
261	5
331	14 - 18 - 28 - 63 - 82 - 135 - 146
352	6 - 10 - 11 - 45 - 129
435	93

Despite our efforts to better spread the coverage of the associative memory, the network converged too fast toward specific small sub-spaces. We may have to review some parameters of the model, especially the Hebbian-anti-Hebbian factors and their relationship.

Another approach would be to recursively cover the small portions of the multi-dimensional space with distinct auto-associative neural networks, rather than trying to cover it all with a unique network.

This approach could be more appropriate for a general collection. Typically, such collections contain several unrelated sub-domains, each including documents that are locally correlated with different magnitudes. A brigade of neural networks may better capture the correlation patterns within each sub-domain. Moreover, a multi-network approach would certainly enhance the scalability of the system. This approach would require the definition of some criteria to set the borders between the sub-domains. An alternative approach would be to classify the documents of the collection using a hierarchical self-organizing map and then, to implement smaller auto-associative neural networks to the output map, one network for each node. In this design, the competitive neural network would serve as the criteria to set the borders between the classes and the brigade of auto-associative neural networks would carry the retrieval task within each sub-domain.

V. CONCLUSION AND FUTURE WORK

In this experiment, we have used a new approach to model both the classification and the retrieval task of the information retrieval problem. We have developed and implemented an auto-association neural network to capture the co-occurrences patterns in a collection of documents. These patterns then served as a new representation for the documents. The trained network was further used to extract the patterns within the submitted queries. The retrieval task

was performed based on the similarities between the document patterns and the query patterns.

The experiment revealed a specific strength of the auto-associative neural network model to retrieve the first relevant documents fast. With this approach, the leading relevant documents were retrieved faster, on average, than with the vector space model. However, most of the trailing relevant documents have been ranked worst. Overall, the auto-associative neural network model performed better with higher average retrieval precisions.

The experiment was conducted on a small collection extracted from the TREC collections. Both the results and the scalability of the model need to be confirmed on a larger scale collection.

Since the neural network converged quickly during the training phase, we believe that the patterns extracted from the documents and stored into the synaptic matrix covered only a small portion of the document space. Therefore, future research will aim to increase this coverage in order to retrieve most of the relevant documents faster. This can take two directions. The first direction is to modify the learning rule to slow down the convergence of the process. The second direction is to adopt a recursive approach to break down the document space into smaller sub-domains. Then each individual sub-domain can be learned and memorized by a smaller auto-associative neural network. This approach would be best indicated for retrieval on general collections.

REFERENCES

- [1] Salton, G. "The SMART Retrieval System – Experiments in Automatic Document Processing", Prentice Hall inc, 1971.
- [2] Hung, C. and Wernter, S. "A Dynamic Adaptive Self-Organising Hybrid Model for Text Clustering", Proceedings of the 3rd IEEE International Conference on Data Mining, pp. 75-82, 2003.
- [3] Ding, Y. and Engels, R. "IR and AI: Using Co-occurrence Theory to Generate Lightweight Ontologies", 12th International Conference on Database and Expert Systems Applications, volume 2, pp. 1676-1685, 2001.
- [4] Schütze, H., and Pedersen, J.O. "A Co-occurrence-based Thesaurus and Two Applications to Information Retrieval", in 4th Proceedings of the RIAO Intelligent Multimedia Information Retrieval Systems and Management, volume 1, pp. 266-274, 1994.
- [5] Syu, I. and Lang, S.D. "A Competition-based Connectionist Model for Information Retrieval Using a Merged Thesaurus", Proceedings of the 3rd international conference on Information and knowledge management, ACM, pp. 164-170, 1994.
- [6] Towell, G. and Voorhees, E.M. "Disambiguating Highly Ambiguous Words", Computational Linguistics Volume 24, Issue 1, Special issue on word sense disambiguation, pp. 125-145, 1998.
- [7] Chen, H. et Kim, J. "GANNET: Information Retrieval Using Genetic Algorithms and Neural Nets", Journal of Management Information Systems, volume 11, no. 3, pp. 7-42, 1995.
- [8] Lingras, P. and Yao, Y.Y. "Neural networks as queries for linear and non-linear retrieval models", Proceedings of the 5th International Conference of the Decision Sciences Institute, Volume II, pp. 574-576. 1999.
- [9] Chung, Y.M, Pottenger, W.M, Schatz, B.R. "Automatic Subject Indexing Using an Automatic Associative Neural Network", Proceedings of the 3rd ACM International Conference on Digital Libraries (DL'98), pp. 59-68. ACM Press, 1998.

- [10] Chandren-Miniyandi, R. “*Neural Network: an Exploration in Document Retrieval System*”, TENCON Proceedings, volume 1, pp. 156–160, 2000.
- [11] Farkas, J. “*Document Classification and Recurrent Neural Networks*”, IBM Centre for Advanced Studies Conference Proceedings of the 1995 conference of the Centre for Advanced Studies on Collaborative research, 1995.
- [12] Mandl, T. “*Tolerant and Adaptive Information Retrieval with Neural Networks*”, Global Dialogue, Science and Technology – Thinking the Future at EXPO 2000 Hannover, 2000.
- [13] Ahmad, K., Vrusias, B. and Ledford, A. “*Choosing Feature Sets for Training and Testing Self-Organising Maps: A Case Study. Neural Computing & Applications*”, vol. 10, pp. 56-66, 2001.
- [14] Amarasiri, R., Alahakoon, D., Smith, K.A. “*HDGSOM: A Modified Growing Self-Organizing Map for High Dimensional Data Clustering*”, 4th International Conference on Hybrid Intelligent Systems (HIS'04), pp. 216-221, 2004.
- [15] Desjardins, G, Godin, R. and Proulx, R. “*A Self-Organizing Map for Concept Classification in Information Retrieval*”, Proceedings of the International Joint Conference on Neural Networks, IEEE, pp. 1570-1574, 2005.
- [16] Kohonen, T. “*Exploration of Very Large Databases by Self-Organizing Maps*”, Proceedings of the IEEE International Conference on Neural Networks, volume 1, pp. 1-6, 1997.
- [17] Lin, X., Soergel, D. and Marchionini, G. “*A Self-Organizing Semantic Map for Information Retrieval*”, Proceedings of the 14th International ACM/SIGIR Conference on Research and Development in Information Retrieval, pp. 262-269, ACM Press, 1991.
- [18] Rauber, A., Merkl, D. and Dittenbach, M., “*The Growing Hierarchical Self-Organizing Maps: Exploratory Analysis of Highdimensional Data*”, IEEE Transactions on Neural Networks, volume 13, no. 6, pp.1331-1341, 2002.
- [19] Hopfield, J. “*Neural networks and physical systems with emergent collective computational abilities*,” Proceedings of the National Academy of Sciences USA, vol. 79, pp. 2554–2558, 1982.
- [20] Gardner, E. “*Structure of metastable states in the Hopfield model*”, Journal of Physics A: Mathematical and General, vol. 19, no. 16, pp. 1047–1052, 1986.
- [21] Molter, C, Salihoglu, U. and Bersini, H. “*Introduction of a Hebbian unsupervised learning algorithm to boost the encoding capacity of Hopfield networks*”, Proceedings of the International Joint Conference on Neural Networks, IEEE, pp. 1552-1557, 2005.
- [22] Bégin, J. and Proulx, R. “*Categorization in Unsupervised Neural Networks: The Eidos Model*”, IEEE Transactions on Neural Networks, Volume 7, no. 1, 1996.
- [23] Meyer-Base, A. Yunmei C, McCullough, S. “*Hebbian and anti-Hebbian learning for independent component analysis*”, Proceedings of the International Joint Conference on Neural Networks, IEEE, Volume 2, pp. 920-925, 2001.